This project is my entry into the Instructables Wireless contest. It meets the criteria of the contest due to the use of WiFi by the ESP8266 to obtain time and weather data from the Internet.

The project is my take on the ever popular Weather Station. Mine is based on an ESP8266, a .96" OLED display and a BME280 environmental sensor array. Weather Stations seem to be a very popular project. Mine differentiates itself from the others by using a BME280 sensor array instead of the popular DHT22 temperature and humidity sensor. The BME280 has a temperature, humidity and air pressure sensor. It also uses the I2C interface. The .96" OLED display used is also I2C. It can be purchased as either I2C or SPI or both. I went with the I2C version to simplify the wiring. With both the OLED display and the BME280 using I2C and 3.3V is was very easy to make a 'Y' cable to connect both devices to the ESP8266. While developing this project I came across multiple weather station projects on the Internet that use the ESP8266, the same OLED display and the BME280. So this is not an original idea, but it is an original implementation.

The BME280 provides inside environment data. Outside weather information is obtained from *OpenWeatherMap.org*. You will need to signup with *OpenWeatherMap.org* to get a key to access the weather data. They offer a free service, which is what I used. See How to get an *OpenWeatherMap ID.pdf* for instructions on how to obtain a key.

An NTP time server is used to get the time of day and day-of-the-week.

The weather, time and environment data are displayed on the OLED display. Each piece of information has it's own formatted screen. The screens are displayed for five seconds before switching to another. *OpenWeatherMap.org* is accessed every fifteen minutes to refresh the weather information. The BME280 is read about every fifty-five seconds.

The ESP8266 is also setup to be a web server. All of the weather information can be accessed using a browser from your phone, tablet of computer. One of the screens that is displayed shows the IP address of the web server.

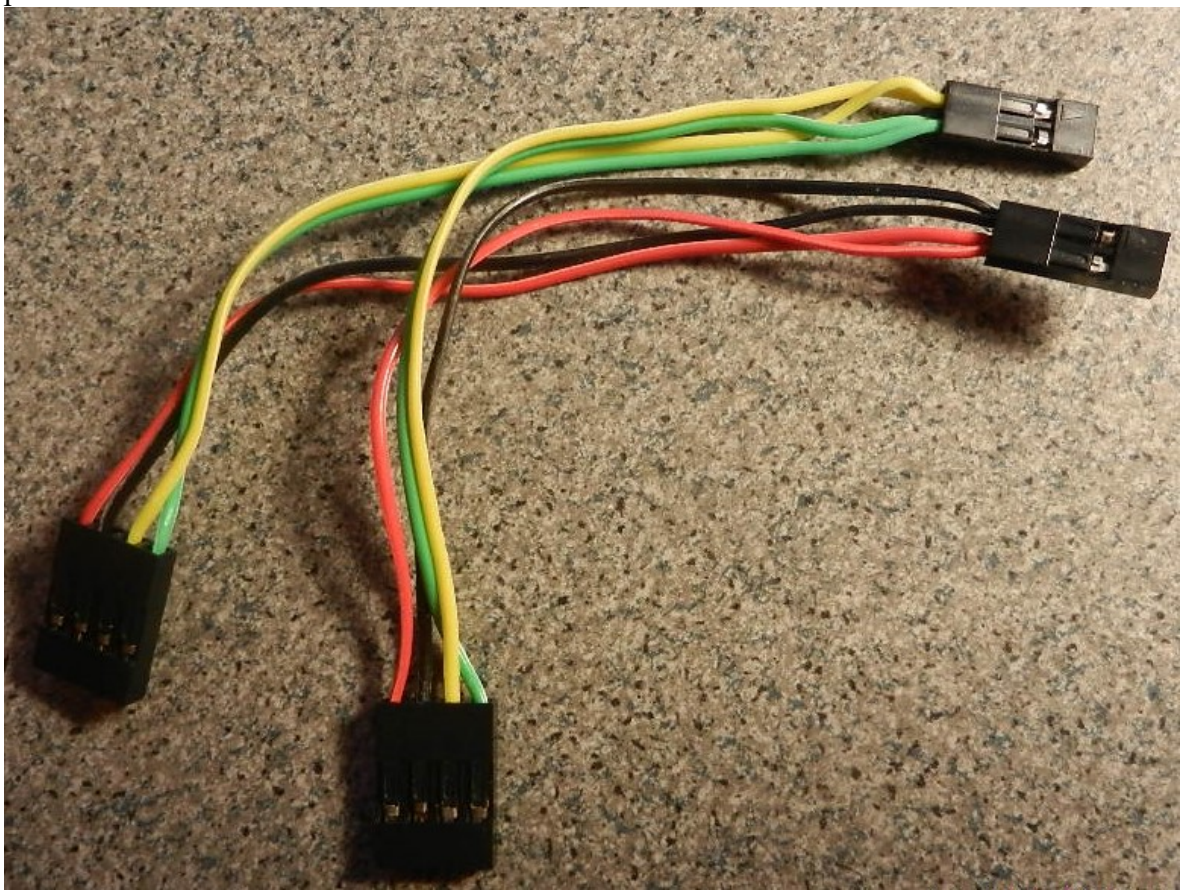
The ESP8266 comes in a variety of shapes and sizes. I chose to use one I bought online. It is a GEEKCREIT DoIt ESP12E Dev Kit V2. This one is fully compatible with the NodeMCU 'standard' for ESP8266 standalone modules. It has an integrated 3.3V regulator, a CH340 as the USB-to-Serial bridge and the NodeMCU auto-reset circuit. You are free to use any ESP8266-12 module that you have. Just be aware that you may have to add a 3.3V regulator or other circuits to program it. I also built one using a Witty Cloud ESP8266. It allowed me to pack everything into a 1.5 inch cube.

Program development was done using the Arduino IDE Version 1.8.0. You can download the latest Arduino IDE here; https://github.com/esp8266/Arduino. The Arduino web site has excellent directions on how to install and use the IDE. Support for the ESP8266 can be installed in the Arduino IDE by following the instructions given by this link: https://github.com/esp8266/Arduino. On the web page, click the "***Clone or Download***"

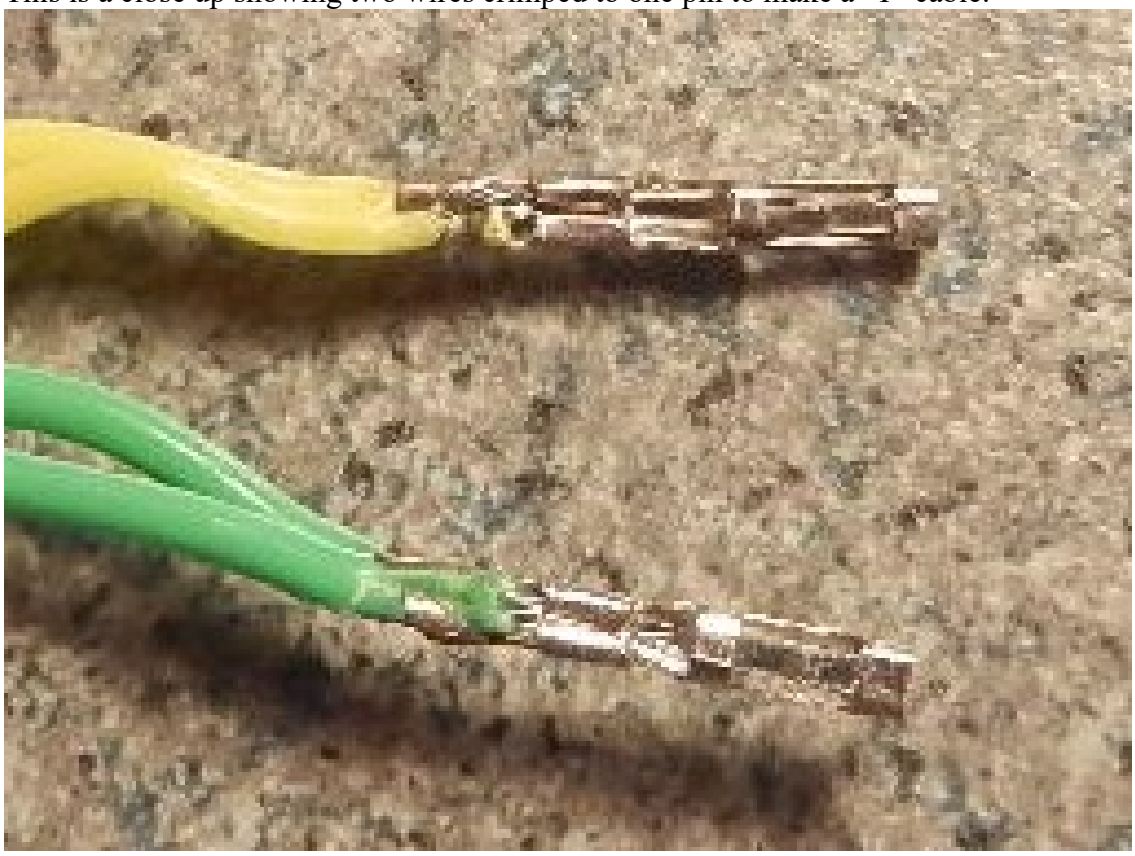
button and select "***Download Zip***". The *ReadMe.md* file has directions on how to add the ESP8266 support to the Arduino IDE. It is a plain text file that you can open with any text editor.

The ESP8266 does not have dedicated I2C hardware. All I2C drivers for the ESP8266 are based on bit-banging. One of the better ESP8266 I2C libraries is the brzo_I2C library. It was written in assembly language for the ESP8266 to make it as fast as possible. The OLED display library I am using uses the brzo_I2C library. I added code to access the BME280 sensor array using the brzo_I2C library. There is no #include for the brzo_I2C library in my file due to it being included in the OLED library.
You can get the OLED library here: https://github.com/squix78/esp8266-oled-ssd1306
You can get the brzo_I2C library here: https://github.com/pasko-zh/brzo_i2c
Both libraries will need to be installed in your Arduino IDE. The Arduino website has directions on how to install zip libraries into the IDE here: https://www.arduino.cc/en/Guide/Libraries.

The wiring harness is the key to this project. It is a basic four wire 'Y' cable. Here is a picture of the harness I made.

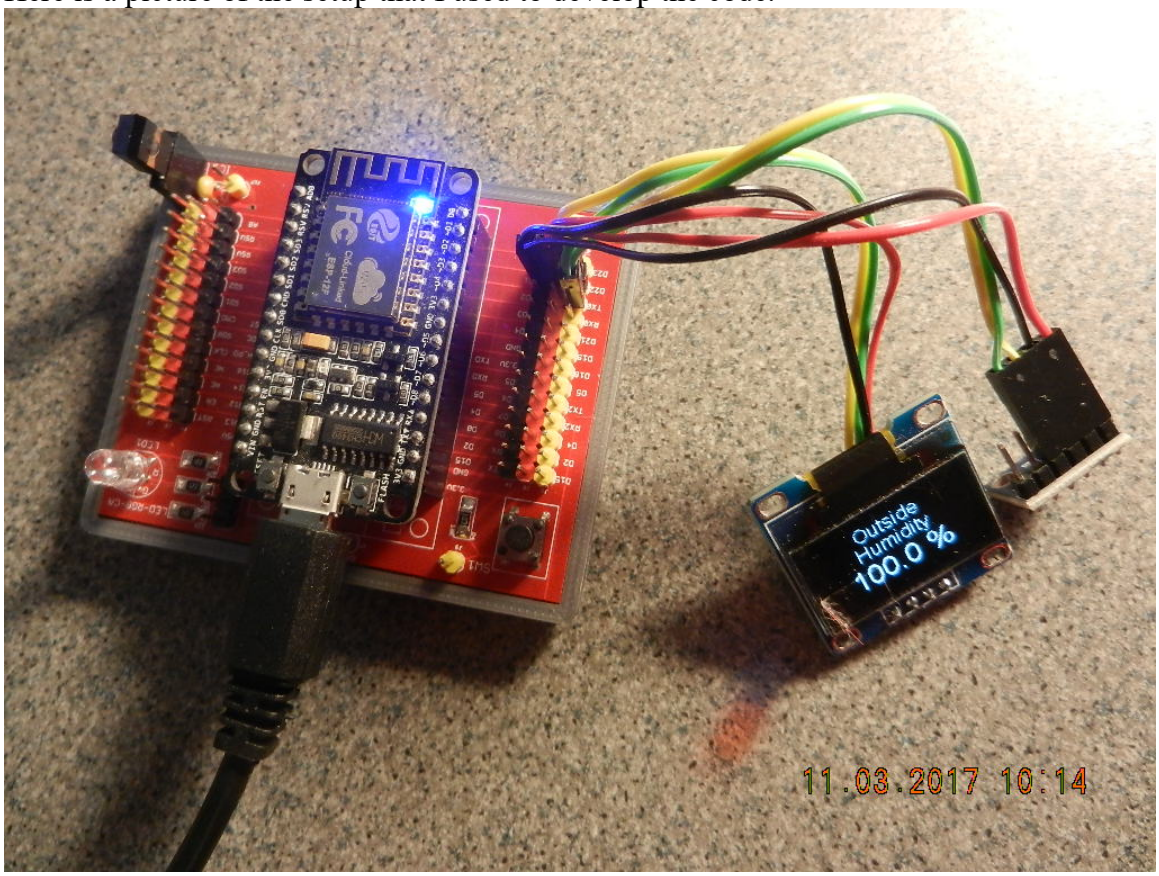


This is a close up showing two wires crimped to one pin to make a ‘Y’ cable.



Strip the wires longer than normal, twist the exposed wires together, then trim the exposed wires to the proper length for the crimp and crimp them.

Here is a picture of the setup that I used to develop the code.



The board that the ESP8266 module is plugged into is a circuit board that I developed as a breakout board for the ESP8266 and ESP32. It will accept the NodeMCU compatible, narrow body ESP8266 boards, The Witty Cloud ESP8266 board or an ESP32 board from GEEKCREIT. All of the available GPIO pins are broken out to headers for easy access. I have found that most development boards never have enough power and ground pins. Every time you want to attach something you need at least a ground pin and most times a pin to power the device. Each row of GPIO pins is accompanied by 3.3V power pin and a ground pin. I use the same layout that First Robotics uses, power in the middle.. I like this layout because if you plug something in backwards you do not release the magic smoke. The board has a couple of extras, an IR sensor, a pushbutton switch and a tri-color LED. Jumpers can be used to connect to any of these features. If you are interested in one of these ESPxx breakout boards then contact me.

All of the source code and other files are available on GitHub here;
https://github.com/BigJBehr/ESP8266-Weather-Station.

I have included a pdf of a diagram of the wiring harness and bill of materials.

I hope you enjoy my take on a Weather Station.