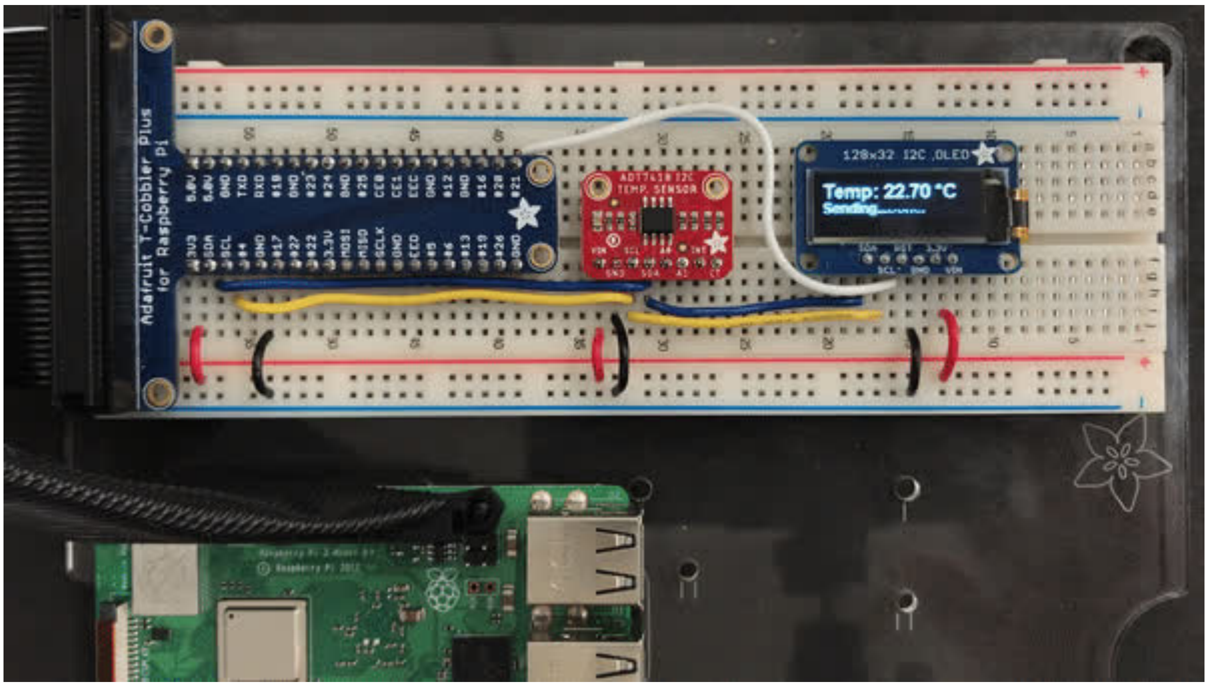
IoT Temperature Logger with Analog Devices ADT7410, Raspberry Pi, and Adafruit IO

Created by Brent Rubell

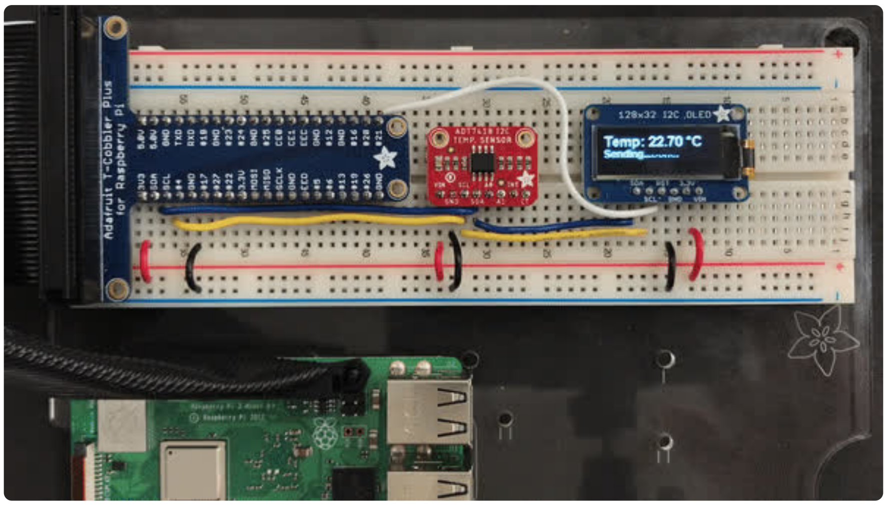


https://learn.adafruit.com/iot-temperature-logger-with-python-and-adafruit-io

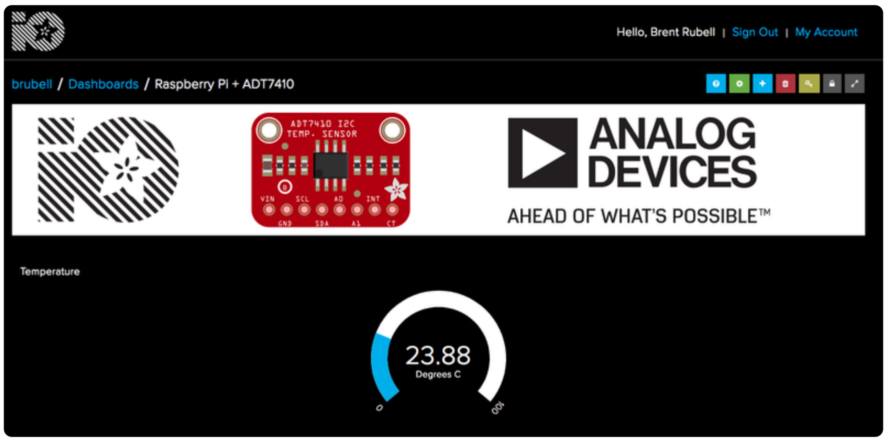
Last updated on 2024-06-03 02:39:46 PM EDT

Table of Contents

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



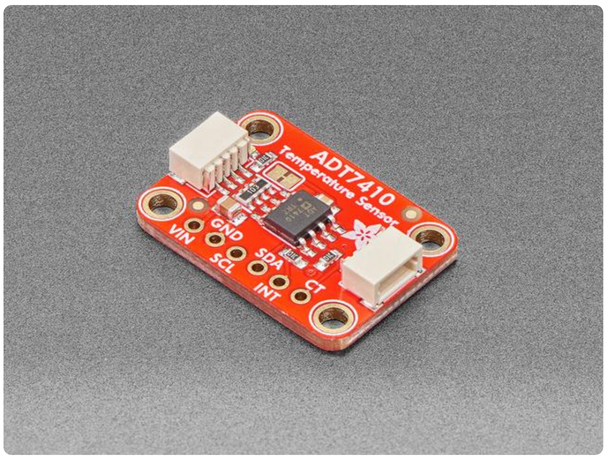
Hello, Brent Rubell | Sign Out | My Account
brubell / Dashboards / Raspberry Pi + ADT7410
ADT7410 I2C TEMP. SENSOR
ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™
Temperature
23.88
Degrees C
0
100

This sensor can be put online using a Raspberry Pi, Python, and Adafruit IO. You'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 (https://adafru.it/BJr) and Analog Devices (https://adafru.it/DPF) for sponsoring the development of this guide - Adafruit has made the ADT7410 PCB "Digi-Key red (https://adafru.it/BJr)" in their honor!

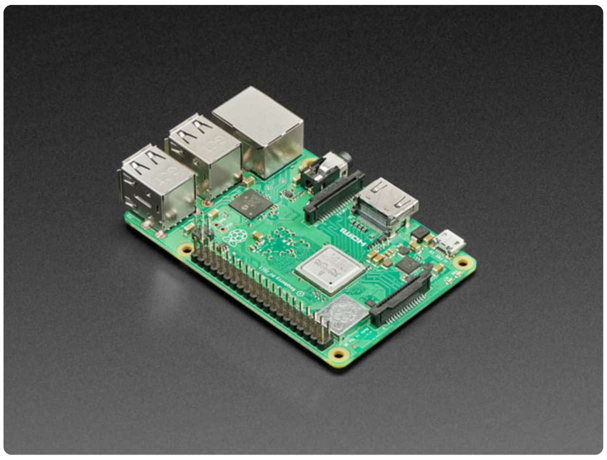
Parts



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



Raspberry Pi 3 - Model B+ - 1.4GHz Cortex-A53 with 1GB RAM

The Raspberry Pi 3 Model B is the most popular Raspberry Pi computer made, and the Pi Foundation knows you can always make a good thing better! And what could make the Pi 3...

https://www.adafruit.com/product/3775

Monochrome 128x32 I2C OLED graphic display

These displays are small, only about 1" diagonal, but very readable due to the high contrast of an OLED display. This display is made of 128x32 individual white OLED pixels, each...

https://www.adafruit.com/product/931

Assembled Pi T-Cobbler Plus - GPIO Breakout
This is the assembled version of the Pi T-Cobbler Plus. It only works with the Raspberry Pi Model Zero, A+, B+, Pi 2, Pi 3 & Pi 4! (Any Pi with 2x20...
https://www.adafruit.com/product/2028

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 Half-size breadboard	https://www.adafruit.com/product/64
1 x Adafruit Pi Dish Adafruit Pi Dish for Raspberry Pi - Includes Breadboard	https://www.adafruit.com/product/942
1 x Wire Bundle Breadboarding wire bundle	https://www.adafruit.com/product/153
1 x 16GB MicroSD Card 16GB Card with NOOBS 2.9	https://www.adafruit.com/product/1583
1 x MicroUSB 5V 2.5A Power Supply 5V 2.5A Switching Power Supply with 20AWG MicroUSB Cable	https://www.adafruit.com/product/1995

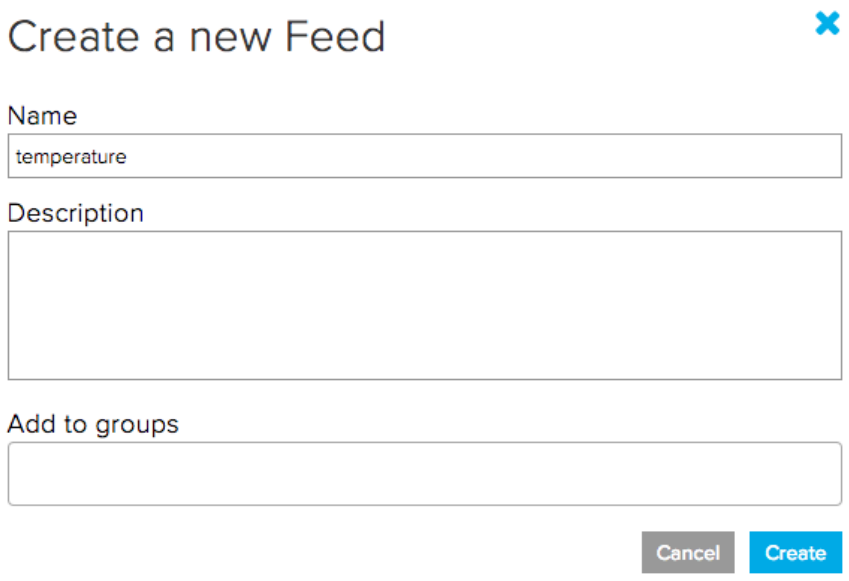
Adafruit IO Setup

Feed Setup

If you do not already have an Adafruit IO account set up, head over to io.adafruit.com (https://adafru.it/fH9) 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 (https://adafru.it/mxC) 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 (https://adafru.it/ioA).



Create a new Feed

Name: temperature

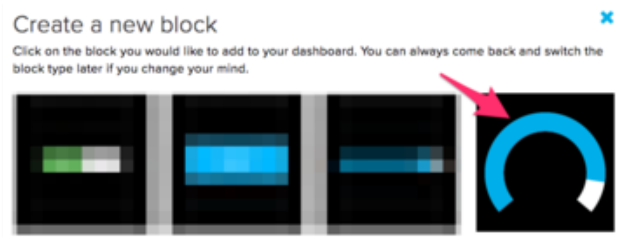
Description:

Add to groups:

Dashboard Setup

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

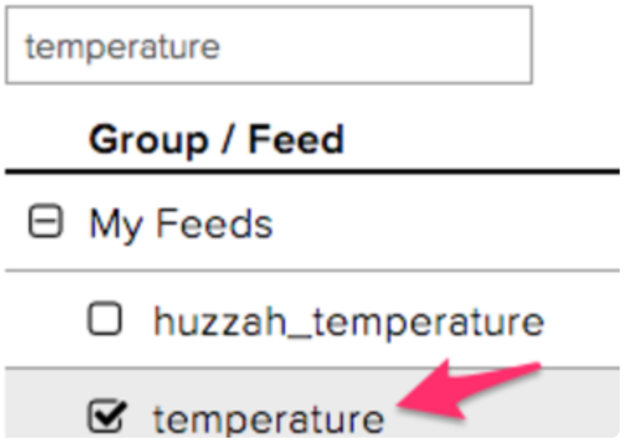
- If you do not know how to create or use Dashboards in Adafruit IO, head over to the Adafruit IO Basics: Dashboards (https://adafru.it/f5m) guide.



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

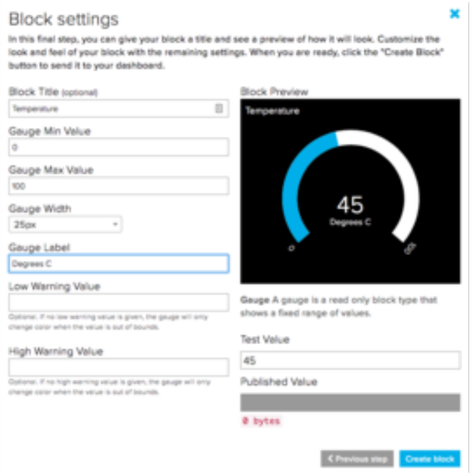
Group / Feed

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

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

Test Value: 45

Published Value:

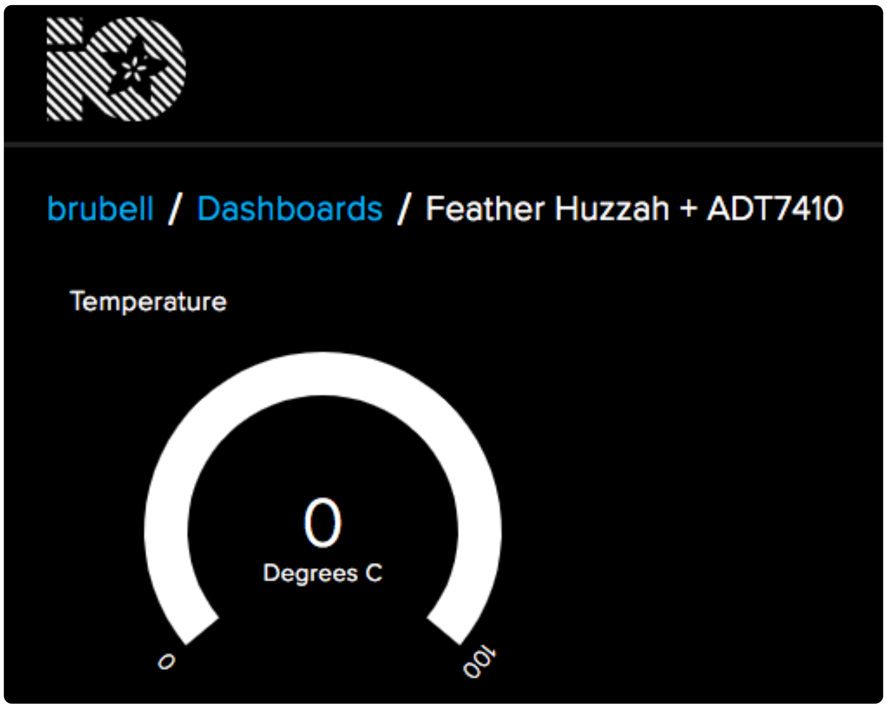
0 bytes

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.

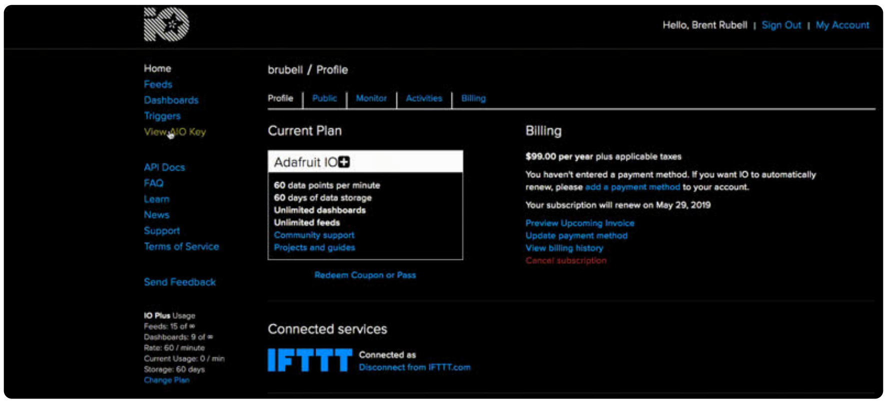
Uncomfortably hot/cold? You can optionally set the gauge change color to warn you if the temperature goes above (or below) a certain value.

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.


Hello, Brent Rubell | Sign Out | My Account

Home
Feeds
Dashboards
Triggers
View AIO Key

API Docs
FAQ
Learn
News
Support
Terms of Service

Send Feedback

IO Plus Usage
Feeds: 15 of ∞
Dashboards: 9 of ∞
Rate: 60 / minute
Current Usage: 0 / min
Storage: 60 days
Change Plan

brubell / Profile

Profile | Public | Monitor | Activities | Billing

Current Plan

Adafruit IO+

60 data points per minute
60 days of data storage
Unlimited dashboards
Unlimited feeds
Community support
Projects and guides

Redeem Coupon or Pass

Billing

$99.00 per year plus applicable taxes

You haven't entered a payment method. If you want IO to automatically renew, please add a payment method to your account.

Your subscription will renew on May 29, 2019

Preview Upcoming Invoice
Update payment method
View billing history
Cancel subscription

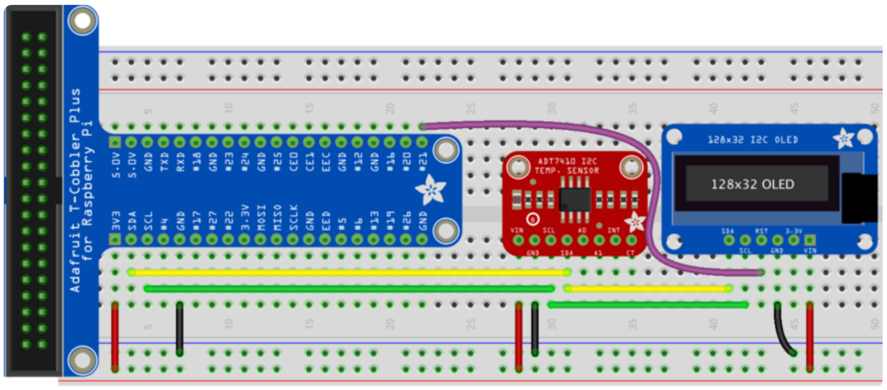
Connected services

IFTTT Connected as
Disconnect from IFTTT.com

Wiring

Need a way to connect your Raspberry Pi to a breadboard? Check out the T-Cobbler Plus - this add-on prototyping board lets you easily connect a Raspberry Pi (Raspberry Pi Model Zero, A+, B+, Pi 2,Pi 3) to a solderless breadboard:

Assembled Pi T-Cobbler Plus - GPIO Breakout
This is the assembled version of the Pi T-Cobbler Plus. It only works with the Raspberry Pi Model Zero, A+, B+, Pi 2, Pi 3 & Pi 4! (Any Pi with 2x20...
https://www.adafruit.com/product/2028



The ADT7410 and the 128x32 OLED each have a unique address and can be connected together as i2c devices, using the Raspberry Pi as a i2c controller (For more information about i2c addressing, check out the guide here... (https://adafru.it/BK0))

Make the following connections **between the Pi and the ADT7410**:

- **Pi 3.3V** to **ADT7410 VIN**
- **Pi GND** to **ADT7410 GND**
- **Pi SCL** to **ADT7410 SCL**
- **Pi SDA** to **ADT7410 SDA**

Make the following connections **between the Pi and the OLED Display**:

- **Pi SCL** to **Display SCL**
- **Pi SDA** to **Display SDA**
- **Pi GPIO #21** to **Display RST**
- **Pi 3.3V** to **Display VIN**
- **Pi GND** to **Display GND**

Python Code

This guide assumes that you've gotten your Raspberry Pi up and running, and have CircuitPython installed.

- If you have not done this yet, visit the installation guide **here** and come back when you're set up. (https://adafru.it/Deo)

Installing Fonts

This project uses a Microsoft TrueType Font. To install it, **enter the following into the terminal on your Raspberry Pi**:

```
sudo apt-get install ttf-mscorefonts-installer
```

Installing CircuitPython Libraries

Since CircuitPython is running on the Raspberry Pi - installing the libraries for this guide is quick and easy.

To **install the library for the display** (https://adafru.it/u1f), **enter the following into the terminal**

```
sudo pip3 install adafruit-circuitpython-ssd1306
```

You'll also need to install the CircuitPython library to read data from the Analog Devices ADT7410 (https://adafru.it/DPz):

```
sudo pip3 install adafruit-circuitpython-adt7410
```

The code for this project is below. **Save it to the Raspberry Pi** as `adafruit_io_adt7410.py`

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

"""
'adafruit_io_adt7410.py'
==================================
Example of sending temperature
values to an Adafruit IO feed using
an ADT7410 breakout.
```

```
Dependencies:
    - Adafruit_Blinka
        (https://github.com/adafruit/Adafruit_Blinka)
    - Adafruit_CircuitPython_SSD1306
        (https://github.com/adafruit/Adafruit_CircuitPython_SSD1306)
    - Adafruit_CircuitPython_ADT7410
        (https://github.com/adafruit/Adafruit_CircuitPython_ADT7410)
"""
# Import standard python modules
import time

# import Adafruit SSD1306 OLED
from PIL import Image, ImageDraw, ImageFont
import adafruit_ssd1306

# import Adafruit IO REST client
from Adafruit_IO import Client

# import Adafruit CircuitPython adafruit_adt7410 library
import adafruit_adt7410

# import Adafruit Blinka
import board
import busio
import digitalio

# Delay between sensor reads, in seconds
DELAY_SECONDS = 30

# Set to your Adafruit IO key.
# Remember, your key is a secret,
# so make sure not to publish it when you publish this code!
ADAFRUIT_IO_KEY = 'ADAFRUIT_IO_KEY'

# Set to your Adafruit IO username.
# (go to https://accounts.adafruit.com to find your username)
ADAFRUIT_IO_USERNAME = 'ADAFRUIT_IO_USERNAME'

# Create an instance of the REST client
aio = Client(ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)

# Set up `temperature` feed
pi_temperature = aio.feeds('temperature')

# Set up OLED
i2c_bus = busio.I2C(board.SCL, board.SDA)
oled_reset = digitalio.DigitalInOut(board.D21)
disp = adafruit_ssd1306.SSD1306_I2C(128, 32, i2c_bus, reset=oled_reset)
# Clear display.
disp.fill(0)
disp.show()

# Create blank image for drawing.
# Make sure to create image with mode '1' for 1-bit color.
width = disp.width
height = disp.height
image = Image.new('1', (width, height))
# Get drawing object to draw on image.
draw = ImageDraw.Draw(image)
# `sudo apt-get install ttf-mscorefonts-installer` to get these fonts
font_big = ImageFont.truetype('/usr/share/fonts/truetype/msttcorefonts/
Arial_Bold.ttf', 16)
font_small = ImageFont.truetype('/usr/share/fonts/truetype/msttcorefonts/
Arial_Bold.ttf', 12)

adt = adafruit_adt7410.ADT7410(i2c_bus, address=0x48)
adt.high_resolution = True
time.sleep(0.25)  # wait for sensor to boot up and get first reading
```

```
while True:
    # clear screen
    draw.rectangle((0, 0, width, height), outline=0, fill=0)

    # Read the temperature sensor
    tempC = adt.temperature
    tempC = round(tempC, 2)

    # display the temperature on the OLED
    draw.text((0, 0), "Temp: %0.2F *C" % tempC, font=font_big, fill=255)

    # Send temperature to Adafruit IO
    print('Sending temperature {0} C to Adafruit IO'.format(tempC))
    draw.text((0, 16), "Sending...", font=font_small, fill=255)
    disp.image(image)
    disp.show()

    aio.send(pi_temperature.key, tempC)
    draw.text((0, 16), "Sending...Done!", font=font_small, fill=255)
    disp.image(image)
    disp.show()

    # Delay for DELAY_SECONDS seconds to avoid timeout from adafruit io
    time.sleep(DELAY_SECONDS)
```

Before running the code, you'll need to **set the `ADAFRUIT_IO_KEY` to the Adafruit IO Key** we saved earlier and **set the `ADAFRUIT_IO_USERNAME` to your Adafruit IO Username**.

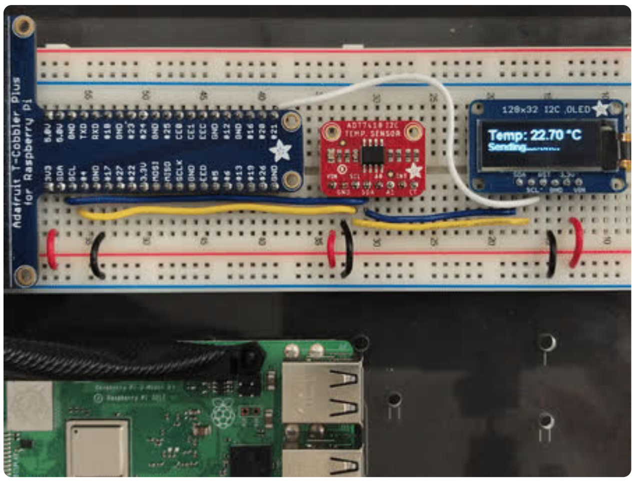
When both of these variables are changed, save the file. You can **run the program by entering the following in your terminal**:

`python adafruit_io_adt7410.py`

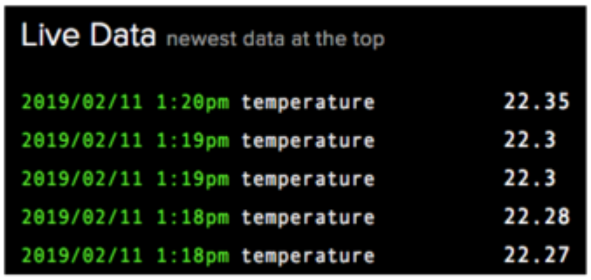
and Press the Enter Key.

You should see the following print out in your terminal

```
Sending temperature 22.26 C to Adafruit IO
Sending temperature 22.27 C to Adafruit IO
Sending temperature 22.25 C to Adafruit IO
Sending temperature 22.27 C to Adafruit IO
Sending temperature 22.28 C to Adafruit IO
```



The display should display the temperature read from the ADT7410 along with the status of the data being sent to Adafruit IO.

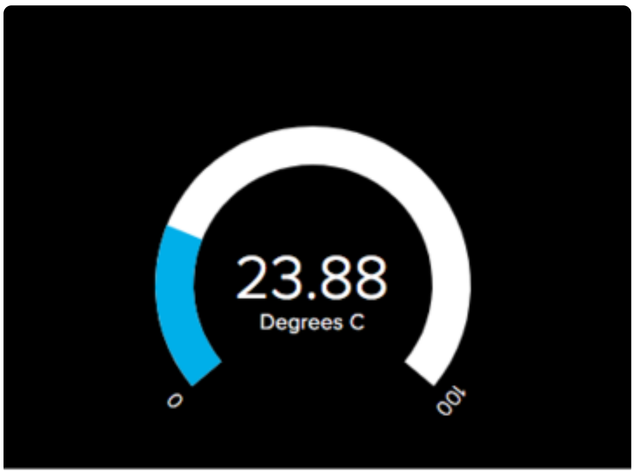


Live Data newest data at the top

```
2019/02/11 1:20pm temperature          22.35
2019/02/11 1:19pm temperature          22.3
2019/02/11 1:19pm temperature          22.3
2019/02/11 1:18pm temperature          22.28
2019/02/11 1:18pm temperature          22.27
```

Next, we're going to check that the data has been received by Adafruit IO. We can do this by visiting the Adafruit IO Monitor page (https://adafru.it/DOK). Every 30 seconds, the page will refresh with the latest values from the ADT7410.

If you want to change the delay, change the variable `DELAY_SECONDS` to the desired delay between sensor reads.



Navigate to the dashboard you created earlier to view the temperature gauge we added. Every 30 seconds, it'll update and display a new value.

For more information about the ADT7410 CircuitPython library, check out the latest docs! (https://adafru.it/DPD)