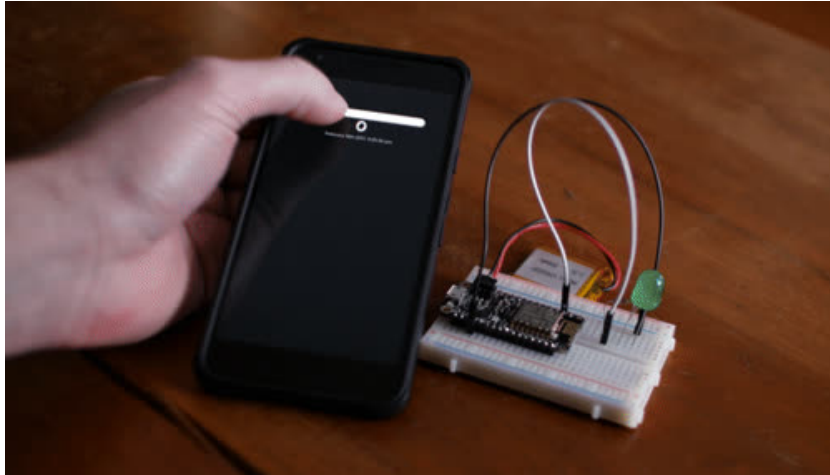


□

Adafruit IO Basics: Analog Output

Created by Todd Treece

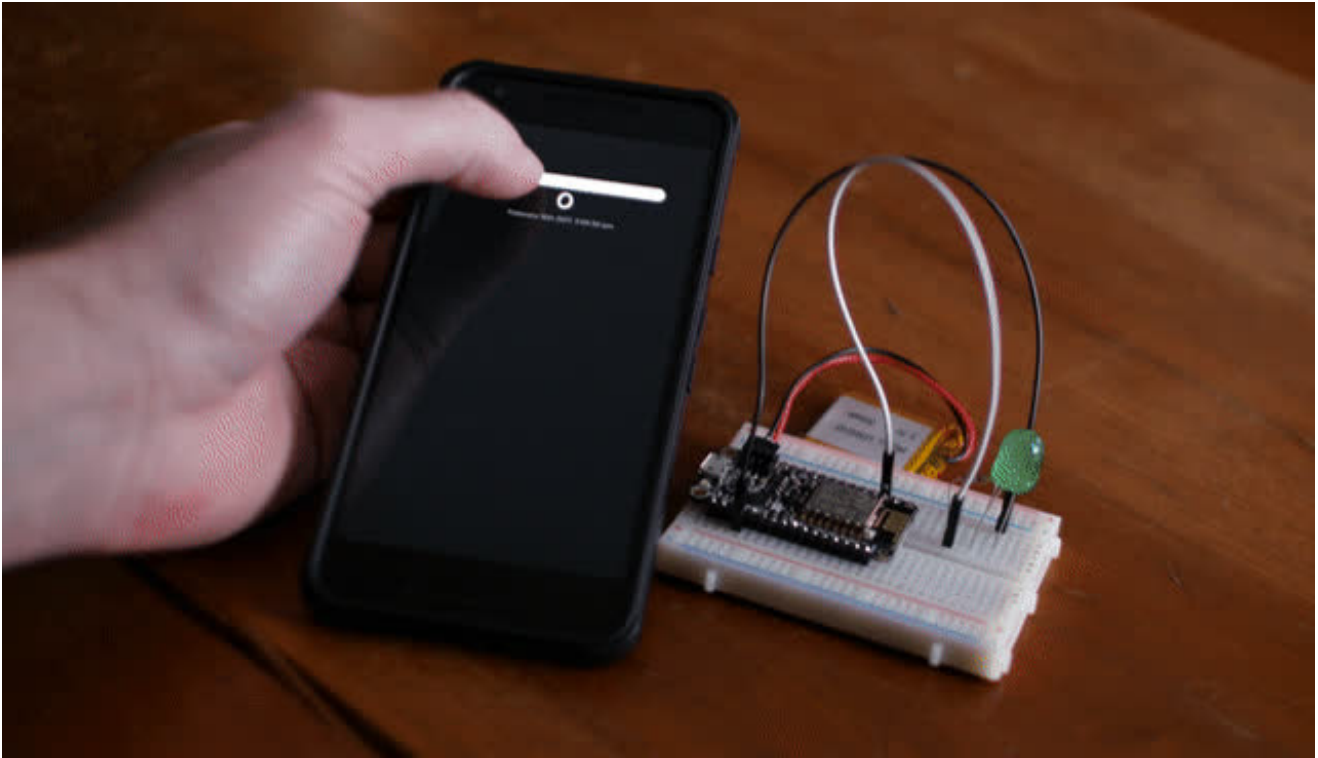


Last updated on 2017-02-16 10:06:14 PM UTC

Guide Contents

Guide Contents	2
Overview	3
Adafruit IO Setup	5
Creating the Analog Feed	6
Adding the Slider Block	6
Wiring	9
Arduino Setup	11
Network Config	12
WiFi Config	12
FONA Config	13
Ethernet Config	13
Code	15

Overview



This guide is part of a series of guides that cover the basics of using Adafruit IO. It will show you how to dim a LED from Adafruit IO using any modern web browser.

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.

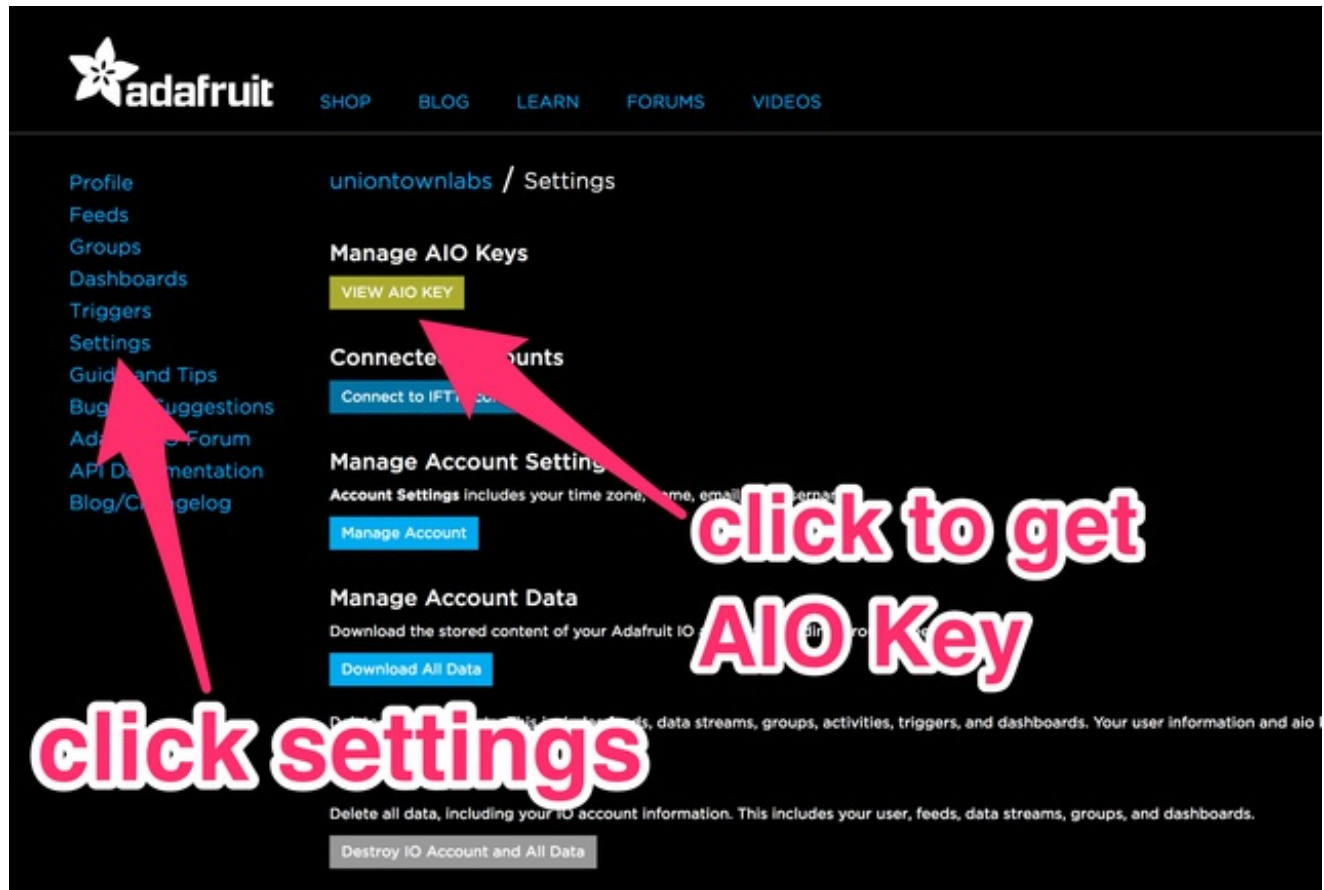
- [Adafruit Feather Huzzah ESP8266 Setup Guide](#)

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!

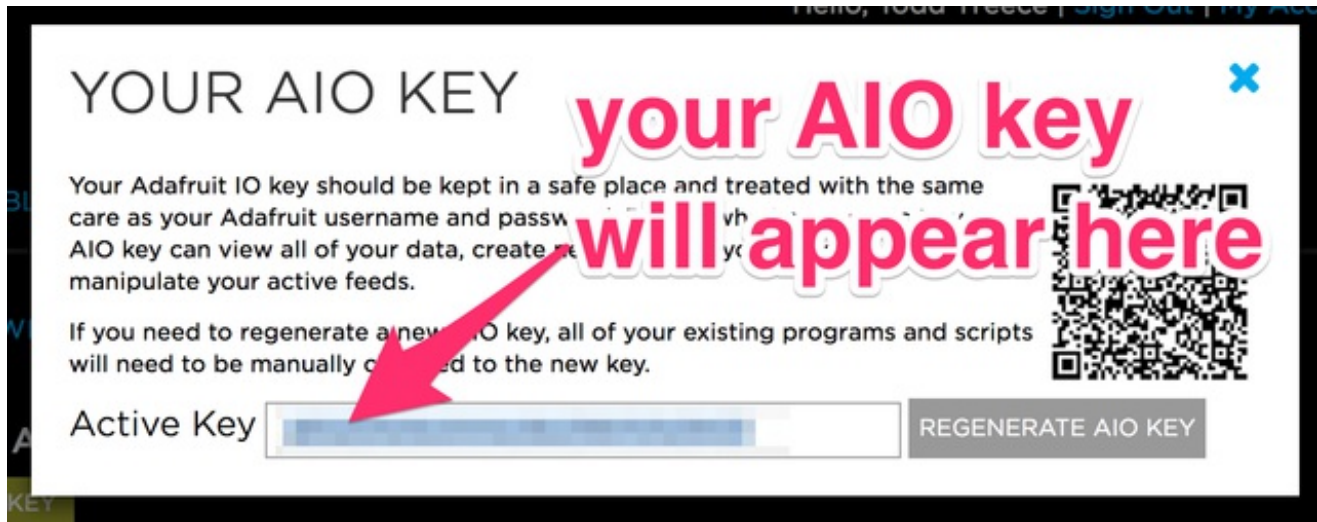
Adafruit IO Setup

The first thing you will need to do is to login to [Adafruit IO](#) and visit the **Settings** page.

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

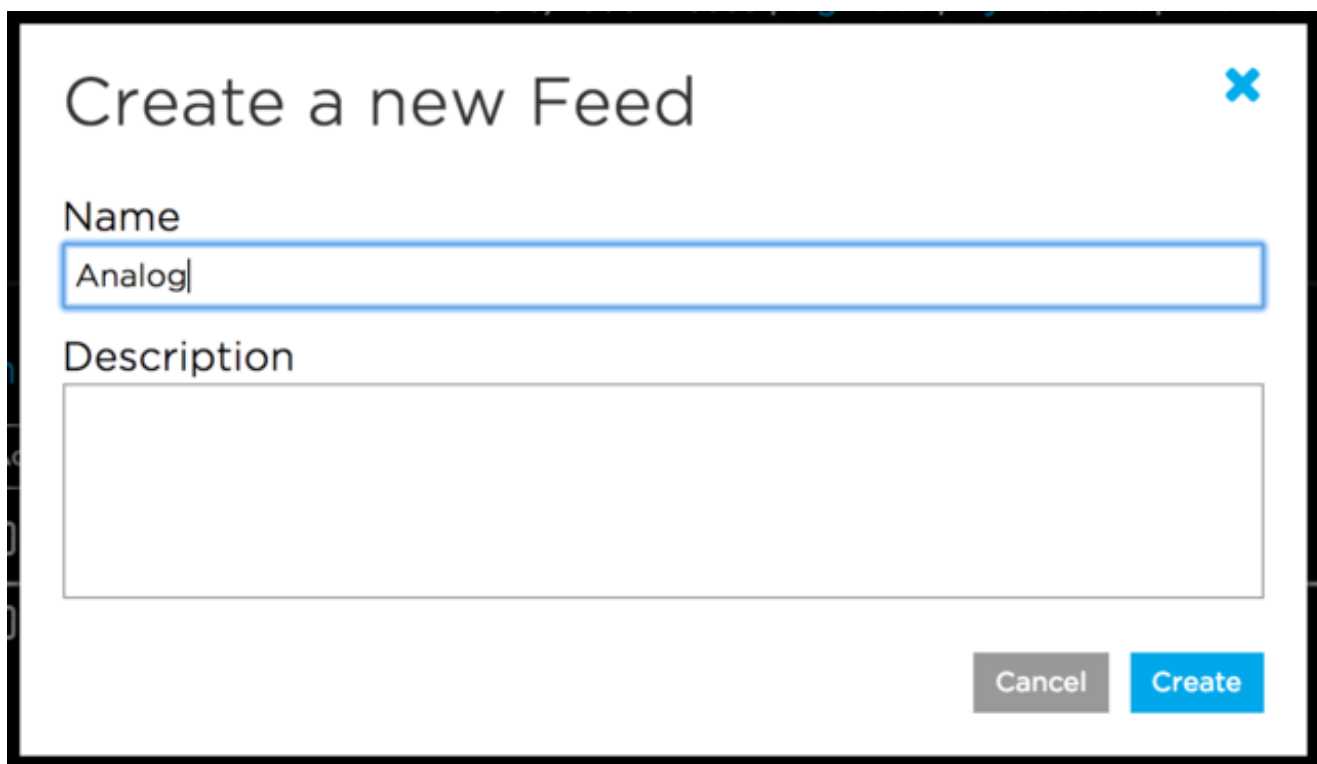


A window will pop up with your Adafruit IO. Keep a copy of this in a safe place. We'll need it later.



Creating the Analog Feed

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

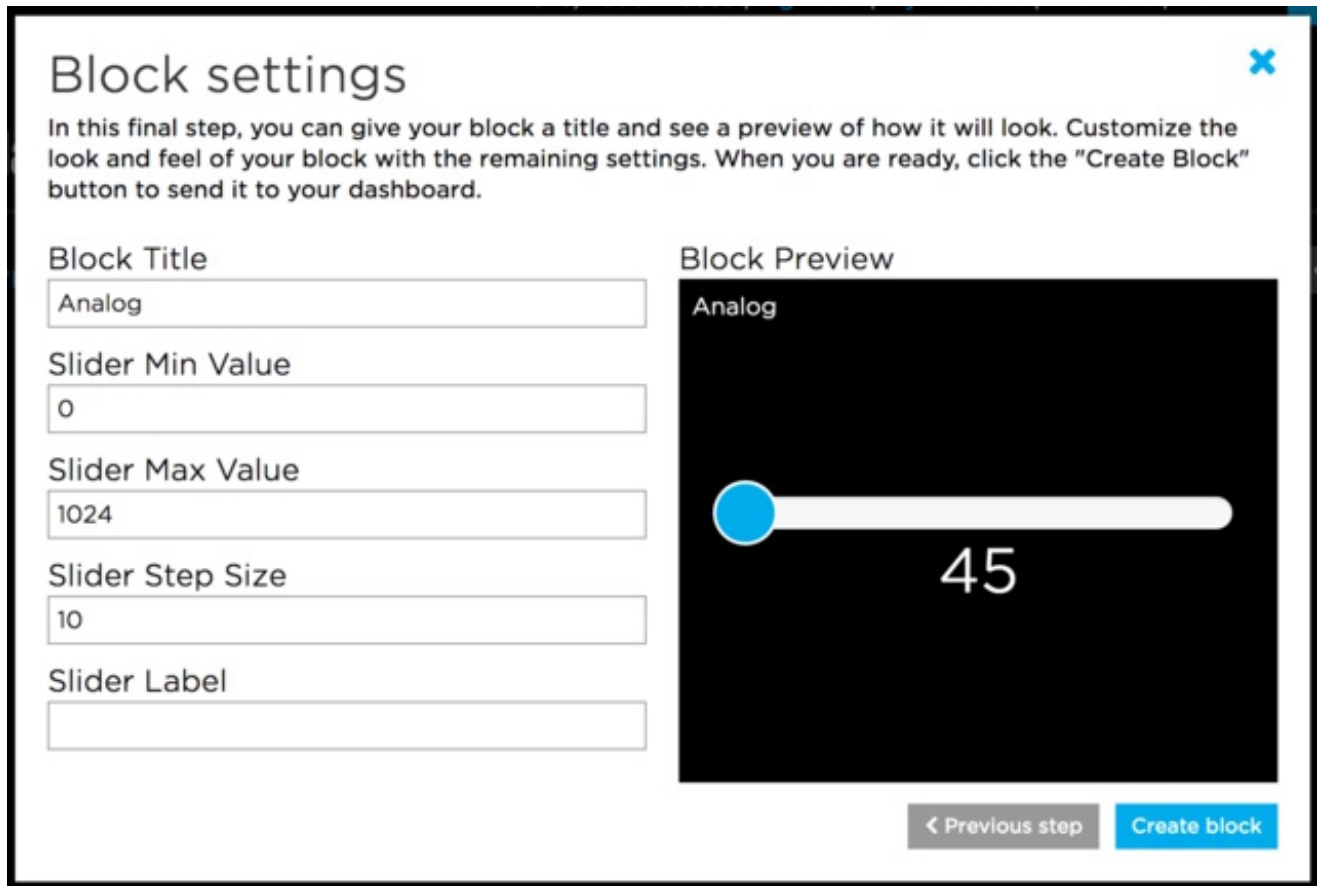


Adding the Slider Block

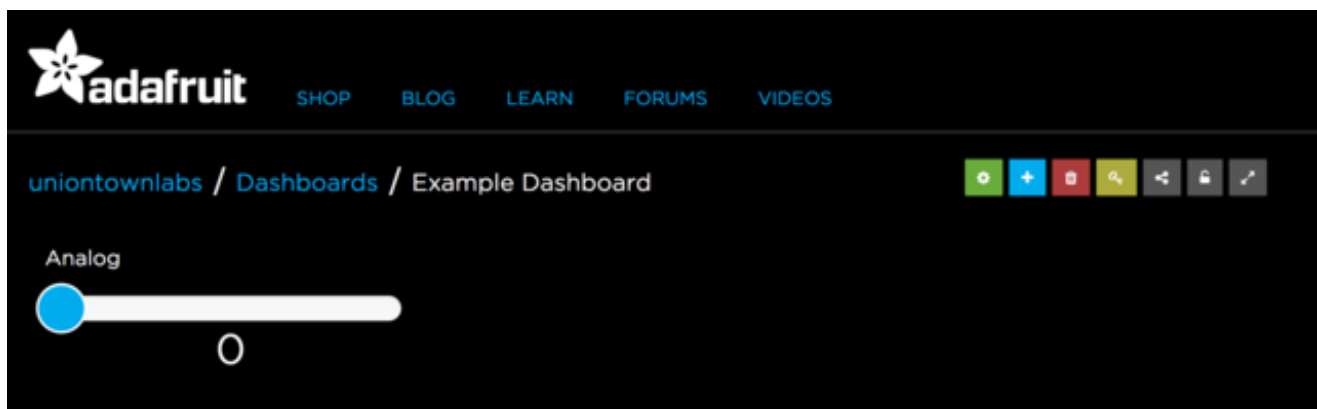
Next, add a new Slider Block to a new or existing dashboard. Name the block whatever you

would like, and set **min value to 0** and **max value to 1024**. Make sure you have selected the **Analog** feed as the data source for the slider.

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



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

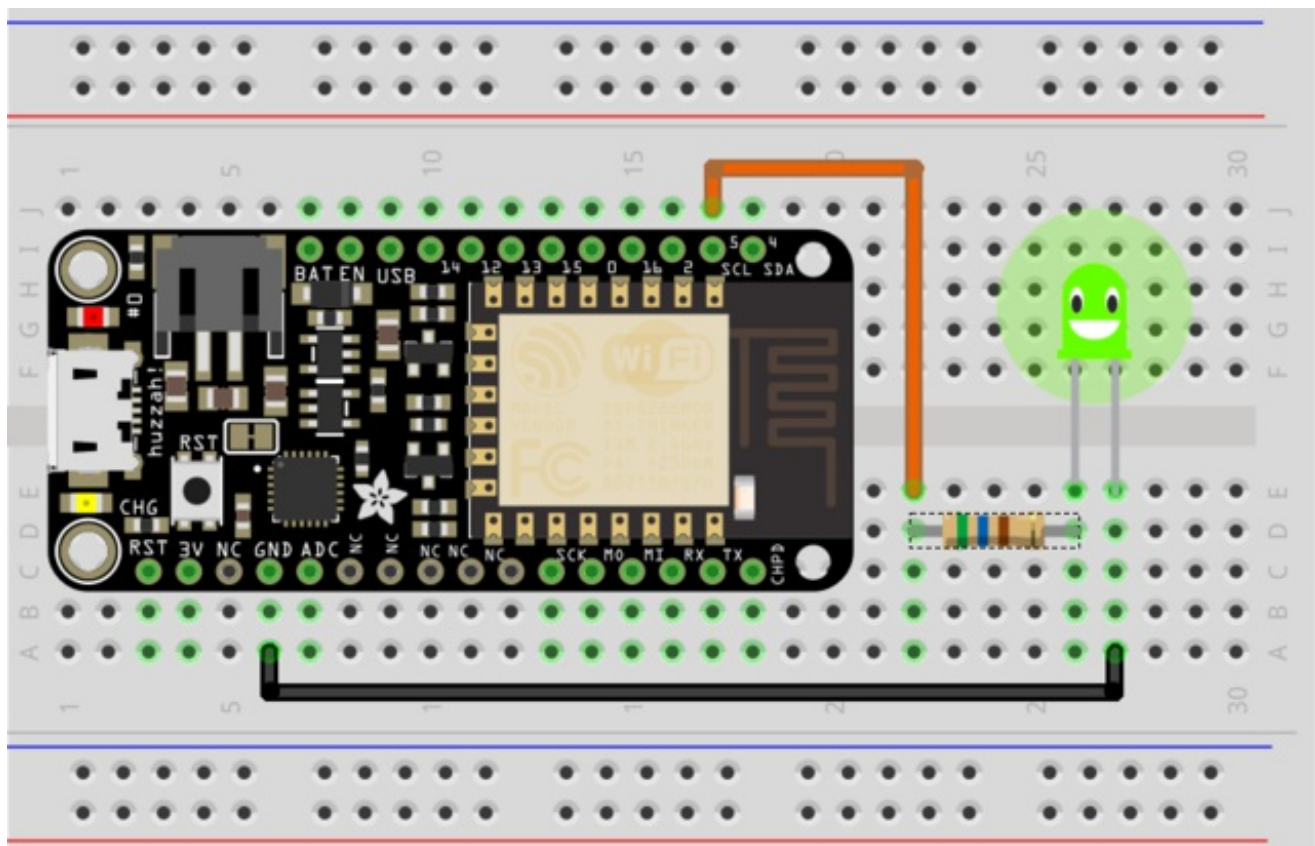
You will need the following parts for this tutorial:

- **1x** Adafruit IO compatible Feather
- **2x** jumper wires
- **1x** 560 ohm resistor
- **1x** 10mm LED

You will need to connect the following pins to the LED and 10k resistor:

- Feather **GND** to LED **cathode** (short leg)
- Feather **Pin 5** to one leg of the **560 ohm resistor**
- LED **anode** (long leg) to the second leg of the **560 ohm resistor**

Note: Resistors are *not* polarized, so the 560 ohm resistor can be connected to the circuit in either direction.



fritzing

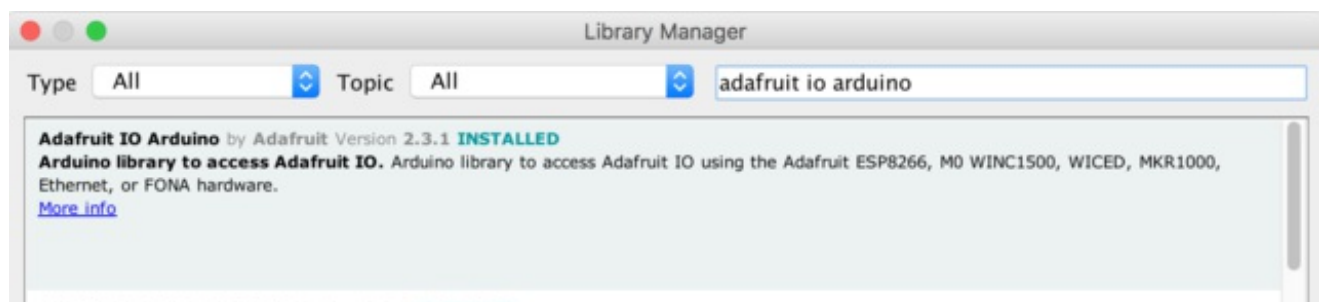
Next, let's look at the example sketch we will be using.

Arduino Setup

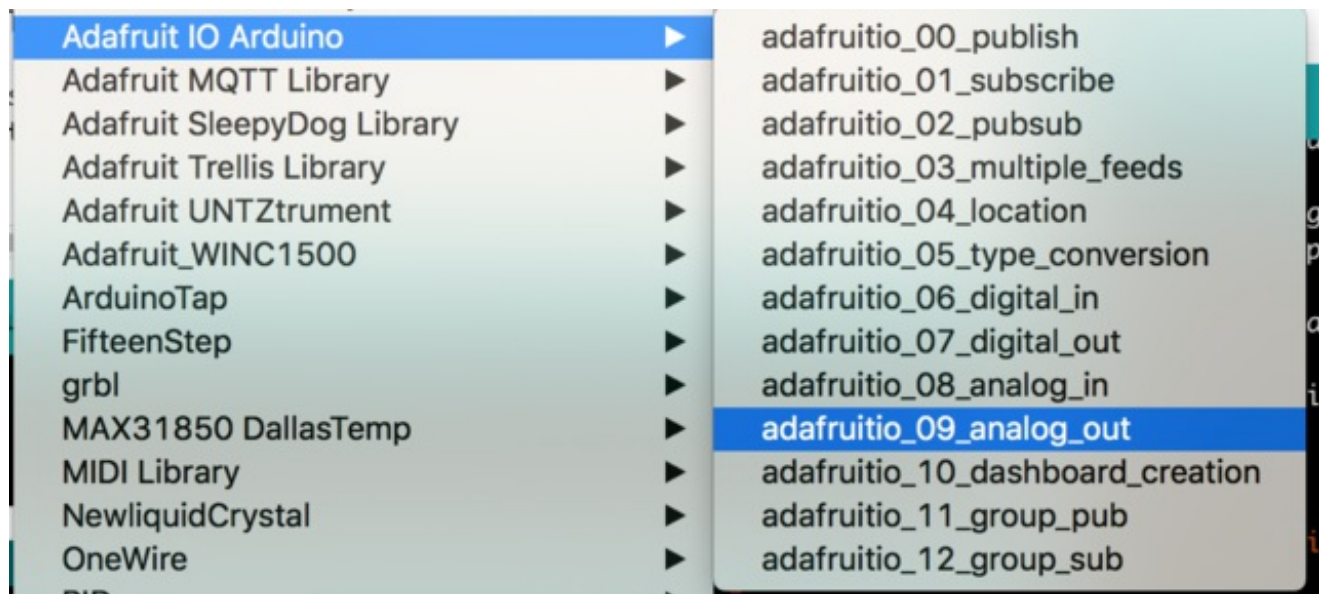
You should go through the setup guides associated with your selected set of hardware, and make sure you have internet connectivity with the device before continuing. The following links will take you to the guides for your selected platform.

- [Adafruit Feather HUZAH ESP8266 Setup Guide](#)

You will need to make sure you have at least **version 2.3.1** of the Adafruit IO Arduino library installed before continuing.



For this example you will need to open the **adafruitio_09_analog_out** 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.

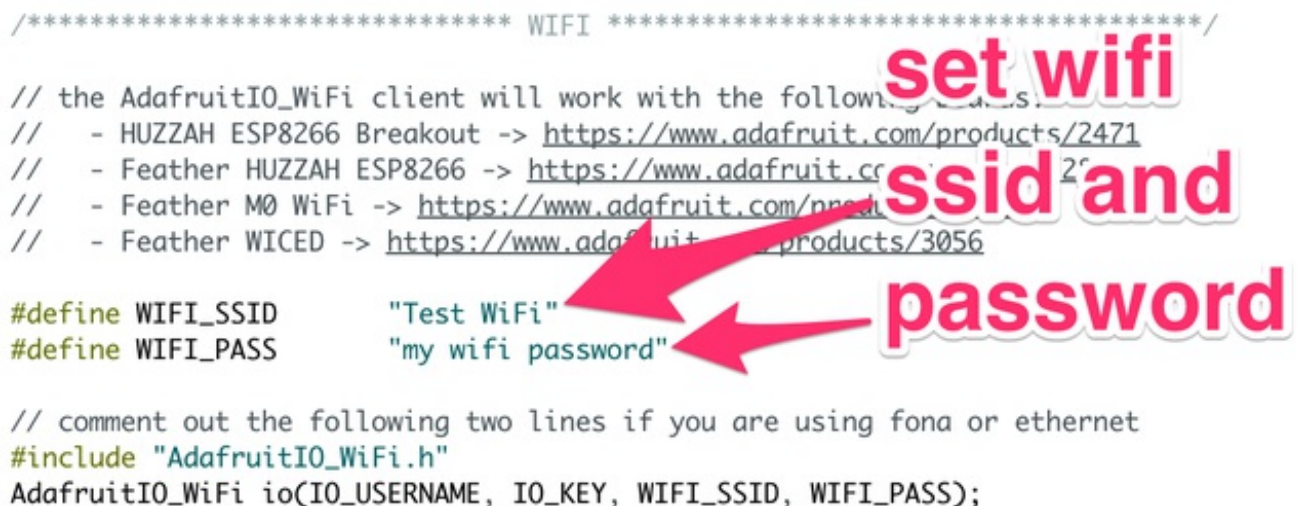


The screenshot shows the Arduino IDE interface with the `config.h` file open. A large pink text overlay at the top reads "set your username and key". Two red arrows point from this text to the `IO_USERNAME` and `IO_KEY` definitions in the code. The code is as follows:

```
config.h §  
/***** Adafruit IO Config *****/  
  
// visit io.adafruit.com you need to create an account,  
// or if you need your Adafruit IO key.  
#define IO_USERNAME "uniontownlabs"  
#define IO_KEY "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
```

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.



The screenshot shows the `WIFI` section of the `config.h` file. A large pink text overlay on the right reads "set wifi ssid and password". Two red arrows point from this text to the `WIFI_SSID` and `WIFI_PASS` definitions in the code. The code is as follows:

```
/***** 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/2472  
// - Feather M0 WiFi -> https://www.adafruit.com/products/3056  
// - 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**

```
/****** WIFI *****/  
// comment out default  
// wifi config lines  
// - Feather Huzzah ESP8266 -> https://www.adafruit.com/products/2821  
// - Feather WiFi -> https://www.adafruit.com/products/3010  
// - Feather WiFi -> 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.

```
/****** FONA *****/  
  
// the AdafruitIO_FONA library:  
// - Feather 32u4 FONA -> https://www.adafruit.com/product/3027  
  
// uncomment the following two lines:  
// 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**

```

/***** WIFI *****/

// comment out default
// wifi config lines
// - Feather Huzzah ESP8266 -> https://www.adafruit.com/products/2821
// https://www.adafruit.com/products/3010
// 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 Ethernet config lines in the Ethernet section of **config.h** to enable Ethernet Wing support.

```

/***** ETHERNET *****/

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

// uncomment both
// ethernet config lines
// and comment out the AdafruitIO_WiFi client in the WiFi 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_09_analog_out** example uses **pin 5** by default, and that can be modified by changing the **LED_PIN** define at the top of the sketch. This pin should correspond to a pin on your feather with PWM capability.

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

```
// this should correspond to a pin with PWM capability
#define LED_PIN 5
```

The next chunk of code sets up an Adafruit IO Feed instance for a feed called **analog**.

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

In the setup function, we attach a function called **handleMessage** to the **analog** feed that will be called whenever your device receives messages for that feed.

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

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

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

}
```

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 **analog** feed gets a message.

We use the `data->toInt()` function to convert the incoming data to an **int**, and set the state of the **LED_PIN** to that value using `analogWrite()`.

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

  // convert the data to integer
  int reading = data->toInt();

  Serial.print("received <- ");
  Serial.println(reading);
  analogWrite(LED_PIN, reading);

}
```

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.

Change the value of the slider on your Adafruit IO dashboard, and you should see something resembling the following in the Arduino Serial Monitor.


```
received <- 940  
received <- 290  
received <- 230  
received <- 110  
received <- 0  
received <- 90  
received <- 320  
received <- 630  
received <- 840  
received <- 1020
```

You should also see your LED change brightness depending on the value you send.

