

**Java2D Graphics – Project 1**

Vincent Testagrossa

CMSC 405 6380 Computer Graphics (2232)

Professor: Lauren King

Due January 24, 2023 11:59 PM

Contents

[Project Description 3](#_Toc125196673)

[Testing 4](#_Toc125196674)

[Figures 6](#_Toc125196675)

[Lessons Learned 8](#_Toc125196676)

# Project Description

The purpose of Project 1 was to demonstrate the use of Java 2D graphics. The scene must contain three images which were generated in separate methods from a 2D array, are at least 25X25 pixels, and have two or more colors. The images are then translated -5 in the X direction, 7 in the Y direction, rotated 45 degrees counter clockwise, 90 degrees clockwise, and finally scaled 2.0 for the X component and 0.5 for the Y component.

Logo

Description automatically generated

# Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Case | Input | Expected Output | Actual Output | Pass / Fail |
| 1 | Creates 3 different, simple, minimum 25x25 binary 3 images. | One 25X25 blue triangle, one 30X30 red triangle, and one 40X40 green triangle, all with black borders. | One 25X25 blue triangle, one 30X30 red triangle, and one 40X40 green triangle, all with black borders. | Pass |
| 2 | Stores images as 2D arrays. | Images are generated as a 2D array by ImageTemplate via a nested loop. | Images are generated as a 2D array by ImageTemplate via a nested loop. | Pass |
| 3 | Consists of two or more colors. | Each image is 3 colors. A black border with a fill color and a white background. | Each image is 3 colors. A black border with a fill color and a white background. | Pass |
| 4 | Uses Java 2D Graphics to display your original images. | g2 is a Graphics2D object that’s used to set the translation, rotation, scaling, and draw each image. | g2 is a Graphics2D object that’s used to set the translation, rotation, scaling, and draw each image. | Pass |
| 5 | Translates each image -5 in x direction. | Each image moves -5 in the x direction in the second frame. | See [Figure 1](#Figure1) and [Figure 2](#Figure2). | Pass |
| 6 | Translates each image +7 in the y direction. | Each image moves +7 in the y direction in the third frame. | [See Figure 3](#Figure3). | Pass |
| 7 | Rotates each image 45° counter clockwise. | Each image rotates 45° counter clockwise in the fourth frame. | See [Figure 4](#Figure4). | Pass |
| 8 | Rotates each image 90° clockwise. | Each image rotates 90° clockwise in the fifth frame. | See [Figure 5](#Figure5). | Pass |
| 9 | Scales each image 2 times for the x component | Each image scales 2 times for the x component in the sixth frame. | See [Figure 6](#Figure6). | Pass |
| 10 | Scales each image 0.5 times for the y component. | Each image scales 0.5 times for the y component in the seventh frame. | See [Figure 7](#Figure7). | Pass |
| 11 | Displays transformations in sequence. | Each transformation happens frame by frame, with one transformation for all the images per frame. | See [Figures 1-7](#_Figures) | Pass |

# Figures

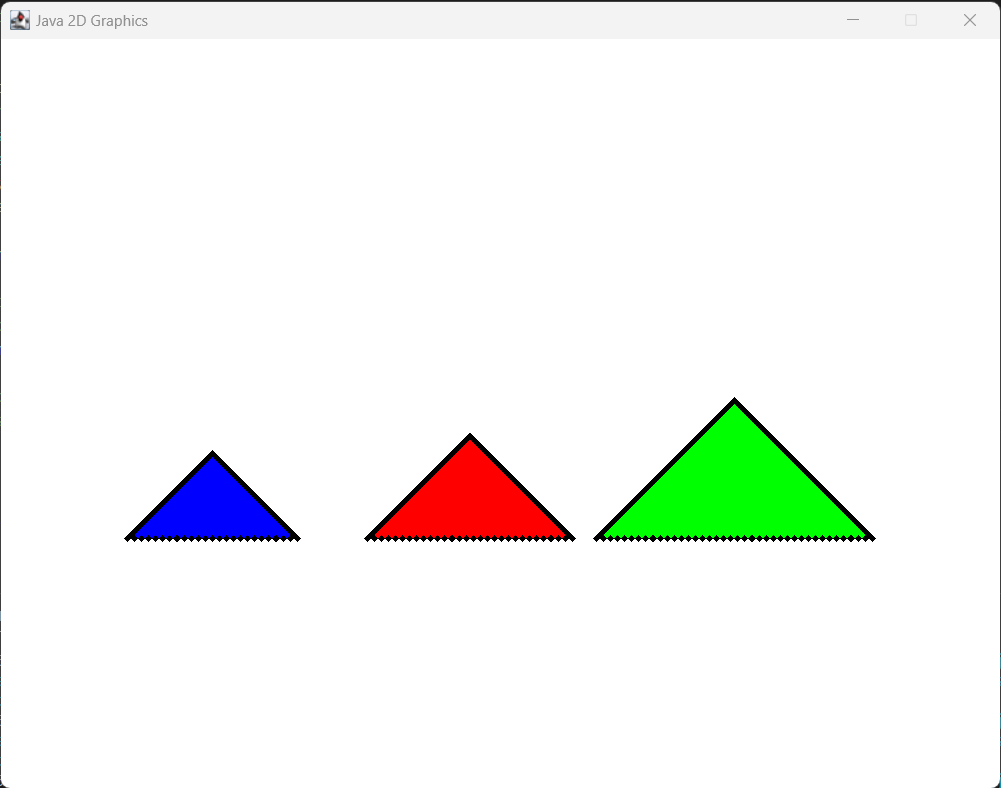


Figure 1 – Triangles are at (0,0) with their respective offsets added.

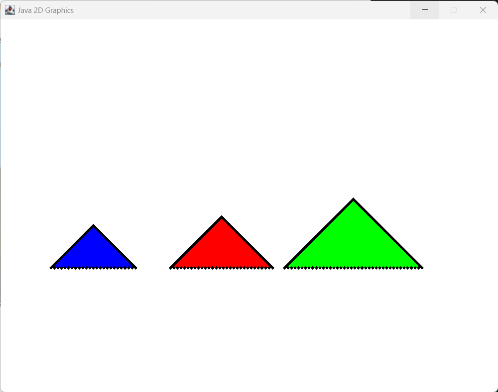


Figure 2 – Triangles move left 5 pixels.

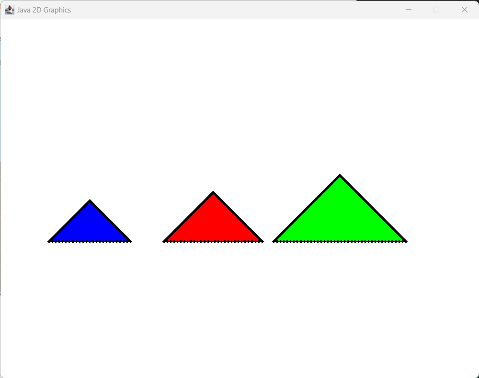


Figure 3 – Triangles move up 7 pixels.

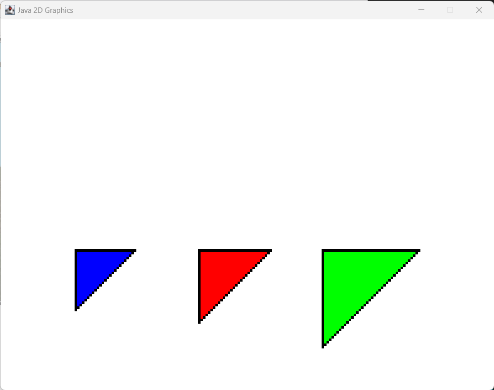


Figure 4 – Triangles are rotated counter clockwise 45 degrees.

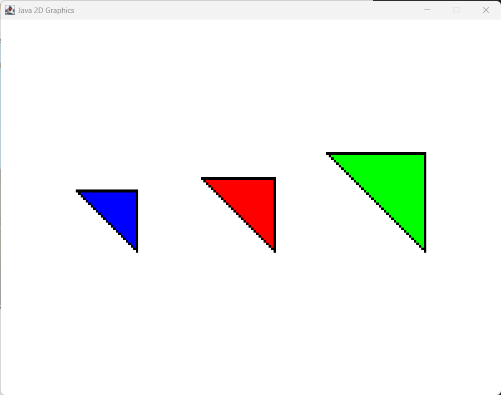


Figure 5 – Triangles are rotated clockwise 90 degrees.

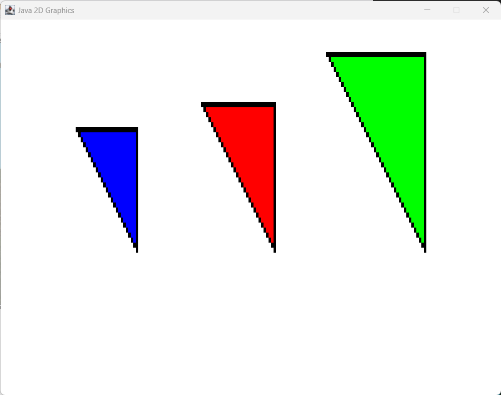


Figure 6 – triangles are scaled by 2 times in the X component.

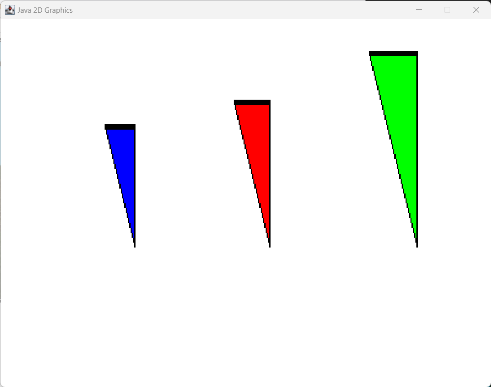


Figure 7 – Triangles are scaled by 0.5 times in the Y component.

# Lessons Learned

1. I gained a new appreciation for the APIs that I’ve used in the past. Having to perform each transform manually was a struggle when I was trying to animate the transforms in “slices” before I decided to rework the project again.
2. It took me a while to understand what the purpose of g2.setTransform(savedTransform) was, which is part of the reason I was struggling to get the previously mentioned transition animations to work correctly. After reworking the project, I’ve figured out what I was doing incorrectly. Part of my issue was a silly mistake about how Java passes arguments (by value instead of by reference), which I should have figured out far sooner.
3. I ran out of time to implement smoother animations, but I think I can do what I wanted now, especially after the creation of the RightTriangle class (which would need to be refactored into a subclass of a new Image class).