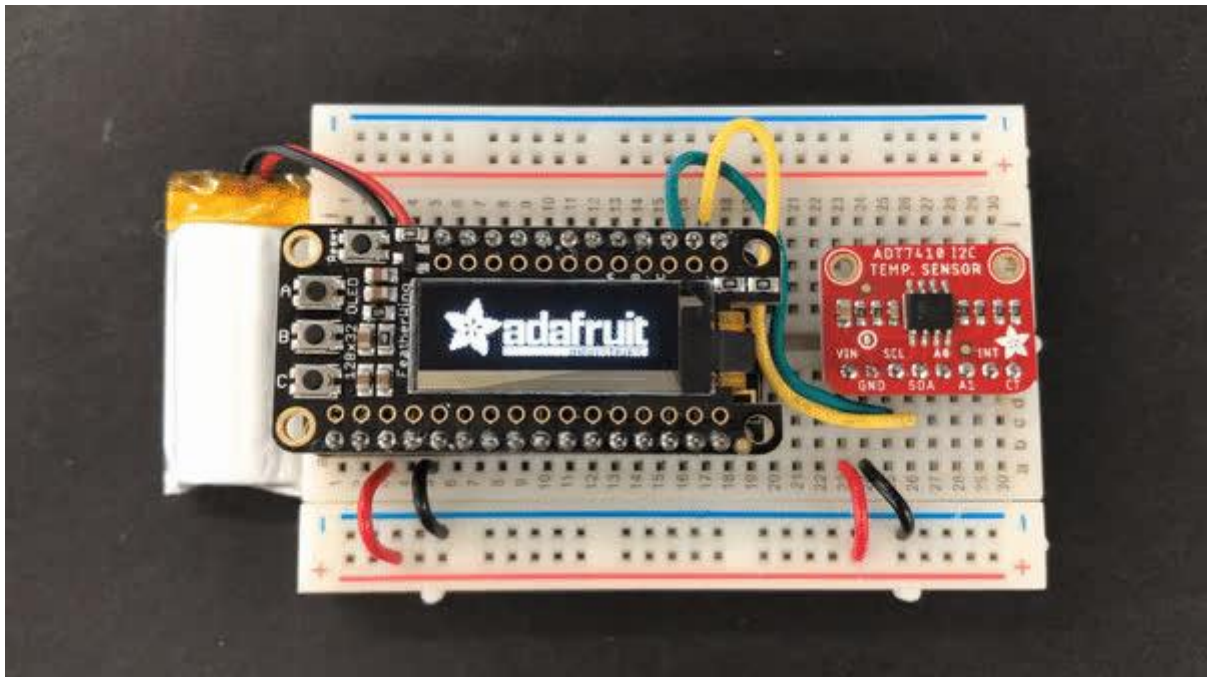




# IoT Temperature Logger with Analog Devices ADT7410, Feather and Adafruit IO

Created by Brent Rubell



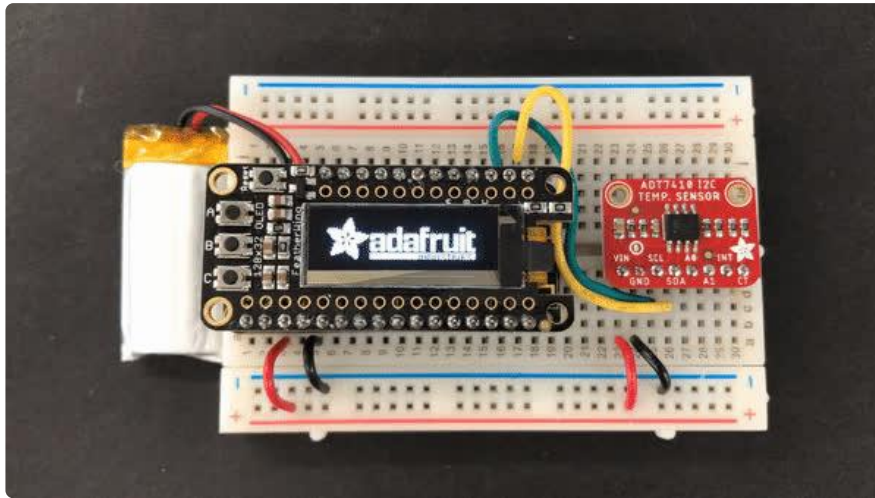
<https://learn.adafruit.com/iot-temperature-logger-with-arduino-and-adafruit-io>

Last updated on 2022-12-01 03:29:26 PM EST

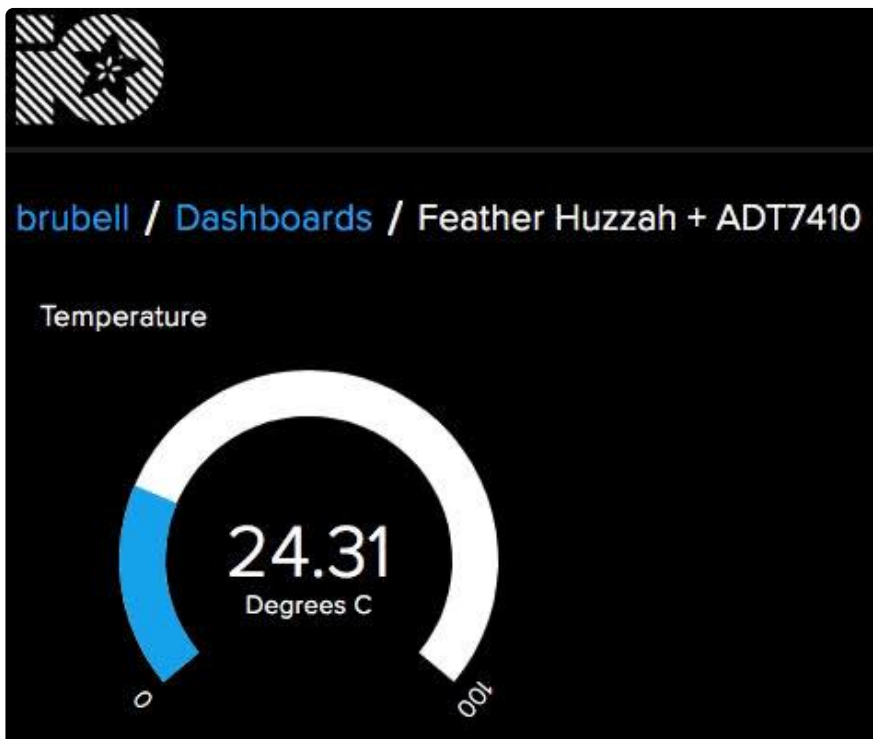
# Table of Contents

<b>Overview</b>	<b>3</b>
<ul style="list-style-type: none"><li>• Parts</li><li>• Materials</li></ul>	
<b>Adafruit IO Setup</b>	<b>5</b>
<ul style="list-style-type: none"><li>• Feed Setup</li><li>• Dashboard Setup</li></ul>	
<b>Wiring</b>	<b>9</b>
<b>Arduino Code</b>	<b>10</b>
<ul style="list-style-type: none"><li>• Code</li><li>• Using Adafruit IO</li></ul>	

# Overview



Analog Devices, known for their reliable and well-documented sensor chips - has a high precision and high resolution temperature sensor on the market, and we've got a breakout to make it easy to use! The Analog Devices ADT7410 gets straight to the point - it's an I2C temperature sensor, with 16-bit 0.0078°C temperature resolution and 0.5°C temperature tolerance. Wire it up to your microcontroller or single-board computer to get reliable temperature readings with ease

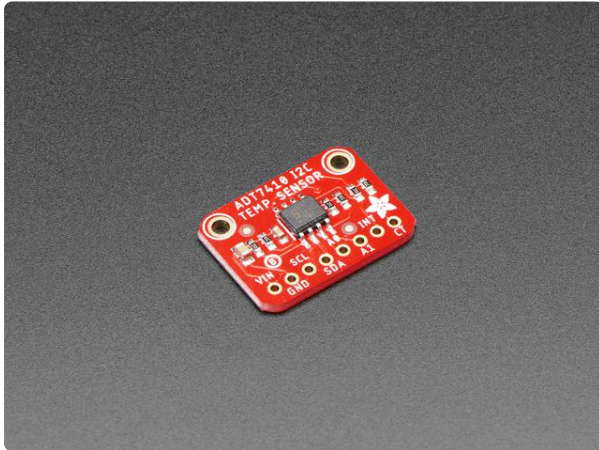


This sensor can easily be brought online using a Feather HUZAZH and Adafruit IO. Yo u'll be up and running in under 15 minutes!

If you're looking for a way to precisely monitor and log temperature data to the cloud, follow along!

[Thanks to Digi-Key \(\)](#) and [Analog Devices \(\)](#) for sponsoring the development of this guide - we've made the ADT7410 PCB "[Digi-Key red \(\)](#)" in their honor!

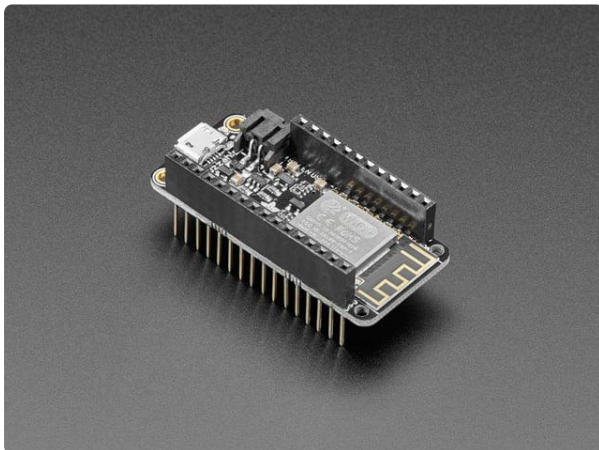
## Parts



### [ADT7410 High Accuracy I2C Temperature Sensor Breakout Board](#)

Analog Devices, known for their reliable and well-documented sensor chips - has a high precision and high resolution temperature sensor on the market, and we've got a breakout to...

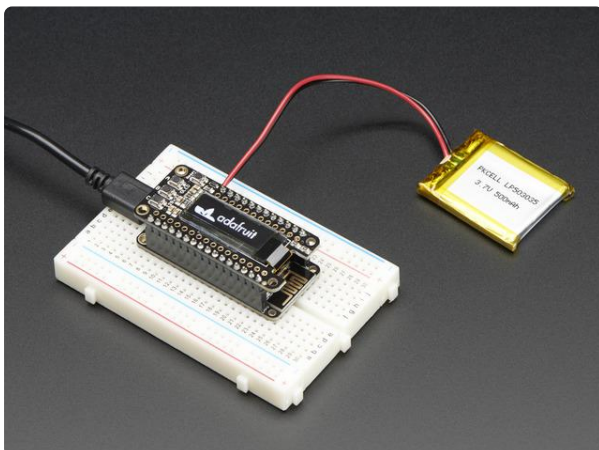
<https://www.adafruit.com/product/4089>



### [Assembled Feather Huzzah w/ ESP8266 WiFi With Stacking 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/3213>



### [Adafruit FeatherWing OLED - 128x32 OLED Add-on For Feather](#)

A Feather board without ambition is a Feather board without FeatherWings! This is the FeatherWing OLED: it adds a 128x32 monochrome OLED plus 3 user buttons to...

<https://www.adafruit.com/product/2900>

## Materials

You'll need the materials below to complete this project. If you do not have them, pick them up from the Adafruit Store.

1 x [Breadboard](https://www.adafruit.com/product/64) <https://www.adafruit.com/product/64>  
Half-size breadboard

---

1 x [Wire Bundle](https://www.adafruit.com/product/153) <https://www.adafruit.com/product/153>  
Breadboarding wire bundle

---

1 x [MicroUSB Cable](https://www.adafruit.com/product/592) <https://www.adafruit.com/product/592>  
USB cable - USB A to Micro-B - 3 foot long

---

## Adafruit IO Setup

### Feed Setup

If you do not already have an Adafruit IO account set up, head over to [io.adafruit.com](https://io.adafruit.com) () to link your Adafruit.com account to Adafruit IO.

The first step is to create a new Adafruit IO feed to hold the AD7410's temperature. Navigate to the [feeds page](#) () on Adafruit IO. Then click Actions -> Create New Feed, and name this feed temperature.

- If you do not already know how to create a feed, head over to [Adafruit IO Basics: Feeds](#) ().

## Create a new Feed ✕

Name

Description

Add to groups

Cancel

Create

## Dashboard Setup

Next, step is to create a dashboard to display the value read by the ADT7410 sensor.

- If you do not know how to create or use Dashboards in Adafruit IO, head over to the [Adafruit IO Basics: Dashboards \(\)](#) guide.

### Create a new block ✕

Click on the block you would like to add to your dashboard. You can always come back and switch the block type later if you change your mind.



Select the Gauge block.

temperature

## Group / Feed

☰ My Feeds

huzzah\_temperature

temperature

Select the temperature feed created earlier.

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 (optional)  
Temperature

Gauge Min Value  
0

Gauge Max Value  
100

Gauge Width  
25px

Gauge Label  
Degrees C

Low Warning Value

Optional: If no low warning value is given, the gauge will only change color when the value is out of bounds.

High Warning Value

Optional: If no high warning value is given, the gauge will only change color when the value is out of bounds.

Block Preview

Temperature

45  
Degrees C

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

Test Value  
45

Published Value  
45

bytes

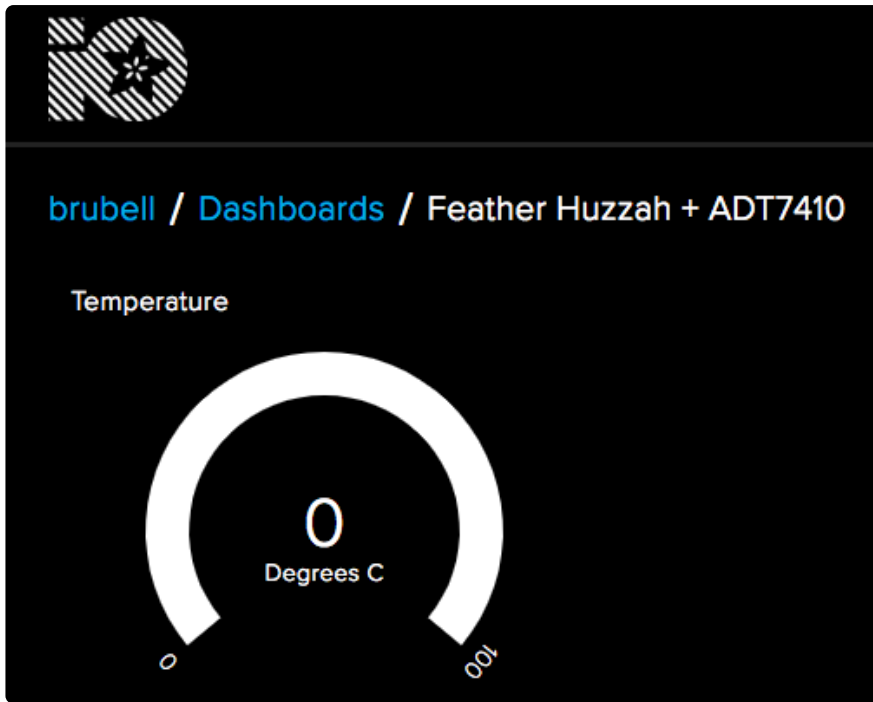
Previous step Create Block

In the Block Settings step, set the Block Title to Temperature, set the Gauge Min/Max Values to the upper and lower temperature thresholds you want to measure.

You can label the gauge by setting the Gauge Label - this example assumes temperature is to be measured in Degrees C.

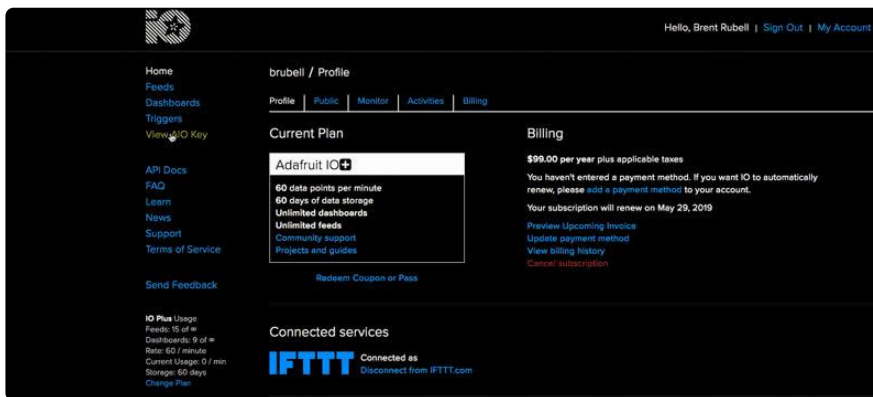
Uncomfortably hot/cold? You can optionally set the gauge change color to warn you if the temperature goes above (or below) a certain value.

After adding the gauge element, your dashboard will look like the following:



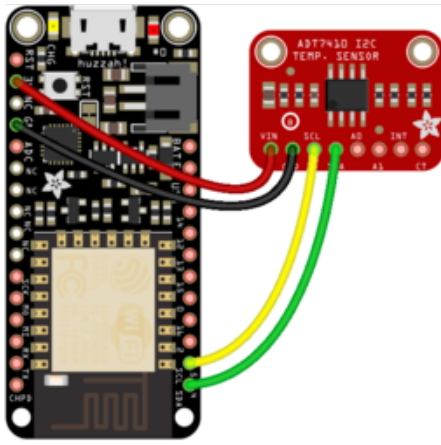
You are also going to need your Adafruit IO username and secret API key.

Navigate to your profile and click the View AIO Key button to retrieve them. Write them down in a safe place, you'll need them for later.





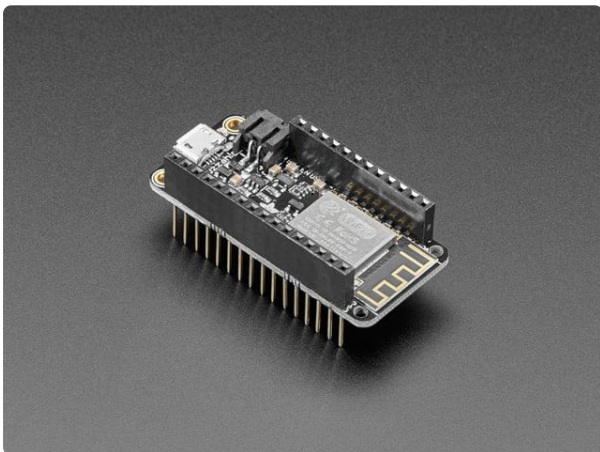
# Wiring



Make the following connections between the ADT7410 and the Feather Huzzah ESP8266

- ADT7410 VIN to Feather 3V
- ADT7410 GND to Feather GND
- ADT7410 SCL to Feather SCL
- ADT7410 SDA to Feather SDA

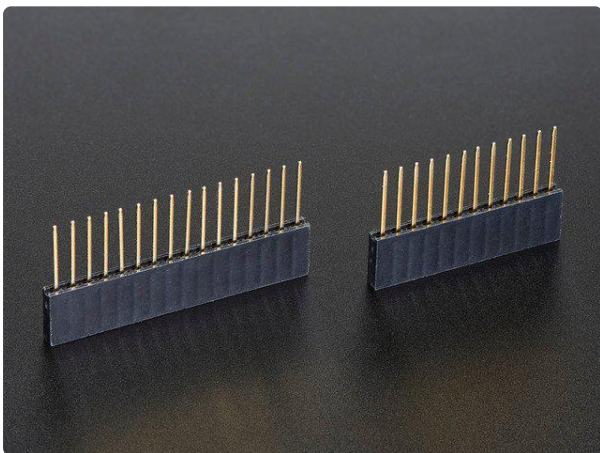
To easily plug in the Feather OLED on top, make sure to solder or select a Huzzah Feather with stacking headers



## [Assembled Feather HUZAZH w/ ESP8266 WiFi With Stacking Headers](https://www.adafruit.com/product/3213)

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/3213>



## [Stacking Headers for Feather - 12-pin and 16-pin female headers](https://www.adafruit.com/product/2830)

These two Female Stacking Headers alone are, well, lonely. But pair them with any of our Feather boards and...

<https://www.adafruit.com/product/2830>

Adding the FeatherWing OLED may require soldering assembly - [follow this page here for instructions](#) ().

---

# Arduino Code

The ESP8266 based Feather Huzzah & the Huzzah ESP8266 breakout are both very popular options for connecting projects to Adafruit IO. This guide assumes you've completed the setup required to get your ESP8266 up and running with Arduino IDE and Adafruit IO.

- If you haven't yet set up your ESP8266 for use with Adafruit IO and the Arduino IDE, [follow along with this guide](#) (). The setup only needs to be performed once.

You'll also need the Adafruit\_ADT7410 library and the Adafruit Unified Sensor library installed.

- If you have not done this yet, [click here to visit the setup guide and come back to this page when you're done.](#) ()

## Code

Copy and paste the following code into an empty Arduino Sketch:

```
// SPDX-FileCopyrightText: 2019 Brent Rubell for Adafruit Industries
//
// SPDX-License-Identifier: MIT

// Adafruit IO ADT7410 Example
//
// Adafruit invests time and resources providing this open source code.
// Please support Adafruit and open source hardware by purchasing
// products from Adafruit!
//
// Written by Ladyada and Brent Rubell for Adafruit Industries
// Copyright (c) 2019 Adafruit Industries
// Licensed under the MIT license.
//
// All text above must be included in any redistribution.

/***** 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_IO_USERNAME"
#define IO_KEY "YOUR_IO_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 Huzzah ESP32 -> https://www.adafruit.com/product/3405
// - Feather M0 WiFi -> https://www.adafruit.com/products/3010
// - Feather WICED -> https://www.adafruit.com/products/3056
```

```

#define WIFI_SSID "WIFI_NAME"
#define WIFI_PASS "WIFI_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);

/***** Configuration *****/

// time between sending data to adafruit io, in minutes
#define MESSAGE_WAIT_SEC (15 * 60)
/***** Example Starts Here *****/
#include <Wire.h>
// adt7410 sensor
#include "Adafruit_ADT7410.h"
// featherwing oled
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <SPI.h>

// Create the OLED display object
Adafruit_SSD1306 display = Adafruit_SSD1306(128, 32, &Wire);

// Create the ADT7410 temperature sensor object
Adafruit_ADT7410 tempsensor = Adafruit_ADT7410();

// set up the 'temperature' feed
AdafruitIO_Feed *huzzah_temperature = io.feed("temperature");

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

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

  Serial.println("Adafruit IO ADT7410 + OLED");
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C); // Address 0x3C for 128x32
  Serial.println("OLED begun");

  // Show image buffer on the display hardware.
  // Since the buffer is initialized with an Adafruit splashscreen
  // internally, this will display the splashscreen.
  display.display();
  delay(1000);

  // Clear the buffer.
  display.clearDisplay();
  display.display();

  // Make sure the sensor is found, you can also pass in a different i2c
  // address with tempsensor.begin(0x49) for example
  if (!tempsensor.begin())
  {
    Serial.println("Couldn't find ADT7410!");
    while (1)
      ;
  }

  // sensor takes 250 ms to get first readings
  delay(250);

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

```

```

display.clearDisplay();
display.display();
display.setTextSize(1);
display.setTextColor(WHITE);
display.setCursor(0, 0);
display.print("Connecting to IO...");
display.display();

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

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

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();

  // Read and print out the temperature, then convert to *F
  float c = tempsensor.readTempC();

  Serial.print("Temp: ");
  Serial.print(c);
  Serial.println("C");

  // clear the display
  display.clearDisplay();
  display.display();

  // print to display
  display.setTextColor(WHITE);
  display.setTextSize(2);
  display.setCursor(0, 0);
  display.print("Temp:");
  display.print(c);

  // send data to adafruit io
  display.setCursor(15, 20);
  display.setTextSize(1);
  display.print("Sending...");
  display.display();
  Serial.println("Sending to Adafruit IO");
  delay(1000);
  huzzah_temperature->save(c, 0, 0, 0, 2);

  // sent to IO
  display.clearDisplay();
  display.display();
  display.setTextSize(2);
  display.setCursor(0, 0);
  display.print("Temp:");
  display.print(c);
  display.setTextSize(1);
  display.setCursor(15, 20);
  display.print("Sending...Done!");
  display.display();

```

```

Serial.print("Waiting ");
Serial.print(MESSAGE_WAIT_SEC);
Serial.println(" seconds");
// wait 15 minutes between sends
for (int i = 0; i < MESSAGE_WAIT_SEC; i++)
{
  delay(1000);
}
}
}

```

Before you upload the sketch to the Huzzah, you'll need to configure the network and Adafruit IO.

Change **IO\_USERNAME** to your Adafruit IO username and **IO\_KEY** to the Adafruit IO Key you saved earlier.

```

/***** 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_IO_USERNAME"
#define IO_KEY      "YOUR_IO_KEY"

```

Next, set up the sketch to use WiFi. Change **WIFI\_SSID** to your router's SSID and **WIFI\_PASS** to your router's password.

```

/***** 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 HUZZAH ESP32 -> https://www.adafruit.com/product/3405
// - Feather M0 WiFi -> https://www.adafruit.com/products/3010
// - Feather WICED -> https://www.adafruit.com/products/3056

#define WIFI_SSID "WIFI_NAME"
#define WIFI_PASS "WIFI_PASS"

```

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

```

Adafruit IO ADT7410 + OLED
OLED begun
Connecting to Adafruit IOAdafruitIO::connect()

Adafruit IO connected.

```

The ADT7410 will read temperature values and the Feather will send them to Adafruit IO. You should see something resembling the following in the Arduino Serial Monitor:

```

Temp: 24.31C
Sending to Adafruit IO
Waiting 900 seconds

```

Want to change the amount of precision (decimal places) sent from the ADT7410 to Adafruit IO?

Modify this line from:

```
huzzah_temperature->save(c, 0, 0, 0, 2);
```

to

```
huzzah_temperature->save(c, 0, 0, 0, decimal_places);
```

Make sure to replace `decimal_places` with the number of decimal places you'd like to send from the ADT7410.

Want to change the delay between sending data to Adafruit IO?

In the code, change this line from:

```
#define MESSAGE_WAIT_SEC (15 * 60)
```

to

```
#define MESSAGE_WAIT_SEC (SECONDS_TO_DELAY_BY * 60)
```

Make sure to replace `SECONDS_TO_DELAY_BY` with the number of seconds to delay between sending sensor data.

## Using Adafruit IO

Next, let's check that the data has been received by Adafruit IO. You can do this [by visiting the Adafruit IO Monitor page \(\)](#). Every 15 minutes, the page will refresh with a new `huzzah-temperature` value.

```
Live Data newest data at the top

2019/02/08 12:23pm huzzah-temperature 24.19
2019/02/08 12:23pm huzzah-temperature 24.19
2019/02/08 12:22pm huzzah-temperature 24.12
2019/02/08 12:22pm huzzah-temperature 24.12
2019/02/08 12:22pm huzzah-temperature 24.12
2019/02/08 12:22pm huzzah-temperature 24.12
2019/02/08 12:22pm huzzah-temperature 24.12
2019/02/08 12:22pm huzzah-temperature 24.06
2019/02/08 12:22pm huzzah-temperature 24.06
2019/02/08 12:22pm huzzah-temperature 24.06
-----

Connections only available while this page is open

2019/02/08 12:17:44pm e63e95d0-de2b-41f1-9d5a-6b65ddb8 connected
```



Navigate to the dashboard you created earlier to view the temperature gauge you've added.

Every 15 minutes, it'll update and display a new value.