Project 3 Documentation

Jade Pearl

CMSC 335

Professor Andrew Seely

**UML Diagram**

Below is the UML diagram for this project:

A screenshot of a computer

Description automatically generated

**User Guide**

For set-up before running the program, make sure all required files are in the same directory prior to running. These are the names of said files that need to be in the same directory:

* TrafficSimulation.java
* TrafficDisplay.java
* Car.java
* TimeStamp.java
* car.png
* red.png
* green.png
* yellow.png

Once all files are confirmed to be in the same directory, open them in a text editor that can run Java, preferably Visual Studio Code by Microsoft. These next directions for running are for Visual Studio Code.

Make sure to have the TrafficSimulation.java file open in the VSCode workspace (window). This file is the one containing the main method. To run the program, you simply need to go into the run tab in the top left corner of the window.

A screenshot of a computer program

Description automatically generated

When Run is clicked, there will be a dropdown menu. Be sure to click on the Start Debugging option and the code will automatically compile and run starting the GUI simulation.

Other text editors such as Eclipse IDE or NetBeans may have a similar run feature and therefore those could be used as well instead of VSCode.

**Test Data/Plan**

This project was not completely finished before I decided to submit it, so there are some test cases that are a failed case, but they will still be included.

|  |  |  |  |
| --- | --- | --- | --- |
| Test # | Description | Screenshot | PASS/FAIL Flag |
| 1 | Working time clocks |  | PASS – Time clocks use current time, so they successfully count in increments of one second intervals |
| 2 | Measurement of X Y coordinates of cars |  | PASS – As the cars move, the X coordinate position changes. In my program, they are moving from left to right in a straight line so the Y coordinate does not change |
| 3 | Cars stop when lights are red, slow down when yellow, drive when green | This is hard to show a screenshot for. | 66% PASS – I did not yet program the cars to slow their speed when its assigned light turns yellow. However, the cars do stop when the light turns red and drive a consistent speed when the light is green. |
| 4 | Successful running of threads |  | PASS (The screenshot is distorted) – The cars and lights are made to run at different speeds and the lights change at different times. This is indicative of successful threads |
| 5 | Functionality of start, stop, continue, and pause buttons |  | FAIL – The buttons were implemented in the GUI, but I was unable to get the buttons to work. The cars and lights start working automatically and the buttons do nothing |
| 6 | Ability to add more cars and intersections | No screenshot | FAIL – this was not implemented in my code and does not show up in the GUI. |

**Lessons Learned**

This project, serving as the final assignment for my course, has been rewarding for me as a developing programmer. I'm sincerely grateful for the challenge it presented as it provided an excellent opportunity to delve into something entirely new and broaden my Java programming skills. The GUI mixed with Threads proved beyond difficult, but I am thankful I got to try my hardest at this experience. Working on this project allowed me to gain invaluable insights into how threads synchronize and interact within a GUI, particularly in the context of a traffic simulation, which is a great use of a real-world problem and example. The complexity of coordinating elements like cars and traffic signals truly kept me interested in programming this project until I felt like I could not work on it further. Overall, I'm appreciative that this project pushed my boundaries and allowed me to gain practical experience while exploring new aspects of Java programming.

**Resources**

*Defining and Starting a Thread (The JavaTM Tutorials > Essential Classes > Concurrency)*. (2019). Oracle.com. <https://docs.oracle.com/javase/tutorial/essential/concurrency/runthread.html>

*Implementing Traffic Signal Using Java Swing Components*. (2021, September 2). GeeksforGeeks. <https://www.geeksforgeeks.org/implementing-traffic-signal-using-java-swing-components/>

*Lesson: Getting Started with Swing (The JavaTM Tutorials > Creating a GUI With Swing)*. (n.d.). Docs.oracle.com. Retrieved November 16, 2023, from <https://docs.oracle.com/javase/tutorial/uiswing/start/index.html>

*Lesson: Using Swing Components (The JavaTM Tutorials > Creating a GUI With JFC/Swing)*. (n.d.). Docs.oracle.com. <https://docs.oracle.com/javase/tutorial/uiswing/components/index.html>

*Lesson: Writing Event Listeners (The JavaTM Tutorials > Creating a GUI With Swing)*. (n.d.). Docs.oracle.com. <https://docs.oracle.com/javase/tutorial/uiswing/events/index.html>

*Multithreading in Java - GeeksforGeeks*. (2016, January 9). GeeksforGeeks. https://www.geeksforgeeks.org/multithreading-in-java/

*Trail: 2D Graphics (The JavaTM Tutorials)*. (n.d.). Docs.oracle.com. https://docs.oracle.com/javase/tutorial/2d/index.html