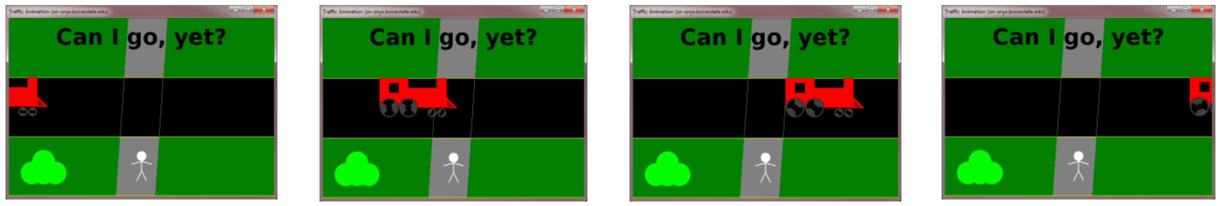
Project 0: Traffic Animation

CS121 Computer Science I

Overview

In this assignment, you are going to create an animation of a vehicle that moves across the window in a traffic lane. You have the flexibility to draw whatever you want, as long as it has the same relative complexity as a multi-part vehicle and still involves multiple shapes and colors.



Objectives

- Add code to a method in a given skeleton class to paint a scene.
- Use methods from the Graphics class.
- Use variables to manage data and expressions to calculate size and position of your shapes.
- Use the Color class.

Getting Started

- Create a new folder in VS Code for this assignment and save TrafficAnimation.java into that folder.
- Don't forget to reference the graphics examples that you checked out in VS Code if you need some help getting started.

BOISE STATE UNIVERSITY

Project Specification

Create an animation of a vehicle moving across the window in a lane. Your scene will include an avatar watching the vehicle move by and some interesting scenery.

Quality Requirements (20 points)

1. Project Submission
    - Code submitted to the correct cs121 section and with **only** the required files
2. `README.md`
    - `README.md` file is named properly and has the required sections including your reflection (see Documentation section)
    - **Reflection**: Write a two paragraph reflection describing your experience with this project. Talk about what worked well and what didn't work so well. Did you run into an issue that took some time to figure out? Tell us about it. Please write the reflection in your `README.md` file.
3. Javadoc Comments
    - Project should have a proper JavaDoc formatted comment (see Documentation section) for the class and all methods.
4. Coding Style
    - Code should be properly and consistently *indented* and *vertically spaced*
    - Variable and Method names should follow the *camelCase* naming convention
    - Class names should follow the *TitleCase* naming convention
    - Constants should be in all *CAPITAL_LETTERS*
    - In-line comments to describe what different code sections do

Code Requirements (80 points)

Complete the TrafficAnimation class by doing the following in its paintComponent() method:

1. Draw your **vehicle** relative to an anchor position (the `xOffset` variable) so that the position of your vehicle will change every time the paintComponent() method is called.

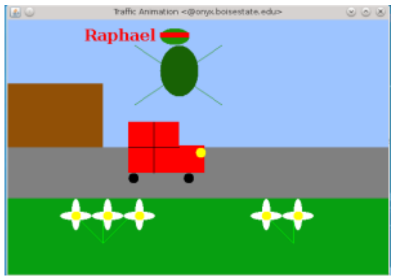
    - Your vehicle must have a minimum of a cab, body, and two wheels. If you choose to make your "vehicle" something other than a car, it must still be composed of at least 4 different shapes. It must also scale proportionately when resized.

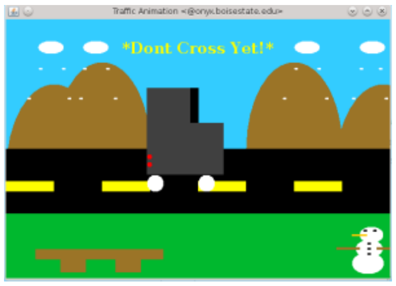
2. When the vehicle goes off one edge of the window, it should come back on the other edge.
    - The `TrafficAnimation()` constructor method in the provided starter code sets up the animation and updates the `xOffset` anchor coordinate for you. All you have to do is make sure that you draw your vehicle relative to this coordinate for it to move and wrap.
3. Draw a visual **lane** along which the vehicle moves, or equivalent background elements to create a scene. (See the examples.) Make sure the lane and scenery scale correctly by specifying the position and dimensions relative to the width and height of the screen.
4. Draw an **avatar** observer to the scene, out of the path of traffic.
    - Your avatar must be composed of a minimum of a circle, with eyes, nose, and mouth (or at least 4 other shapes). It must also scale proportionately when resized.
5. Use **at least five *different* methods** from the `Graphics` class to draw the vehicle and environment. This means different shapes (rectangles, ovals, lines) as well as both fill and draw methods. (Multiple fillRect method calls is nice, but it is just using one method -- use at least 5 different methods.)
6. Display some **text** in the window - a title, a quotation or whatever seems appropriate for our scene.
7. Use **at least three colors** in your scene. **At least one of them should be a custom color**. There are a number of sites on the web where you can find color samples with their RGB values.
8. Make your scene adjust appropriately when the window gets resized. For example, centered text should remain centered, the vehicle should travel the entire width of the window, and all graphics should adjust or scale as needed. This image provides some hints on how scaling and positioning can be accomplished.

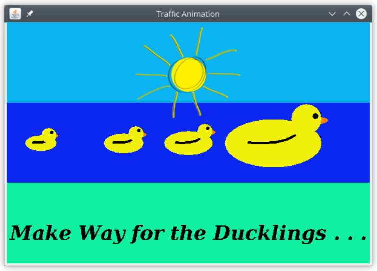
NOTE: You may add whatever else you want to the scene to make the picture interesting! :)

Inspiration

Here are some sample solutions to give you an idea of what your scene could look like. You can be creative!







Testing

Once you have completed your program, copy your **TrafficAnimation.java** file and **README.md** file into a directory on the onyx server, with no other files (if you did the project on your computer). Test that you can compile and run your program from the console in the lab to make sure that it will run properly for the grader.

```
javac TrafficAnimation.java
java TrafficAnimation
```

Documentation

Update the class **javadoc comment** located immediately before the class header.

- Provide a good description of the program at the beginning of the javadoc comment.
- Your class comment must include an @author tag with your name at the end of the comment. This will list you as an author of the program when documentation is generated from the javadoc comments.

Include a plain-text file called **README.md** that describes your program and how to use it. Expected formatting and content are described in README_TEMPLATE.md. See README_EXAMPLE.md for an example.