



IoT & Observability

Monitoring our world with Grafana

Dan Cech

Internet of Things

"IoT involves extending Internet connectivity to any range of traditionally dumb or non-internet-enabled physical devices and everyday objects."

Connected devices make it easy to connect, monitor & control your world!

In this workshop we'll explore how we can use Grafana together with the ESP32 platform to monitor our surroundings and visualize data over time.

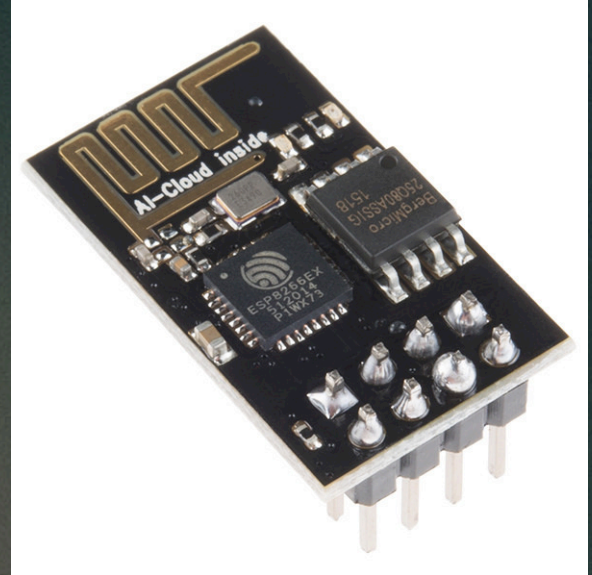


ESP-what?

In 2014 Espressif Systems launched the ESP8266, an all-in-one chip with a 32-bit CPU and WiFi

It became hugely popular both with hobbyists and device manufacturers because of its capabilities and low per-unit cost.

It's also possible to program the ESP8266 with the Arduino IDE, and to use the huge Arduino library.



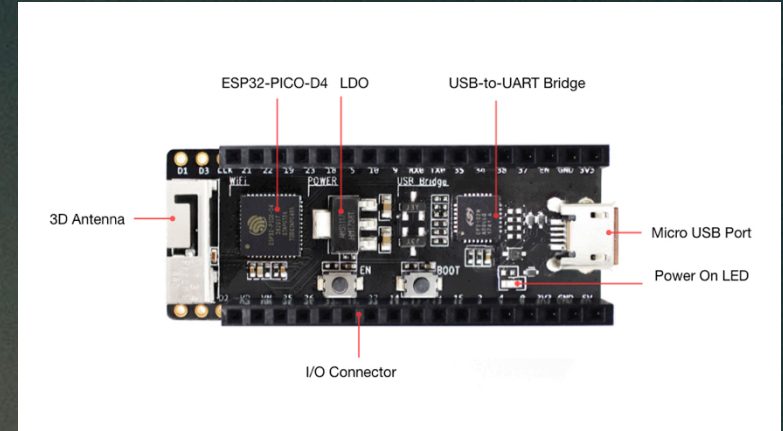
<https://commons.wikimedia.org/wiki/File:ESP-01.jpg>

ESP32

In 2016, Espressif released the successor to the ESP8266, the ESP32:

- Faster dual-core CPU
- More memory
- Bluetooth 4.2
- More connectivity options!

This tiny chip is amazing, and with the ESP32 Arduino core it too can be programmed like an Arduino



<https://docs.espressif.com/projects/esp-idf/en/latest/get-started/get-started-pico-kit.html>

What's in the bag?

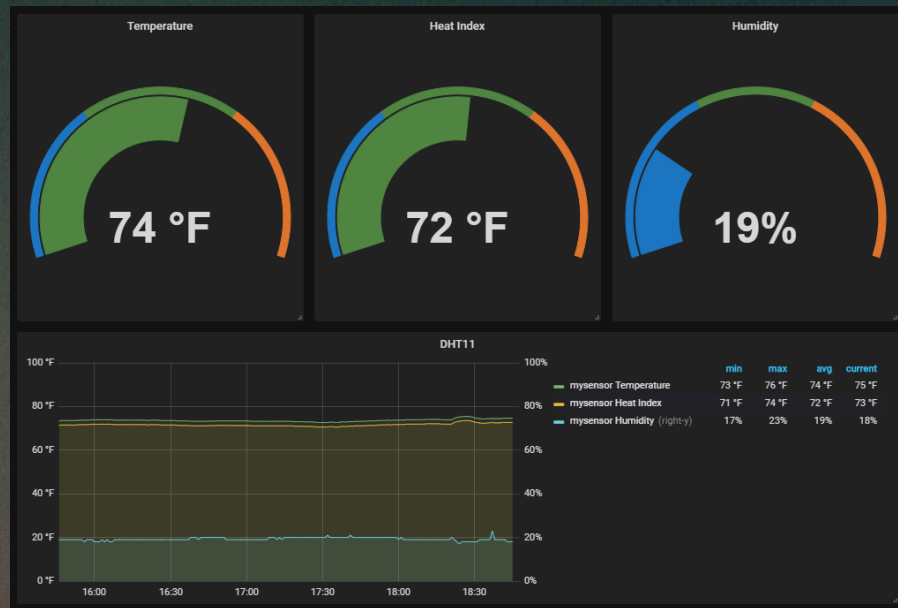
- ESP32-PICO-KIT V4.1
The brains of the operation
- DHT11 Temperature and Humidity sensor
What we'll use to gather data
- Jumper wires
- Micro-USB cable
Plumbing

If you only have USB-C, let us know and we have USB-C to USB-A adapters



So, what are we going to do with it?

- Observe temperature and humidity in the room on a regular basis
- Calculate the Heat Index
- Send the data to a Time Series DataBase (TSDB) for storage
 - Graphite
 - Prometheus
- Graph it with Grafana!



Hardware

USB

- Plug the USB cable into the ESP32

Power

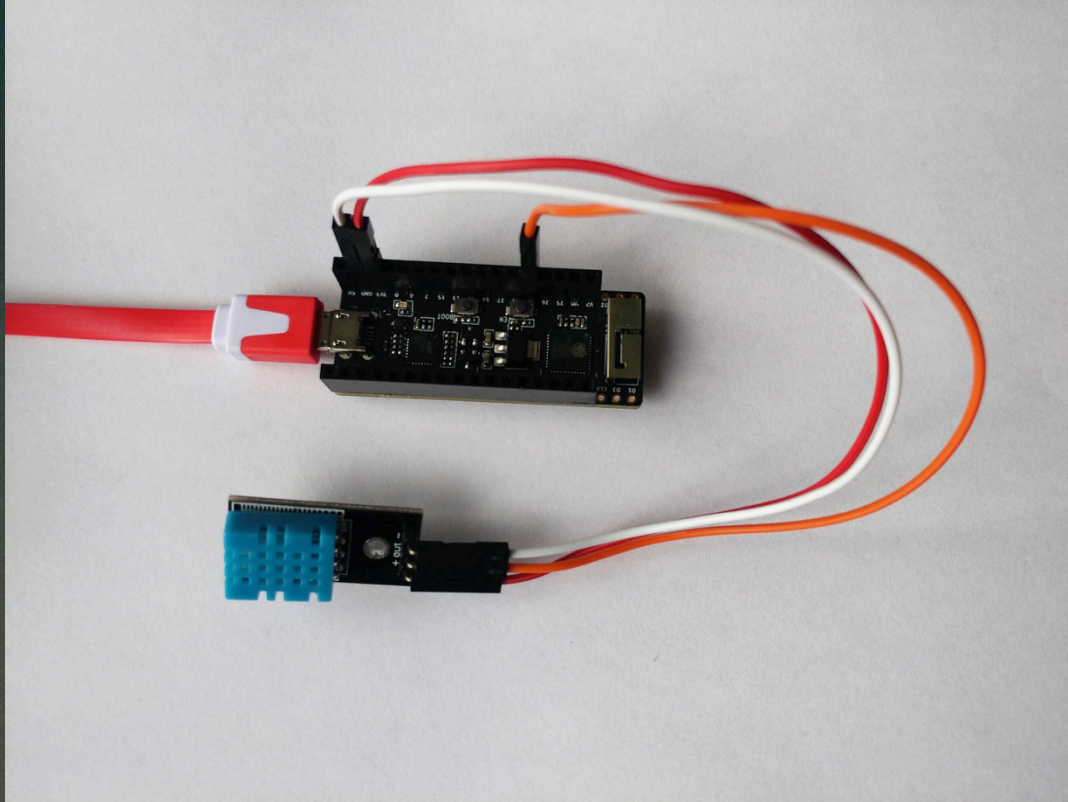
- Connect the `-` pin on the DHT11 to `GND` on the ESP32 (2nd pin)
- Connect the `+` pin on the DHT11 to `3V3` on the ESP32 (3rd pin)

Signal

- Connect the `out` pin on the DHT11 to pin `32` on the ESP32 (13th pin)



Hardware



Getting Started

All the links you'll need are at:

<https://github.com/DanCech/IoTWorkshop>



CP210x Driver

Note: Only required if your OS doesn't recognize the USB Serial automatically

Download the Silicon Labs CP210x UART to USB Driver (URL in README)

<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

OSX: Unzip, open the .dmg image

Double click the .pkg and follow the prompts

When prompted you will have to open Security & Privacy and allow software from
“Silicon Laboratories Inc”



The Arduino IDE

- arduino.cc
- Download the 1.8.8 IDE for your OS
- Install it



The screenshot shows the Arduino website's 'SOFTWARE' section. The navigation bar includes links for HOME, STORE, SOFTWARE, EDUCATION, RESOURCES, COMMUNITY, and HELP. The 'SOFTWARE' dropdown menu is open, showing 'ONLINE TOOLS' and 'DOWNLOADS'. The 'DOWNLOADS' link is highlighted. Below the navigation bar, there is a section for the 'ARDUINO WEB EDITOR' with a description: 'Start coding online with the Arduino Web Editor, save your sketches in the cloud, and always have the most up-to-date version of the IDE, including all the contributed libraries and support for new Arduino boards.' There are buttons for 'GETTING STARTED' and 'CODE ONLINE'. Below this, there is a section titled 'Download the Arduino IDE' with a large Arduino logo and the text 'ARDUINO 1.8.8'. The text describes the IDE as open-source and available for Windows, Mac OS X, and Linux. It also mentions that the software can be used with any Arduino board and refers to the 'Getting Started' page for installation instructions. On the right side of this section, there are links for 'Windows installer', 'Windows ZIP file', 'Windows app', 'Mac OS X 10.8 Mountain Lion or newer', 'Linux 32 bits', 'Linux 64 bits', 'Linux ARM', 'Release Notes', 'Source Code', and 'Checksums (sha512)'. The URL 'https://www.arduino.cc/en/Main/Software' is visible at the bottom of the page.

ARDUINO 1.8.8

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. Refer to the [Getting Started](#) page for installation instructions.

Windows installer, for Windows XP and up
Windows ZIP file for non admin install

Windows app Requires Win 8.1 or 10
Get ARM

Mac OS X 10.8 Mountain Lion or newer

Linux 32 bits
Linux 64 bits
Linux ARM

Release Notes
Source Code
Checksums (sha512)

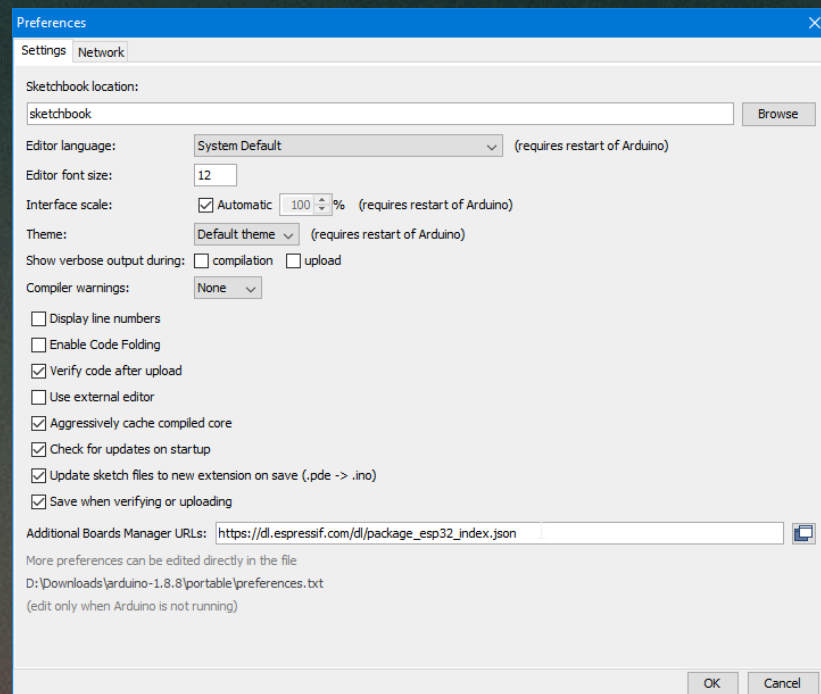
<https://www.arduino.cc/en/Main/Software>

The ESP32 core

This open-source board definition adds support in the Arduino IDE for programming ESP32 boards.

<https://github.com/espressif/arduino-esp32>

- Open File > Preferences
- Add a Boards Manager URL
- https://dl.espressif.com/dl/package_esp32_index.json



The ESP32 Core (cont)

Open Boards Manager

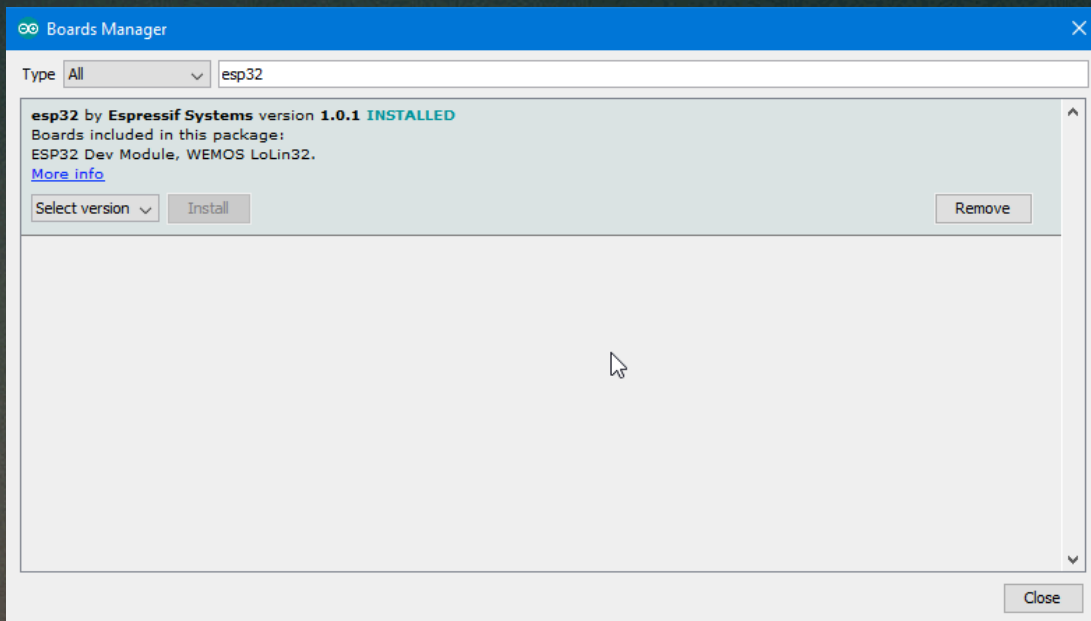
Tools -> Boards: *

-> Board manager

Wait for it to update the list

Search for ESP32

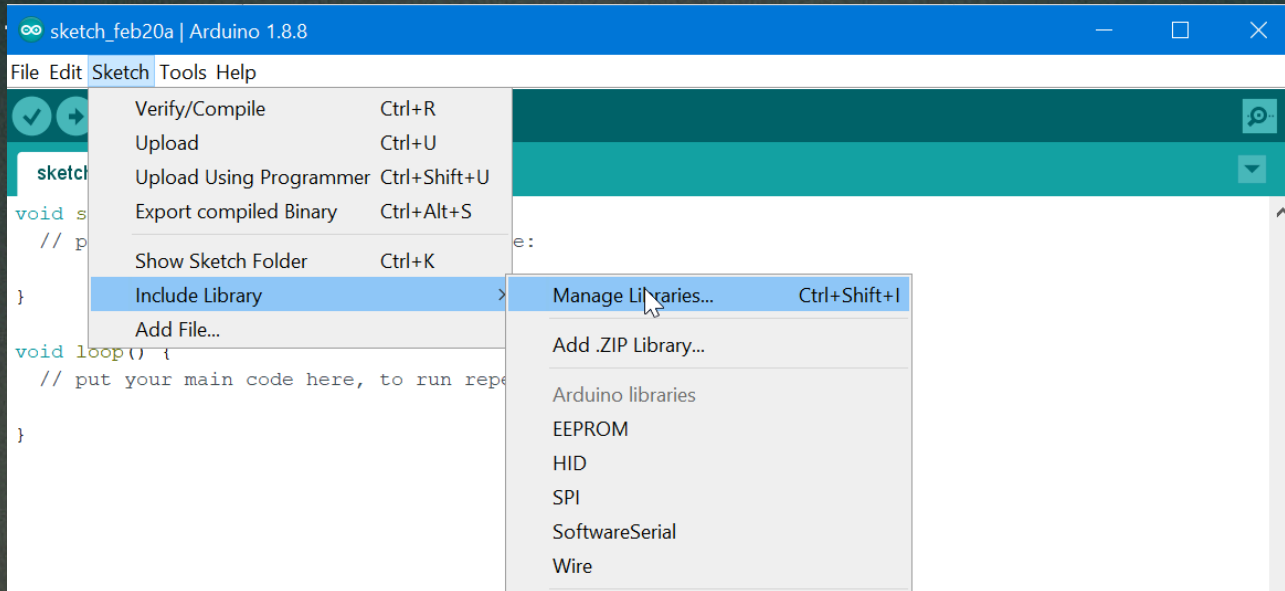
Install version 1.0.1



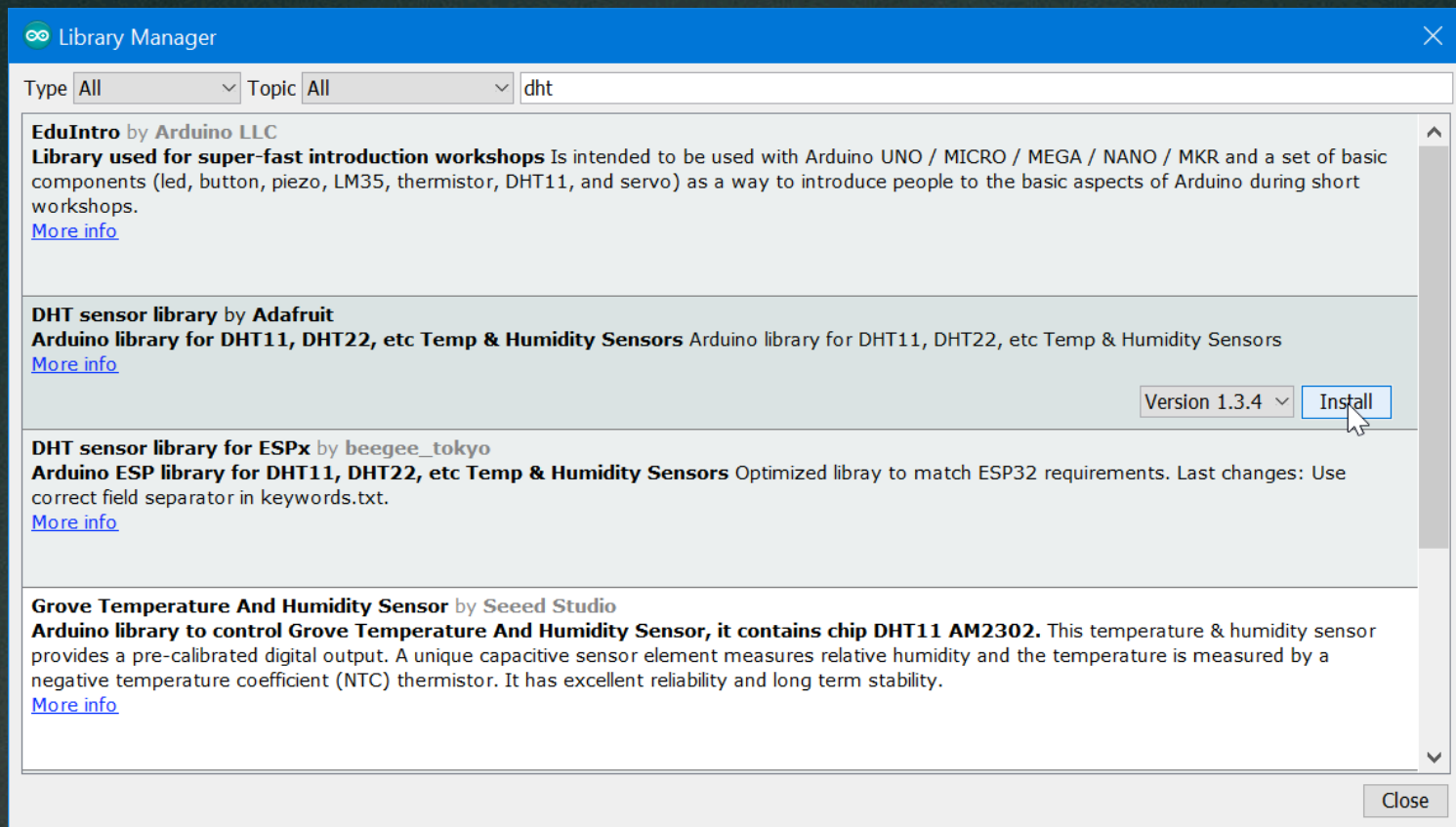
Libraries

To read data from the DHT11 sensor we will use the libraries published by Adafruit

- Open



Adafruit DHT Sensor Library



The screenshot shows the Arduino IDE Library Manager window. The search filter is set to 'dht'. Four libraries are listed. The second library, 'DHT sensor library' by Adafruit, is highlighted, and its 'Install' button is being clicked by a mouse cursor. The 'Version 1.3.4' dropdown is also visible next to the 'Install' button.

Library Manager

Type Topic

EduIntro by **Arduino LLC**
Library used for super-fast introduction workshops Is intended to be used with Arduino UNO / MICRO / MEGA / NANO / MKR and a set of basic components (led, button, piezo, LM35, thermistor, DHT11, and servo) as a way to introduce people to the basic aspects of Arduino during short workshops.
[More info](#)

DHT sensor library by **Adafruit**
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors
[More info](#) Version 1.3.4

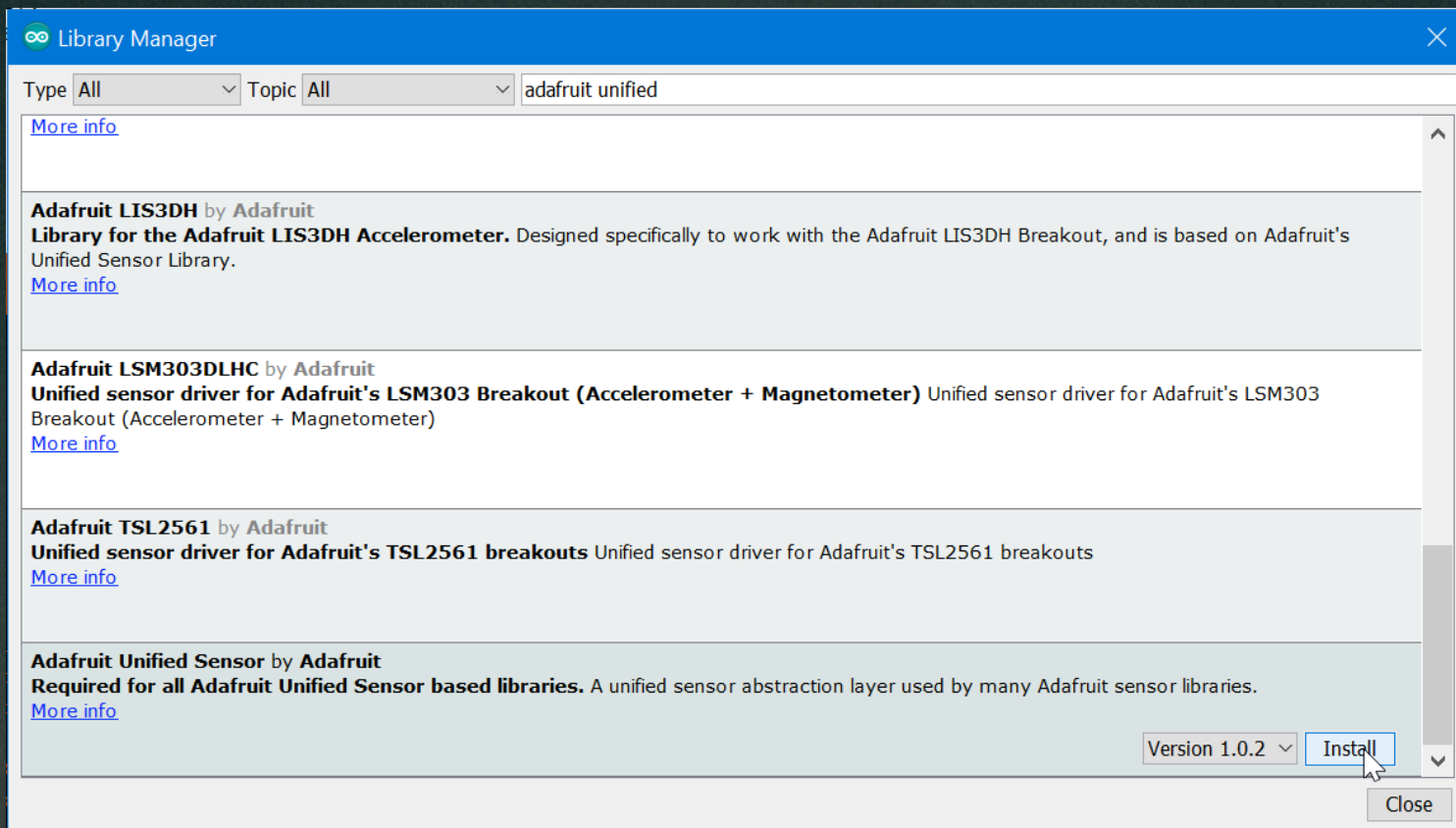
DHT sensor library for ESPx by **beegee_tokyo**
Arduino ESP library for DHT11, DHT22, etc Temp & Humidity Sensors Optimized library to match ESP32 requirements. Last changes: Use correct field separator in keywords.txt.
[More info](#)

Grove Temperature And Humidity Sensor by **Seeed Studio**
Arduino library to control Grove Temperature And Humidity Sensor, it contains chip DHT11 AM2302. This temperature & humidity sensor provides a pre-calibrated digital output. A unique capacitive sensor element measures relative humidity and the temperature is measured by a negative temperature coefficient (NTC) thermistor. It has excellent reliability and long term stability.
[More info](#)

Close



Adafruit Unified Sensor Library



The screenshot shows the Arduino IDE Library Manager window. The search bar contains 'adafruit unified'. The results list several libraries, with the 'Adafruit Unified Sensor' library at the bottom. A mouse cursor is hovering over the 'Install' button for this library.

Library Manager

Type: All Topic: All Search: adafruit unified

[More info](#)

Adafruit LIS3DH by **Adafruit**
Library for the Adafruit LIS3DH Accelerometer. Designed specifically to work with the Adafruit LIS3DH Breakout, and is based on Adafruit's Unified Sensor Library.
[More info](#)

Adafruit LSM303DLHC by **Adafruit**
Unified sensor driver for Adafruit's LSM303 Breakout (Accelerometer + Magnetometer) Unified sensor driver for Adafruit's LSM303 Breakout (Accelerometer + Magnetometer)
[More info](#)

Adafruit TSL2561 by **Adafruit**
Unified sensor driver for Adafruit's TSL2561 breakouts Unified sensor driver for Adafruit's TSL2561 breakouts
[More info](#)

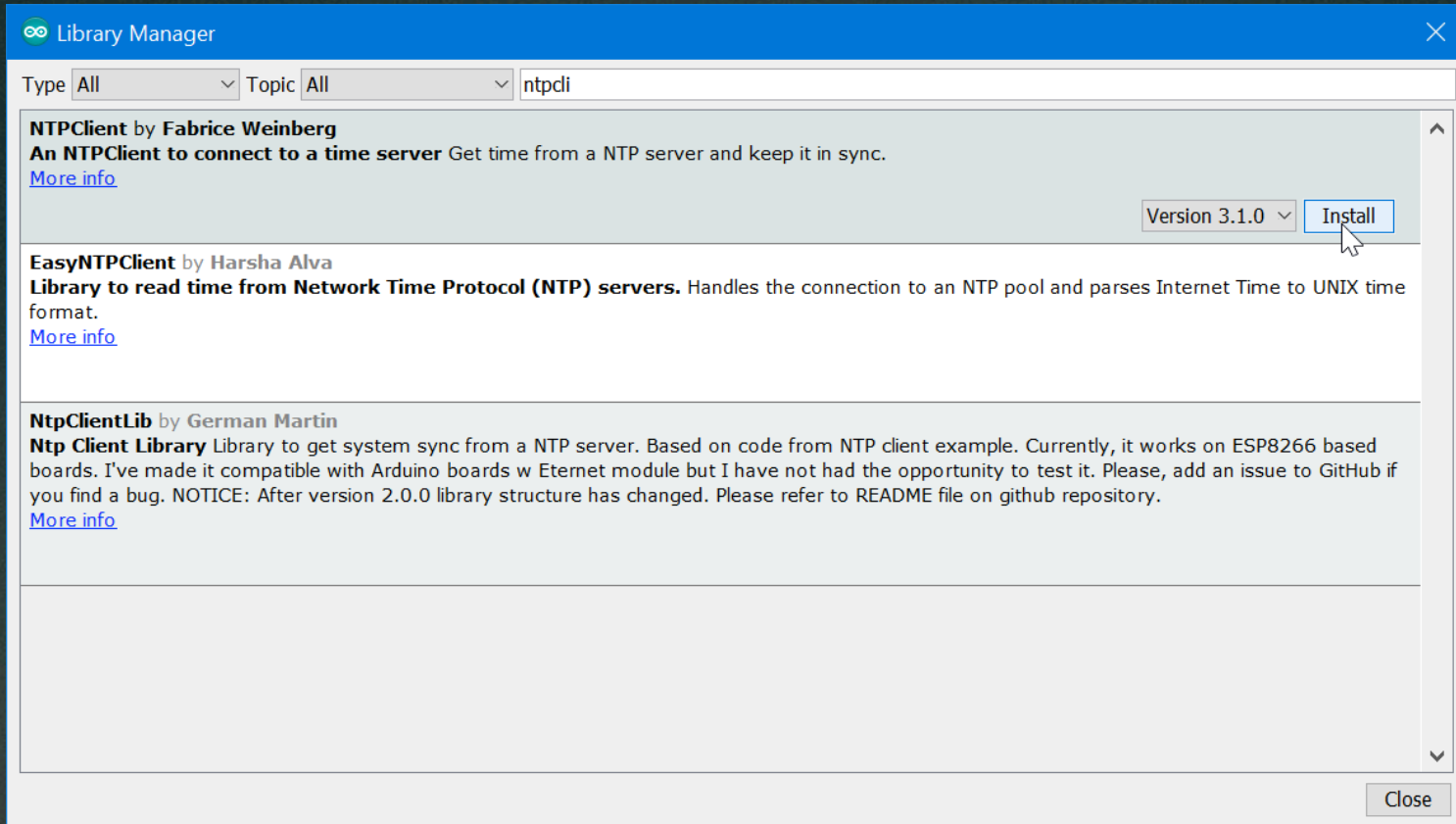
Adafruit Unified Sensor by **Adafruit**
Required for all Adafruit Unified Sensor based libraries. A unified sensor abstraction layer used by many Adafruit sensor libraries.
[More info](#)

Version 1.0.2 **Install**

Close



NTPClient Library



The screenshot shows the Arduino IDE Library Manager window. The search bar contains 'ntpcli'. Three libraries are listed:

- NTPClient** by **Fabrice Weinberg**
An NTPClient to connect to a time server Get time from a NTP server and keep it in sync.
[More info](#)
Version 3.1.0 **Install**
- EasyNTPClient** by **Harsha Alva**
Library to read time from Network Time Protocol (NTP) servers. Handles the connection to an NTP pool and parses Internet Time to UNIX time format.
[More info](#)
- NtpClientLib** by **German Martin**
Ntp Client Library Library to get system sync from a NTP server. Based on code from NTP client example. Currently, it works on ESP8266 based boards. I've made it compatible with Arduino boards w Ethernet module but I have not had the opportunity to test it. Please, add an issue to GitHub if you find a bug. NOTICE: After version 2.0.0 library structure has changed. Please refer to README file on github repository.
[More info](#)

The 'Install' button for the first library is highlighted with a mouse cursor. A 'Close' button is at the bottom right.

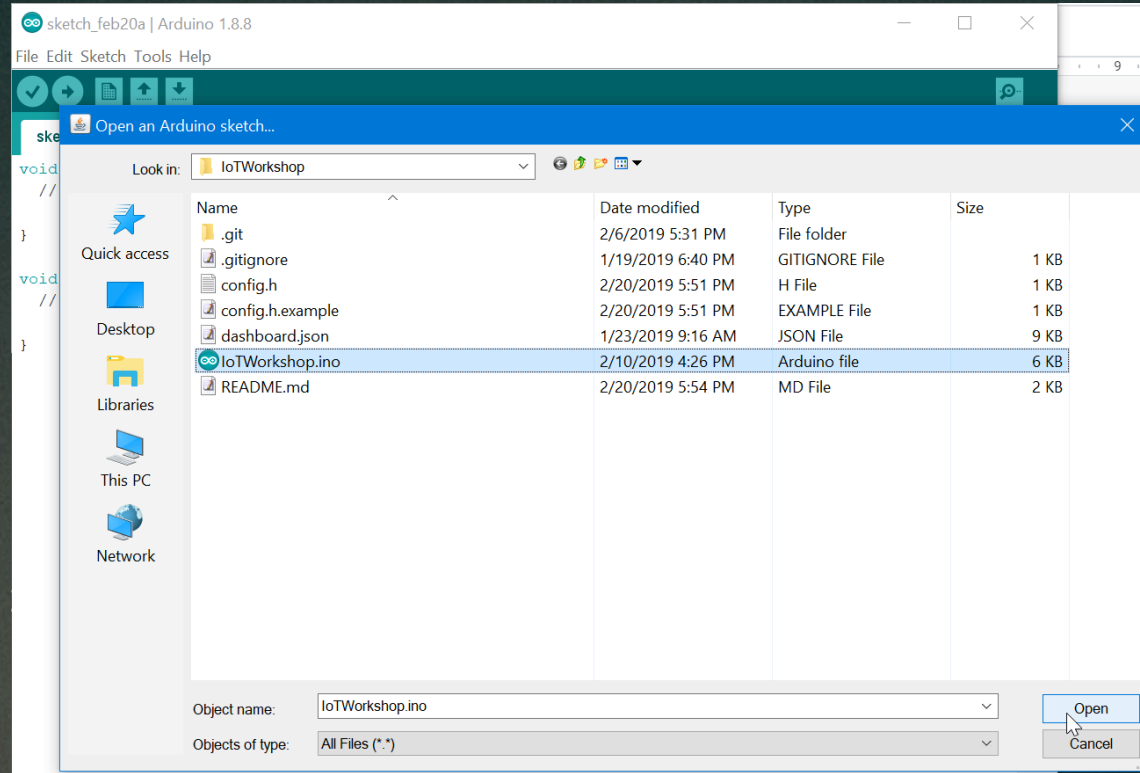


Download the project

- We need to download the project files into your arduino IDE sketchbook folder
- You can do that via git by going to your sketchbook folder and running:
`git clone https://github.com/DanCech/IoTWorkshop.git`, or
- You can download the .zip from [the link in the project readme](#) and extract it into your sketchbook folder
- After downloading, copy `config.h.example` to `config.h`, this file will hold the configuration for the device.



Open the IoTWorkshop project in Arduino IDE



Let's walk through the sketch

- Line 1-7: Include required libraries & config.h
- Line 10-11: Set up NTP client
- Line 14: Set up DHT Sensor
- Line 17: Set up HTTP client for Hosted Metrics
- Line 20: Set up UDP client for carbon protocol
- Line 25-42: `setupWiFi` function to connect to WiFi network
- Line 47-60: `formatTime` function to format a timestamp
- Line 65-91: `submitHostedGraphite` function to send to hosted graphite
- Line 96-110: `submitCarbon` function to send to hosted prom.
- Line 115-127: `setup` function to initialize at startup
- Line 132-192: `loop` function, where the magic happens



void setup()

- This function is called when the ESP32 starts
- It starts the Serial debug connection
- Calls setupWiFi() to connect to the WiFi network
- Initializes the NTP (Network Time Protocol) client
- Initializes the DHT sensor



```
void loop()
```

The core of the system, this function is called in an endless loop, it:

- checks the WiFi connection and reconnects if required
- updates the time via NTP and gets the current timestamp
- reads the current temperature and humidity from the DHT11 sensor
- calculates the heat index
- outputs the readings via Serial
- sends the stats to Hosted Graphite
- sends the stats to carbon relay
- sleeps for 30 seconds



Create TSDB & Grafana instance

Go to

<https://grafana.com/loki#get>

and follow the instructions to get set up.

(If you already have a Hosted Metrics instance, feel free to use that)



config.h

This file contains the configuration for the project

- WIFI_SSID GrafanaCon WiFi, Sponsored By...
- WIFI_PASSWORD packetdotcom
- TZ_OFFSET timezone offset to use for formatted dates
- ID an identifier for this sensor
- INTERVAL reporting interval
- DHT_PIN ESP32 pin the DHT11 signal is connected to
- DHT_TYPE DHT11 or DHT22 (both are supported by the lib)



config.h

- HM_API_KEY Grafana.com API key with MetricPublisher role
- HM_GRAPHITE_HOST graphite-us-central1.grafana.net
- HM_GRAPHITE_INSTANCE Hosted Graphite Instance ID
- CARBON_HOST IP Address of carbon server
- CARBON_PORT carbon server port
- HM_ROOT_CA The root CA cert to use for SSL validation



Setting the board type

This tells the Arduino IDE which profile and base libraries to use when compiling the firmware image, and how to flash it to the board

- Open the "Tools" menu
- In the Board submenu, select "ESP32 Pico Kit"
- Plug in the USB cable
- In the Port submenu, select the new COM port



The serial monitor

This allows us to see the debug output from the board, including the bootloader

- Select "Serial Monitor" from the Tools menu
- Set the port speed to 115200



Building and uploading the firmware

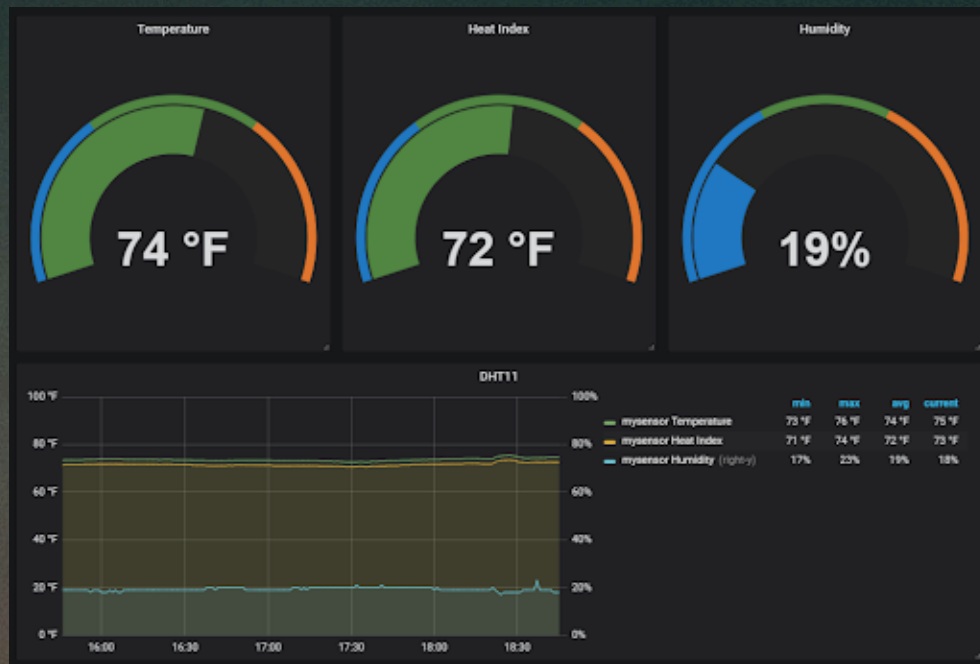
Use the Upload button in the UI to build and upload the firmware to the ESP32

- The serial monitor will be blank during the upload
- Progress will be displayed in the screen at the bottom of the IDE
- Once the upload is complete, the Serial Monitor will show:
 - Start up & connect to WiFi
 - Readings from the sensor
 - Results of submitting metrics



The dashboard

- 3 Singlestat panels
 - Temperature
 - Heat Index
 - Humidity
- 1 Graph panel



<https://grafana.com/dashboards/9848>



Thanks!