IoT & Observability

Monitoring our world with Grafana

Dan Cech

Grafana Labs

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!

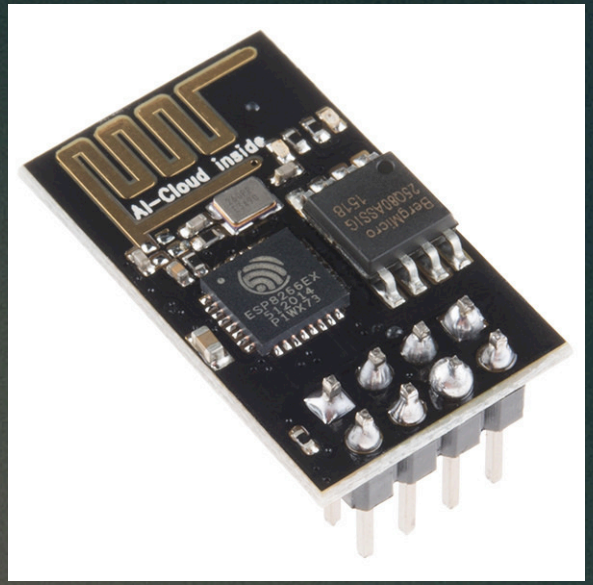
In this workshop we'll explore how we can use Grafana together with the ESP32 platform to monitor our surroundings and visualize data over time.

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.



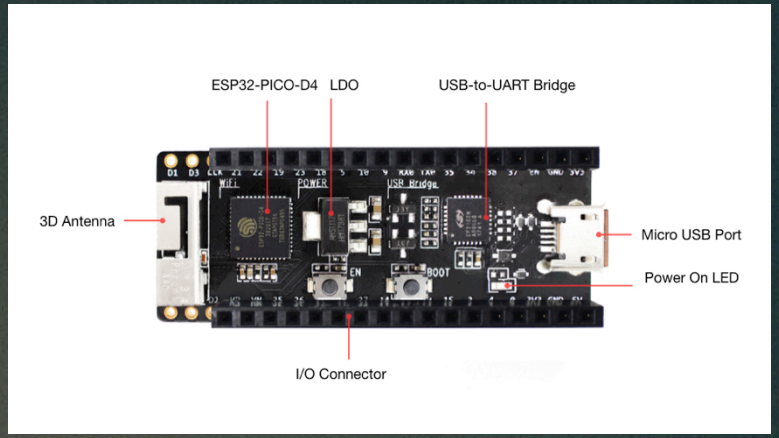
https://commons.wikimedia.org/wiki/File:ESP-01.jpg

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

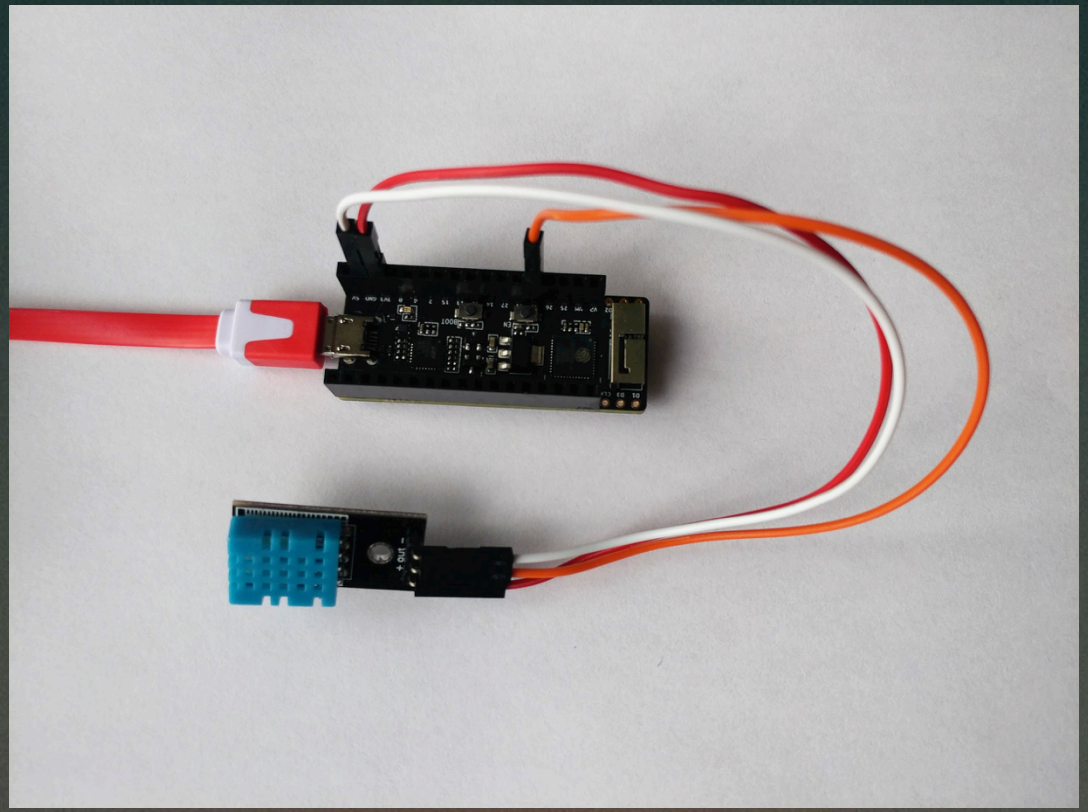
Power

- 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

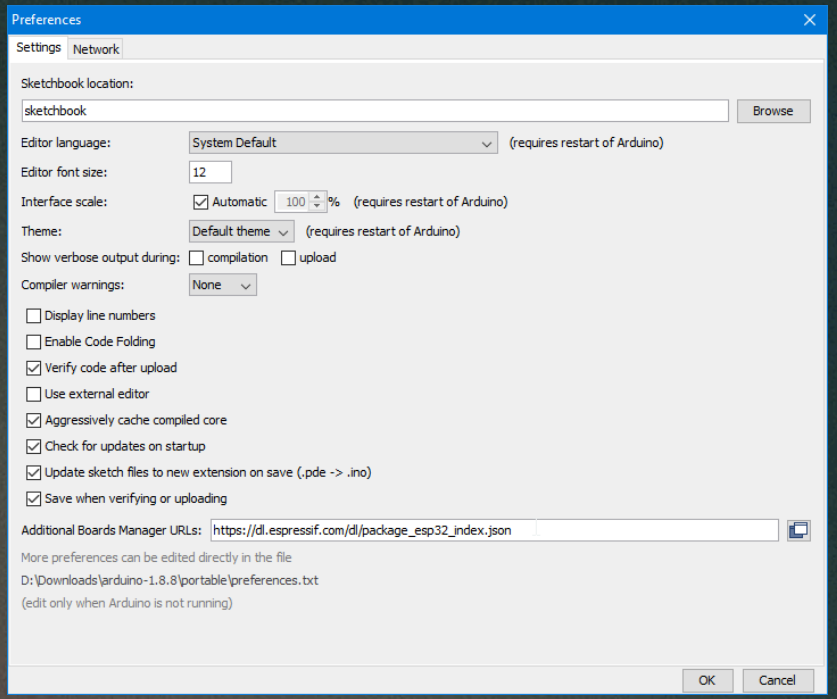
- arduino.cc
- Download the 1.8.8 IDE for your OS
- Install it

The 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

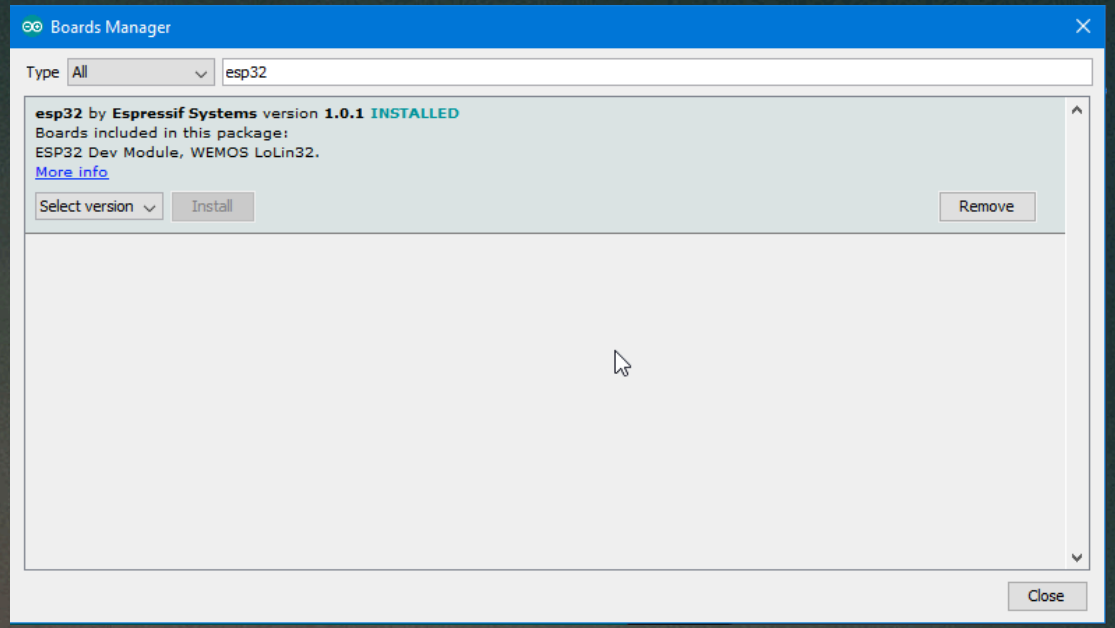
Tools -> Boards: *

-> Board manager

Wait for it to update the list

Search for ESP32

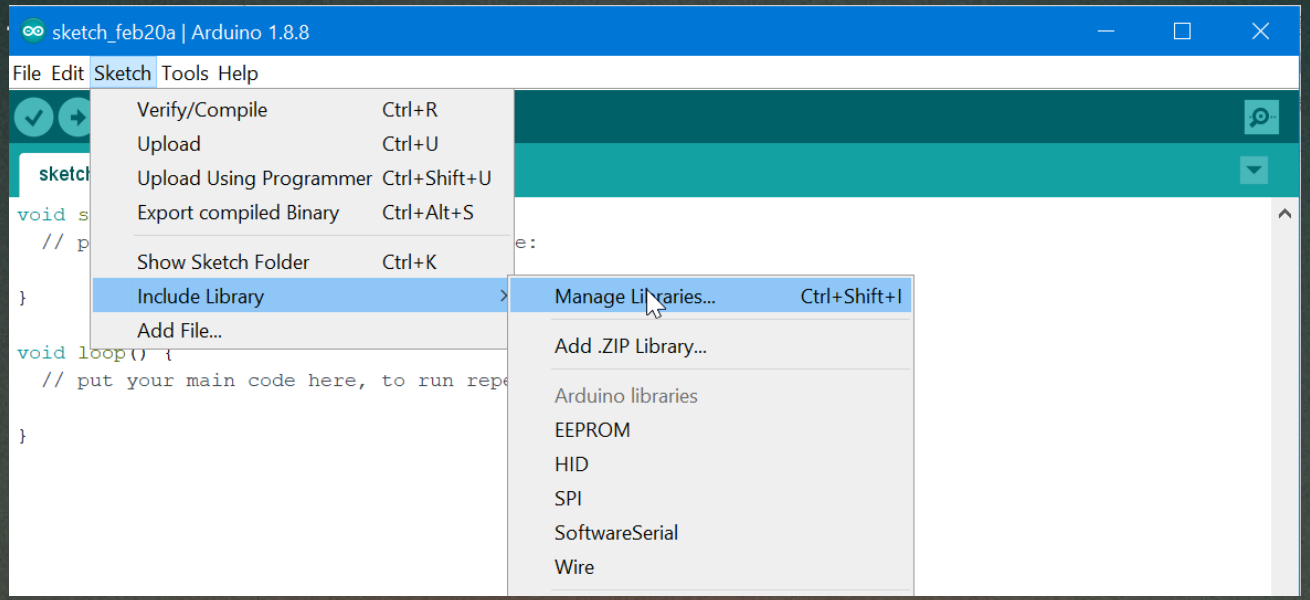
Install version 1.0.1



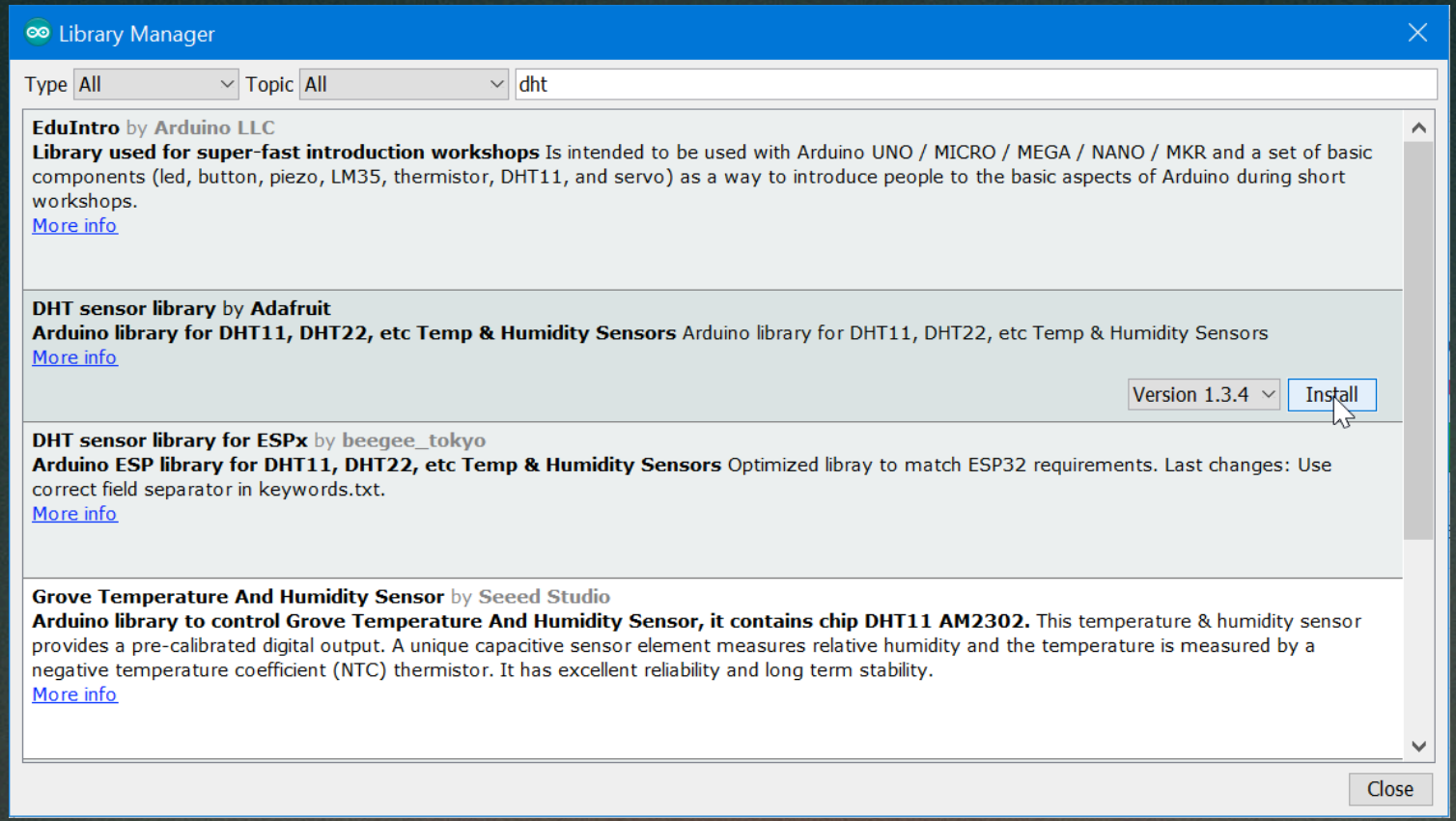
Libraries

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

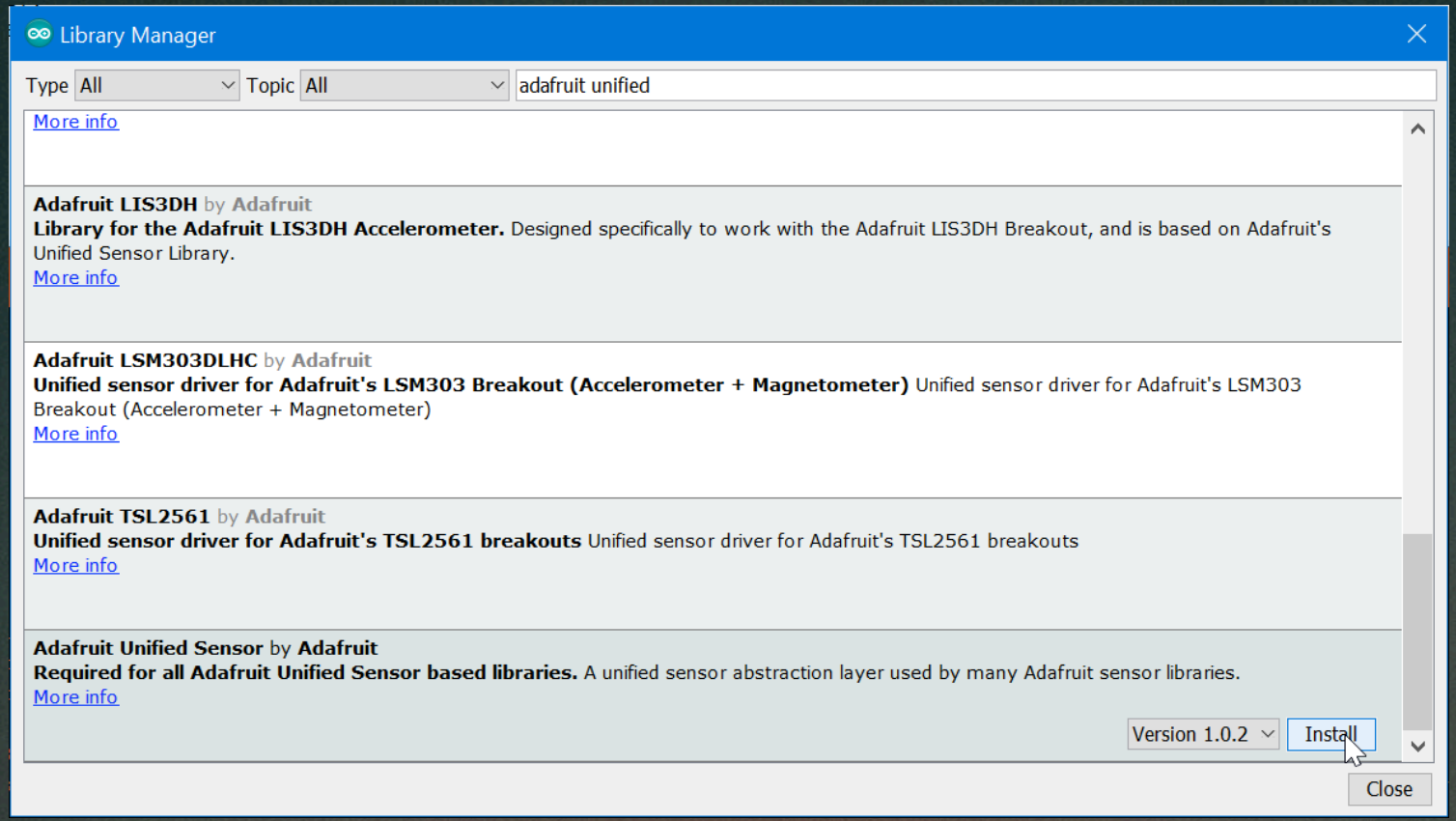
- Open



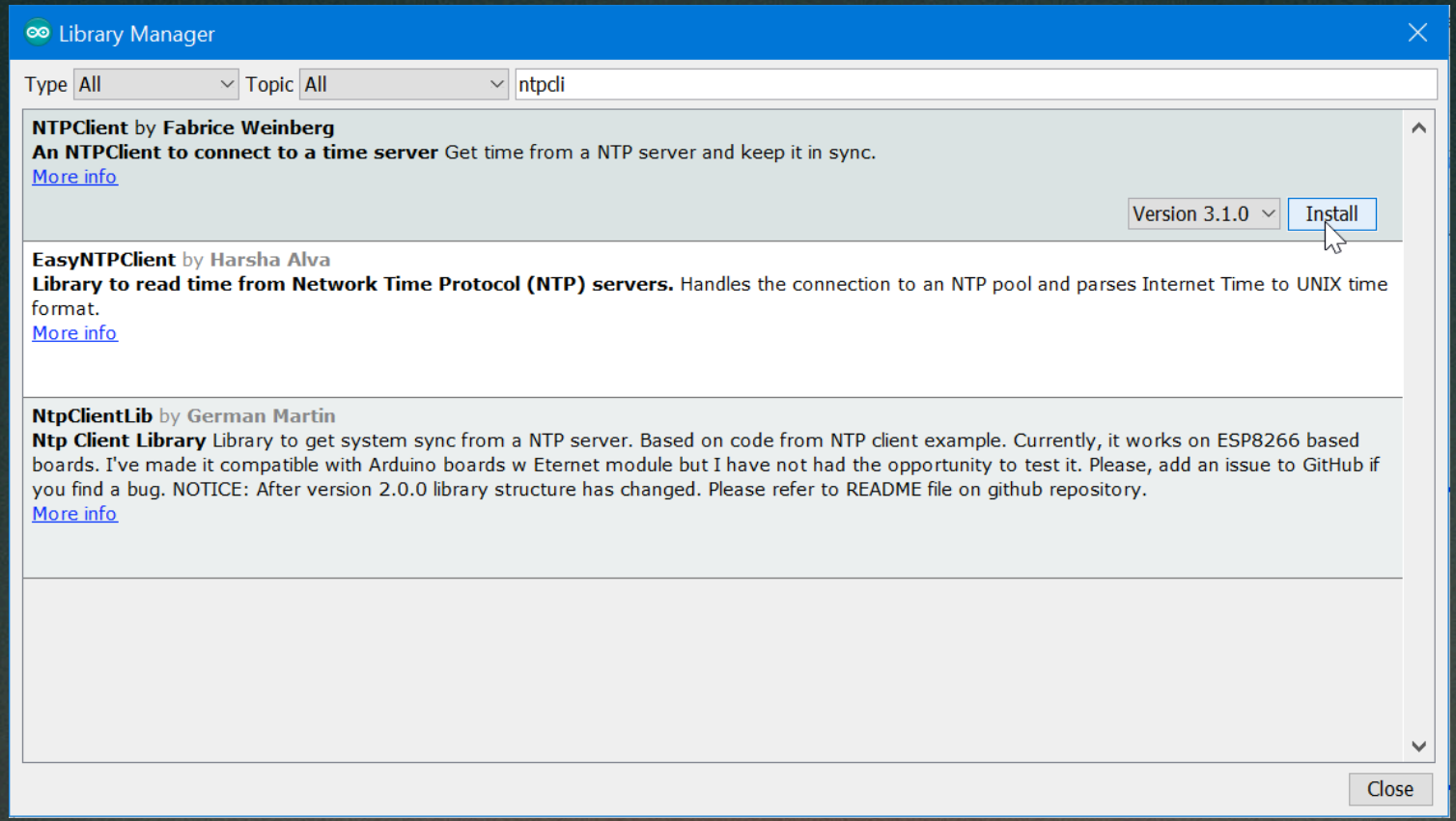
Adafruit DHT Sensor Library



Adafruit Unified Sensor Library



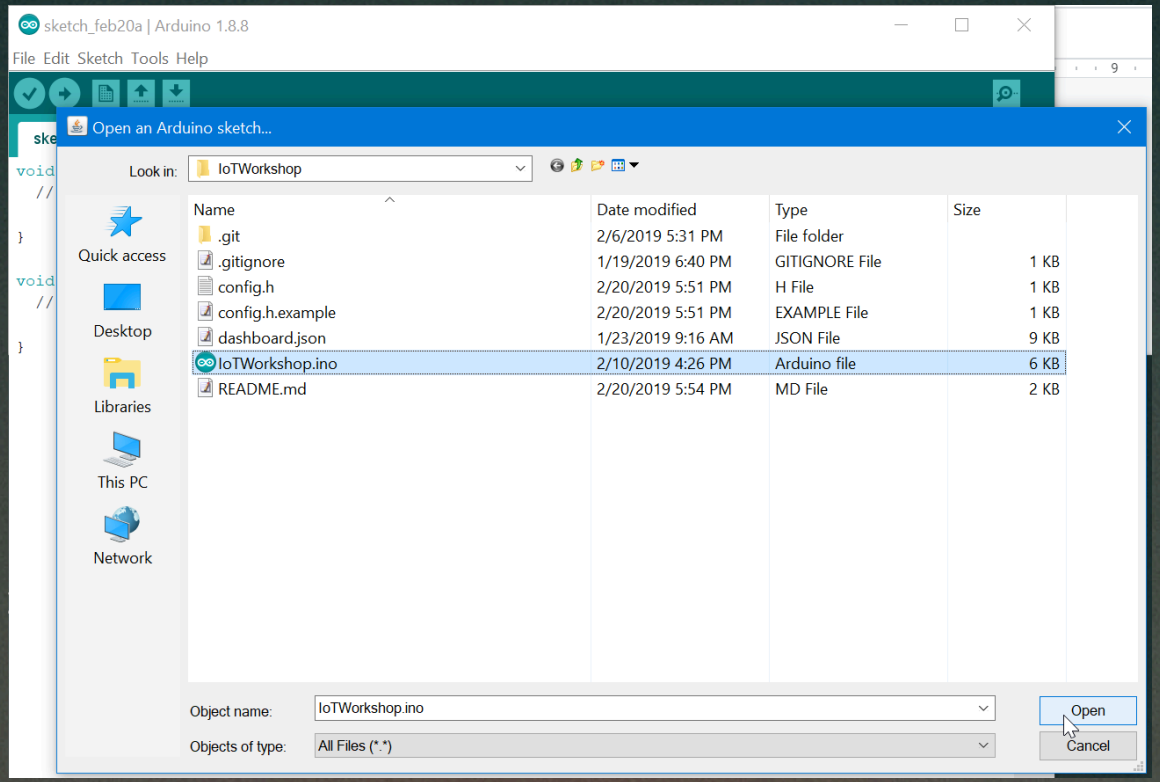
NTPClient Library



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_SSD` 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

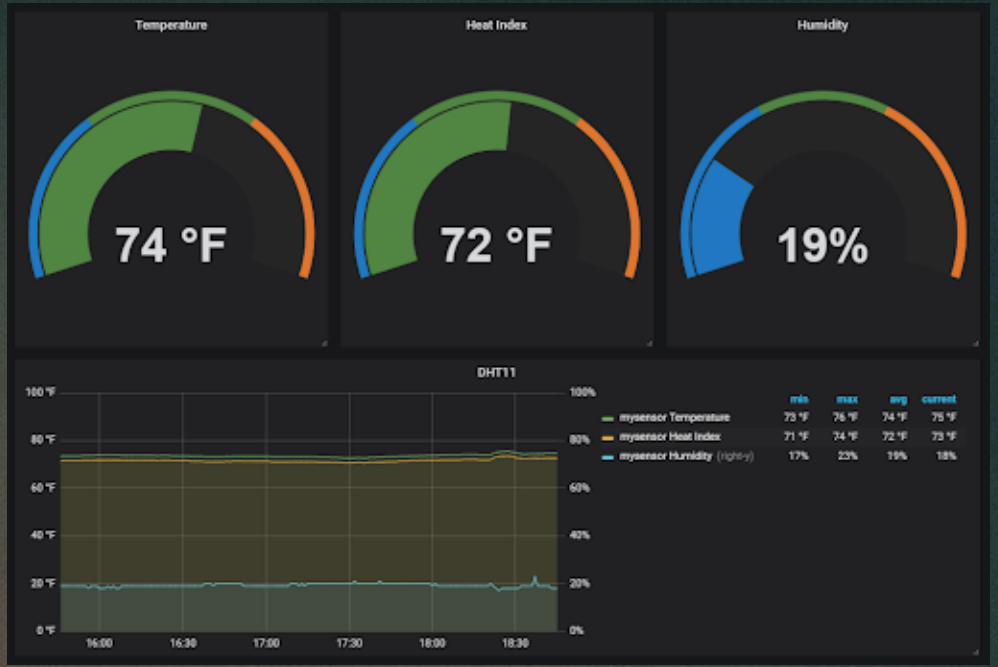
Building and uploading the firmware

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!