Team #2 Documentation

Smart Recycling Attachment

**Table of Contents**

[**Introduction 1**](#_gjdgxs)

[**Materials 1**](#_30j0zll)

[**Instructions 1**](#_1fob9te)

[**Conclusion 1**](#_3znysh7)

[**References 2**](#_2et92p0)

# Introduction

**What is the concept behind your project? What problem are you solving, or what product are you creating? Tell anyone reading this or using the product what they’re getting into.**

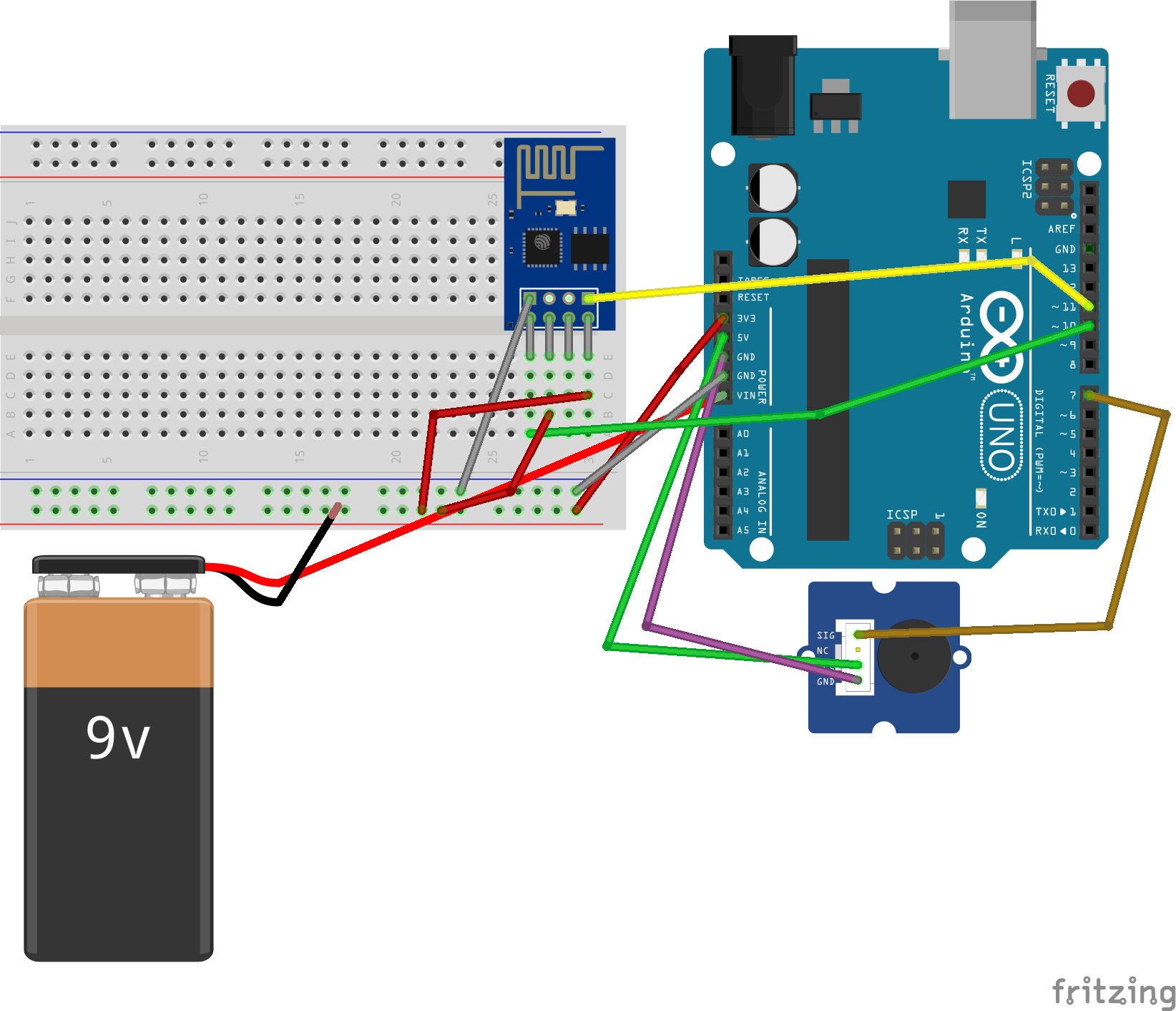
Our product, the Smart Recycling Attachment, is a handy attachment for recycling bins and trash cans alike, and its function is to let the users know how full the bin is by using an ultrasonic ranger to see how much space is left within the bin. The Smart Recycling Attachment is the first step to managing multiple waste bins through wireless connection, so that you could effortlessly know how which bins are close to being full and which need to be emptied. In its current state, it is able to send the sensory data (how much space is left in the bin) to a mobile application Blynk.

The base of the device is the Arduino UNO. The Arduino allows us to manage the connections between the other parts like the ultrasonic ranger and the ESP8266-01, while also allowing us to communicate and relate these parts in program code. Using the ESP WiFi Shield, the Arduino was able to connect to the WiFi as long as the WiFi information was stored in the program code itself and flashed into the Arduino properly. The Grove Ultrasonic Ranger is responsible for sending the sensory information to the Arduino, which is processed and sent to a virtual pin that corresponds to a box in the mobile application.

# Materials

* Arduino UNO
* Grove Ultrasonic Ranger
* Breadboard
* ESP8266-01
* Jumper Wires
* 9v Alkaline Battery + Adapter
* Box

# Instructions

****

This circuit diagram consists of every single wire connection you will need to make, and if you follow this diagram exactly then you will have replicated our project. We have also included three different test files and the final code on our Github. Please refer back to this diagram and our Github across these steps. <https://github.com/kevin-j-chan/qp-trash-can>

Step 1: Getting the Grove Ultrasonic Ranger to work

To connect the Arduino Uno to the Grove Ultrasonic Ranger, you will only have to make three different connections to three of the pins on the Arduino. You have to connect the SIG pin on the ranger to pin 7 on the Arduino, connect VCC on the ranger to 5v on the Arduino, and finally connect GND on the VCC to GND on the Arduino.

To test that the ranger works, you can go to our Github, and upload the test\_ranger.ino program to the Arduino, and the Serial Monitor will print out the distance every second.

Step 2: Getting the ESP8266-01 WiFi Shield to work

To connect the WiFi Shield to the Arduino Uno, you will have to first set up the WiFi shield such that it is able to place itself on the breadboard. To replicate our project exactly, we removed the plastic spacers from the WiFi shield pins, and bent the bins into a shape that will allow it to be directly connected to the breadboard. The picture below shows what the final WiFi shield should look like.



However, we later found another, easier method that includes having male-to-female jumper wires which you can just use to connect the WiFi shield to the breadboard without having to risk bending any pins that we would recommend.

The connections to make for testing the WiFi shield by itself is different from the final connections that were made to use with Blynk. Because we want to test the WiFi shield by itself without Blynk, we can use the Arduino as a bridge to talk to the module directly. Connect the RX pin on the ESP to the RX pin on the Arduino, the TX pin on the ESP to the TX pin on the Arduino, the GND pin on the ESP to the other GND pin on the Arduino that the ultrasonic ranger is not taking up, and both the VCC and the CH\_PD pins on the ESP need to be connected to the 3.3V pin on the Arduino. For these last two pins, you can use the main circuit diagram as a reference to connect to the breadboard.

From there, you can open up a new empty sketch in the Arduino IDE, and upload the empty sketch to the Arduino. Open up the Serial Monitor, and enter in the command AT and an OK will reply if you have set it correctly. Then, run the command AT+CWMODE=1 to set the ESP TO AT and STA mode which will allow you to connect to a network, and finally you are able to connect to a WiFi network by running AT+CWJAP=”WIFI\_NAME”,”WIFI\_PASS”, where WIFI\_NAME and WIFI\_PASS are your network names and password respectively.

Step 3: Getting Blynk to work

Now that the ESP is confirmed to work, you can finally undo some of your connections so that it matches the main diagram above exactly. You can open the test\_esp\_blynk.ino file on our Github to test that it connects to Blynk. Just make sure to set your auth code properly according to your Blynk project, and that the WiFi name and password are also set accordingly as well.

Open the Serial Monitor to confirm that it has connected to Blynk if the message CONNECTED TO WIFI pops up.

One of the largest issues we ran into this stage was that it was constantly giving us the error: “ESP is not responding”. To fix this issue, remember to set the AT commands beforehand to the ESP, and try running the program again.

Step 4: Running the final project

On our Blynk app, we have created two value displays, one value display is set to display the distance, and the other display is used for debugging to check that the Arduino should be updating the distance at that point. The value display for the distance is set to virtual pin 4, and the other value display is set to virtual pin 5. Make sure that the frequencies for both of these displays is set to PUSH.

Then, you can upload our trash\_can.ino file from our Github to get the final code for our project, and the behavior should be the exact same as our main project. If you want to run your project without connecting the Arduino to the computer, then you can add a 9V battery to your project like we have described in our main circuit diagram.

# Conclusion

In the end, our project was able to be developed to its most basic functioning state, and can be improved with more time. In its most basic functioning, the unit can take in sensory data and send the data to the mobile application Blynk. What this means is that our recycling bin attachment can see how much space is left (in inches) in the recycling bin.

Given more time, improvements would be like adding notifications when the space is running lower and lower and more stability within the device itself. Furthermore, things we would change is that we would add an LED display that would produce output from the serial so we are not running the project completely in the blind. Improvements that could be made would be to fix the instability of the WiFi connection as it constantly disconnects from the Blynk server, as well as taking a very long time before it is ready to write to the virtual pins. We could also figure out a better design that would allow you to replace the battery more easily.

Our greatest obstacle came around week 7, when we realized that the only thing we had accomplished was to add the ultrasonic ranger onto the Arduino. None of us had any experience with working on electronics or hardware, so it was very doubtful that we would finish the project. When we tried soldering the WiFi shield onto our arduino, we failed miserably and were almost about to cancel the project.

Our proudest moment was when we finally got the WiFi shield to locate all of the public networks available nearby and to actually connect to one successfully. We had spent a solid week and a half making what seemed to be zero progress on what is the most crucial part of an IoT project, and having it finally connect to something made the project seem doable from that point on.

# References

<https://medium.com/@cgrant/using-the-esp8266-wifi-module-with-arduino-uno-publishing-to-thingspeak-99fc77122e82>

This link helped us greatly in figure out what to do to check that our WiFi module was working by itself without having to use Blynk.

<http://www.instructables.com/id/Making-ESP8266-01-module-breadboard-friendly/>

This link helped us place the WiFi shield into the breadboard in a manner that did not require us to solder.

<https://www.sciencebuddies.org/science-fair-projects/references/how-to-use-a-breadboard>

This link taught us about how breadboards work, and the circuit diagrams would not be possible without the knowledge from this tutorial.