Java Final Project Report

--XDF\_2DGraphics

Group member:

Bonan Chen

Jiahui Zhang

Yujie Ren

## 1.Abstarct

Scalable Vector Graphics (SVG) is an [XML](https://en.wikipedia.org/wiki/XML)-based [vector image format](https://en.wikipedia.org/wiki/Vector_image_format) for [two-dimensional](https://en.wikipedia.org/wiki/Two-dimensional) graphics with support for interactivity and animation. SVG allows for three types of graphic objects: vector graphic shapes (e.g., paths consisting of straight lines and curves), images and text. Graphical objects can be grouped, styled, transformed and composited into previously rendered objects. The feature set includes nested transformations, clipping paths, alpha masks, filter effects and template objects.

Since Java do not provide native SVG support, our group has to figure out how to parse SVG, how to save it into internal format, and how to display it on the screen.

Our group created a SVG viewer, and imported SVG file into the SVG viewer. After parsing SVG file, our group extracted most of shape attribute information in the SVG file and cached this information into DOM tree structure. At last, we printed the information in the DOM tree and draw them on the JSVG Canvas which is a drawing board to display SVG file.

## 2.Role of Batik for SVG

Batik is a Java-based toolkit for applications or applets that want to use images in the SVG format for various purposes, such as display, generation or manipulation. It is a main library that is utilized in our project.

Examples of [modules](https://xmlgraphics.apache.org/batik/using/index.html) are the [SVG Parser](https://xmlgraphics.apache.org/batik/using/parsers.html), the [SVG Generator](https://xmlgraphics.apache.org/batik/using/svg-generator.html) and the [SVG DOM](https://xmlgraphics.apache.org/batik/using/dom-api.html). Another ambition for the Batik project is to make it highly [extensible](https://xmlgraphics.apache.org/batik/using/extending.html) —for example, Batik allows the developer to handle custom SVG elements. With Batik, we can manipulate SVG documents anywhere Java is available.

In our project, all figures being parsed are drawn by shape class and saved into DOM tree. DOM objects that represent the SVG content corresponding to the rendering calls made on the SVGGraphics2D instance. In other words, every time a program invokes a rendering method, such as Rect, on a SVGGraphics2D instance, a new DOM object, representing the SVG equivalent, is appended to the DOM tree.

## 3.Component

Our project can be divided into three parts. One is importing SVG, one is parsing SVG information and extracting information, the other is drawing SVG according to the extracted information.

# Parse SVG abstract class

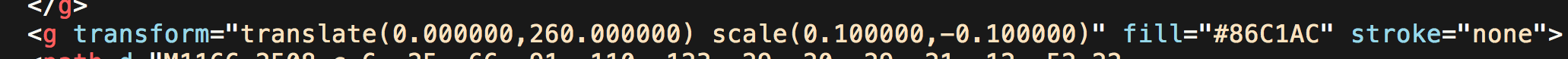
Creating SVG file abstract class will increase speed of code and help us simplify subclass definition. It contains common attributes that most SVG should have. These attributes will be passed into drawing method at last. The class should also contain parse() method since all of subclasses need to inherit this method and implement it to parse SVG file.

# Parse Root class



SVG file contains some root attributes such as width and height of display area, viewBox. It describes dimension of display area, so our group have to extract this information. We take width, height and viewBox as key words and use regex to match these variables in SVG file and save them into variables in class. To test extract information, we write an output method to print variables.

# Parse G class



The <g> [SVG](https://developer.mozilla.org/en-US/docs/Web/SVG) element is a container used to group other SVG elements. Transformations applied to the <g> element are performed on all of its child elements, and any of its attributes are inherited by its child elements. It can also group multiple elements to be referenced later with the [<use>](https://developer.mozilla.org/en-US/docs/Web/SVG/Element/use) element. As we all know, there are so many attributes in <g> element. Because of time limitation, we cannot extract all attributes from g element. We extract regular attributes in <g> element, and set them into its child elements. Although it cannot display all <g> elements, it does not influence the display on the screen.

# Parse circle class



Circle element is to draw circle in SVG file. It contains radius, position of circle center, stroke, and style these information. All we need to do is to extract them out of xml language with regex and save them into private variables, then create get method to get these variables.

# Parse rect class



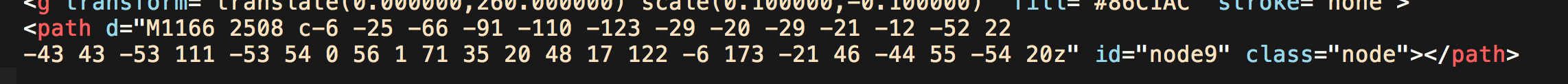
Rect element is to draw rectangle in SVG file. Some unique attributes such as width, height, rx, ry are in this element. We use regex to extract these attributes, save them into private variables, and create get method to get these variables.

# Parse Ellipse class



Ellipse element is to draw ellipse in SVG file. It has unique attributes such as cx, cy, rx, ry. We use regex to extract them and style attribute, save them into private variables, and create get method to get these variables.

# Parse Path class

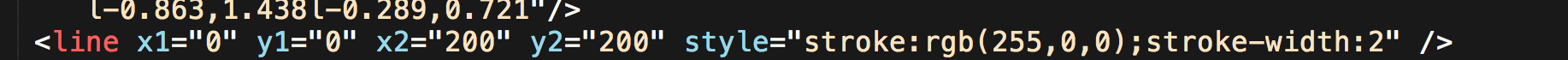


Paths create complex shapes by combining multiple straight lines or curved lines. It has a unique attribute d which is actually a string which contains a series of path descriptions.  In the path element, there are Moveto, Lineto, Curveto, Arcto, and ClosePath instructions. We use regex to extract d and style attributes, save them into private variables, and create get method to get these variables.

# Parse Poly class

This class is to extract polyline and polygon information, since polyline and polygon have the same information in SVG file except for tags. We use regex to extract points and style information in the file, save them into private variables, and create get method to get these variables.

# Parse line class



This class is used to extract line information in SVG. We use regex to extract start position. end position and style information in the file, save them into private variables, and create get method to get these variables.

# Shape

Shape class is an abstract class. It can help us simplify define other shape class.

Circle, Rect, Line, Polyline, Path, Polygon class

In order to draw SVG on the screen, our group use batik library to create DOM tree structure. As we know, each SVG consists of different shapes. Each shape consists of different attributes. When draw SVG on the screen, parse classes extract all attributes of shapes, then add these attributes into corresponding shapes. These shapes are also added to a DOM tree. Finally, printing the tree on the JSVG Canvas will display SVG on the screen. These shape classes are used to add attributes after SVG is extracted.

# Import SVG class

This class is to import SVG class line by line in the form of string. Each line which is imported need to be parsed. At first, each line should be identified which tag it consist. According to identified tag, it calls corresponding parse class to extract information until this shape has been extracted completely. Then, it will continue identifying other tag until all SVG has been imported.

Compound Shape class

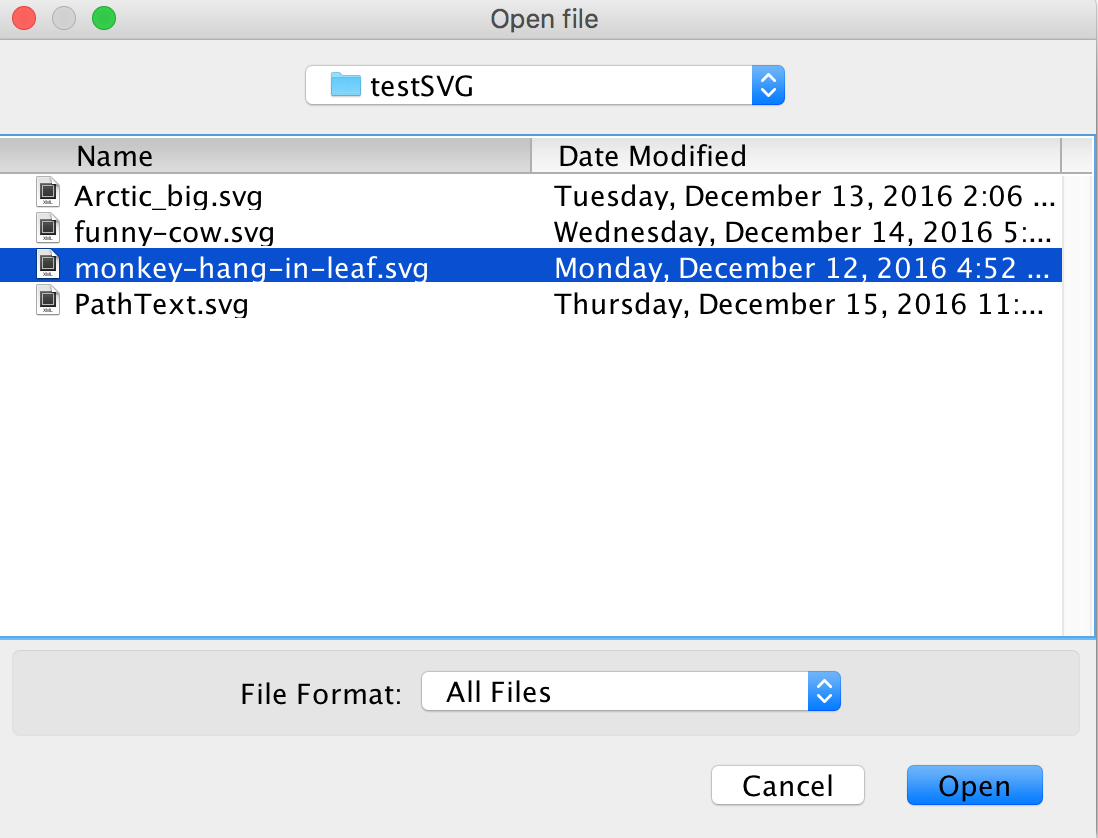
This class is to print DOM tree on the JSVG Canvas.

Drawing class

Drawing class is a test class. Our group create a panel added to a frame. JSVG Canvas is added to the panel and displayed on the screen.

## 4.Function description

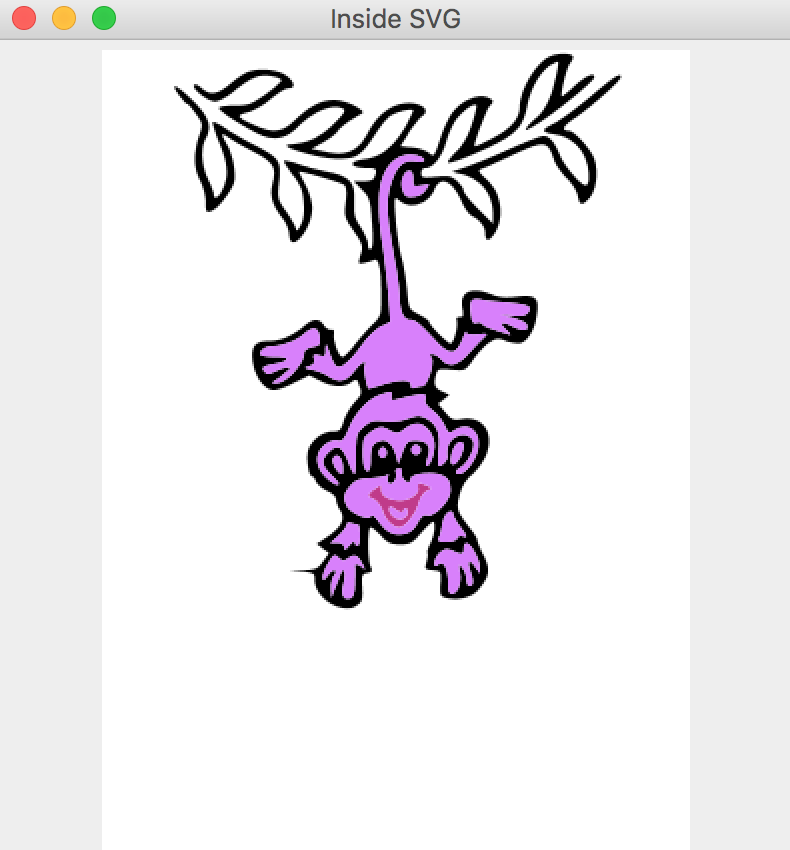
# Funtion1 import SVG into document processor

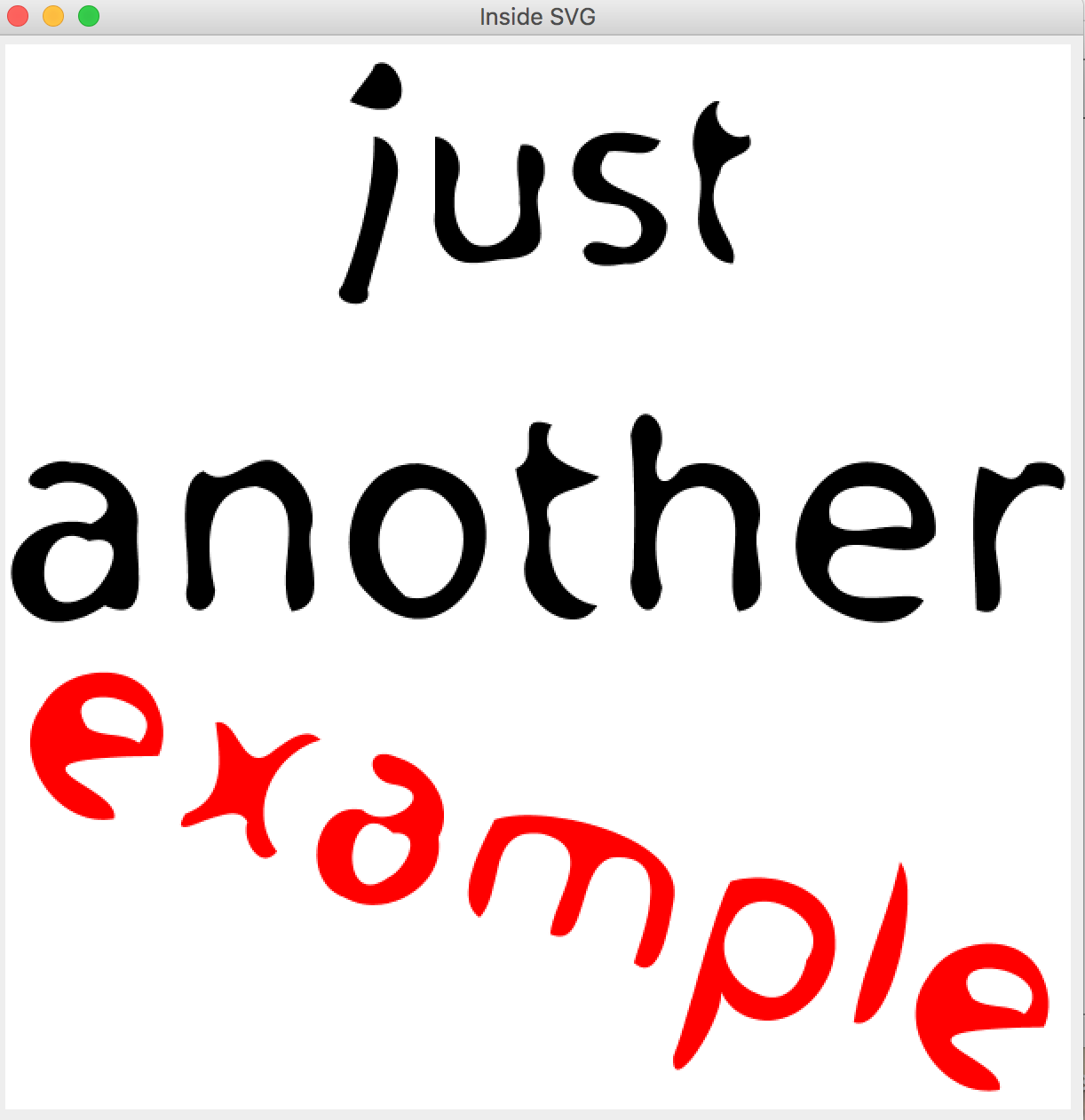


# Function2, parse SVG file and extract shape information

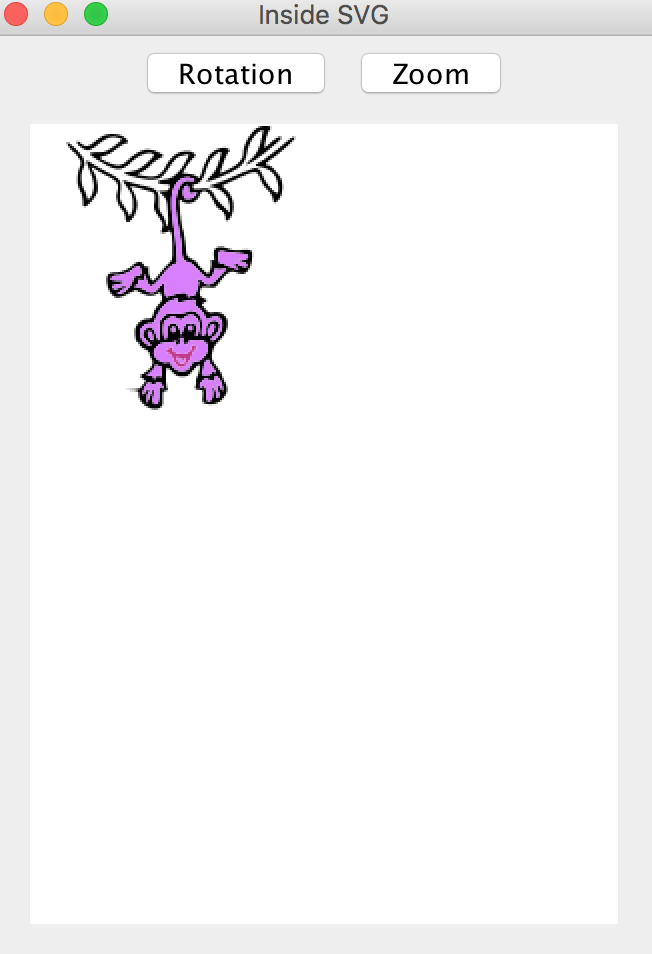
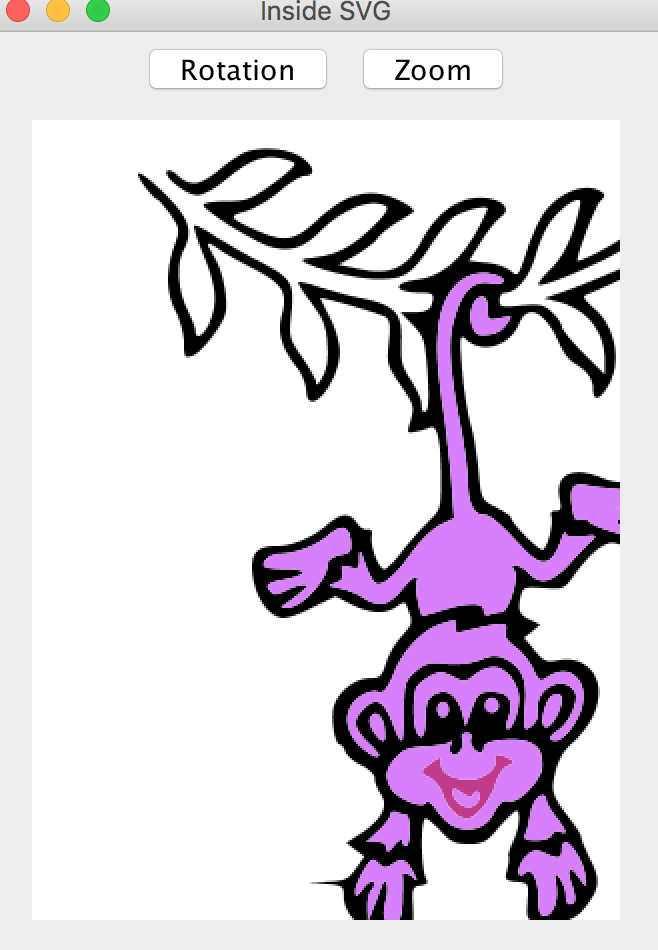


# Function 3, draw SVG on the screen

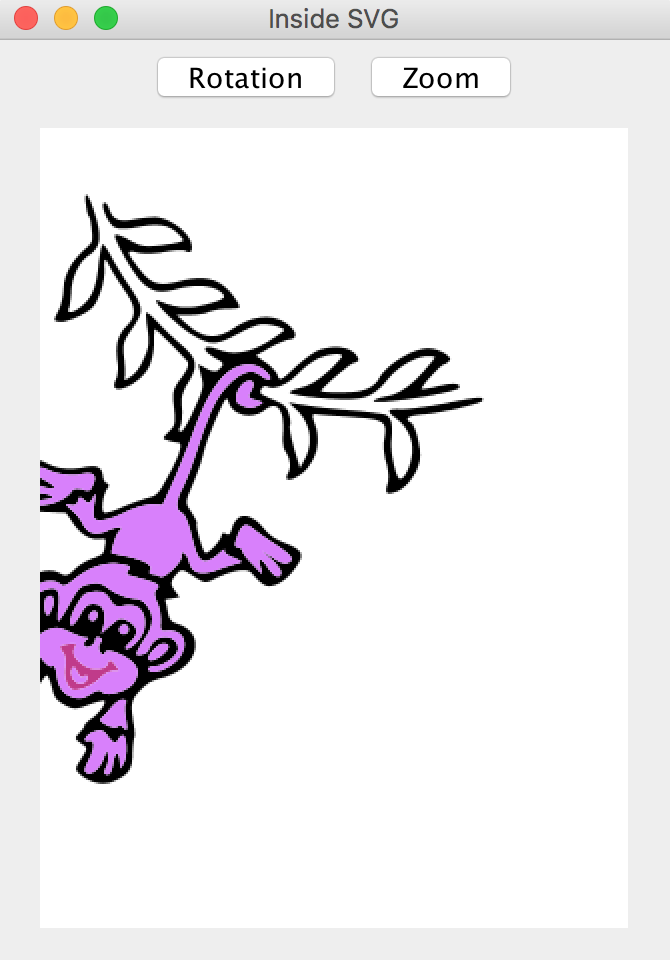




# Function 4, implement SVG file zoom in or zoom out



# Function 5, implement rotation function



## 5.Future work

1) Implement elements which are unable be displayed on the screen such as gradient, text.

2) Complete zoom in and zoom out of SVG.

3) Fix display bug.