

MiniCAD 实验报告

功能介绍

完成了绝大多数要求的功能，包括：

- 能用鼠标绘制直线、矩形、椭圆、多边形等等；
- 能选中并移动、删除图形；
- 能保存和加载图形

基本数据结构

项目依赖于两种数据结构，分别是 `./src/shapes/` 下的 `Shape` 类和 `./src/states/` 下的 `State`.

- `Shape` 类。作为抽象类，说明了其子类所需要实现的接口。包括：

```
public abstract void render(Graphics2D canvas);
public abstract void setBegin(int x, int y);
public abstract void setEnd(int x, int y);
public abstract void move(int x, int y);
public abstract void resize(int delta);
public abstract void setStokeColor(Color color);
public abstract void setStokeSize(int delta);
public abstract void selected();
public abstract void removeSelected();
```


这些接口的目的是为了实现对图形的更改、移动、创建等操作。其中 `render` 函数作为绘制函数，由图形所在 `Swing` 组件调用。实现了 `Shape` 抽象类的子类包括：`Ellipse`, `FilledEllipse`, `FilledRect`, `Line`, `Polygon`, `Polyline`, `Rect`, `Text` 等。

- `State` 类。同样作为抽象类，说明了在其子类需要实现的接口：

```
public abstract State input(MouseEvent event, Mode mode, CanvasAction
action, Shape shape);
```

这个类主要用于实现程序需要用到的有限状态机的各种状态。其唯一要求的实现的函数 `input` 将作为有限状态机的输入。所有的状态包括：`Move`, `Ready`, `Selected`, `Wait`, `Working`。

基本架构

项目采用 MVC (Model-View-Control) 的架构。各个组分的关系如下图所示： `frame` 下面详细介绍三个组件的情况：

Model

Model 存储当前画布所有的图形数据，其本身继承自`ArrayList<Shape>`。每当一个图形在画布上被绘制完毕，它就会被存储到Model中。当 Control对Model中的数据进行修改后，View马上根据Model中的数据渲染画布。

View

View 中实现了程序的GUI界面。View主要包括三个组件：Canvas（画布）、ControlPanel(控制面板)和MenuBar。其中MenuBar置于界面顶部，提供Save和Load选项；ControlPanel位于界面右侧，用于选择功能；Canvas占主要空间,是用户主要进行交互的空间。三个组件包含于一个JFrame对象，其采取`BorderLayout`，Canvas在`CENTER`，ControlPanel在`EAST`。而其中ControlPanel又采用`GridLayout`。在Canvas和ControlPanel中都需要设置监听器。其中Canvas的监听器用于监听鼠标的`Press, Drag, Release, Move`等事件，接着把事件消息传递到Control，由其采取相应的事件处理方式（调整有限状态机中的信息）。而ControlPanel监听功能键，事件发生后同样传递到Control，改变Control内部的属性值，包括`mode`（当前模式）等。

Control

Control中实现了一个有限状态机，由其来决定对Model中数据的修改。有限状态机的草图形式如下：  DFA