



Using IFTTT with Adafruit IO to Make an IoT Door Detector

Created by Todd Treece



<https://learn.adafruit.com/using-ifttt-with-adafruit-io>

Last updated on 2024-06-03 01:47:29 PM EDT

Table of Contents

Overview	3
<ul style="list-style-type: none">• Adafruit IO + IFTTT	
Wiring	3
<ul style="list-style-type: none">• Low Power Usage• Battery tracking	
Adafruit IO Setup	4
<ul style="list-style-type: none">• Creating the Feeds	
Arduino Code	6
<ul style="list-style-type: none">• Arduino Sketch• Configuring Adafruit IO and WiFi• Code• Upload and Test	
IFTTT <-> Adafruit.io	12
<ul style="list-style-type: none">• IFTTT Connection• Creating the IFTTT Recipe	

Overview

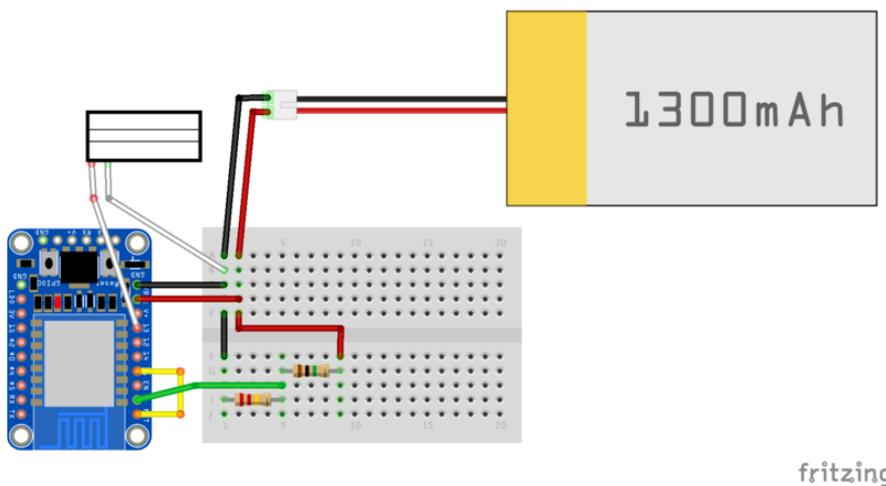


Way cheaper than a guard dog who can use an iPhone, this project will show you how you can use an Adafruit HUZAZH ESP8266 WiFi microcontroller board with a door sensor to email/tweet/text you when your door has opened!

Adafruit IO + IFTTT

This tutorial will show building a full-fledged IoT project using Adafruit IO and a Huzzah ESP8266, adding sensors, and then connecting it to [IFTTT \(if-this-then-that\)](https://adafru.it/iOe) (<https://adafru.it/iOe>) an API gateway that can communicate with Adafruit IO to add tons of connectivity options to your project.

Wiring



Our schematic is fairly simple, we will have one pin (**GPIO #13**) monitor the door sensor.

Low Power Usage

We can put the ESP8266 in low power mode and then 'wake up' every few seconds. This is way better for battery life than just keep the ESP8266 awake forever. To do that, tie pin GPIO #16 to the RST pin, that way the auto-wakeup will work.

Battery tracking

We will also keep track of the battery level by creating a high-resistor-divider on VBat to reduce the voltage of the battery from $220\text{k}\Omega/1220\text{k}\Omega = 1/5.5$ times. This means a max voltage of the battery (4.2V) is 0.75V, well within the range of the ESP8266's 1.0V-max ADC

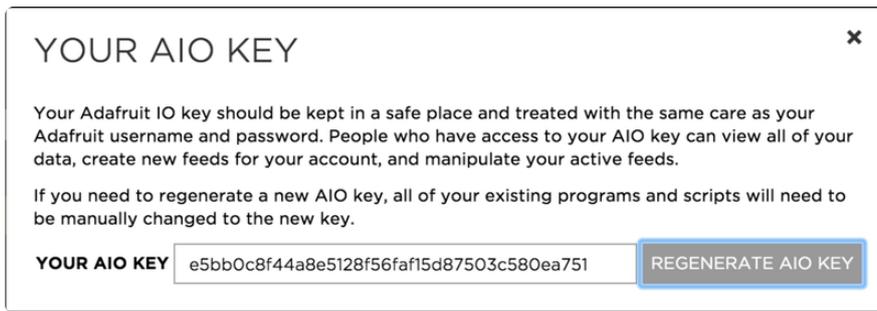
Note that even with low-power sleeping, this setup draws about 20mA on average!

- **Pin 16** to **RST** (this lets us use the low power mode)
- **Pin 13** to one side of **door sensor**
- **GND** to opposite side of **door sensor**
- **VBat** to the **battery +**
- **GND** to **battery -**
- **GND** to one side of the **220k Ω** resistor
- **VBat** to one side of the **1M Ω** resistor
- **Pin A** to the opposite side of the **1M Ω** resistor and **220k Ω** resistor

Adafruit IO Setup

Before you even upload code to the arduino, you'll need to get your IO account going - that way you can properly test your code and connection!

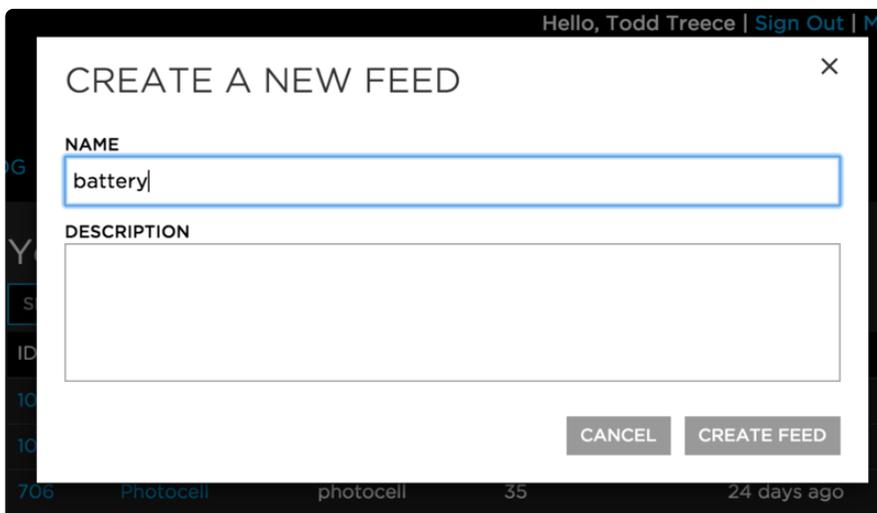
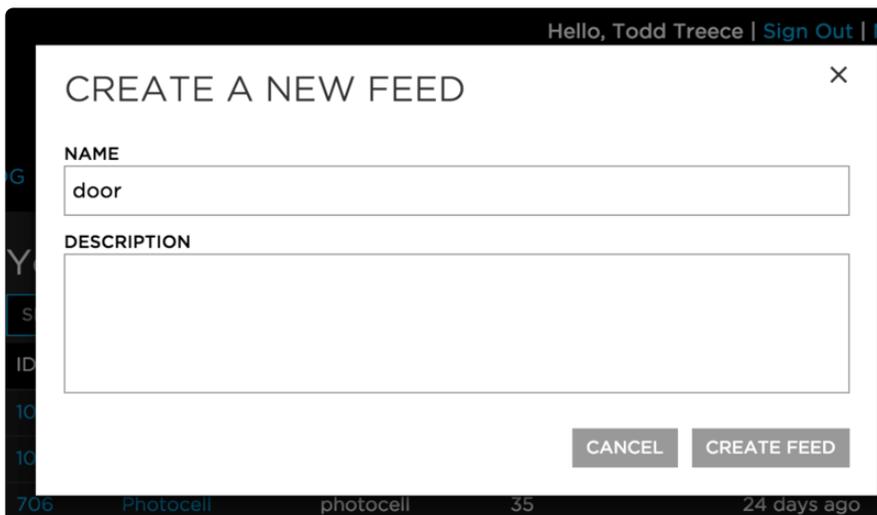
The first thing you will need to do is to login to your [Adafruit IO account \(https://adafru.it/eZ8\)](https://adafru.it/eZ8) and get your Adafruit IO Key if you haven't already. Click the **AIO KEY** button on the right hand side of the window, to access your key.



A window will pop up with your Adafruit IO key. Keep a copy of this in a safe place. We'll need it later. (Don't use the one in the image above, its not a real key, its just to indicate what the key looks like)

Creating the Feeds

You will now need to create feeds called "door" and "battery". If you need help getting started with creating feeds on Adafruit IO, check out the [Adafruit IO Feed Basics guide \(https://adafru.it/ioA\)](https://adafru.it/ioA).



Now you can start pushing data into the feeds, by wiring up and uploading the sketch to your Arduino!

Arduino Code

You will need the [Adafruit IO Arduino](https://adafru.it/fpd) (<https://adafru.it/fpd>) library installed to compile the example sketch. The easiest way to install the library is by using the [Arduino IDE v.1.6.4+ Library Manager](https://adafru.it/fCN) (<https://adafru.it/fCN>), and searching for **Adafruit IO Arduino**.

You will also need to have the ESP8266 Arduino board package installed. For more info about installing the ESP8266 package, visit the [HUZZAH ESP8266 setup guide](https://adafru.it/irC) (<https://adafru.it/irC>).

Ensure the board package used for the Huzzah ESP8266 is \geq version 2.4.0, older versions have an issue with waking from DeepSleep.

```
http://arduino.esp8266.com/stable/package_esp8266com_index.json
```

Use the package index URL above to get the latest version of the ESP8266 package.

You must have your Adafruit IO account set up first before you try to connect!

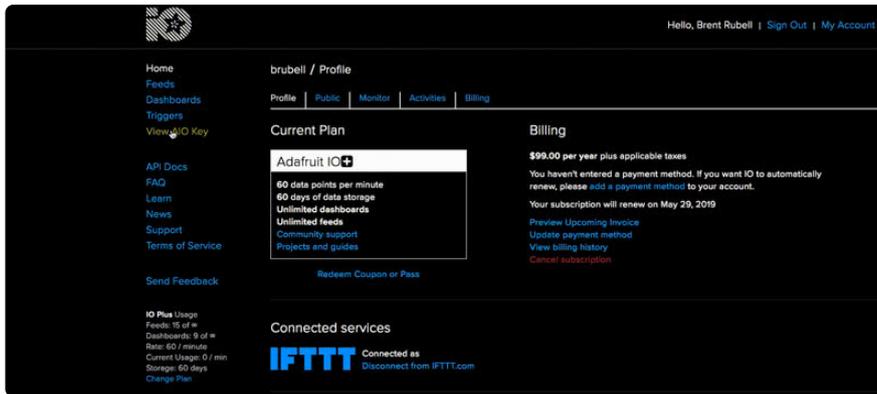
Arduino Sketch

The Arduino sketch for this project is fairly straight forward. We'll be using the [IFTT Door Detector sketch](https://adafru.it/BZn) (<https://adafru.it/BZn>) for this guide. Make sure to download both the sketch (IFTTT_Door_Detector.ino) and the configuration file (config.h).

Configuring Adafruit IO and WiFi

To configure your ESP8266 with your Adafruit IO account, you will need to set up your `IO_USERNAME` and `IO_KEY` in the `config.h` tab.

To find your `IO_USERNAME`, [navigate to your profile on Adafruit IO](https://adafru.it/BmD) (<https://adafru.it/BmD>) and click **View AIO Key**. Copy the **Username** field (ctrl+c or command+c)



Then, in the `config.h` tab, replace the `"your_username"` text with your the username from your profile:

```

adafruitio_06_digital_in  config.h
/***** Adafruit IO Config *****/

// visit io.adafruit.com if you need to create an account,
// or if you need your Adafruit IO key. Replace with your IO Username
#define IO_USERNAME  "your_username"
#define IO_KEY       "your_key"

```

To find your `IO_Key`, navigate to your profile, click **View AIO Key**, and copy the **ACTIVE KEY** field to your clipboard (`ctrl+c` or `command+c`).

In `config.h`, replace the `IO_KEY` with the IO Key copied to your clipboard.

```

adafruitio_06_digital_in  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  "your_username"
#define IO_KEY       "your_key" ← Replace with your IO Key

```

Next, we're going to configure the sketch to use your WiFi network. In the `config.h` tab, replace `"your_ssid"` with your WiFi's SSID and `"your_pass"` with your WiFi's password:

```
adafruitio_06_digital_in - config.h | Arduino 1.8.5
Verify
adafruitio_06_digital_in 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 "your_username"
#define IO_KEY "your_key"

/***** 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 "your_ssid"
#define WIFI_PASS "your_pass"

// 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);
```

Code

By default, the sketch sends the battery level to Adafruit IO once every 5 minutes, and checks the state of the door once every 3 seconds. You can modify these intervals by changing the `BATTERY_INTERVAL` and `SLEEP_LENGTH` constants. Setting these constants to lower intervals will result in reduced battery life.

```
#define BATTERY_INTERVAL 5
```

```
#define SLEEP_LENGTH 3
```

The bulk of the work for this example happens in the `setup()` function. The sketch restarts into `setup()` after it wakes from sleep, so the main `loop()` never runs.

```
void setup() {
  // start the serial connection
  Serial.begin(115200);

  while (!Serial);
  Serial.println("AdafruitIO Door Detector");

  EEPROM.begin(512);
  pinMode(DOOR, INPUT_PULLUP);

  // get the current count position from eeprom
  byte battery_count = EEPROM.read(0);

  // we only need this to happen once every X minutes,
  // so we use eeprom to track the count between resets.
  if(battery_count >= ((BATTERY_INTERVAL * 60) / SLEEP_LENGTH)) {
    // reset counter
```

```

    battery_count = 0;
    // report battery level to Adafruit IO
    battery_level();
  } else {
    // increment counter
    battery_count++;
  }

  // save the current count
  EEPROM.write(0, battery_count);
  EEPROM.commit();

  // if door isn't open, we don't need to send anything
  if(digitalRead(DOOR) == LOW) {
    Serial.println("Door closed");
    // we don't do anything
  } else {
    // the door is open if we have reached here,
    // so we should send a value to Adafruit IO.
    Serial.println("Door is open!");
    door_open();
  }

  // we are done here. go back to sleep.
  Serial.println("zzzz");
  ESP.deepSleep(SLEEP_LENGTH * 1000000);
}

```

The code that connects and sends the state of the door to Adafruit IO can be found within the **door_open()** function.

```

void door_open(){
  connect_AI0();

  // grab the door feed
  AdafruitIO_Feed *door = io.feed("door");

  Serial.println("Sending door value to feed..");
  door->save(1);
  io.run();
}

```

The HUZZAH 8266's battery level is sent the Adafruit IO battery feed once every five minutes. If you want to increase or decrease this interval, you can edit the **BATTERY_INTERVAL** constant. Be warned, though, decreasing the number will decrease your HUZZAH's battery life.

```

void battery_level() {

  // read the battery level from the ESP8266 analog in pin.
  // analog read level is 10 bit 0-1023 (0V-1V).
  // our 1M & 220K voltage divider takes the max
  // lipo value of 4.2V and drops it to 0.758V max.
  // this means our min analog read value should be 580 (3.14V)
  // and the max analog read value should be 774 (4.2V).
  int level = analogRead(A0);

  // convert battery level to percent
  level = map(level, 580, 774, 0, 100);
  Serial.print("Battery level: "); Serial.print(level); Serial.println("%");
  connect_AI0();
}

```

```

// grab the battery feed
AdafruitIO_Feed *battery = io.feed("battery");

// send battery level to AIO
battery->save(level);
io.run();
}

```

Upload and Test

OK now that you have everything wired up, and your [Adafruit IO \(https://adafru.it/fsU\)](https://adafru.it/fsU) account set up, upload the above sketch to your HUZZAH ESP8266 board via the FTDI cable or another serial connection.

After uploading, open up the serial port. You should see a long stream of text that looks like this:

```

COM110
r 1e0rB Ec ânÀ à E ,i pE<Ž,B i8 'BÇ'ÜäE p ònnä Ä;ònÄ'Üä
Door closed
zzzz
r 1e0rB E# ânÀ ^à E ,i pE|Ž,B i8 'YÇ'ÜäE p ònnä Ä;ònÄ'Üä
Door closed
zzzz
r 1e0rB E# ânÀ ^à E ,i pE<Ž,B i8 'YÇ'äE p ònnä Ä;ònÄ'ä
Door closed
zzzz
r 1e0rB E# ânÀ ^à E ,i pE|Ž,Y ix 'YÇ'ÜäE p ònnä Ä;ònÄ'Üä
Door is open!
Starting WiFi
.....Connecting to Adafruit.io
Sending to Adafruit.io
zzzz
r 1e0rB E# ânE à E ,i pE<Ž,B i8 'BÇ'ÜäE p ònnä Ä;ònÄ'Üä
Door closed
zzzz
r 1e0rB E# ânÀ à E ,i pE<Ž,B i8 'BÇ'ÜäE p ònnä Ä;ònÄ'Üä
Door closed
zzzz
r 1e0rB E# ânÀ à E ,i pE<Ž,B ix 'BÇ'ÜäE p ònnä Ä;ònÄ'Üä
Door closed

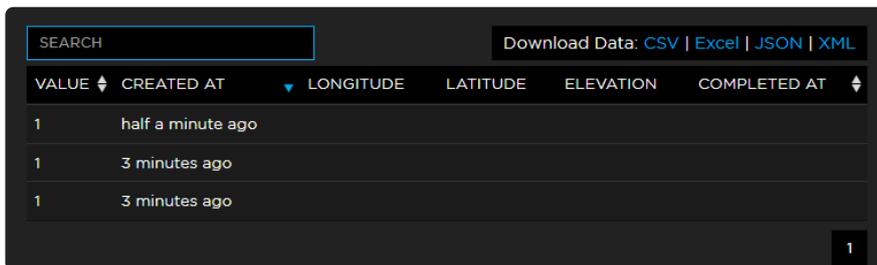
```

What you're seeing is the HUZZAH reading the sensor (**Door closed** and **Door is open!**) and then going to sleep (the **zzz** text indicates its about to go into sleep mode. When the ESP8266 goes to sleep, it resets the board, and so the weird text

after the 'zzz' is the reset debug string. Its normal but some terminal programs print it out a little differently.

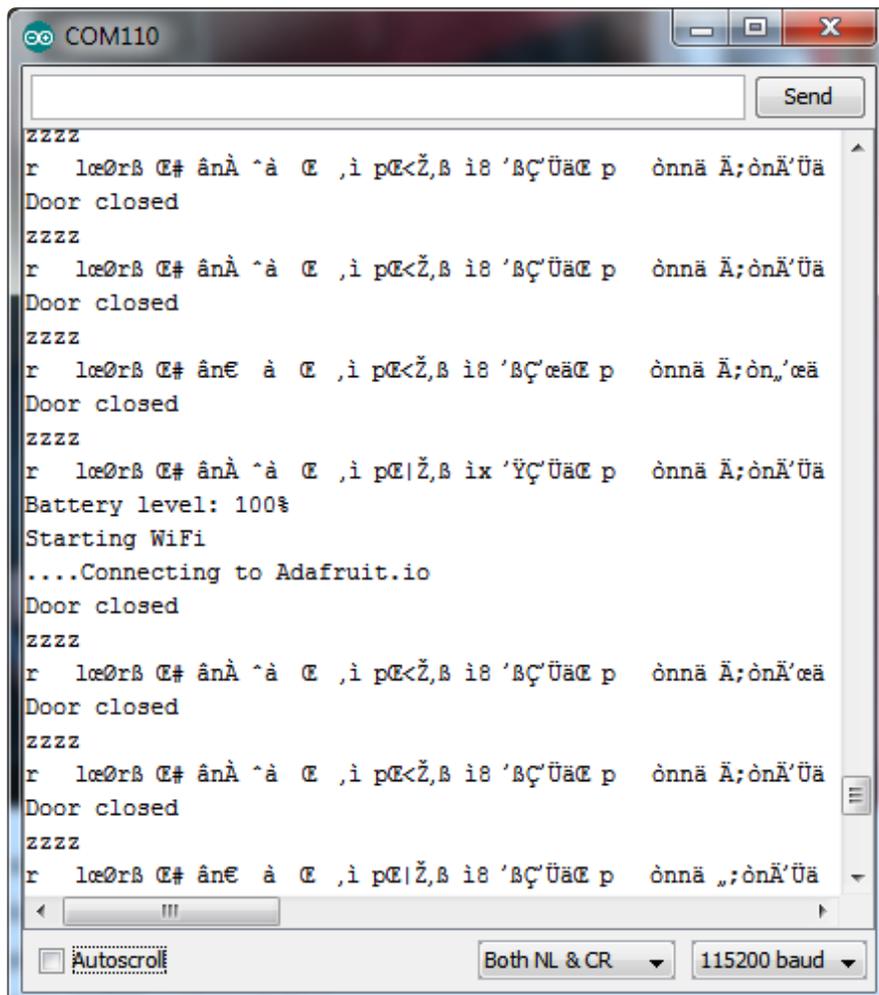
This program only pushes data to the feed when the door is open, since that's a rare event. So when the door is closed, nothing 'happens'. When you remove the magnet half from the sensor half, you'll get that text that says the door is open, and it will connect to WiFi and update your feed

Log into your [Adafruit IO \(https://adafru.it/fsU\)](https://adafru.it/fsU) account and look at the **door** feed to see the messages as they are logged!



VALUE	CREATED AT	LONGITUDE	LATITUDE	ELEVATION	COMPLETED AT
1	half a minute ago				
1	3 minutes ago				
1	3 minutes ago				

After the board is active for five minutes, you'll also see the board measure the battery voltage and upload that to the separate **battery** feed.



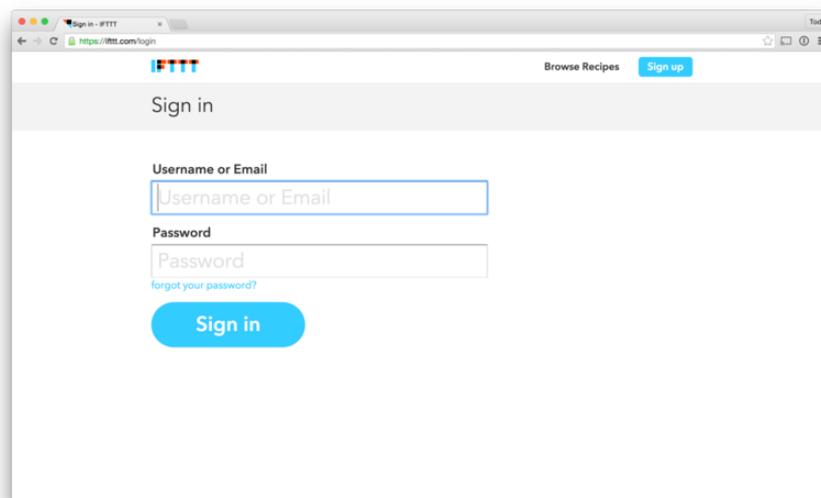
```
COM110
zzzz
r lœ0rB œ# ânÀ ^à œ ,i pœ<Ž,B i8 'BÇ'Üäœ p ònnä Ä;ònÄ'Üä
Door closed
zzzz
r lœ0rB œ# ânÀ ^à œ ,i pœ<Ž,B i8 'BÇ'Üäœ p ònnä Ä;ònÄ'Üä
Door closed
zzzz
r lœ0rB œ# ân€ à œ ,i pœ<Ž,B i8 'BÇ'œäœ p ònnä Ä;òn„'œä
Door closed
zzzz
r lœ0rB œ# ânÀ ^à œ ,i pœ|Ž,B ix 'YÇ'Üäœ p ònnä Ä;ònÄ'Üä
Battery level: 100%
Starting WiFi
...Connecting to Adafruit.io
Door closed
zzzz
r lœ0rB œ# ânÀ ^à œ ,i pœ<Ž,B i8 'BÇ'Üäœ p ònnä Ä;ònÄ'œä
Door closed
zzzz
r lœ0rB œ# ânÀ ^à œ ,i pœ<Ž,B i8 'BÇ'Üäœ p ònnä Ä;ònÄ'Üä
Door closed
zzzz
r lœ0rB œ# ân€ à œ ,i pœ|Ž,B i8 'BÇ'Üäœ p ònnä „;ònÄ'Üä
```



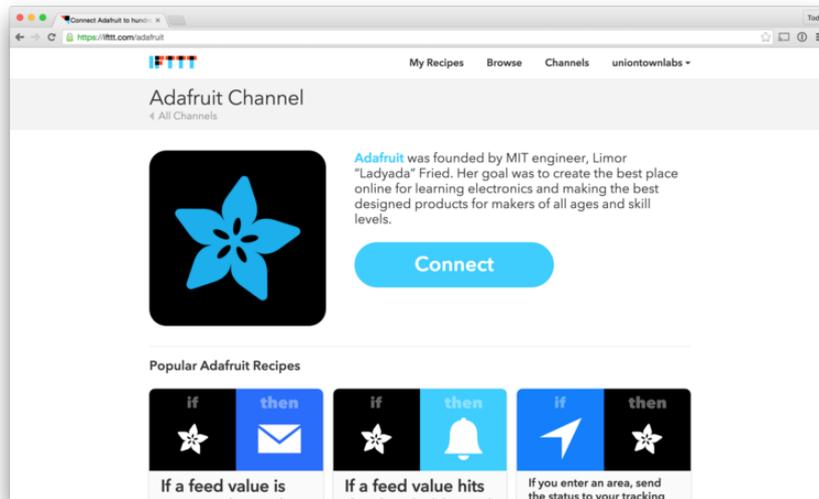
IFTTT <-> Adafruit.io

IFTTT Connection

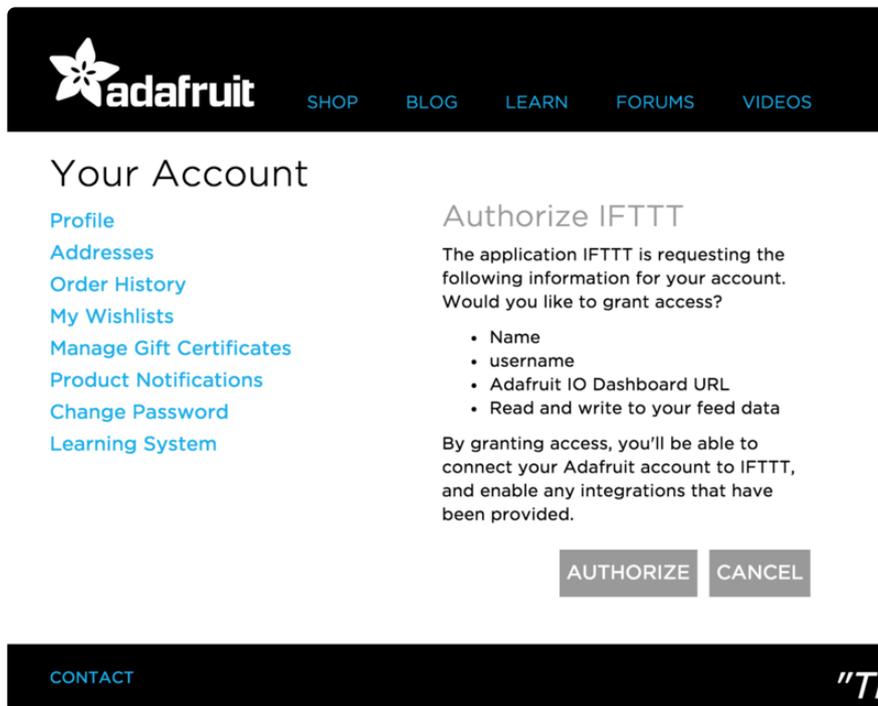
OK don't forget you will have to visit ifttt.com (<https://adafru.it/fYY>) and sign up for an account.



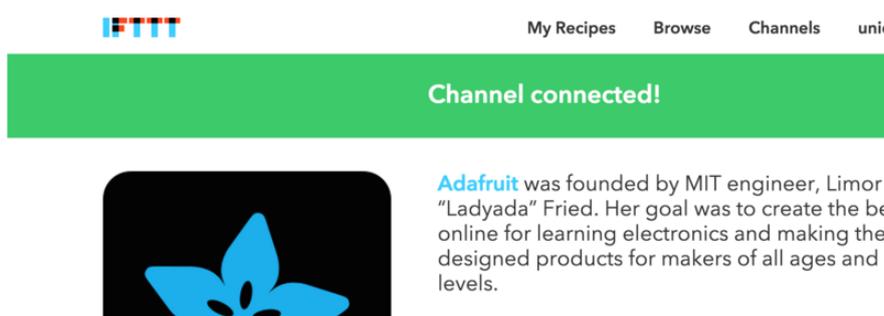
Once you have signed in, visit [ifttt.com/adafruit](https://adafru.it/fYZ) (<https://adafru.it/fYZ>) and click the **Connect** button to connect your IFTTT account to Adafruit IO.



You will be redirected to your Adafruit account page, and will be asked to authorize IFTTT. Click the **AUTHORIZE** button to grant access.

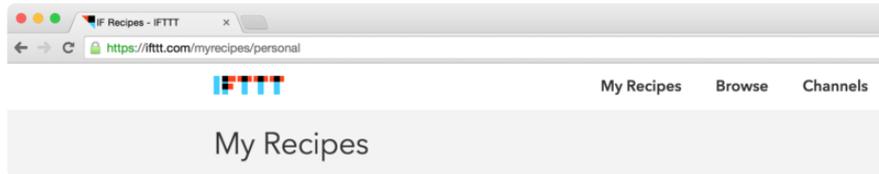


You will then be redirected back to IFTTT, and should see a **Channel connected!** message at the top of the page.

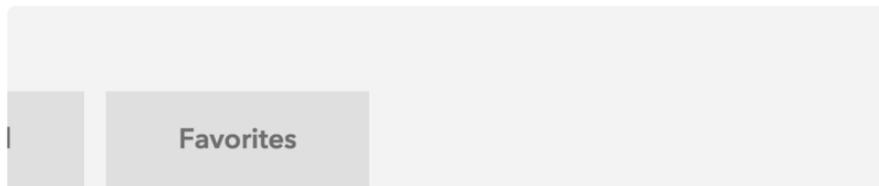


Creating the IFTTT Recipe

Next, navigate to your personal [IFTTT Recipes \(https://adafru.it/fZ0\)](https://adafru.it/fZ0) page to start the recipe creation process.



Click the **Create a Recipe** button.

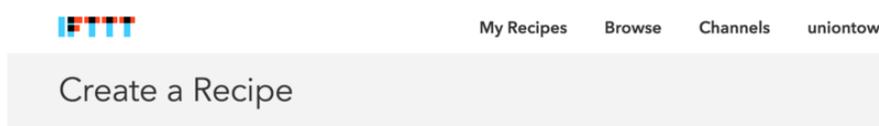


ground.



Recipes yet! Check out one of
find one you'll love.

Click the blue **this** block of text to get started.



ifthisthen**that**

Click this to get started.

Click on the **Adafruit** channel.

Choose Trigger Channel st

Showing Channels that provide at least one Trigger. [View all C](#)



500px



Adafruit



Amazon Alexa

Click on **Monitor a feed** in Adafruit IO.



Choose a Trigger

step 2 of 7

Any new data

This Trigger fires any time there is new data in your feed.

Monitor a feed on Adafruit IO

This Trigger fires anytime it validates the data that you send to your feed. Example: If Feed Temperature > 80, fire Trigger.

Select the **battery** feed from the dropdown, and make the relationship less than the value of **10**. Click the **Create Trigger** button when finished.

Complete Trigger Fields step 3 of 7

Monitor a feed on Adafruit IO

Feed

battery

Relationship

less than

Value

10

The value to compare against.

Create Trigger

Then, click the blue **that** block of text to select what happens when the battery level is low.



Next, choose an action to happen when the trigger activates. In this example, we will be sending an email.

Choose Action Channel step 4 of 7

[back](#)

Showing Channels that provide at least one Action. [View all Channels](#)

email



Email



Email Digest



Gmail

Select the **send me an email** option.

Choose an Action

step 5 of 7

Send me an email

This Action will send you an HTML based email. Images and links are supported.

Enter the **subject** and **body** messages that you would like sent to you when the trigger happens, and then click **Create Action**.

Complete Action Fields

step 6 of 7

Send me an email

Subject

FeedName low

Body

The FeedValue is Operator TriggerValue at
CreatedAt !

Create Action

Edit the recipe title, and click **Create Recipe** to complete the process.

if then

Data on battery feed is less than
10

Recipe Title

If data on battery feed is less than 10, then send me
an email at me@example.com

59

use '#' to add tags

Create Recipe

Next. Repeat the same process for the **door** feed, but instead of monitoring for specific values, choose to monitor for **Any new data**.

Complete Trigger Fields step 3 of 7

Any new data

 Feed name

door

Create Trigger

Your recipes are now finished and waiting for door and battery changes.

IF Recipes run automatically in the background.

Create a Recipe

if  then 

created less than a minute ago
never run

If any new data on door feed, then send me an email at me@example.com

if  then 

created 2 minutes ago
never run

If data on battery feed is less than 10, then send me an email at me@example.com

You will then be setup to receive IFTTT emails whenever someone opens the door!

Back Door Open 

Inbox x



IFTTT Action <action@ifttt.com>

to me 

Someone is breaking into your house.