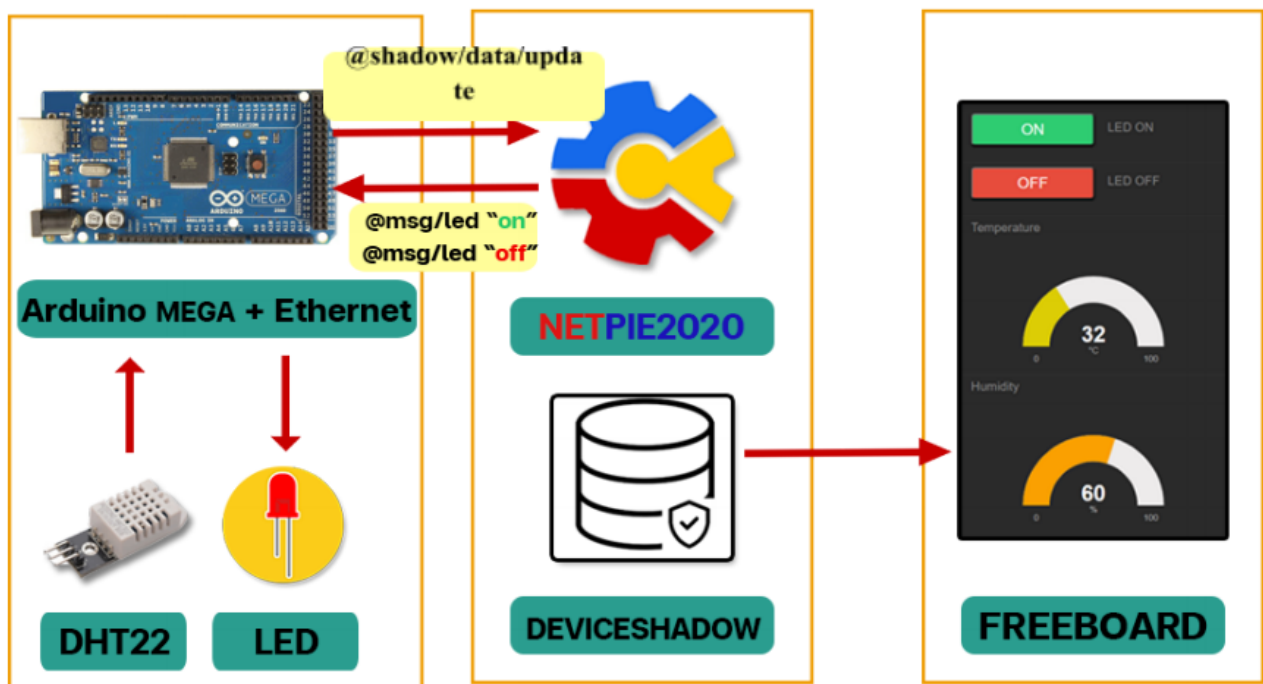


Example of Connecting and Using Arduino Mega + Ethernet with NETPIE2020

Project Description

This example demonstrates how to transmit temperature and humidity readings from DHT22 to NETPIE2020, to present them on Freeboard and control the on/off action of the LED connected to the Arduino Mega + Ethernet.



Working model

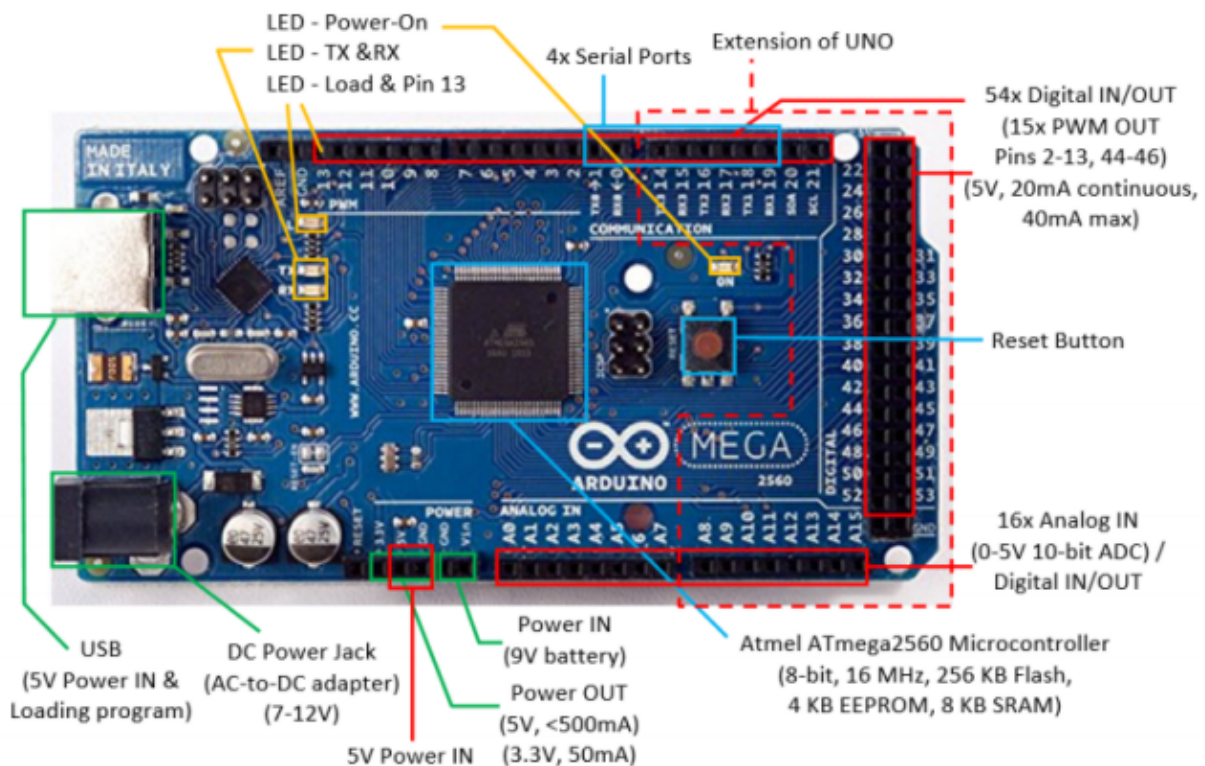
Require Basic Knowledge of

1. Using NETPIE2020
2. Using NETPIE Freeboard

Equipment Used

1.Arduino Mega + Ethernet

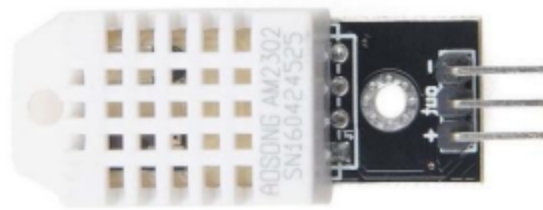
Arduino Mega is the largest board among the Arduino boards family that uses an Atmega2560 microcontroller. This microcontroller is different from the Atmega328 that is used with the Arduino Uno. The Mega has 54 Digital Input/Output pins (15 PWM pins), 16 Analog Input pins, an ethernet shield used to plug in the ethernet cable for internet connection, and sd card slot.



Features of Arduino Mega

2.DHT22 Temperature and Humidity Sensor

The DHT22 is a sensor module that measures temperature and humidity. It requires a voltage of 3-5V and can measure temperature ranging from -40 - 80°C (tolerance ± 0.5 °C), air humidity ranging from 0 - 100% (tolerance 2 - 5%).



DHT22 sensor

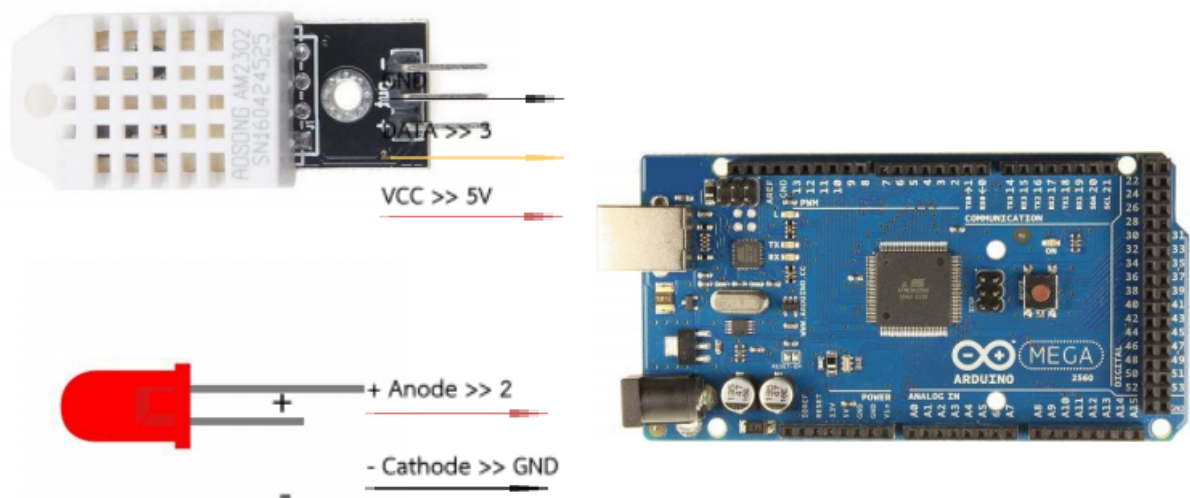
Pin Description

Pin1: Power +Ve (5V Max wrt. GND)

Pin2: Digital I/O interface connected to a microcontroller

Pin3: Power ground or Power -Ve

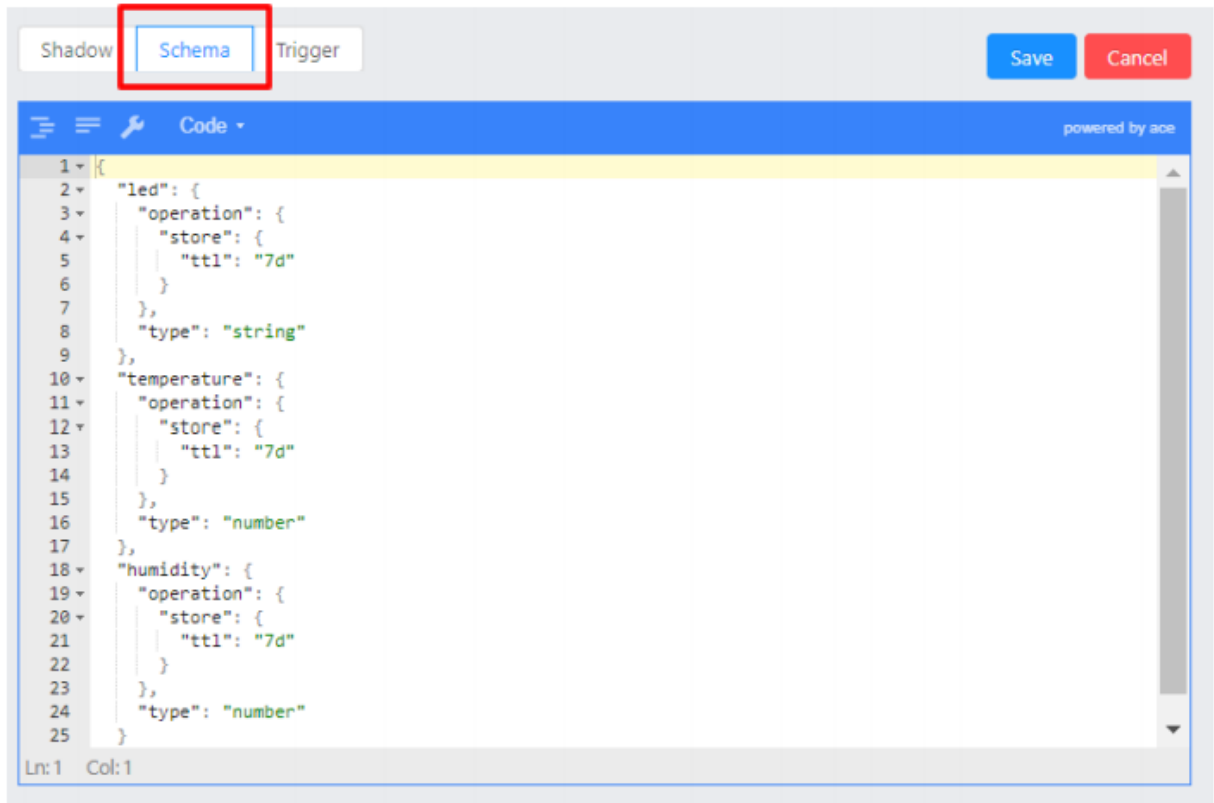
Arduino Mega Circuit Connection



Circuit connection of DHT22 and LED to Arduino Mega + Ethernet

Defining Device Schema

The first part is to define the device schema, which is the data structure for the devices generating data. The server checks the data structure defined in the device schema and performs the required actions before storing the data in the Timeseries database. These actions include converting the data units and data validation.



Description of Schema

Stores the first variable 'led' of type string in the Timeseries database with a retention period of 7 days.

Stores the second variable 'temperature' of type number in the Timeseries database with a retention period of 7 days.

Stores the third variable 'humidity' of type number in the Timeseries database with a retention period of 7 days.

Description of Arduino Program

First, the required libraries for executing the program should be imported, followed by defining the program parameters like LED PIN and DHT PIN. Next, configure the network parameters like server, port, client ID, username, and password for connecting to the NETPIE2020.

```
#include <WiFi.h>
#include <Ethernet.h>
#include <PubSubClient.h>
#include <Wire.h>
#include <DHT.h>

#define LED 2
#define DHTTYPE DHT22
#define DHTPIN 3

byte My_MAC_address[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED }; // 000000 MAC address 000000Ethernet
const char* mqtt_server = "broker.netpie.io";
const int mqtt_port = 1883;
const char* mqtt_Client = "b49f7b71-bc60-4846-ab8d-f66a1ac0548f"; //ArduinoMEGA
const char* mqtt_username = "p79ckbXTtoEHpxA49qB1LUdpXc9xqjrY3";
const char* mqtt_password = "vV9_or(anY0Pn37dOzk-wY_nek85DjJ6";
```

The second part is to run the MQTT commands.

```
EthernetClient Etherclient;
PubSubClient client(Etherclient);
DHT dht(DHTPIN, DHTTYPE);
```

The MQTT connect function is used to connect to the MQTT server. If the connection is successful, it will display the message saying 'connected'. But, if the connection is unsuccessful, 'failed' message is displayed and will try to reconnect automatically.

```

void reconnect() {
    while (!client.connected()) {

        if (client.connect(mqtt_Client, mqtt_username, mqtt_password)) {
            Serial.println("connected");
            client.subscribe("@msg/led"); // subscribe Topic ที่ Freeboard ส่งมา
        }

        else {
            Serial.print("failed, rc=");
            Serial.print(client.state());
            Serial.println("try again in 5 seconds");
            delay(5000);
        }
    }
}

```

The callback function is used to check messages received from the Freeboard, which first checks the topic name, which is '@msg/led', then checks the message payload. If the received message contains 'on', led is turned on and if the message contains 'off', led is turned off. The status of the led is published in the JSON format to update the shadow.

```

void callback(char* topic, byte* payload, unsigned int length) {
    Serial.print("Message arrived [");
    Serial.print(topic);           // [msg/led]
    Serial.print("] ");

    String message;
    for (int i = 0; i < length; i++) {
        message = message + (char)payload[i];
    }
    Serial.println(message); // on , off

    if(String(topic) == "@msg/led") {
        if (message == "on") {
            digitalWrite(LED,1);
            client.publish("@shadow/data/update", "{\"data\" : {\"led\" : \"on\"}}");
            Serial.println("LED ON");
        }
        else if (message == "off") {
            digitalWrite(LED,0);
            client.publish("@shadow/data/update", "{\"data\" : {\"led\" : \"off\"}}");
            Serial.println("LED OFF");
        }
    }
}

```

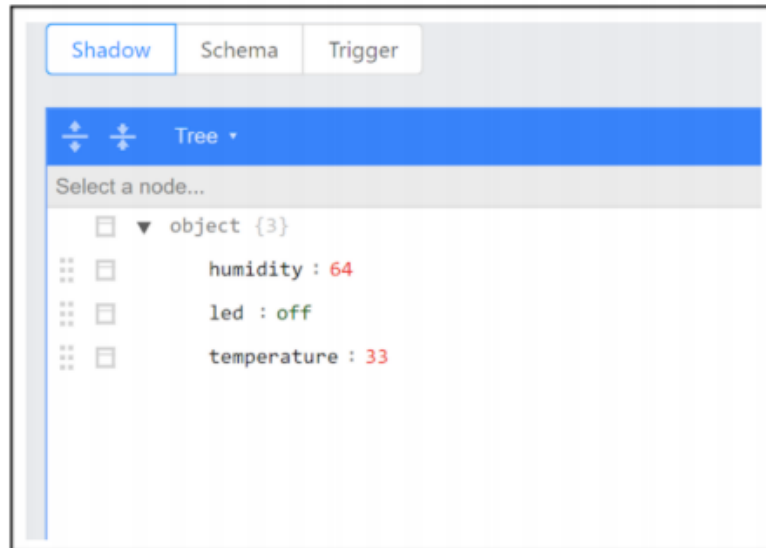
The setup function attempts to use the internet by running the command Ethernet.begin and initializes the dht by running the dht.begin command.

```
void setup() {  
  Serial.begin(115200);  
  pinMode(LED, OUTPUT);  
  digitalWrite(LED, 0); // เริ่มต้นให้ LED ดับ  
  while(Ethernet.begin(My_MAC_address) != 1) // start the Ethernet connection:  
  {  
    Serial.print(".");  
  }  
  Serial.print("My IP :");  
  Serial.println(Ethernet.localIP());  
  
  client.setServer(mqtt_server, mqtt_port);  
  client.setCallback(callback);  
  dht.begin(); // start DHT22  
  
}
```

The last part is to define the variables Temperature and Humidity and publish the values on Topic: shadow/data/update for every 2 seconds.

```
void loop() {  
  if (!client.connected()) {  
    reconnect();  
  }  
  client.loop();  
  int humidity = dht.readHumidity();  
  int temperature = dht.readTemperature();
```

Messages sent to the NETPIE2020 and the values are saved in the shadow.



Creating a Device on NETPIE2020

1.Start by selecting the menu Device List > Create and name the device as shown in the figure below.

Create

X

* Name:

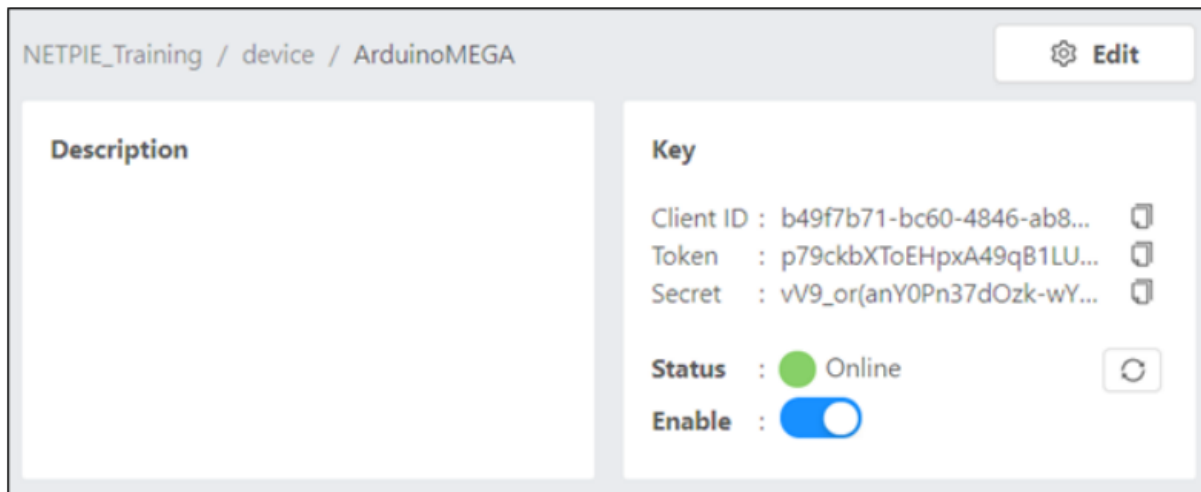
Description:

Tag:

+ New Tag

Cancel

Create



NETPIE_Training / device / ArduinoMEGA Edit

Description

Key

Client ID : b49f7b71-bc60-4846-ab8...

Token : p79ckbXToEHpxA49qB1LU...

Secret : vV9_or(anY0Pn37dOzk-wY...

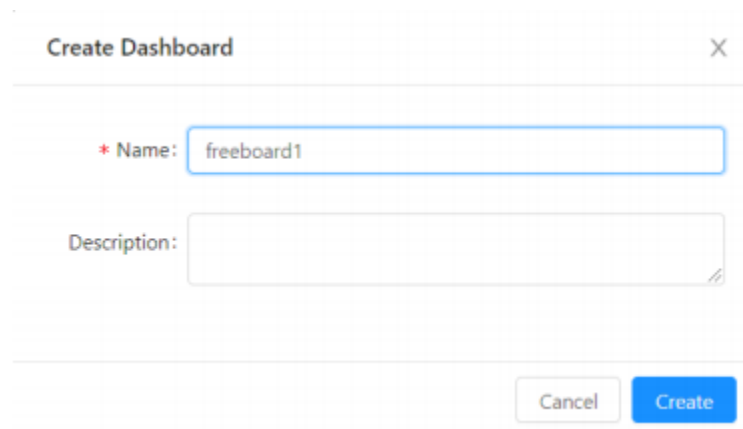
Status : Online

Enable : ☒

Creating device on NETPIE2020

Creating a Freeboard for Monitoring Temperature, Humidity, and Controlling the LED

1.Start by selecting the menu Freebord > Create and name the Freeboard as shown in the figure below.

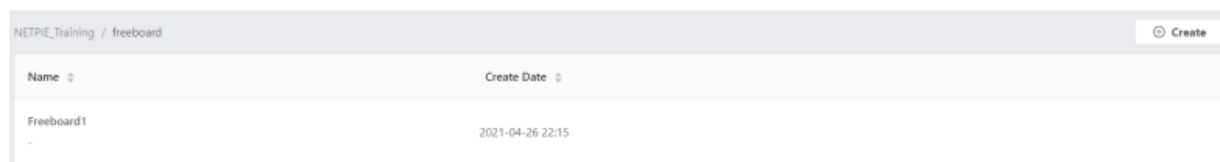


Create Dashboard X

* Name: freeboard1

Description:

Cancel Create



NETPIE_Training / freeboard Create	
Name	Create Date
Freeboard1	2021-04-26 22:15

2.Fill in the details, which include Name, Device ID, and Device Token that can be found in the DeviceList named ArduinoMega.

The image shows a 'DATASOURCE' configuration window with the following fields and values:

- NAME:** ArduinoMEGA_D4
- DEVICE ID:** b49f7b71-bc60-4846-ab8d-f86a1ac0548f
Client ID for Device ArduinoMega
- DEVICE TOKEN:** p79ckbXToEHpxA49qB1LUmpXc9xqjrY3
Token for Device ArduinoMega
- SUBSCRIBED TOPICS:**
Topic Arduino Subscribe
- FEED:** YES ☒
- SINCE:** 6
Display data points since ... ago
Hour
- DOWN SAMPLING:** 1
Resolution of the data points
Minute

At the bottom right, there are 'SAVE' and 'CANCEL' buttons.

(NOTE: If the user enables the Feed, Freeboard keeps on calling the API. So, it is better to turn off the Feed if the user is not intended to use it.)

3.Create a widget of type Gauge for displaying the Temperature and Humidity values by clicking on 'ADD PANE' and fill in the details as shown in the figure below.

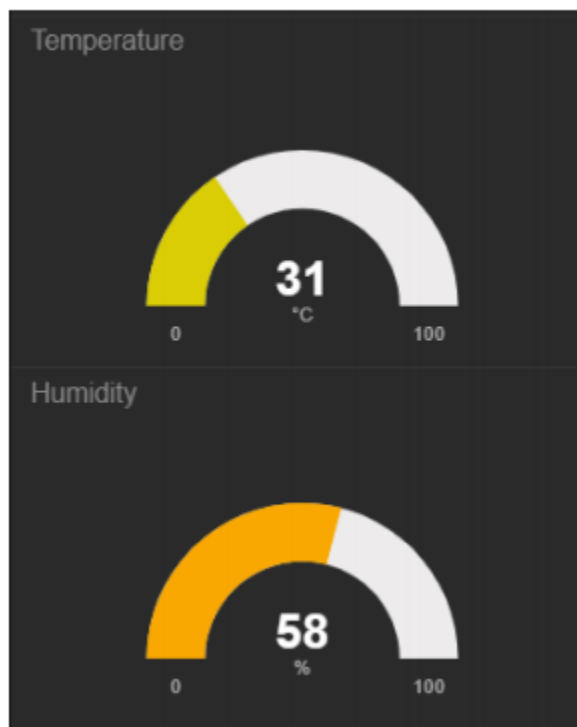
WIDGET

TYPE	Gauge	
TITLE	Temperature	
VALUE	<code>datasources["ArduinoMEGA_D4"]["shadow"]["temperature"]</code>	+ DATASOURCE ✕ JS EDITOR
UNITS	°C	
MINIMUM	0	
MAXIMUM	100	

[SAVE](#) [CANCEL](#)

WIDGET

TYPE	Gauge	
TITLE	Humidity	
VALUE	<code>datasources["ArduinoMEGA_D4"]["shadow"]["humidity"]</code>	+ DATASOURCE ✕ JS EDITOR



Displaying Temperature and Humidity values stored in Shadow

4. Create ON/OFF buttons and set the ONCLICK ACTION on the ON button to `netpie["DataSourceName"].publish("@msg/led","on")` and `netpie["DataSourceName"].publish("@msg/led","off")` for the OFF button.

WIDGET

A simple button widget that can perform Javascript action.

TYPE: **Button**

BUTTON CAPTION: **ON**

LABEL TEXT: **LED ON**

BUTTON COLOR: **Green**

ONCLICK ACTION: `netpie["ArduinoMEGA_D4"].publish("@msg/led", "on")` [+ DATASOURCE](#) [✕ JS EDITOR](#)

Add some Javascript here.

ONCREATED ACTION:
JS code to run after a button is created

[SAVE](#) [CANCEL](#)

Settings for ON button

WIDGET

A simple button widget that can perform Javascript action.

TYPE: **Button**

BUTTON CAPTION: **OFF**

LABEL TEXT: **LED OFF**

BUTTON COLOR: **Red**

ONCLICK ACTION: `netpie["ArduinoMEGA_D4"].publish("@msg/led", "off")` [+ DATASOURCE](#) [✕ JS EDITOR](#)

Add some Javascript here.

ONCREATED ACTION:
JS code to run after a button is created

[SAVE](#) [CANCEL](#)

Settings for OFF button



ON	LED ON
OFF	LED OFF