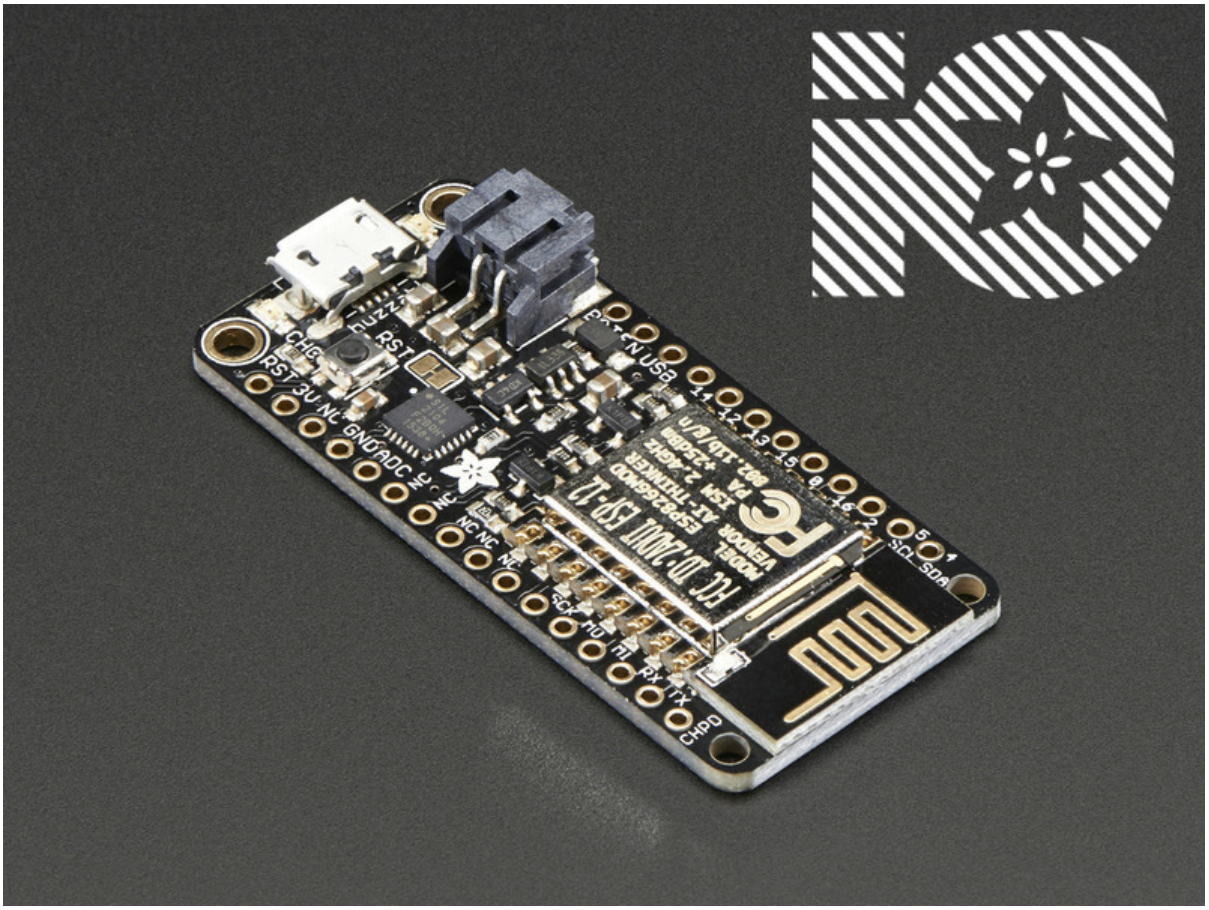
Adafruit IO Basics: ESP8266 + Arduino

Created by Todd Treece

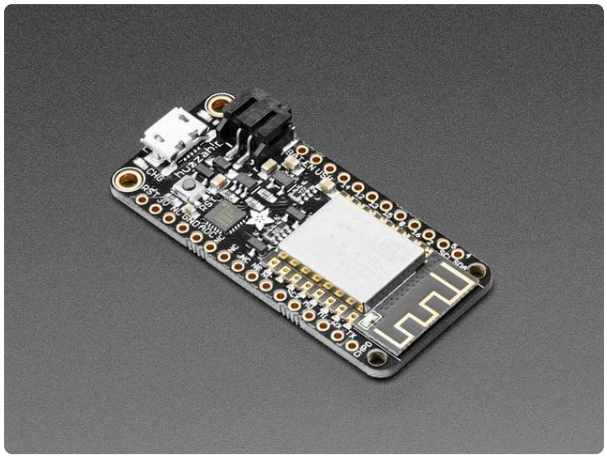


https://learn.adafruit.com/adafruit-io-basics-esp8266-arduino

Last updated on 2024-03-08 02:20:24 PM EST

Table of Contents

Overview



Adafruit Feather HUZZAH with ESP8266 - Loose Headers
Feather is the new development board from Adafruit, and like its namesake, it is thin, light, and lets you fly! We designed Feather to be a new standard for portable microcontroller...
https://www.adafruit.com/product/2821

The ESP8266 based Feather HUZZAH & the HUZZAH ESP8266 breakout are both very popular options for connecting projects to Adafruit IO. In this guide we are going to walk through the setup needed to get your ESP8266 up and running with the Arduino IDE & Adafruit IO. This same basic setup can be used as you progress through our Adafruit IO Basics series of guides.

Before you continue with this guide, you should consider running through the guides for the ESP8266 Feather or the ESP8266 breakout. We will cover all of the basic setup needed for connecting your ESP8266 to Adafruit IO, but the individual guides go into greater detail about each board.

- Adafruit HUZZAH ESP8266 breakout
- Adafruit Feather HUZZAH ESP8266

Pros/Cons of the ESP8266

Here are some quick pros & cons if you are considering using the ESP8266 for your Adafruit IO project.

Pros

- Low cost
- Great support via the ESP8266 Arduino community
- Can be programmed using Lua & Python (MicroPython), in addition to Arduino
- Fast Uploads

Cons

- Power hungry
- Limited number of GPIO pins
- One analog input pin

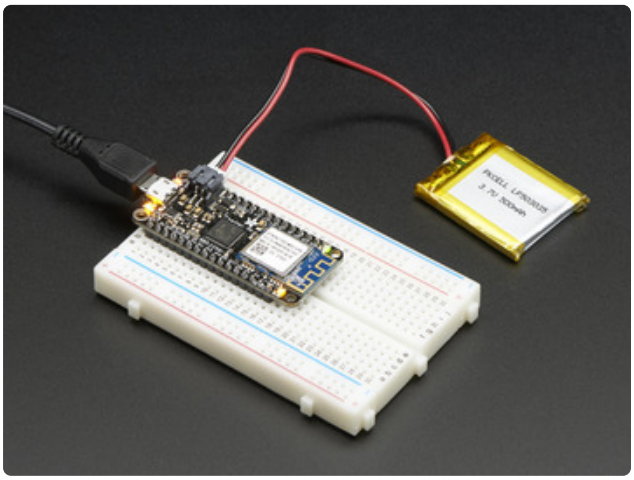
Lets get started with assembly.

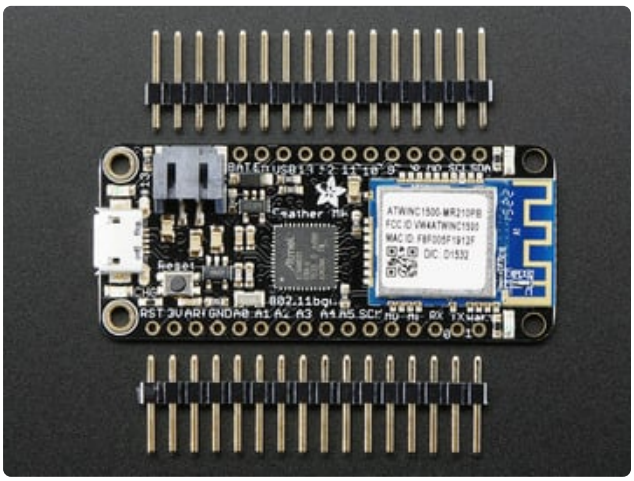
Assembly

We ship Feathers fully tested but without headers attached - this gives you the most flexibility on choosing how to use and configure your Feather

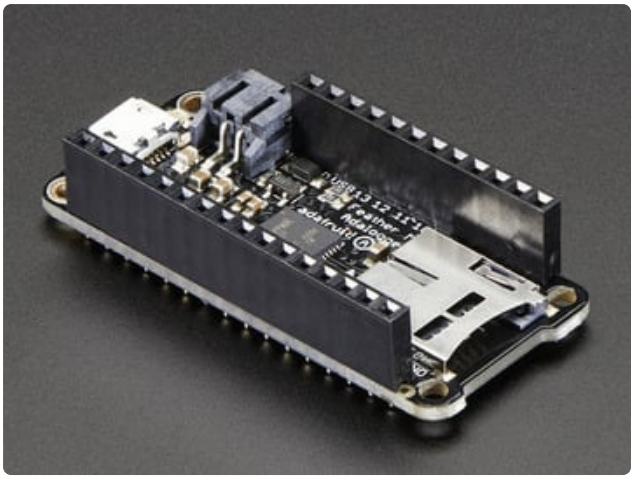
Header Options!

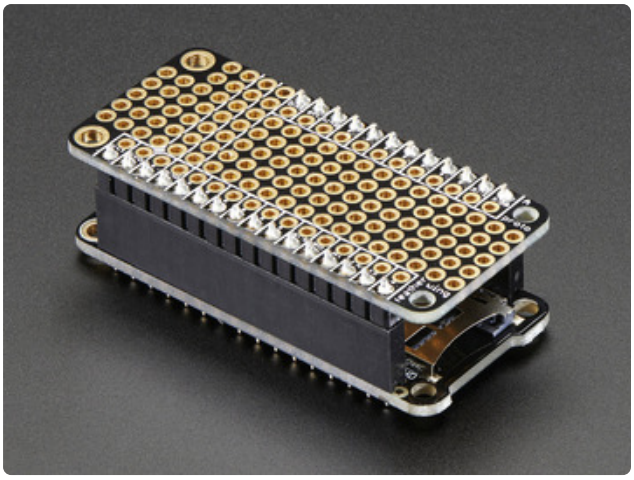
Before you go gung-ho on soldering, there's a few options to consider!

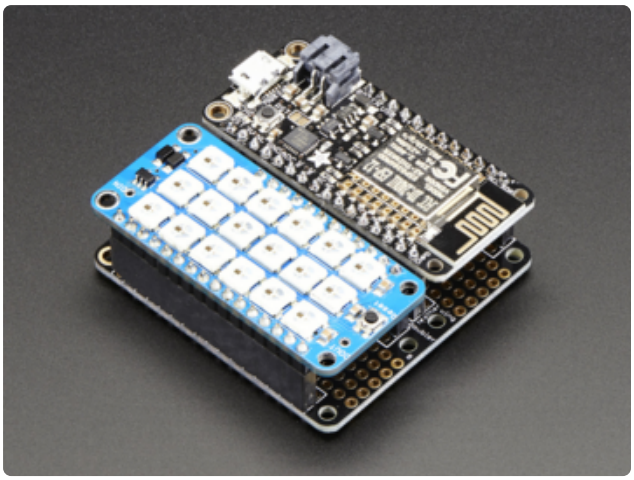




The first option is soldering in plain male headers, this lets you plug in the Feather into a solderless breadboard

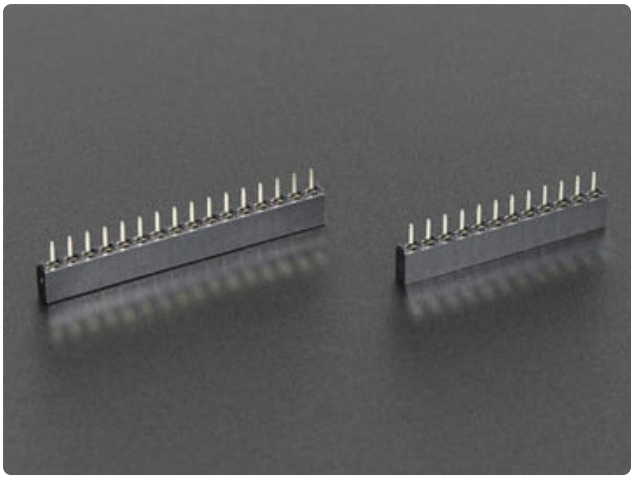


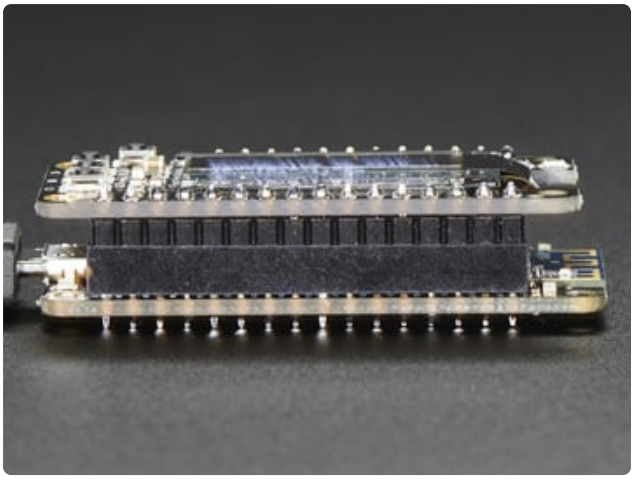




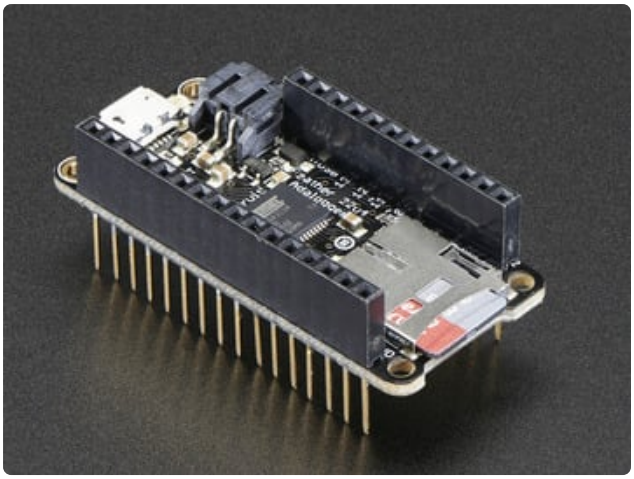
Another option is to go with socket female headers. This won't let you plug the Feather into a breadboard but it will let you attach featherwings very easily

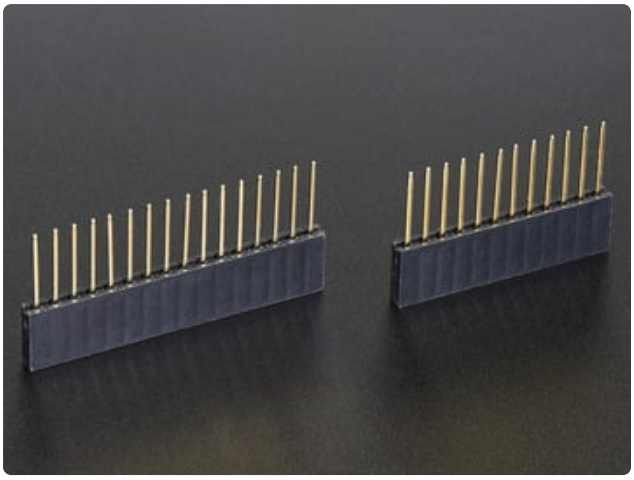
A few Feather boards require access to top-side components like buttons or connectors, making stacking impractical. Sometimes you can stack in the opposite order—FeatherWing underneath—or, if both Feather and Wing require top-side access, place the boards side-by-side with a FeatherWing Doubler (http://adafru.it/2890) or Tripler (http://adafru.it/3417).





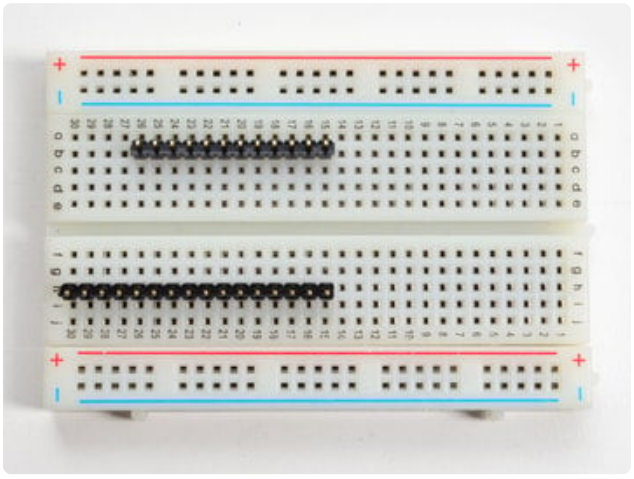
We also have 'slim' versions of the female headers, that are a little shorter and give a more compact shape





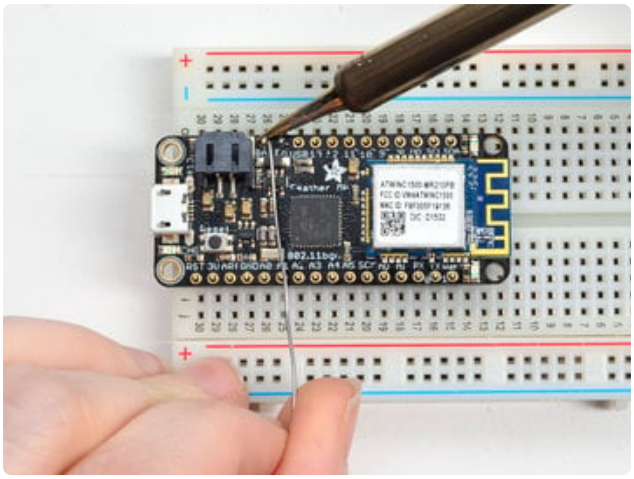
Finally, there's the "Stacking Header" option. This one is sort of the best-of-both-worlds. You get the ability to plug into a solderless breadboard and plug a featherwing on top. But its a little bulky

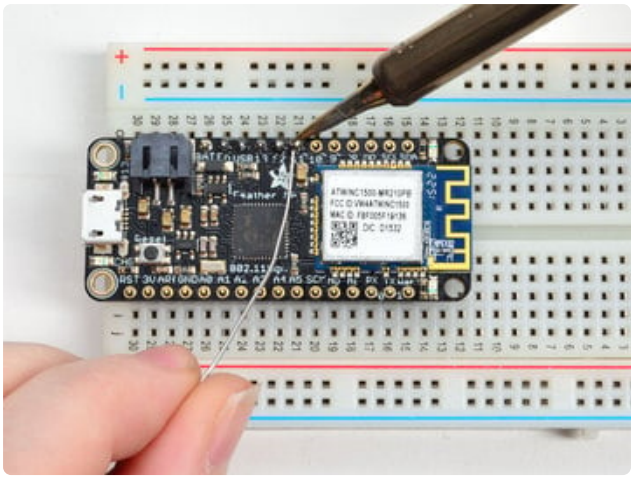
Soldering in Plain Headers

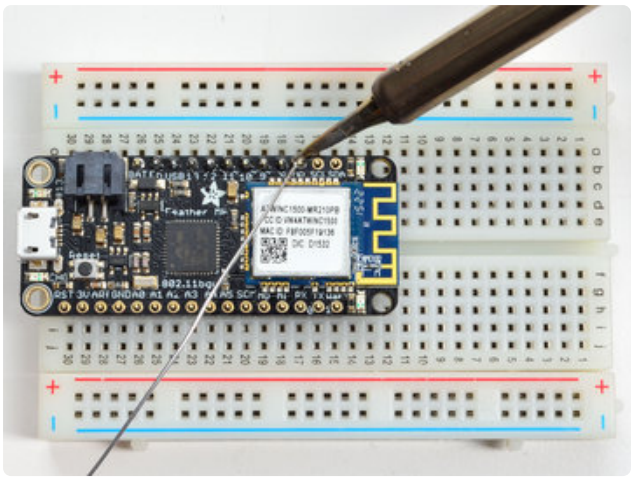


Prepare the header strip:

Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - **long pins down**







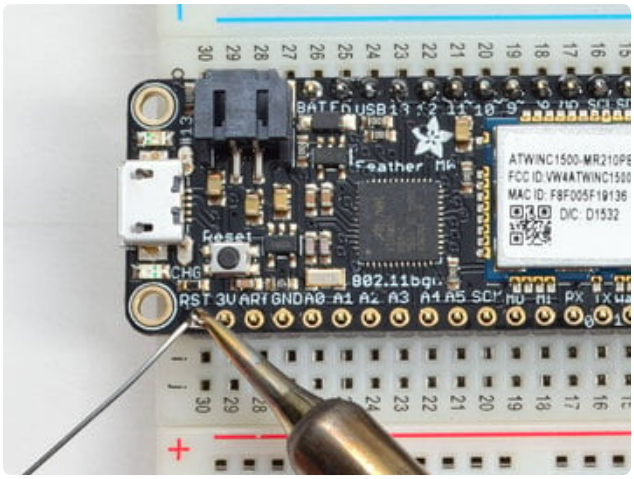
Add the breakout board:

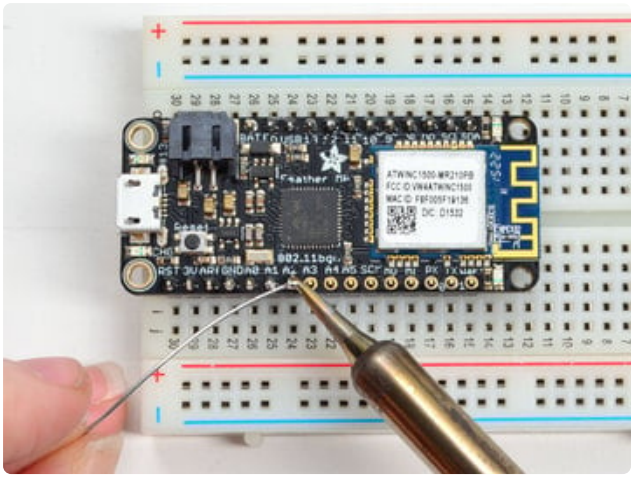
Place the breakout board over the pins so that the short pins poke through the breakout pads

And Solder!

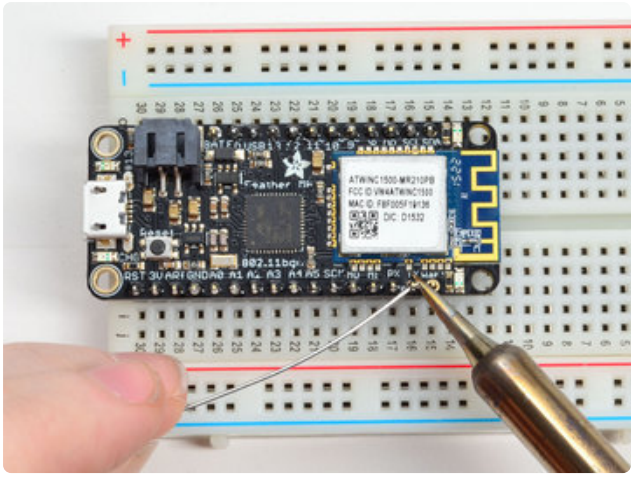
Be sure to solder all pins for reliable electrical contact.

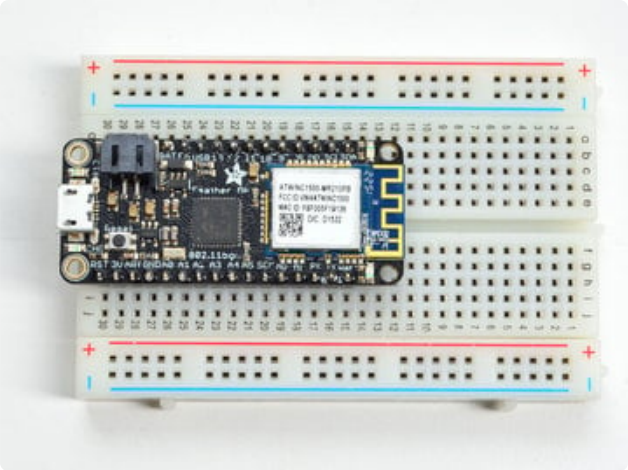
(For tips on soldering, be sure to check out our Guide to Excellent Soldering (https://adafru.it/aTk)).





Solder the other strip as well.



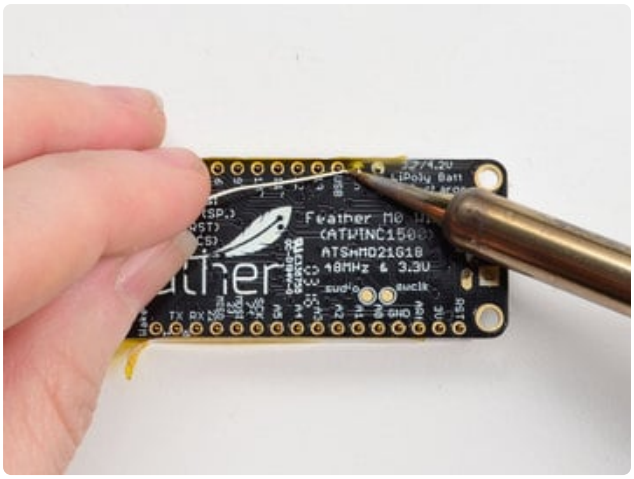


You're done! Check your solder joints visually and continue onto the next steps

Soldering on Female Header

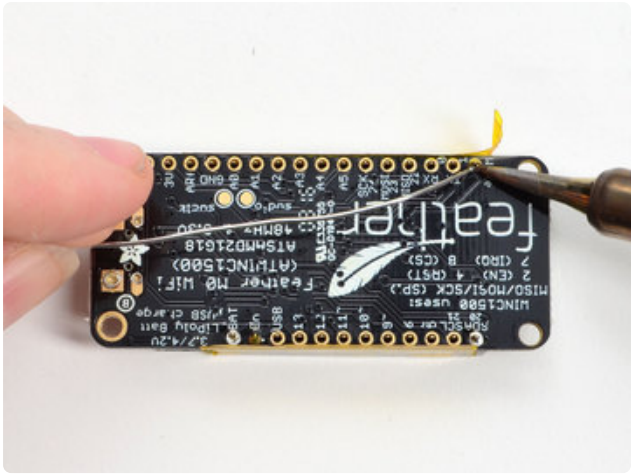
Tape In Place

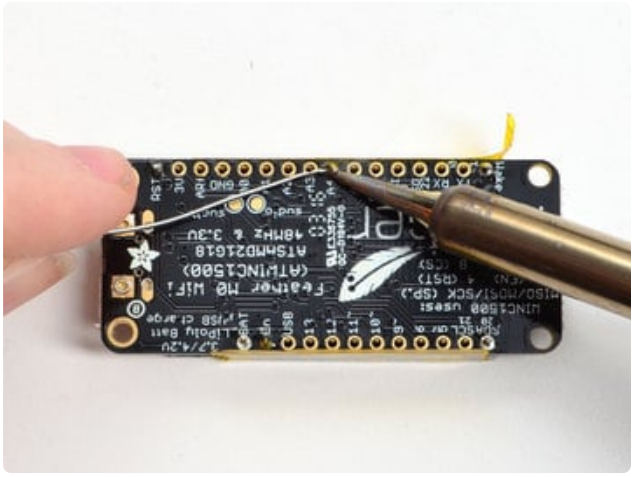
For sockets you'll want to tape them in place so when you flip over the board they don't fall out

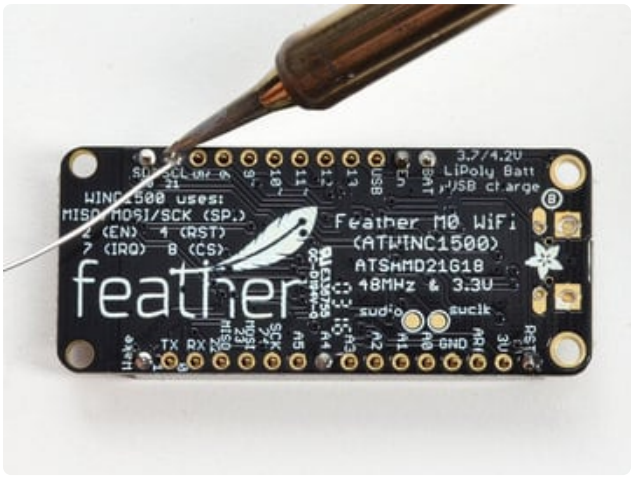


Flip & Tack Solder

After flipping over, solder one or two points on each strip, to 'tack' the header in place







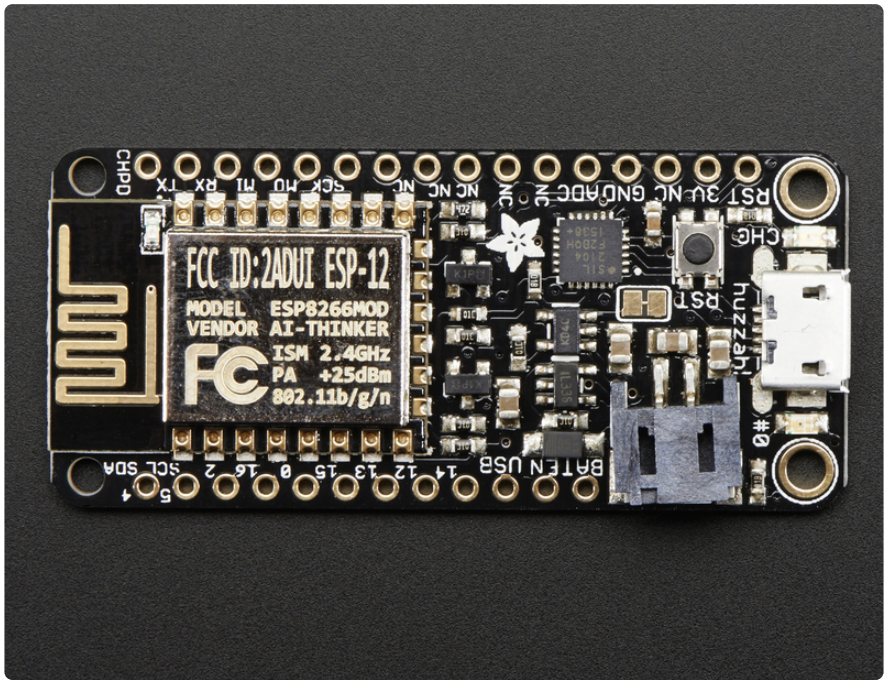
And Solder!

Be sure to solder all pins for reliable electrical contact.

(For tips on soldering, be sure to check out our Guide to Excellent Soldering (https://adafru.it/aTk)).

You're done! Check your solder joints visually and continue onto the next steps

Using Arduino IDE



While the Feather HUZZAH ESP8266 comes pre-programmed with NodeMCU's Lua interpretter, you don't have to use it! Instead, you can use the Arduino IDE which may

be more familar. **This will write directly to the firmware, erasing the NodeMCU firmware**, so if you want to go back to Lua, use the flasher to re-install it (https://adafru.it/f1O)

Don't forget to visit esp8266.com for the latest and greatest in ESP8266 news, software and gossip! (https://adafru.it/f1F)

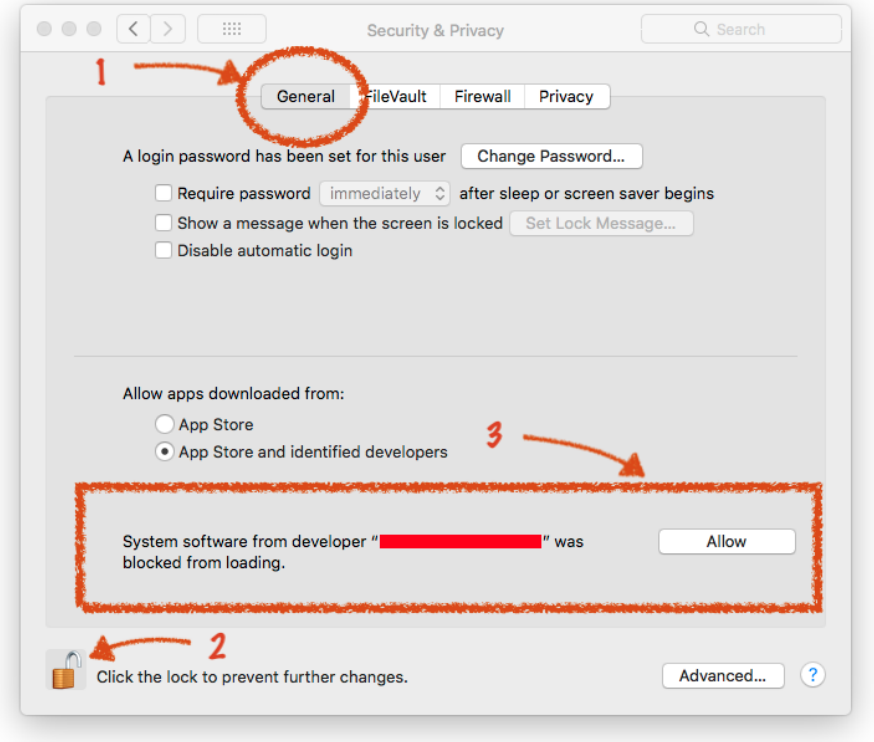
In order to upload code to the ESP8266 and use the serial console, connect any data-capable micro USB cable to the Feather HUZZAH and the other side to your computer's USB port.

Don't forget you will also need to install the SiLabs CP2104 Driver:

Click here to download the CP2104 USB Driver

https://adafru.it/vrf

On Mac OS 10.13 and higher, in addition to installing, you will have to give the CP2104 kernel driver permission to load. You can find out if you need to give additional permission by visiting your Security & Privacy settings system preferences screen after installing and looking for the message that says, 'System software from developer "SiLabs" was blocked from loading', like in the picture below.



To allow the driver to load, click the lock icon, enter your password, and click "Allow" next to the warning message. After that, you may have to restart your computer before following the steps below and connecting to your Huzzah in the Arduino app.

If you are using Mac OS 10.12.6 (Sierra) and you cannot upload with the latest Mac OS VCP driver, please try the legacy v4 driver below. Note you will need to uninstall the v5 driver using uninstall.sh (in the driver package)

Download the CP2104 Legacy USB Driver

https://adafru.it/ymF

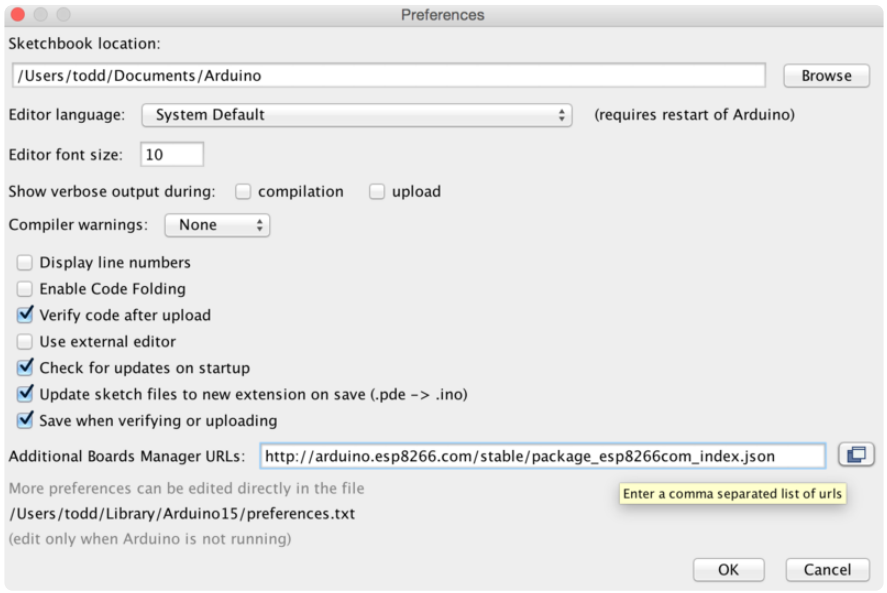
Install the Arduino IDE 1.6.8 or greater

Download Arduino IDE from Arduino.cc (1.6.8 or greater) (https://adafru.it/f1P) from Arduino.cc

The latest is usually the best

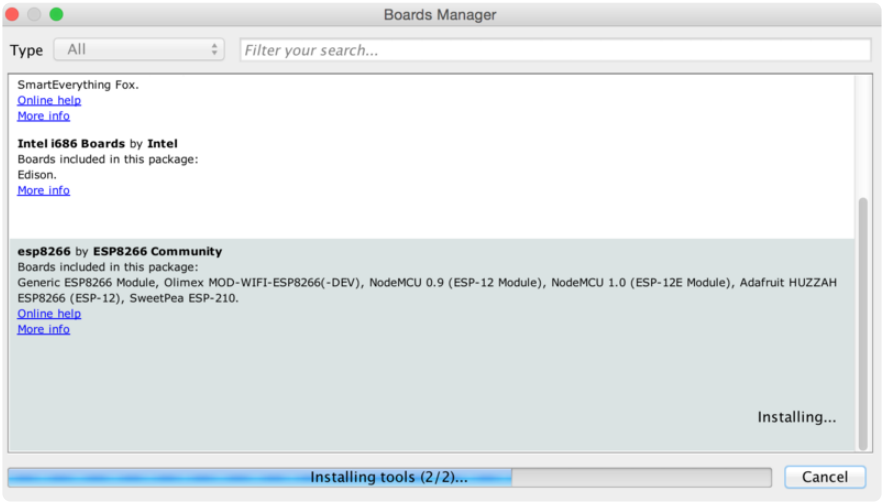
Install the ESP8266 Board Package

Enter `http://arduino.esp8266.com/stable/package_esp8266com_index.json` into Additional Board Manager URLs field in the Arduino v1.6.4+ preferences.



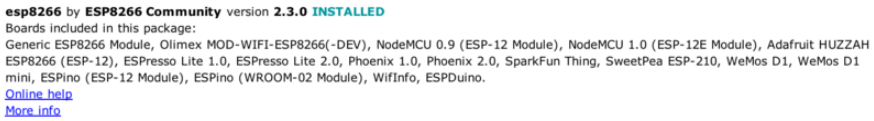
Visit our guide for how to add **new boards to the Arduino 1.6.4+ IDE** for more info about adding third party boards (https://adafru.it/f7X).

Next, use the **Board manager** to install the ESP8266 package.



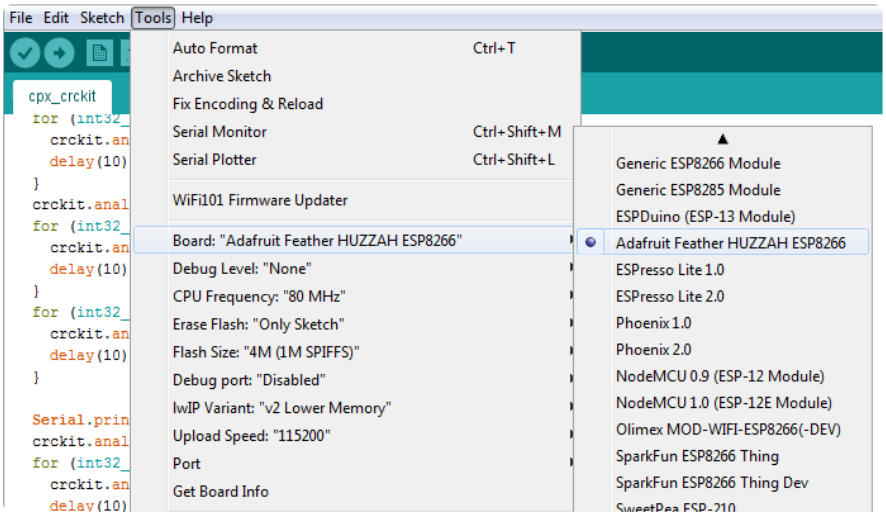
If you want to use this board with Adafruit IO Arduino - make sure you're on version 2.5.1 or ABOVE.

After the install process, you should see that esp8266 package is marked INSTALLED. Close the Boards Manager window once the install process has completed.

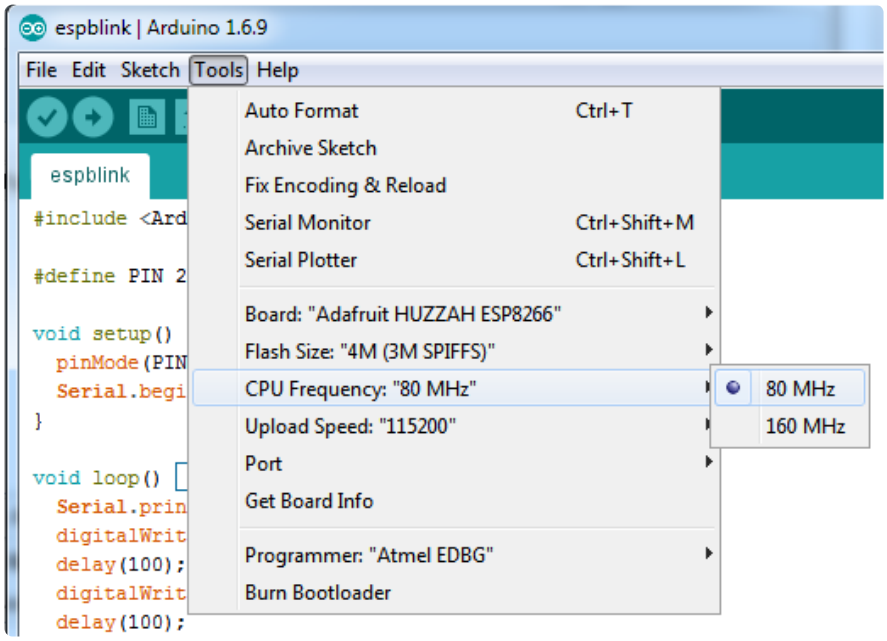


Setup ESP8266 Support

When you've restarted, select **Adafruit Feather HUZZAH ESP8266** from the Tools->Board dropdown

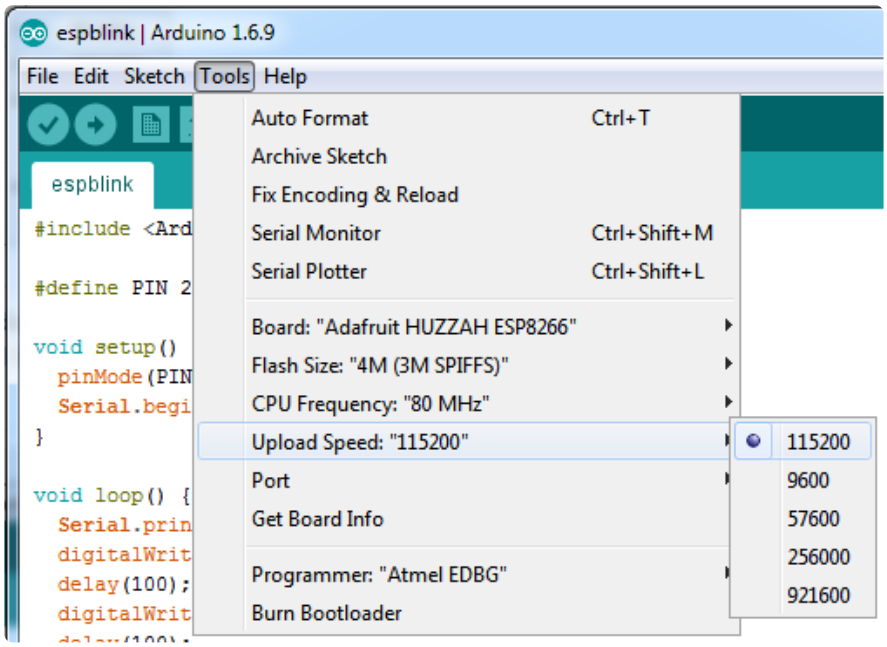


80 MHz as the CPU frequency

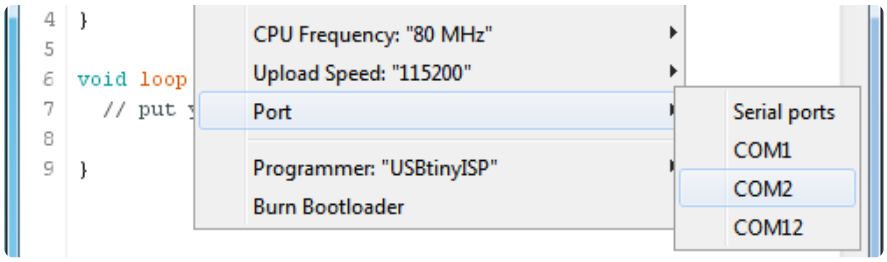


You can keep the **Flash Size**at "4M (3M SPIFFS)

For **Upload Speed**, select 115200 baud (You can also try faster baud rates, we were able to upload at a blistering 921600 baud but sometimes it fails & you have to retry)



The matching COM port for your FTDI or USB-Serial cable



On a mac, you should look for the "SLAB_USBtoUART" port

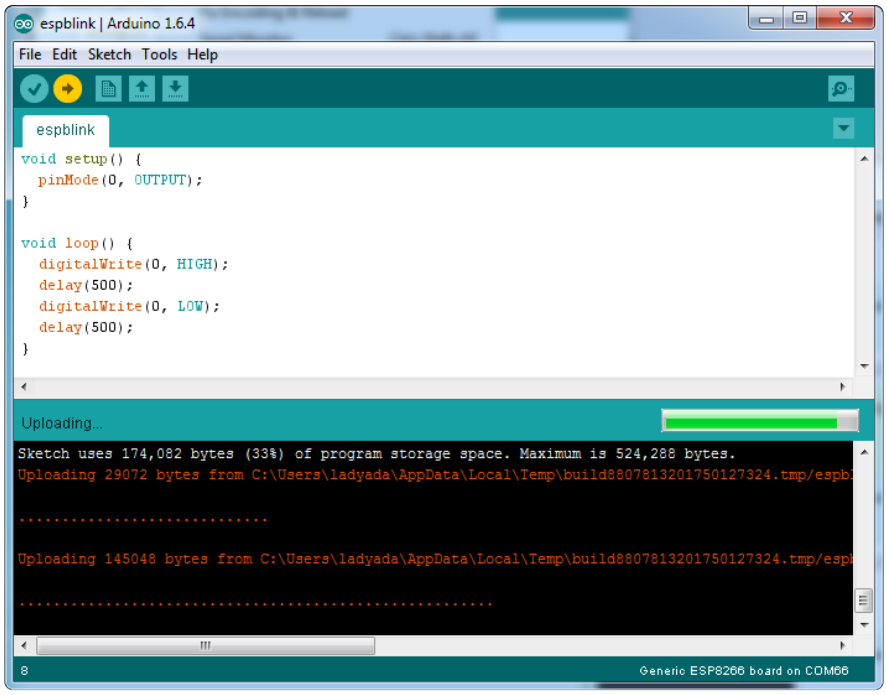
Blink Test

We'll begin with the simple blink test

Enter this into the sketch window (and save since you'll have to)

```
void setup() {
  pinMode(0, OUTPUT);
}

void loop() {
  digitalWrite(0, HIGH);
  delay(500);
  digitalWrite(0, LOW);
  delay(500);
}
```

Now you can simply upload! The **Feather HUZZAH** has built in auto-reset that puts it into bootloading mode automagically



The sketch will start immediately - you'll see the LED blinking. Hooray!

Connecting via WiFi

OK once you've got the LED blinking, lets go straight to the fun part, connecting to a webserver. Create a new sketch with this code:

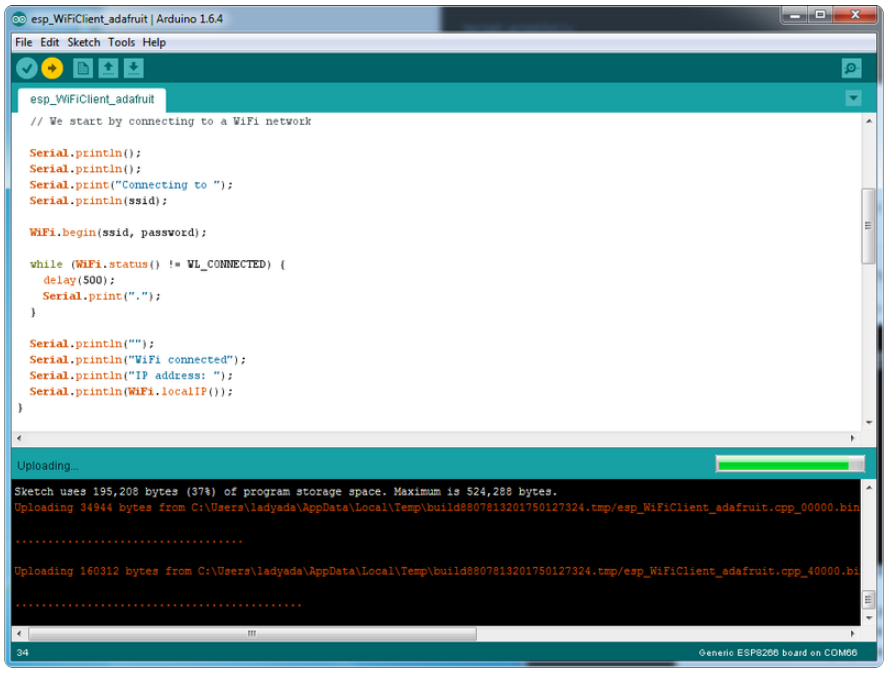
```
/*
 *  Simple HTTP get webclient test
 */

#include &lt;ESP8266WiFi.h&gt;

const char* ssid     = "yourssid";
const char* password = "yourpassword";

const char* host = "wifitest.adafruit.com";

void setup() {
  Serial.begin(115200);
  delay(100);

  // We start by connecting to a WiFi network

  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
```

```
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

int value = 0;

void loop() {
  delay(5000);
  ++value;

  Serial.print("connecting to ");
  Serial.println(host);

  // Use WiFiClient class to create TCP connections
  WiFiClient client;
  const int httpPort = 80;
  if (!client.connect(host, httpPort)) {
    Serial.println("connection failed");
    return;
  }

  // We now create a URI for the request
  String url = "/testwifi/index.html";
  Serial.print("Requesting URL: ");
  Serial.println(url);

  // This will send the request to the server
  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");
  delay(500);

  // Read all the lines of the reply from server and print them to Serial
  while(client.available()){
    String line = client.readStringUntil('\r');
    Serial.print(line);
  }

  Serial.println();
  Serial.println("closing connection");
}
```

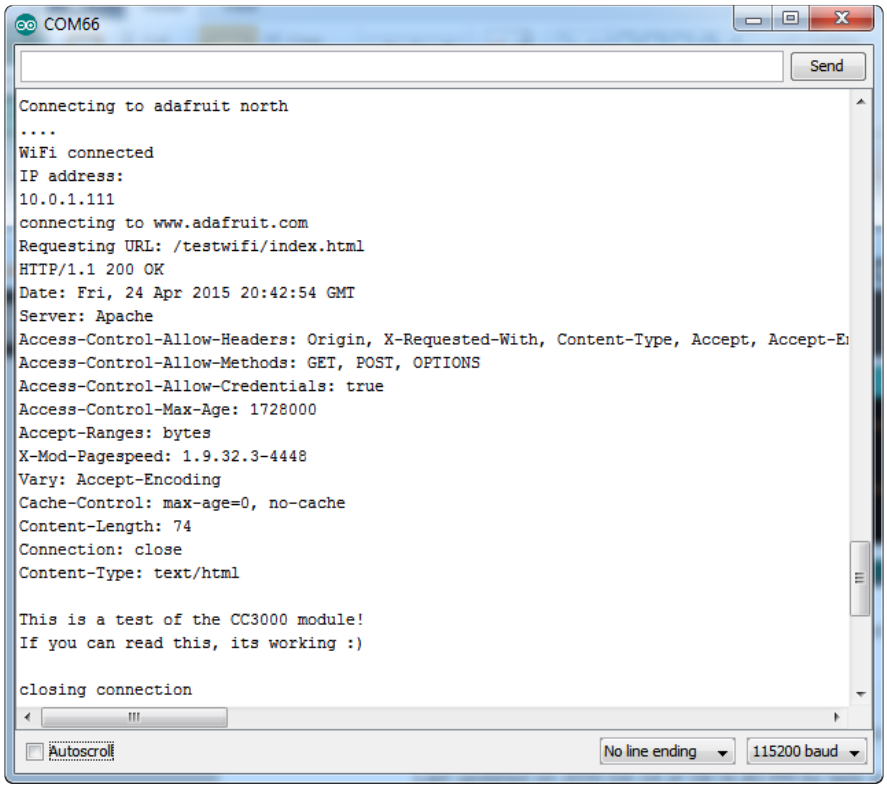
Dont forget to update

```
const char* ssid     = "yourssid";
const char* password = "yourpassword";
```

to your access point and password, then upload the same way: get into bootload mode, then upload code via IDE



Open up the IDE serial console at 115200 baud to see the connection and webpage printout!



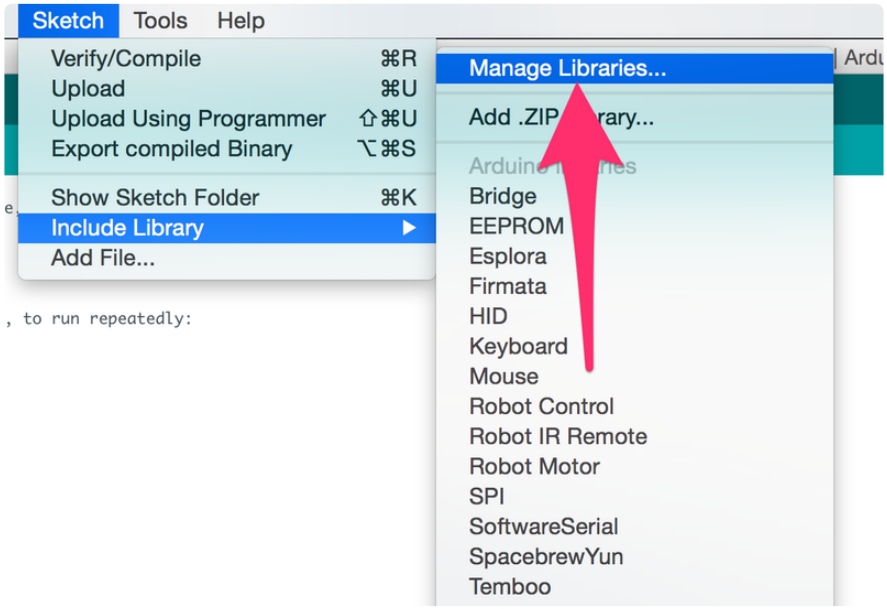
That's it, pretty easy!

This page was just to get you started and test out your module. For more information, check out the ESP8266 port github repository (https://adafru.it/eSH) for much more up-to-date documentation!

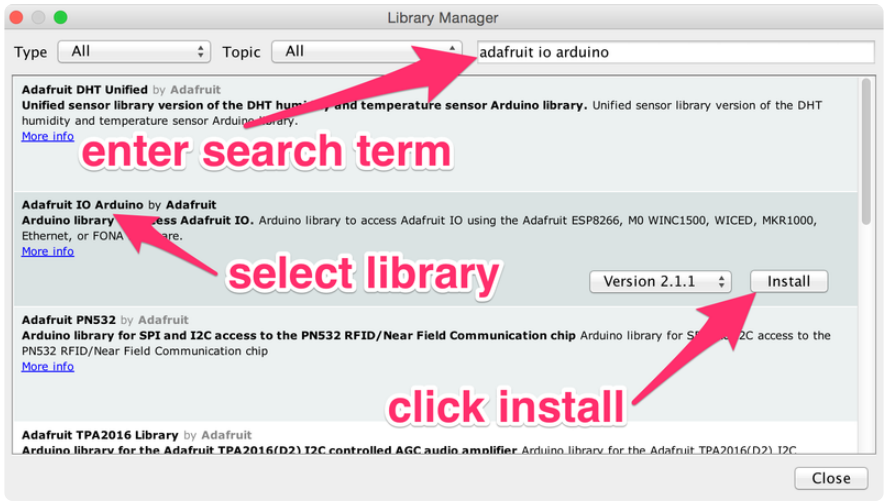
Arduino IO Library

Install the Required Libraries

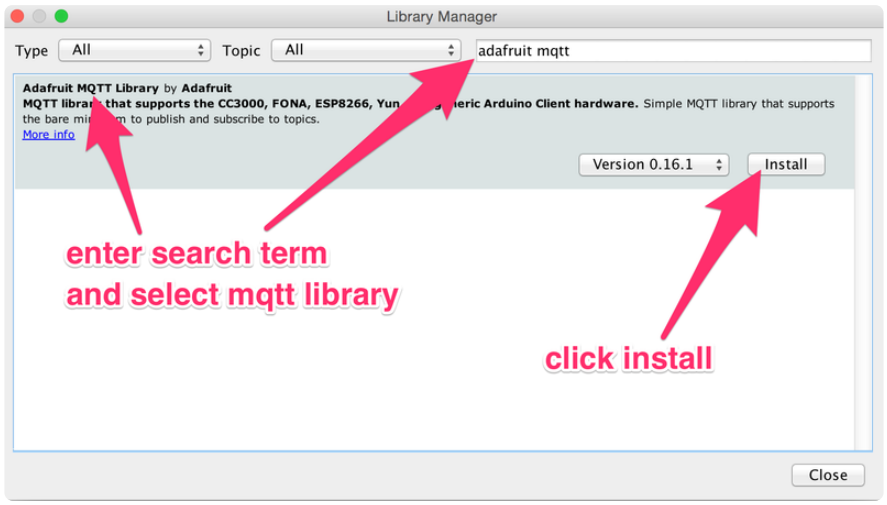
Now we will need to install the Adafruit IO, Adafruit MQTT, and ArduinoHttpClient libraries using the Arduino **Library Manager**. Navigate to the **Manage Libraries...** option in the **Sketch -> Include Library** menu.



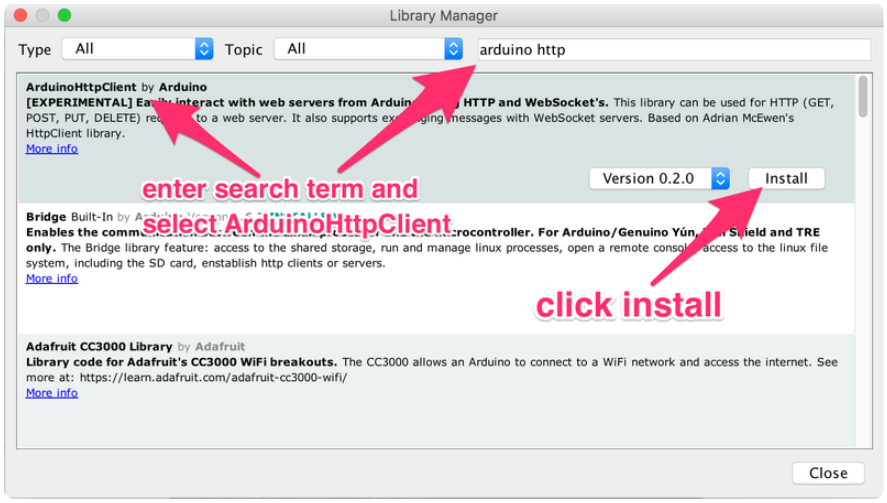
Enter **Adafruit IO Arduino** into the search box, and click **Install** on the **Adafruit IO Arduino** library option to install version 3.2.0 or higher.



Enter **Adafruit MQTT** into the search box, and click **Install** on the **Adafruit MQTT** library option to install version 1.0.0 or higher.



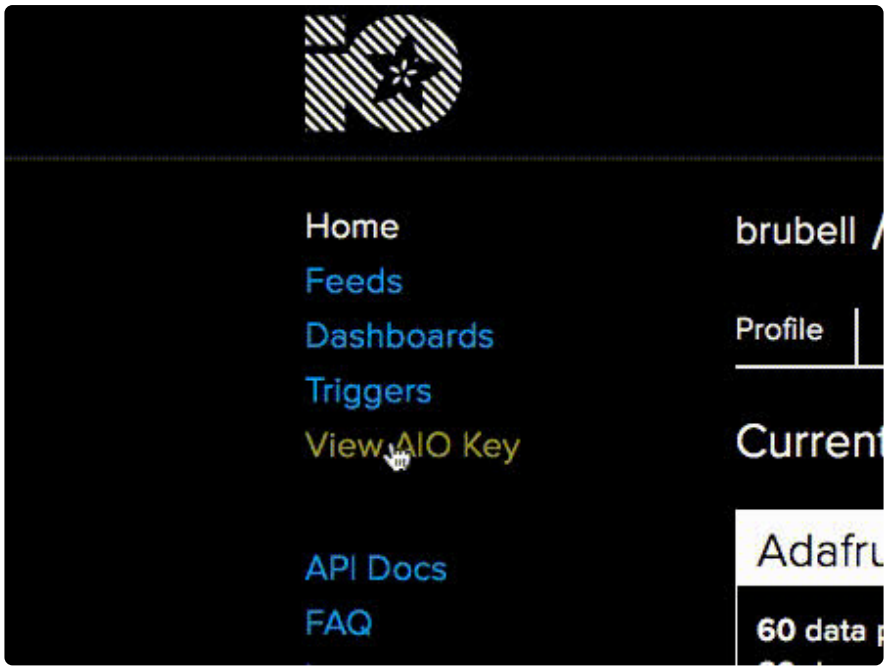
Enter **ArduinoHttpClient** into the search box, and click **Install** on the **ArduinoHttpClient** library option to install version 0.2.0 or higher.



Adafruit IO Setup

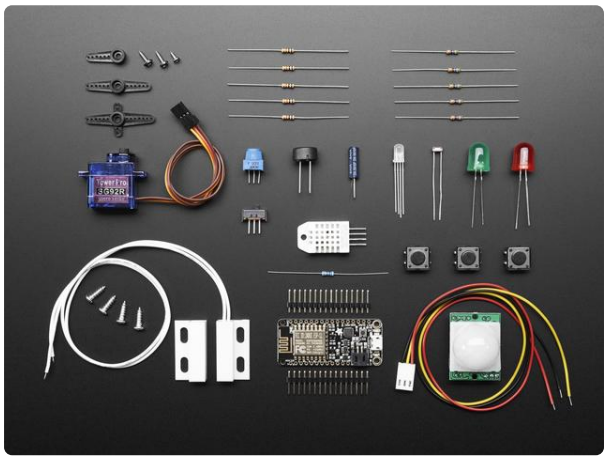
You are also going to need your Adafruit IO username and secret API key.

Navigate to your profile **(https://adafru.it/fsU) and click the View AIO Key button** to retrieve them. Write them down in a safe place, you'll need them for the next step.



Example Sketches

In the Adafruit IO Basics series of guides, the examples will be using hardware found in our Adafruit IO starter kit. If you are following along, you might want to consider using the starter kit below.



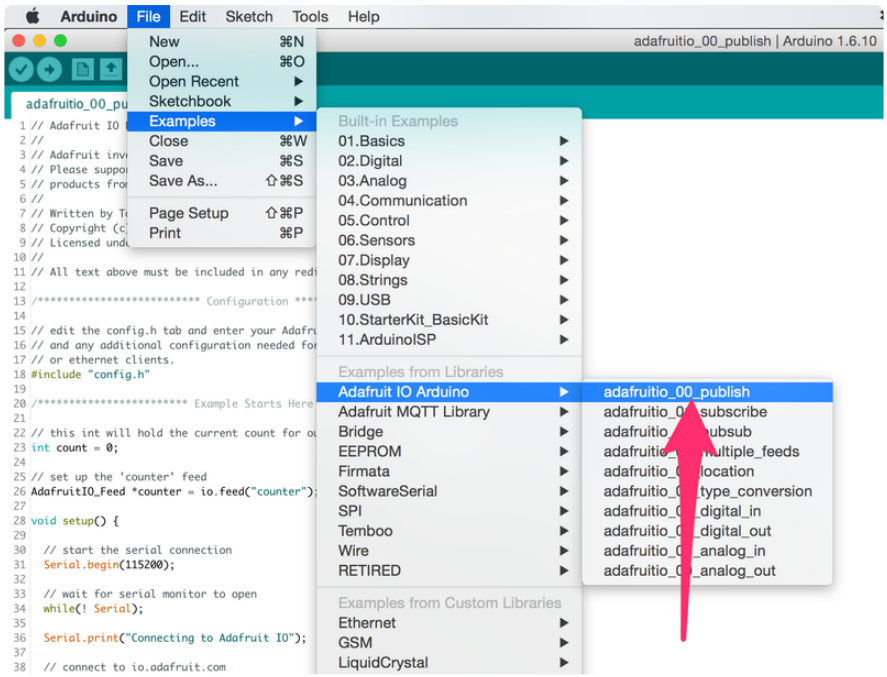
Huzzah! Adafruit.io Internet of Things Feather ESP8266

OK you've signed up for Adafruit.io and you're ready to build something cool and Internet-connected. All you need is this starter kit which...

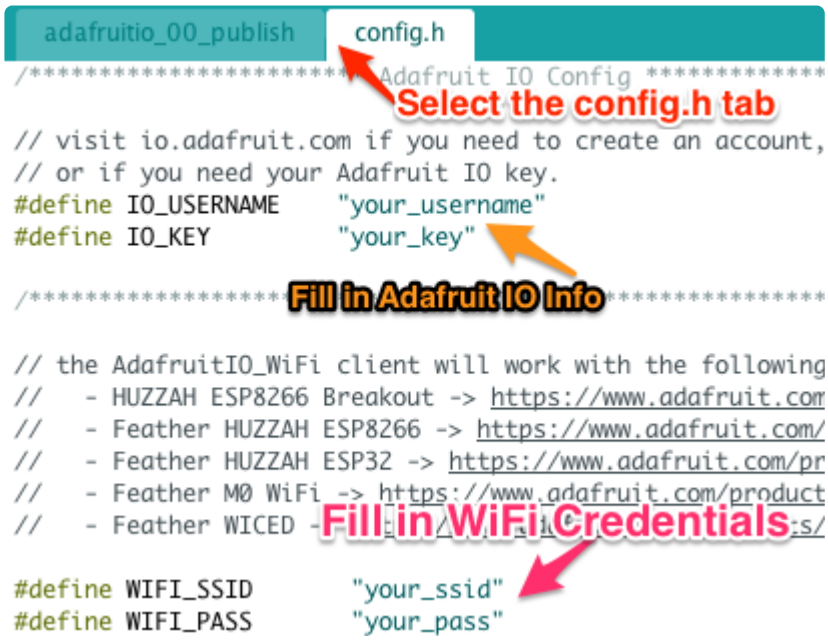
https://www.adafruit.com/product/2680

Example Sketch Setup

Now that we have installed all of the dependencies, we can try to run one of the Adafruit IO example sketches. Navigate to the **adafruitio_00_publish** sketch by opening the **File -> Examples -> Adafruit IO Arduino** menu.

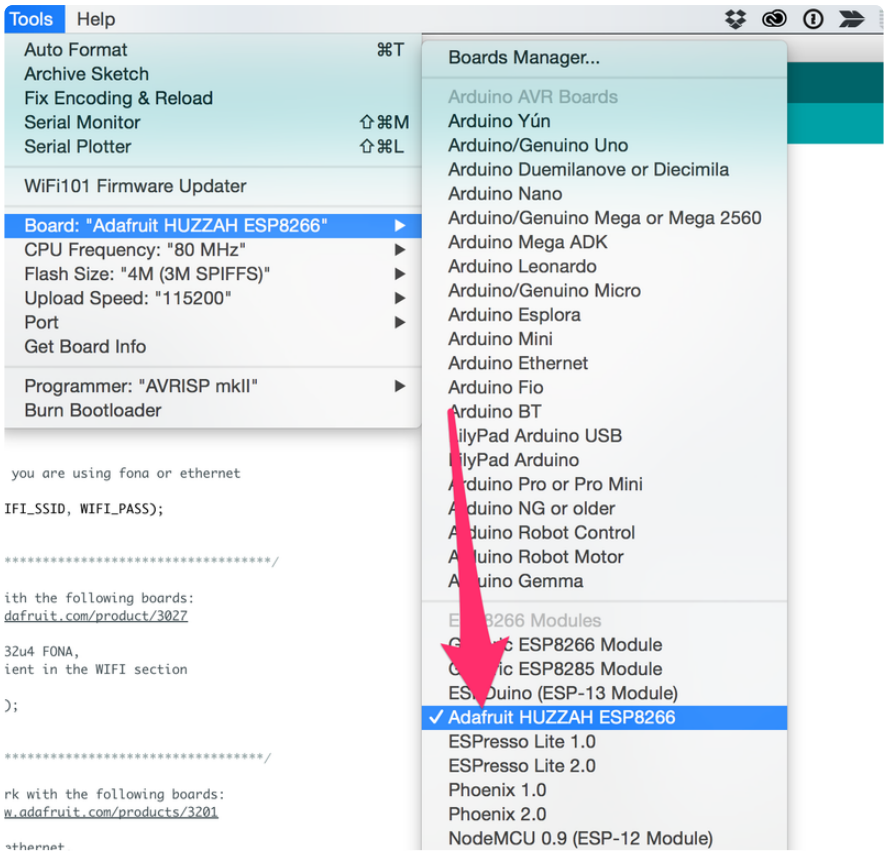


Click on the **config.h** tab, and replace the placeholders with your Adafruit IO credentials and WiFi connection info.

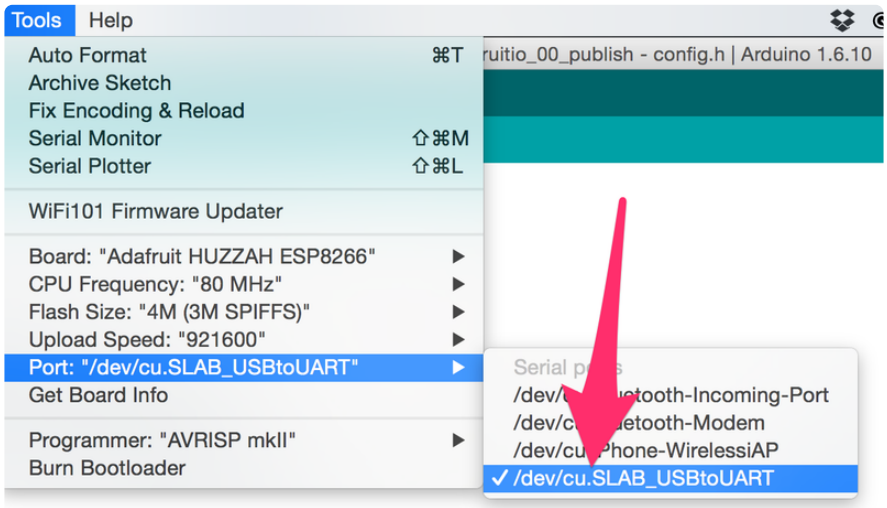


Uploading the Sketch

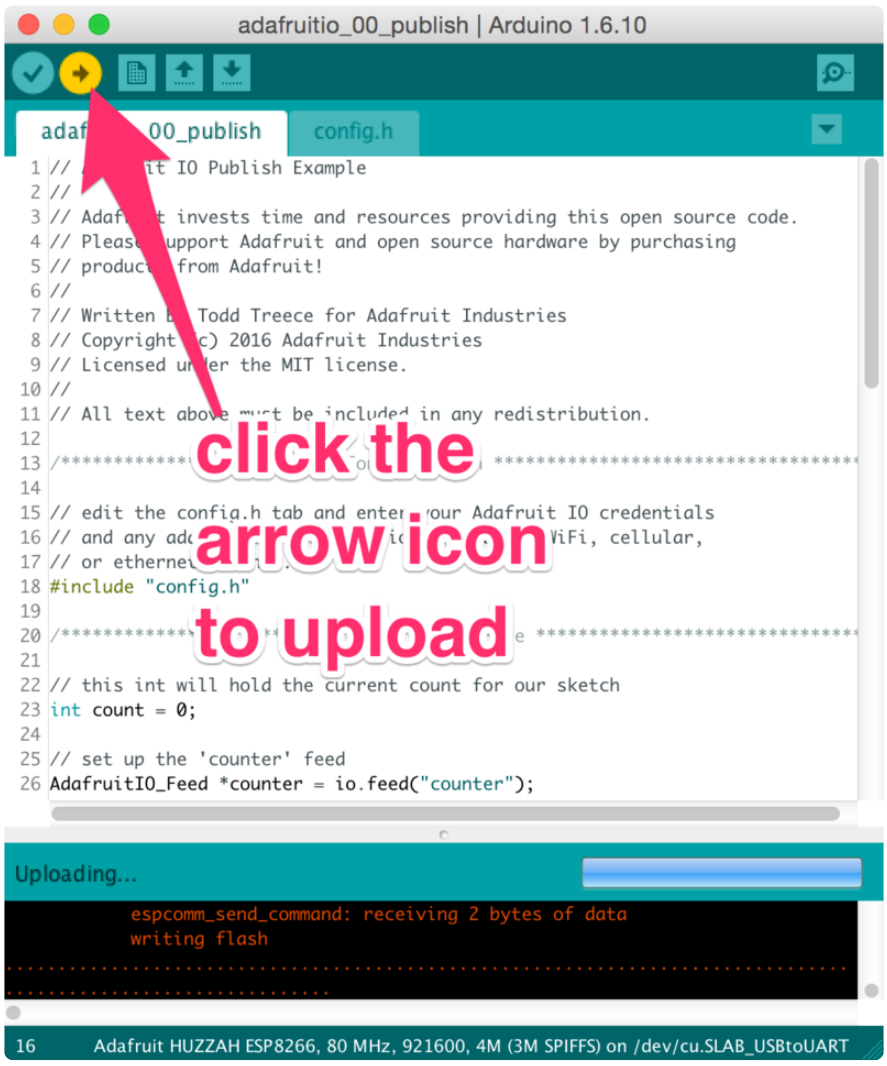
Next we will need to select the **Adafruit HUZZAH ESP8266** from the **Tools -> Board** menu.



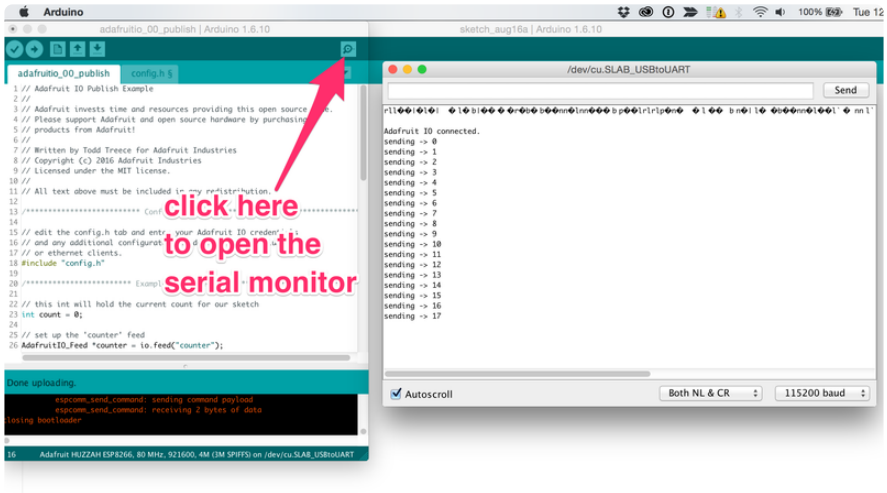
Then, select the proper COM port on Windows, or USB device on OS X.



Use the arrow icon (➡) to upload the example sketch to the ESP8266. It might take a while, but you should see a **Upload complete.** message at the bottom of the window when the process has finished.



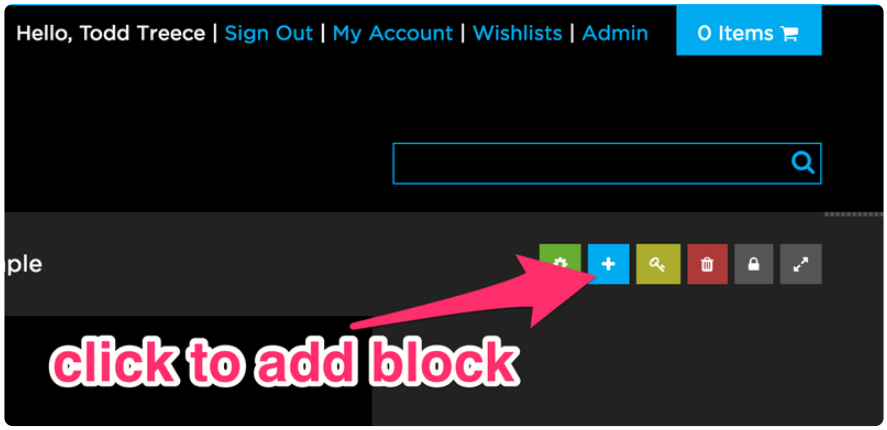
You can now click the serial monitor icon to view the output of the sketch. If everything goes as expected, you should see counter values being sent to Adafruit IO. If not, check your WiFi and Adafruit IO credentials in **config.h** and try uploading your sketch again using the process above.



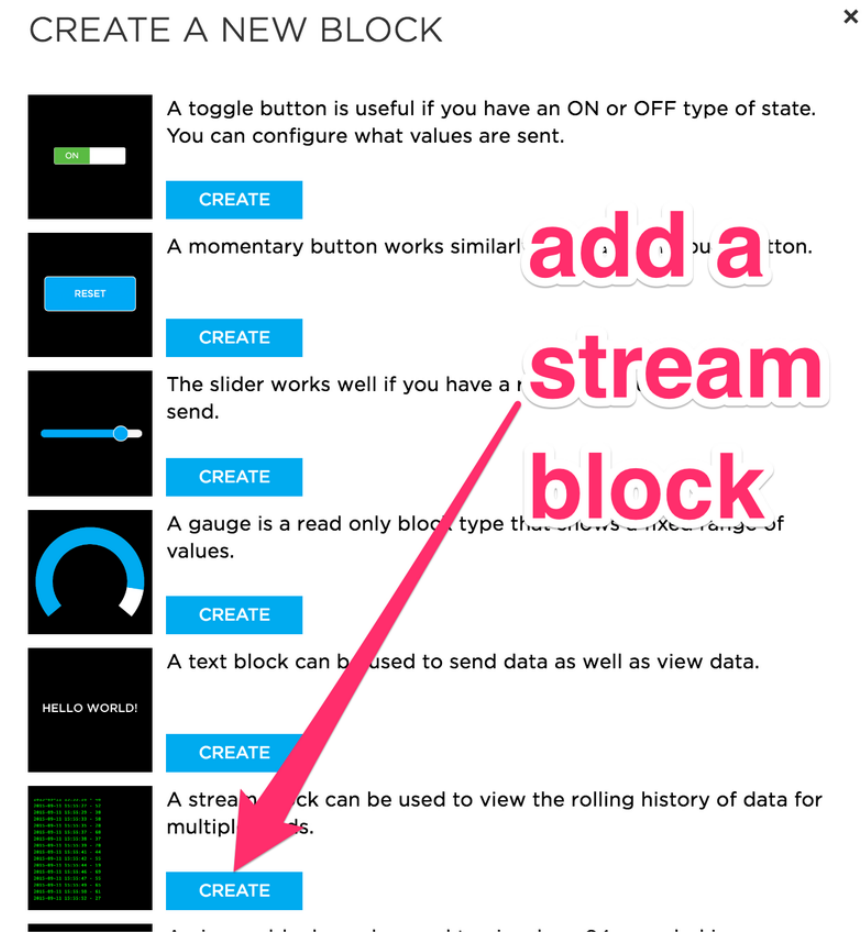
Hitting an error while compiling? Check the ESP8266 IO FAQ: https://learn.adafruit.com/adafruit-io-basics-esp8266-arduino/adafruit-io-faq

Viewing Data on Adafruit IO

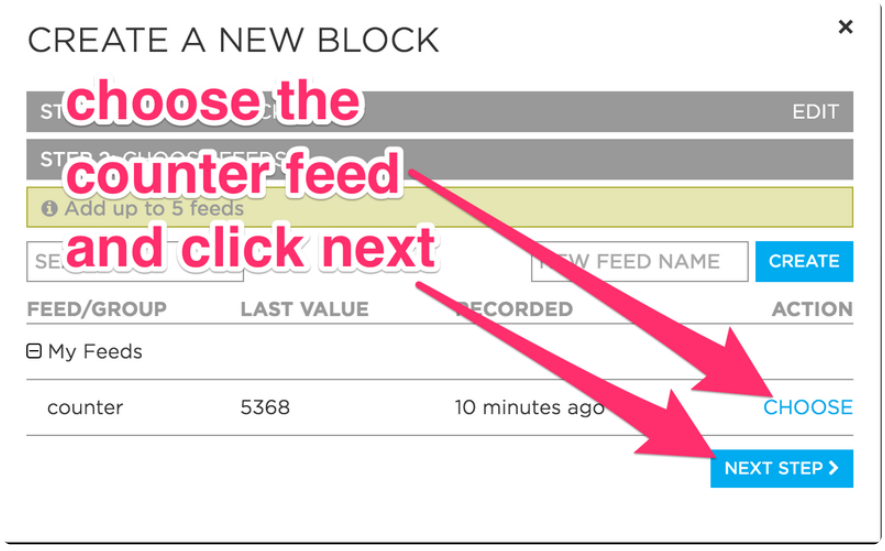
Now that your ESP8266 is sending data to Adafruit IO, you can view the data stream on io.adafruit.com by adding a stream block to your dashboard. To do this, click on the + icon on the right hand side of the dashboard.



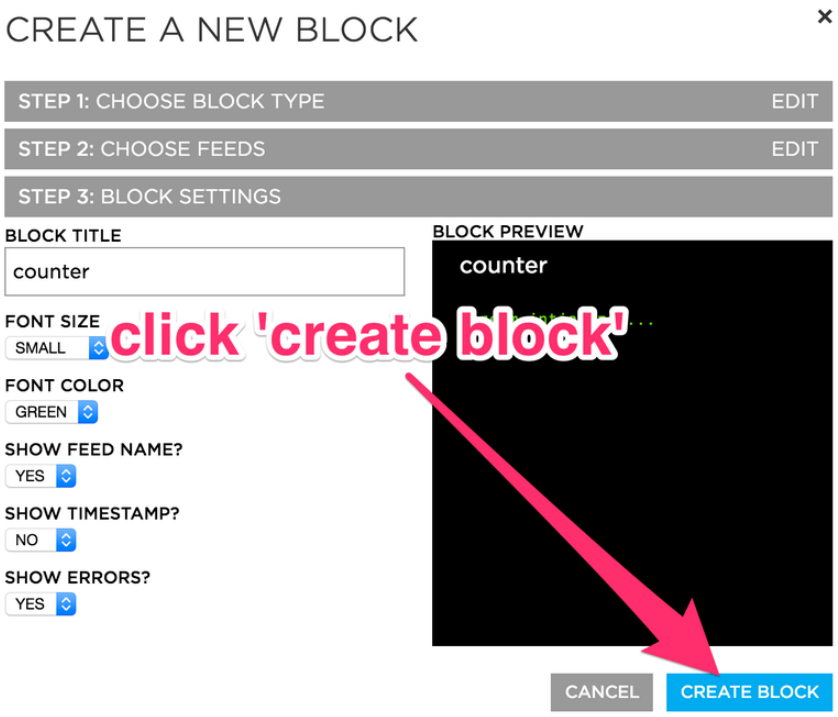
Add a new **stream block** by selecting it from the modal.



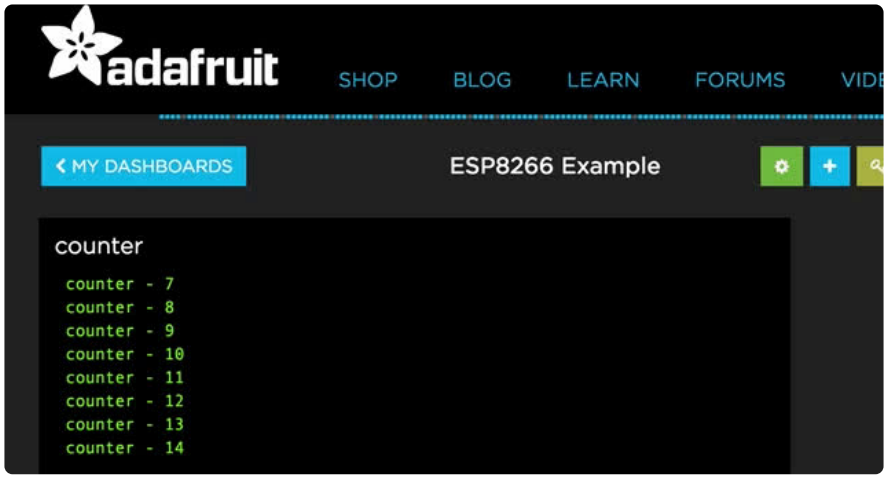
Next, choose the **counter** feed from the list, and click the **Next Step** button.



Modifythe stream block options as needed, and click the **create block** button when you are finished.



You should now see data flowing into your stream block from your ESP8266.



Next Steps

If you would like to continue your educational journey with your ESP8266 & Adafruit IO, check out the Adafruit IO Basics series of guides.

Adafruit IO FAQ

Encountering an issue with your Adafruit IO Arduino Project?

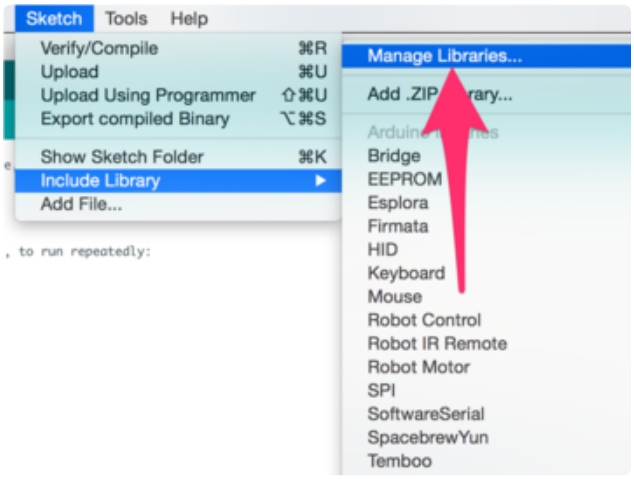
If you're having an issue compiling, connecting, or troubleshooting your project, check this page first.

Don't see your issue? Post up on the Adafruit IO Forum with your issue (https://adafru.it/plC).

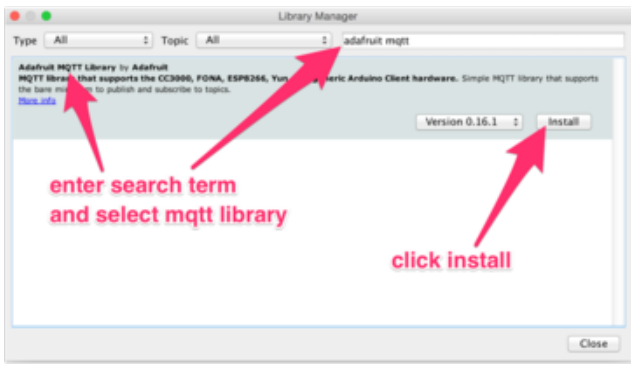
I encounter the following error when compiling my sketch:

`fatal error: Adafruit_MQTT.h: No such file or directory, #include "Adafruit_MQTT.h"`

The Adafruit IO Arduino library is dependent on our Adafruit IO MQTT Library.



To resolve this error, from the Arduino IDE, navigate to the **Manage Libraries...** option in the **Sketch -> Include Library** menu.



To resolve this error, from the Arduino IDE, navigate to the **Manage Libraries...** option in the **Sketch -> Include Library** menu.

My Serial Monitor prints "..." endlessly after the "Connecting to Adafruit IO" message

Your board is not connecting to Adafruit IO, but why? Let's find out:

First, check in `config.h` that you have the correct `IO_USERNAME`, `IO_KEY`, `WIFI_SSID`, and `WIFI_PASS` are set correctly.

Next, we're going to modify the while loop which waits for an IO connection in your sketch. Change the line in the status check loop from `Serial.println(.);` to `Serial.println(io.statusText());`

```
// wait for a connection
while(io.status() < AIO_CONNECTED) {
Serial.println(io.statusText());
delay(500);
}
```

Verify and re-upload the sketch. If you're receiving a **Network disconnected** error message, the board is not able to talk to the internet. Re-check your hardware, connections, and router settings.

If it's still not showing **Adafruit IO connected**, check the IO status on the Adafruit Status page (https://adafru.it/Oc0) to make sure the service is online.

My data isn't displaying, is Adafruit IO's {service/MQTT/API} down?

Possibly - you can check IO status on the Adafruit Status page (https://adafru.it/Oc0).

Is my data being sent properly? Am I sending too much data?

There's a monitor page built-into Adafruit IO (https://adafru.it/DOK) which provides a live view of incoming data and error messages. Keep this page open while you send data to your Adafruit IO devices to monitor data and errors.