Workshop IBM Cloud and IBM Watson IoT Platform Programming the ESP8266

V1, March 27, 2019

Yves Debeer, Developer Advocate, IBM
yves_debeer@.ibm.com

Introduction

This guide will assist you to start connecting a ESP8266 to the IBM Watson IoT platform.
Assuming you have already installed the Arduino software onto your laptop and are able to communicate to your ESP8266 via the serial port.

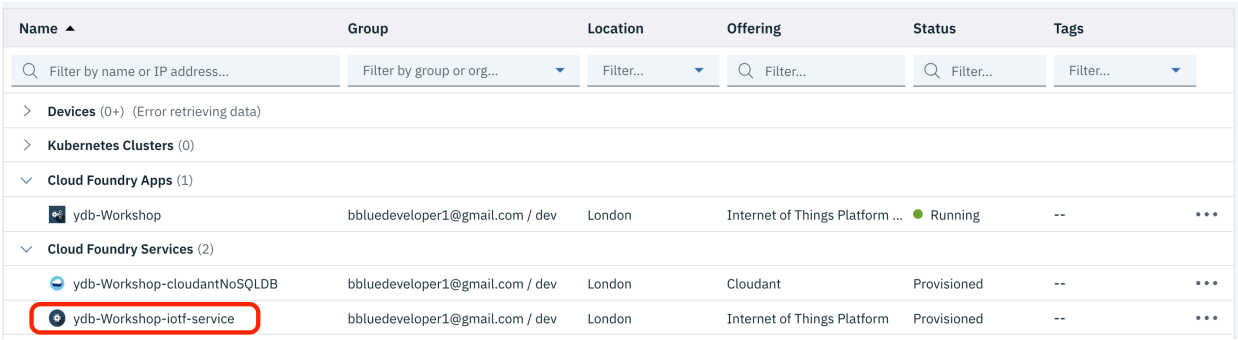
Devices in IBM Watson IoT Platform

The first step is to register a new device.

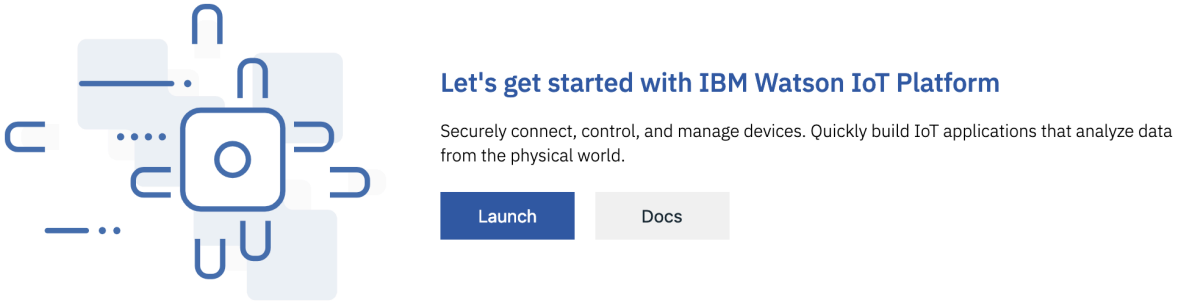
Open the IBM Watson IoT Platform

Go back to the IBM Cloud Dashboard. If needed follow the next steps.

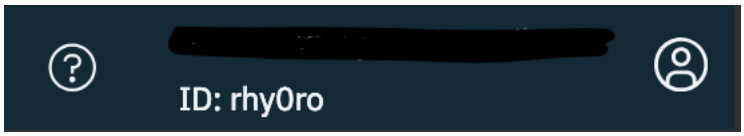
1. Open a browser window and go to https://cloud.ibm.com/login and login to your account.
2. Go to the Dashboard by clicking on the top left so-called “Hamburger-icon” and selecting “Dashboard”.
3. Within the Dashboard find the find the Internet of Things Platform service listed under “Cloud Foundry Services”. The name will be different from the one in this screenshot. Click on it to open the main page.



4. Observe the Welcome page click on ***Launch*** button to enter into the IBM Watson IoT Platform organization space. The IoT organization is a space used for connecting and managing devices to the IBM Watson IoT Platform, so that your applications can access their live and historical data.



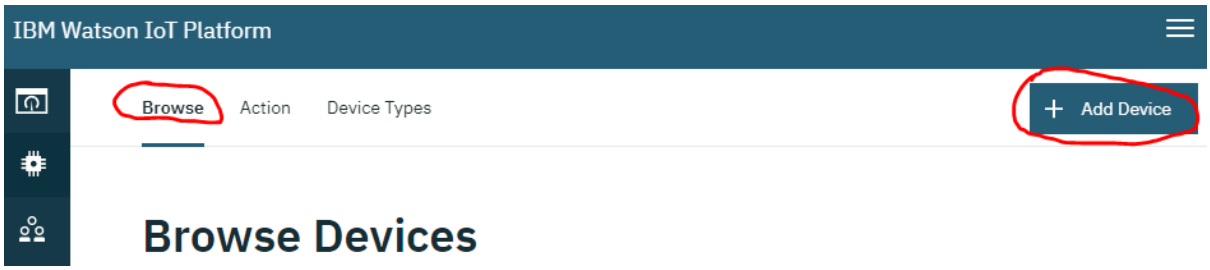
5. Observe the right top corner of the page, where you'll find the six character Organization ID that is created for you to identify your instance of the Watson IoT Platform service. Here you can add, connect and manage your IoT devices.



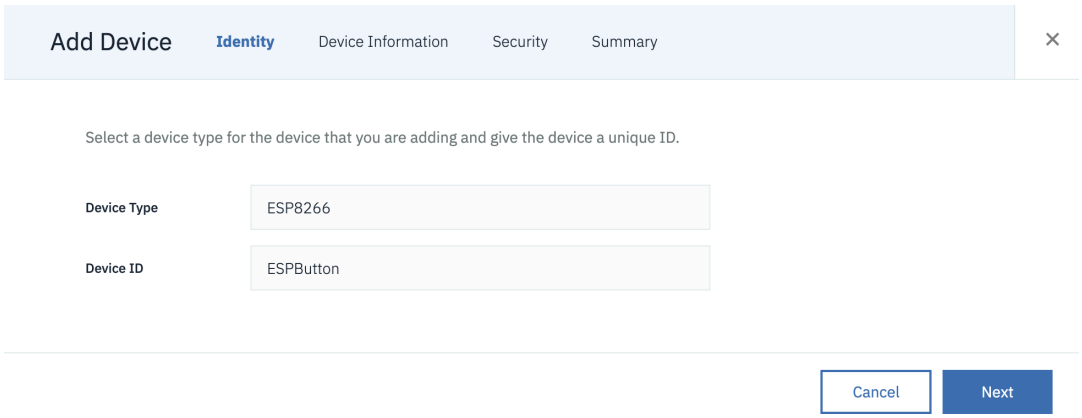
Add Device in IBM Watson IoT Platform

A device can be defined that has a connection to the internet and has data it wants to get into the cloud. And devices can accept commands from applications as well. You need to add the device in IBM Watson IoT Platform Organization before connecting it to the IoT. Carry out the following steps to add the device in IBM Watson IoT Platform,

1. In the IBM Watson IoT Platform dashboard choose the Devices menu on the left, click the Browse tab and then Add Device button as shown below.

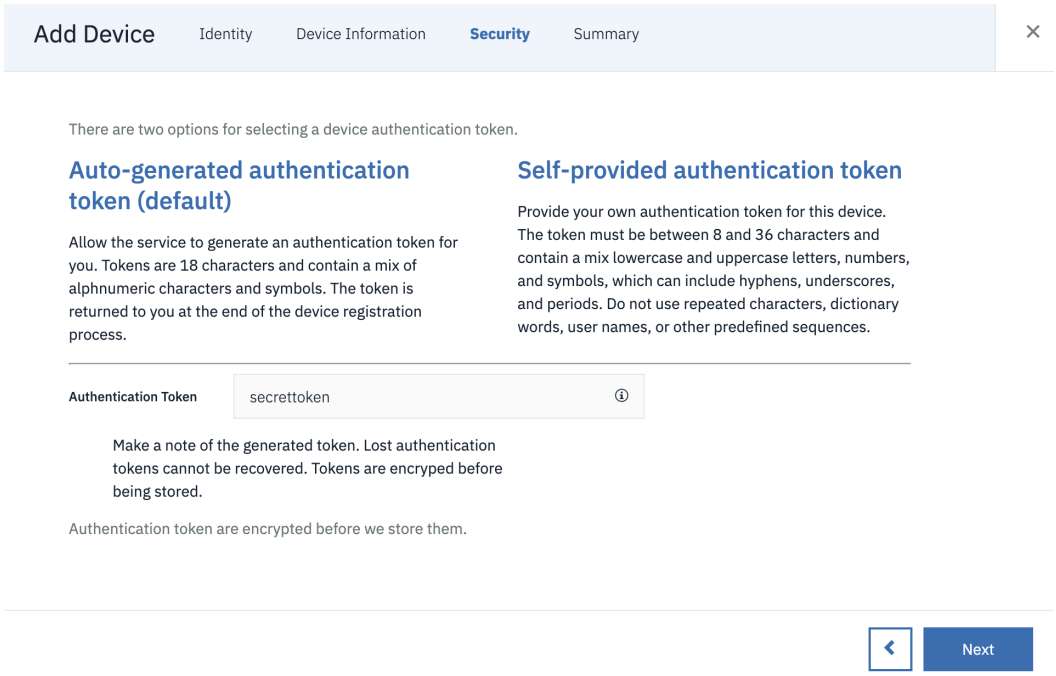


2. Choose the device type "**ESP8266**" and a device ID "ESPButton" (the device type will be automatically created for you.

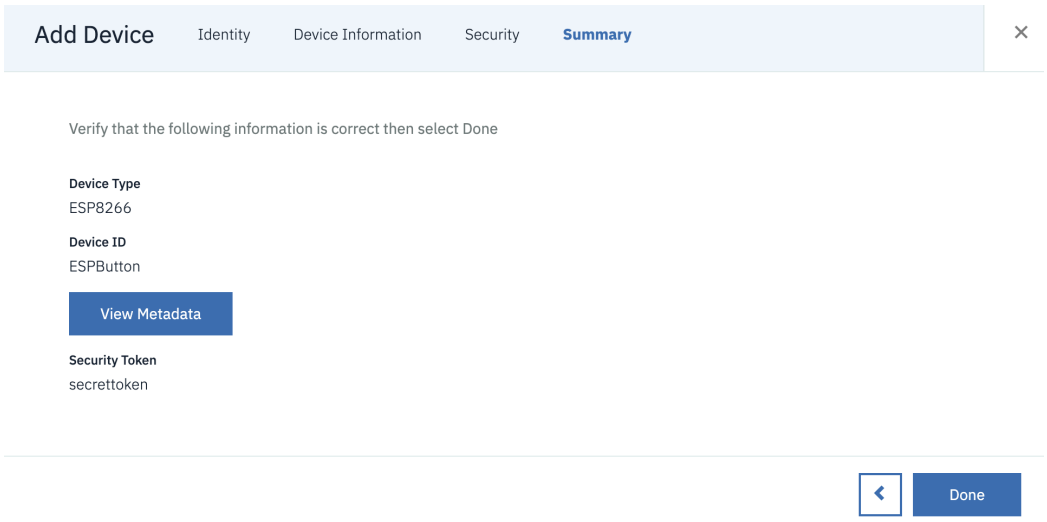


3. In the Device Information page you can enter more information about your device. Click Next.

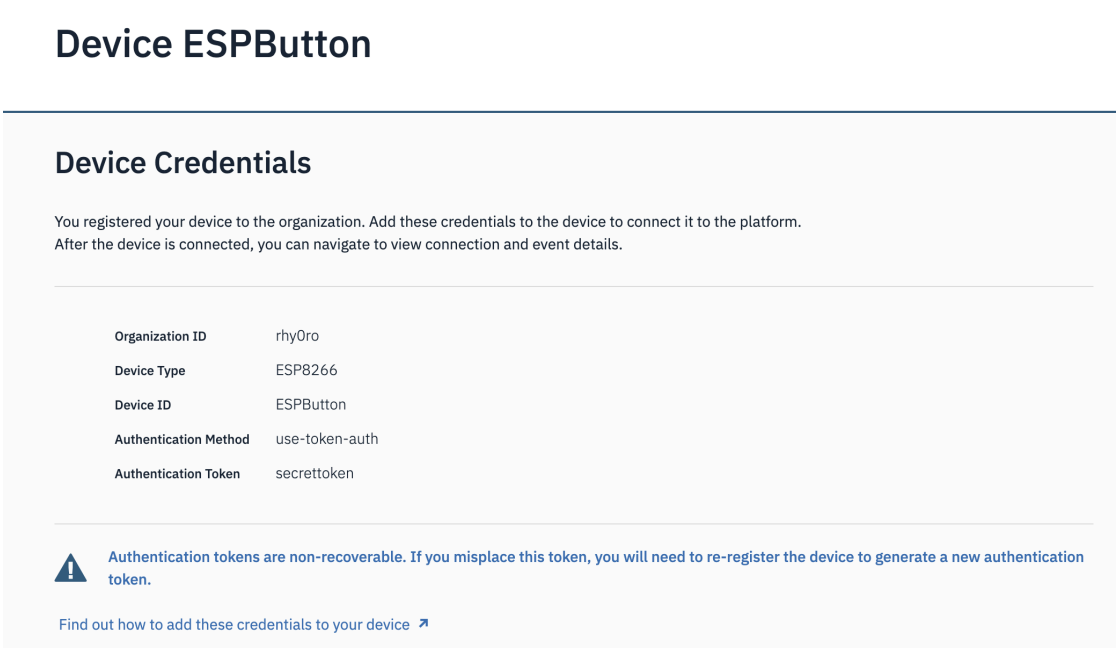
4. In the next page, you can either add your own authentication token, or allow the IBM Watson IoT Platform to generate a token for you. The IBM Watson IoT Platform generated token will be 18 characters long and will contain a mix of alphanumeric characters and symbols. The token will be returned to you at the end of the registration process. In case if you want to add your own token, enter the token as shown below, then click Next.



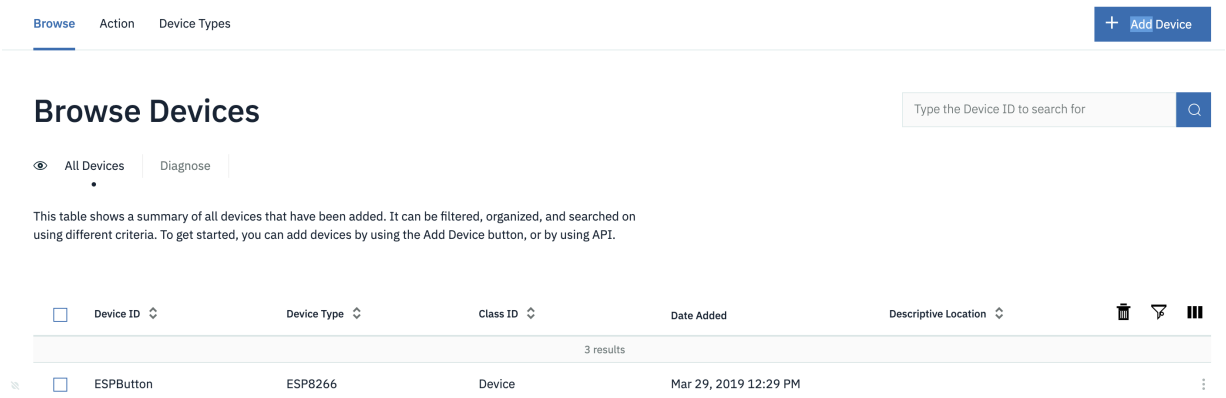
5. As shown below, you will be given a summary page to verify the details before adding the device to IBM Watson IoT Platform. Verify and click Done.



7. At this step, the device is registered to your Organization and you will be provided with the registration details as marked below. To get your device connected, you need to add the credentials to your device. So make a note of them, especially of the token since this is the last time that you'll see it.



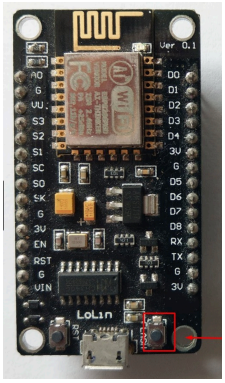
8. Click the browser's Back button to get back to the main dashboard and observe that the device is added in your Organization.



So, now we have successfully added a device in the IBM Watson IoT Platform organization.

Programming the ESP8266 using Arduino

Let's start with a basic sketch which will allow to detect when the "Flash"-button is pressed on the ESP8266.

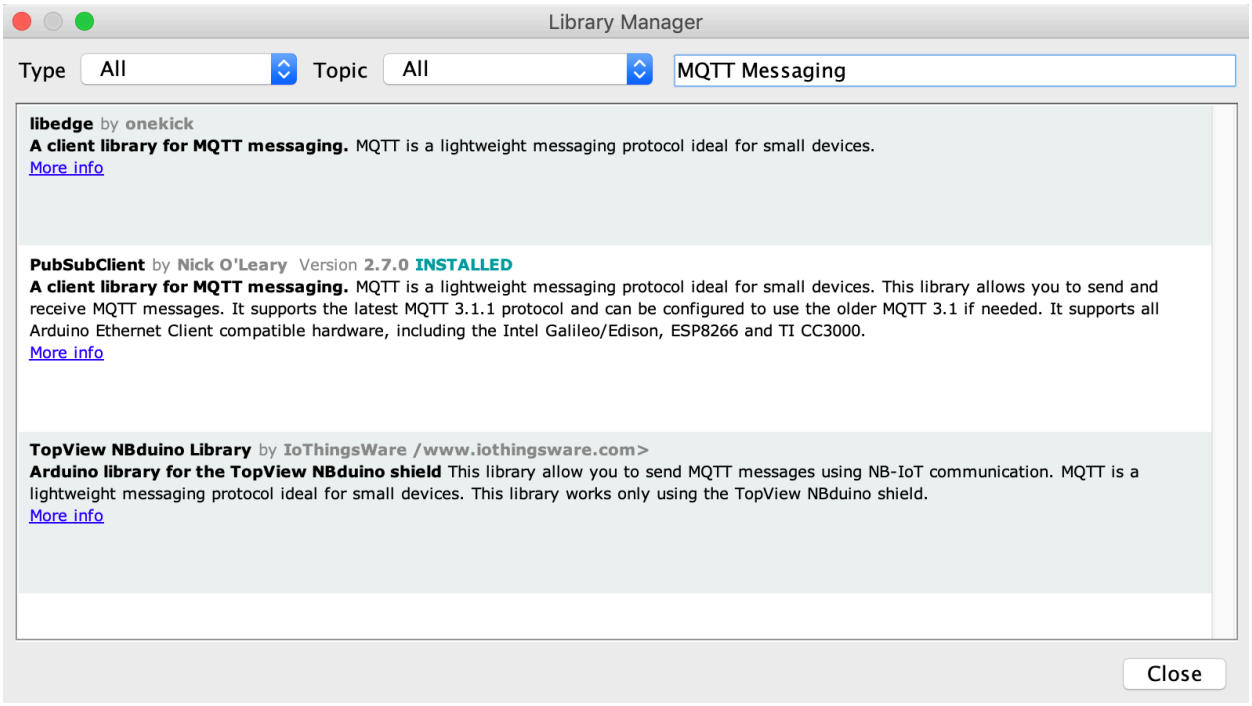


You can find the code here. Open the code in the Arduino editor and upload the code to the ESP8266 using either the “arrow”-button on top of the editor or via editor menu : Sketch -> Upload. If you are seeing errors with uploading the code, make sure to select the correct Board and Port from within the Arduino Tools menu.

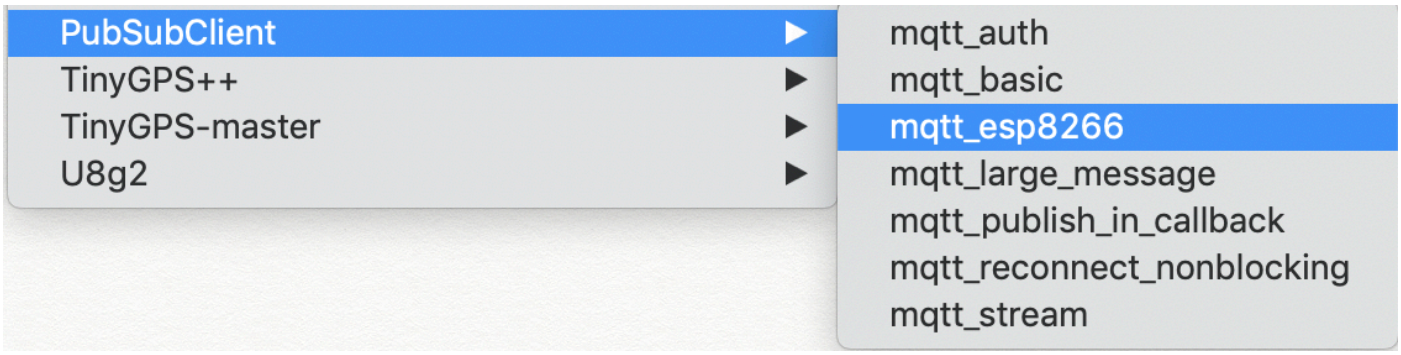
Next, open the “Serial Monitor” via Tools –> Serial Monitor and press the “Flash”-button on the board. You should now see a “Button Pressed” message in the monitor.

Next install the library “PubSubClient” in the Arduino editor via the menu : “Sketch” -> “Include Library” -> “Manage Libraries…”

Install the PubSubClient library by Nick O’Leary



Open the example sketch “mqtt_esp8266” from PubSubclient via the menu “File” -> “Examples”



Start adapting the code as described below. You can also find a copy of the full code here: https://github.com/yvesdebeer/IoT-Workshop-2019/blob/master/mqtt_esp8266_basic.ino

Fill in the correct ssid and password to connect to the WiFi network.

```
const char* ssid = ".......";
const char* password = ".......";
```

Add the MQTT parameters to start of your code and adapt those to correspond with your environment:

Fill in the correct “mqtt_server” variable to correspond with your environment/organization:

for example:

```
const char* mqtt_server = "[redacted].messaging.internetofthings.ibmcloud.com";
```

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

// Watson IoT connection details
#define MQTT_HOST "rhy0ro.messaging.internetofthings.ibmcloud.com"
#define MQTT_PORT 1883
#define MQTT_DEVICEID "d:rhy0ro:ESP8266:ESPButton"
#define MQTT_USER "use-token-auth"
#define MQTT_TOKEN "secrettoken"
#define MQTT_TOPIC "iot-2/evt/status/fmt/json"
#define MQTT_TOPIC_DISPLAY "iot-2/cmd/update/fmt/json"

// Update these with values suitable for your network.
```

Adapt the reconnect() function in the code and replace this with the following code in order to connect the MQTT client to the IBM Cloud IoT Platform:

```
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect(MQTT_DEVICEID, MQTT_USER, MQTT_TOKEN)) {
      Serial.println("connected");
      client.subscribe(MQTT_TOPIC_DISPLAY);
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
```

Adapt the loop() function at the end of the code and replace it with the following:

```
void loop() {

  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  buttonState = digitalRead(button);
  if (buttonState == HIGH) {
    delay(500);
  }
  else {
    Serial.println("Button Pressed");
    String payload = "{ \"d\" : {";
    payload += "\"Button Pressed\":\""; payload += "True\",";
    payload += "\"Local IP\":\""; payload += WiFi.localIP().toString(); payload += "\"";
    payload += "}}";
    Serial.println(payload);

    if (client.publish(MQTT_TOPIC, (char*) payload.c_str())) {
      Serial.println("Publish ok");
    } else {
      Serial.println("Publish failed");
    }
    delay(500);
   }
}
```

Finally define the two variables to the beginning of your code:

```
const int button = 0;
int buttonState = 0;
```

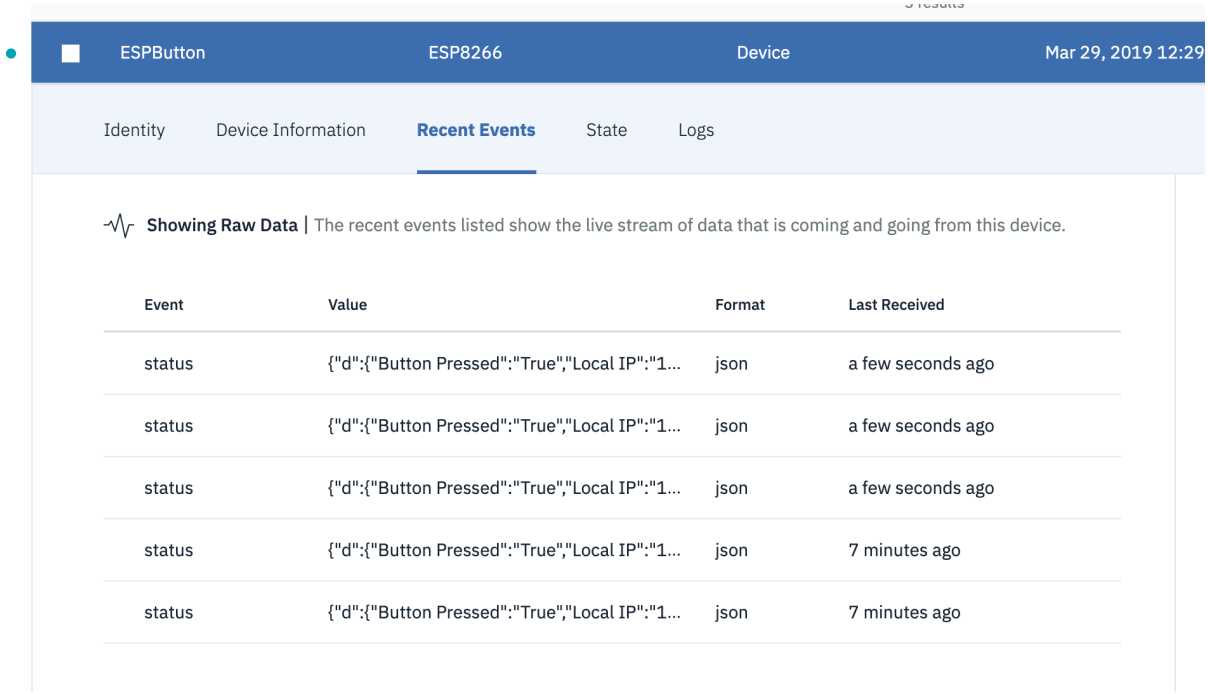
Now you should be able to compile and upload the code to the ESP8266 board.
You can do this by clicking the upload button :

After a successful compilation, watch the Arduino monitor log : you should see a successful connection the WiFi and MQTT server. Next try to press the “Flash’ button on the board. This should result in a JSON payload being published via MQTT to the IBM Cloud IoT Platform.

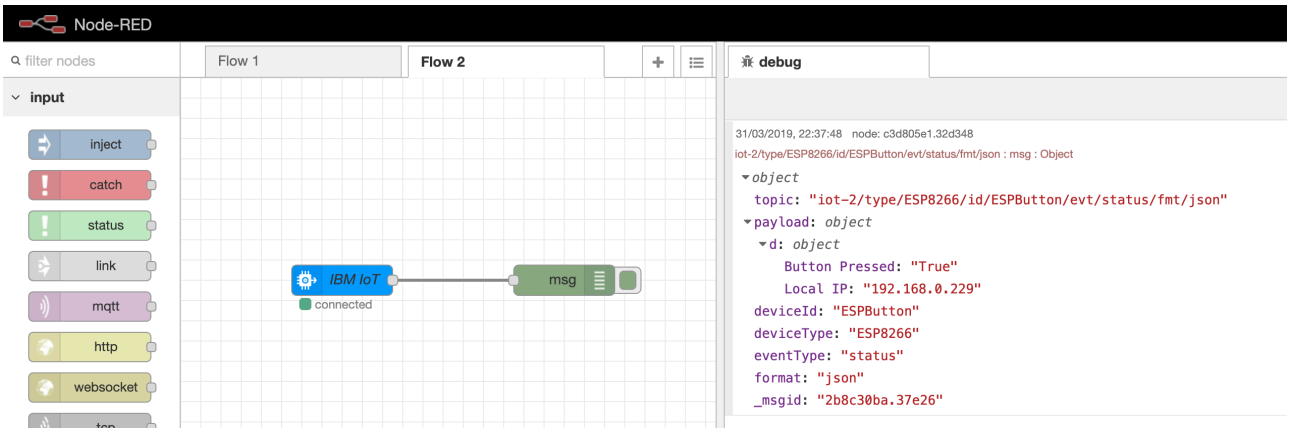
```
WiFi connected
IP address:
192.168.0.229
Attempting MQTT connection...connected
Button Pressed
{ "d" : {"Button Pressed":"True","Local IP":"192.168.0.229"}}
Publish ok
Attempting MQTT connection...connected
```

Autoscroll | No line ending | 115200 baud | Clear output

You can also verify the arrival of the JSON payload within the IBM Cloud IoT Platform.
Select your device ESPButton from the Browse Devices. Select the “Recent Events“ tab as shown below.

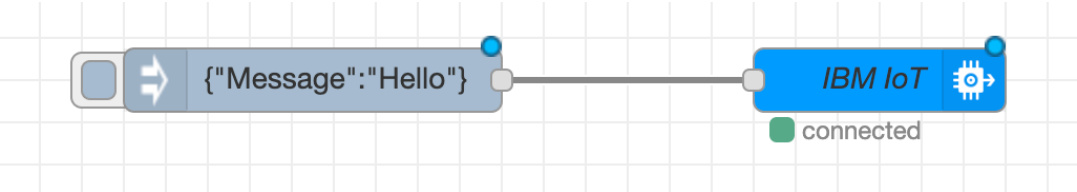


Next, select you Node-Red environment and watch the JSON payload in the Debug tab as soon as you hit the “Flash”-button on the ESP8266 board.

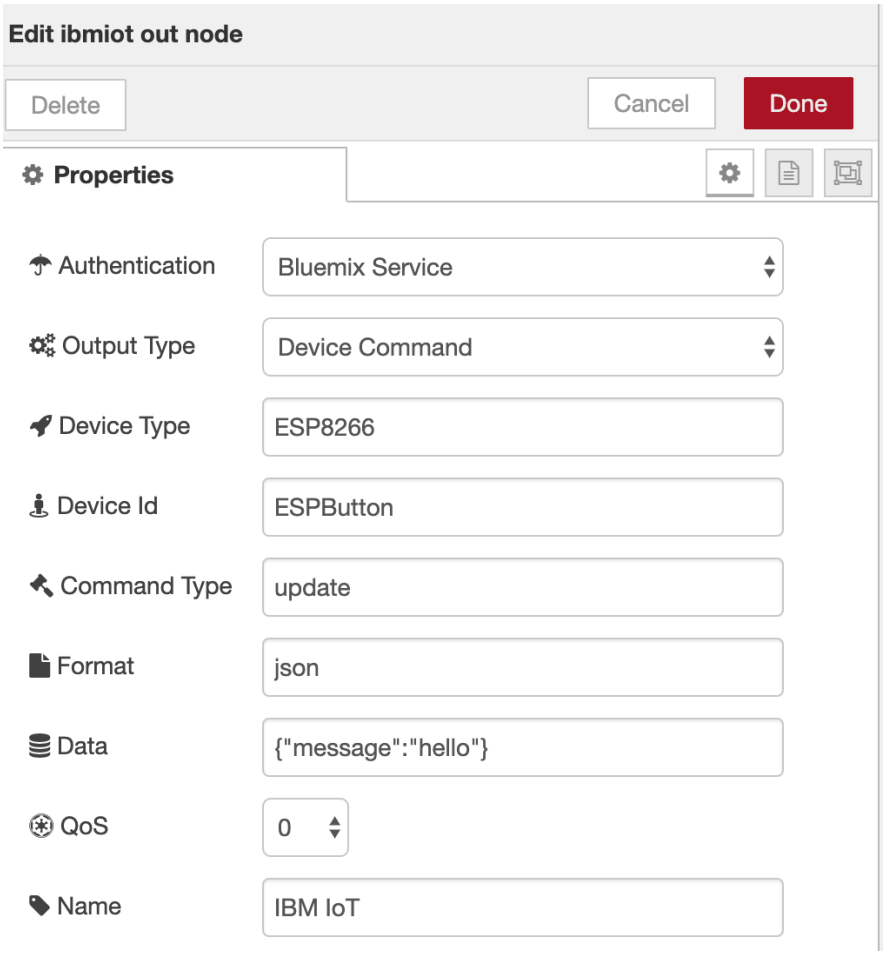


As a final step we can check if we are able to send a command back to the board.
Examine the Arduino code and watch how we defined a callback function which will be called

when a message is pushed on a topic MQTT_TOPIC_DISPLAY="iot-2/cmd/update/fmt/json". We do this within Node-Red with a flow as show below:



The settings of the IBM IoT out node is shown below



Hit the Inject button on the Inject node and watch the Arduino monitor for the result. Notice the JSON payload being received on the board.

```
Attempting MQTT connection...connected
Button Pressed
{ "d" : {"Button Pressed":"True","Local IP":"192.168.0.229"}}
Publish ok
Attempting MQTT connection...connected
Attempting MQTT connection...connected
Attempting MQTT connection...connected
Message arrived [iot-2/cmd/update/fmt/json] {"Message":"Hello"}
Attempting MQTT connection...connected
Attempting MQTT connection...connected
```

To summarize – the Arduino code example shows how to monitor a sensor, in this case the “Flash”-button.The code also allows to receive a command from an external program. An example could be to monitor a room temperature and when a threshold is reached, a valve or switch could be turned on to start cooling.