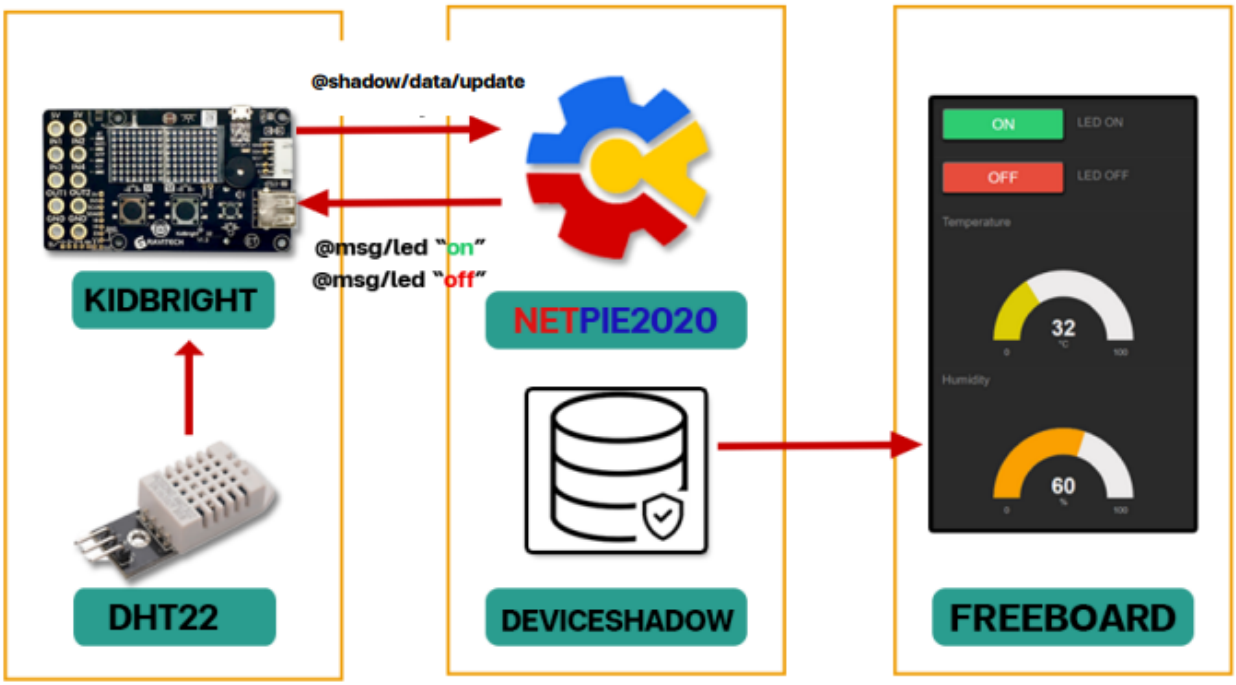
Example of Connecting and Using Kidbright with NETPIE2020

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 Kidbright.



Working model

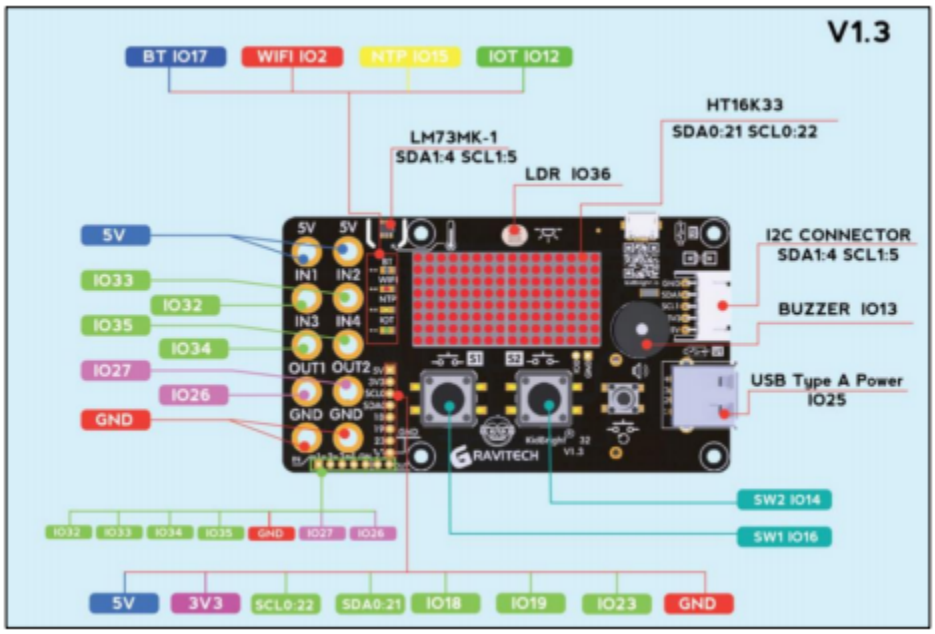
Require Basic Knowledge of

1.Using NETPIE2020
2.Using NETPIE Freeboard

Equipment Used

1.Kidbright

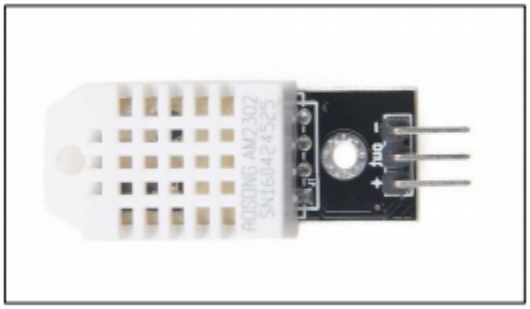
Kidbright is a small embedded board that can run a set of instructions. It can connect to external devices and expansion boards. It consists of a ESP32, LED display screen of size 16 X 8, and a detection sensor.



Features of Kidbright

2.DHT22 Temperature and Humidity Sensor

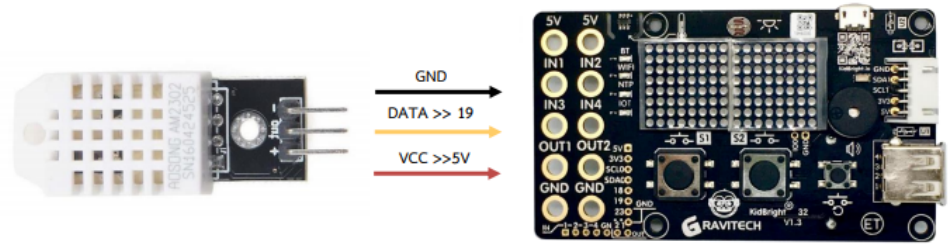
The DHT22 is a sensor module that measures temperature and humidity. It requires a voltage of 3-5V and can measure temperature ranging from -40 - 80°C (tolerance ±0.5 °C), air humidity ranging from 0 - 100% (tolerance 2 - 5%).



Pin Description

Pin1: Power +Ve (5V Max wrt. GND)
Pin2: Digital I/O interface connected to the microcontroller
Pin3: Power Ground or Power -Ve

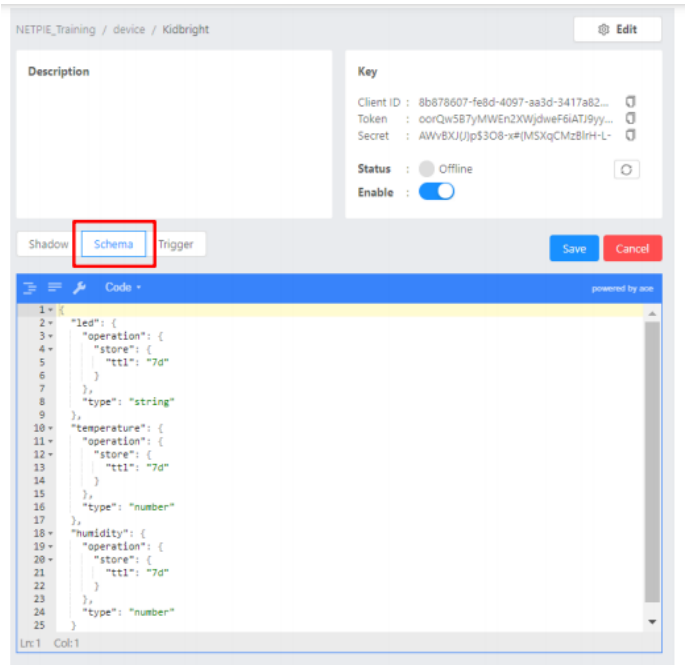
Kidbright Circuit Connection



Circuit connection between DHT22 and Kidbright

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 <PubSubClient.h>
#include <DHT.h>

#define LED 12
#define DHTTYPE DHT22
#define DHTPIN 19

const char* ssid = "pp";
const char* password = "12345687";
const char* mqtt_server = "broker.netpie.io";
const int mqtt_port = 1883;
const char* mqtt_Client = "8b878607-fe8d-4097-aa3d-3417a82abbcb";  //kidbright
const char* mqtt_username = "oorQw5B7yMWEn2XWjdweF6iATJ9yy3Sf";
const char* mqtt_password = "AWvBXJ(J)p$3O8-x#(MSXqCMzBlrH-L-";
```

The second part is to run the MQTT commands.

```
char msg[100];
long lastMsg = 0;

WiFiClient espClient;
PubSubClient client(espClient);
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);
  Serial.print("] ");
  String message;
  for (int i = 0; i < length; i++) {
    message = message + (char)payload[i];
  }
  Serial.println(message);

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

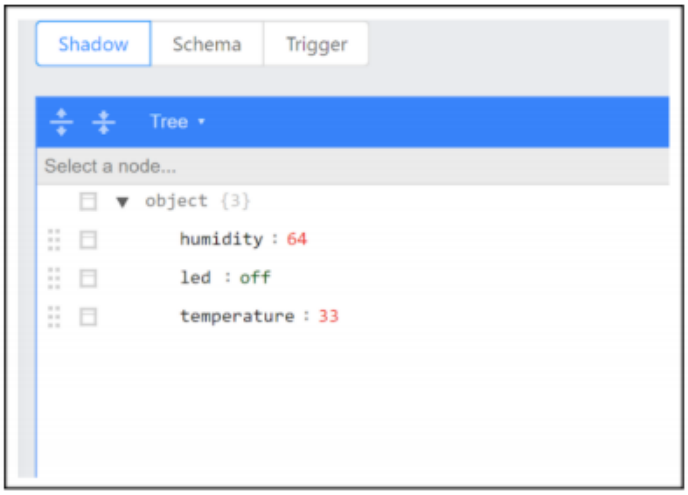
The setup function attempts to use the internet by running the command Ethernet.begin and initializes the dht by running the dht.begin command.

```
void setup() {
  Serial.begin(115200);
  pinMode(LED,OUTPUT);
  digitalWrite(LED,1); // เริ่มต้นให้ LED ดับ
  Serial.println("Starting...");
  if (WiFi.begin(ssid, password)) {
    while (WiFi.status() != WL_CONNECTED) {
      delay(1000);
      Serial.print(".");
    }
  }
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  client.setServer(mqtt_server, mqtt_port);
  client.setCallback(callback); // รับข้อความจาก MQTT
  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.

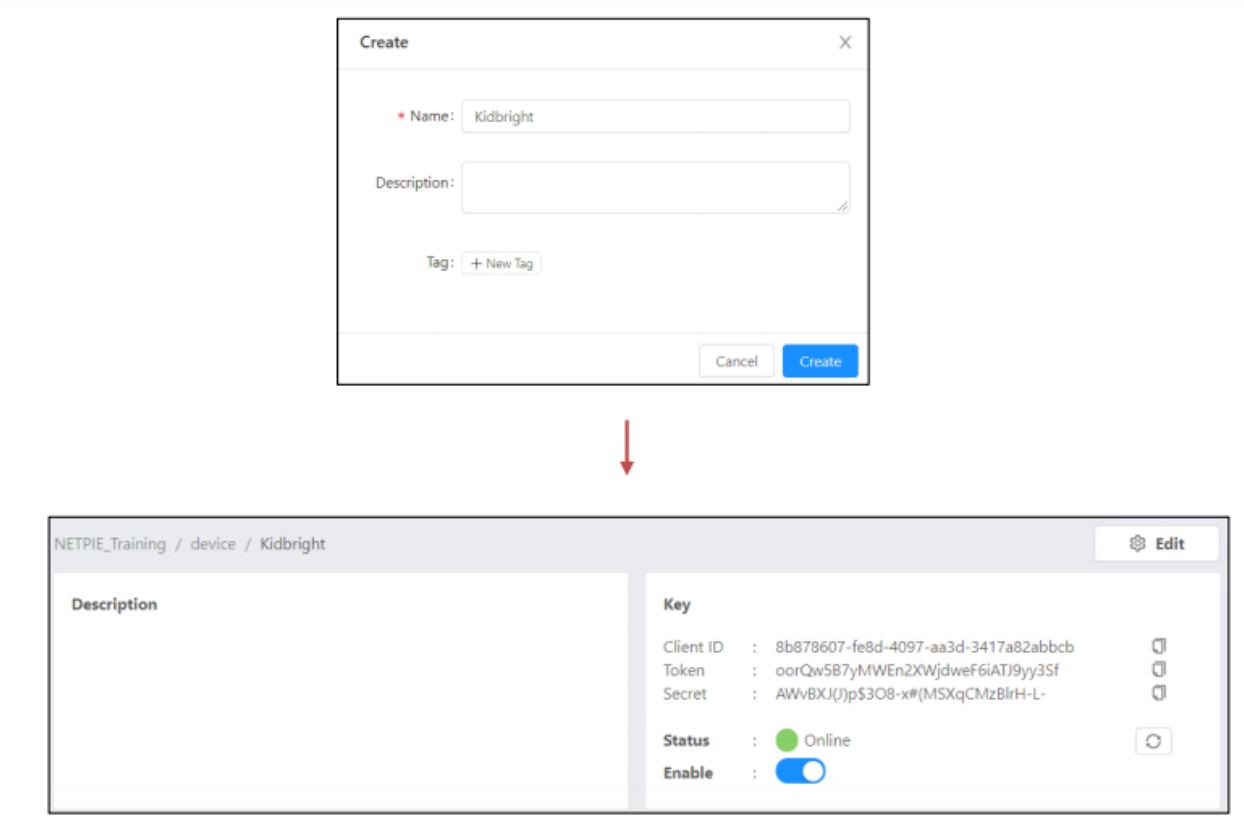
```
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  int humidity = dht.readHumidity();
  int temperature = dht.readTemperature();
  String data = "{\"data\": {\"temperature\":" + String(temperature) + ", \"humidity\": \"" +  String(humidity) + "\"}}";
  Serial.println(data);
  data.toCharArray(msg, (data.length() + 1));
  client.publish("@shadow/data/update", msg);
  delay(2000);
}
```

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



Creating a Device on NETPIE2020

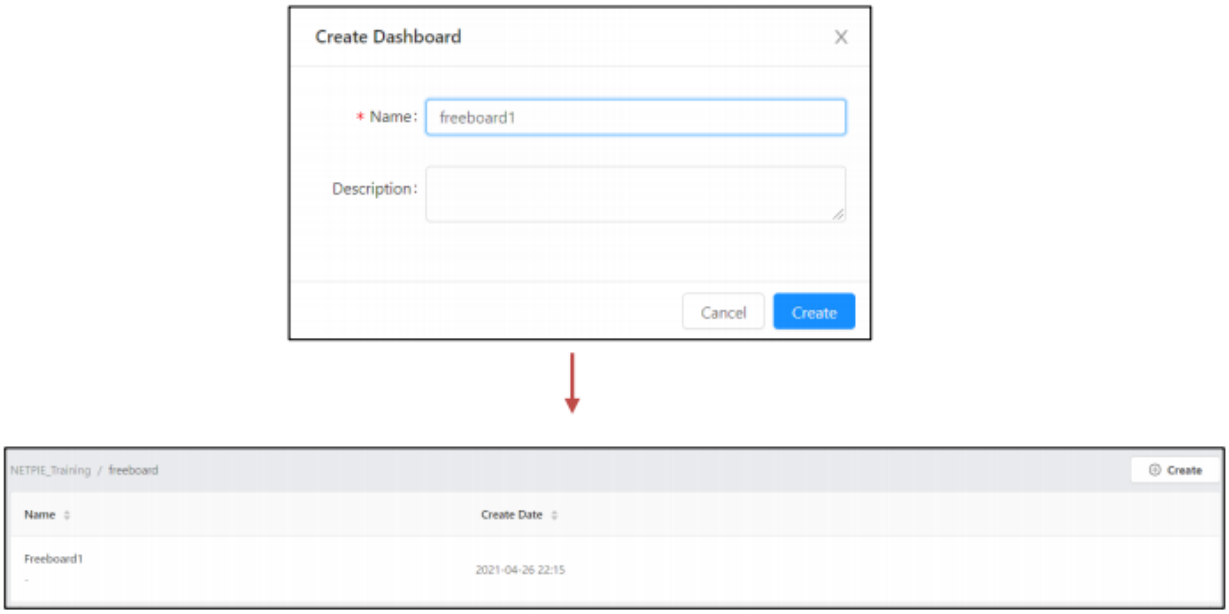
1.Start by selecting the menu Device List > Create and name the device as shown in the figure below.



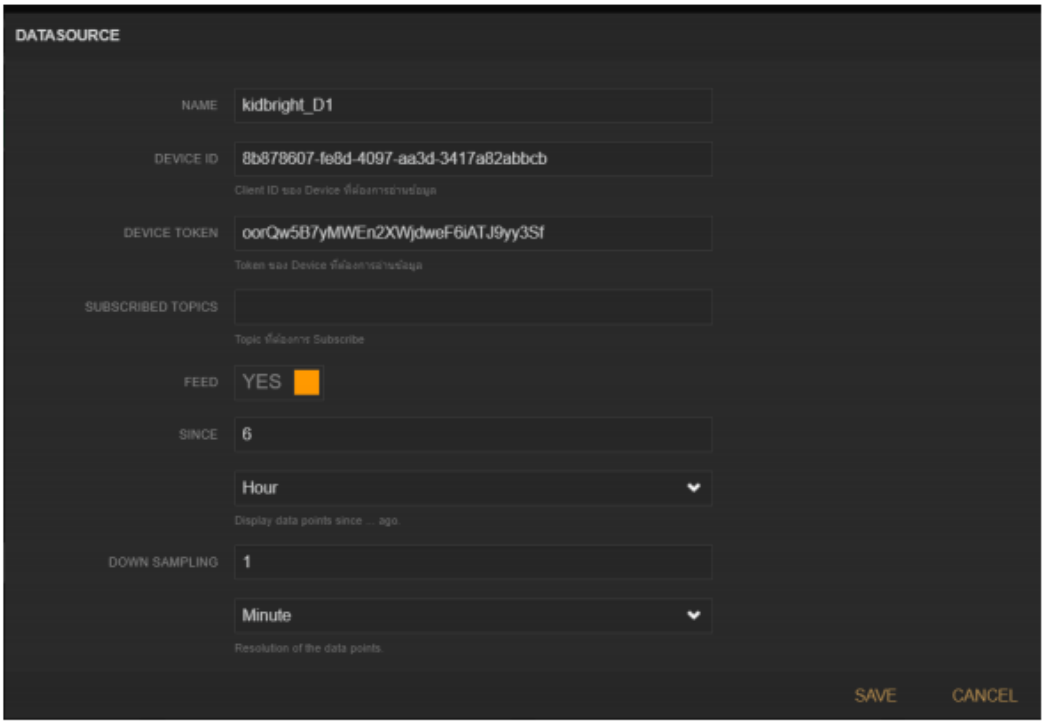
Creating device on NETPIE2020

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.

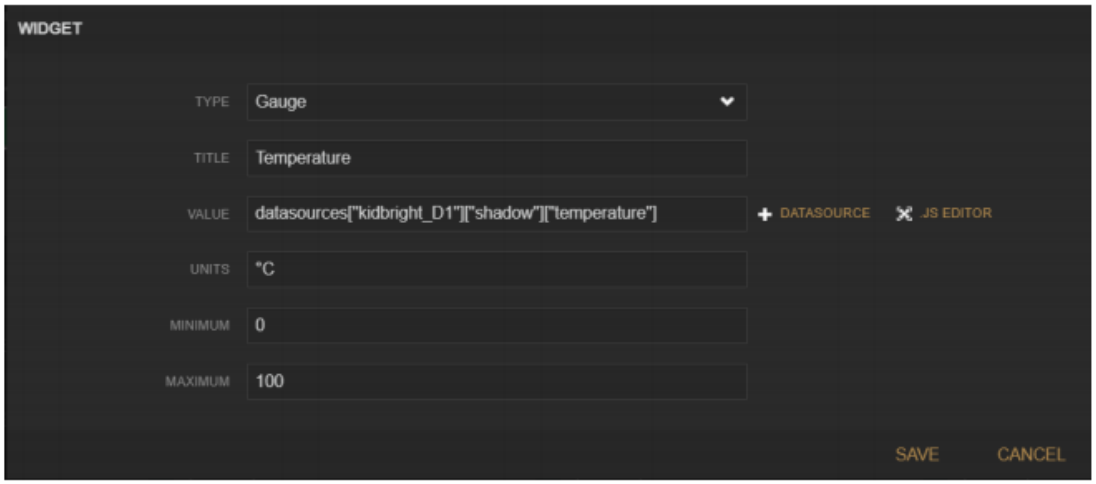


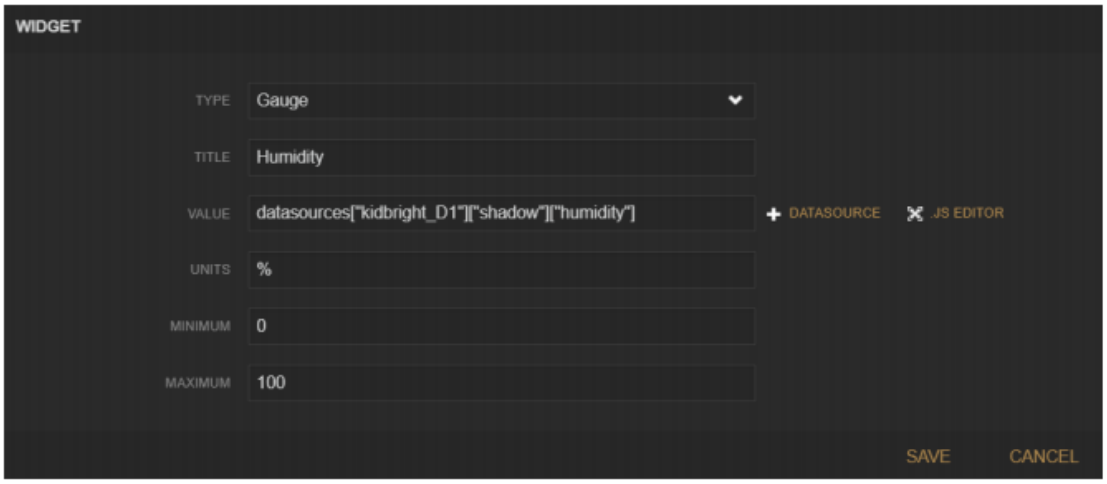
2.Fill in the details, which include Name, Device ID, and Device Token that can be found in the DeviceList named Kidbright.

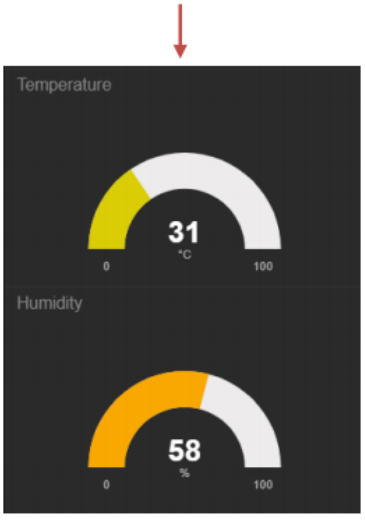


(NOTE: If the user enables the Feed, Freeboard keeps on calling the API. So, it is better to turn off the Feed if the user is not intended to use it.)

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.

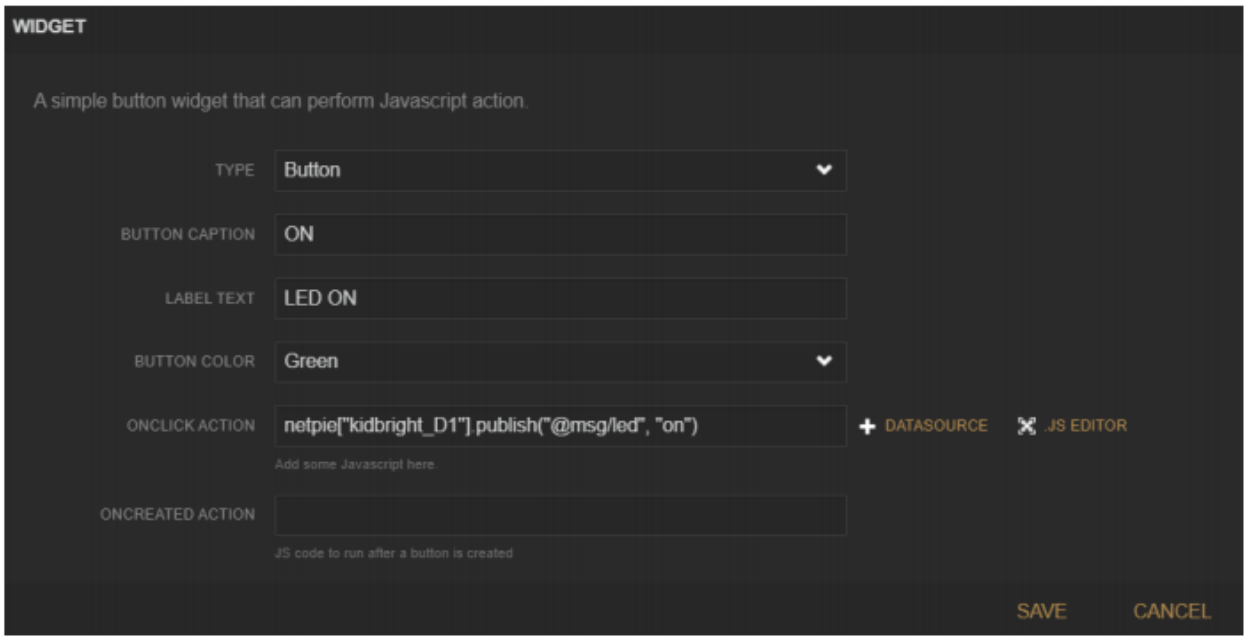




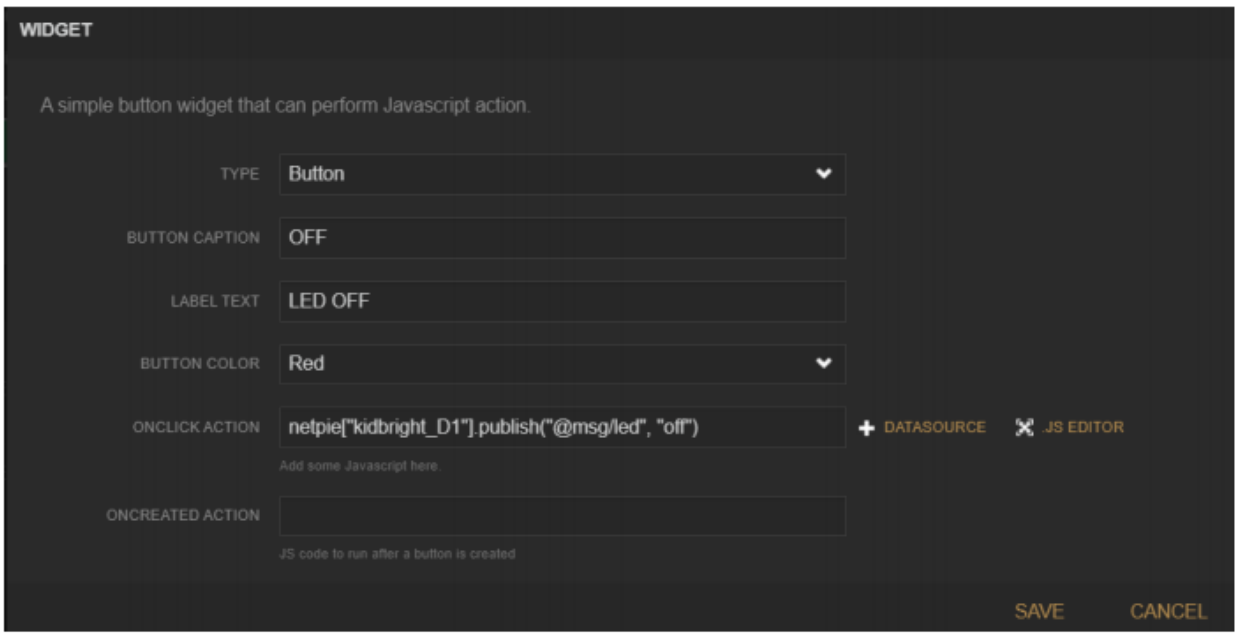


Displaying Temperature and Humidity values stored in Shadow

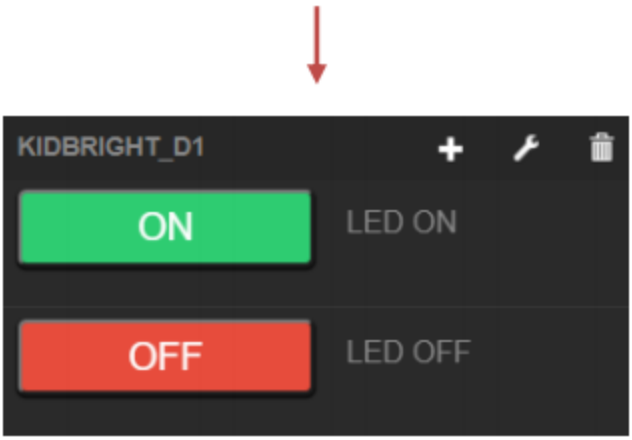
4.Create ON/OFF buttons and set the ONCLICK ACTION on the ON button to netpie[“DataSourceName”].publish(“@msg/led”,“on”) and netpie[“DataSourceName”].publish(“@msg/led”,“off”) for the OFF button.



Settings for ON button



Settings for OFF button


KIDBRIGHT_D1
ON
LED ON
OFF
LED OFF