Ricky Nguyen ID: 219461201

EECS 1021

April 10, 2022

MINOR PROJECT

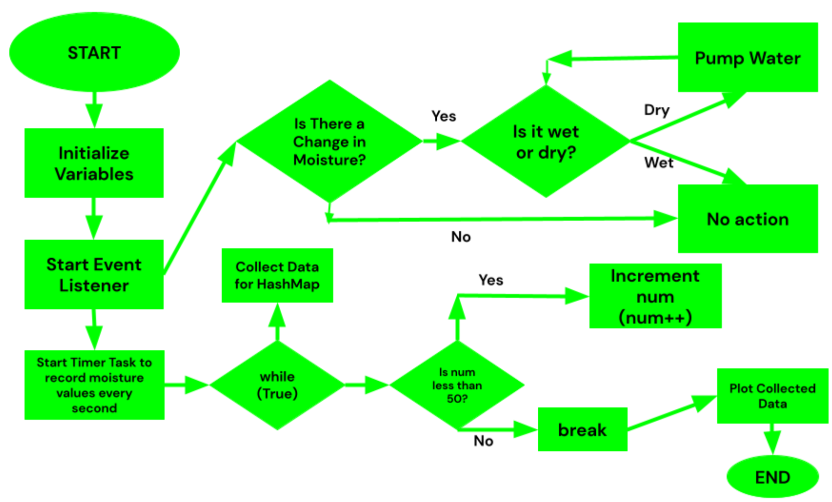
INTRODUCTION

Technology is constantly improving as days go by. We as humans want to utilize this to benefit our daily lives. How can we make daily processes simple and more efficient? This is what this minor project attempts to achieve. Watering a plant can be automated through a few lines of code which ultimately, will prove to be helpful and far more efficient.

CONTEXT

What an automated machine will do is utilize a state machine to moisturize a plant. To put it simply, the Java program will sense the moisture within the soil and act by pumping water if it is dry or not pumping water when it is wet. This automated machine can save time as well as be extremely helpful in cases where a human may forget to water their plant. Keeping plants alive is a crucial component of Earth. Automating watering systems can relieve humans from parts of this duty. The greener the Earth is; the healthier it is.

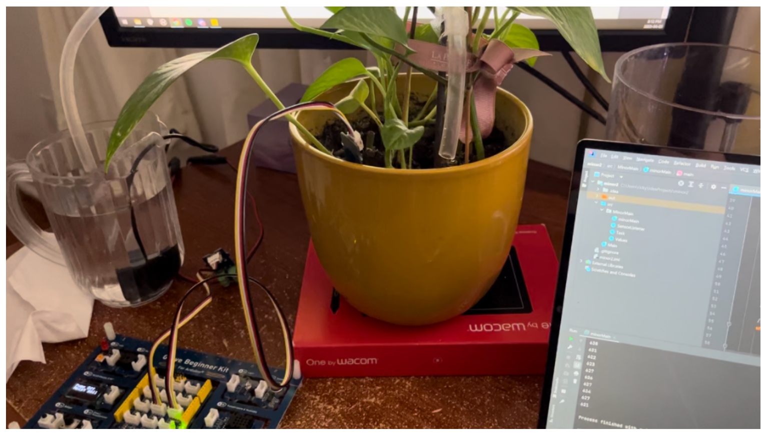
TECHNICAL REQUIREMENTS/SPECIFICATION



This picture on the right covers specifications that the system does. For instance, check moisture and respond or use timer task to plot data. More general requirements can be displaying soil and action status on the OLED display. StdLib and Firmata4j are significant components of this project. Firmata4j opens the connections between the program and the Arduino board. It also allows us to get pins and values of the moisture sensor or MOSFET switch. On the other hand, StdLib allows the program to plot data collected from the Arduino board as a function of time. In this program, the state machine is backed up by event listeners. These event listeners have conditional statements that respond to threshold moisture values. For plotting data, a while loop was introduced. In this while loop, the data put method was utilized to collect seconds and the moisture value in that time interval. This continues for a finite time until data for each is used to plot the HashMap.

COMPONENTS LIST

- MOSFET switch - Receives prompted commands from the program and switches the pump on or off depending on the situation
- Moisture sensor - Reads moisture levels and sends data back to the program
- Adruino board - Communicated with the program; sends and receives data
- USB cable
- Water pump and tube
- Water Reservoir
- Potted Plant
- 9V Battery
- Extra cables

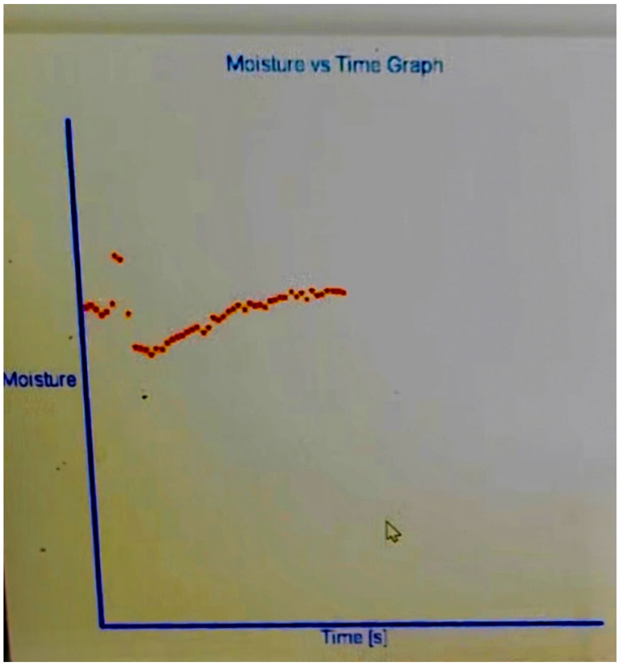


PROCEDURE

First, I drew a flow chart to fully grasp the understanding of how the code would work. This first sketch was not perfect as it was missing details. However, these details would later be filled in as I progressed. I began by initializing all variables required and using a print statement to obtain moisture values. From here I determined what values would be considered moist and dry and took note of these thresholds. With these values, I started to set up the event listener and filled the inside with conditional statements using the values observed. Now the machine knows what state the plant is in, how it should respond, and when. Now that the hardest part is over, I can set up the timer task to get moisture values every second and set up the graph using StdDraw. Lastly, I inserted a while loop that will collect data from the moisture every second and break after a certain amount of time then use StdDraw to plot the HashMap data. In addition to this, extra features such as the OLED display were introduced. Overall, the project changed significantly compared to the start. It started with a few simple lines of code which printed out moisture values to collections of methods and classes which cooperate to complete one goal; the automation of plant watering.

TEST

The testing of the system was simply achieved by observing the moisture values and the response to these values. When observing the graph, the spikes of voltage (moisture) would turn on the pump while the voltage drops would turn off the pump. In addition to this, to solve the issue of unknown threshold values of moisture, trial and error was a key role in finding the right values. This was a clear indication of a successful project through tests and debugging.



LEARNING OUTCOMES

My project addresses all learning outcomes required for this project with great efficiency and effectiveness. How?

- CLO 1: Demonstrate ability to debug. I observed graph values and compared them to actions to test the success of the program as well as utilized trial and error to find important threshold moisture values.
- CLO 2: Build an application that meets the given requirements. StdLib and HashMap were effectively used to graph out collected data from the moisture sensor while Firmata4j opened the connection between the program and Arduino.
- CLO 3: Use ready-made collections to solve problems. HashMap was utilized to collect moisture data for graphing.
- CLO 4: Build an event-driven application that controls sensors and actuators. Event listeners were used to monitor moisture changes and react accordingly.
- CLO 5: Describe how you used object-oriented concepts in your program. Many imports were used which made the program capable to connecting with physical objects.

CONCLUSION

Ultimately, the automation of watering plants can be extremely beneficial as it provides a quick response to conditions through state machines. This creates a stepping stone towards becoming a more efficient and healthier society.