CS5001: Object Oriented Modelling Design & Programming Practical 5

Student IDs: 150019538

November 20, 2015

1 Introduction

The assignment was to create a simple vector drawing program. This solution satisfies all the basic criteria and enhancements specified in the assignment. Some additional enhancements are:

- List view of all shapes created where shapes are selectable.
- Possibility to change the order of the shapes.
- Support for stroke width and color.
- Undo/redo support for shape colors, order changes, movements, resize and rotation, stroke width etc.
- SVG file import (only fully supports files created in this drawing application).
- SVG file export. File can be opened in any SVG-compatible application such as Firefox or Chrome.

2 How the System Works

Important aspects to know about how the system works:

- To start drawing shapes, one needs to click "File" and "New..." to create the project canvas.
- To create parallelograms, click on the "Parallelogram"-tool. Then click and drag to create the rectangle and adjust its size. Upon mouse release, the tool will go into "skew"-mode instead and the mouse cursor changes. Then you can click and pull left or right to adjust the x skew to create a parallelogram.
- To change order of the shapes, select a shape with the select tool or click on the shape in the list. Then click the "Up"-button to move it upwards or the "Down"-button to move it downwards.
- To save or open SVG files, simply click "Open..." or "Save As..." buttons in the file menu.

3 Design Decisions

3.1 Java FX

Java FX was chosen as user-interface framework instead of Swing due to a couple of reasons. First of all, I had never worked with Java FX before so I wanted to learn it. In addition, Swing is rather outdated. Java FX is a newer platform with a better API and support for CSS. It seems to be the future platform of use for GUI applications.

3.2 MVC Pattern

The Model View Controller (MVC) pattern was chosen as the basis for this application because this is an ideal application for it. The MVC pattern separates business logic from the data and from the presentation layer which decouples classes and makes them more coherent. The drawing application only needs one view and one model which makes it suitable for MVC.

3.3 Observer Pattern

The observer pattern was used to enable the view classes to easily be updated when the model changes, for example when new shapes are added or removed. It is especially beneficial when multiple classes need to be updated due to model changes.

3.4 Factory pattern

The factory pattern was used to handle the mouse events in a efficient manner. There are multiple mouse event handlers, one for each tool and for major functions such as resize and rotate. There is a main *MouseEventHandler* which all shapes are registered in and that routes mouse events to the appropriate handler depending on which tool is currently selected. All mouse event handlers implement the *ToolEventHandler* interface, so the *EventHandlerFactory* can return the correct handler depending on which tool is passed to it.

3.5 SVG file export/import

SVG was chosen as the format for the opening and saving of files. This is because it is an excellent format to save vector drawings in since it is standardized, based on XML which makes it possible to open in multiple applications and browsers. This way, you can view the drawings in any program or even edit them in other programs.

An alternative considered was simply serializing the Java objects and open them later on. However, this was rejected because it would mean that it would not be able to open or edit in other applications.

4 Problems

There were some minor problems in the development of the application. For example, SVG files created in other programs are not possible to open in this application because it does not support all SVG XML-tags.