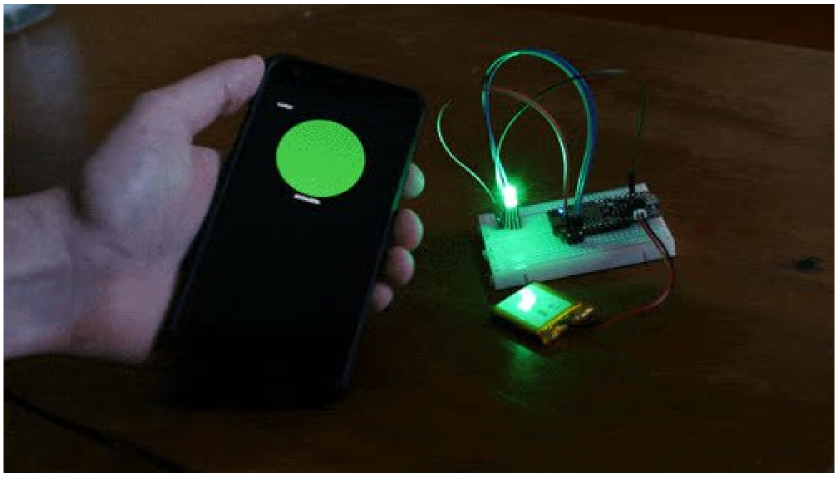
Adafruit IO Basics: Color

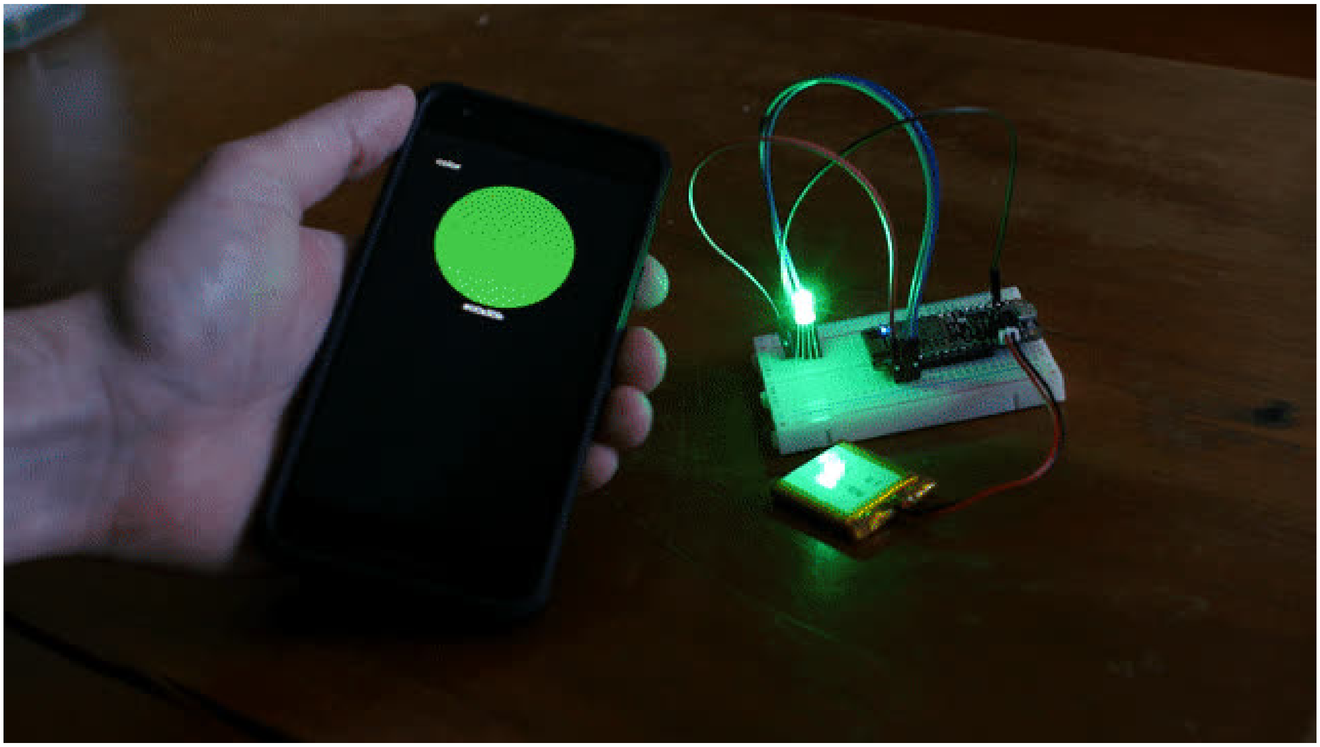
Created by Todd Treece



Last updated on 2017-02-20 04:33:33 PM UTC

Guide Contents

Overview



This guide is part of a series of guides that cover the basics of using Adafruit IO. It will show you how to send color data from Adafruit IO to a RGB LED.

If you haven't worked your way through the Adafruit IO feed and dashboard basics guides, you should do that before continuing with this guide so you have a basic understanding of Adafruit IO.

- Adafruit IO Basics: Feeds
- Adafruit IO Basics: Dashboards

You should go through the setup guides associated with your selected set of hardware, and make sure you have internet connectivity with the device before continuing. The following links will take you to the guides for your selected platform.

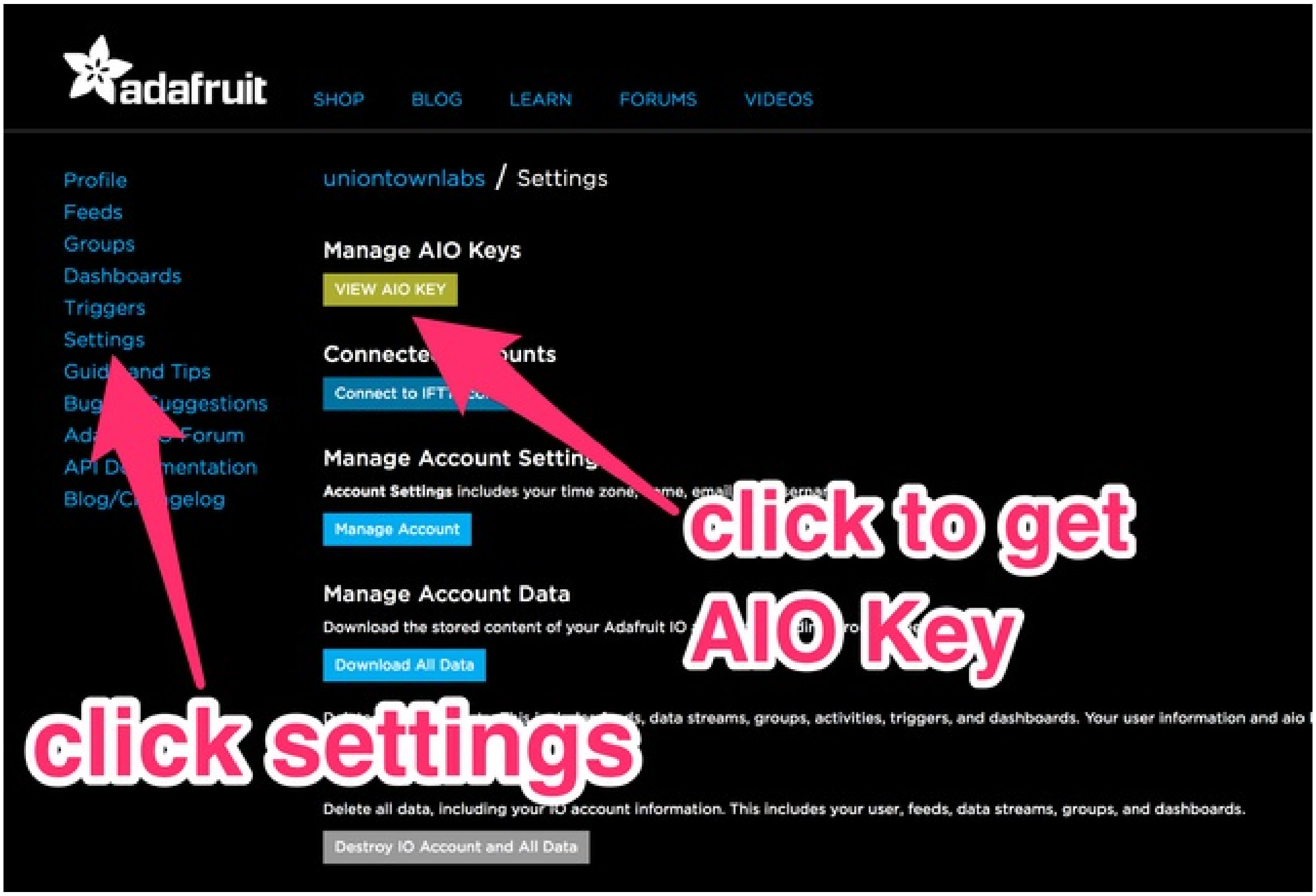
- Adafruit Feather HUZZAH ESP8266 Setup Guide

If you have went through all of the prerequisites for your selected hardware, you are now ready to move on to the Adafruit IO setup steps that are common between all of the hardware choices for this project. Let's get started!

Adafruit IO Setup

The first thing you will need to do is to login to[Adafruit IO](https://io.adafruit.com) and visit the **Settings** page.

Click the **VIEW AIO KEY** button to retrieve your key.



Account Settings includes your time zone, me, email ername

Download the stored content of your Adafruit IO ro

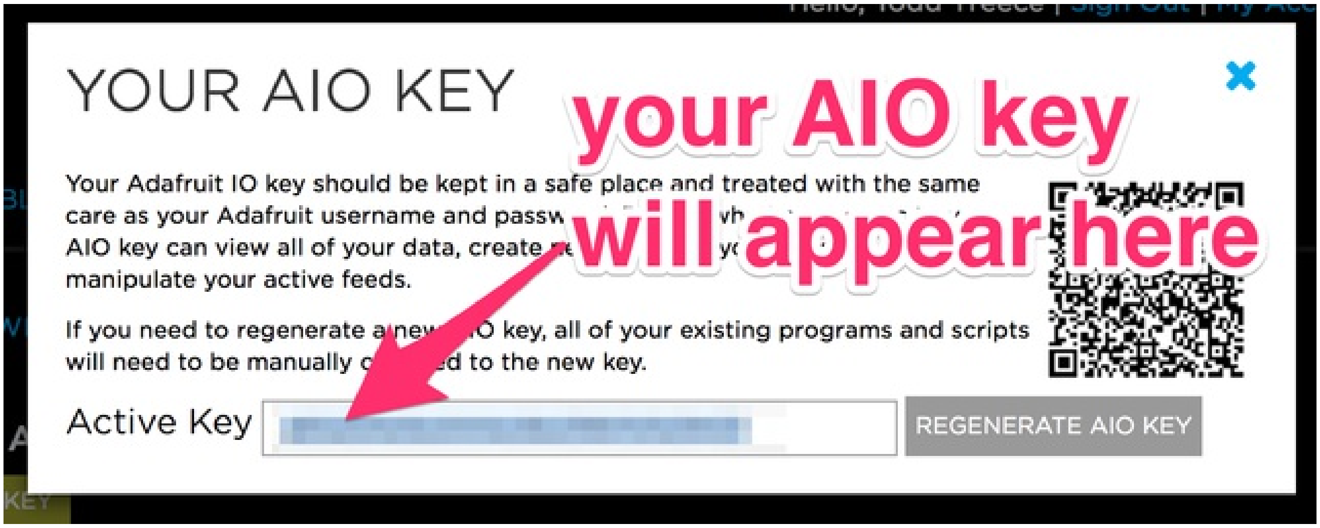
s, data streams, groups, activities, triggers, and dashboards. Your user information and aio

Delete all data, including your IO account information. This includes your user, feeds, data streams, groups, and dashboards.

click to get AIO Key

click settings

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



YOUR AIO KEY

your AIO key will appear here

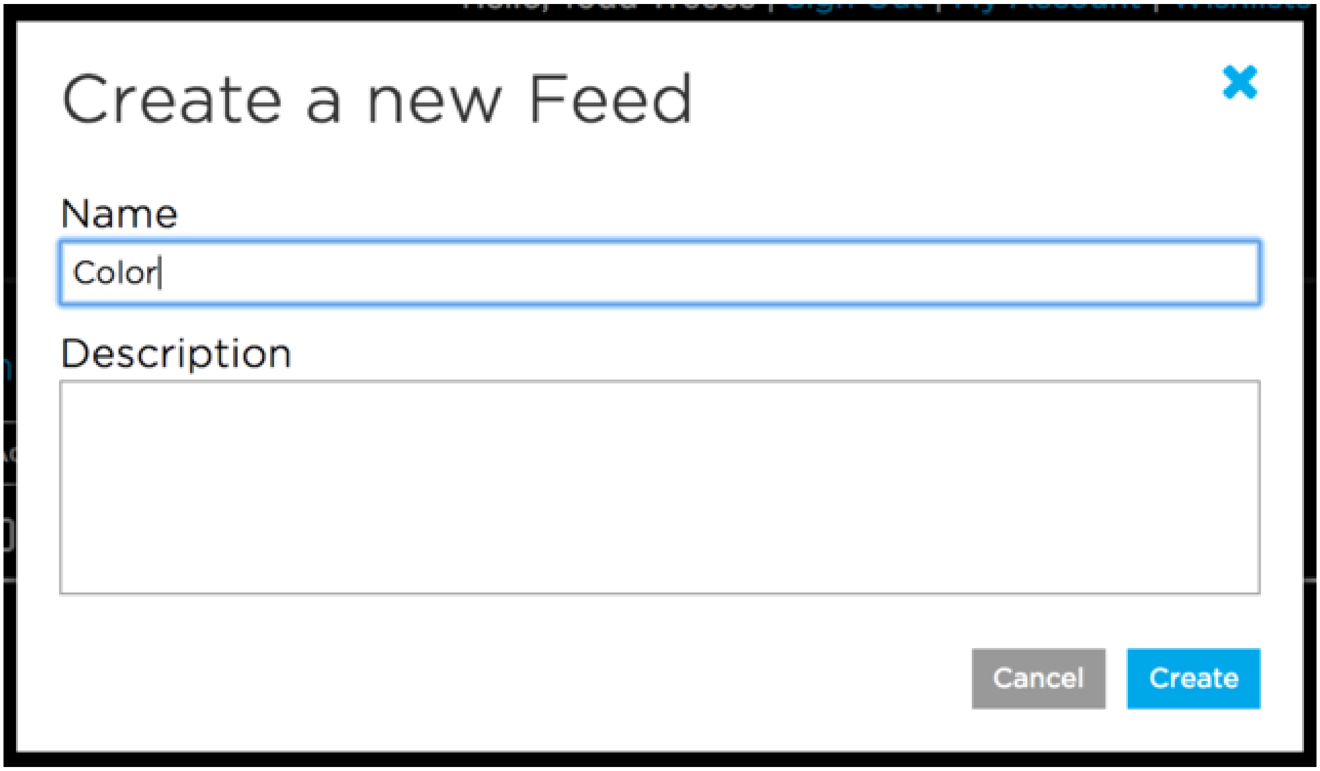
Your Adafruit IO key should be kept in a safe place and treated with the same care as your Adafruit username and passw… AIO key can view all of your data, create … manipulate your active feeds.

If you need to regenerate a new …O key, all of your existing programs and scripts will need to be manually c…d to the new key.

Active Key [redacted]

Creating the Color Feed

Next, you will need to create a feed called**Color**. If you need help getting started with creating feeds on Adafruit IO, check out the Adafruit IO Feed Basics guide (http://adafru.it/ioA).



Create a new Feed

Name

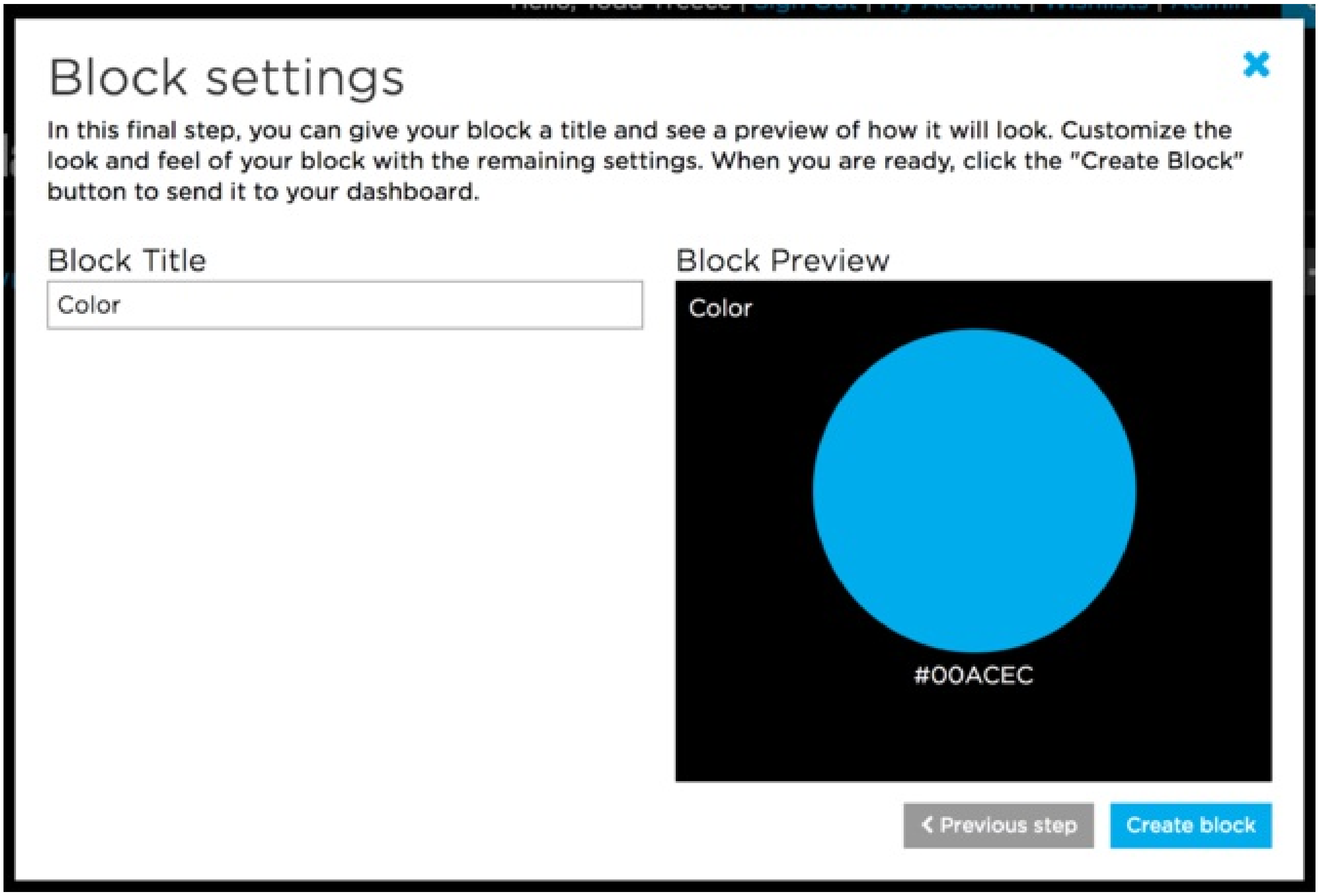
Color

Description

Adding the Color Block

Next, add a new Color block to a new or existing dashboard. Name the block whatever you would like, but make sure you have selected the ***Color*** feed as the data source for the block.

If you need help getting started with Dashboards on Adafruit IO, check out the Adafruit IO Dashboard Basics guide (http://adafru.it/f5m).



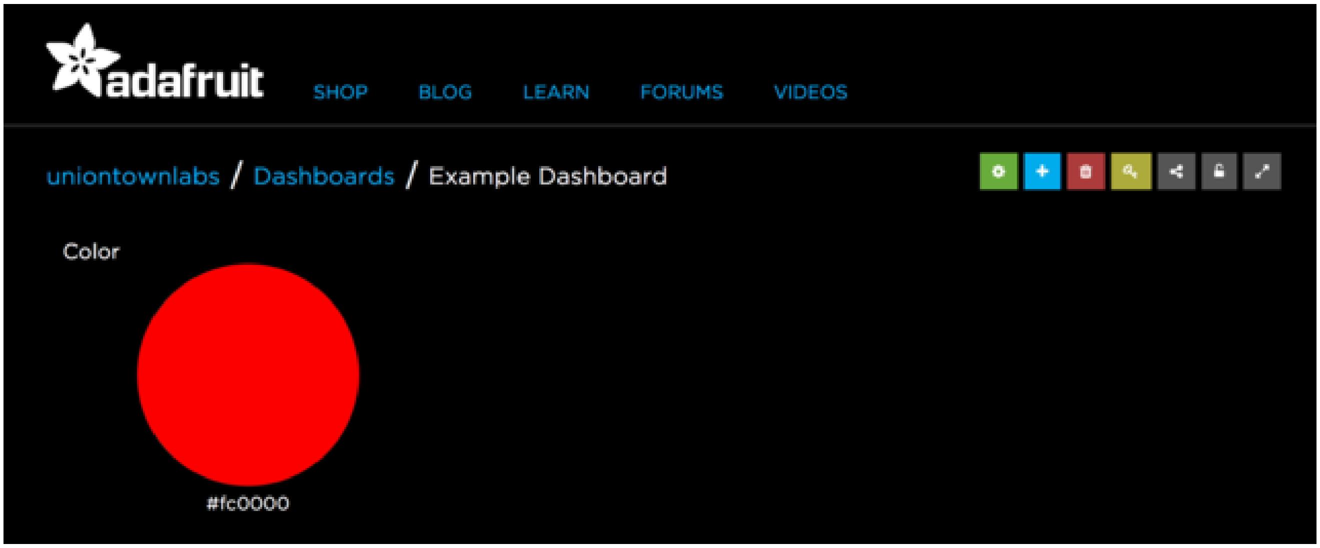
Block settings

In this final step, you can give your block a title and see a preview of how it will look. Customize the look and feel of your block with the remaining settings. When you are ready, click the "Create Block" button to send it to your dashboard.

Block Title: Color

Block Preview

When you are finished editing the form, click *Create Block* to add the new block to the dashboard.



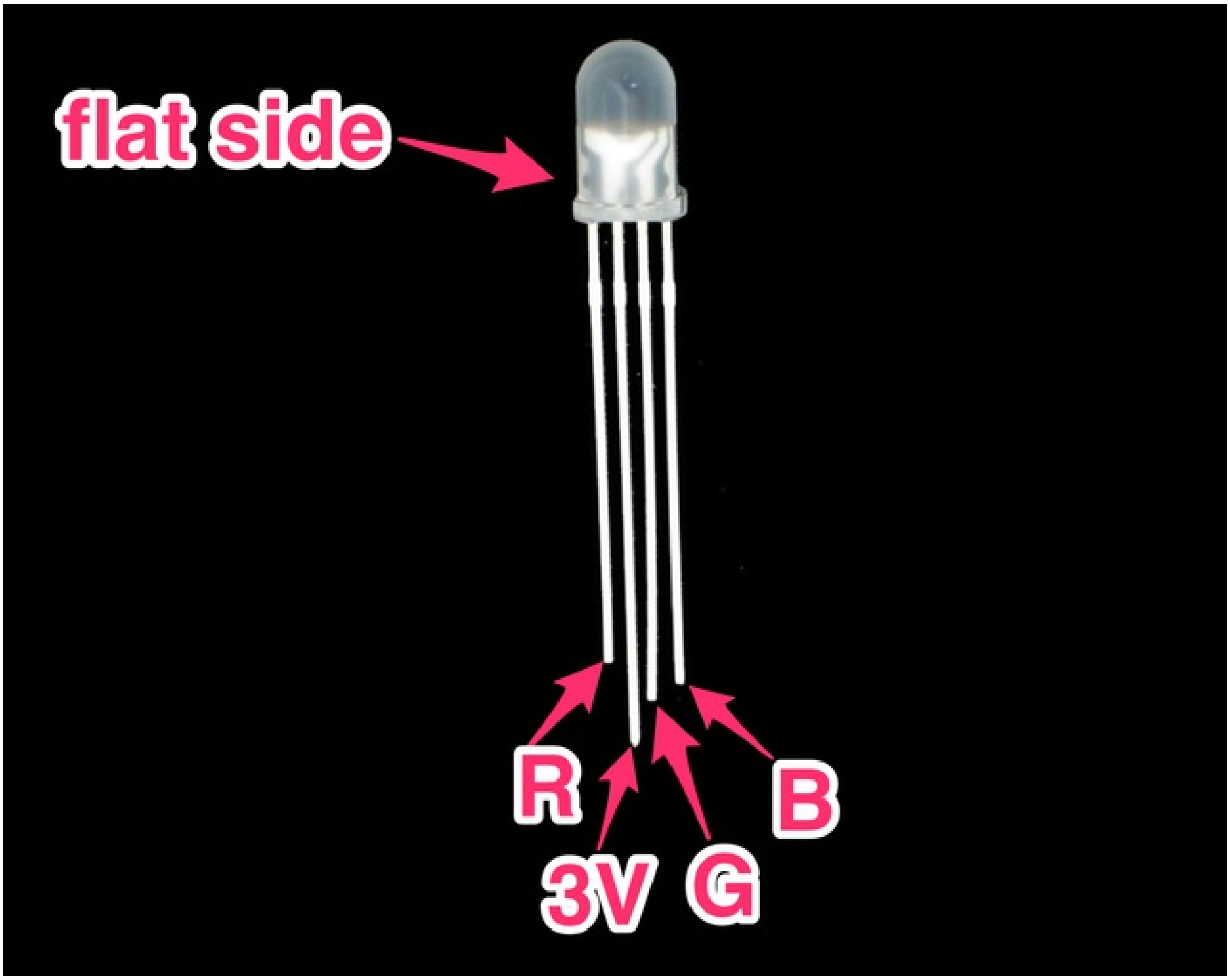
Next, we will look at wiring the circuit.

Wiring

You will need the following parts for this tutorial:

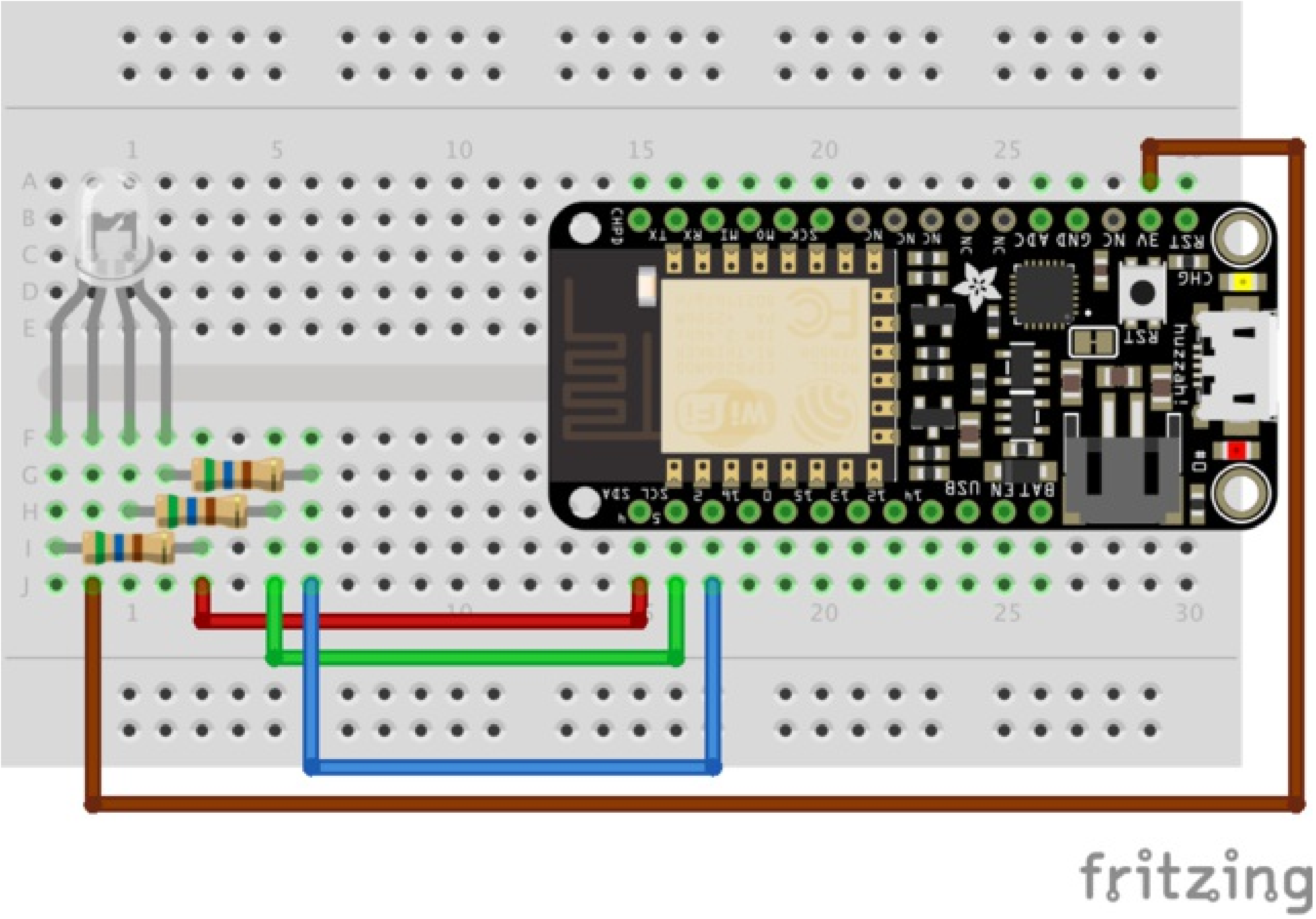
- **1x** Adafruit IO compatible Feather
- **1x** diffused RGB LED - common anode
- **3x** 560 ohm resistors
- **4x** jumper wires

We will be looking at the LED with the flat side of the LED on the left. If it's hard for you to see which side is the flat side, you can use the long pin as your guide. The long pin will be closer to the flat side of the LED.



We will be using a **Feather ESP8266** in this example, but you can adjust the pins in the sketch if you are using a different board. We will need to connect the following pins from the Feather to the resistors and RGB LED:

- Feather **pin 4** to one leg of a**560 ohm resistor**, and the other leg of the resistor to the **first leg** (R) of the LED
- Feather **3V** to the **second leg** (3V) of the LED
- Feather **pin 5** to one leg of a**560 ohm resistor**, and the other leg of the resistor to the **third leg** (G) of the LED
- Feather **pin 2** to one leg of a**560 ohm resistor**, and the other leg of the resistor to the **fourth leg** (B) of the LED



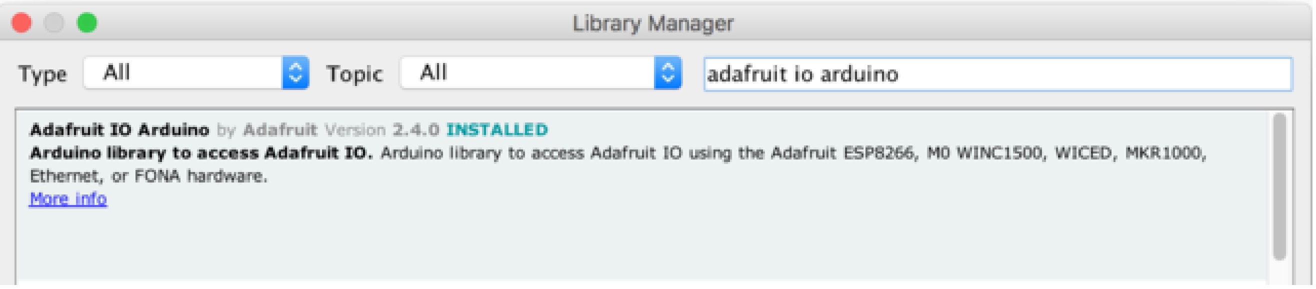
Next, let's look at the example sketch we will be using.

Arduino Setup

You should go through the setup guides associated with your selected set of hardware, and make sure you have internet connectivity with the device before continuing. The following links will take you to the guides for your selected platform.

- Adafruit Feather HUZZAH ESP8266 Setup Guide

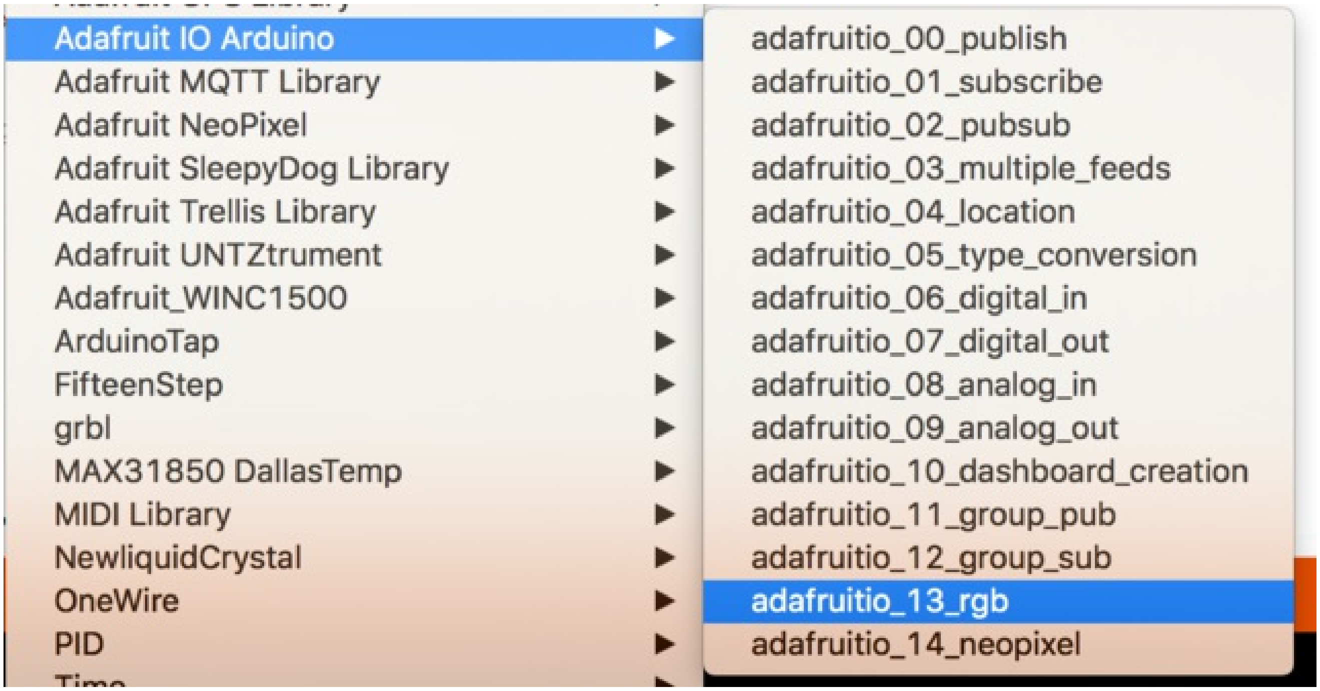
You will need to make sure you have at least**version 2.4.0** of the Adafruit IO Arduino library installed before continuing.



Type: All   Topic: All   adafruit io arduino

Adafruit IO Arduino by Adafruit Version 2.4.0 **INSTALLED**
Arduino library to access Adafruit IO. Arduino library to access Adafruit IO using the Adafruit ESP8266, M0 WINC1500, WICED, MKR1000, Ethernet, or FONA hardware.
More info

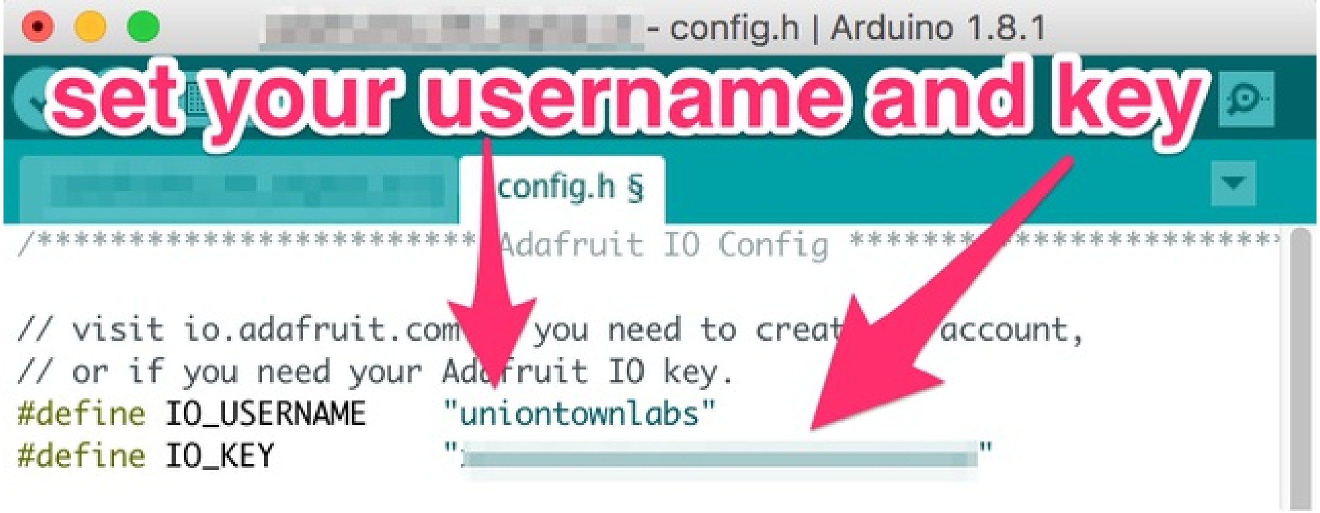
For this example, you will need to open the**adafruitio_13_rgb** example in the**Adafruit IO Arduino** library.



Next, we will look at the network configuration options in the sketch.

Network Config

To configure the network settings, click on the **config.h** tab in the sketch. You will need to set your Adafruit IO username in the **IO_USERNAME** define, and your Adafruit IO key in the **IO_KEY** define.

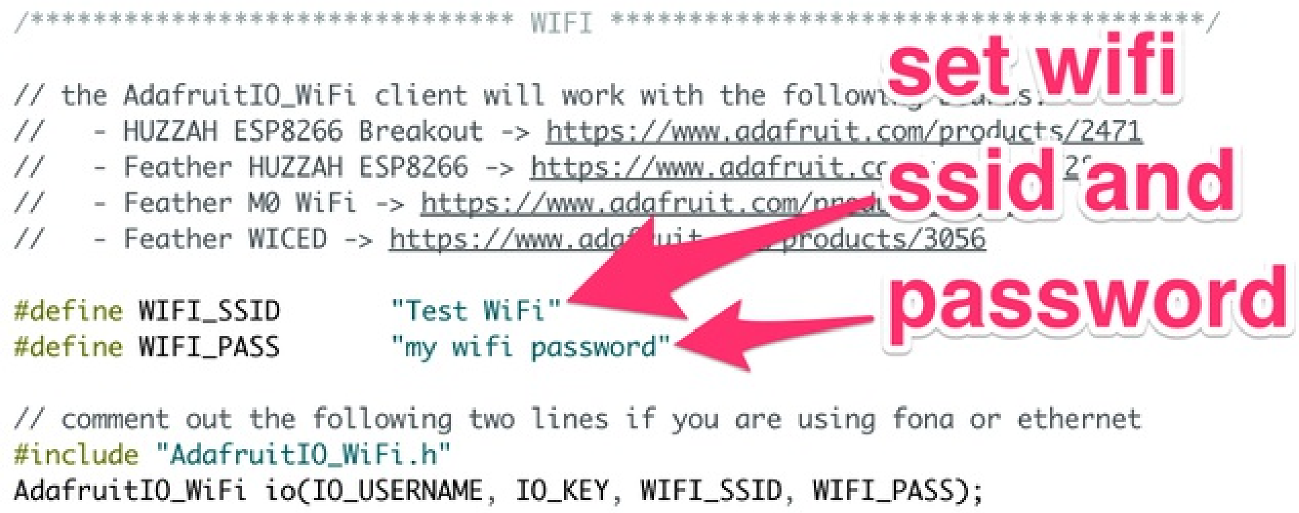


[redacted] - config.h | Arduino 1.8.1
set your username and key
config.h §

```
/************************ Adafruit IO Config *******************************/

// visit io.adafruit.com if you need to create an account,
// or if you need your Adafruit IO key.
#define IO_USERNAME    "uniontownlabs"
#define IO_KEY         "[redacted]"
```

WiFi Config

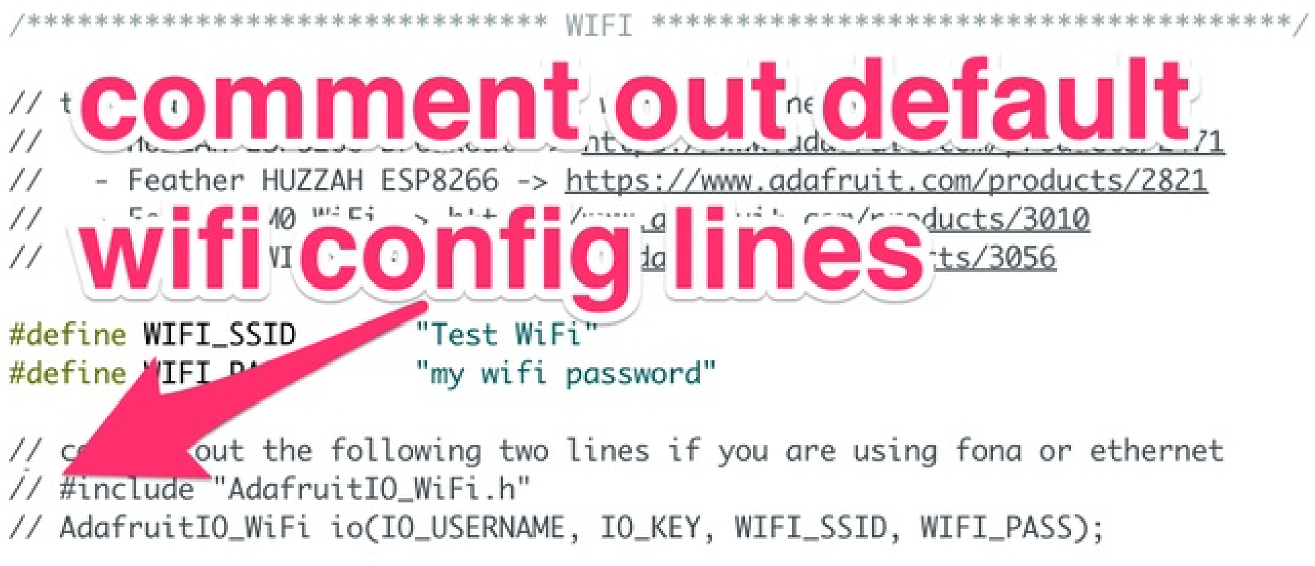
WiFi is enabled by default in **config.h** so if you are using one of the supported WiFi boards, you will only need to modify the **WIFI_SSID** and **WIFI_PASS** options in the **config.h** tab.



```
/******************************* WIFI **************************************/

// the AdafruitIO_WiFi client will work with the following boards:
//   - HUZZAH ESP8266 Breakout -> https://www.adafruit.com/products/2471
//   - Feather HUZZAH ESP8266 -> https://www.adafruit.com/products/2821
//   - Feather M0 WiFi -> https://www.adafruit.com/products/3010
//   - Feather WICED -> https://www.adafruit.com/products/3056

#define WIFI_SSID       "Test WiFi"
#define WIFI_PASS       "my wifi password"

// comment out the following two lines if you are using fona or ethernet
#include "AdafruitIO_WiFi.h"
AdafruitIO_WiFi io(IO_USERNAME, IO_KEY, WIFI_SSID, WIFI_PASS);
```

FONA Config

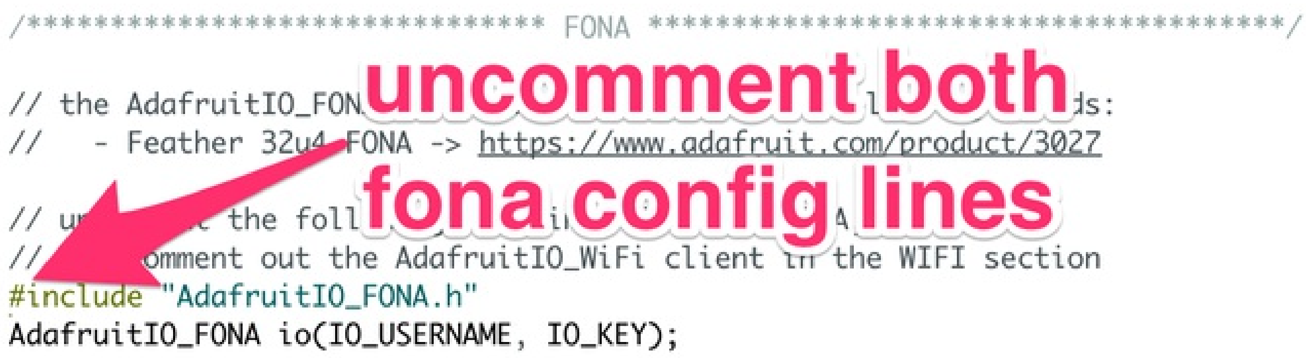
If you wish to use the FONA 32u4 Feather to connect to Adafruit IO, you will need to first comment out the WiFi support in **config.h**



comment out default wifi config lines

```
/******************************* WIFI **************************************/

// the AdafruitIO_WiFi client will work with the following boards:
//   - HUZZAH ESP8266 Breakout -> https://www.adafruit.com/products/2471
//   - Feather HUZZAH ESP8266 -> https://www.adafruit.com/products/2821
//   - Feather M0 WiFi -> https://www.adafruit.com/products/3010
//   - Feather WICED -> https://www.adafruit.com/products/3056

#define WIFI_SSID       "Test WiFi"
#define WIFI_PASS       "my wifi password"

// comment out the following two lines if you are using fona or ethernet
// #include "AdafruitIO_WiFi.h"
// AdafruitIO_WiFi io(IO_USERNAME, IO_KEY, WIFI_SSID, WIFI_PASS);
```

Next, remove the comments from both of the FONA config lines in the FONA section of **config.h** to enable FONA support.

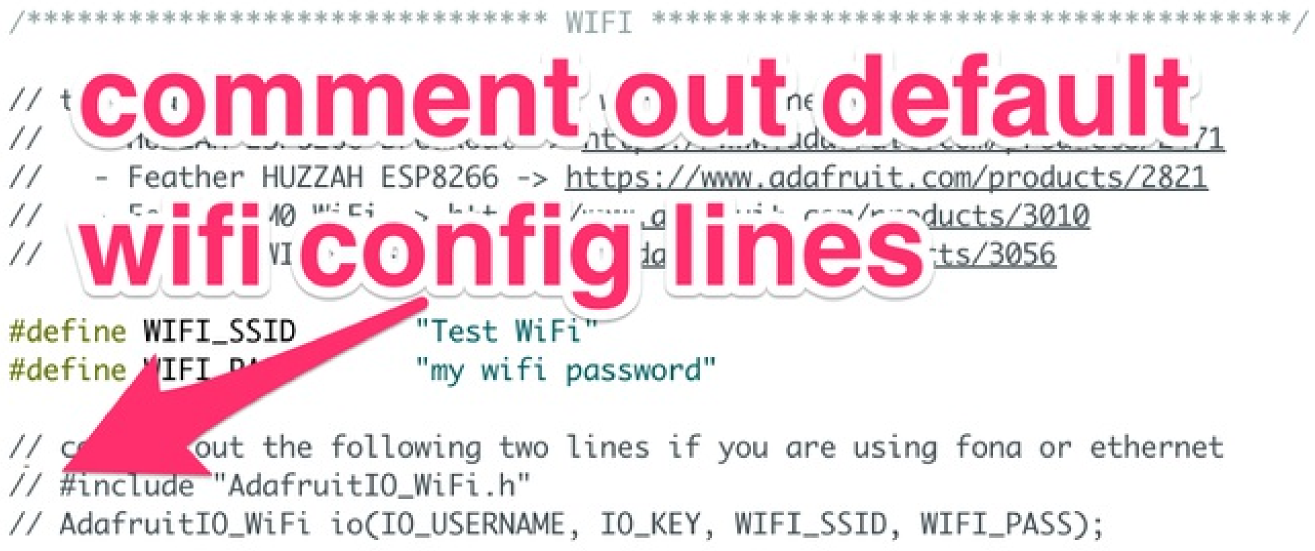


uncomment both fona config lines

```
/******************************* FONA **************************************/

// the AdafruitIO_FONA client will work with the following boards:
//   - Feather 32u4 FONA -> https://www.adafruit.com/product/3027

// uncomment the following two lines for 32u4 FONA,
// and comment out the AdafruitIO_WiFi client in the WIFI section
#include "AdafruitIO_FONA.h"
AdafruitIO_FONA io(IO_USERNAME, IO_KEY);
```

Ethernet Config

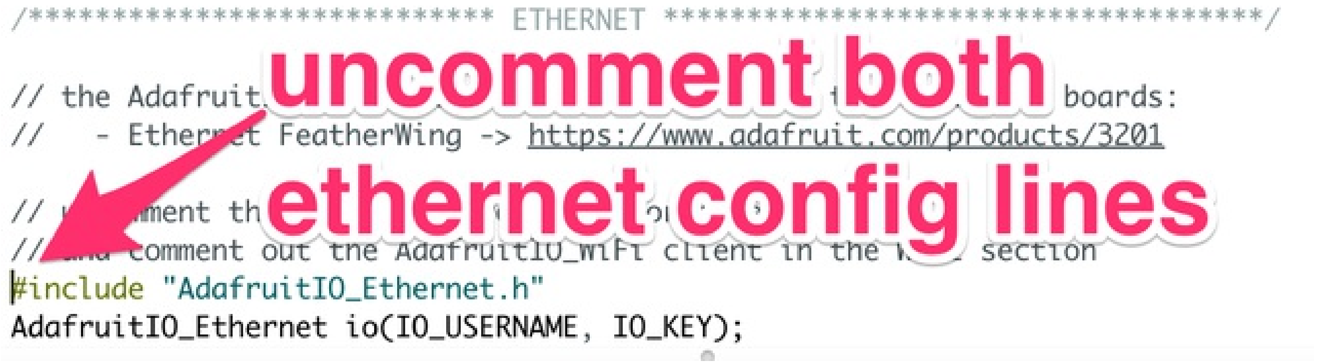
If you wish to use the Ethernet Wing to connect to Adafruit IO, you will need to first comment out the WiFi support in **config.h**



comment out default wifi config lines

```
/******************************* WIFI ***************************************/

// t
//                                        71
//   - Feather HUZZAH ESP8266 -> https://www.adafruit.com/products/2821
//                                    ducts/3010
//            WI                          ts/3056

#define WIFI_SSID       "Test WiFi"
#define              "my wifi password"

// c         out the following two lines if you are using fona or ethernet
// #include "AdafruitIO_WiFi.h"
// AdafruitIO_WiFi io(IO_USERNAME, IO_KEY, WIFI_SSID, WIFI_PASS);
```

Next, remove the comments from both of the Ethernet config lines in the Ethernet section of **config.h** to enable Ethernet Wing support.



uncomment both ethernet config lines

```
/**************************** ETHERNET ************************************/

// the Adafruit                          boards:
//   - Ether   t FeatherWing -> https://www.adafruit.com/products/3201

//    ment th
//     comment out the AdafruitIO_WiFi client in the     section
#include "AdafruitIO_Ethernet.h"
AdafruitIO_Ethernet io(IO_USERNAME, IO_KEY);
```

Next, we will look at how the example sketch works.

Code

The **adafruitio_13_rgb** example uses pins 4, 5, & 2 for red, green, and blue by default. These pins can be modified by changing the **RED_PIN**, **GREEN_PIN**, and **BLUE_PIN** defines. You will need to choose pins on your board that support PWM output.

```
/************************ Example Starts Here *******************************/

// default PWM pins for ESP8266.
// you should change these to match PWM pins on other platforms.
#define RED_PIN   4
#define GREEN_PIN 5
#define BLUE_PIN  2
```

The next chunk of code sets up an instance of the**color** feed.

```
// set up the 'color' feed
AdafruitIO_Feed *color = io.feed("color");
```

In the setup function, we connect to Adafruit IO, and attach a function called **handleMessage** to the **color** feed. This function will be called whenever your device receives messages for that feed.

We also use a special function for the ESP8266 platform that sets the**analogWrite** range to 0-255. This will be ignored on all other platforms.

The code will wait until you have a valid connection to Adafruit IO before continuing with the sketch. If you have any issues connecting, check **config.h** for any typos in your username or key.

```
void setup() {

  // start the serial connection
  Serial.begin(115200);

  // wait for serial monitor to open
  while(! Serial);

  // connect to io.adafruit.com
  Serial.print("Connecting to Adafruit IO");
  io.connect();

  // set up a message handler for the 'color' feed.
  // the handleMessage function (defined below)
  // will be called whenever a message is
  // received from adafruit io.
  color->onMessage(handleMessage);

  // wait for a connection
  while(io.status() < AIO_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  // we are connected
  Serial.println();
  Serial.println(io.statusText());

  // set analogWrite range for ESP8266
  #ifdef ESP8266
    analogWriteRange(255);
  #endif

}
```

Next, we have the main `loop()` function. The first line of the loop function calls `io.run();` this line will need to be present at the top of your loop in every sketch. It helps keep your device connected to Adafruit IO, and processes any incoming data.

```
void loop() {

  // io.run(); is required for all sketches.
  // it should always be present at the top of your loop
  // function. it keeps the client connected to
  // io.adafruit.com, and processes any incoming data.
  io.run();

}
```

The final chunk of code is the **handleMessage** function. This is the function that is called whenever the **color** feed gets a message.

We use the **data->toRed()**, **data->toGreen()**, and **data->toBlue()** functions to convert the incoming hex color values to integers that will be compatible with **analogWrite**.

Because we are using a **common anode** RGB LED, we will need to flip the incoming RGB values. We do this by subtracting the values from 255 before sending the values to **analogWrite**.

If you are using a **common cathode** RGB LED, you can send the values directly to **analogWrite** from **data->toRed()**, **data->toGreen()**, and **data->toBlue()**.

```
// this function is called whenever a 'color' message
// is received from Adafruit IO. it was attached to
// the color feed in the setup() function above.
void handleMessage(AdafruitIO_Data *data) {

  // print RGB values and hex value
  Serial.println("Received:");
  Serial.print("  - R: ");
  Serial.println(data->toRed());
  Serial.print("  - G: ");
  Serial.println(data->toGreen());
  Serial.print("  - B: ");
  Serial.println(data->toBlue());
  Serial.print("  - HEX: ");
  Serial.println(data->value());

  // invert RGB values for common anode LEDs
  analogWrite(RED_PIN, 255 - data->toRed());
  analogWrite(GREEN_PIN, 255 - data->toGreen());
  analogWrite(BLUE_PIN, 255 - data->toBlue());

}
```

Upload the sketch to your board, and open the Arduino Serial Monitor. Your board should now connect to Adafruit IO.

```
Connecting to Adafruit IO....

Adafruit IO connected.
```

You can now use the color block on Adafruit IO to set a color, and you should see something resembling the following in the Arduino Serial Monitor.

```
Received:
  - R: 0
  - G: 0
  - B: 0
  - HEX: #000000
Received:
  - R: 210
  - G: 31
  - B: 31
  - HEX: #d21f1f
```

You should also see the RGB LED update with the color you picked in the color block.