Notes for Joseph

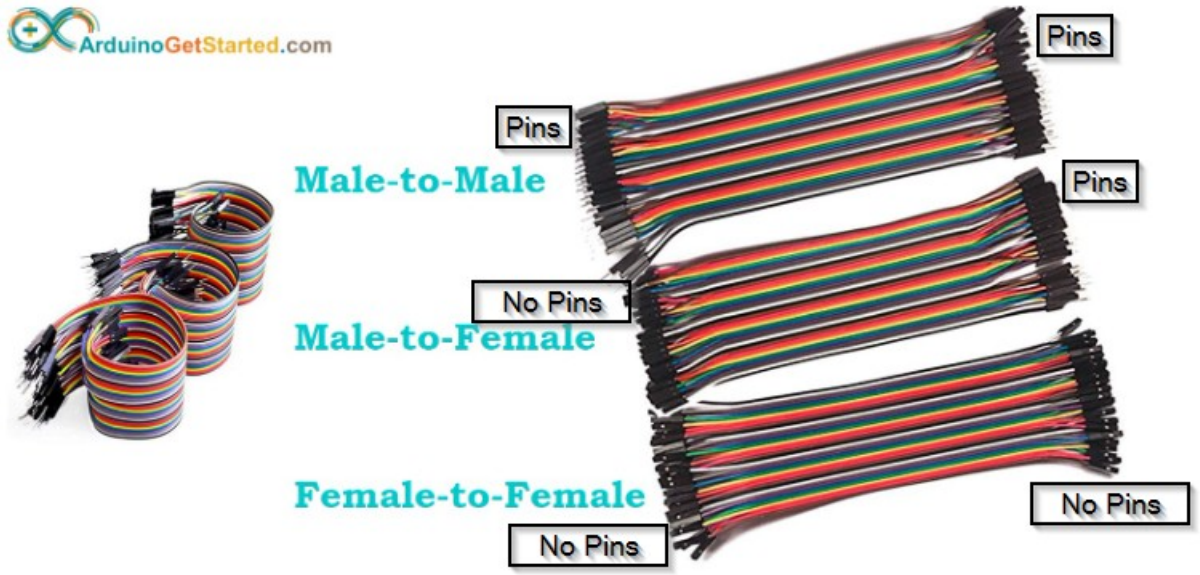
Notes:

- If you're ever curious about what a component does you need to go to the documents!

****Special Note**** As a coder or an engineer, reading documentation is a key component of your job. Once you understand the logic behind how to code, all you really need to advance your skills is how to use more complex code bases, libraries, etc. To understand these you need to get used to reading… A LOT!

Instructions:

1) Follow all the steps below to set-up the physical build of your device.
    a) Attach your components to the breadboard and the Arduino according to the diagram.
    b) Carefully choose which types of connectors you need:



1) Male to Male: Used to connect the Arduino to the Breadboard.
2) Male to Female: Use to connect a component to the Breadboard or Arduino.
3) Male to Male: Used to connect to components together.
4) Your LED and Buzzer will be connected directly to the Breadboard.

2) Copy the code into your Integrated Development Environment (IDE).

Step 0: Getting Set-up Arduino on Computer

****Special Note**** We'll go over this if you want, but in coding we often start counting with the number 0 rather than 1.

- Drivers are pieces of software that allow your computer to speak to your physical device. Think of it like a translator. Without the right drivers installed, your computer won't know how to talk to your device.

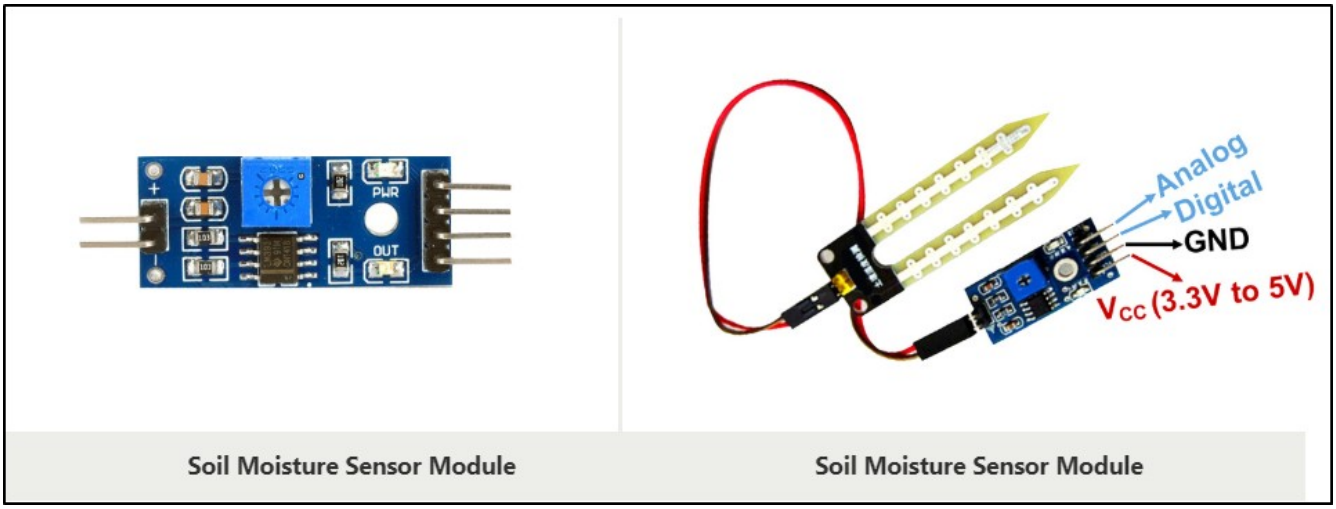
- Integrated Development Environments (IDE) are where you write your code. They make things easy for you because they catch spelling mistakes, color code things of significance, and allow you to "tab things out". "Tab things out" means that you can auto-complete words when you're typing to save you time. **BASICALLY, if you aren't good at spelling, IDEs can make things easier for you**.

- Download Driver and Install IDE Video: https://youtu.be/7-XKZOTyW9Y
- Link to Arduino IDE Download: https://www.arduino.cc/en/software
- Link to Arduino Driver Download: https://cdn.shopify.com/s/files/1/0750/1787/9870/files/WINDOWS-INVENTR-USB-DRIVERS.EXE?v=1717744722

Step 1: The Soil Sensor

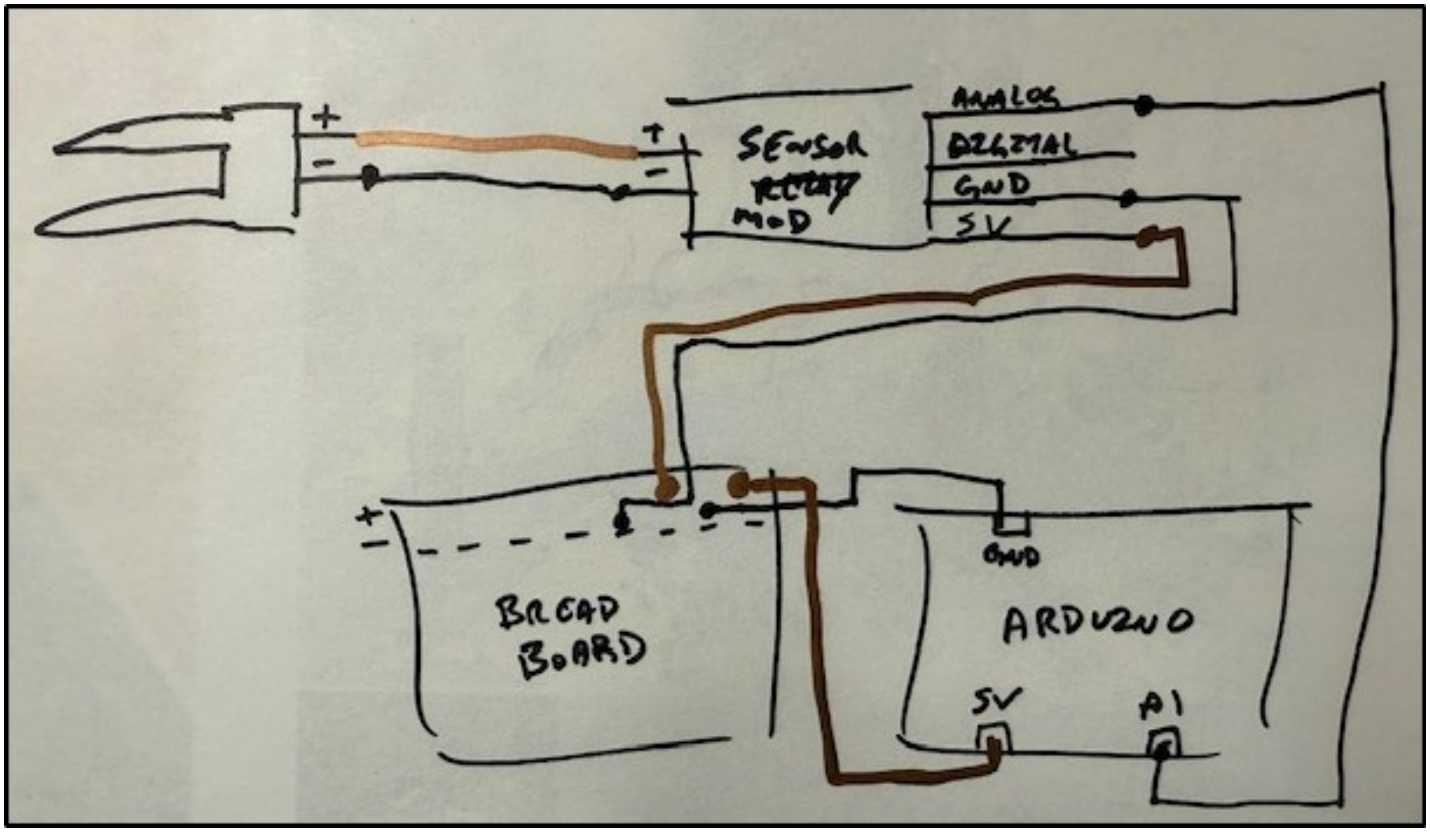
Components:

- Soil Sensor: https://components101.com/modules/soil-moisture-sensor-module



- No water = 1024 (~974 after several weeks no watering)
- All Water = 300-350

Diagram:



Step 2: The Water Pump

Components:

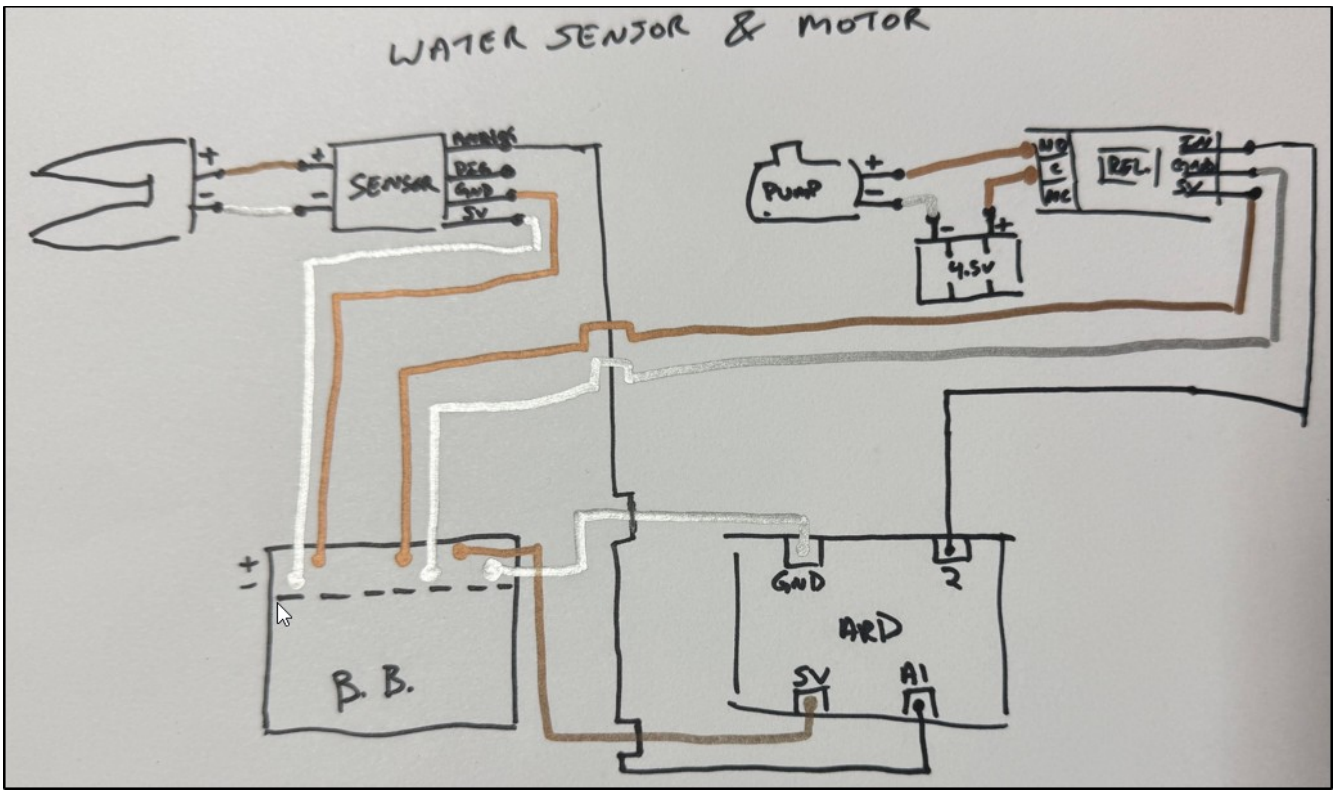
- Water Pump w/ Tube

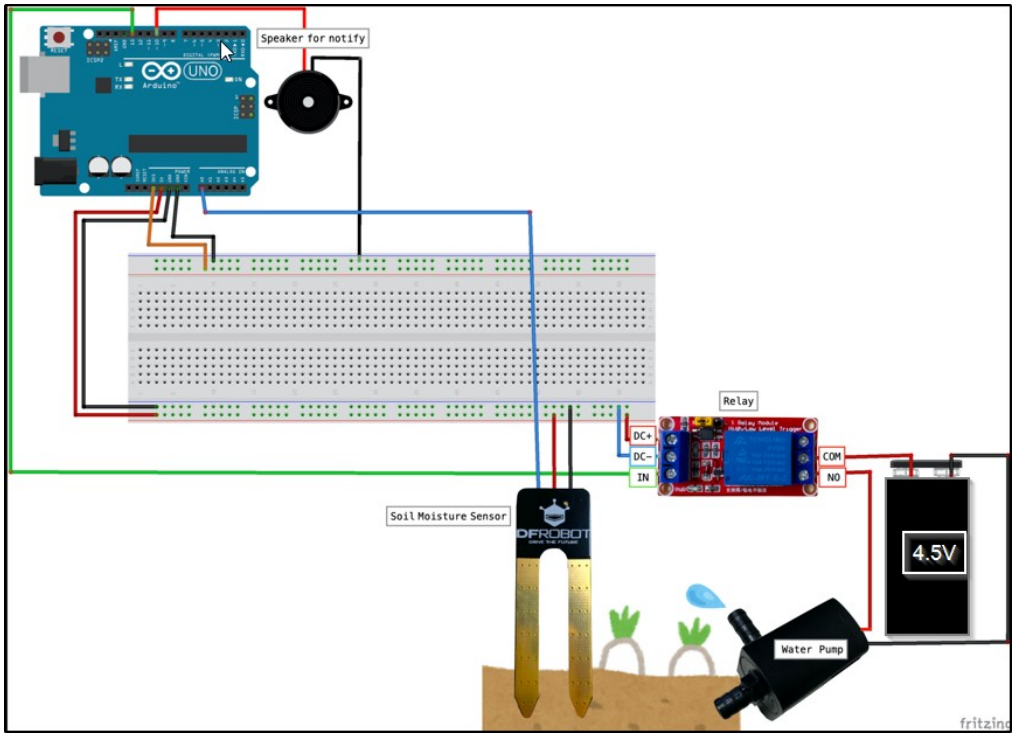
- Relay

- Battery Casing

- 3 AA Batteries

Diagrams:





Why do we need batteries to connect to the water pump?

The water pump requires its own electrical source for the entire system to function properly. Otherwise, when the pump turns on it prevents the sensor from continuing to work and breaks the loop you wrote in your code.

Why do we need the relay?

The relay is what allows us to control the water pump from the Arduino using code. Because the water pump needs the additional power, we need something that can turn that power source on or off.

How do relays work?

https://www.youtube.com/watch?v=ZbBQZV7oljk

https://forum.arduino.cc/t/relay-songle-srd-05vdc-sl-c-10a/574537

Basically, there's constant power going through the relay, but when a certain condition is met the relay switches. It allows you to control greater power sources than what the Arduino offers naturally.

****Special note for coding** The importance of checking sources.**

I spent 5 hours trying to figure out why my code wasn't working. When the pump was supposed to be on it was off and vice versa. This is because I copied code from here: https://projecthub.arduino.cc/lc_lab/automatic-watering-system-for-my-plants-e4c4b9, but the author was using a different relay than ours. Our relay is "ACTIVE LOW" rather than the author's "ACTIVE HIGH" relay.

Step 3: The Ultra-Sonic Sensor (Check Water Levels)

Components:

- Ultrasonic Sensor
  - Information on the sensor: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf
- LED
- Something to Hold the Sensor (I used a piece of cardboard with enough space to hold

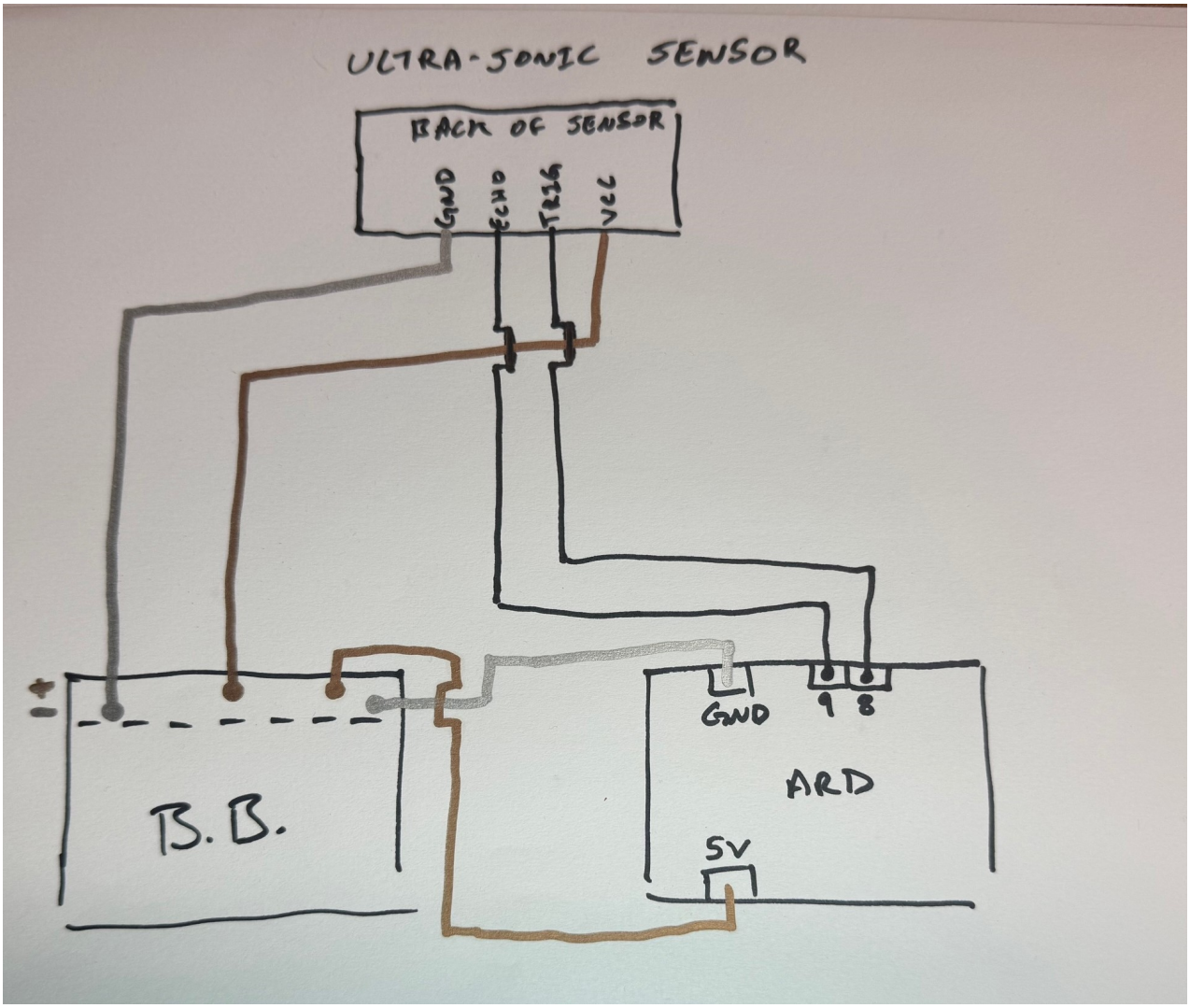
Measurements I used when writing code:

Height of Sensor to bottom of bucket: 14.5 cm

Height I want to refill water at: 5cm

****NOTE**** Your code will change based on your own measurements. Make sure to measure in centimeters.

Diagram:



Step 4: Feedback Components

Feedback is when a device provides an audio or visual clue to the user that something has happened. For this we will you BOTH!

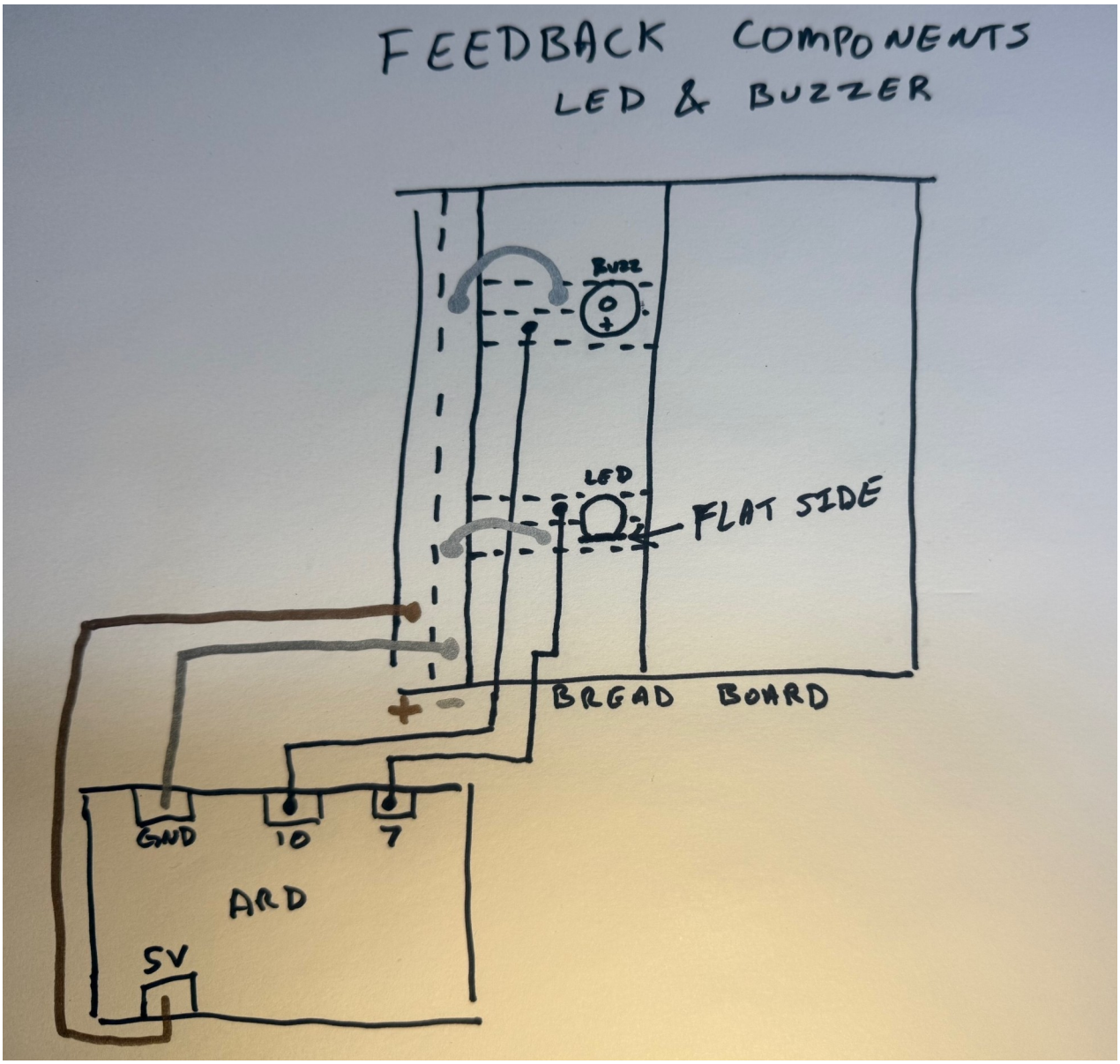
- The LED provides the visual clue in the form of a light.
- The BUZZER provides the audio clue in the form of sound.

Components:

- LED

- Buzzer

Diagram:



The Code

```
//Music
#define BEAT 300
#define PIN 10
#define DO 262
#define RE 294
#define MI 330
#define FA 349
#define SO 392
#define RA 440
#define SI 494
#define HDO 523

//----------------------

//Timer
int counter = 0;

//Ultrasonic Sensor
int trig = 8;
int echo = 9;
int duration;
float distance;

//LED
int led = 7;

//moisture sensor
int water_count = 0;

//relay
int relay = 2;

void setup() {
  //Used for testing
  Serial.begin(9600);


  //Ultrasonic Sensor
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);

  //relay
  pinMode(relay, OUTPUT);
  pinMode(A1, INPUT);
}
```

```
void loop() {
  //Check water resevoir levels and soil humidity once a minute.

  if(counter == 1){
    checkWater();
    checkMoisture();
  }

  delay(10000);//10sec
  counter++;
  Serial.print("Counter is: ");
  Serial.println(counter);

  if(counter >= 6){

    counter = 0;
  }
}

void checkWater(){
  //Check the water level in the bucket.
  digitalWrite(trig, LOW);

  delayMicroseconds(1);
  digitalWrite(trig, HIGH);
  delayMicroseconds(11);

  digitalWrite(trig, LOW);
  duration = pulseIn(echo, HIGH);

  if (duration>0)
  {
    distance = duration/2;
    distance = distance*340*100/1000000; // ultrasonic speed is 340m/s = 34000cm/s = 0.034cm/us

    if(distance > 12){   //This number should be (the height of your sensor to the bottom of the bucket)
minus (the max height of water before its time to refill) in centimeters
      digitalWrite(led, HIGH);
    }else{

      digitalWrite(led, LOW);
    }

  }
}

//moisture sensor
void checkMoisture(){
```

```
  //Measure soil humidity
  int moisture = analogRead(A1);
  //For testing
  Serial.print("Moisture is:");
  Serial.println(analogRead(A1));

  if(moisture >= 700){
    digitalWrite(relay, LOW); //Our relay is "ACTIVE LOW" which means the relay triggers on a low
signal
    delay(5000);
    digitalWrite(relay, HIGH);
    completeWatering();
  }
  else{
    digitalWrite(relay, HIGH);
  }
}

//Music
void completeWatering(){
  //Let them know that watering is complete.
      tone(PIN,DO,BEAT) ; // C
      delay(BEAT) ;
      tone(PIN,RE,BEAT); // D
      delay(BEAT) ;
      tone(PIN,MI,1200) ; // E
      delay(BEAT) ;
      delay(BEAT) ;
      delay(BEAT) ;
      tone(PIN,RE,BEAT) ; //  D
      delay(BEAT) ;
      tone(PIN,DO,BEAT) ; // C
      delay(BEAT) ;
      delay(BEAT) ;
      tone(PIN,DO,BEAT) ; // C
      delay(BEAT) ;
      tone(PIN,RE,BEAT) ; // D
      delay(BEAT) ;
      tone(PIN,MI,BEAT) ; // E
      delay(BEAT) ;
      tone(PIN,RE,BEAT) ; // D
      delay(BEAT) ;
      tone(PIN,DO,BEAT) ; // C
      delay(BEAT) ;
      tone(PIN,RE,1200) ; // D
      delay(BEAT) ;
      delay(BEAT) ;
}
```

Testing the Soil Sensor

Once you have the soil sensor setup you'll hold the sensor in air (which should be all resistance) and in a bucket of water (all conductance). You'll monitor the serial output of the Arduino program to see how this compares to what you have. Use the following code by itself in a new IDE instance and open the serial monitor.

Special Note This is a part of coding. You build concepts in small parts and **TEST** them before moving onto the next step to ensure things are working as expected. Remind me to talk more to you about this.

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  checkMoisture();
}

void checkMoisture(){
   int moisture = analogRead(A1);

  Serial.println("Moisutre Sensor Value:");
  Serial.println(moisture);
  delay(5000);
}
```