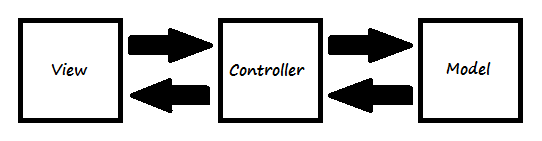
Architecture Design

MVC

- Model-View Controller pattern is used while developing this software to ensure that model (data) is not directly interacting with the view (interface) of the software. In order to achieve this, we have separated our software to 3 main packages (model, view, controller).



1. Model

- Model provides the data and determines the current state of application. The data which is required in this software are the svg elements and its attributes such as Rectangle, Line and Circle which has the attributes of fill, stroke and etc.

2. View

- View determines the appearance of the software such as the drawing panel for display and drawing svg elements, drawing kit as the toolkit for selecting the different types of actions to perform and etc.

3. Controller

- Controller interprets the events (mouse, keystroke) based on user interactions from the view and either updates the view or specify changes to model such as mouse click, mouse drag or keyboard actions.

Benefits of using MVC

1. Less coupling between data and appearance (model and view)

2. Easier to perform maintenance and modification for future enhancement and updates

3. Assurance of consistency for the view with current model

Basic Data Type Classes

1. PAColor

- This class is used for defining and determining the colours of svgs’ in this program instead of using the standard java.awt.Color which has limited choice of colours. It has an enumeration of all available svg color keywords which is more than the default in the Java Color class. In addition, PAColor also validates and correctly converts attributes read from svg files into proper Color objects to be used in Java application.

2. PAUnit

- This class is used to read unit lengths from attributes of svg elements and converts it to pixels unit based on dpi. The units length includes mm, cm, in, em, ex, pc, pt, px and %.

3. PASVGElement, PARectangle, PALine, PACircle, PASVGGroup

- This class is used to define the svg elements and is the parent class for PARectangle (<rect>), PALine (<line>), PACircle (<circle>) and PASVGGroup (<g>). PASVGElement contains basic attributes which all subclasses will have to implement which are id, stroke, stroke-width and fill (with the exception for PALine as lines do not have fill attribute).

4. PASVGContainer

- As PASVGElement is a generic class for all the svg elements, PASVGContainer has a JCF which in this program is a LinkedList to store all the PASVGElement. Other than that, PASVGContainer also stores the file name which would be “Untitled” for newly created svg by the program or the actual file name from the existing opened svg and the File of it. In addition, it also stores a PASVGTag object which contains all the attributes for <svg>.

5. PASVGTag

- This class contains all the attributes set for <svg> tag. The example for the common attributes of <svg> tags are width and height.

6. PAAttributeConstant

- This interface contains all the constants needed for this program such as the default fill and default stroke when an error is found when reading from the svg.

7. PASVGImport

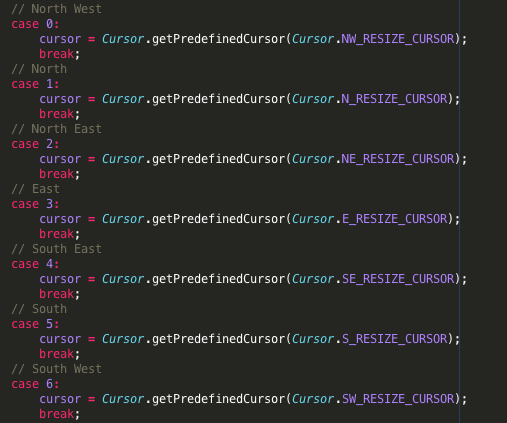
- This class reads and validate a file which is then converted to be readable by Java. In addition, this class also read all svg elements from the svg file and stores into a LinkedList in PASVGContainer.

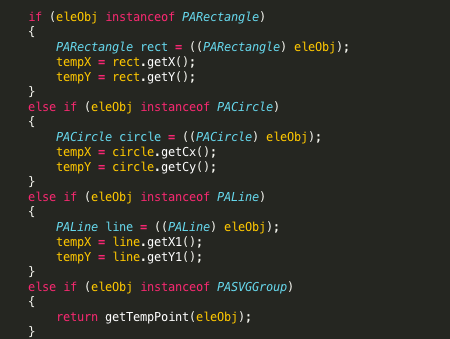
Coding Style

Every programmer has their own unique coding convention of style. As this is a group assignment, we have faced problems such as we are not able to get used to each other’s convention. In the end, we just agreed to 1 coding convention based on majority votes.

1. “If Else” vs. Switch case

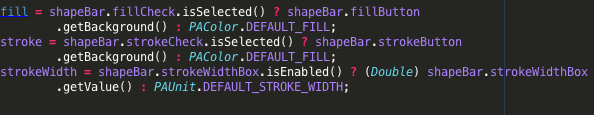
- In some cases when there is a conditional statement to check, though some would use if else to check for the correct condition, switch case would perform better than using if else. Thus, we changed “if else” to switch to improve the performance even though the changes in performance are actually minimal and not very significant from user’s view.



However, not all cases is suitable or even usable by switch case. For example:

2. “If Else” vs. “Condition ? Result A : Result B”

- In some cases, using “? :” would reduce the number line of code for simple “if else”. For example:



3. Setters

- During setting of a variable:

i) setFill(color);

ii) this.fill = color;

- Althought both ways results are the same, the coding convention is clearly different as 1 prefer to create a setter method to set the variable and the other just use usual assignment method.

4. Indentation

- Though indentation will not affect the outcome of the program, but not all of us prefer to same indentation.