**Image Creation/Transformation with Java 2D Graphics**

John Kucera

Prof. Amitava Karmaker

CMSC 405 6380 Computer Graphics (2218)

25 August 2021

**Included files**: ImageCreation.java, ImageTransformation.java

**Purpose of program**: To create 3 images with Java 2D graphic methods and perform transformations on them in a cycling animation

**Descriptions of methods**:

* *ImageCreation* class:
  + Store 25x25 pixel positions as integers in arrays to later be “converted” into images. The integers represent different colors for the pixels.
    - Letter W: Left side is BLACK, Right side is RED
    - Number 5: Horizontal lines are BLUE, Vertical lines are GREEN
    - Shape Star: Top side is PINK, Bottom side is CYAN
  + *getImage()* method: Use the arrays that represent images to create and return Java BufferedImages. Pixel colors are set depending on the stored integers.
* *ImageTransformation* class: Extends JPanel as a container for images to be displayed
  + Creates BufferedImages from the *ImageCreation getImage()* method
  + *main()* method: Label and edit properties of JFrame that will display the images. Starts an animation timer that has an action listener which continuously assigns frame number as time goes on and repaints JFrame to show each transformation.
  + *ImageTransformation()* method: Class constructor. Sets size of JFrame.
  + *paintComponent(Graphics g)* method: Uses Java 2D Graphics to drawing and transforming images. Based on frame number, different transformations are performed on images such as translations, rotation, and scale. The dimensions by which transformation happens is applied to drawn images, then repainted in *main* method.
  + *applyWindowToViewportTransformation()* method: Edit properties of viewport, such as where the images will be displayed and how zoomed in it is.
  + *resetImages()* method: Reassign variables that represent image positions to their initial values (before transformations). Used so that all images can continue to cycle.
  + *drawMyImage()* method: Apply values that represent image positions and transformations to the image given as parameter. Images are properly distanced by tracking image number and translating them a distance of pixels based on their number.

**Program running in Netbeans IDE**:

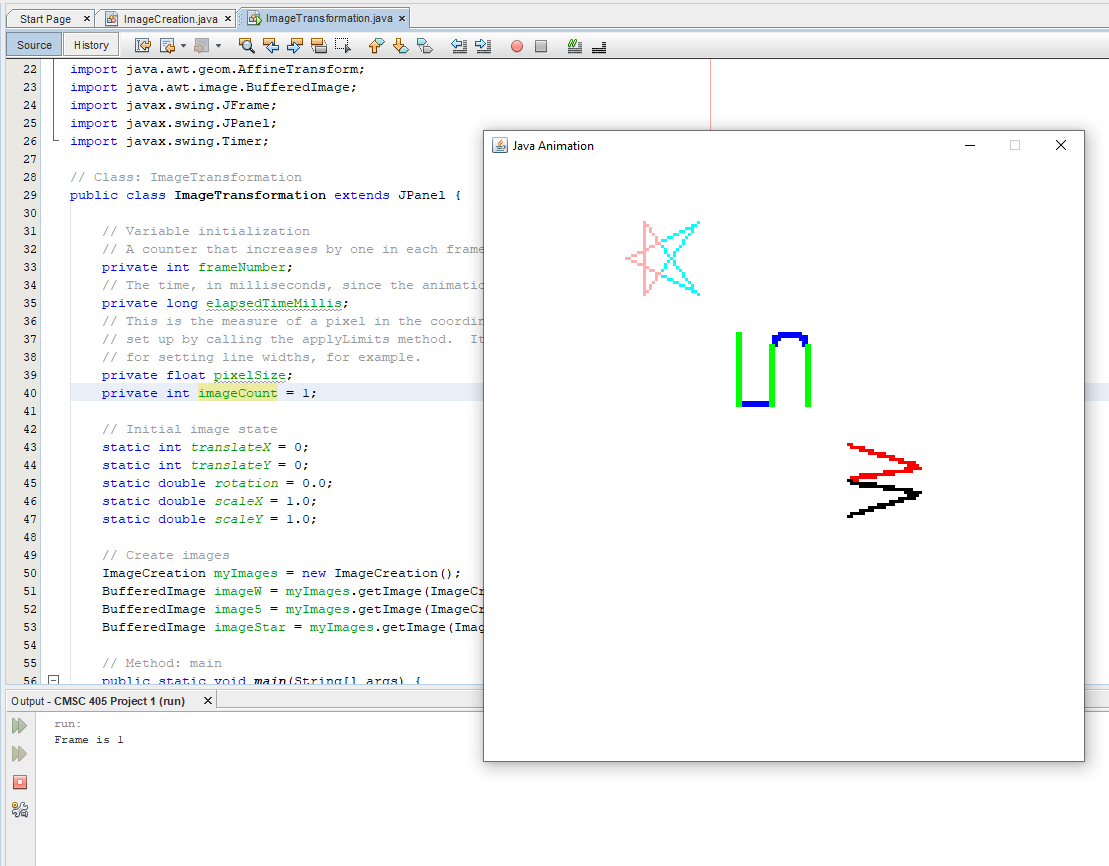


Figure 1: Program running in Netbeans IDE

**Test Plan & Results**

|  |  |
| --- | --- |
| **Plan** | |
| After pressing Run and the JFrame is displayed, first make sure that images are properly drawn and colored. Then, as animation goes on, make sure each desired transformation is performed in the proper sequence. Frame number should be output in console. After all transformations, images should be reset to initial states and transformations should start occurring again. | |
| **Frame # + Description** | **Result** |
| **(SUCCESS)**  Frame 1. Images are drawn perfectly, creating proper W, 5, and Star shapes which are all 25x25. W has Black left side and Red right side. 5 has Blue and Green lines. Star are Pink top and Cyan bottom. They are properly distanced from each other and can be seen in the JFrame. In the console, “Frame is 1” is correctly printed as output. | Figure 2: Frame 1 |
| **(SUCCESS)**  Frame 2. Transformation occurs correctly from initial position, translating -5 in the x direction and +7 in the y direction. “Frame is 2” is correctly printed as console output. | Figure 3: Frame 2 |
| **(SUCCESS)**  Frame 3. Transformation occurs correctly from Frame 2, rotating 45 degrees counterclockwise. “Frame is 3” is correctly printed as console output. | Figure 4: Frame 3 |
| **(SUCCESS)**  Frame 4. Transformation occurs correctly from Frame 3, rotating 90 degrees clockwise. “Frame is 4” is correctly printed as console output. | Figure 5: Frame 4 |
| **(SUCCESS)**  Frame 5. Transformation occurs correctly from Frame 4, scaling 2 times in x component and 0.5 times in y component. “Frame is 5” is correctly printed as console output. | Figure 6: Frame 5 |
| **(SUCCESS)**  Frame 1 (Repeated). All previous transformations are gone, and images are reset to initial state and position. “Frame is 1” is correctly printed as console output, and transformations continue like this infinitely. | Figure 7: Frame 1 (Repeated) |