Jake Grosse
CS-410: Operating Systems
1 October 2021

Project Description of Meeting Requirements

I believe I met all of the requirements of the lab to the highest degree and with the following I hope you agree with me.

Writing the Code:

- Any code which wasn't provided to the class was written in its entirety by me, Jake Grosse. The only co-theorizing was in the form of non-code brainstorming with Matt Bushnell briefly, and Nate Williams and Ted Wendt in a more elongated context for concept handling.
- The only use of anyone else's code was to compare for debugging once I had completed my code to a sufficient point. I found that I was not instructing the Workers' threads what object they were handling after which I added "this" as the first parameter of the thread constructor within Worker.

Running the Code:

- The entire program runs as intended under the given input. (Multi-threaded programming with plants and workers on their own individual threads).
- There are no opportunities for invalid inputs; therefore, I handle them perfectly.
- I added a build.xml file which runs the program with Apache ANT.
- I handle InterruptedExceptions when joining threads. That's about it… Not sure if that constitutes the extra points, but I hope so!

Internal Code Documentation:

- All methods and variables are appropriately and intuitively named.
- Any code which came from the classroom has Nate Williams mentioned in comments even though this was stated unnecessary.
- Every class (including the enumeration, State) has a Javadoc comment which includes author and description.
- Every public method has a Javadoc comment with description, parameters if applicable, and returns if applicable.
- Every private method has comments within the code explaining the purpose of the method.
- Any areas perceived as obscure or areas which helped me learn have comments within the methods to explain either what is happening, why it needs to happen, or both.

External Design and Documentation:

- This document is fantastically formatted, appealing, and detailed!
- The class diagram is up to UML standards, is all-encompassing, and is visually appealing.
- Every piece of the assignment is in the GitHub repository for this lab in one place.

Jake Grosse
CS-410: Operating Systems
1 October 2021

Challenges Encountered

Throughout this process, my main challenges were in thought process. Coding for me has always been the easy part of any project in computer science. I code quickly, effectively, and relatively error-free most of the time; though, the last claim has changed in this class due to its difficulty and the broad concepts which are difficult to wrap my mind around. I really struggled visualizing the layout of this project in my head and when speaking about it out loud with my professors and peers. This resulted in making a program which handled everything in the plant instead of the worker. Both Jaden and Matt told me verbatim that "it's easier when you think of it like an actual plant with actual workers instead of a piece of code" which helped me break through my mental block quite efficiently along with a brief meeting with Nate Williams. I realized the mistake in my thinking and code, and, from there, it only took thirty minutes to draft up new code from scratch, thanks to the templates on the Moodle page, and fifteen to debug. Most of that time was spent finding a simple error—never telling the Worker threads what they were supposed to be running—a literal one word fix (plus a comma). That being incredibly frustrating, the trouble stopped there. Overall, I greatly enjoyed the project as an assignment, as a professional task, and as a learning experience! Thank you for the great opportunity!

Jake Grosse
CS-410: Operating Systems
1 October 2021

UML Class Diagram

Note that the "<<interface>>" is actually referencing the Runnable interface and the "<<enumeration>>" is referencing the State enumeration. For whatever reason, the Word photo viewer does not fully support SVG files as noted at the bottom of the image.