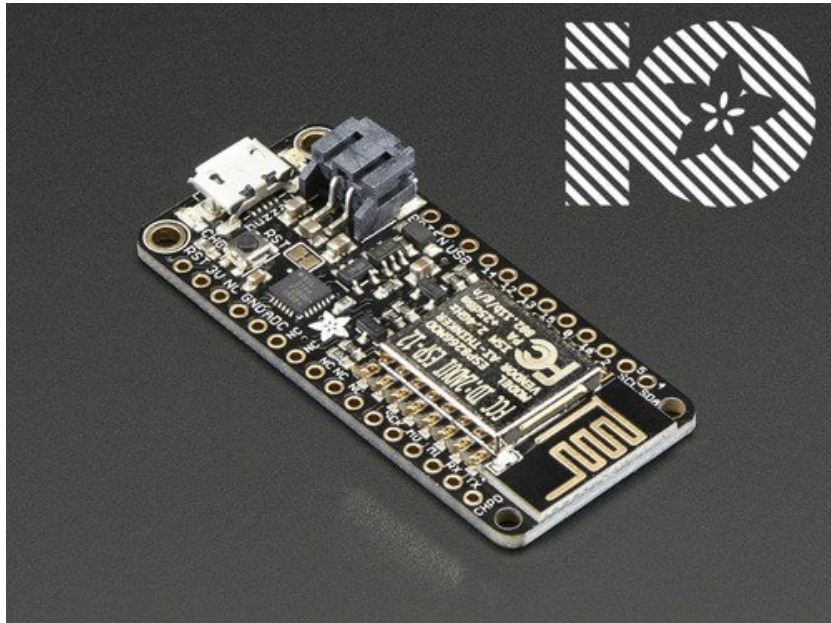


□

## Adafruit IO Basics: ESP8266 + Arduino

Created by Todd Treece

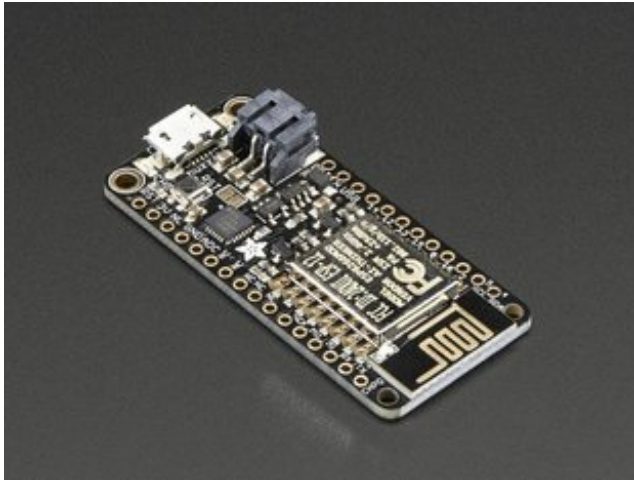


Last updated on 2017-03-27 10:31:41 PM UTC

# Guide Contents

Guide Contents	2
Overview	3
Adafruit Feather HUZZAH with ESP8266 WiFi	3
Pros/Cons of the ESP8266	3
Pros	4
Cons	4
Assembly	5
Header Options!	5
Soldering in Plain Headers	8
Prepare the header strip:	8
Add the breakout board:	9
And Solder!	9
Soldering on Female Header	11
Tape In Place	12
Flip & Tack Solder	13
And Solder!	14
Using Arduino IDE	17
Install the Arduino IDE 1.6.8 or greater	18
Install the ESP8266 Board Package	18
Setup ESP8266 Support	19
Blink Test	21
Connecting via WiFi	22
Arduino IO Library	27
Install the Required Libraries	27
Adafruit IO Setup	30
Example Sketches	31
Huzzah! Adafruit.io Internet of Things Feather ESP8266	31
Example Sketch Setup	31
Uploading the Sketch	33
Viewing Data on Adafruit IO	37
Next Steps	41

# Overview



## Adafruit Feather HUZAH with ESP8266 WiFi

PRODUCT ID: 2821

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 cores....

<http://adafru.it/n6A>

\$16.95

IN STOCK

The ESP8266 based Feather HUZAH & the HUZAH 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 HUZAH ESP8266 breakout](#)
- [Adafruit Feather HUZAH 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

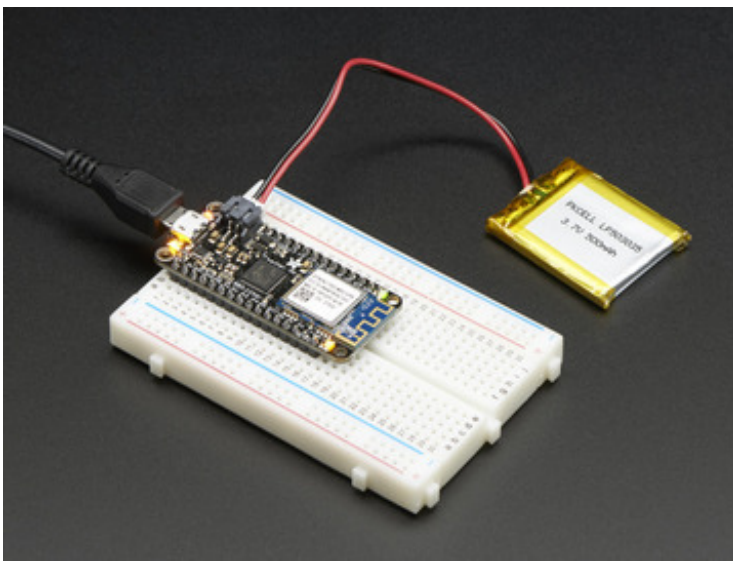
Lets get started with assembly.

# 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

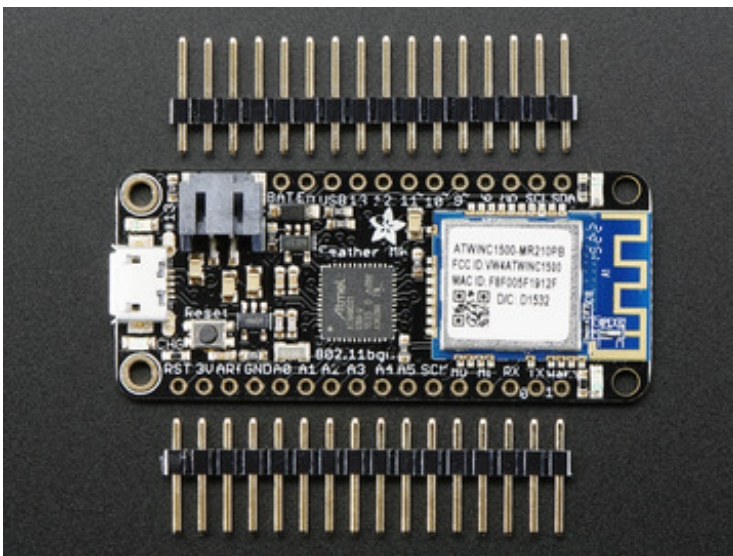
## Header Options!

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

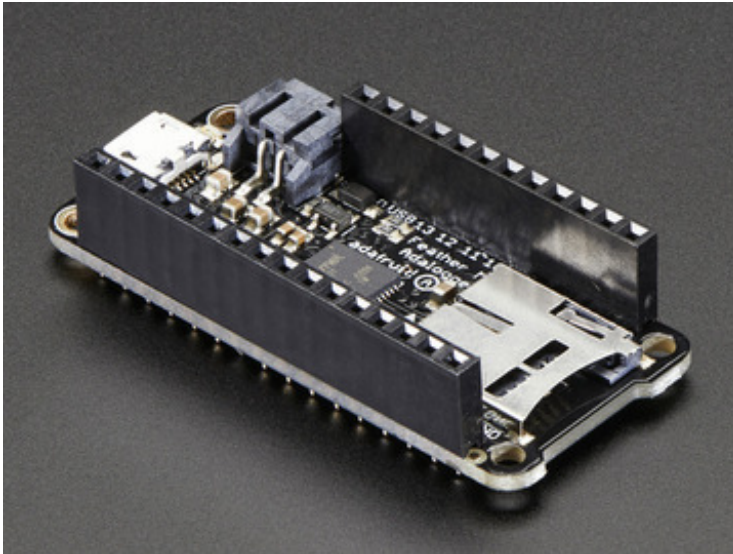


- 

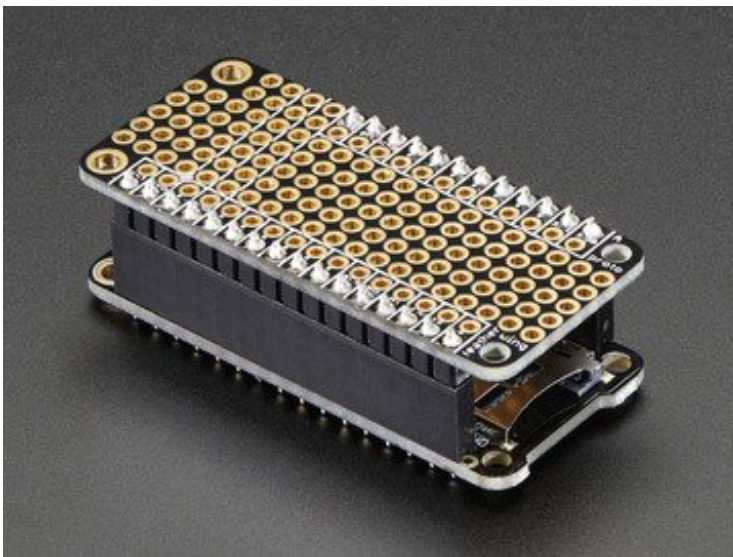
The first option is soldering in plain male headers, this lets you plug in the Feather into a solderless breadboard



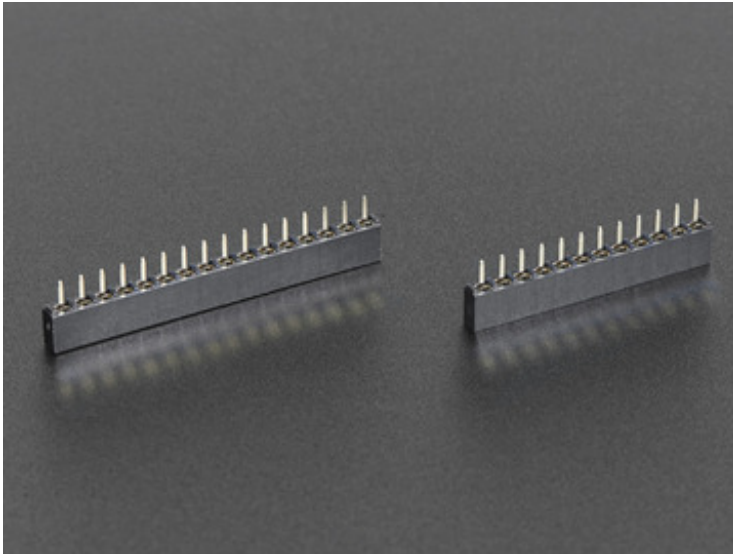
-



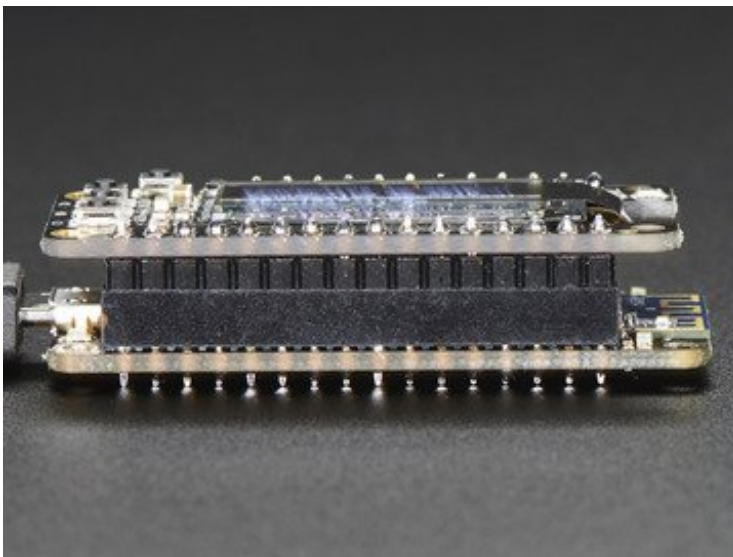
Another option is to go with socket female headers. This won't let you plug the Feather into a breadboard but it will let you attach featherwings very easily



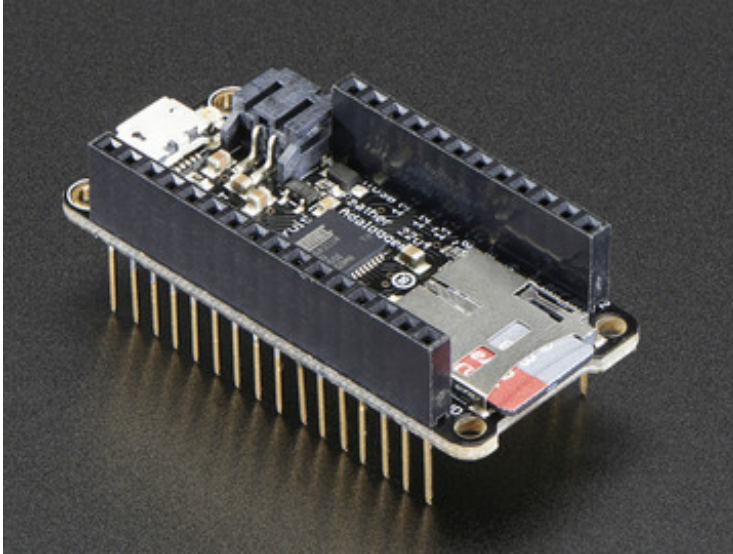




We also have 'slim' versions of the female headers, that are a little shorter and give a more compact shape



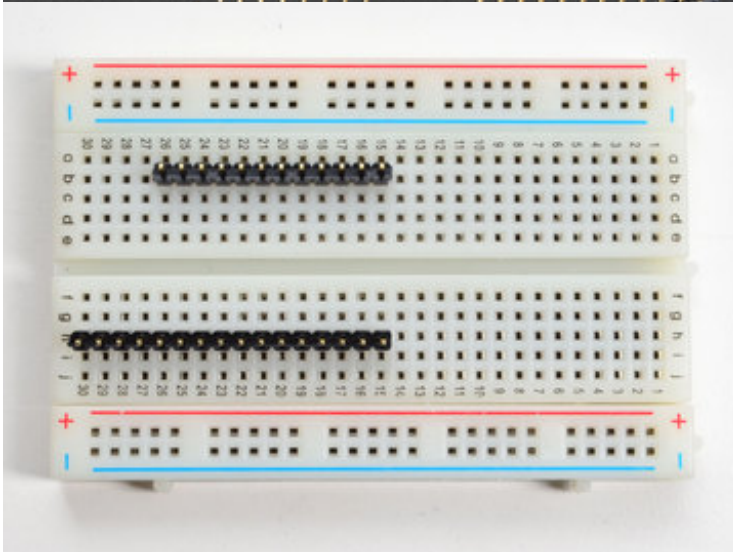
Finally, there's the "Stacking Header" option. This one is sort of the best-of-both-worlds. You get the ability to plug into a



solderless breadboard *and* plug a featherwing on top. But its a little bulky

•

## Soldering in Plain Headers



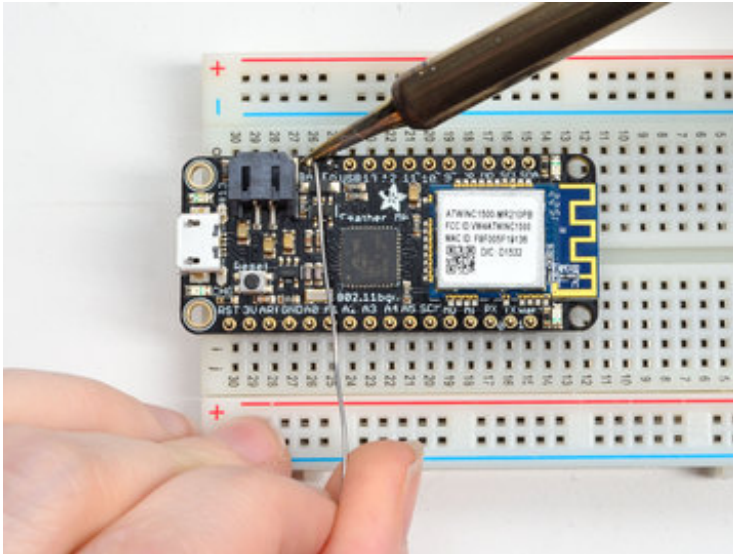
•

•

## Prepare the header strip:

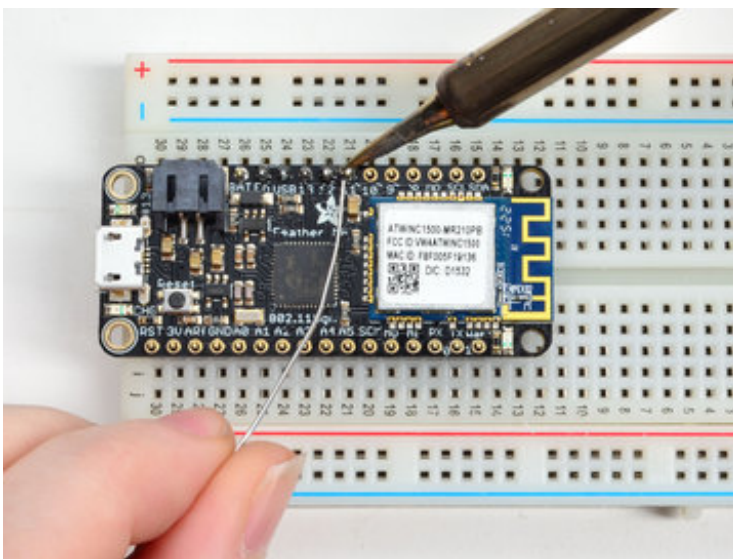
Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - **long pins down**





## Add the breakout board:

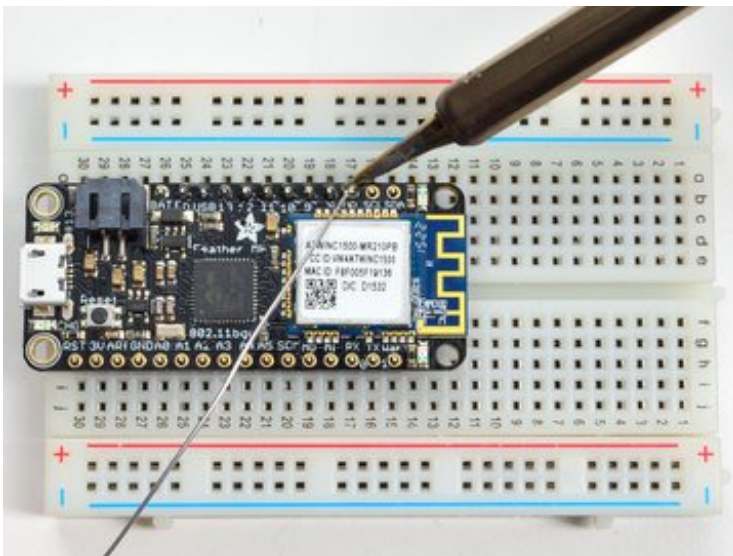
Place the breakout board over the pins so that the short pins poke through the breakout pads

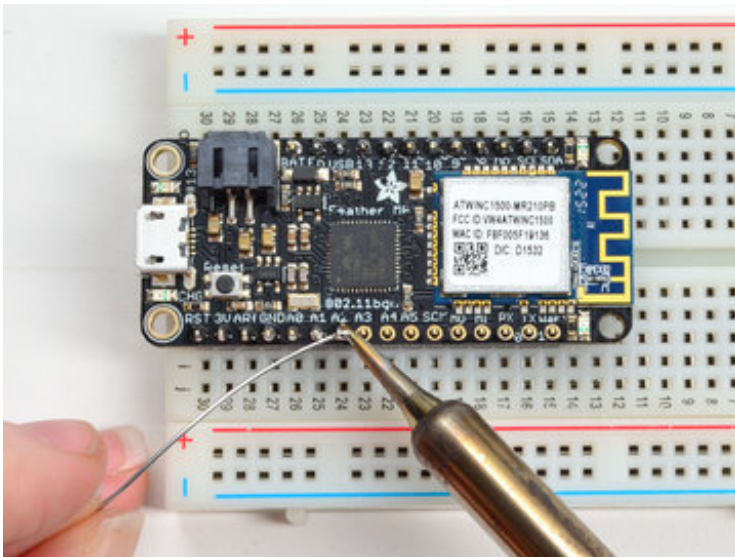
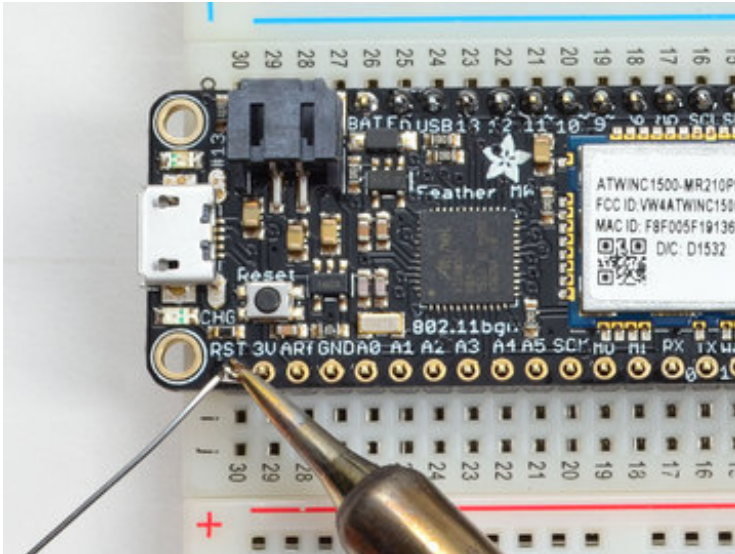


## And Solder!

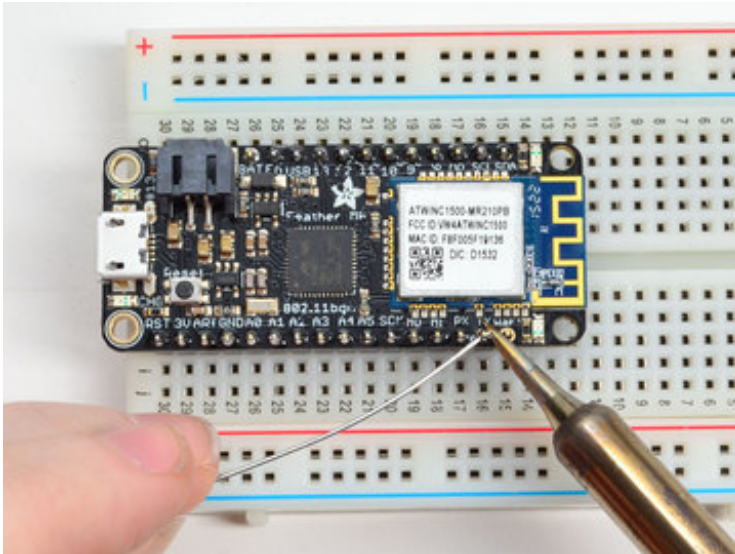
Be sure to solder all pins for reliable electrical contact.

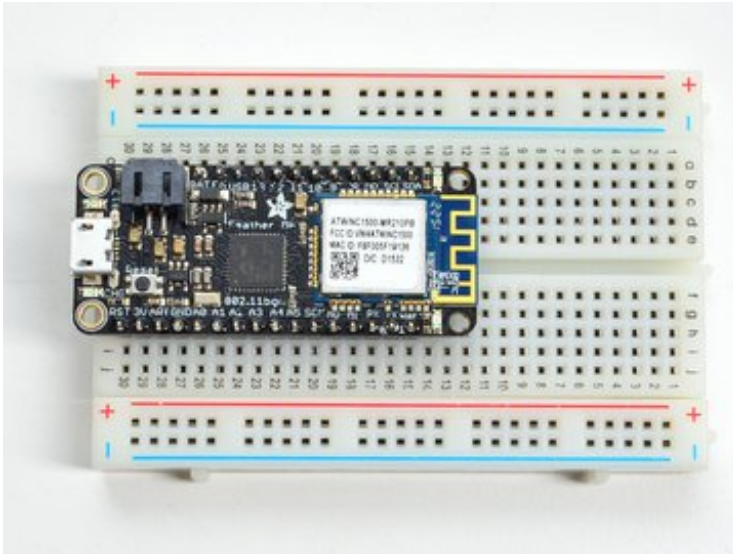
(For tips on soldering, be sure to check out our [Guide to Excellent Soldering](http://adafruit.it/aTk) (<http://adafruit.it/aTk>)).





Solder the other strip as well.





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

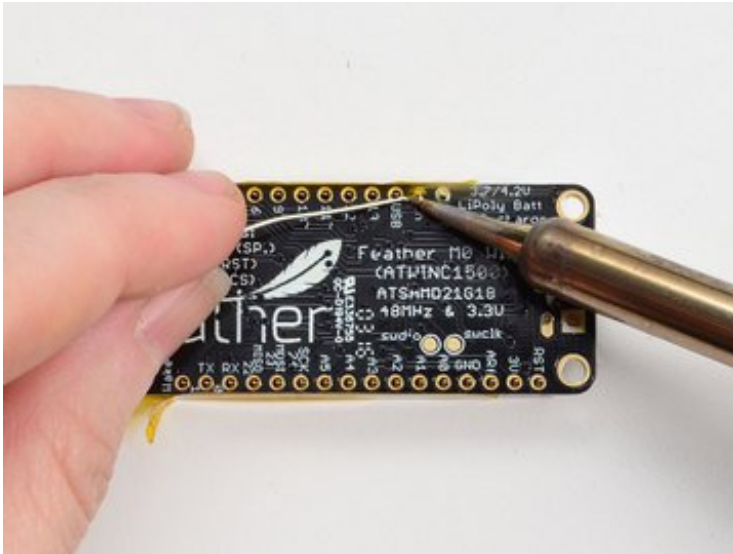
## Soldering on Female Header



## 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



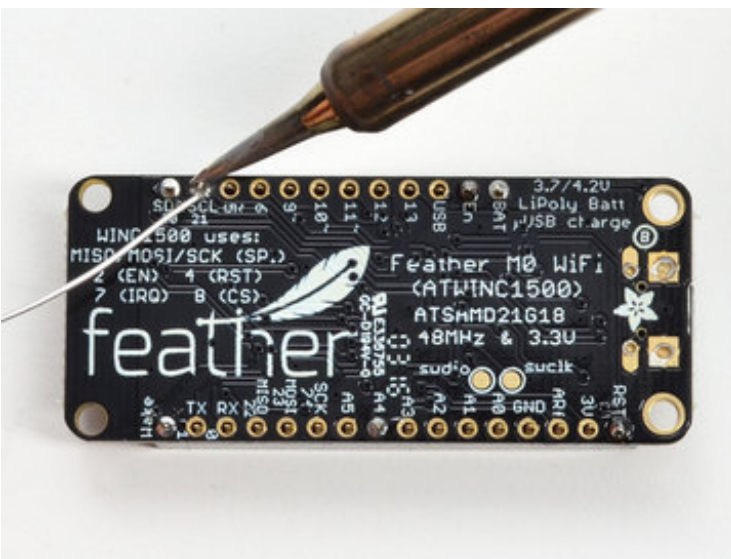
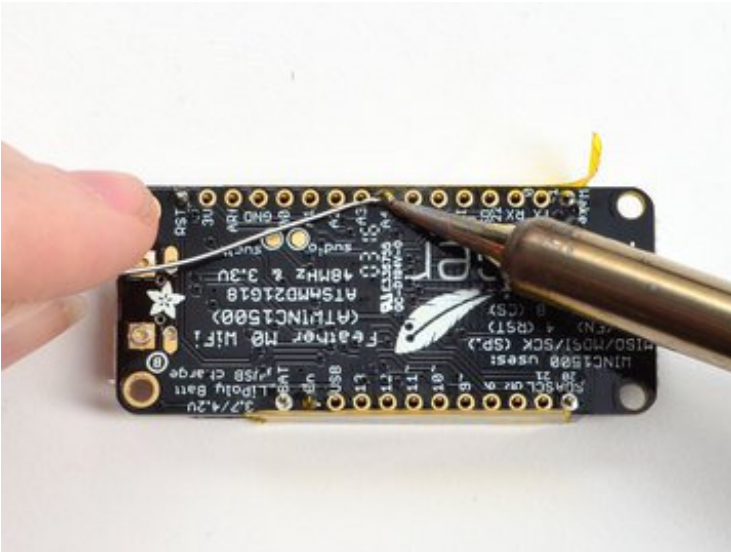


## **And Solder!**

Be sure to solder all pins for reliable electrical contact.

*(For tips on soldering, be sure to*

check out our [Guide to Excellent Soldering](http://adafruit.it/aTk) (<http://adafruit.it/aTk>).

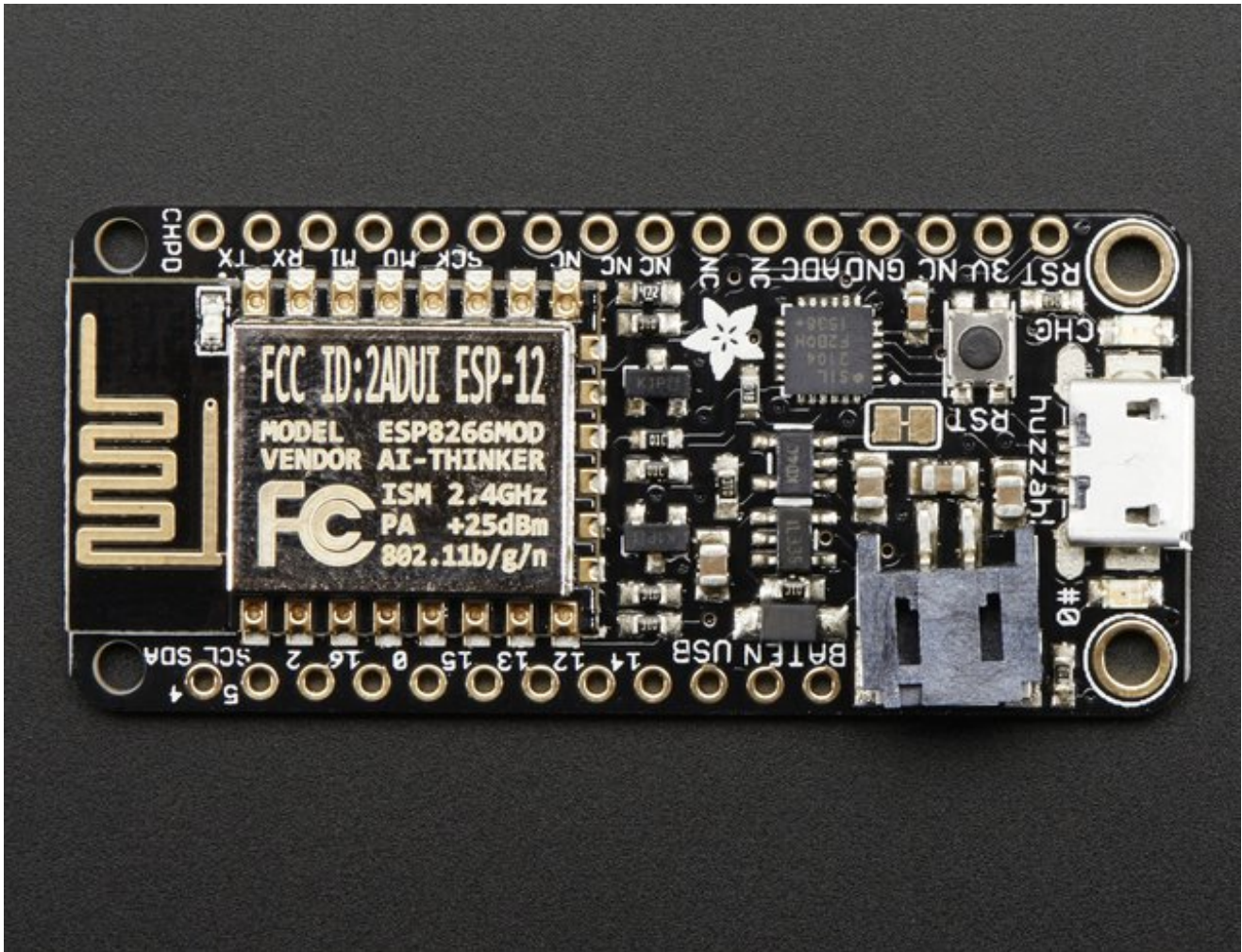




You're done! Check your solder joints visually and continue onto the next steps



# Using Arduino IDE





Don't forget to install the USB driver for the CP2104 USB-to-Serial chip!

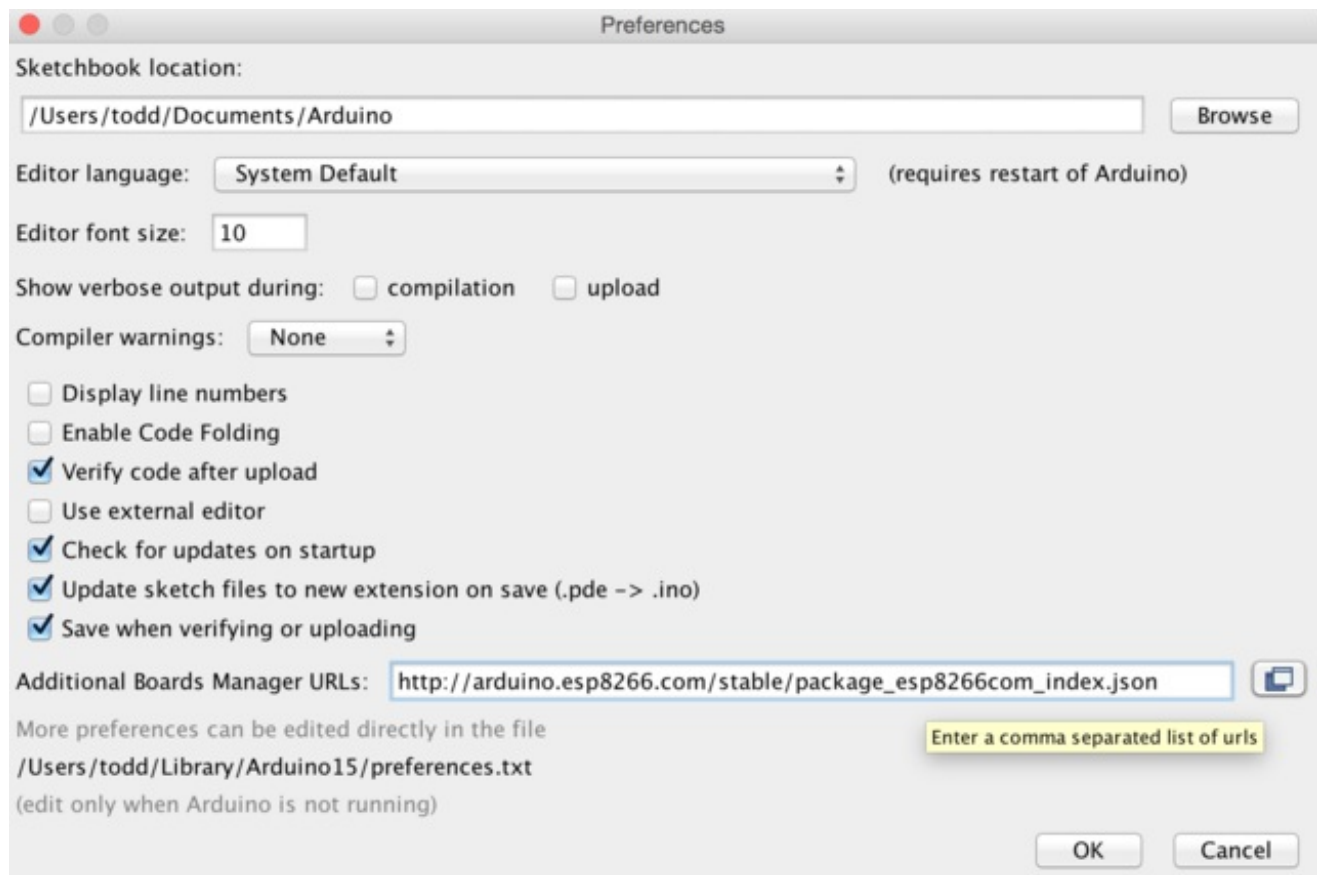
## Install the Arduino IDE 1.6.8 or greater

[Download Arduino IDE from Arduino.cc \(1.6.8 or greater\)](http://adafru.it/f1P) (<http://adafru.it/f1P>) from Arduino.cc

The latest is usually the best

## Install the ESP8266 Board Package

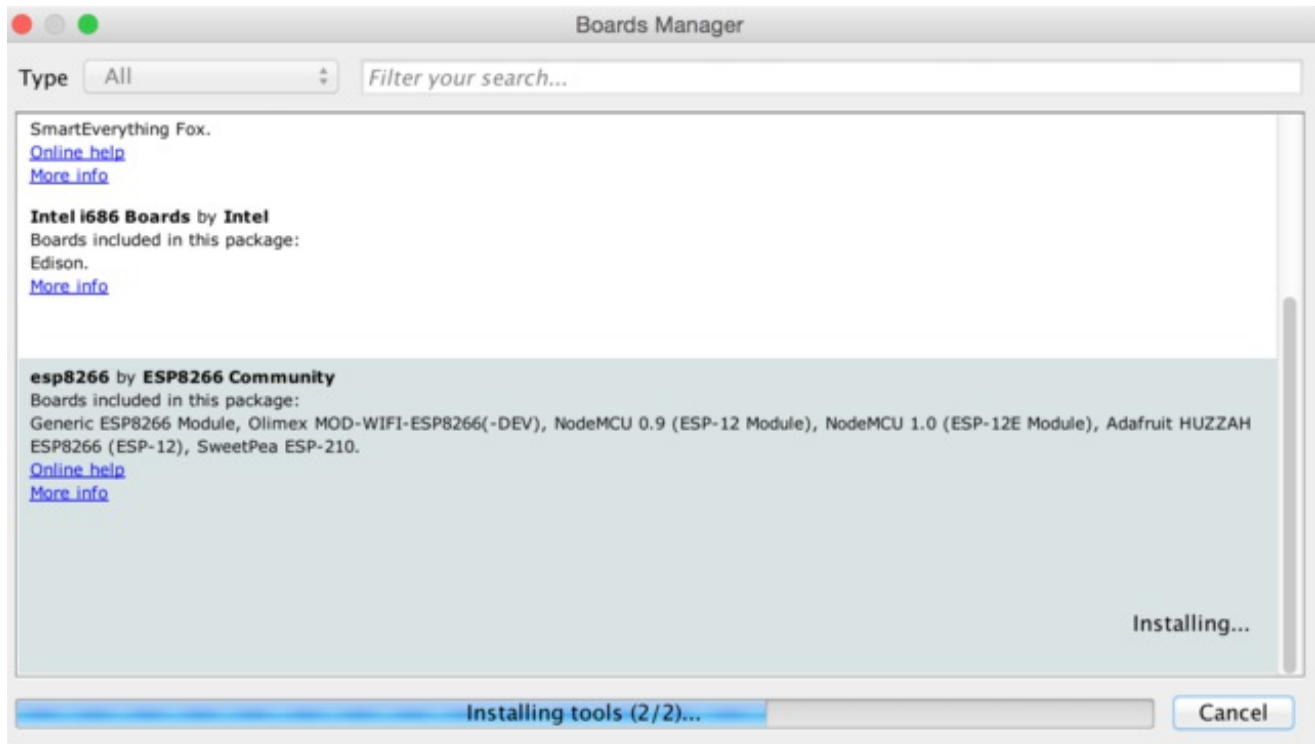
Enter [http://arduino.esp8266.com/stable/package\\_esp8266com\\_index.json](http://arduino.esp8266.com/stable/package_esp8266com_index.json) into *Additional Board Manager URLs* field in the Arduino v1.6.4+ preferences.



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

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





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

**esp8266 by ESP8266 Community** version **2.3.0** **INSTALLED**

Boards included in this package:

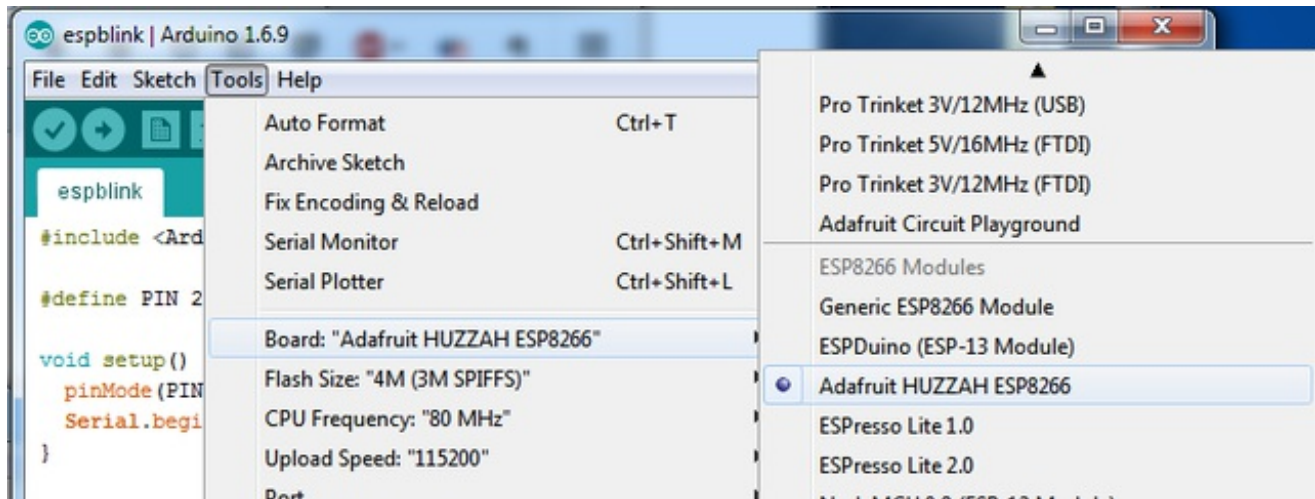
Generic ESP8266 Module, Olimex MOD-WIFI-ESP8266(-DEV), NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Adafruit HUZZAH ESP8266 (ESP-12), ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, SparkFun Thing, SweetPea ESP-210, WeMos D1, WeMos D1 mini, ESPino (ESP-12 Module), ESPino (WROOM-02 Module), WifInfo, ESPDuino.

[Online help](#)

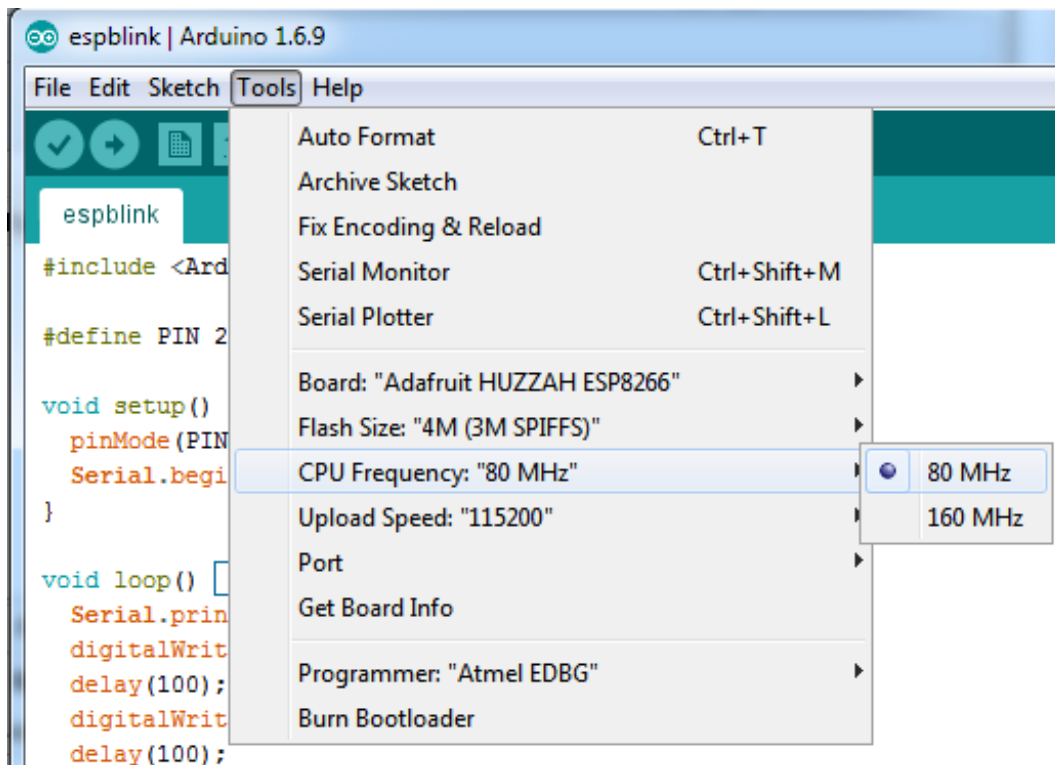
[More info](#)

## Setup ESP8266 Support

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

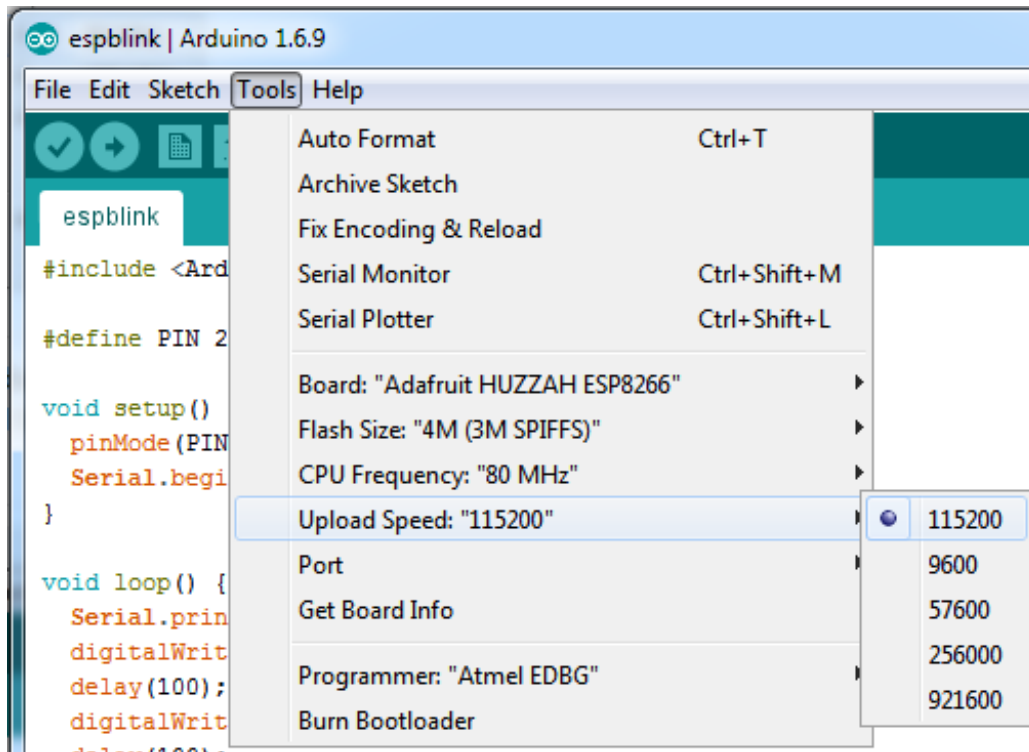


80 MHz as the CPU frequency

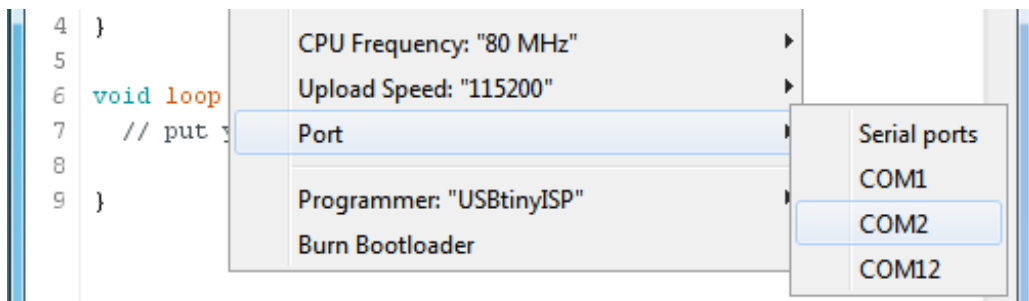


You can keep the **Flash Size** at "4M (3M SPIFFS)"

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



## Blink Test

We'll begin with the simple blink test

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

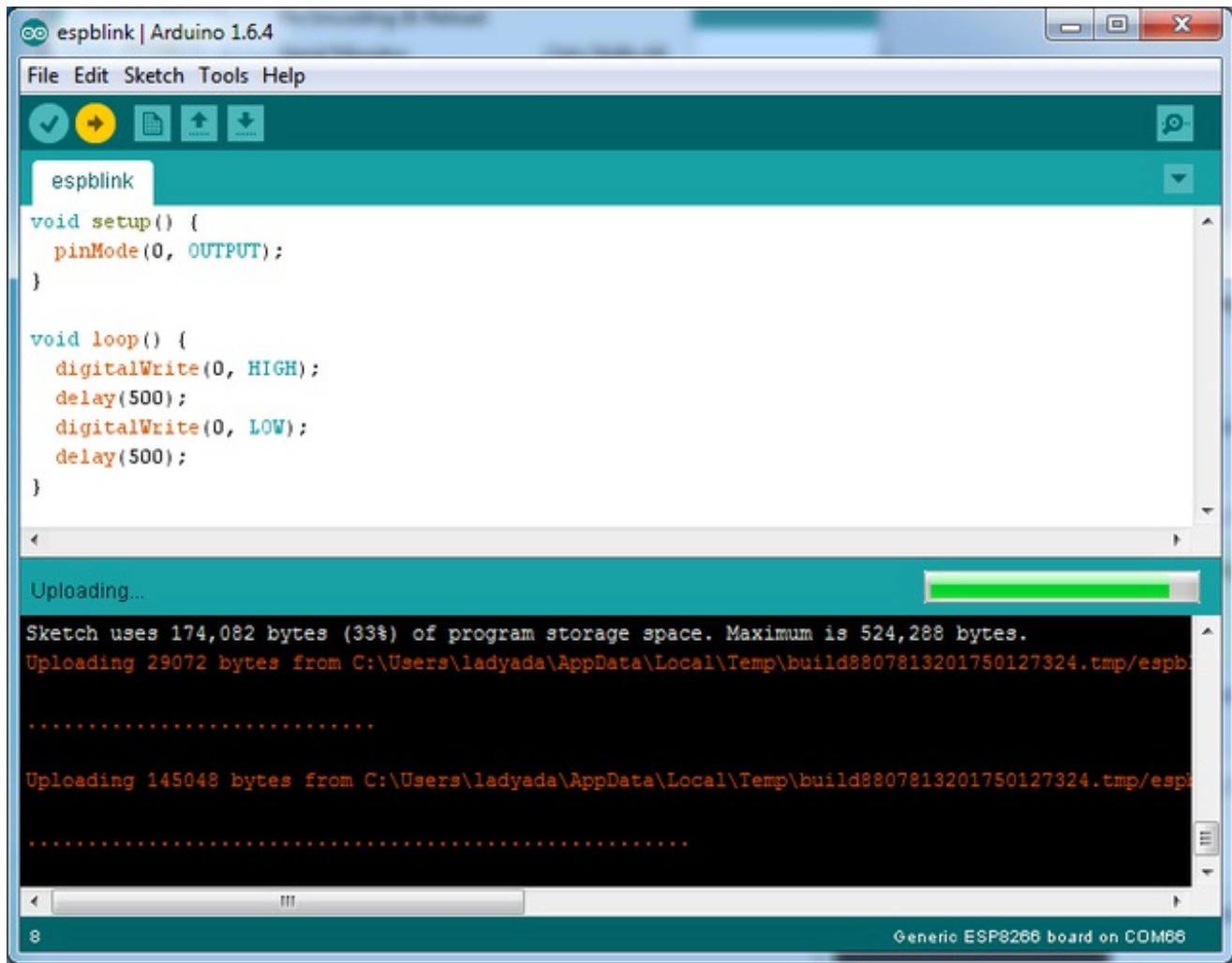
```

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 automatically



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 <ESP8266WiFi.h>
```

```

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");
}

```

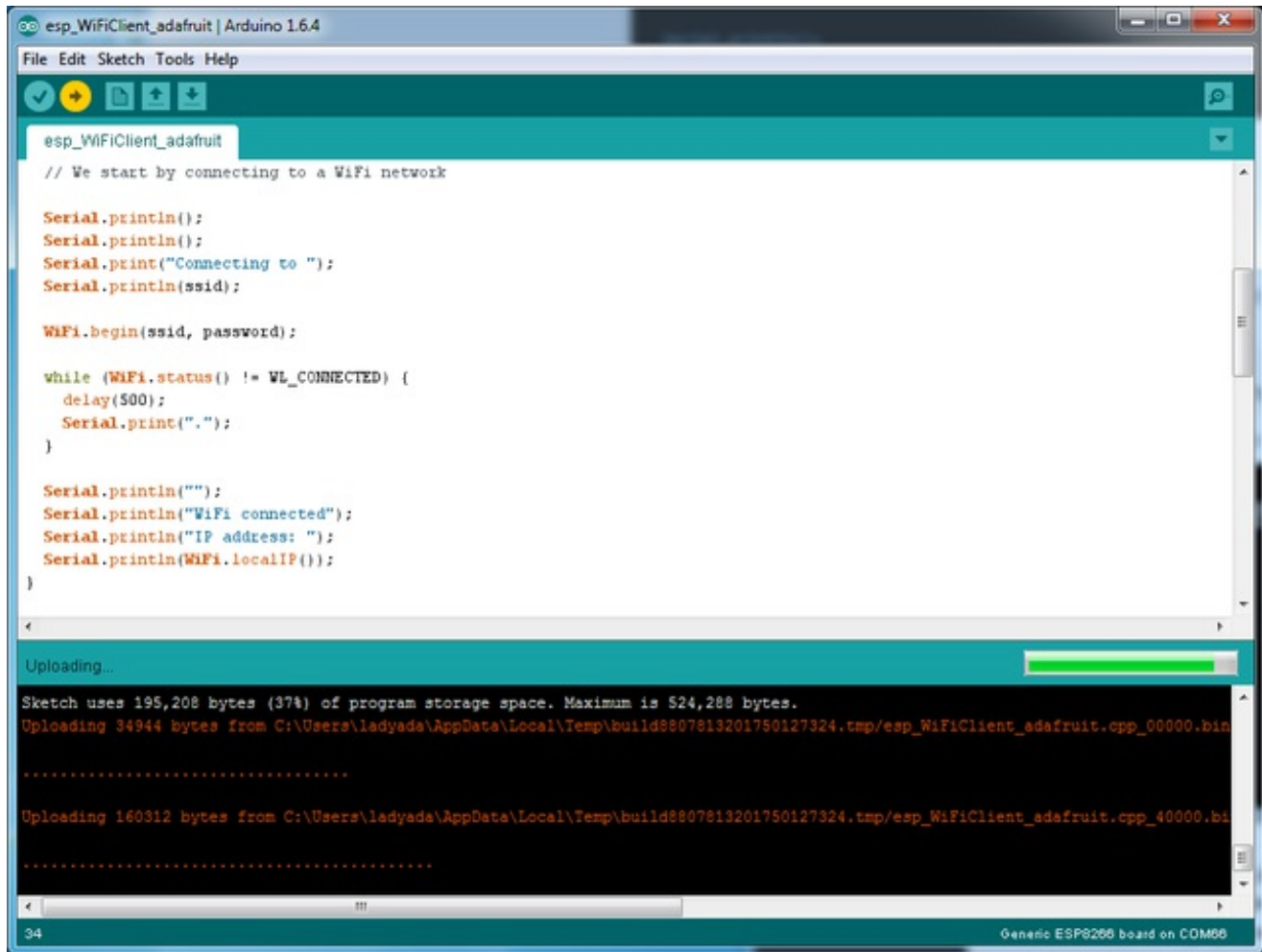
Dont forget to update

```

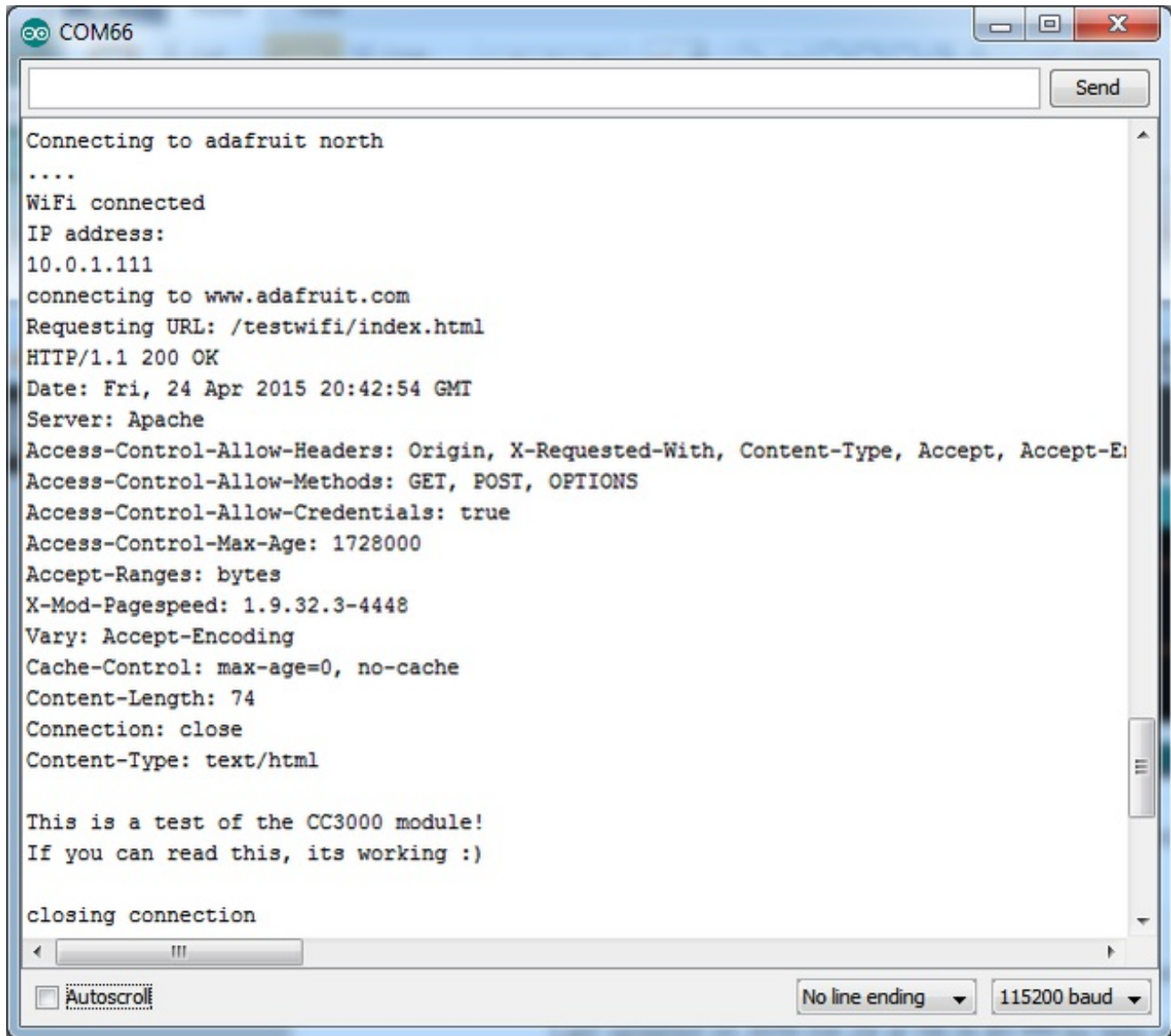
const char* ssid    = "yourssid";
const char* password = "yourpassword";

```

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!



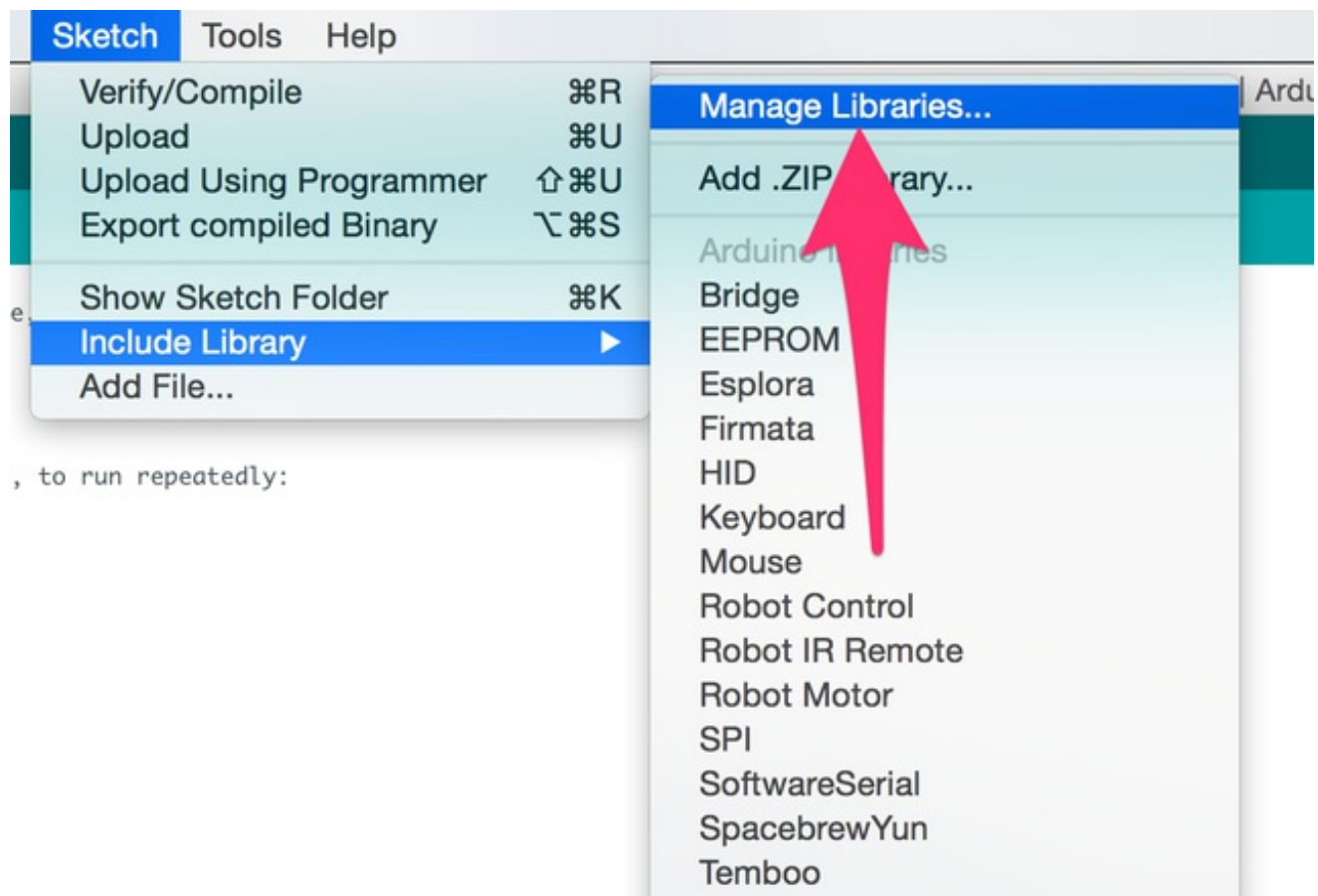
That's it, pretty easy!

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

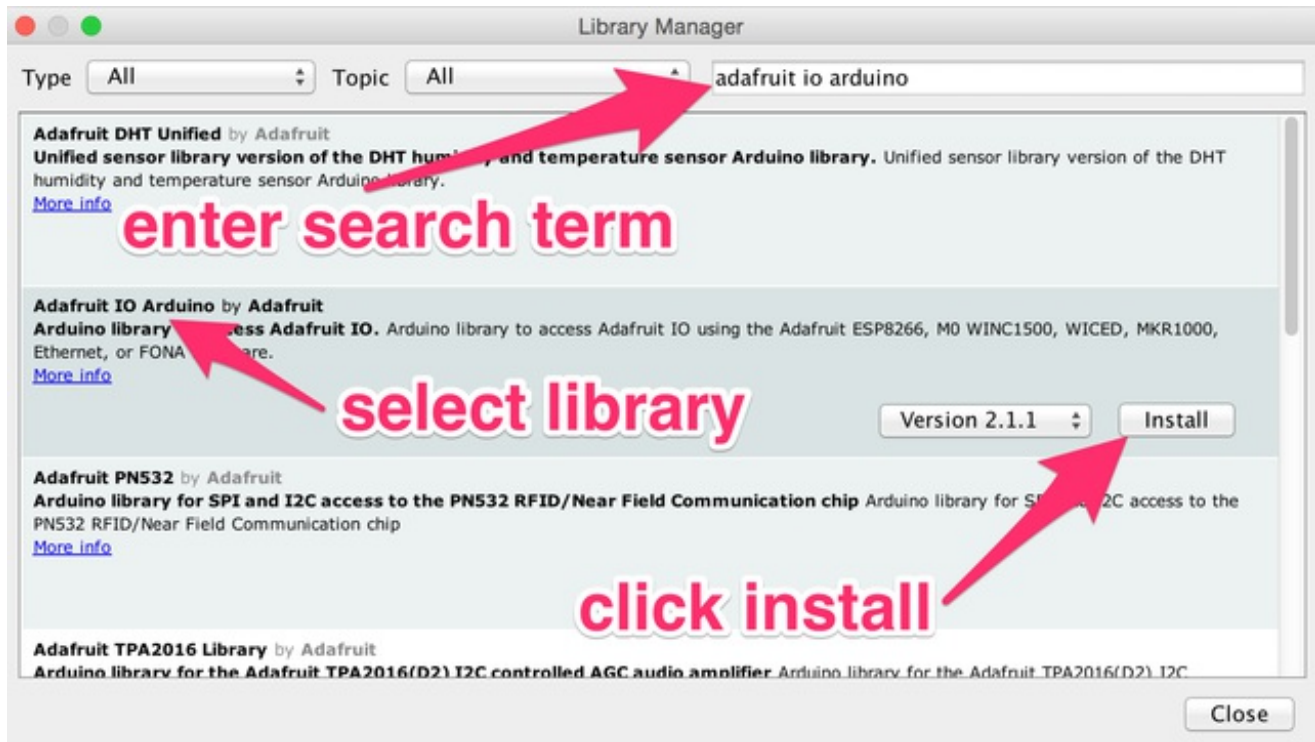
# Arduino IO Library

## Install the Required Libraries

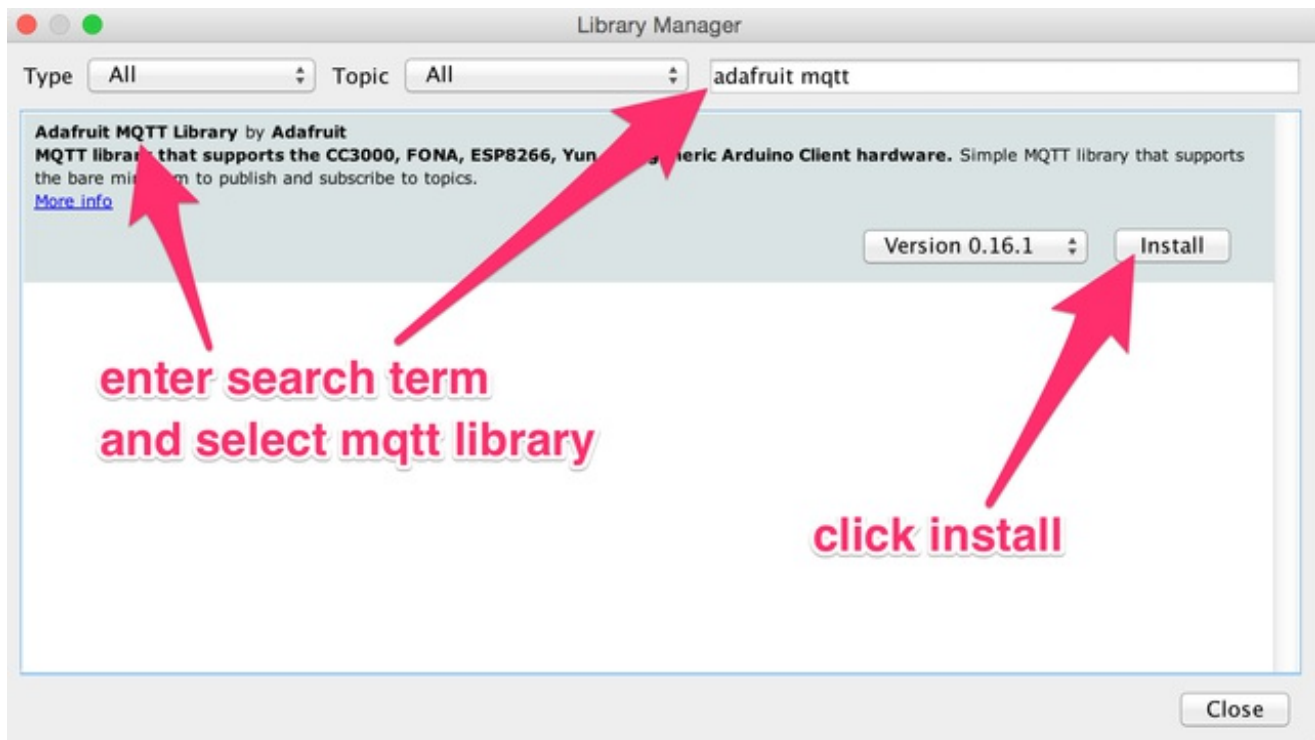
Now we will need to install the Adafruit IO, Adafruit MQTT, and ArduinoHttpClient libraries using the Arduino **Library Manager**. Navigate to the **Manage Libraries...** option in the **Sketch -> Include Library** menu.



Enter **Adafruit IO Arduino** into the search box, and click **Install** on the **Adafruit IO Arduino** library option to install version 2.1.1 or higher.

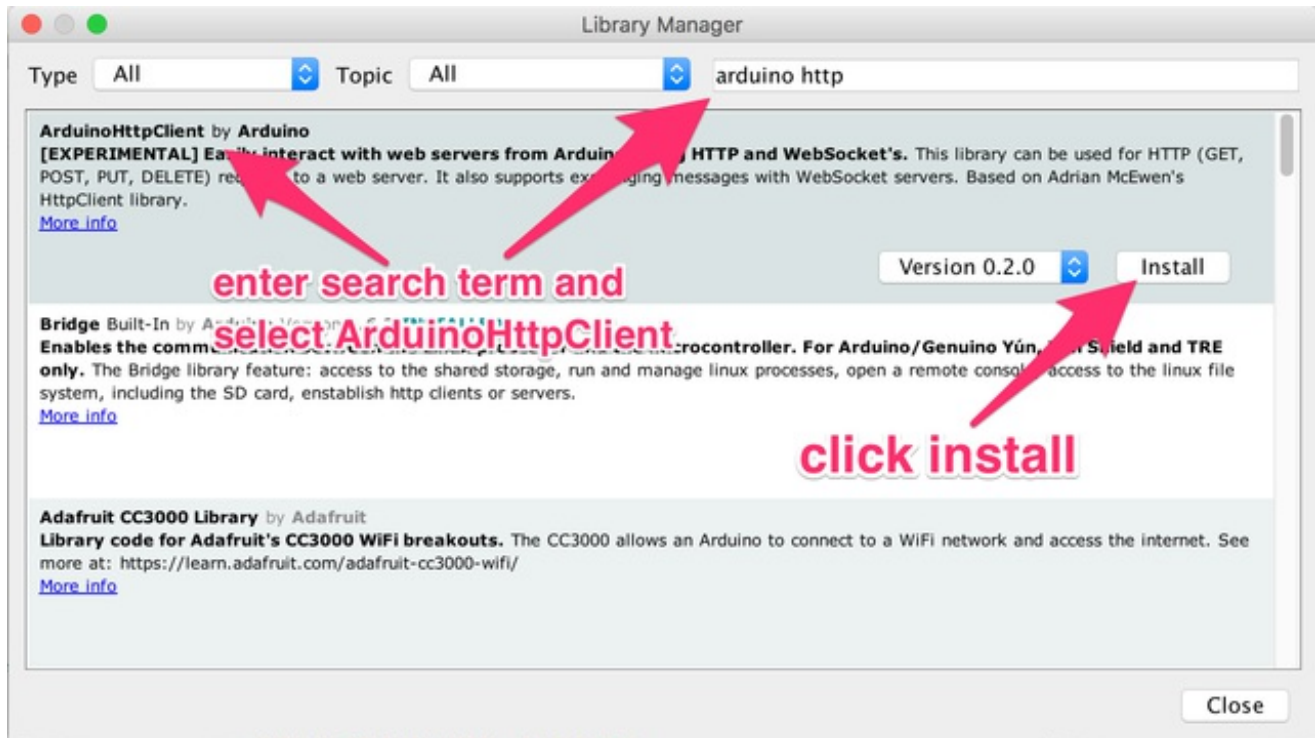


Enter **Adafruit MQTT** into the search box, and click **Install** on the **Adafruit MQTT** library option to install version 0.16.1 or higher.



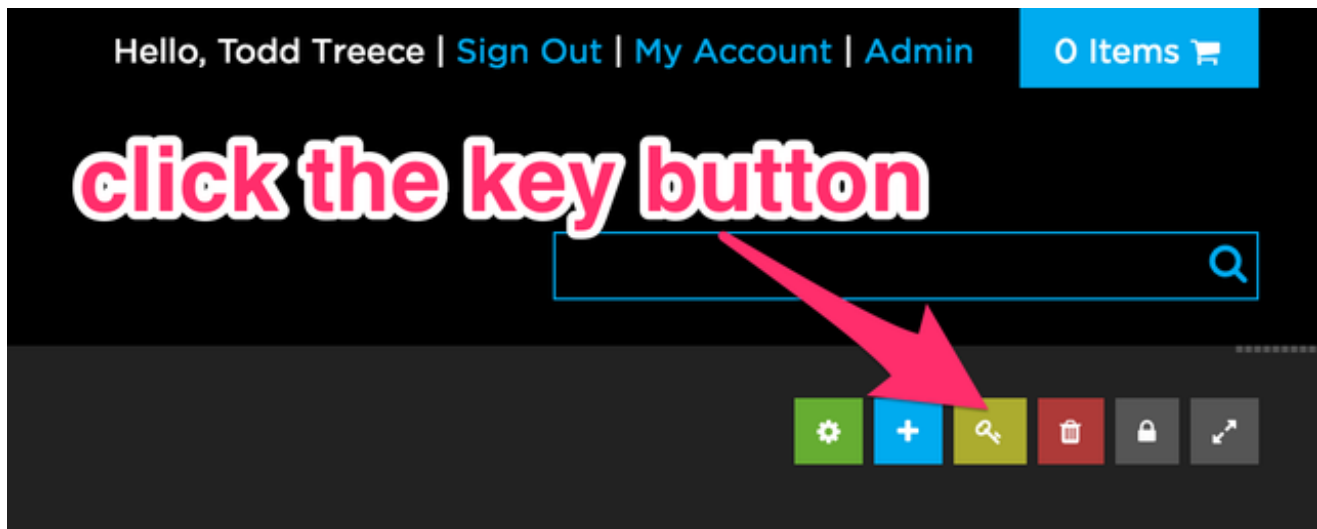
Enter **ArduinoHttpClient** into the search box, and click **Install** on the **ArduinoHttpClient** library option to install version 0.2.0 or higher.



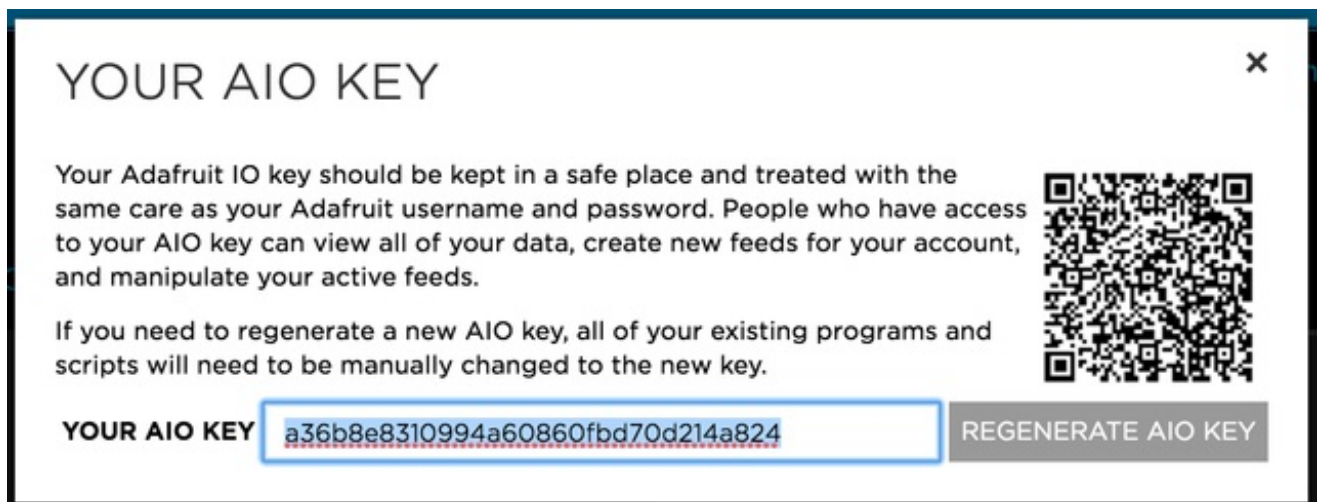


# Adafruit IO Setup

The first thing you will need to do is to login to your [Adafruit IO account](#) and get your Adafruit IO Key if you haven't already. Click the **key button** on the right hand side of the dashboard window to retrieve your key.



A window will pop up with your Adafruit IO Key. Keep a copy of this in a safe place. We'll need it later.



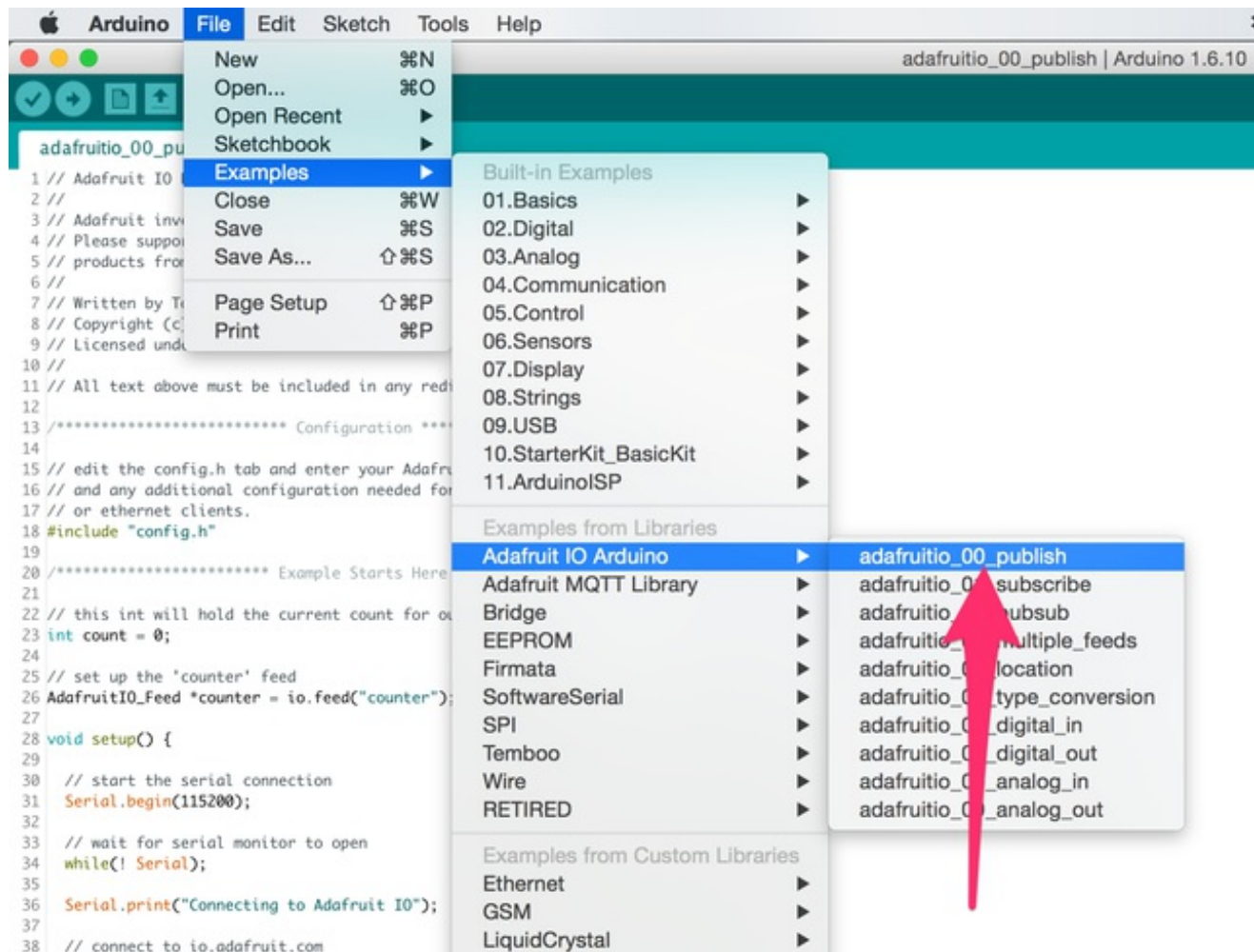
[illegible]

PRODUCT ID: 2680

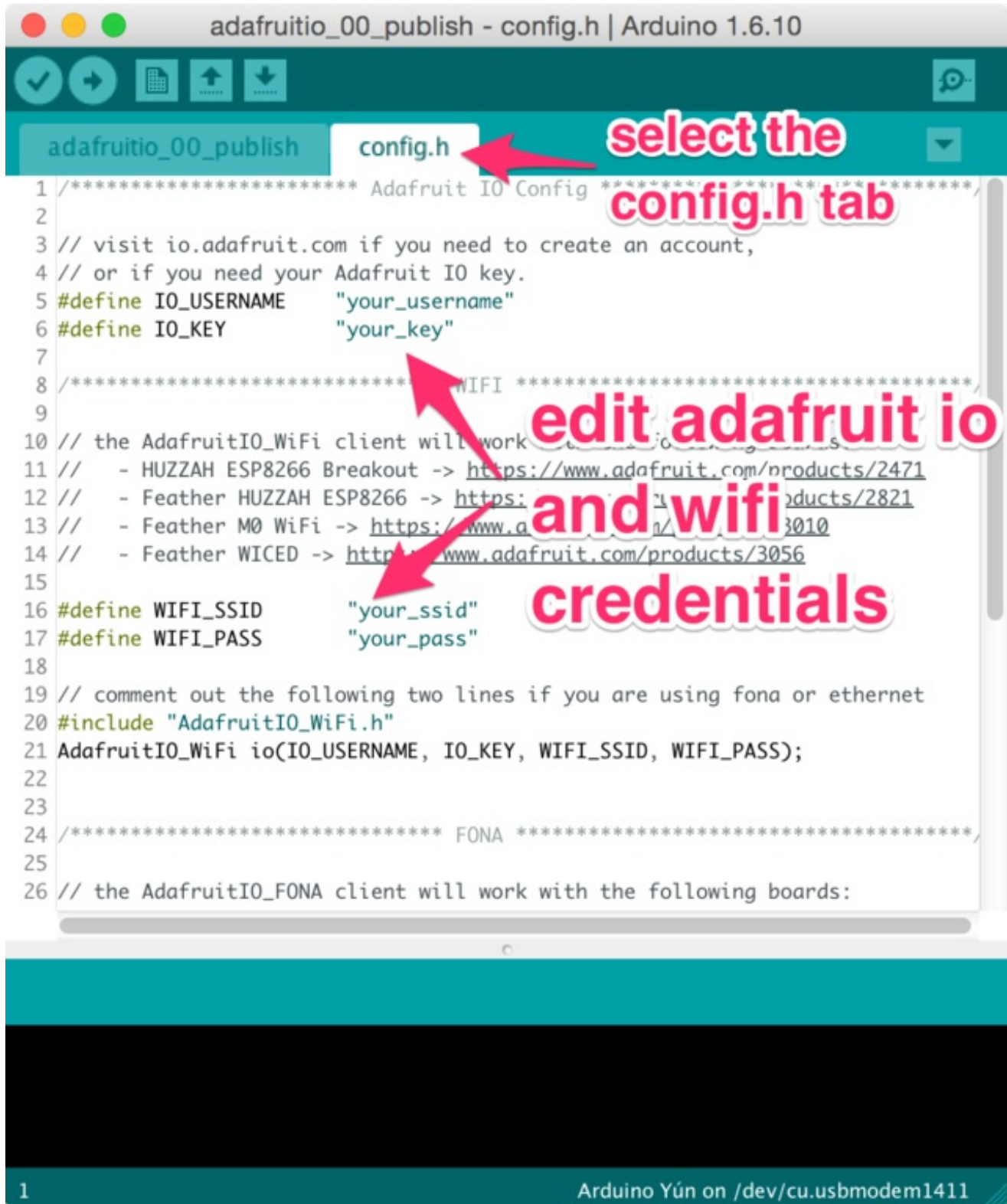
<http://adafru.it/g3d>

IN STOCK

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.



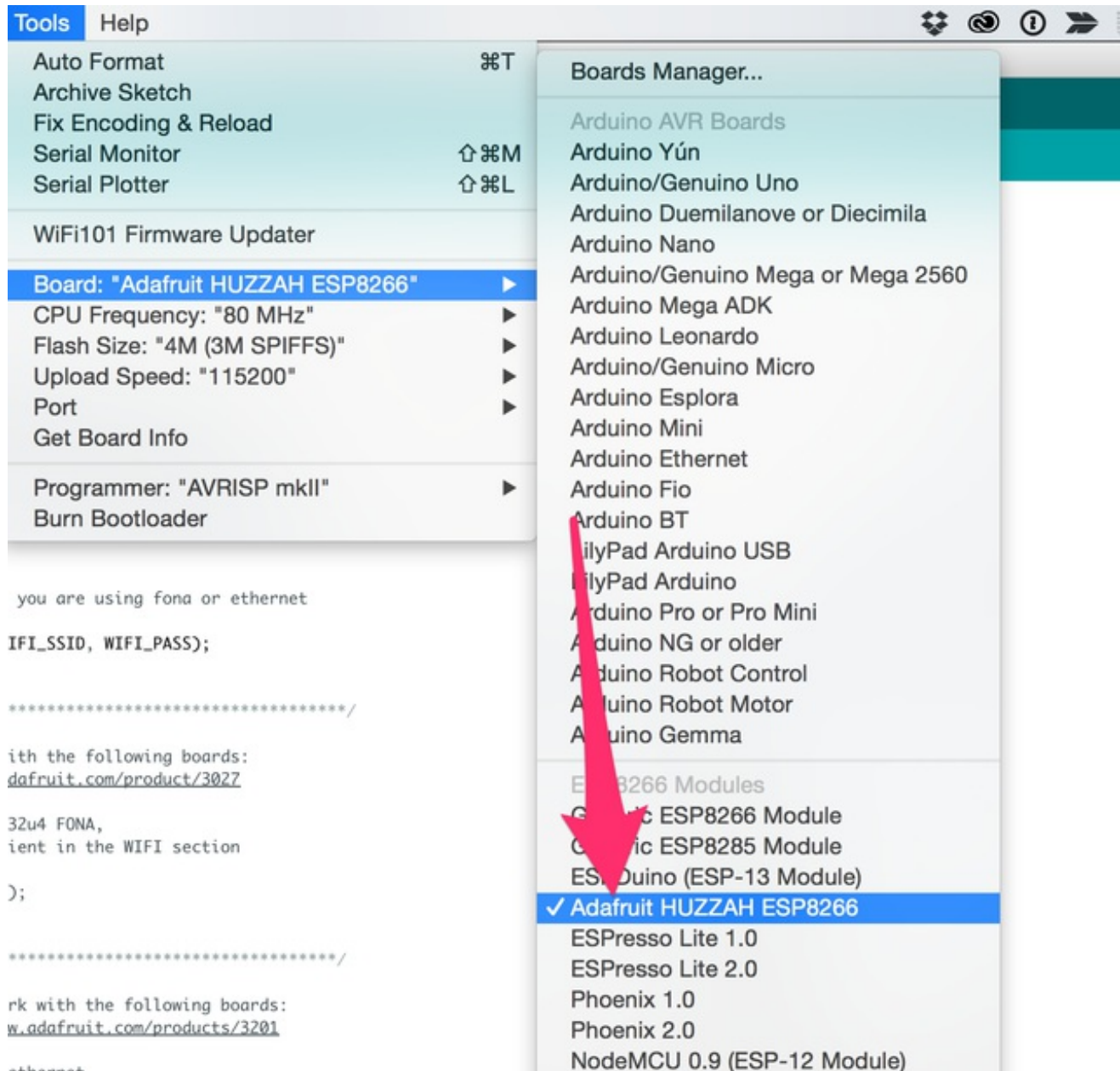
Click on the **config.h** tab, and replace the placeholders with your Adafruit IO credentials and WiFi connection info.



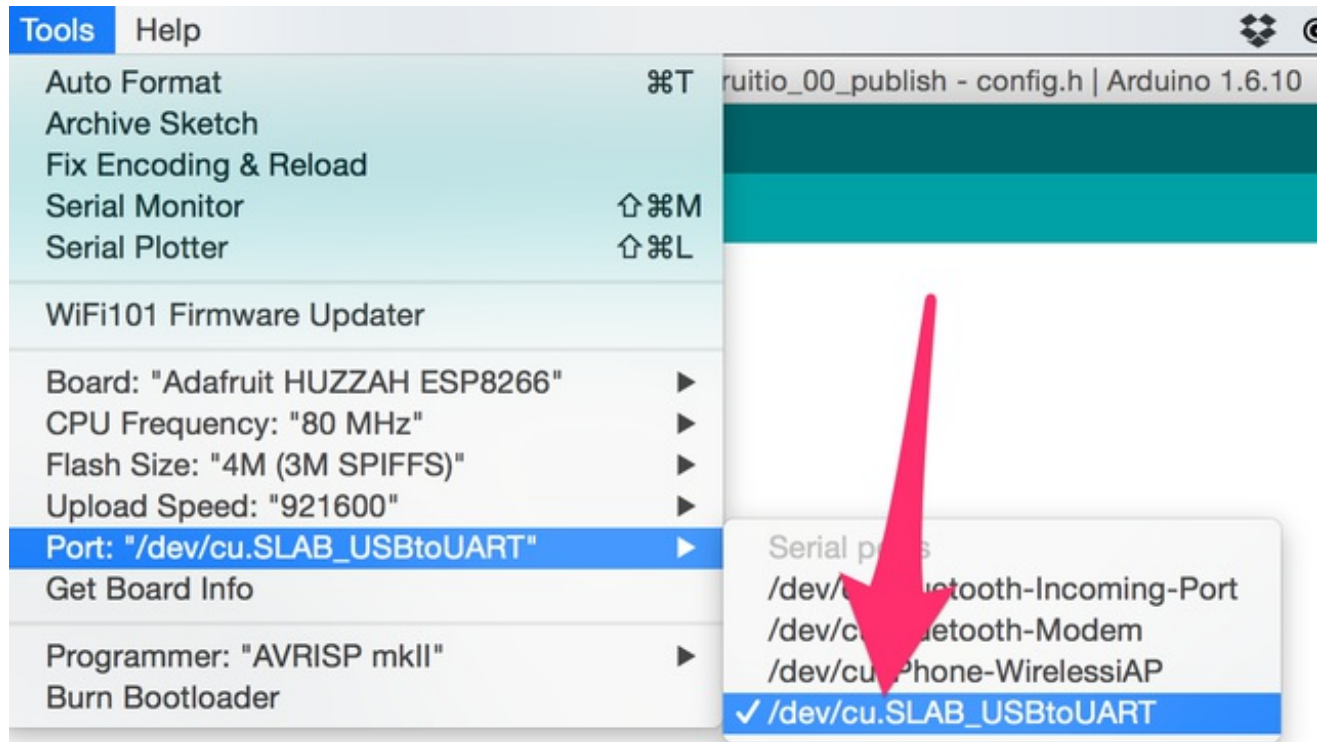
## Uploading the Sketch

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

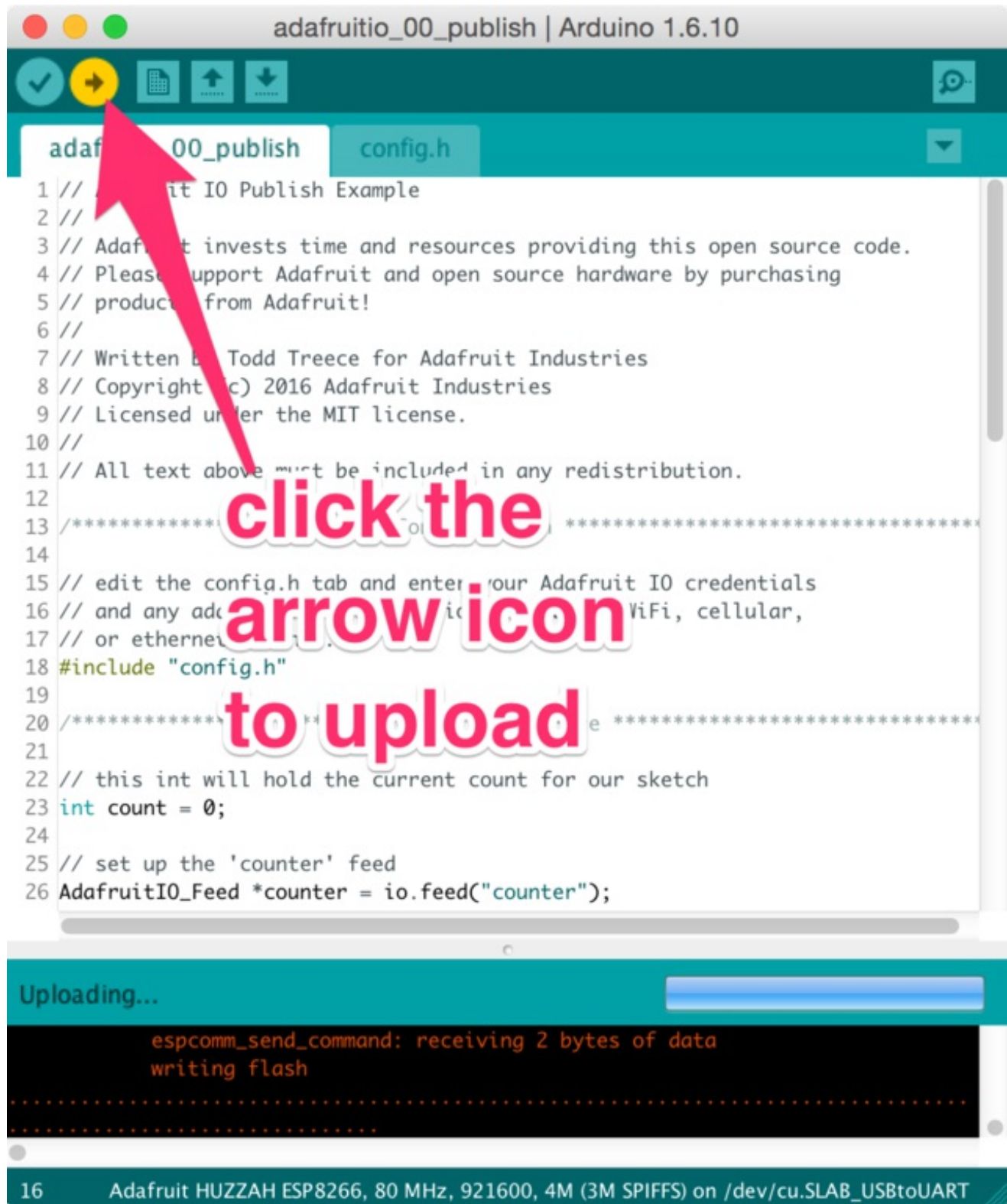




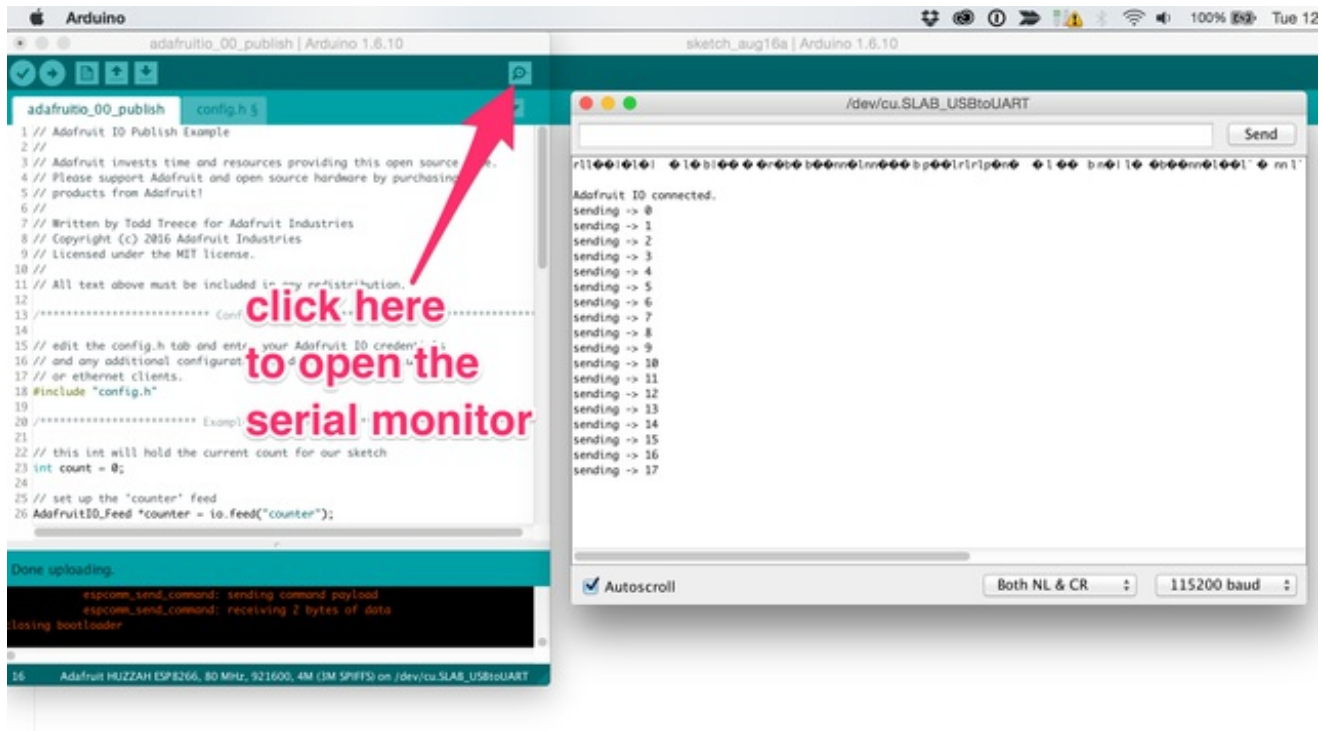
Then, select the proper COM port on Windows, or USB device on OS X.



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.

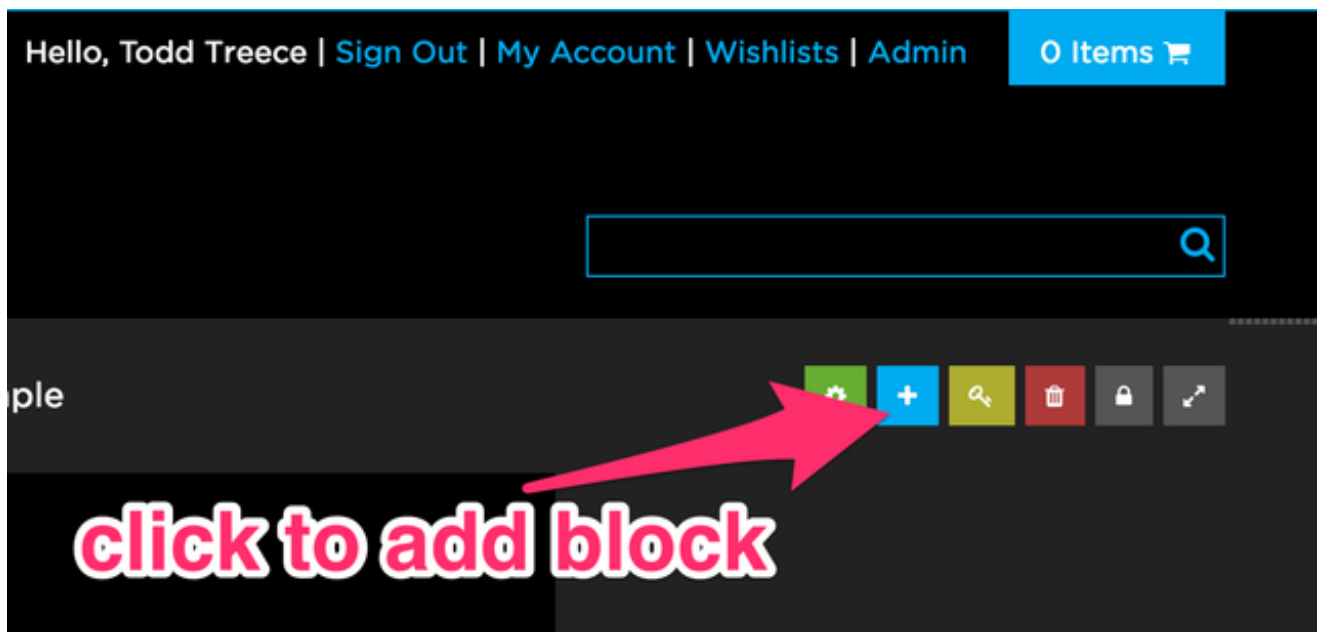


You can now click the serial monitor icon to view the output of the sketch. If everything goes as expected, you should see counter values being sent to Adafruit IO. If not, check your WiFi and Adafruit IO credentials in **config.h** and try uploading your sketch again using the process above.



## 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](https://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.




## CREATE A NEW BLOCK






A toggle button is useful if you have an ON or OFF type of state. You can configure what values are sent.

CREATE




A momentary button works similarly to a toggle button.

CREATE




The slider works well if you have a range of values to send.

CREATE




A gauge is a read only block type that shows a range of values.

CREATE



HELLO WORLD!

CREATE



```
2022-08-12 01:00:00 100
2022-08-12 01:00:01 100
2022-08-12 01:00:02 100
2022-08-12 01:00:03 100
2022-08-12 01:00:04 100
2022-08-12 01:00:05 100
2022-08-12 01:00:06 100
2022-08-12 01:00:07 100
2022-08-12 01:00:08 100
2022-08-12 01:00:09 100
2022-08-12 01:00:10 100
2022-08-12 01:00:11 100
2022-08-12 01:00:12 100
2022-08-12 01:00:13 100
2022-08-12 01:00:14 100
2022-08-12 01:00:15 100
2022-08-12 01:00:16 100
2022-08-12 01:00:17 100
2022-08-12 01:00:18 100
2022-08-12 01:00:19 100
2022-08-12 01:00:20 100
```

A stream block can be used to view the rolling history of data for multiple feeds.

CREATE

add a  
stream  
block

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



## CREATE A NEW BLOCK



**choose the counter feed and click next**

STEP 1: CHOOSE FEEDS

STEP 2: CHOOSE FEEDS

*i* Add up to 5 feeds

SELECT A FEED NAME

FEED/GROUP	LAST VALUE	RECORDED	ACTION
My Feeds			
counter	5368	10 minutes ago	<input type="button" value="CHOOSE"/>

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

# CREATE A NEW BLOCK



STEP 1: CHOOSE BLOCK TYPE		EDIT
STEP 2: CHOOSE FEEDS		EDIT
STEP 3: BLOCK SETTINGS		
<b>BLOCK TITLE</b>	<b>BLOCK PREVIEW</b>	
<input type="text" value="counter"/>		
<b>FONT SIZE</b>	<b>click 'create block'</b>	
<input type="text" value="SMALL"/>		
<b>FONT COLOR</b>		
<input type="text" value="GREEN"/>		
<b>SHOW FEED NAME?</b>		
<input type="text" value="YES"/>		
<b>SHOW TIMESTAMP?</b>		
<input type="text" value="NO"/>		
<b>SHOW ERRORS?</b>		
<input type="text" value="YES"/>		
		<input type="button" value="CANCEL"/> <input type="button" value="CREATE BLOCK"/>

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

counter

```
counter - 7
counter - 8
counter - 9
counter - 10
counter - 11
counter - 12
counter - 13
counter - 14
```

## 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.