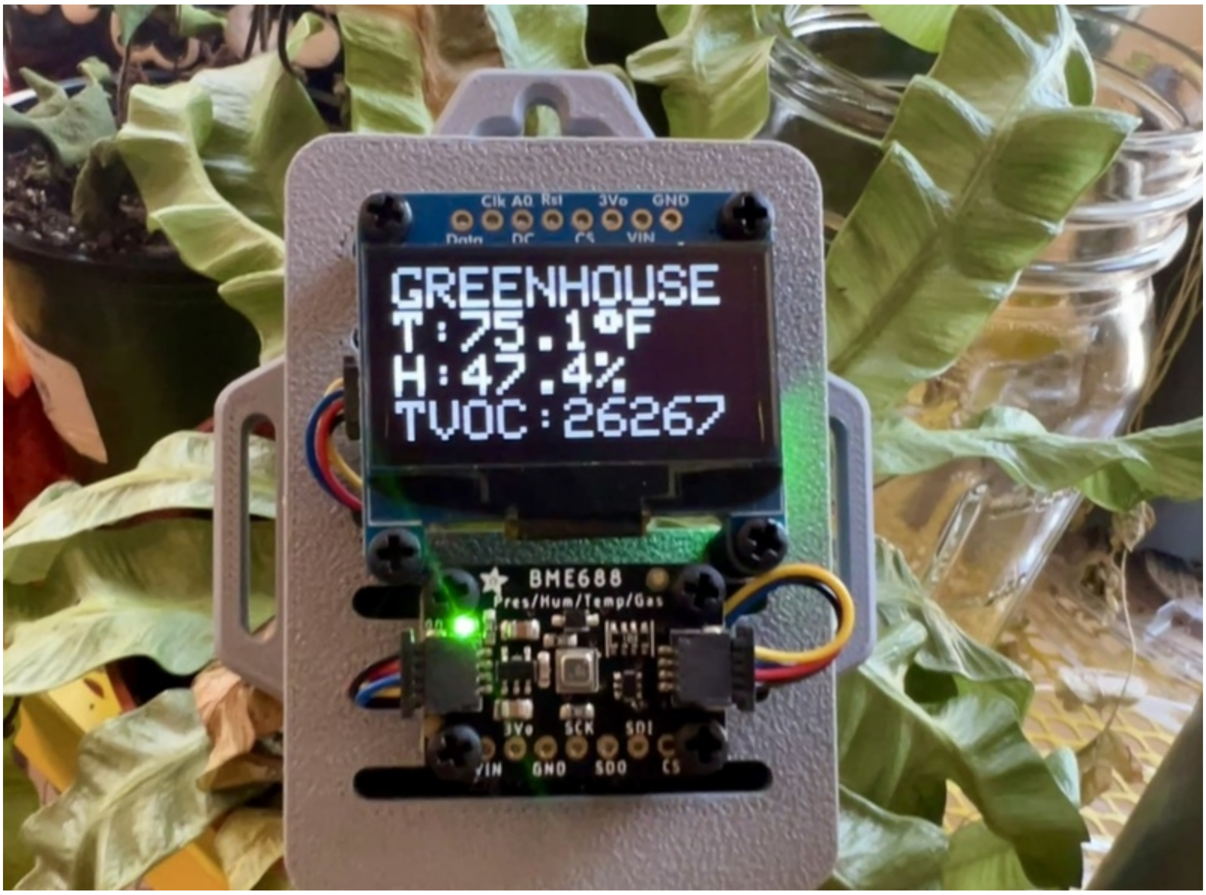
Monitor Your Greenhouse with a No-Code Environmental Sensor

Created by Brent Rubell



https://learn.adafruit.com/monitor-your-greenhouse-with-a-no-code-environmental-sensor

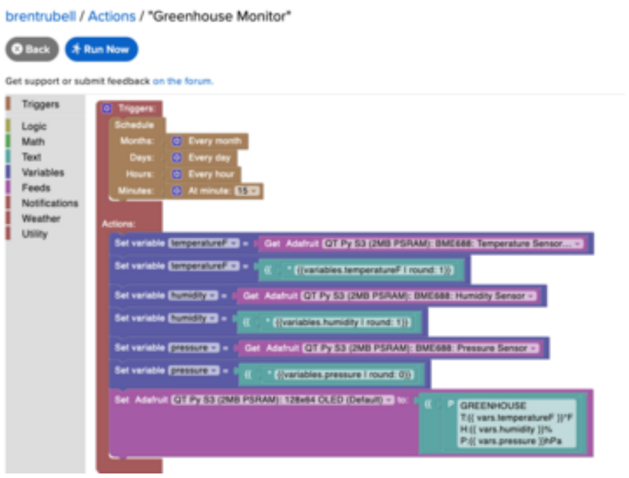
Last updated on 2025-07-14 07:04:12 PM EDT

Table of Contents

Overview

If you've wanted to display sensor readings on your device running WipperSnapper, you're in luck! **This guide will walk you through the process of building a device that reads environmental measurements and displays them on an OLED.**

The process for setting up and using a display on a development board running WipperSnapper is simple, there is no coding or soldering - you just click!



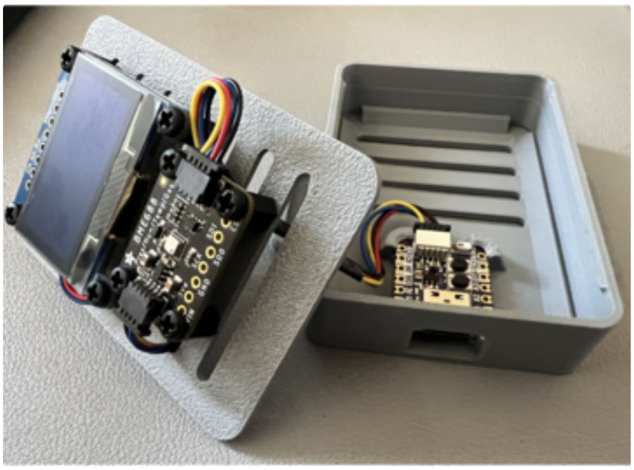
Build Project Logic with Drag and Drop

Actions are a way to do something when a situation occurs. Adafruit IO's actions present a block-based interface, allowing you to create a project simply by dragging and dropping "blocks" onto a diagram.

In this guide, you'll use an Action to build a workflow that:

- Obtains data from sensors every 15 minutes
- Rounds the data, so we can show it on a display
- Configures a display to show the sensor data

All of this is possible in Adafruit IO - no programming required!



Simple Assembly and Wiring

This project uses the snap-on enclosure for Adafruit QT Py and STEMMA QT cables to connect sensors. There is no soldering or difficult assembly required to build this project. A screwdriver is the only tool required!

What is WipperSnapper?

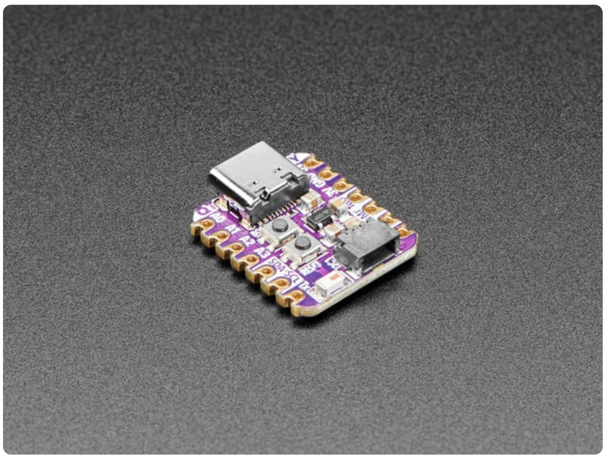
WipperSnapper is a firmware designed to turn any WiFi-capable board into an Internet-of-Things device without programming a single line of code. WipperSnapper connects to [Adafruit IO](https://adafru.it/fsU) (https://adafru.it/fsU), a web platform designed ([by Adafruit!](https://adafru.it/Bo5) (https://adafru.it/Bo5)) to display, respond, and interact with your project's data.

Simply load the WipperSnapper firmware onto your board, add credentials, and plug it into power. Your board will automatically register itself with your Adafruit IO account.

From there, you can add components to your board such as buttons, switches, potentiometers, sensors, and more! Components are dynamically added to hardware, so you can immediately start interacting, logging, and streaming the data your projects produce without writing code.

Parts

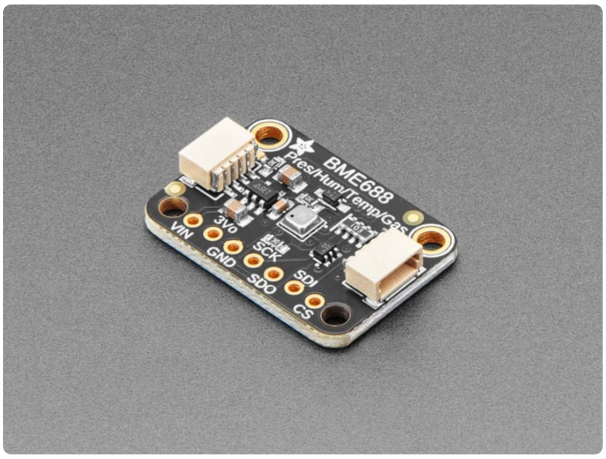
WipperSnapper Firmware supports a wide variety of different boards from Adafruit and others. This greenhouse monitor will use the Adafruit QT Py ESP32 microcontroller board.



[Adafruit QT Py ESP32-S2 WiFi Dev Board with STEMMA QT](https://www.adafruit.com/product/5325)
What has your favorite Espressif WiFi microcontroller, comes with our favorite connector - the STEMMA QT, a chainable I2C port, and has...
https://www.adafruit.com/product/5325

The Bosh BME688 is perfect for this greenhouse monitoring project as it provides temperature, humidity, barometric pressure, and VOC gas sensing capabilities in a sub-$20 package.

If you want to use a different sensor, WipperSnapper firmware supports 104 different types of components.



[Adafruit BME688 - Temperature, Humidity, Pressure and Gas Sensor](https://www.adafruit.com/product/5046)
The long-awaited BME688 from Bosch gives you all the environmental sensing you want in one small package. This little sensor...
https://www.adafruit.com/product/5046

I selected the "Monochrome 1.3" 128x64 OLED graphic display" for this project because the display makes its own light, no backlight is required. This reduces the power required to run the OLED.

At the time of release, WipperSnapper supports the following display components:

OLED Displays with I2C:

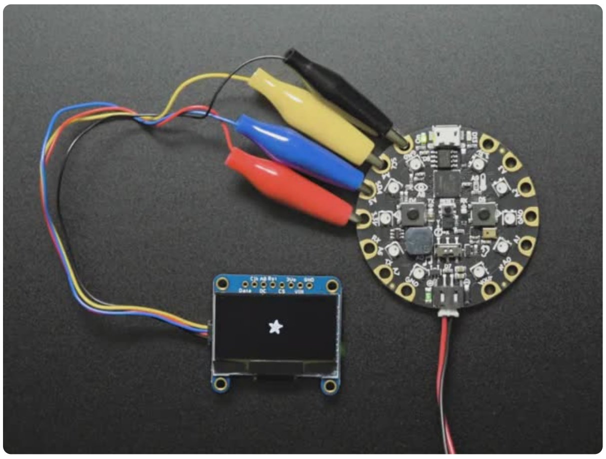
- Monochrome 1.3" 128x64 OLED graphic display - STEMMA QT / Qwiic (http://adafru.it/938)
- Monochrome 0.91" 128x32 I2C OLED Display - STEMMA QT / Qwiic (http://adafru.it/4440)
- Monochrome 0.96" 128x64 OLED Graphic Display - STEMMA QT PID: 326 (http://adafru.it/326)

LED Displays with I2C Backpacks:

- Quad Alphanumeric Display - Red 0.54" Digits w/ I2C Backpack - STEMMA QT / Qwiic (http://adafru.it/1911)
- Assembled Adafruit 0.56" 4-Digit 7-Segment Display - I2C Backpack - STEMMA QT (http://adafru.it/5601)
- i2c / SPI character LCD backpack - STEMMA QT / Qwiic (http://adafru.it/292)

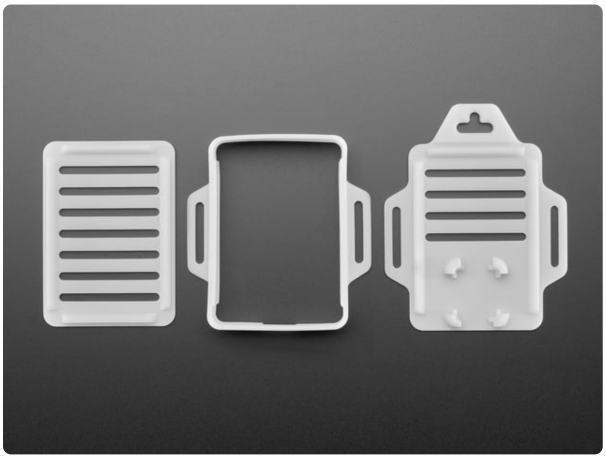
Character LCD Displays with I2C Backpacks:

- Assembled Standard LCD 16x2 (http://adafru.it/1447)
- RGB backlight negative LCD 16x2 (http://adafru.it/399)
- Standard LCD 20x4 (http://adafru.it/198)
- RGB backlight positive LCD 16x2 (http://adafru.it/398)



Monochrome 1.3" 128x64 OLED graphic display - STEMMA QT / Qwiic
These displays are small, only about 1.3" diagonal, but very readable due to the high contrast of an OLED display. This display is made of 128x64 individual white OLED pixels,...
https://www.adafruit.com/product/938

The following parts are required to build the project's enclosure:



[Snap-on Enclosure for Adafruit QT Py](https://www.adafruit.com/product/6002)

Here is a cute and minimal enclosure for your QT Py to keep it safe during use and transport. This case has been...

https://www.adafruit.com/product/6002

1 x Machine Screw and Stand-off Set

Black Nylon Machine Screw and Stand-off Set – M2.5 Thread

https://www.adafruit.com/product/3299

Optional Parts

1 x Screwdriver

Adafruit Pocket Screwdriver - Black

https://www.adafruit.com/product/3284

1 x USB A to USB C Cable

Pink and Purple Woven USB A to USB C Cable - 2 meters long

https://www.adafruit.com/product/5044

1 x Adafruit IO+ Subscription Pass

Adafruit IO+ Subscription Pass – One Year

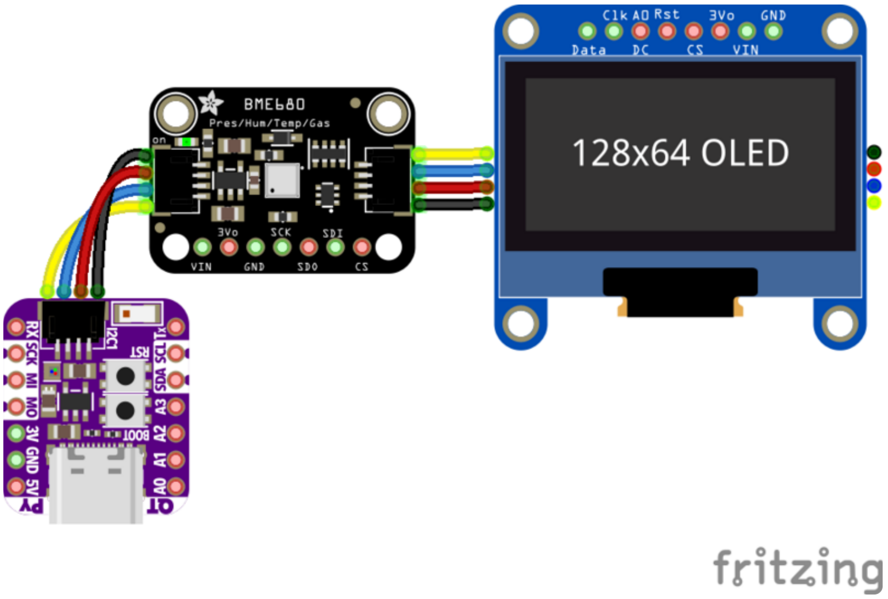
https://www.adafruit.com/product/3792

Assemble

Circuit Diagram

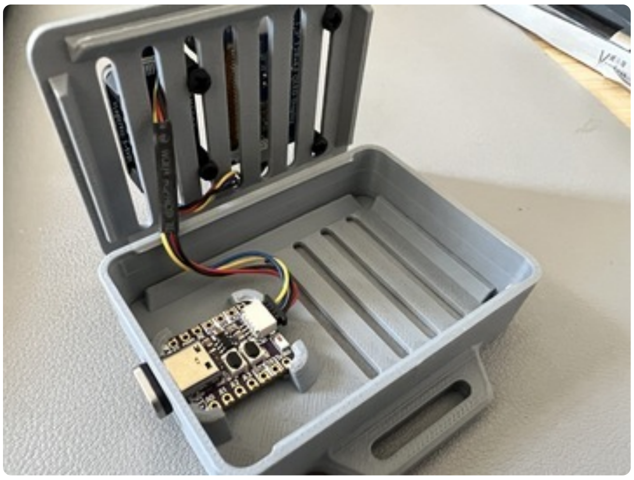
The diagram below provides a general visual reference for wiring the project's components. The project uses STEMMA QT (https://adafru.it/Ft4) cables and connectors to provide a no-soldering solution for wiring.

NOTE: The BME680 is shown on this diagram instead of the BME688. The wiring for both is identical.



Assembly

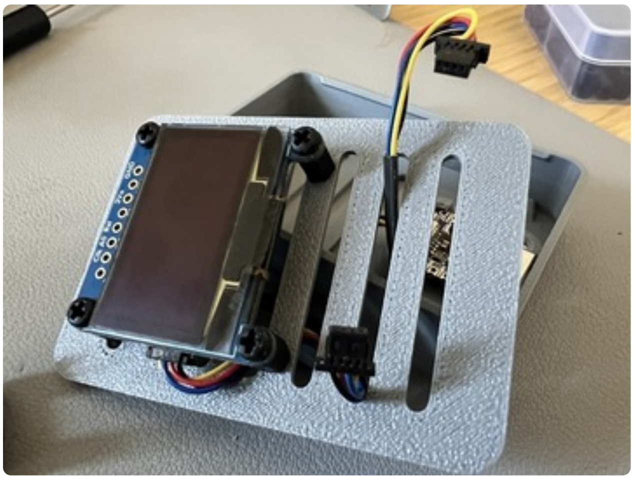
The Snap-on Enclosure for the Adafruit QT Py does a lot of heavy lifting for us as it provides a both enclosure for the QT Py and mounting rails for the display and the BME280 sensor .



Assemble the snap-enclosure frame. These parts should press-fit together.

Inside the frame, there are four mounting tabs for the QT Py. Take your QT Py and snap it into the bottom piece of the frame.

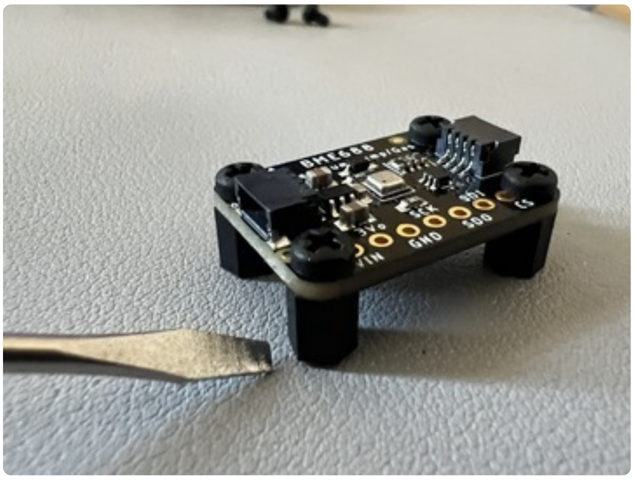
Then attach a STEMMA cable to the QT Py.

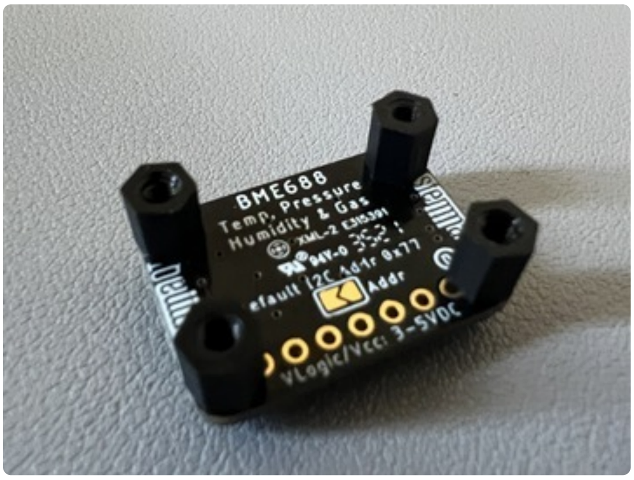


Before attaching the OLED and BME688 to the enclosure, you may want to route two STEMMA QT cables through the openings of the enclosure's front panel.

Route the cable connected to the QT Py through a row on the enclosure's bottom row. This cable will connect the QT Py to the BME688.

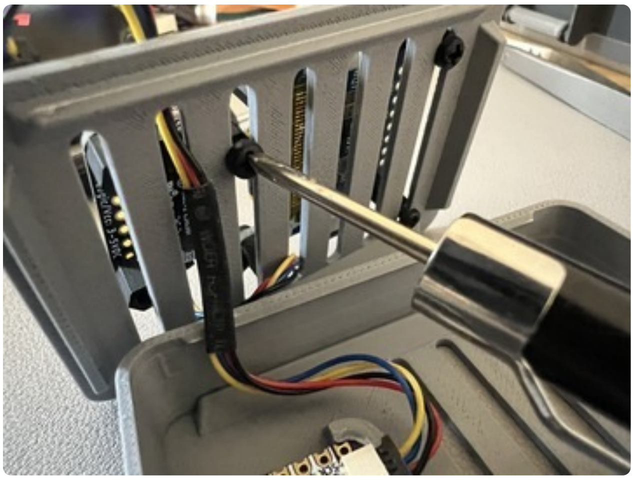
Route another QT Py cable through the same row, and then route it through a row towards the top. This cable will connect the BME688 to the OLED.





Push an 4mm screw through one of the BME688's mounting holes. Using a Philips-head screwdriver, screw it (on the other side) into a 20 x 6mm long F-F hex standoff.

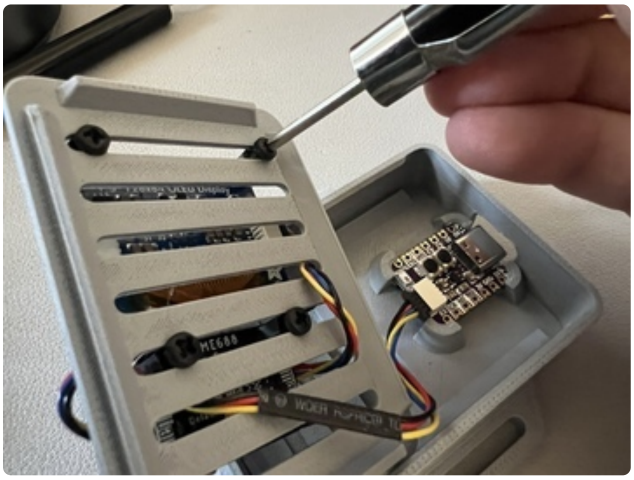
Repeat this for the remaining 3 mounting holes.



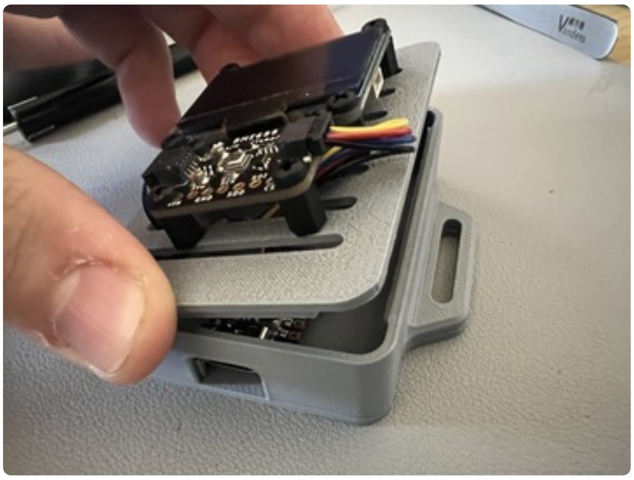
Line the BME688 up with the front-panel. Make sure the stand-off's hole is lined up with the row.

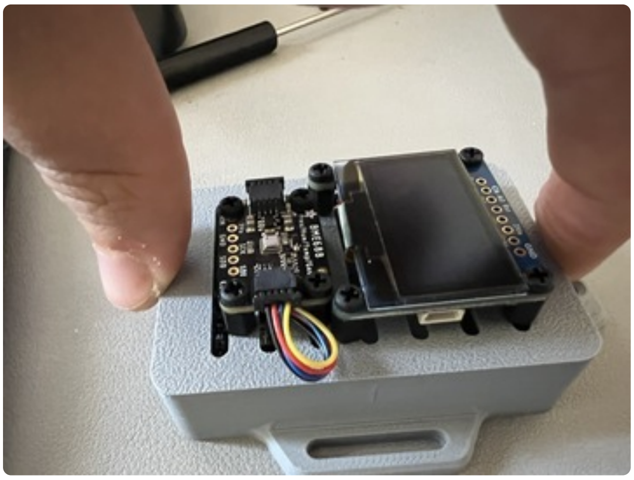
Screw a 4mm screw through the back of the enclosure, into the standoff.

Repeat this for the remaining 3 screws.



To mount the OLED display, repeat the steps above.





Check your wiring and ensure everything is connected!

Place the cover on the enclosure and press it down until it snaps.

WipperSnapper Setup

> The WipperSnapper firmware and ecosystem are in BETA and are actively being developed to add functionality, more boards, more sensors, and fix bugs. We encourage you to try out WipperSnapper with the understanding that it is not final release software and is still in development. If you encounter any bugs, glitches, or difficulties during the beta period, or with this guide, please contact us via http://io.adafruit.com/support

What is WipperSnapper

WipperSnapper is a firmware designed to turn any WiFi-capable board into an Internet-of-Things device without programming a single line of code. WipperSnapper connects to Adafruit IO (https://adafru.it/fsU), a web platform designed (by Adafruit! (https://adafru.it/Bo5)) to display, respond, and interact with your project's data.

Simply load the WipperSnapper firmware onto your board, add credentials, and plug it into power. Your board will automatically register itself with your Adafruit IO account.

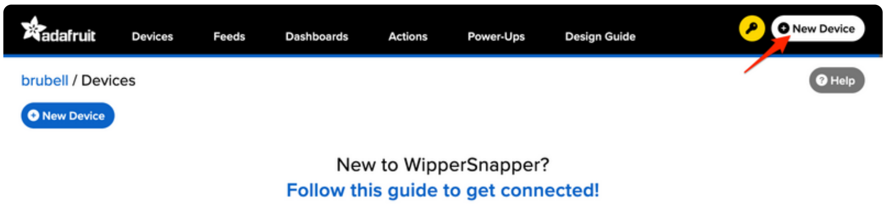
From there, you can add components to your board such as buttons, switches, potentiometers, sensors, and more! Components are dynamically added to hardware, so you can immediately start interacting, logging, and streaming the data your projects produce without writing code.

Sign up for Adafruit.io

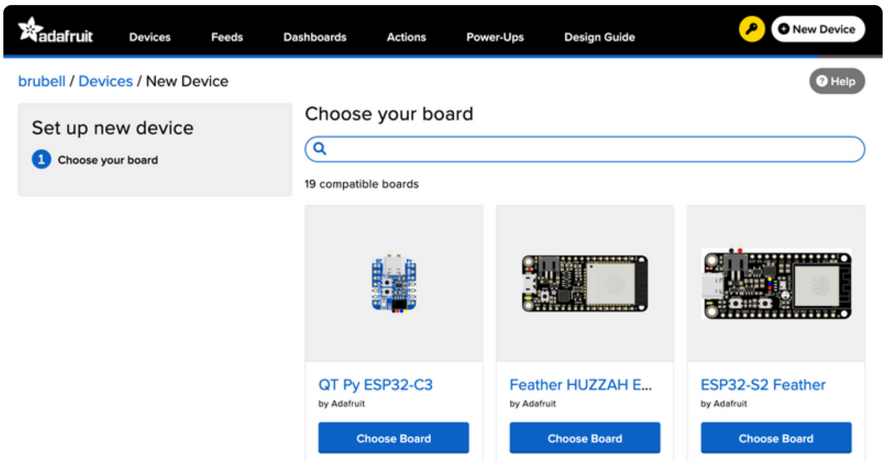
You will need an Adafruit IO account to use WipperSnapper on your board. If you do not already have one, head over to io.adafruit.com (https://adafru.it/fsU) to create a free account.

Add a New Device to Adafruit IO

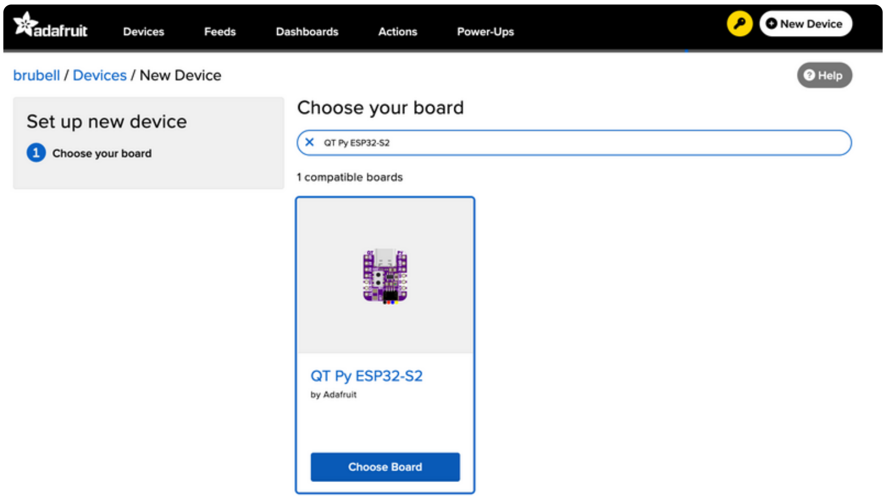
Log into your Adafruit IO (https://adafru.it/fsU) account. Click the New Device button at the top of the page.



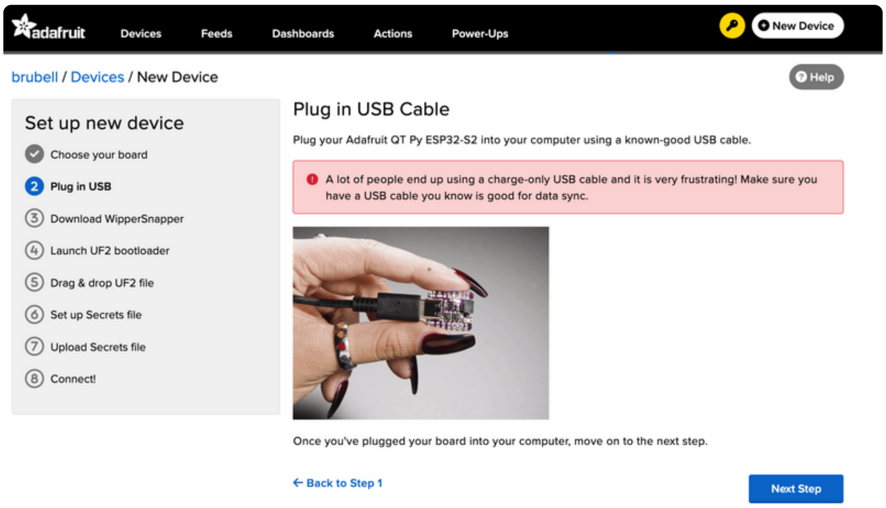
After clicking New Device, you should be on the board selector page. This page displays every board that is compatible with the WipperSnapper firmware.



In the board selector page's search bar, search for the QT Py ESP32-S2. Once you've located the board you'd like to install WipperSnapper on, click the Choose Board button to bring you to the self-guided installation wizard.

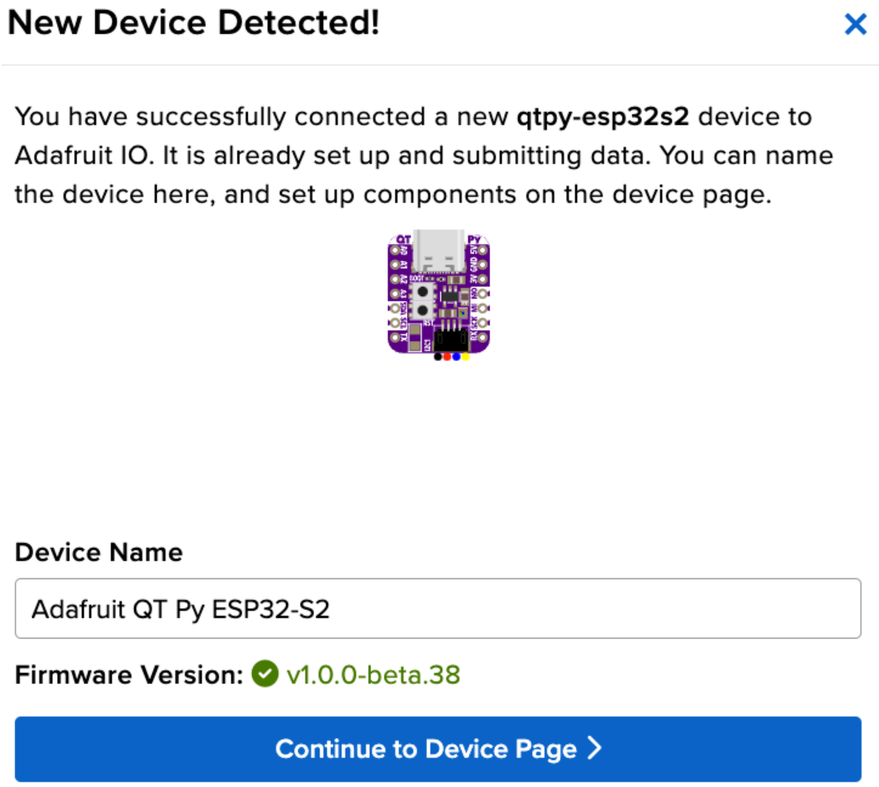


Follow the step-by-step instructions on the page to install Wippersnapper on your device and connect it to Adafruit IO.

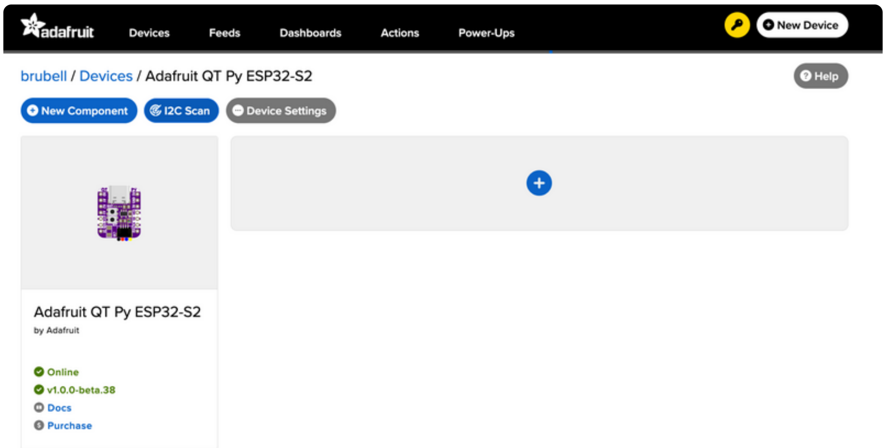


If the installation was successful, a popover should appear displaying that your board has successfully been detected by Adafruit IO.

Give your board a name and click "Continue to Device Page".



You should be brought to your board's device page.



Feedback

Adafruit.io WipperSnapper is in **beta** and you can help improve it!

If you have suggestions or general feedback about the installation process - visit https://io.adafruit.com/support (https://adafru.it/Sgb), click "Contact Adafruit IO Support" and select "I have feedback or suggestions for the WipperSnapper Beta".

Troubleshooting

If you encountered an issue during installation, please try the steps below first.

If you're still unable to resolve the issue, or if your issue is not listed below, get in touch with us directly at https://io.adafruit.com/support (https://adafru.it/Sgb). Make sure to click "Contact Adafruit IO Support" and select "There is an issue with WipperSnapper. Something is broken!"

I don't see my board on Adafruit IO, it is stuck connecting to WiFi

First, make sure that you selected the correct board on the board selector.

Next, please make sure that you entered your WiFi credentials properly, there are no spaces/special characters in either your network name (SSID) or password, and that you are connected to a 2.4GHz wireless network.

If you're still unable to connect your board to WiFi, please make a new post on the WipperSnapper technical support forum with the error you're experiencing, the LED colors which are blinking, and the board you're using. (https://adafru.it/V6a)

I don't see my board on Adafruit IO, it is stuck "Registering with Adafruit IO"

Try hard-resetting your board by unplugging it from USB power and plugging it back in.

If the error is still occurring, please make a new post on the WipperSnapper technical support forum with information about what you're experiencing, the LED colors which are blinking (if applicable), and the board you're using. (https://adafru.it/V6a)

"Uninstalling" WipperSnapper

WipperSnapper firmware is an application that is loaded onto your board. There is nothing to "uninstall". However, you may want to "move" your board from running WipperSnapper to running Arduino or CircuitPython. You also may need to restore your board to the state it was shipped to you from the Adafruit factory.

Moving from WipperSnapper to CircuitPython

Follow the steps on the Installing CircuitPython page (https://adafru.it/Amd) to install CircuitPython on your board running WipperSnapper.

- If you are unable to double-tap the RST button to enter the UF2 bootloader, follow the "Factory Resetting a WipperSnapper Board" instructions below.

Uploading this sketch will overwrite WipperSnapper. If you want to re-install WipperSnapper, follow the instructions at the top of this page.

Moving from WipperSnapper to Arduino

If you want to use your board with Arduino, you will use the Arduino IDE to load any sketch onto your board.

First, follow the page below to set up your Arduino IDE environment for use with your board.

Setup Arduino IDE

https://adafru.it/-hE

Then, follow the page below to upload the "Arduino Blink" sketch to your board.

Upload Arduino Blink Sketch

https://adafru.it/10aT

Uploading this sketch will overwrite WipperSnapper. If you want to re-install WipperSnapper, follow the instructions at the top of this page.

Factory Resetting a WipperSnapper Board

Sometimes, hardware gets into a state that requires it to be "restored" to the original state it shipped in. If you'd like to get your board back to its original factory state, follow the guide below.

Factory Reset

https://adafru.it/XRe

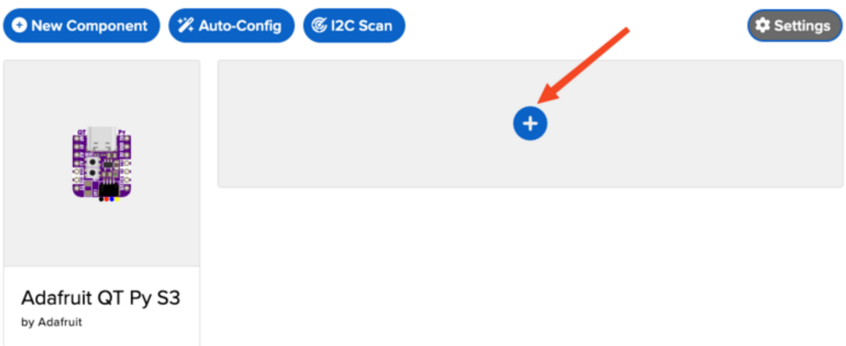
Adafruit IO Device Configuration

Before we set up an Action, we need to attach and configure components (like the display and sensor) to the QT Py.

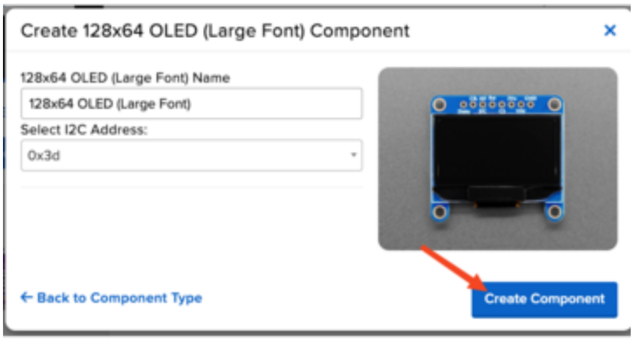
I2C Display Components, like OLED or Character LCD displays, are supported only on WipperSnapper Firmware Beta 103 or later. If your device is running an older firmware version, please update it before continuing this guide.

Add a Display Component

To add the **Monochrome 128x64 OLED** to your QT Py, navigate to the device page. Click the "+" to bring up the component picker.



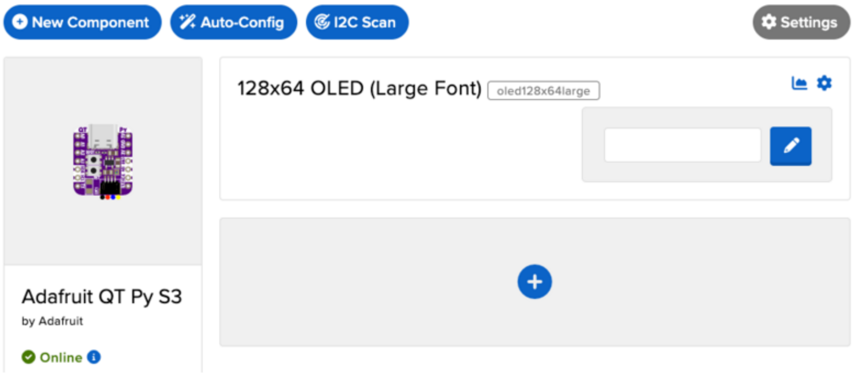
On the Component Picker, Search for the 128x64 OLED and **select the 128x64 OLED (Large Font) component**. This component uses the large font size.



If you want a smaller font size to fit more data, you can select the 128x64 OLED (Default).

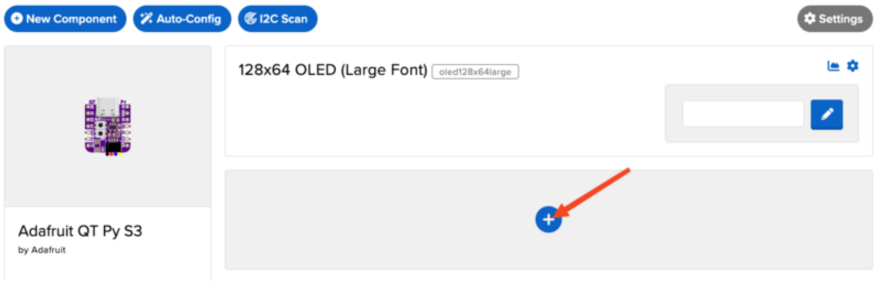
Click Create Component.

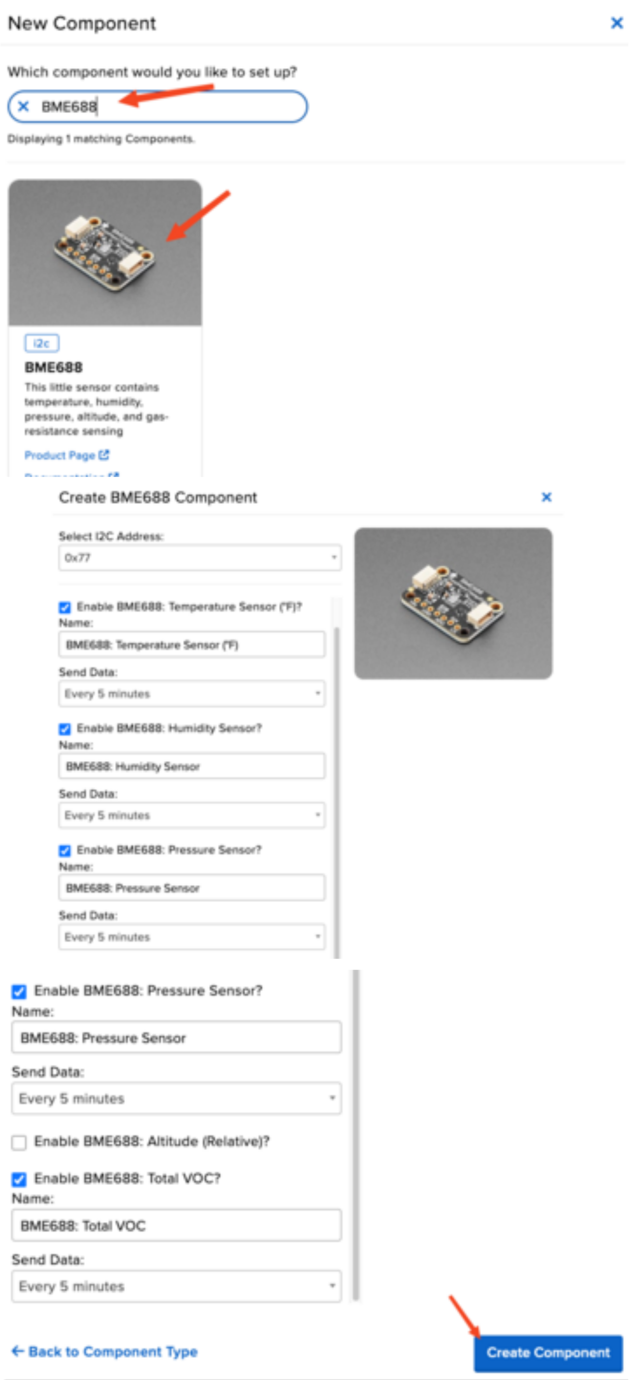
The 128x64 OLED component should appear on your device page.



Add a BME680 Component

To add the BME680 component, click the "+" icon.





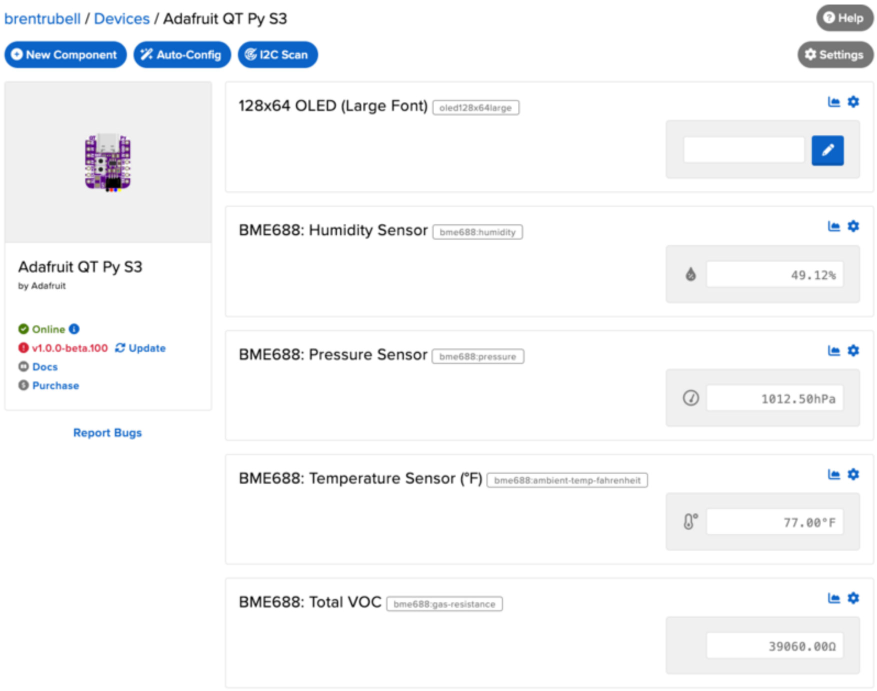
On the Component Picker, search for the BME688 and click it.

From the next page, you can choose which sensors are read, and how often.

For this project, I wanted the QT Py to read the temperature, relative humidity, air pressure, and total volatile organic compounds (TVOC) and send these measurements to Adafruit.io every 5 minutes.

Click Create Component.

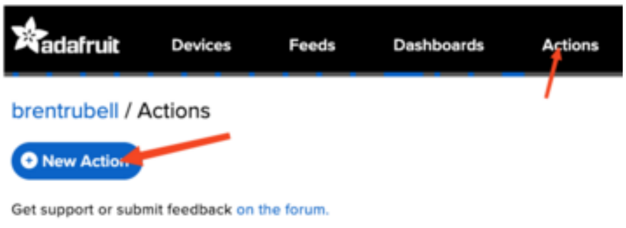
Make sure the Device Page looks like the following screenshot before moving on to set up an Action.



Set up an Adafruit IO Action

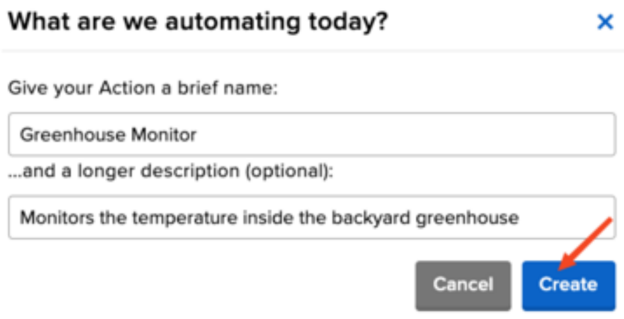
Let's use Adafruit IO's new (Blockly-based) Actions to show the values of the sensors on the OLED display.

- [If you do not know what Adafruit IO Actions are - check out this guide here >>>](https://adafru.it/1a5t) (https://adafru.it/1a5t)



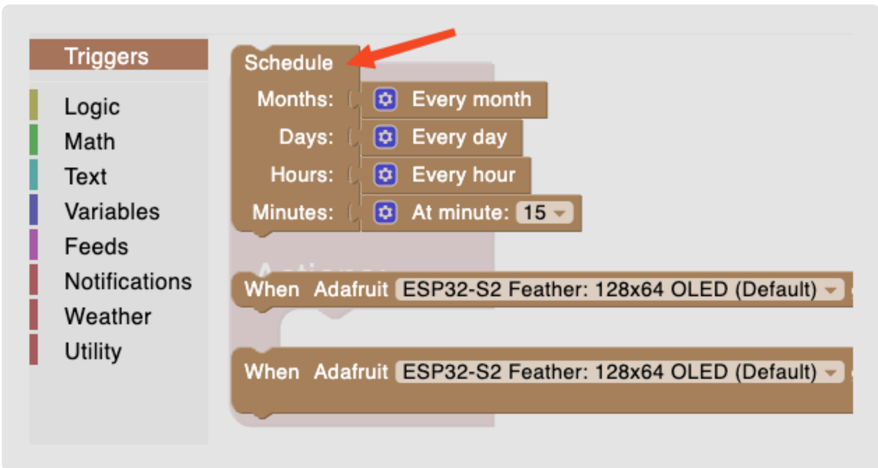
Navigate to the Actions page on Adafruit IO. **Click** New Action.

Give your action a name (I named mine Greenhouse Monitor) and **click** Create.

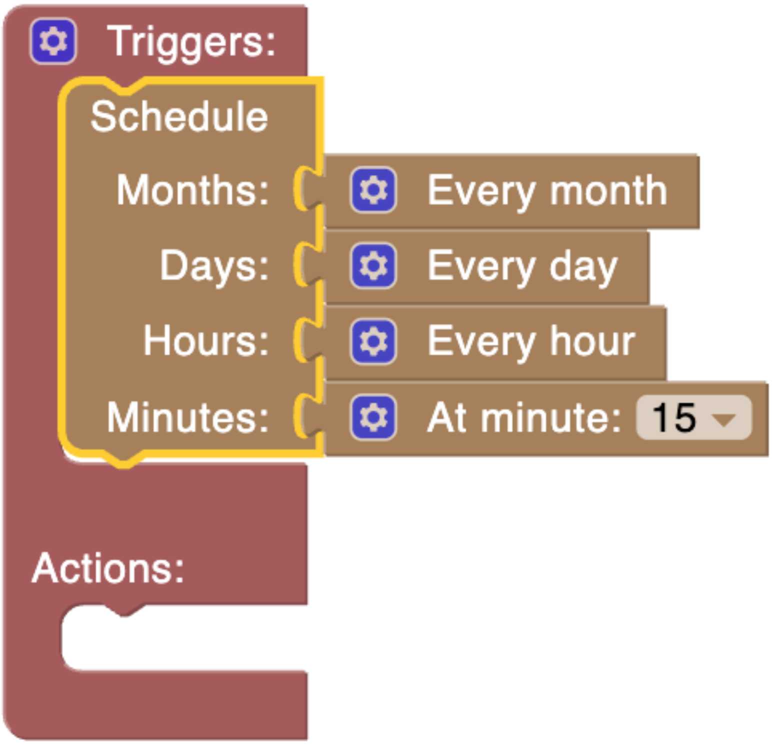


Let's configure this workflow to execute every 15 minutes.

To do this, expand the Block Toolbox by clicking **Triggers** and **select** the Schedule block.



Drag the **Schedule** block under the Triggers workflow. By default, it is configured to execute every 15 minutes.



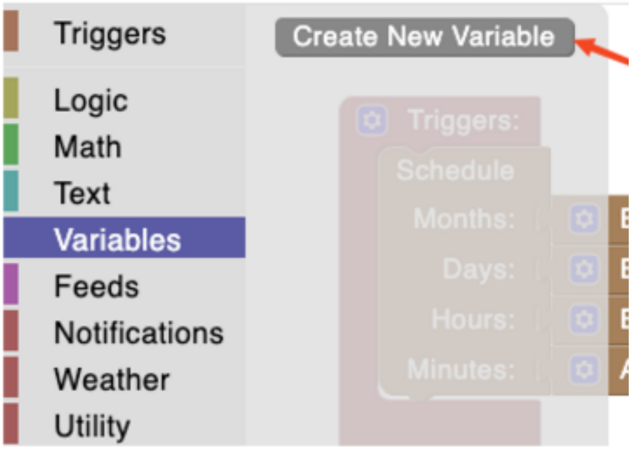
Modifying Sensor Values

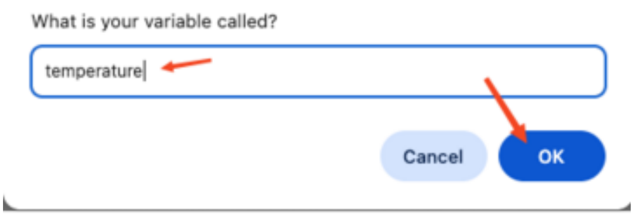
The BME688 produces precise measurements. However, these measurements often have too many decimal points to display on an OLED.

Before displaying these sensor values, we are going to take these measurements and round them to fit on the OLED's screen.

Adafruit QT Py S3

Feed Name	Key	Last value
128x64 OLED (Large Font)	qtpy-esp32s3-n4r212086...	GREENHOUSE T:0°F H:0...
BME688: Humidity Sensor	qtpy-esp32s3-n4r212086...	43.963565826416016
BME688: Pressure Sensor	qtpy-esp32s3-n4r212086...	1012.5599975585938
BME688: Temperature Sensor (°F)	qtpy-esp32s3-n4r212086...	78.59912872314453
BME688: Total VOC	qtpy-esp32s3-n4r212086...	36425





We'll need to store the component's feed value in a variable. Once we do that, we can modify the variable (in order to truncate it to a specific number of decimal places) and then reference when we print it to the display.

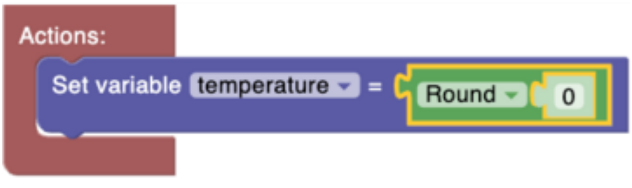
On the Toolbox, **Click** the Variables tab.

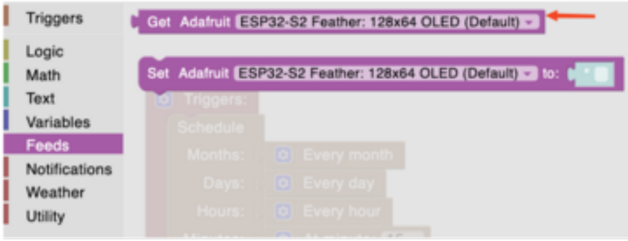
Click Create New Variable.

Enter the name `temperature`.

On the Toolbox, Click the Math tab

Drag the Round block to the right side of the Set Variable block's "=" sign





On the Toolbox, **Click** the Feeds tab.

Drag the Get Feed block to the diagram and place it inside the Round Block.

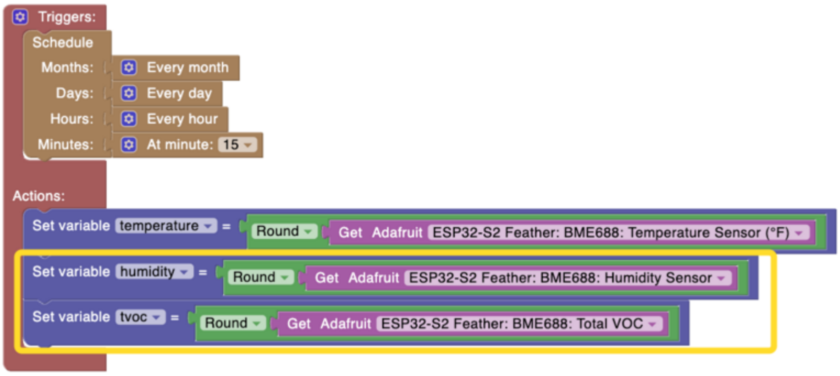
Set the feed to QT Py S3: BME680 Temperature Sensor**.**

When the Actions' Actions section runs, it will set the variable to the last value saved to the feed (i.e: the most recent temperature value from the BME688).

Ensure your diagram should looks like the following screenshot before moving on:

Next, repeat the steps above for the BME688's humidity sensor and TVOC sensor.

Your diagram should look like the following.

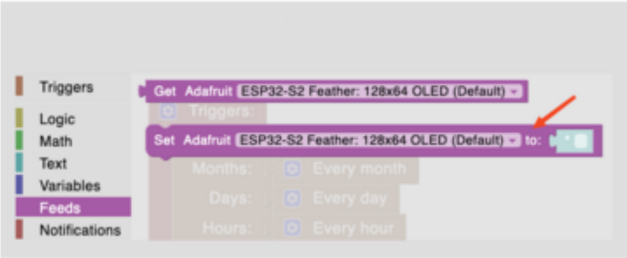


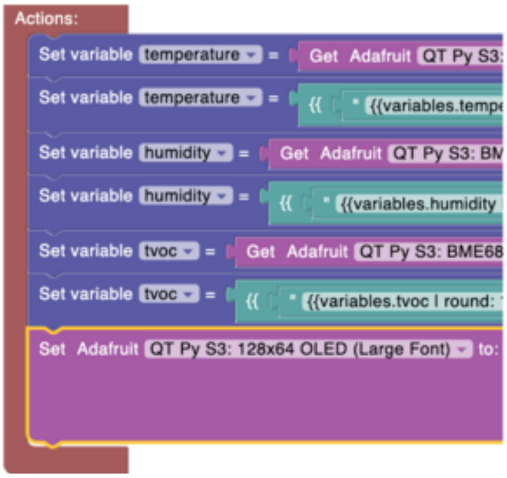
Showing Temperature Values on the Display

Finally, we'll want to print the temperature values to the display.

- If you do not have an OLED - the process for this is the same for any "Display Component" in Adafruit IO.

Like the sensors component, the OLED component is an Adafruit IO Feed that you can write to. What's displayed on the OLED is the last value written to the feed.

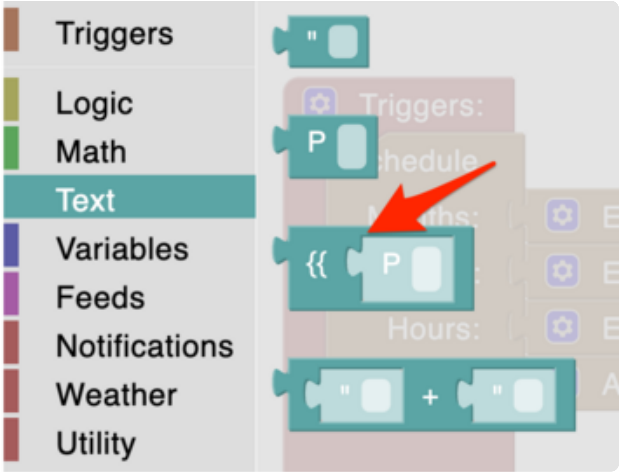




On the Toolbox, **Click** the Feeds tab.

Drag the Set Feed block to the diagram and place it underneath your Actions

Set the feed to QT Py S3: 128x64 OLED (Large Font)**.**



Next, we'll use a combination of a the template block and paragraph block to display formatted values (the variables from the sensor and static text)

On the Toolbox, **Click** the Text tab.

From the Text tab, **select the Template block.**

Drag the Template block to the diagram and place it inside the Set Feed Block.

To tell the display what to show, you'll need to write to the paragraph block. This block lives within the Template block but is presently grayed out.

The 128x64 OLED's **Large Font** mode can write up to 4 lines of text on the display.

For the first line of the paragraph block, I added a label called Greenhouse. The next 3 lines of the display are populated with the temperature, humidity and TVOC values.

To render a variable on the display, you'll reference a variable by using the following syntax: `{{vars.variable}}`.

The string rendered by the following line:

`T:{{ vars.temperature }}°F`

Would look like this on the display:

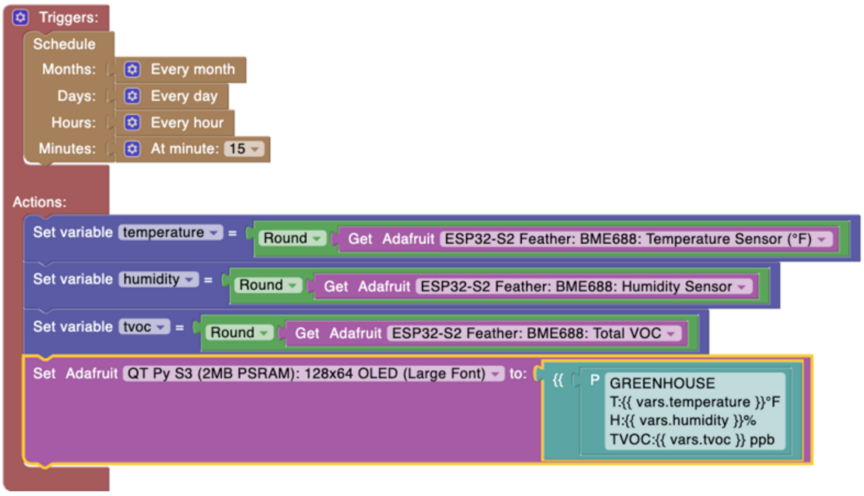
T: 74°F

The snippet below is the value of the paragraph block, used by this project.

```
GREENHOUSE
T:{{ vars.temperature }}°F
```

```
H:{{ vars.humidity }}%
TVOC:{{ vars.tvoc }} ppb
```

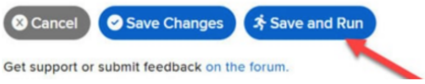
After placing the text above within the paragraph block, the final Action should look like the following screenshot.



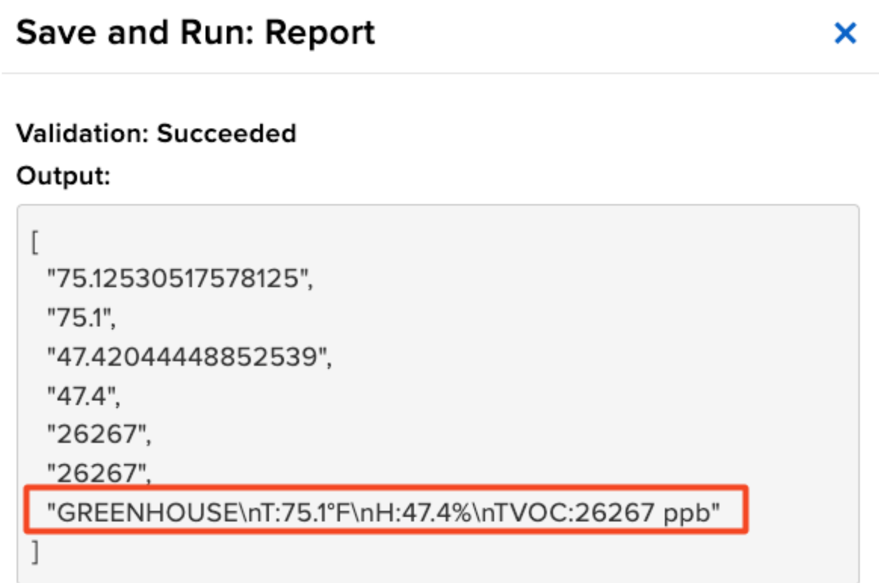
With the Action complete, let's move on to using it!

Usage

Once you have your new Adafruit IO Action set up like above, click the Save and Run button.



Once the Action runs, you should see a report showing the output of each block:



You should also see the OLED update, reflecting the final string you set: