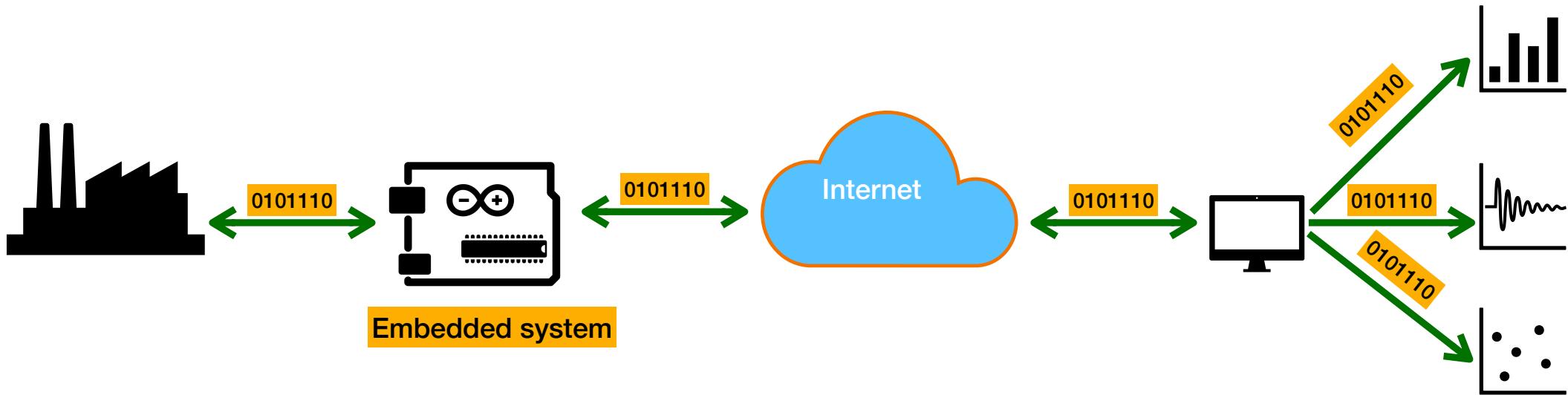


# **IoT with ESP32**

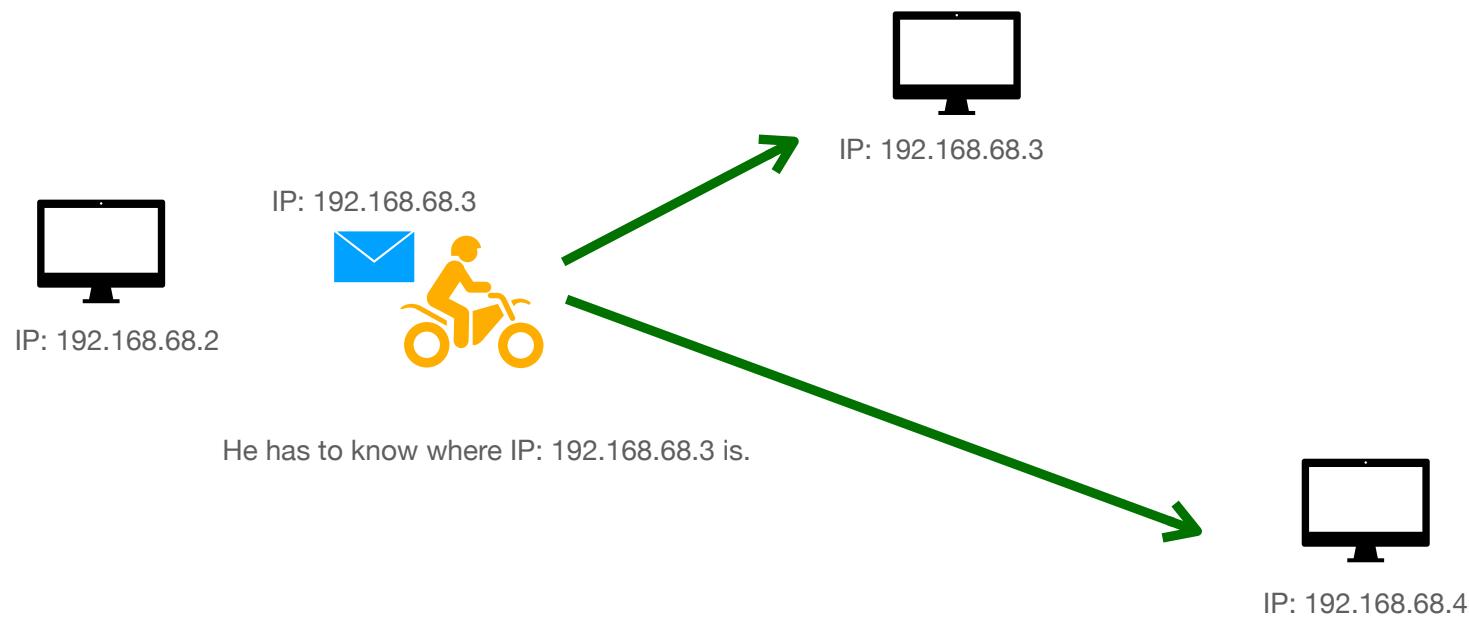
**Apichart Intarapanich**

# IoT

Machines talk to machines

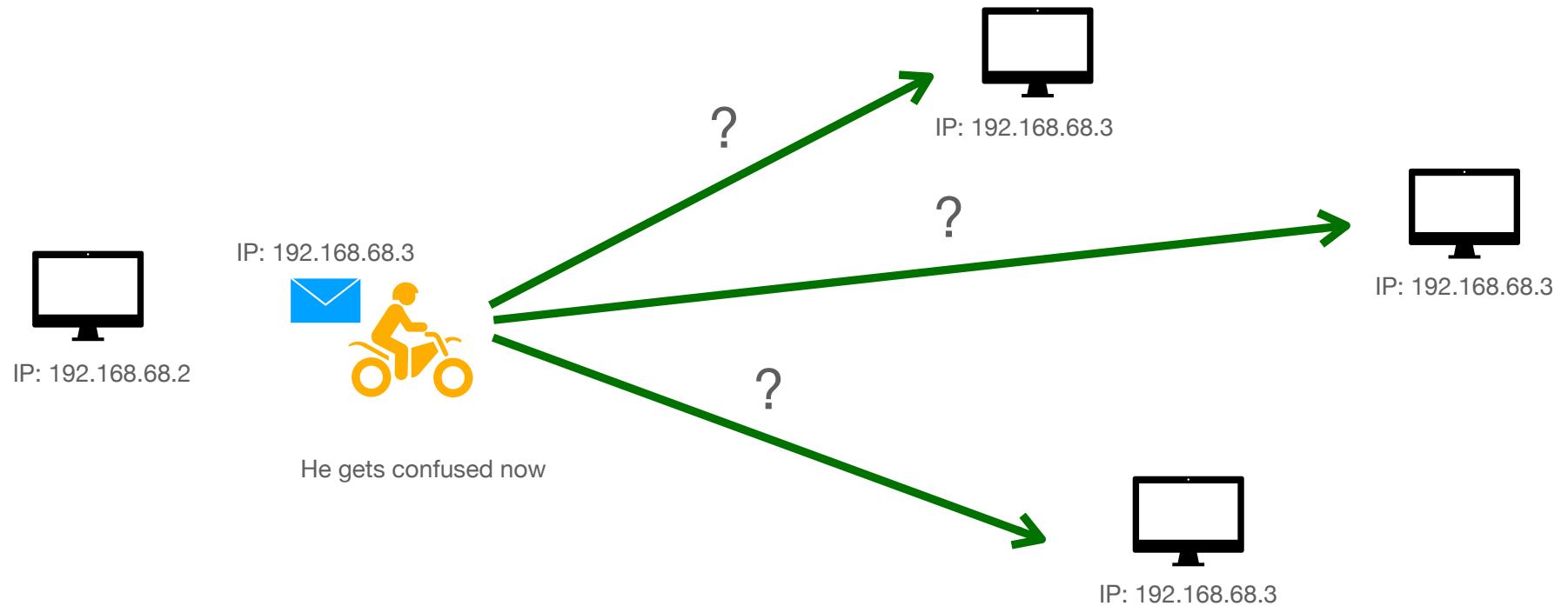


# How does the internet work?



# IP address

IP address should be unique



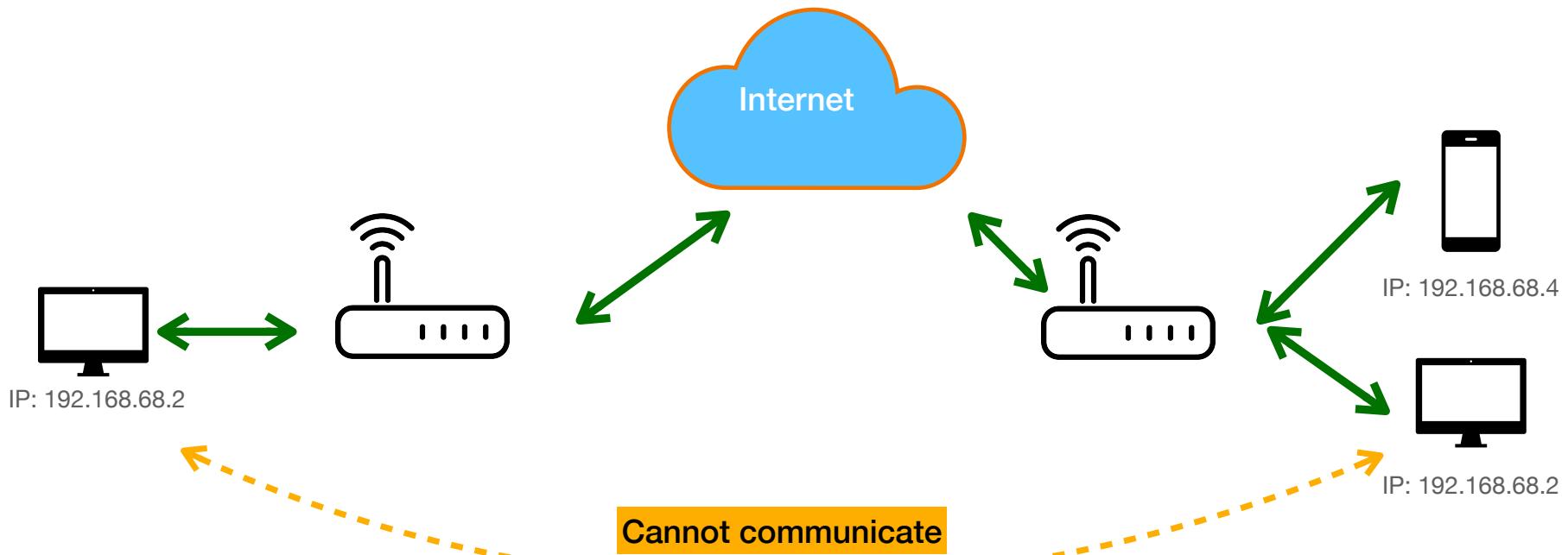
# **IP Version 4**

## **32-bit IP address**

- 4 billion IP address but there are 7 billions people on the planet
- To solve this problem, we use Network Translation Table (NAT) at the router
- The unique IP V4 address called *public IP*. As name infer, the address can be seen by everyone on the planet.

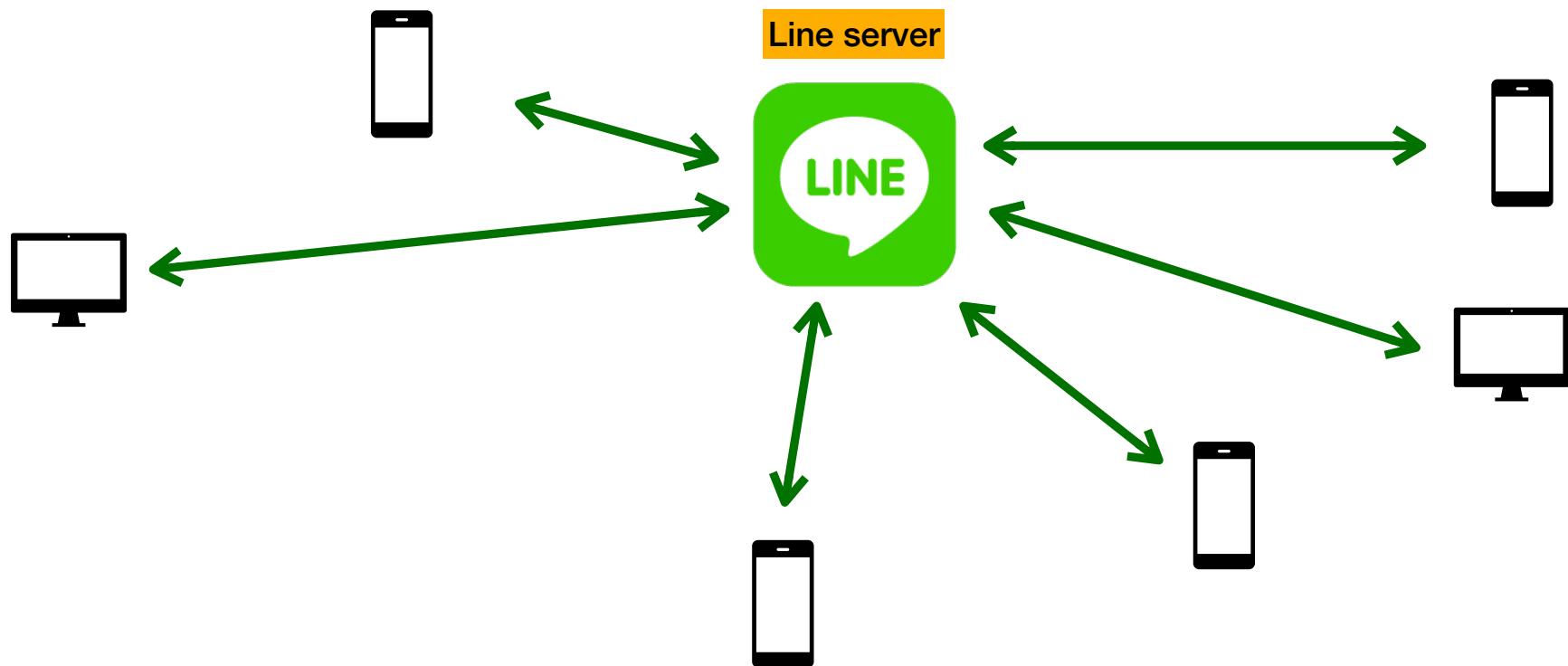
# NAT Problem

Direct end-to-end communication



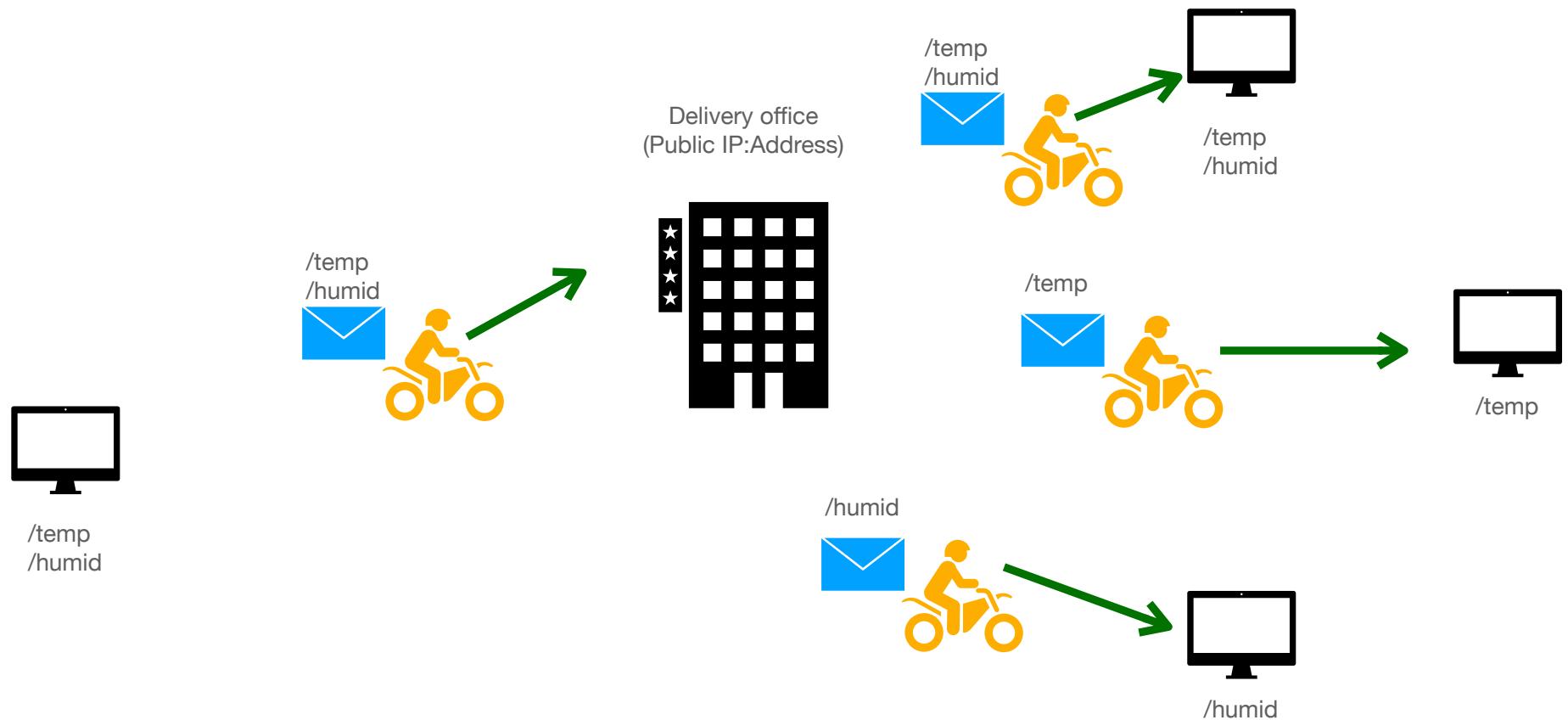
# Chat app

## Line chat



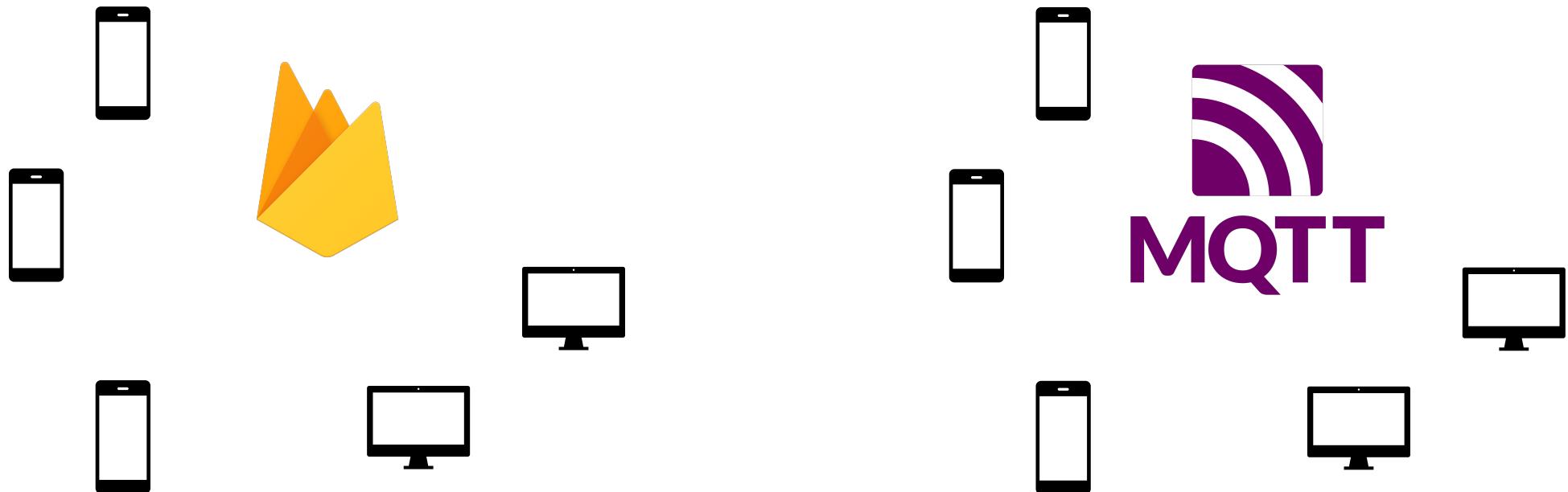
# Create our own transport system

Create a private virtual network

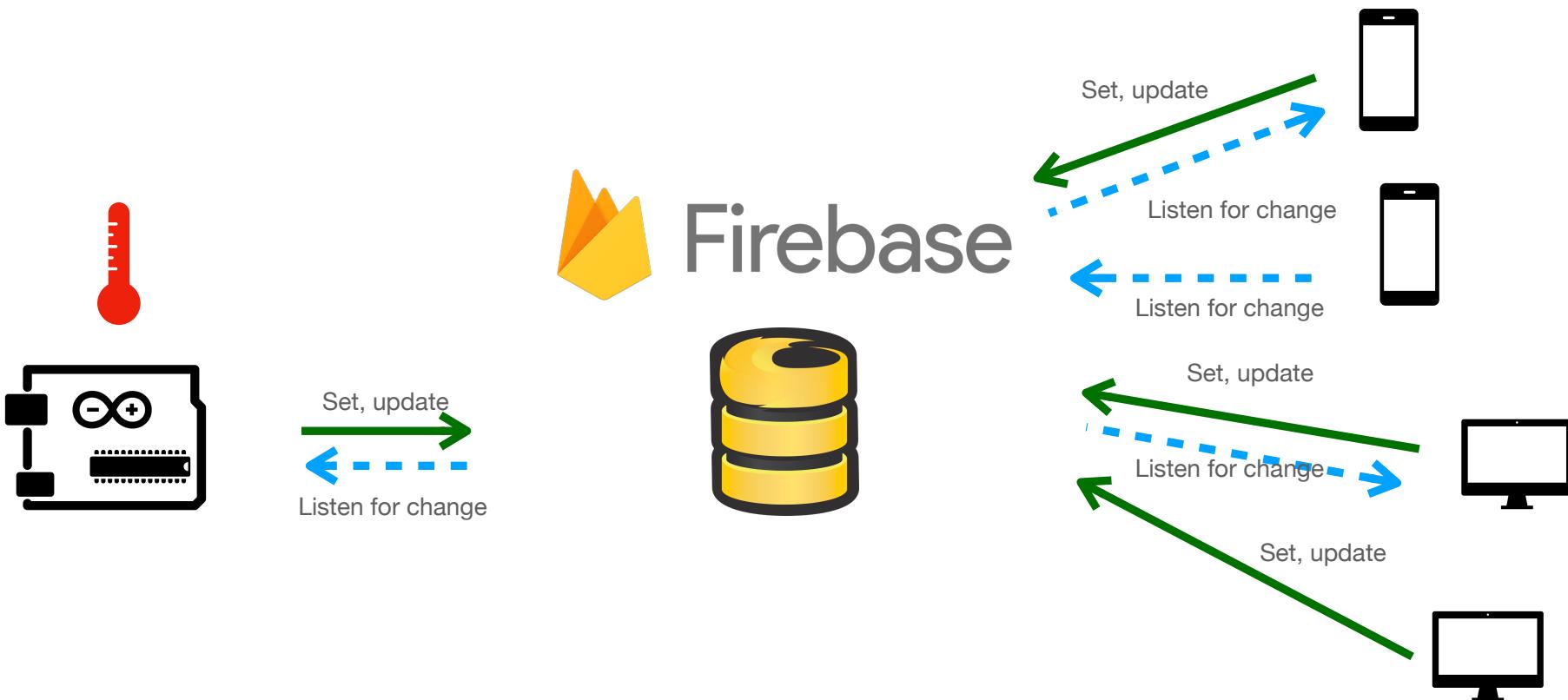


# Delivery office

In this class we will use Google Firebase and MQTT Broker

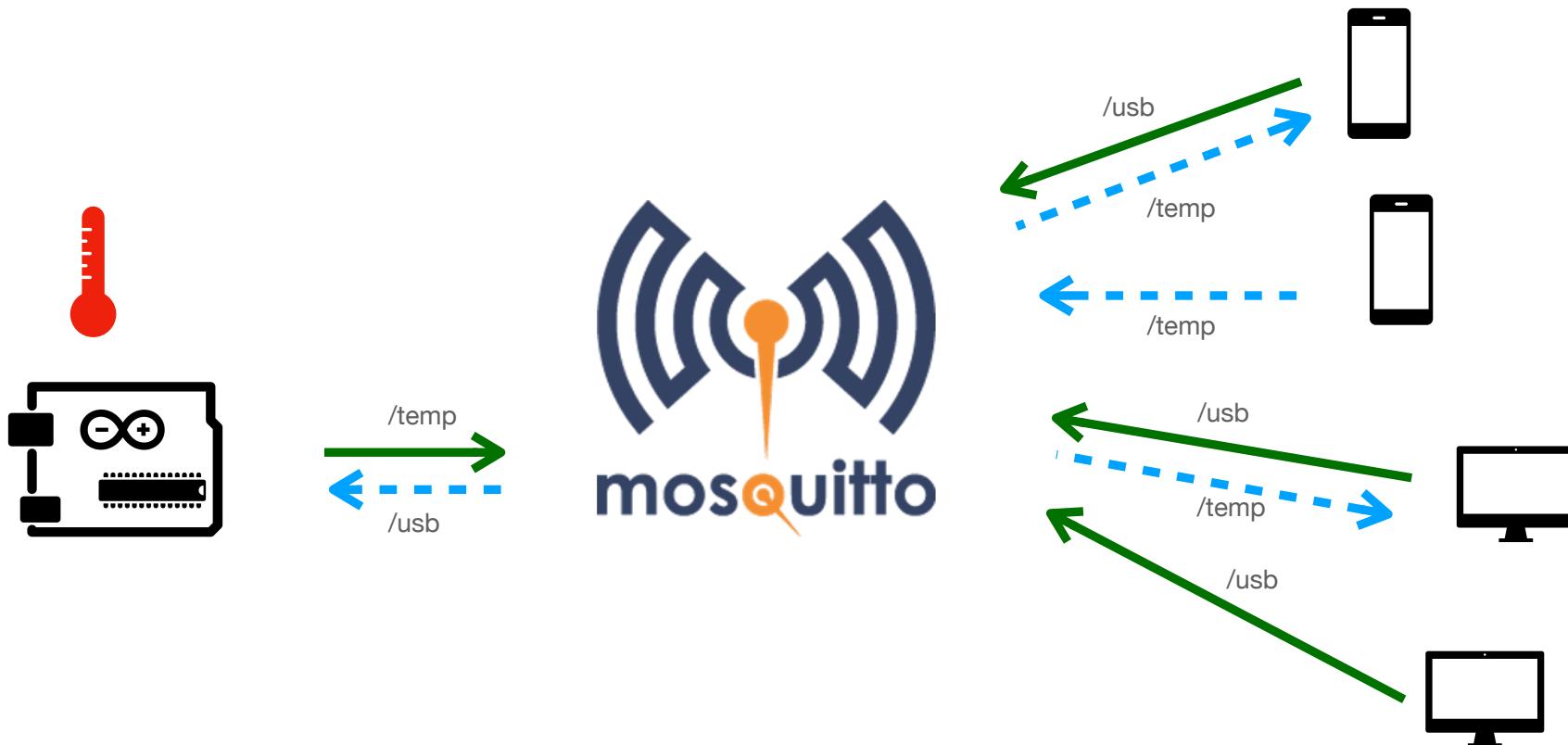


# Firebase realtime database



# MQTT publisher and subscriber

Pub/sub model



# Topics and QoS

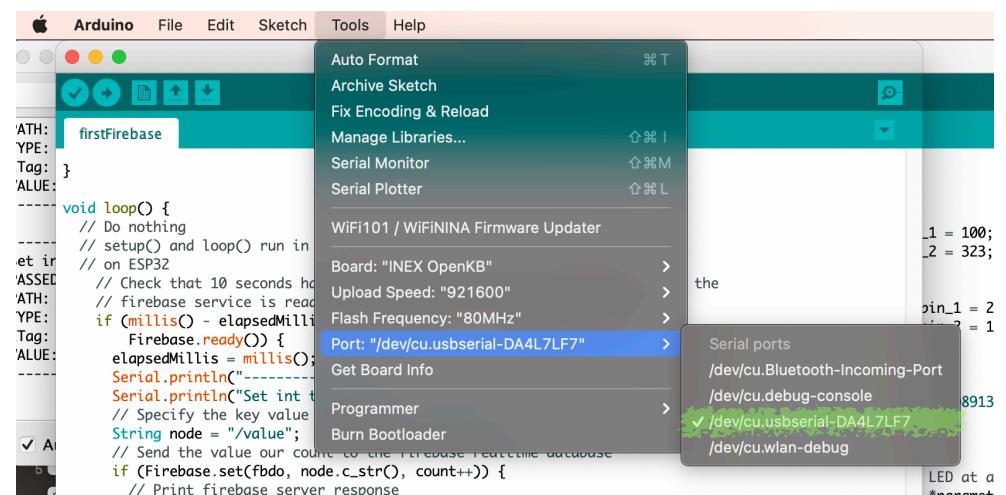
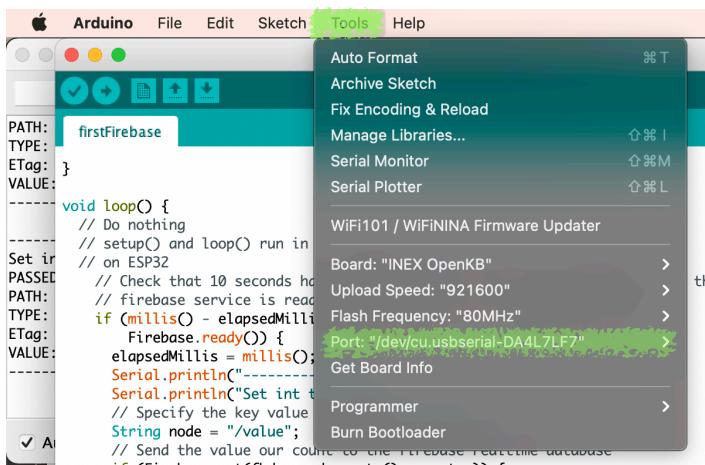
- Topic : /temp
- QoS levels : 0, 1 and 2
  - At most once (0)
    - This service level guarantees a best-effort delivery
  - At least once (1)
    - QoS level 1 guarantees that a message is delivered at least one time to the receiver
  - Exactly once (2)
    - This level guarantees that each message is received only once

# **Why is MQTT protocol used in IoT**

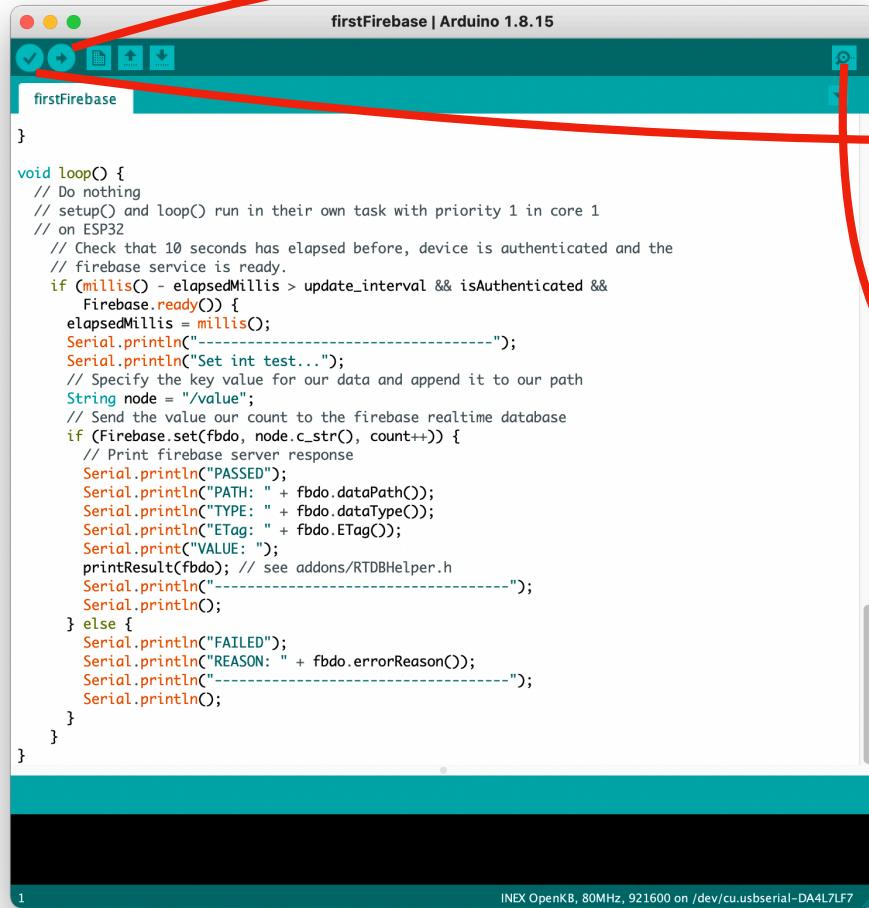
- There are many protocol to connect computers over the internet
- Only few of them can be fit in embedded system
- MQTT is a protocol that can fit in embedded system

# Basic Arduino IDE

## Check board connection



# Basic Arduino IDE



```
firstFirebase | Arduino 1.8.15
firstFirebase

}

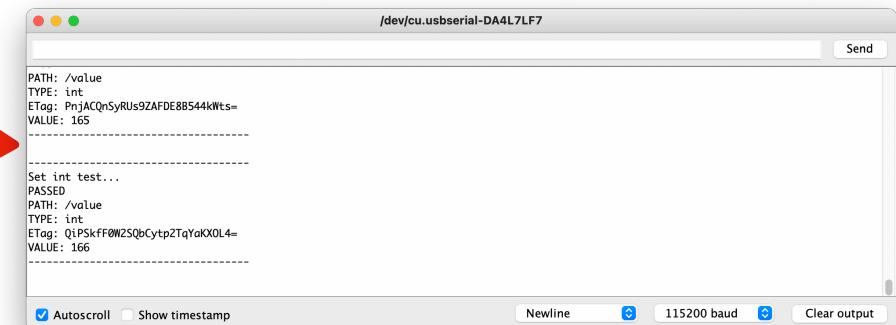
void loop() {
  // Do nothing
  // setup() and loop() run in their own task with priority 1 in core 1
  // on ESP32
  // Check that 10 seconds has elapsed before, device is authenticated and the
  // firebase service is ready.
  if (millis() - elapsedMillis > update_interval && isAuthenticated &&
      Firebase.ready()) {
    elapsedMillis = millis();
    Serial.println("-----");
    Serial.println("Set int test...");
    // Specify the key value for our data and append it to our path
    String node = "/value";
    // Send the value our count to the firebase realtime database
    if (Firebase.set(fbdo, node.c_str(), count++)) {
      // Print firebase server response
      Serial.println("PASSED");
      Serial.println("PATH: " + fbdo.dataPath());
      Serial.println("TYPE: " + fbdo.dataType());
      Serial.println("ETag: " + fbdo.ETag());
      Serial.print("VALUE: ");
      printResult(fbdo); // see addons/RTDBHelper.h
      Serial.println("-----");
      Serial.println();
    } else {
      Serial.println("FAILED");
      Serial.println("REASON: " + fbdo.errorReason());
      Serial.println("-----");
      Serial.println();
    }
  }
}
```

1 INEX OpenKB, 80MHz, 921600 on /dev/cu.usbserial-DA4L7LF7

Compile program

Compile and download

Open serial terminal



/dev/cu.usbserial-DA4L7LF7

```
PATH: /value
TYPE: int
ETag: PnJAQnSyRUs9ZAFDE8B544kWts=
VALUE: 165
-----
Set int test...
PASSED
PATH: /value
TYPE: int
ETag: QlPSkFF0W2SqbCyp2TqYakX0L4=
VALUE: 166
-----
```

Autoscroll  Show timestamp

Newline  115200 baud  Clear output

# Sensors test

# Test temperature sensor

## readTemp.ino LM73

```
#include <Wire.h>
#define LM73_ADDR 0x4D

int analog_value = 0;
double temp=0;

// Pins
static const int led_pin_1 = 2;
static const int led_pin_2 = 12;

float readTemperature() {
    Wire1.beginTransmission(LM73_ADDR);
    Wire1.write(0x00); // Temperature Data Register
    Wire1.endTransmission();

    uint8_t count = Wire1.requestFrom(LM73_ADDR, 2);
    float temp = 0.0;
    if (count == 2) {
        byte buff[2];
        buff[0] = Wire1.read();
        buff[1] = Wire1.read();
        temp += (int)(buff[0]<<1);
        if (buff[1]&0b10000000) temp += 1.0;
        if (buff[1]&0b01000000) temp += 0.5;
        if (buff[1]&0b00100000) temp += 0.25;
        if (buff[0]&0b10000000) temp *= -1.0;
    }
    return temp;
}

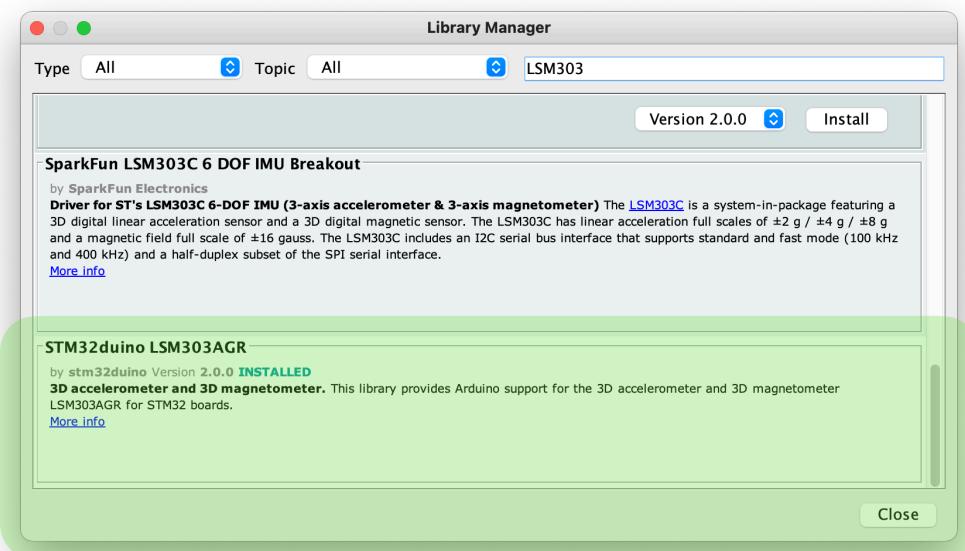
void setup() {
```

```
    Serial.begin(115200);
    Wire1.begin(4, 5);
}

void loop() {
    temp = readTemperature();
    Serial.print("Temp:");
    Serial.println(temp);
    delay(5000);
}
```

# Test LSM303 sensor

## lsm303Test.ino LSM303 is an accel and mag sensors



```
#include <LSM303AGR_ACC_Sensor.h>
#include <LSM303AGR_MAG_Sensor.h>

#define DEV_I2C Wire1 //Define which I2C bus is
used. Wire1 for the Arduino Due
#define SerialPort Serial

// Components.
LSM303AGR_ACC_Sensor Acc(&DEV_I2C);
LSM303AGR_MAG_Sensor Mag(&DEV_I2C);

void setup() {
  // Led.
  pinMode(12, OUTPUT);
  DEV_I2C.begin(4, 5);

  // Initialize serial for output.
  SerialPort.begin(115200);

  // Initialize I2C bus.
  DEV_I2C.begin();

  // Initialize components.
  Acc.begin();
  Acc.Enable();
  Acc.EnableTemperatureSensor();
  Mag.begin();
  Mag.Enable();
}

void loop() {
  // Led blinking.
  digitalWrite(12, HIGH);
  delay(250);
  digitalWrite(12, LOW);
  delay(250);

  // Read accelerometer LSM303AGR.
  int32_t accelerometer[3];
  Acc.GetAxes(accelerometer);

  // Read temperature LSM303AGR.
  float temperature;
  Acc.GetTemperature(&temperature);

  // Read magnetometer LSM303AGR.
  int32_t magnetometer[3];
  Mag.GetAxes(magnetometer);

  // Output data.
  SerialPort.print(" | Acc[mg]: ");
  SerialPort.print(accelerometer[0]);
  SerialPort.print(" ");
  SerialPort.print(accelerometer[1]);
  SerialPort.print(" ");
  SerialPort.print(accelerometer[2]);
  SerialPort.print(" | Mag[mGauss]: ");
  SerialPort.print(magnetometer[0]);
  SerialPort.print(" ");
  SerialPort.print(magnetometer[1]);
  SerialPort.print(" ");
  SerialPort.print(magnetometer[2]);
  SerialPort.print(" | Temp[C]: ");
  SerialPort.print(temperature, 2);
  SerialPort.println(" |");
}
```

# Create our own server

# Software installation

- Server app : Nodejs
- MQTT broker : mosquitto
- Application development environment : Node-red

# Install nodejs on the server

Link to command lines <https://github.com/fllay/iotTraining/wiki/Server-software-installation>

```
sudo apt-get update  
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.35.2/install.sh | bash  
exit
```

```
ssh root@203.150.243.118
```

```
root@voip:~# nvm --version  
0.35.2
```

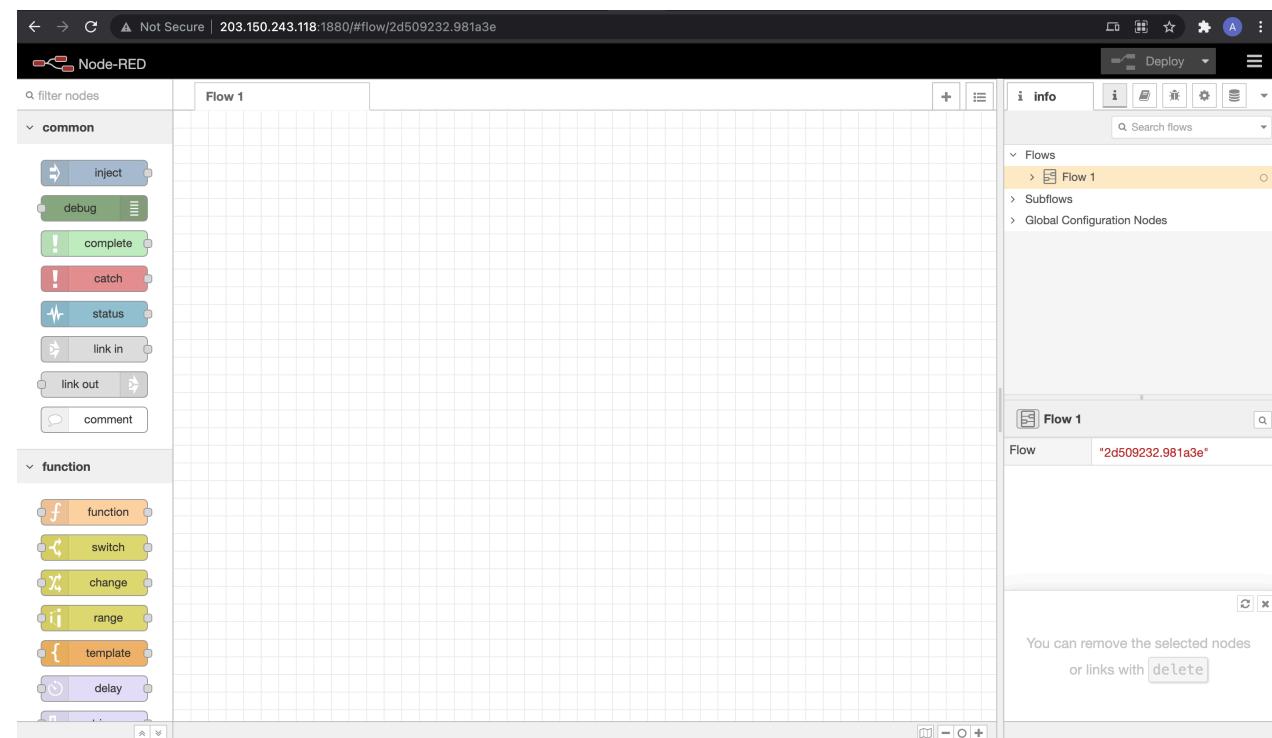
```
root@voip:~# nvm install 10.5  
Downloading and installing node v10.5.0...  
Downloading https://nodejs.org/dist/v10.5.0/node-v10.5.0-linux-x64.tar.xz...  
#####
Computing checksum with sha256sum  
Checksums matched!  
Now using node v10.5.0 (npm v6.1.0)  
Creating default alias: default -> 10.5 (-> v10.5.0)
```

```
root@voip:~# node -v  
v10.5.0  
root@voip:~# npm -v  
6.1.0
```

# Install node red on the server

```
npm install -g --unsafe-perm node-red
```

```
http://203.150.243.118:1880/
```



# Start node red at boot

```
root@voip:~# sudo npm install -g pm2
[.....] | fetchMetadata: sill resolveWithNewModule moment@2.29.1 checking
installable status
:
/usr/bin/pm2-runtime -> /usr/lib/node_modules/pm2/bin/pm2-runtime

+ pm2@5.1.0
added 180 packages from 203 contributors in 190.922s
```

```
root@voip:~# which node-red
/root/.nvm/versions/node/v10.5.0/bin/node-red
```

```
root@voip:~# pm2 start /root/.nvm/versions/node/v10.5.0/bin/node-red -- --v
[PM2] Spawning PM2 daemon with pm2_home=/root/.pm2
[PM2] PM2 Successfully daemonized
[PM2] Starting /root/.nvm/versions/node/v10.5.0/bin/node-red in fork_mode (1 instance)
[PM2] Done.
```

<b>id</b>	<b>name</b>	<b>mode</b>	<b>↳</b>	<b>status</b>	<b>cpu</b>	<b>memory</b>
<b>0</b>	node-red	<b>fork</b>	0	<b>online</b>	0%	22.3mb

```
root@voip:~# pm2 save
[PM2] Saving current process list...
[PM2] Successfully saved in /root/.pm2/dump.pm2
```

```
root@voip:~# pm2 startup
[PM2] Init System found: systemd
Platform systemd
Template

:
+
-----+
[PM2] Freeze a process list on reboot via:
$ pm2 save

[PM2] Remove init script via:
$ pm2 unstartup systemd
```

```
root@voip:~# pm2 list
```

<b>id</b>	<b>name</b>	<b>mode</b>	<b>↳</b>	<b>status</b>	<b>cpu</b>	<b>memory</b>
<b>0</b>	node-red	<b>fork</b>	0	<b>online</b>	0%	77.0mb

# Install MQTT Broker

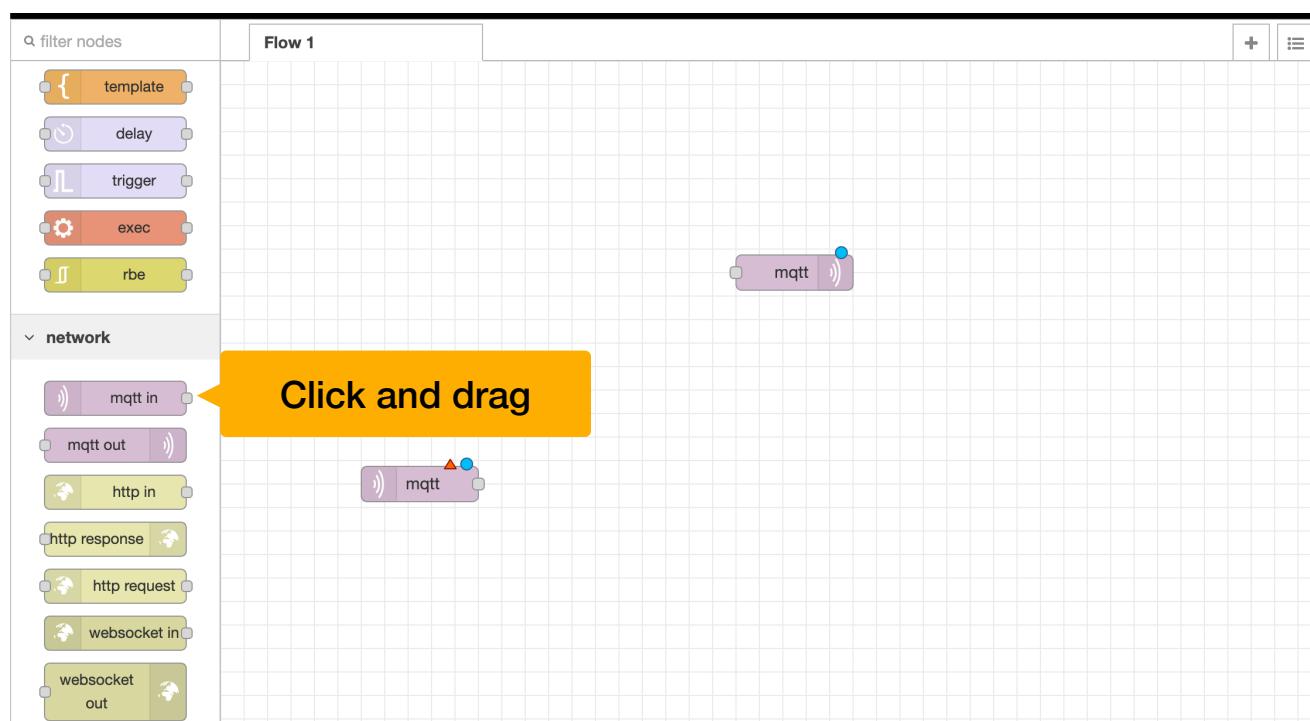
```
root@voip:~# sudo apt update -y && sudo apt install mosquitto mosquitto-clients -y

root@voip:~# sudo systemctl status mosquitto
● mosquitto.service - LSB: mosquitto MQTT v3.1 message broker
  Loaded: loaded (/etc/init.d/mosquitto; generated)
  Active: active (running) since Wed 2021-07-14 22:52:48 +07; 2min 32s ago
    Docs: man:systemd-sysv-generator(8)
   Tasks: 1 (limit: 1151)
  CGroup: /system.slice/mosquitto.service
          └─5133 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

Jul 14 22:52:48 voip systemd[1]: Starting LSB: mosquitto MQTT v3.1 message broker...
Jul 14 22:52:48 voip mosquitto[5127]: * Starting network daemon: mosquitto
Jul 14 22:52:48 voip mosquitto[5127]: ...done.
Jul 14 22:52:48 voip systemd[1]: Started LSB: mosquitto MQTT v3.1 message broker.
```

# Test MQTT with node red

## Create MQTT nodes



Edit mqtt out node > Add new mqtt-broker config node

Enabled     0 nodes use this config     On all flows

**Properties**

Name: server1

**Connection**

Server: 203.150.243.118    Port: 1883

Use TLS

Protocol: MQTT V3.1.1

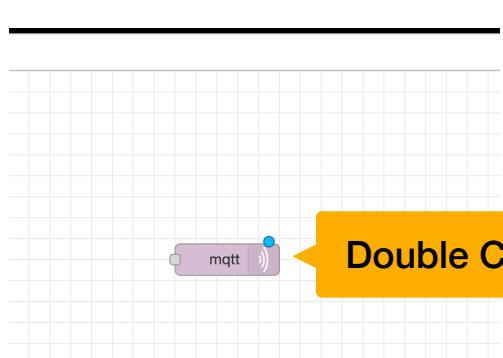
Client ID: Leave blank for auto generated

Keep Alive: 60

Session:  Use clean session

# Test MQTT with node red

## Config MQTT nodes



Double Click

Edit mqtt out node

Properties

Server: Add new mqtt-broker...

Topic: Topic

QoS:  Retain:

Name: Name

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

Enabled:

Click

Edit mqtt out node > Add new mqtt-broker config node

Properties

Name: server2

Connection Security Messages

Server: 203.150.243.118 Port: 1883

Use TLS:

Protocol: MQTT V3.1.1

Client ID: Leave blank for auto generated

Keep Alive: 60

Session: Use clean session:

Enabled:  0 nodes use this config On all flows

Click

Edit mqtt out node

Properties

Server: server2

Topic: /temp

QoS:  Retain:

Name: Name

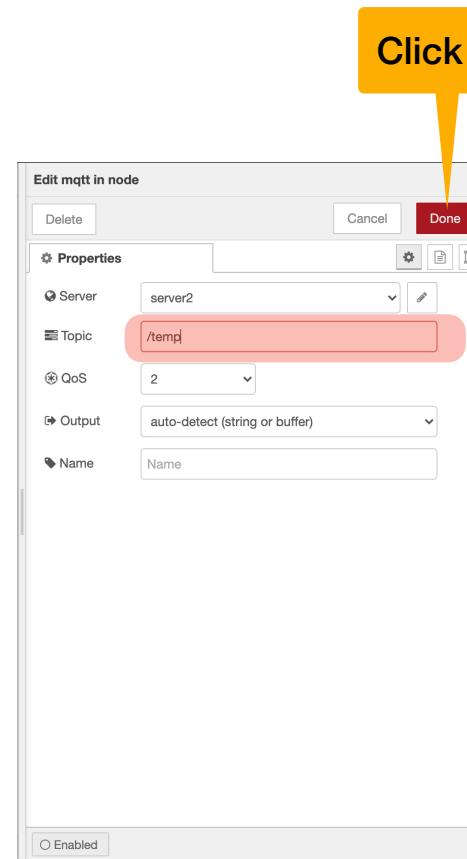
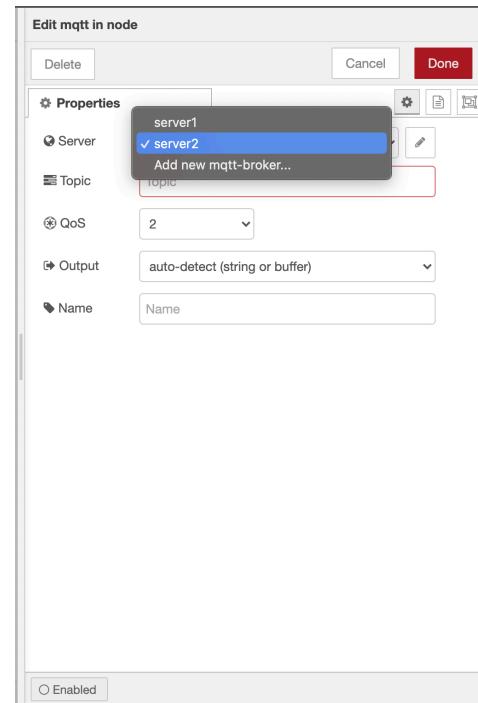
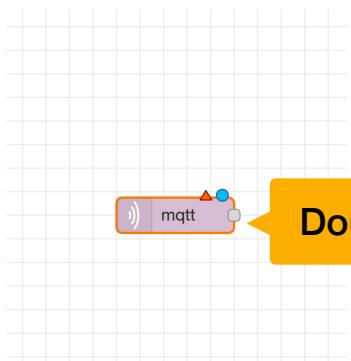
Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

Enabled:

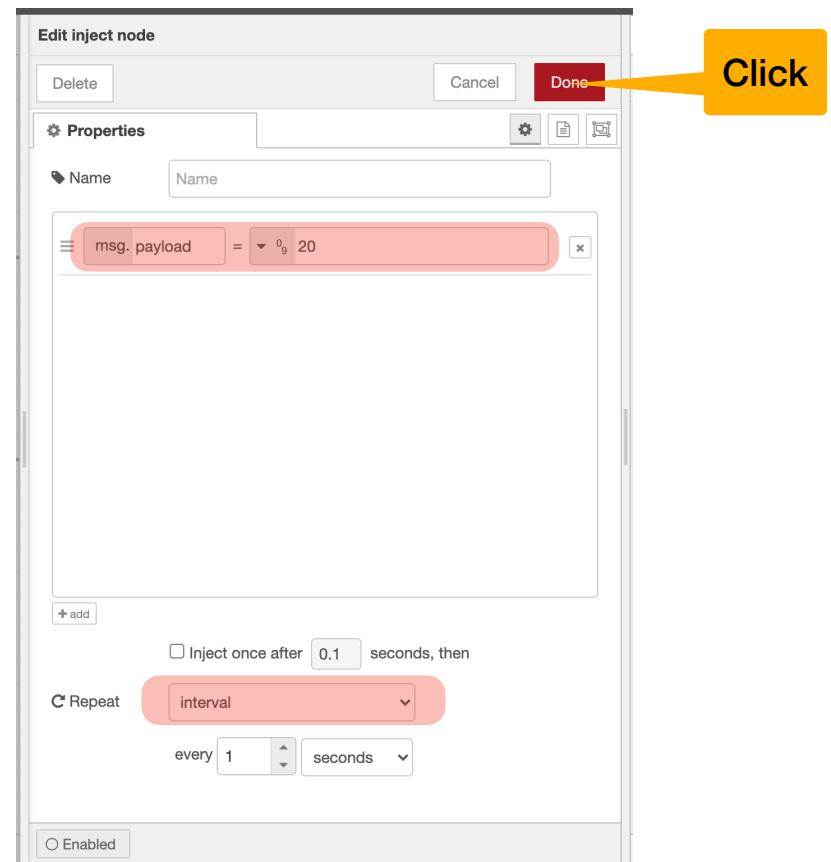
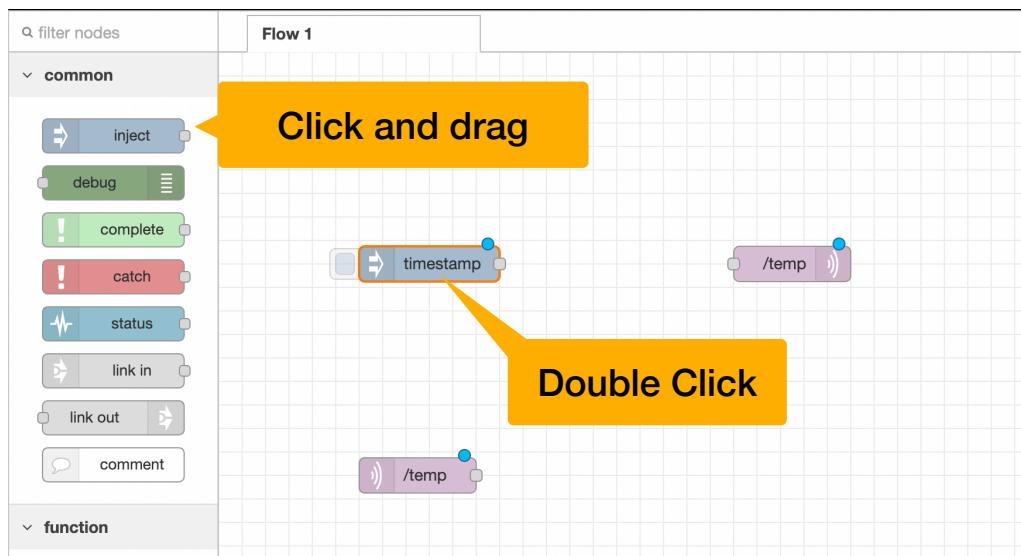
Click

# Test MQTT with node red

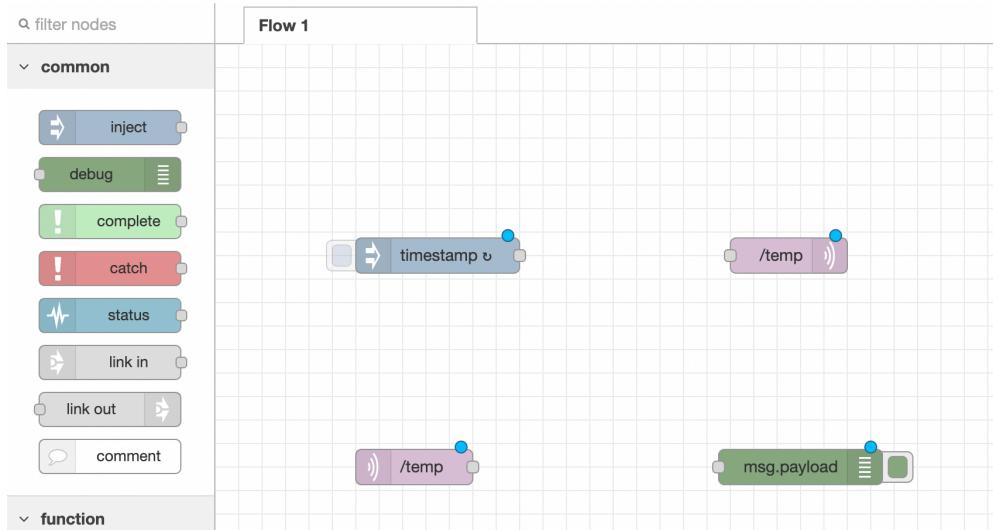
## Config MQTT nodes



# Create message source

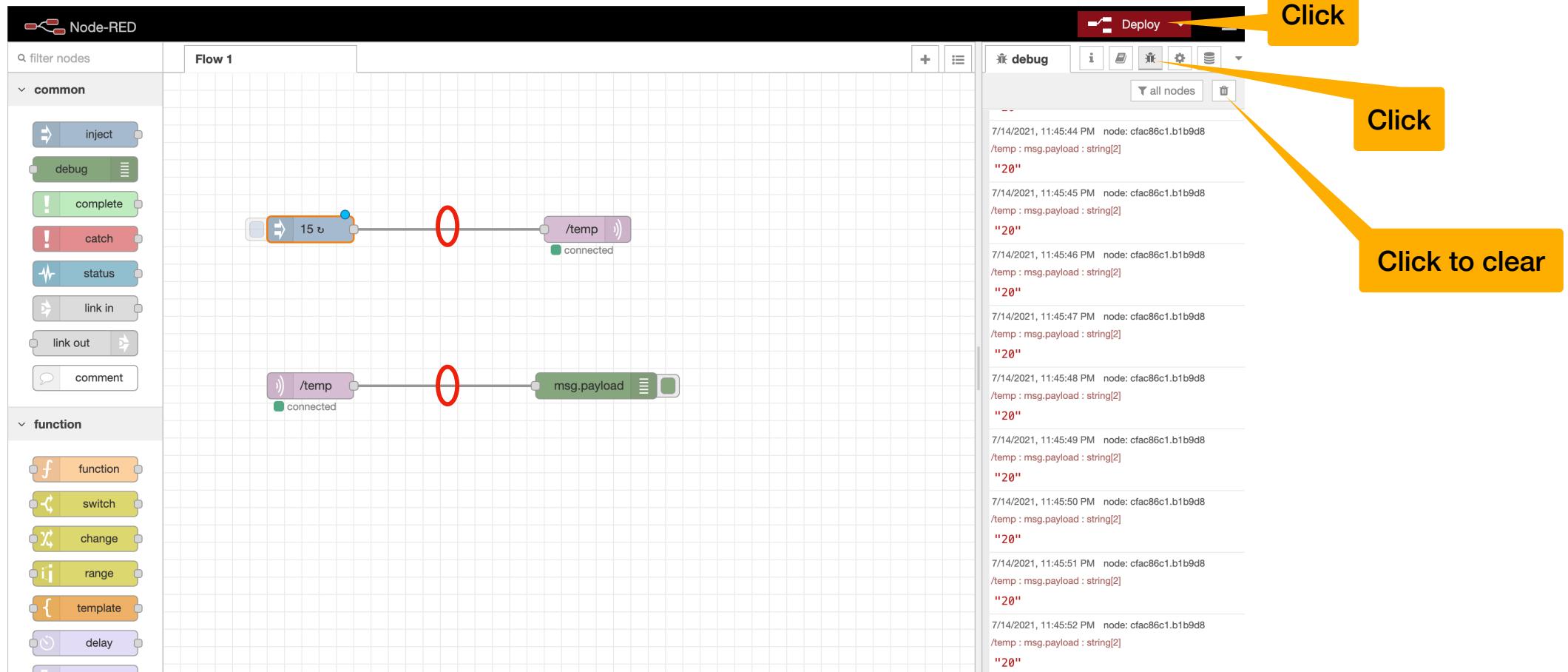


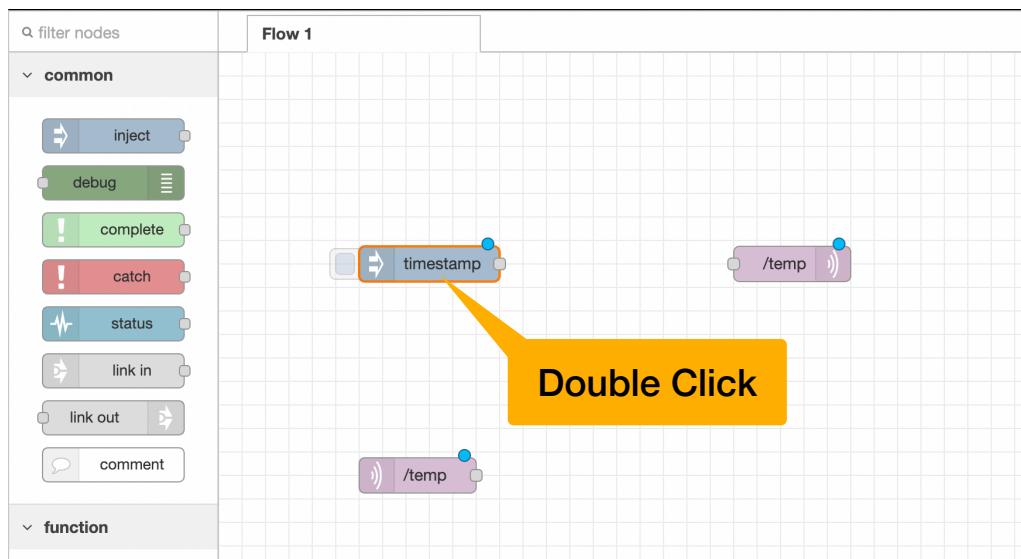
# Create message debug



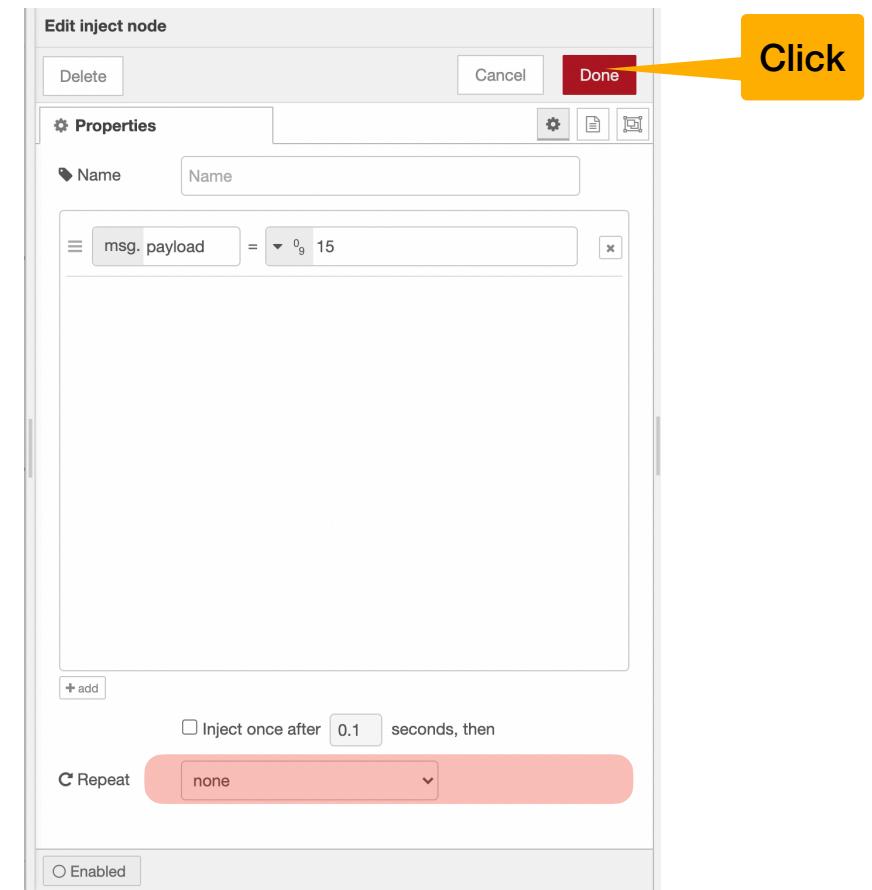
# Connect nodes

## Connect nodes and deploy



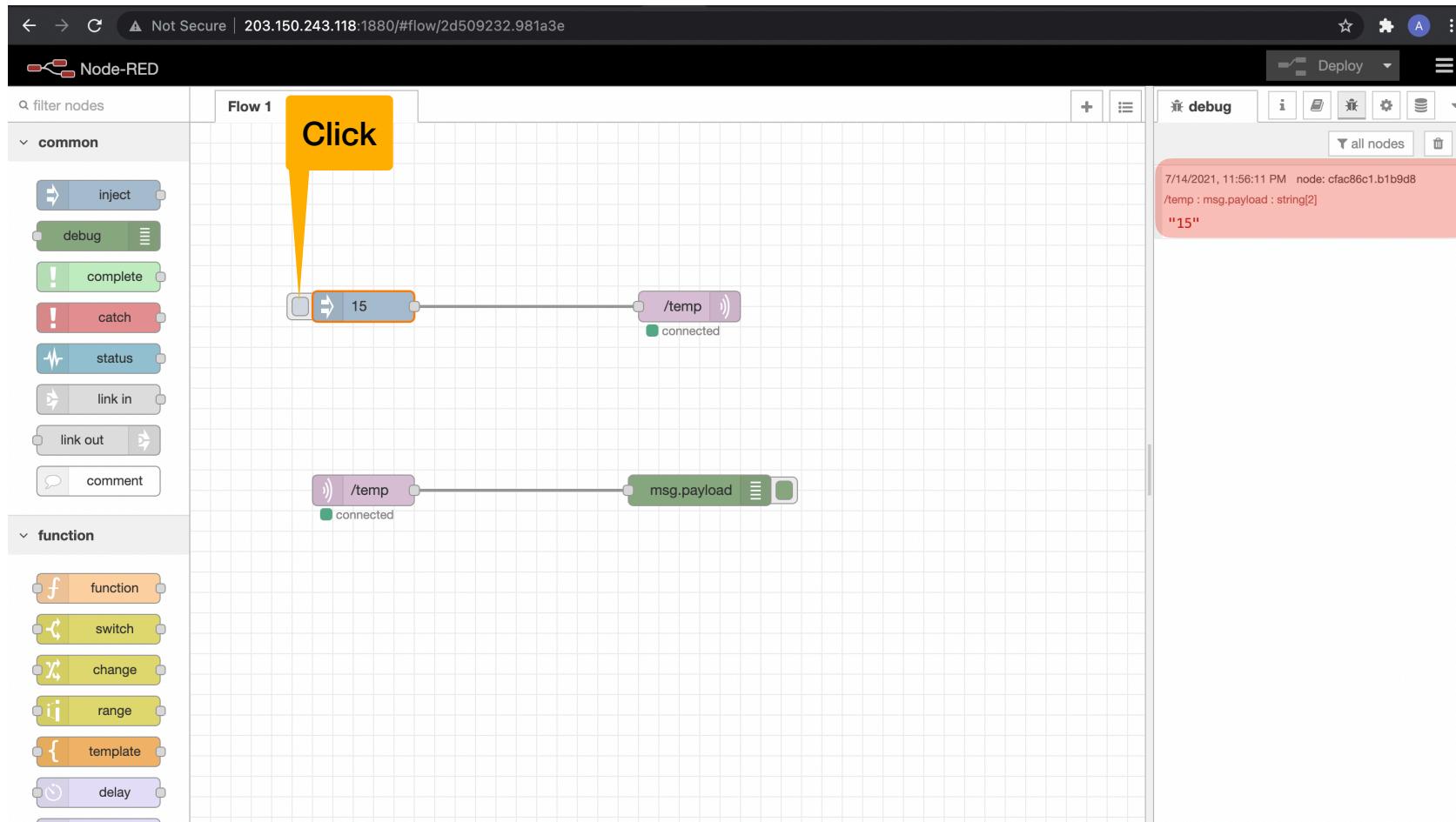


Double Click



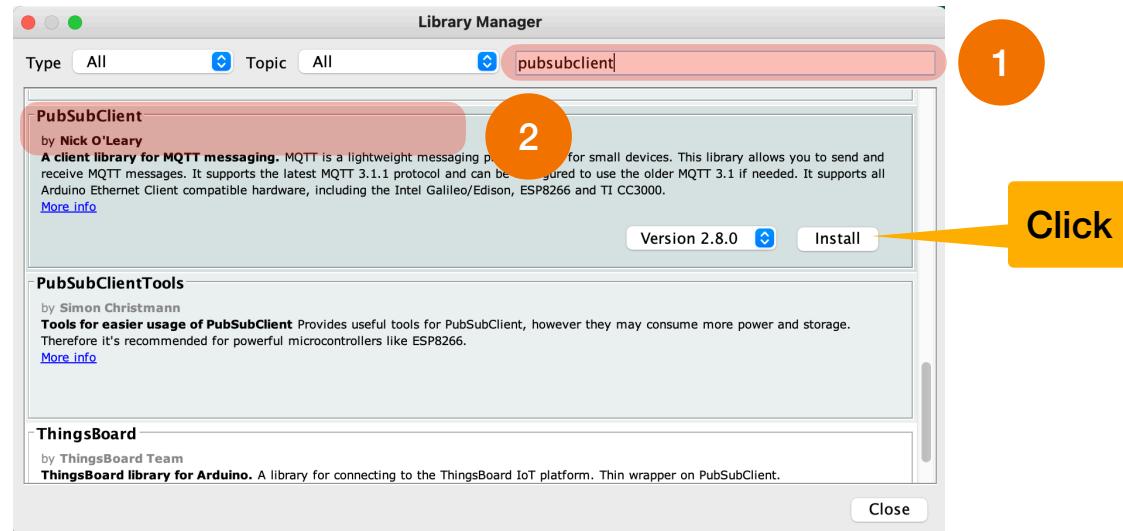
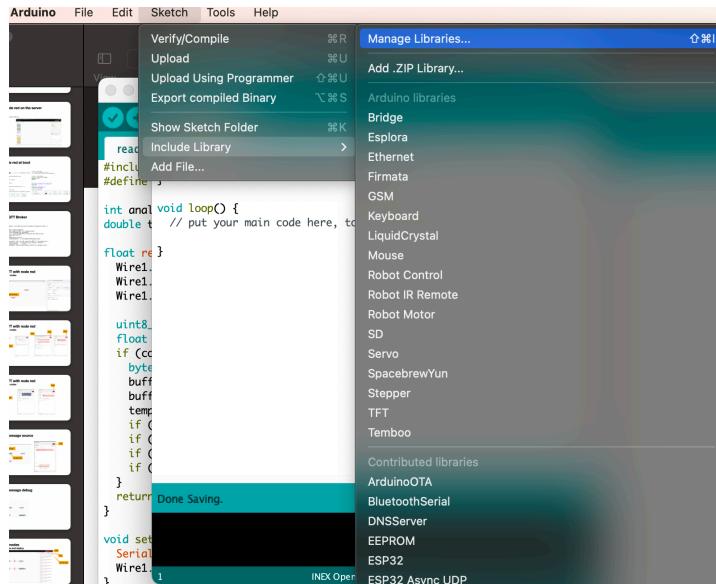
Click

# Test click to inject



# Using ESP32 with our server

## Install Arduino pubsubclient



# **Using ESP32 with our server**

## **ESP32 MQTT connection to Mosquito**

- Use ESP32 connect to wifi
- Once WiFi is connected, we create a MQTT client on ESP32.
- Then, we connect the MQTT client to our Mosquito server
- We can then create topics for publish and subscribe

# Using ESP32 with our server

## Connect to WiFi and MQTT broker : esp32mqtt.ino

```
void setup_wifi() {  
  
    delay(10);  
    // We start by connecting to a WiFi network  
    Serial.println();  
    Serial.print("Connecting to ");  
    Serial.println(ssid);  
  
    WiFi.begin(ssid, password);  
  
    while (WiFi.status() != WL_CONNECTED) {  
        digitalWrite(led_pin_1, HIGH);  
        digitalWrite(led_pin_2, HIGH);  
        delay(500);  
        digitalWrite(led_pin_2, LOW);  
        digitalWrite(led_pin_1, LOW);  
        delay(500);  
  
    }  
  
    digitalWrite(led_pin_1, LOW);  
    Serial.println("Connected to the WiFi  
network");  
  
    Serial.println("");  
    Serial.println("WiFi connected");  
    Serial.println("IP address: ");  
    Serial.println(WiFi.localIP());  
}
```

```
const char* ssid = "CAKE";  
const char* password = "0891300633";  
const char* mqtt_server = "203.150.243.118";
```

Put your IP here

```
void setup() {  
    // Configure pin  
    pinMode(led_pin_1, OUTPUT);  
    pinMode(led_pin_2, OUTPUT);  
    Serial.begin(115200);  
    setup_wifi();  
    client.setServer(mqtt_server, 1883);  
  
    if (!client.connected()) {  
  
        digitalWrite(led_pin_2, HIGH);  
        delay(500);  
        digitalWrite(led_pin_2, LOW);  
        delay(500);  
    }  
  
    digitalWrite(led_pin_2, LOW);  
    client.setCallback(callback);  
}
```

# Using ESP32 with our server

Publish and subscribe to topics : esp32mqtt.ino

```
void setup() {
    // Configure pin
    pinMode(led_pin_1, OUTPUT);
    pinMode(led_pin_2, OUTPUT);
    Serial.begin(115200);
    setup_wifi();
    client.setServer(mqtt_server, 1883);

    if (!client.connected()) {

        digitalWrite(led_pin_2, HIGH);
        delay(500);
        digitalWrite(led_pin_2, LOW);
        delay(500);
    }

    digitalWrite(led_pin_2, LOW);
    client.setCallback(callback)
}
```

```
void callback(char* topic, byte* payload, unsigned int length) {
    Serial.print("Message arrived [");
    Serial.print(topic);
    Serial.print("] ");
    for (int i = 0; i < length; i++) {
        Serial.print((char)payload[i]);
    }
    Serial.println();
}
```

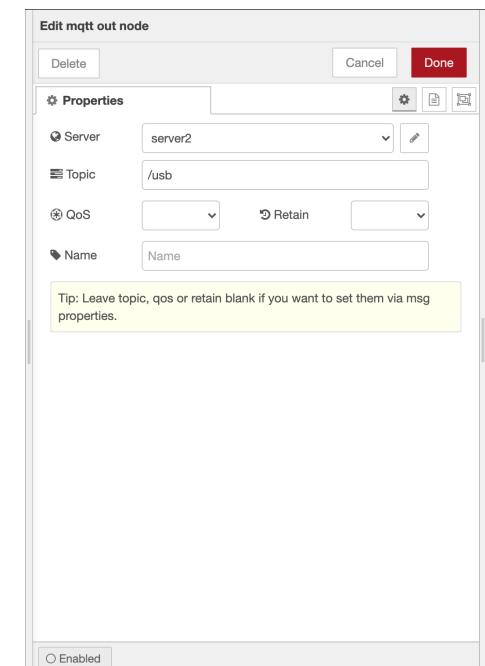
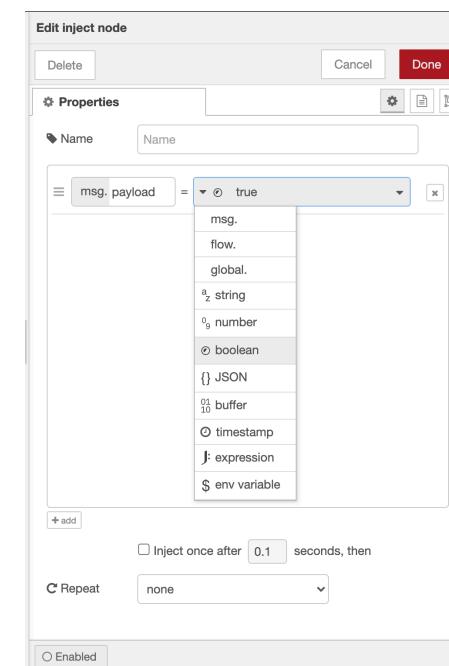
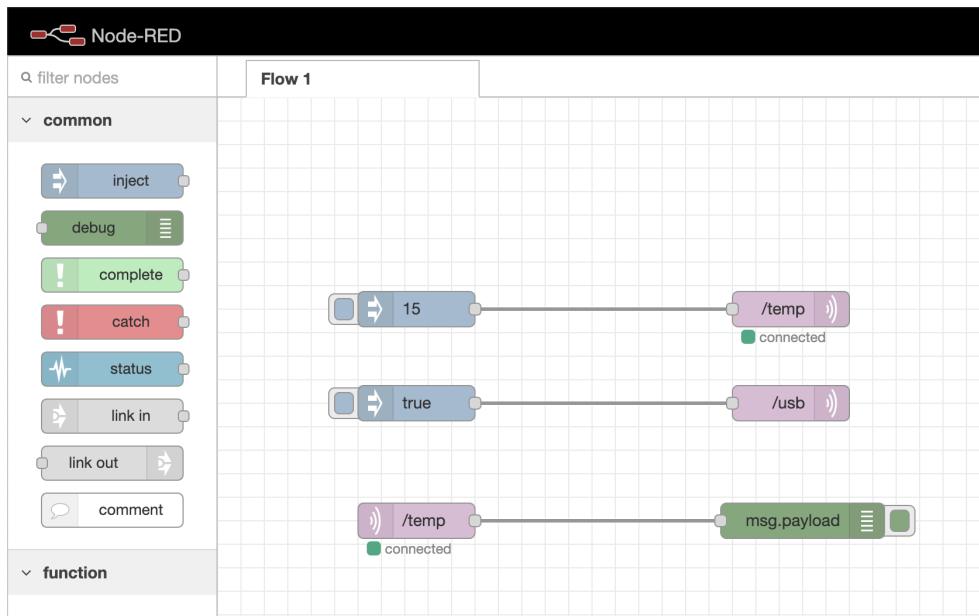
```
void reconnect() {
    // Loop until we're reconnected
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");
        // Attempt to connect
        if (client.connect("ESP32Client")) {
            Serial.println("connected");
            // Once connected, publish an announcement
            client.publish("/temp", 0);
            // ... and resubscribe
            client.subscribe("/usb");
        } else {
            Serial.print("failed, rc=");
            Serial.print(client.state());
            Serial.println(" try again in 5 seconds");
            // Wait 5 seconds before retrying
            delay(5000);
        }
    }
}
```

**Publish a topic**

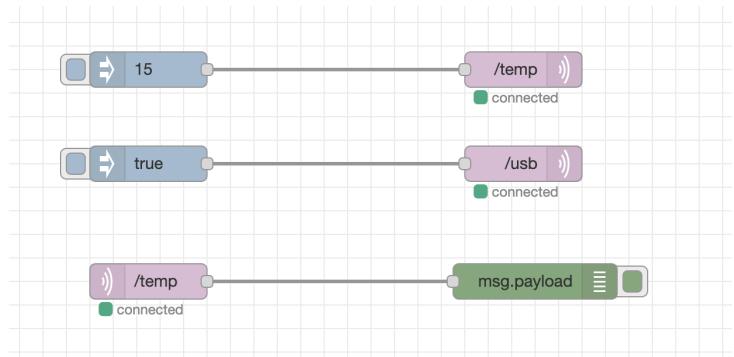
**Subscribe to a topic**

# Add one more publisher to node red

## It will be used for turn on/off an USB device



# Test publish and subscribe

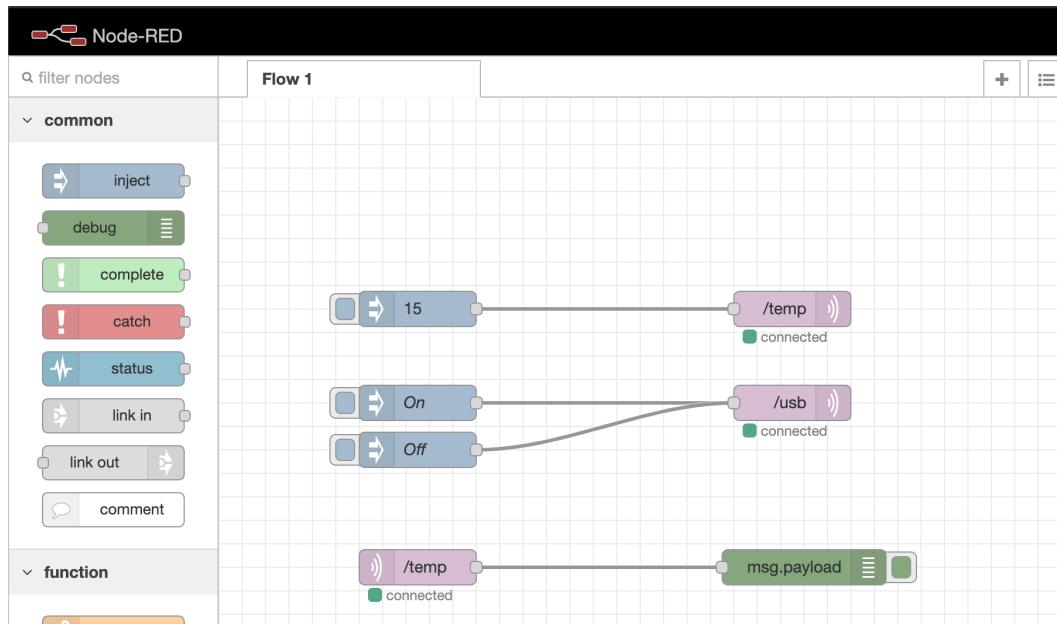


```
Publish message: 464
Publish message: 465
Publish message: 466
Publish message: 467
Publish message: 468
Message arrived [/usb] true
Publish message: 469
Publish message: 470
Publish message: 471
Publish message: 472
Message arrived [/usb] true
Publish message: 473
Publish message: 474
Publish message: 475
Publish message: 476
Publish message: 477
Publish message: 478
Publish message: 479
Publish message: 480
Publish message: 481
Publish message: 482
Publish message: 483
```

The terminal window displays a log of published messages. The log shows a sequence of publish messages followed by two 'Message arrived' events for the '/usb' topic, which triggered further publish messages. The publish messages are numbered sequentially from 464 to 483.

# Using an USB device with MQTT

## Node red flow



Three MQTT configuration dialog boxes are shown:

- Edit inject node (On):** Properties: Name = On, msg.payload = %\_1. Options: Inject once after 0.1 seconds, then. Repeat: none. Enabled: checked.
- Edit inject node (Off):** Properties: Name = Off, msg.payload = %\_0. Options: Inject once after 0.1 seconds, then. Repeat: none. Enabled: checked.
- Edit mqtt out node (/usb):** Properties: Server = server2, Topic = /usb, QoS = 2, Retain: checked. Name: Name. Options: Leave topic, qos or retain blank if you want to set them via msg.properties. Enabled: checked.

# Using an USB device with MQTT

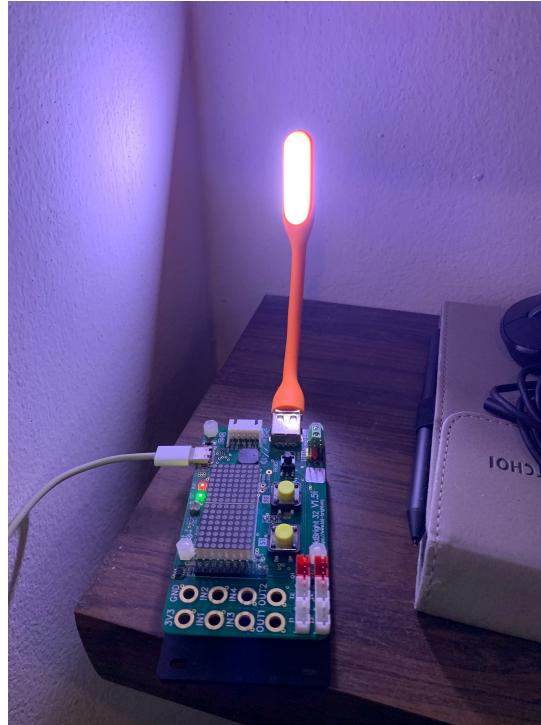
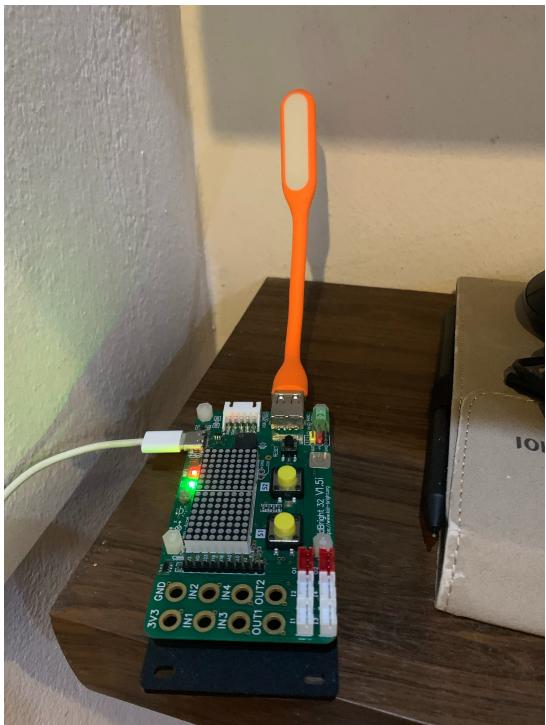
Modify ESP32 code : esp32mqttTempAndUSB.ino

```
void callback(char* topic, byte* payload, unsigned int length) {  
    Serial.print("Message arrived [");  
    Serial.print(topic);  
    Serial.print("] ");  
    if(strcmp((char *) topic, "/usb") == 0){  
        payload[length] = '\0';  
        String s = String((char*)payload);  
        int f = s.toInt();  
        for (int i = 0; i < length; i++) {  
            Serial.print((char)payload[i]);  
        }  
        if(f == 1){  
            digitalWrite(usb_pin, LOW);  
            Serial.print("1");  
        }else {  
            digitalWrite(usb_pin, HIGH);  
            Serial.print("0");  
        }  
    }  
    Serial.println();  
}
```

```
// Pins  
static const int led_pin_1 = 2; //WiFi  
static const int led_pin_2 = 12; //IoT  
static const int usb_pin = 25; //USB
```

```
void setup() {  
    // Configure pin  
    pinMode(led_pin_1, OUTPUT);  
    pinMode(led_pin_2, OUTPUT);  
    pinMode(usb_pin, OUTPUT);  
  
    digitalWrite(usb_pin, HIGH);
```

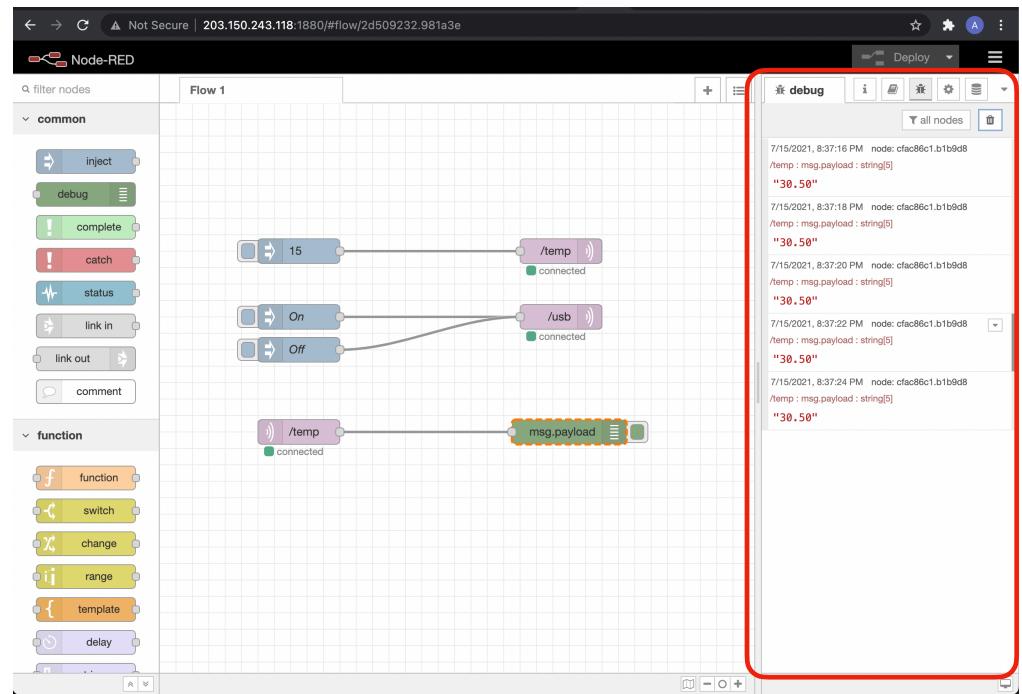
# Connect Hardware and test



# Publish temperature to Node red

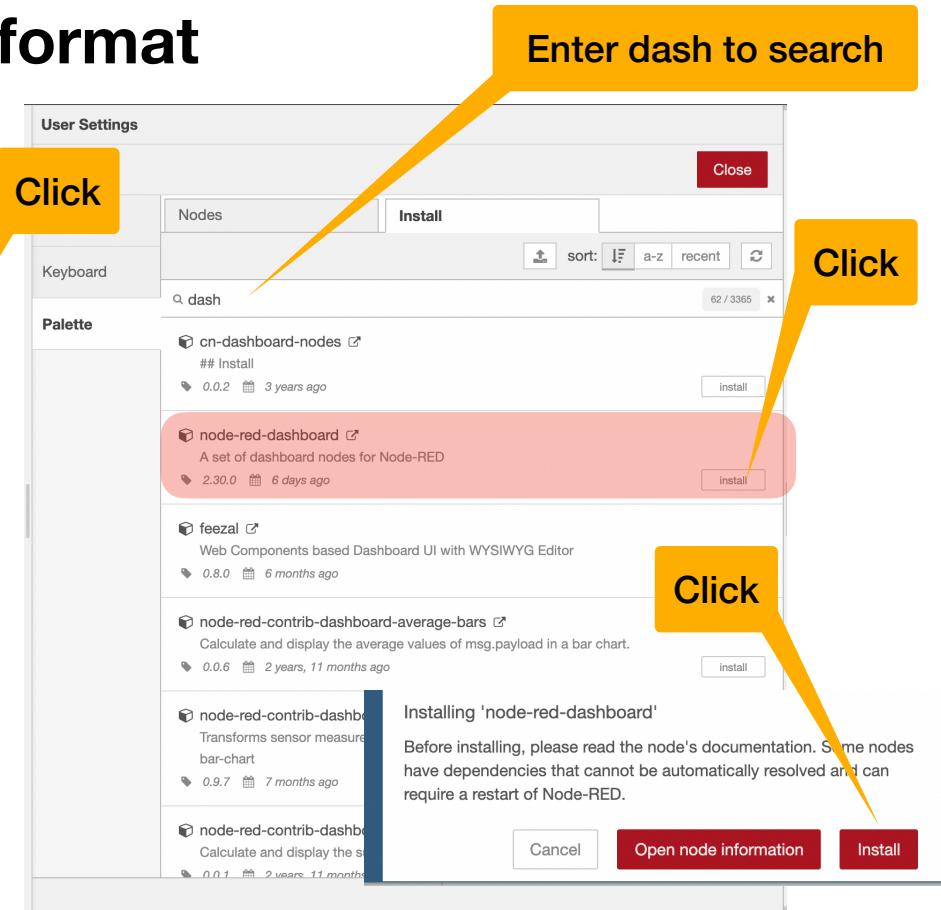
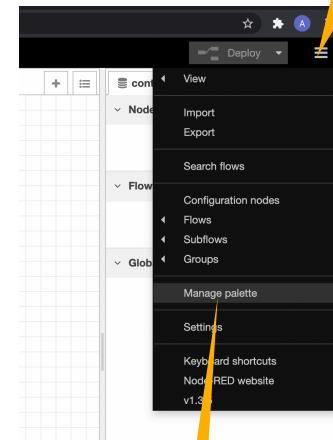
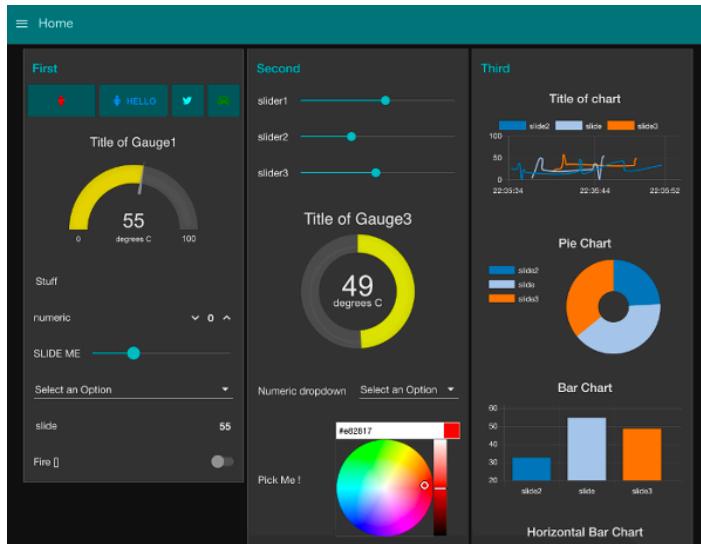
## esp32mqttTempAndUSB.ino

```
void loop() {  
  
    if (!client.connected()) {  
        reconnect();  
    }  
    client.loop();  
  
    long now = millis();  
    if (now - lastMsg > 2000) {  
        lastMsg = now;  
        ++value;  
        //snprintf (msg, 75, value);  
        sprintf(msg,"%2.2f",readTemperature());  
        //Serial.print("Publish message: ");  
        //Serial.println(msg);  
        client.publish("/temp", msg);  
    }  
}
```



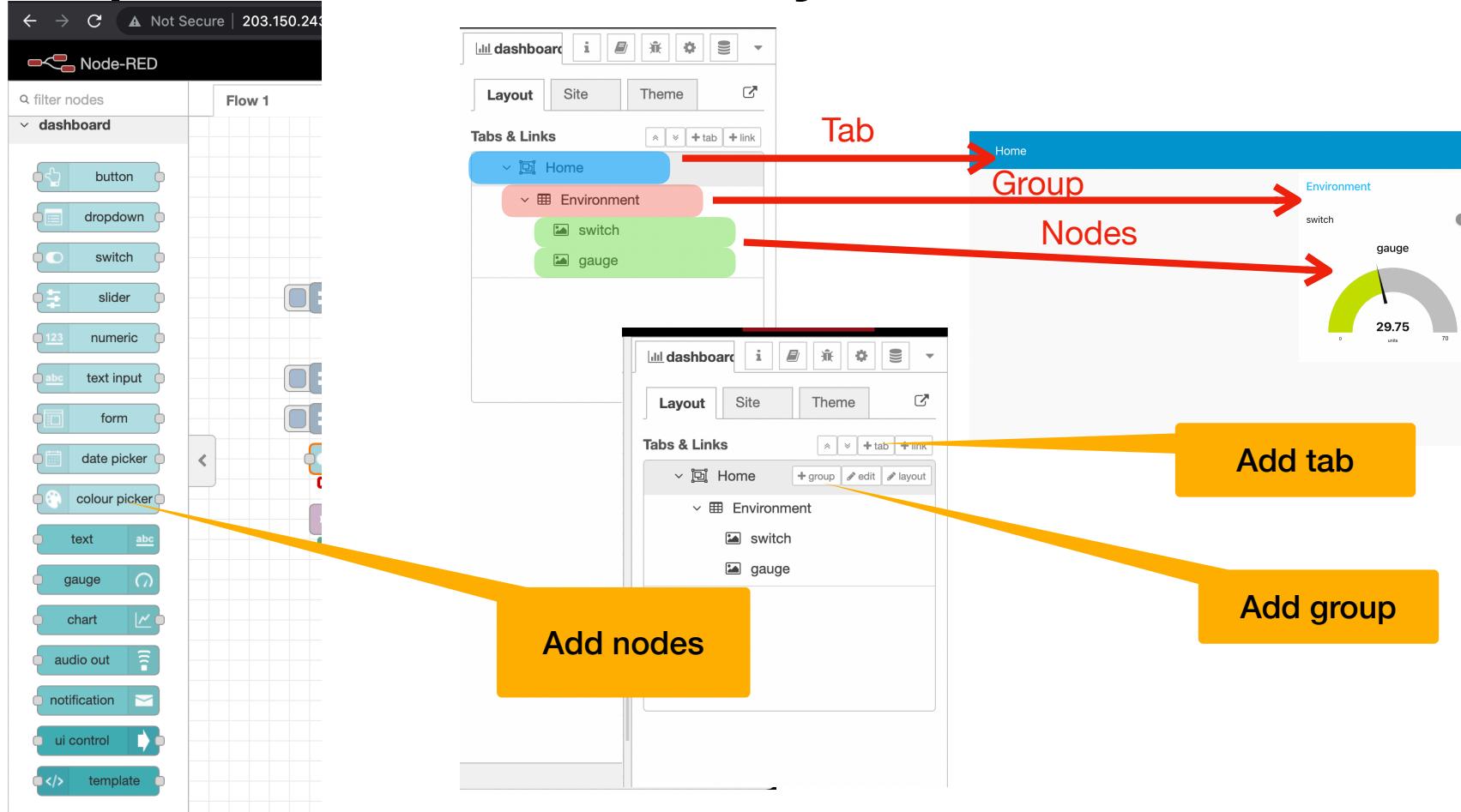
# Node red dashboard

Dashboard can display data in a nice format



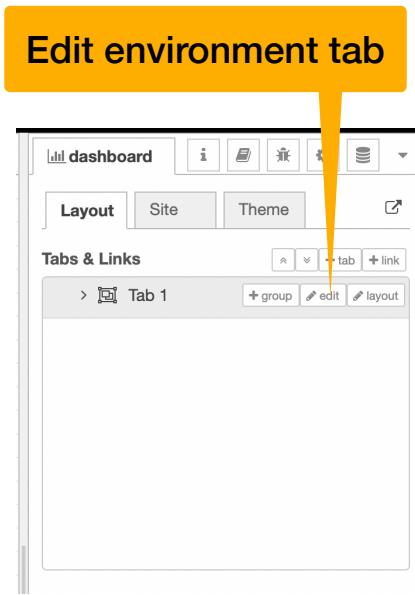
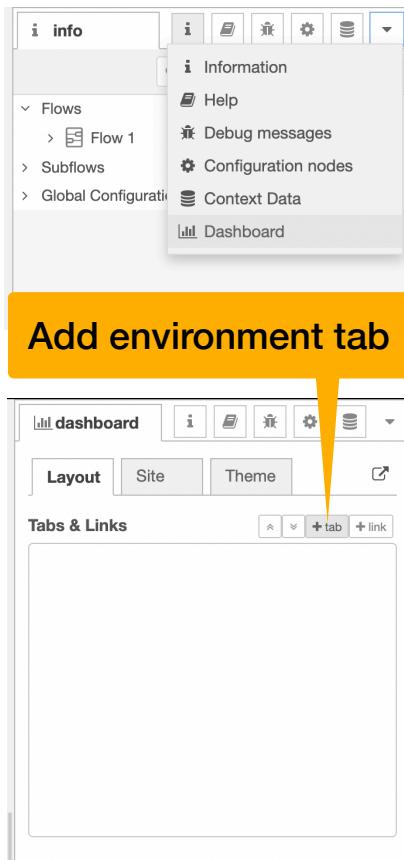
# Check dashboard component

## Components and dashboard layout



# Create the first dash board

## Create an Environment tab



Click

Edit dashboard tab node

Delete Cancel Update

Properties

Name: Environment

Icon: dashboard

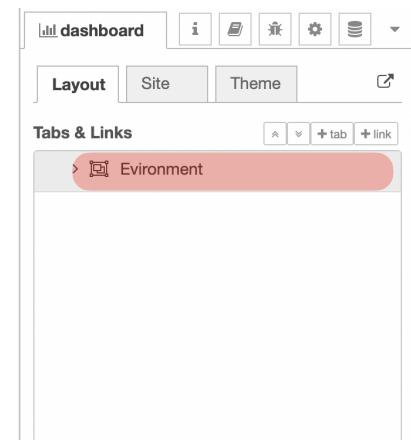
State: Enabled

Nav. Menu: Visible

The **Icon** field can be either a [Material Design icon](#) (e.g. 'check', 'close') or a [Font Awesome icon](#) (e.g. 'fa-fire'), or a [Weather icon](#) (e.g. 'wi-wu-sunny').

You can use the full set of google material icons if you add 'mi-' to the icon name. e.g. 'mi-video\_game\_asset'.

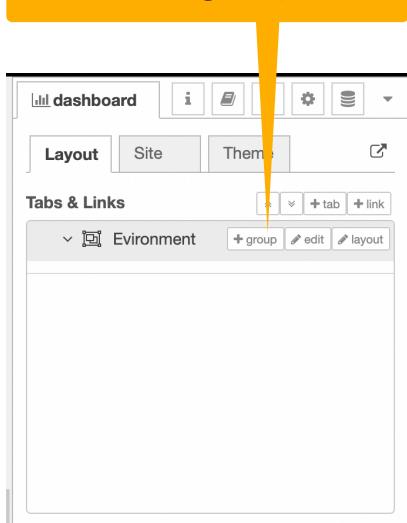
Enabled 0 nodes use this config On all flows



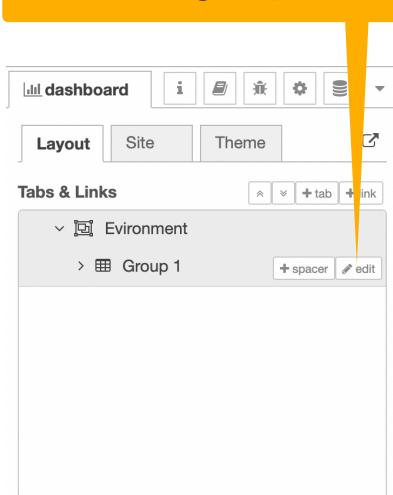
# Create the first dash board

## Create KidBright 1 group

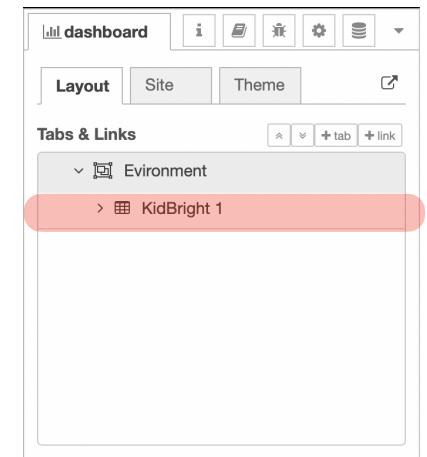
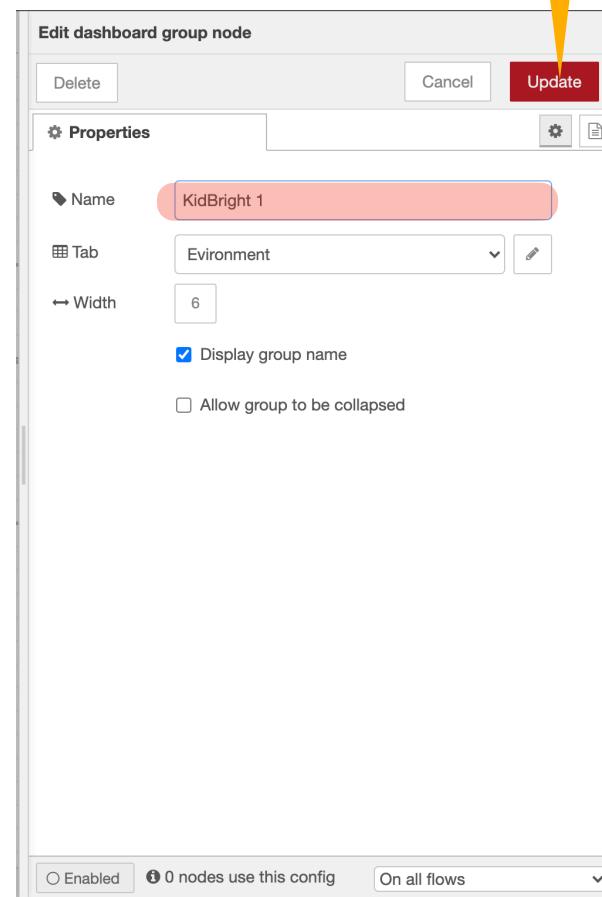
Add group



Edit group

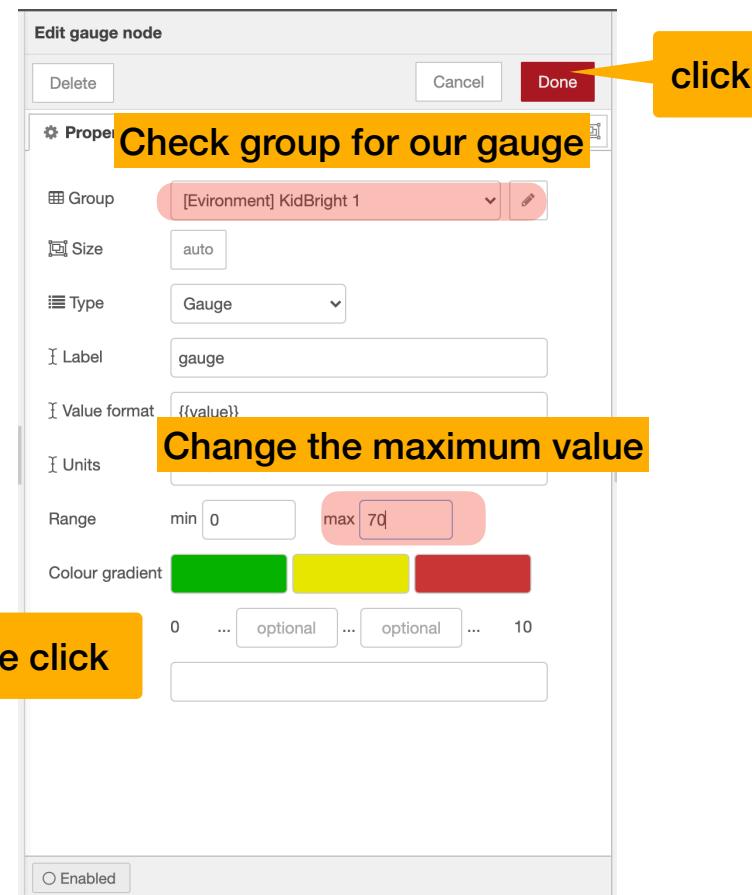
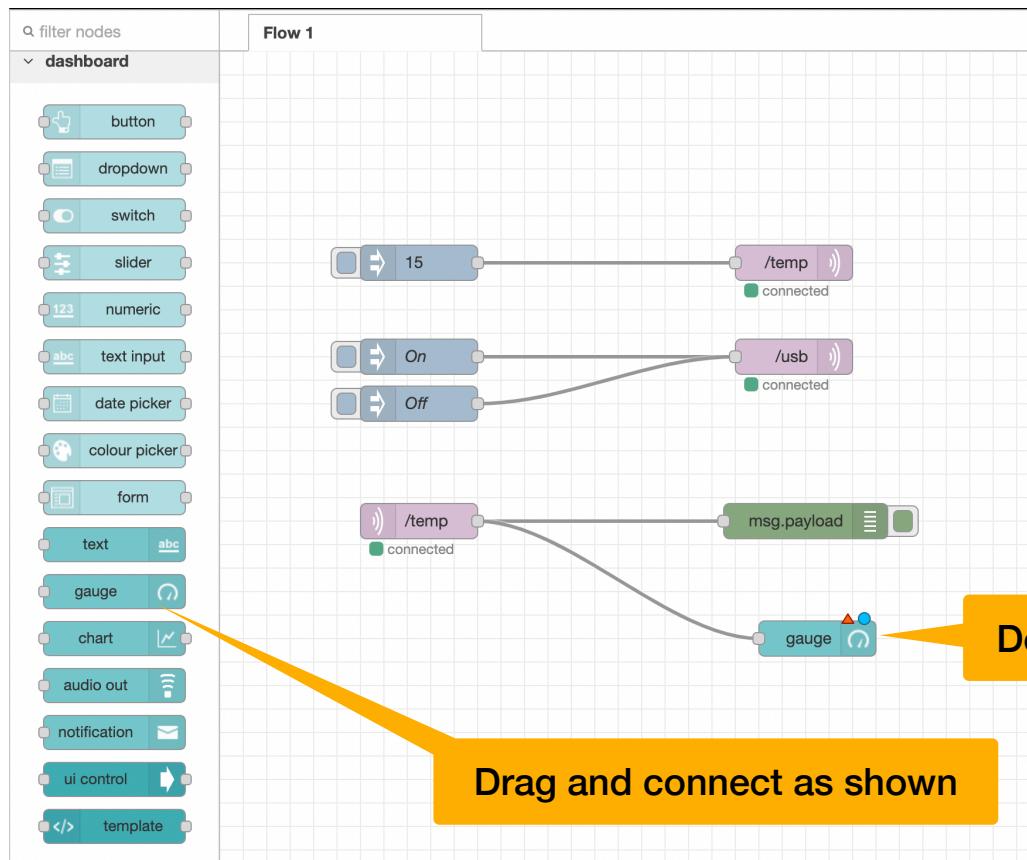


Click



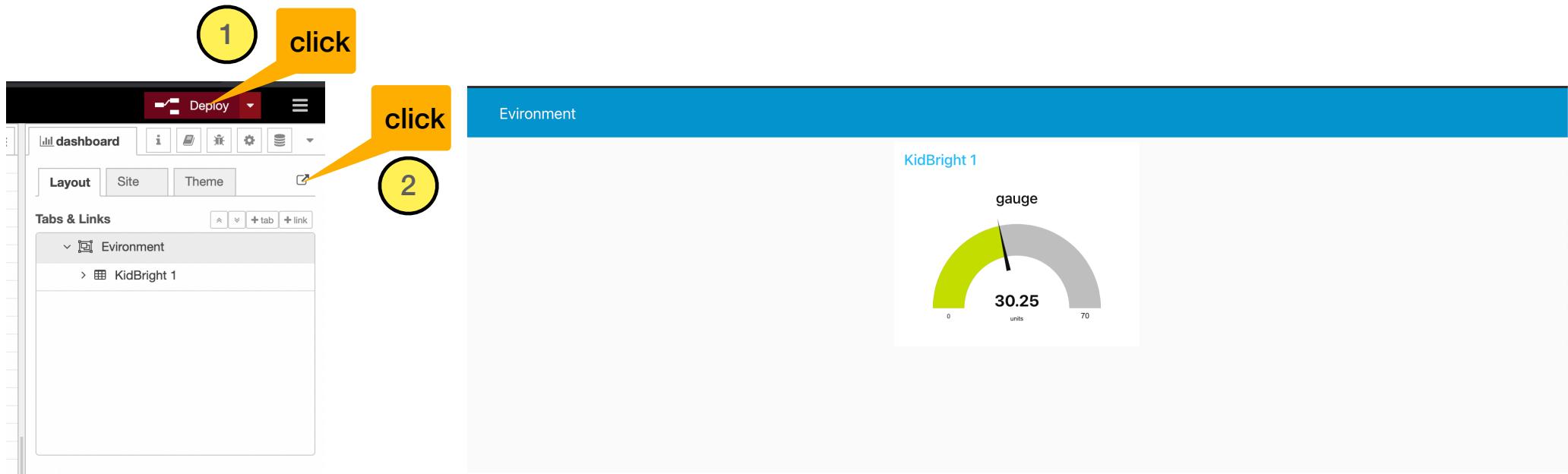
# Create the first dash board

## Using Gauge node to display temperature



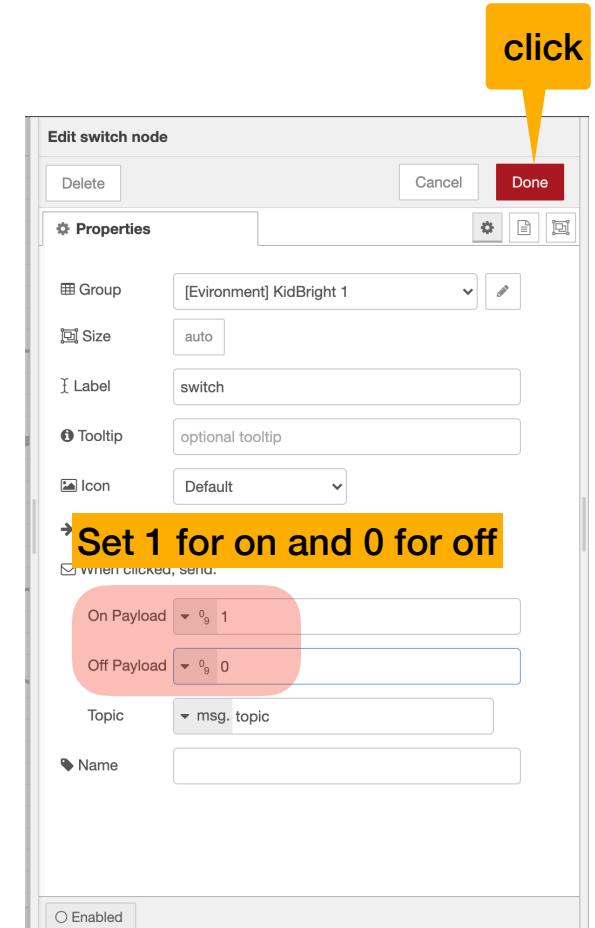
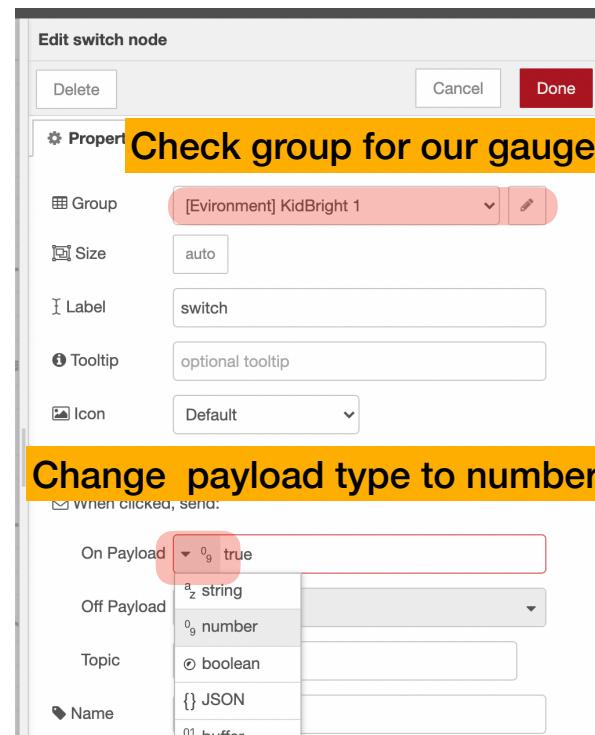
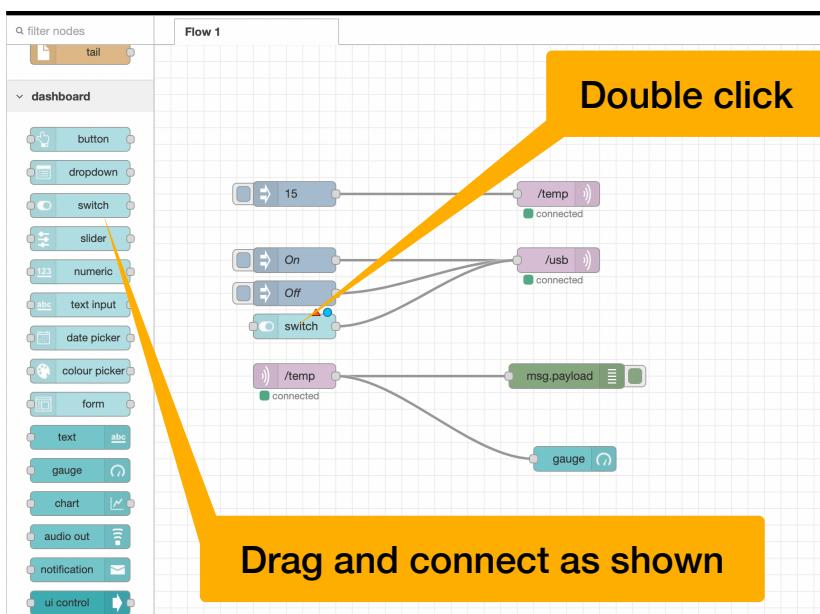
# Create the first dash board

Deploy our dashboard



# Add a switch to control USB light

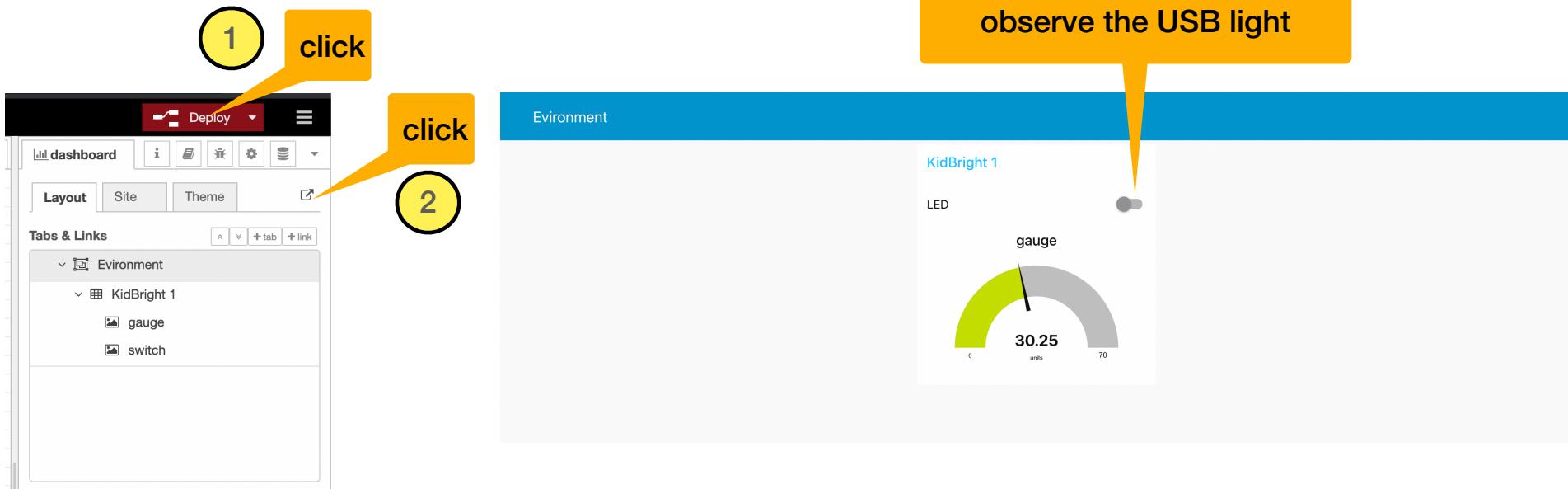
## Add dashboard switch



click

# Add a switch to control USB light

## Deploy and test switch



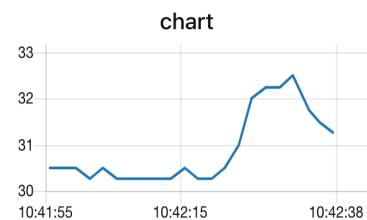
Due to bug in switch, you may need to set the value 0 and 1 many times until there is no error when deploy button is click. You can reconfig switch by double click on the switch icon in flow

# Exercise

# Display temperature using graph

KidBright 1

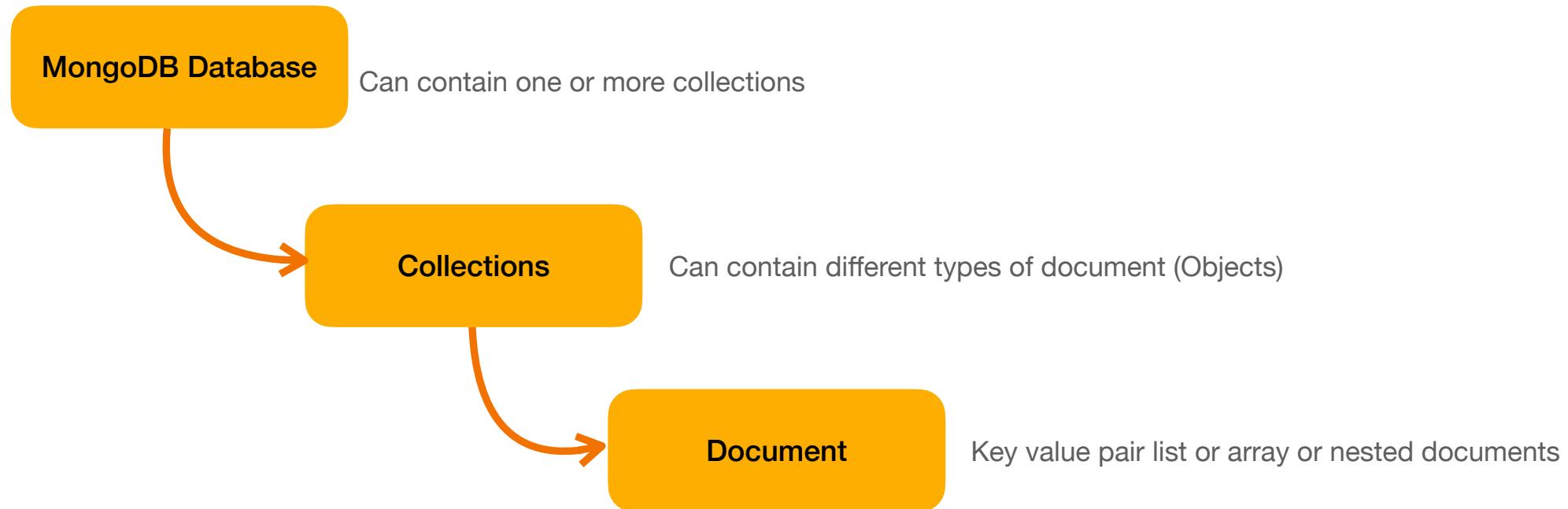
LED



gauge



# MongoDB : NoSQL database



# Install and config mongoldb

Link to command lines <https://github.com/fllay/iotTraining/wiki/MongoDB-installation>

```
sudo apt install -y mongodb
```

```
sudo systemctl status mongodb
```

```
root@voip:~# sudo systemctl status mongodb
● mongodb.service - An object/document-oriented database
  Loaded: loaded (/lib/systemd/system/mongodb.service; enabled; vendor preset: enabled)
  Active: active (running) since Fri 2021-07-16 11:30:37 +07; 31s ago
    Docs: man:mongod(1)
 Main PID: 22049 (mongod)
   Tasks: 23 (limit: 1151)
  CGroup: /system.slice/mongodb.service
          └─22049 /usr/bin/mongod --unixSocketPrefix=/run/mongodb --config /etc/mongodb.conf
```

```
Jul 16 11:30:37 voip systemd[1]: Started An object/document-oriented database.
```

```
sed "s;^#[\n ]*\!(bind_ip *= *\\)\!(.*\\);# \1\\2\\n\\10.0.0.0;" "/etc/mongodb.conf" > /etc/mongodb.conf.new
```

```
mv /etc/mongodb.conf.new /etc/mongodb.conf
```

```
systemctl restart mongodb
```

# Download and install MongoDB Compass

## MongoDB Compass allows database monitoring from PC

The screenshot shows the MongoDB website with the main navigation bar at the top. The navigation bar includes links for Cloud, Software, Pricing, Learn, Solutions, Docs, a search icon, Contact, Sign In, and a prominent green 'Try Free' button.

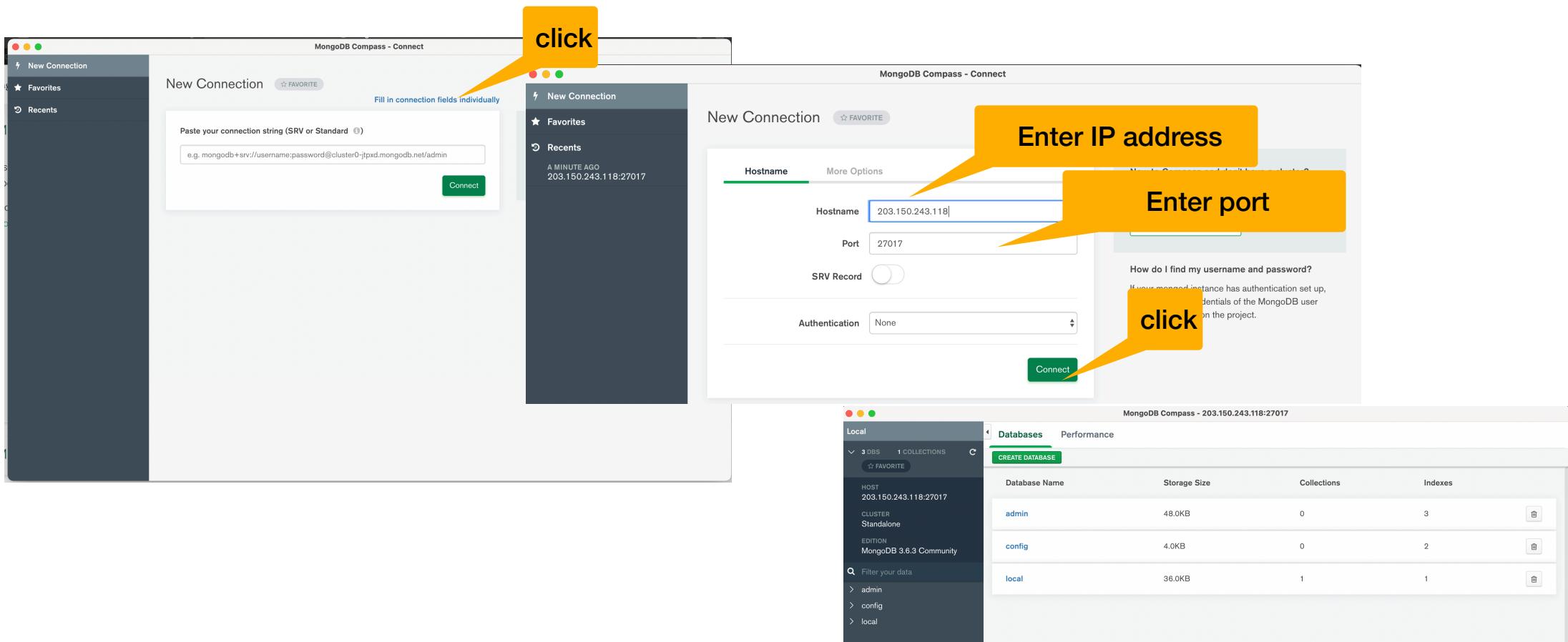
The main content area features a large heading 'MongoDB Compass' with a subtext: 'The easiest way to explore and manipulate your MongoDB collection'. Below this is a 'Try it now' button.

A central section titled 'MongoDB Compass' provides a brief overview: 'As the GUI for MongoDB, MongoDB Compass allows you to make smarter decisions about document structure, querying, indexing, document validation, and more. Commercial subscriptions include technical support for MongoDB Compass.' It also mentions that Compass is available in several versions: 'Compass', 'Readonly Edition', and 'Isolated Edition'.

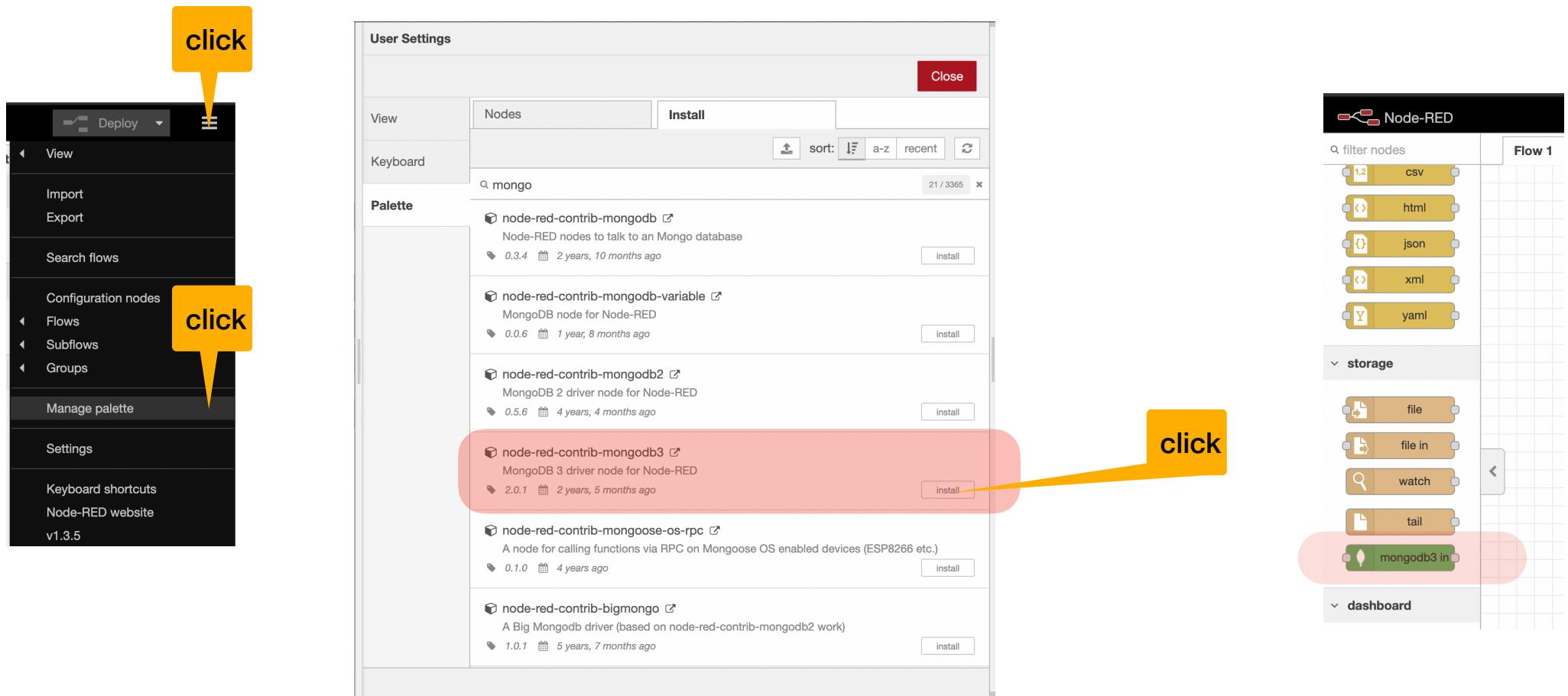
To the right, there's a sidebar titled 'Available Downloads' with dropdown menus for Version (1.28.1 (Stable)), Platform (OS X 64-bit (10.10+)), and Package (dmg). A large green 'Download' button is located here, along with a 'Copy Link' option.

At the bottom right, there are links for 'Documentation' and 'Archived releases'.

# Connect MongoDB Compass to database



# Install node-red MongoDB



# Create node-red flow

## Put a function block

The screenshot shows the Node-RED interface with a flow named "Flow 1". The flow consists of several nodes: an "15" node, two "On" and "Off" switch nodes, an "LED" node, a "/temp" sensor node, a "msg.payload" node, a "gauge" node, and a "chart" node. A "Double click" callout points to a "function" node. The "function" node's properties are being edited in a modal window titled "Edit function node". The "Properties" tab is selected, and the "On Message" tab is active. The code in the "On Message" tab is:

```
const temp = msg.payload  
const obj = {  
  "id":1,  
  "temp":temp  
}  
msg.payload = obj  
return msg;
```

A yellow callout labeled "click" points to the "Done" button in the modal window. Another yellow callout labeled "Put this javascript" points to the code in the "On Message" tab.

# Create node-red flow

## Config mongoDB block

The image shows the Node-RED interface with a flow titled "Flow 1". The flow consists of several nodes: a "15" node connected to a "/temp" node, an "On" node and an "Off" node both connected to a "/usb" node, an "LED" node, a "/temp" node, a "function" node, and a "mongodb3" node. A yellow callout box labeled "Double click" points to the "mongodb3" node.

Below the flow, four configuration screens for the "mongodb3" node are displayed:

- Edit mongodb3 in node**: Shows a "click" button pointing to the "Server" dropdown which is set to "External service" and "Add new mongodb3...".
- Edit mongodb3 in node > Add new mongodb3 config node**: Shows a "click" button pointing to the "URI" field which contains "mongodb://host1.yourprovider.com:27017,host2.". It also shows fields for "Name", "Username", "Password", "Connection Options", and "Parallelism Limit".
- Edit mongodb3 in node > Add new mongodb3 config node**: Shows a "click" button pointing to the "URI" field which contains "mongodb://203.150.243.118:27017/mqtt". It also shows fields for "Name", "Username", "Password", "Connection Options", and "Parallelism Limit".
- Edit mongodb3 in node**: Shows a "click" button pointing to the "Collection" dropdown which is set to "temp". It also shows fields for "Name", "Username", "Password", "Connection Options", and "Parallelism Limit". A red callout box labeled "Choose operation insertOne" points to the "Operation" dropdown which is set to "insertOne".

At the bottom, a green node configuration bar is shown with the URI "mongodb://203.150.243.118:27017/mqtt temp insertOne". Red callout boxes point to the "Database name" ("mongodb://203.150.243.118:27017/mqtt"), "Collection name" ("temp"), and "Operation" ("insertOne").

# Deploy and turn KidBright on

A Scratch script is shown at the top, with a message block reading "mongodb://203.150.243.118:27017/mqtt temp insertOne". A red callout below it says "13, success: 13, error: 0". To the right, a red box contains the text "MongoDB node is running". Below this, two screenshots of the MongoDB Compass interface are shown. The left screenshot shows the "Databases" tab with three databases: "READ\_ME\_TO\_RECOVER\_YOUR\_DATA", "admin", and "config". An orange arrow points from the text "click to refresh" to the "refresh" button in the top right corner of the interface. The right screenshot shows the same interface after refreshing, with an additional database named "mqtt" listed.

click to refresh

MongoDB node is running

New database will appear

Database Name	Storage Size	Collections	Indexes
READ_ME_TO_RECOVER_YOUR_DATA	16.0KB	1	1
admin	48.0KB	0	3
config	20.0KB	0	2

Database Name	Storage Size	Collections	Indexes
READ_ME_TO_RECOVER_YOUR_DATA	16.0KB	1	1
admin	48.0KB	0	3
config	24.0KB	0	2
mqtt	16.0KB	1	1

# Verify data in database

The image displays two side-by-side MongoDB Compass windows. The left window is titled 'MongoDB Compass - 203.150.243.118:27017/mqqt'. It shows a table with the following data:

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
temp	247	46.0 B	11.1 KB	1	36.0 KB	

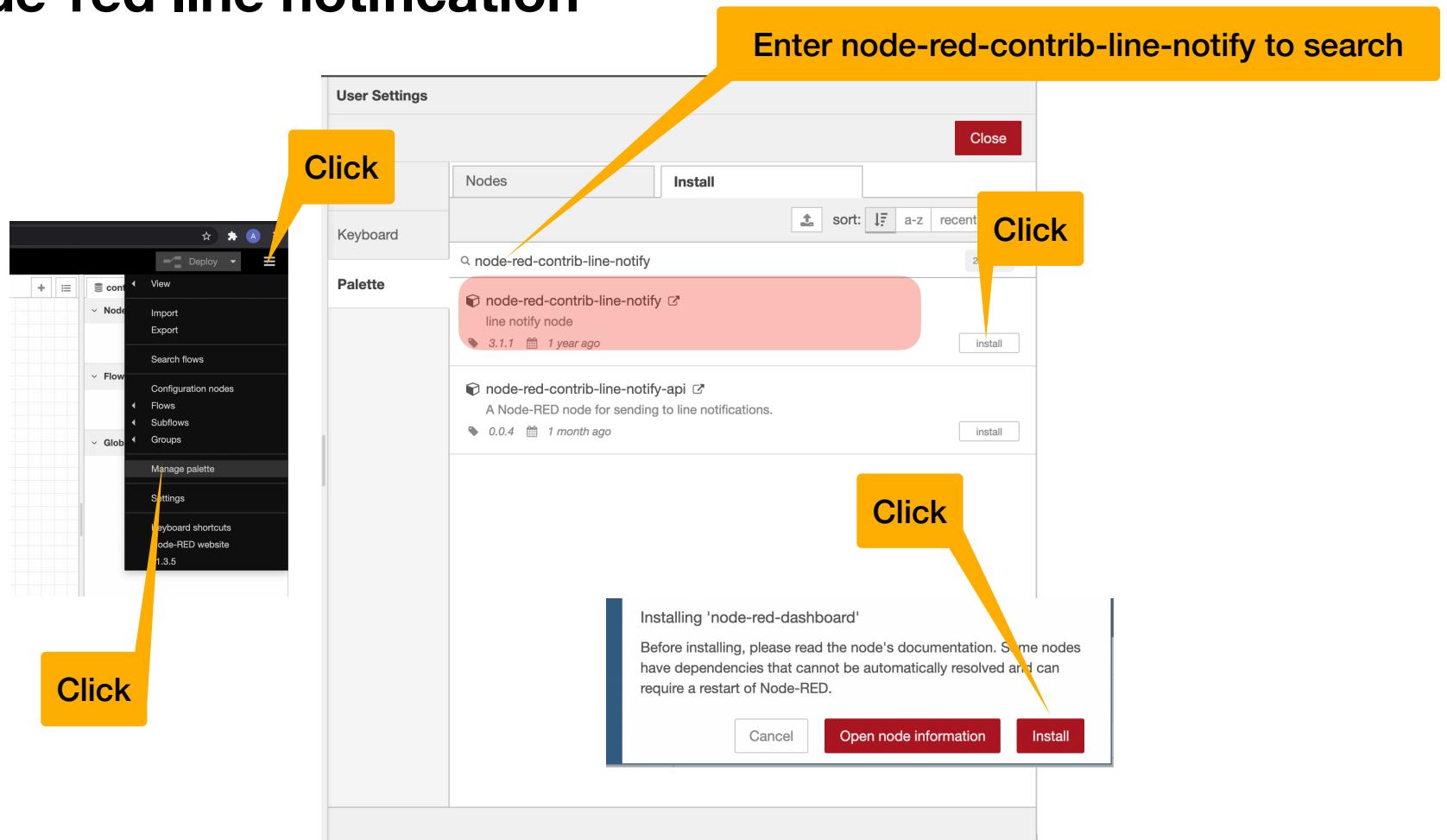
The right window is titled 'MongoDB Compass - 203.150.243.118:27017/mqqt.temp'. It shows a table with the following data:

_id	temp
ObjectId("60f252e587307b2b0724ccd5")	"32.50"
ObjectId("60f252e687307b2b0724ccd6")	"32.50"
ObjectId("60f252e787307b2b0724ccd7")	"32.50"
ObjectId("60f252e987307b2b0724ccd8")	"32.50"
ObjectId("60f252eb87307b2b0724ccd9")	"32.75"
ObjectId("60f252ed87307b2b0724ccda")	"32.75"

Two yellow callout boxes with the word 'click' are overlaid on the interface. One points to the 'CREATE COLLECTION' button in the top-left of the left window. The other points to the 'temp' collection in the left sidebar of the left window.

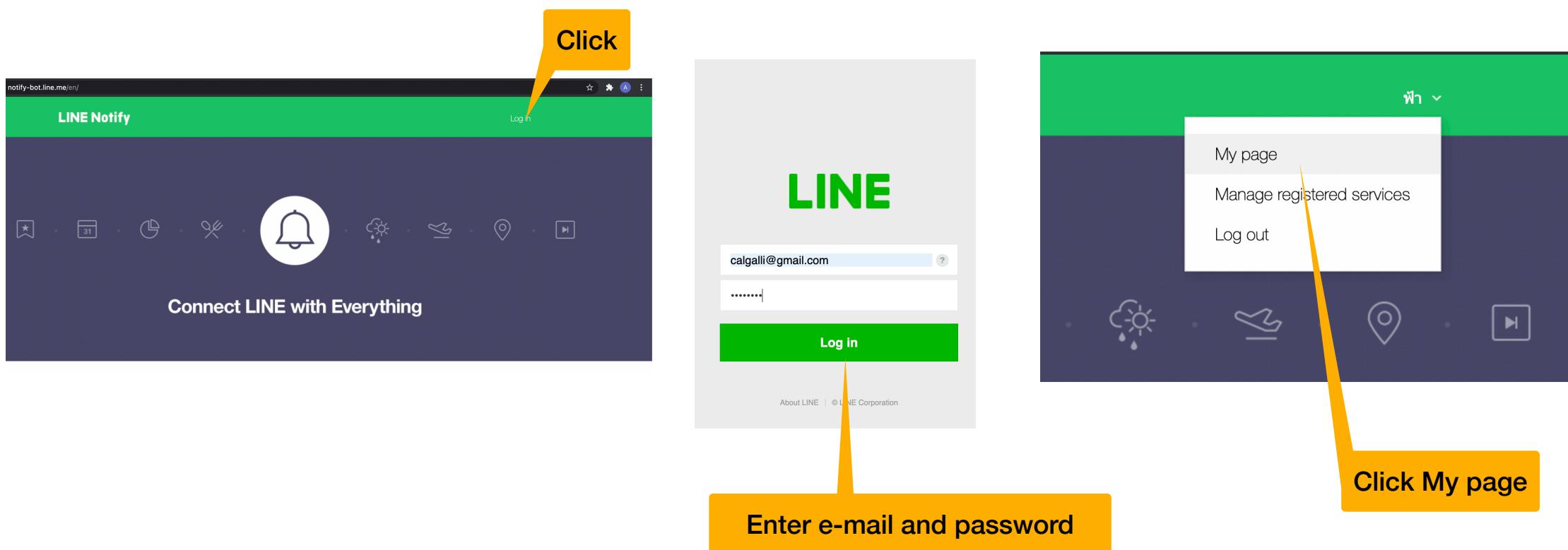
# Line notify : Show notification on LINE app

## Install node-red line notification



# Get Line notification token

Login <https://notify-bot.line.me/en/>



# Generate token

## Generate access token (For developers)

By using personal access tokens, you can configure notifications without having to add a web service.



**Generate token**

Please enter a token name to be displayed before each notification.

Select a chat to send notifications to.

Search by group name

- 1-on-1 chat with LINE Notify
- ##Intarapanich##
- (TA)Trainer for EEC#1
- AI Designer

**Choose yourself**

Note: Revealing your personal access token can allow a third party to obtain the names of your connected chats as well as your profile name.

**Generate token**

**Your token is:**

**CABOs5IoZV6EMSDWPI2rG9mJIELACsqlupLer**

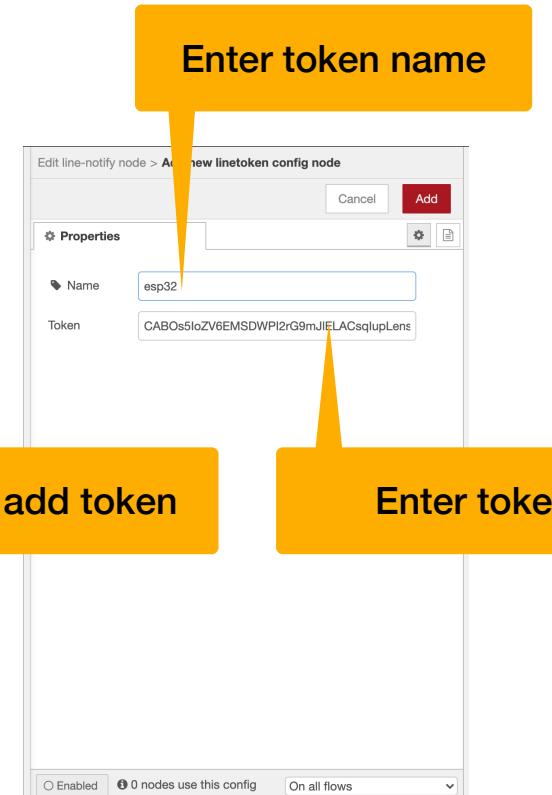
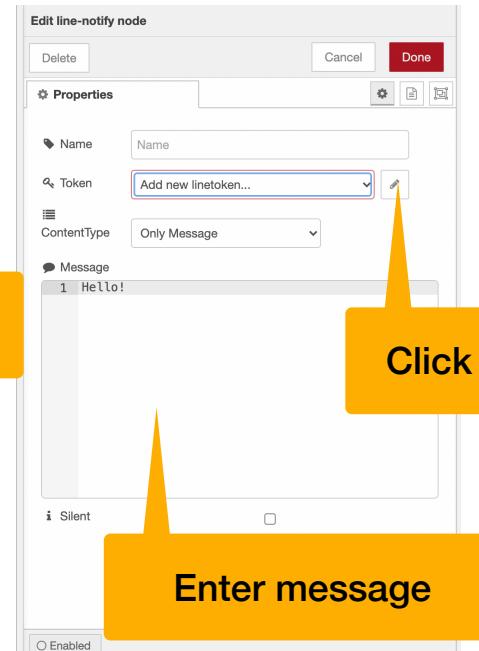
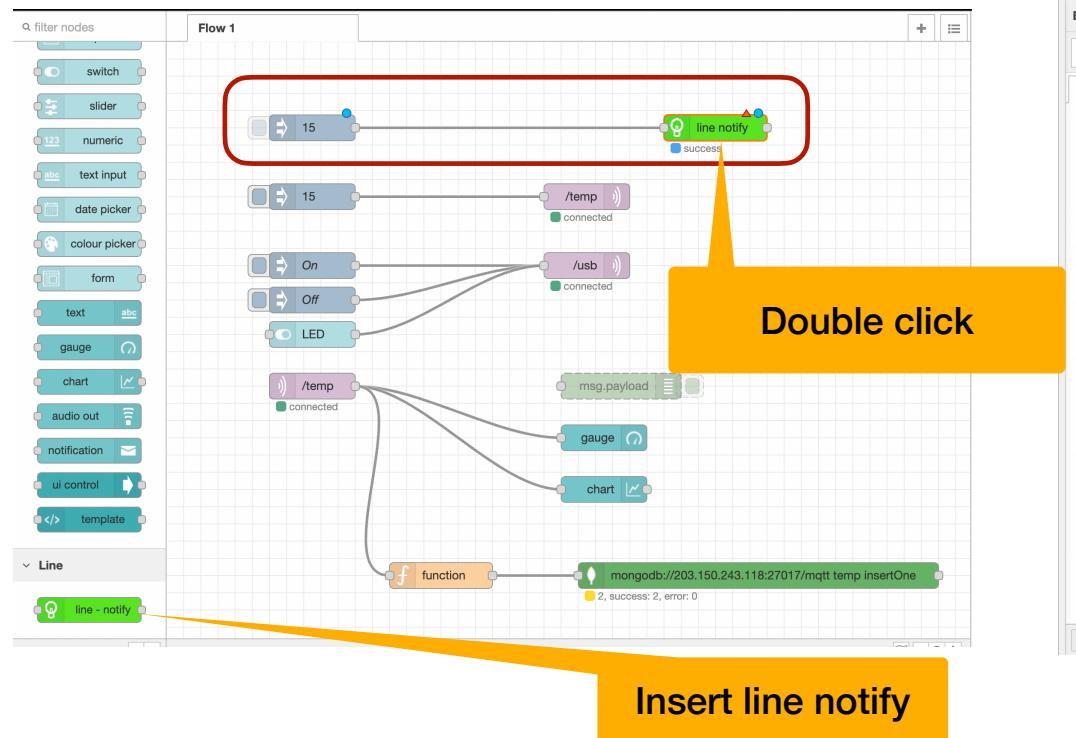
If you leave this page, you will not be able to view your newly generated token again. Please copy the token before leaving this page.

**Copy** **Close**

**Copy token**

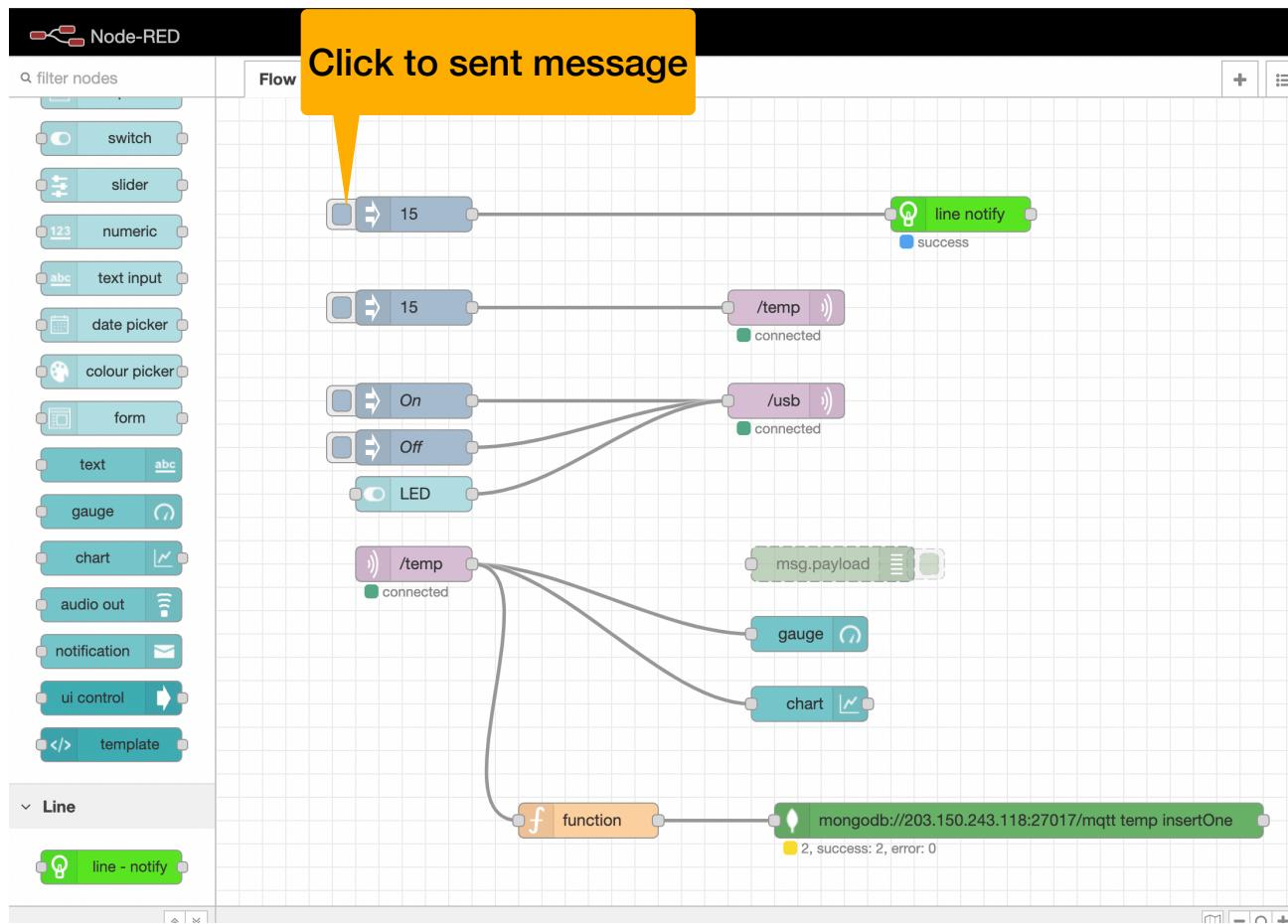
# Config line notify

## Enter the token and message



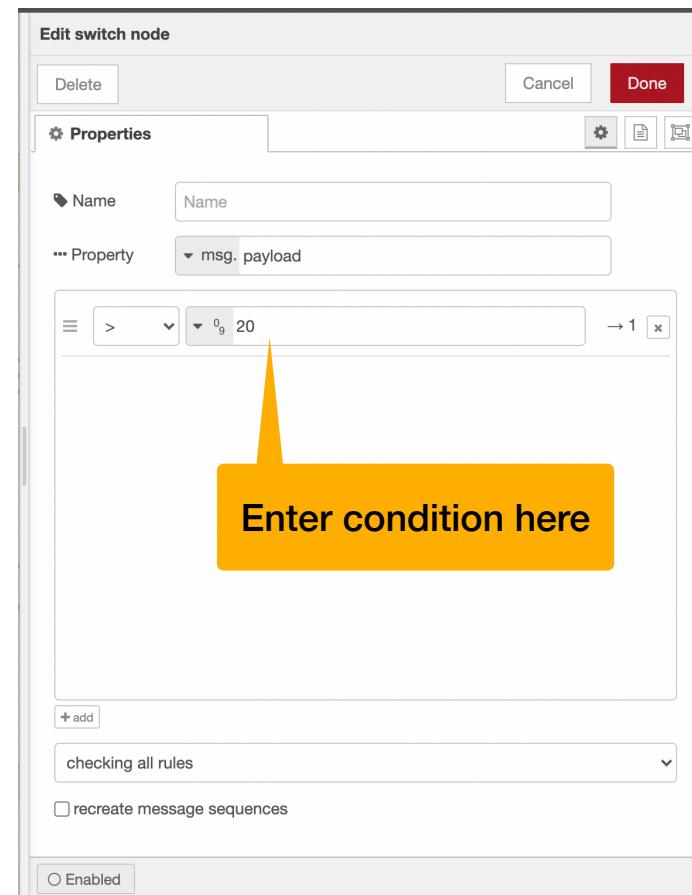
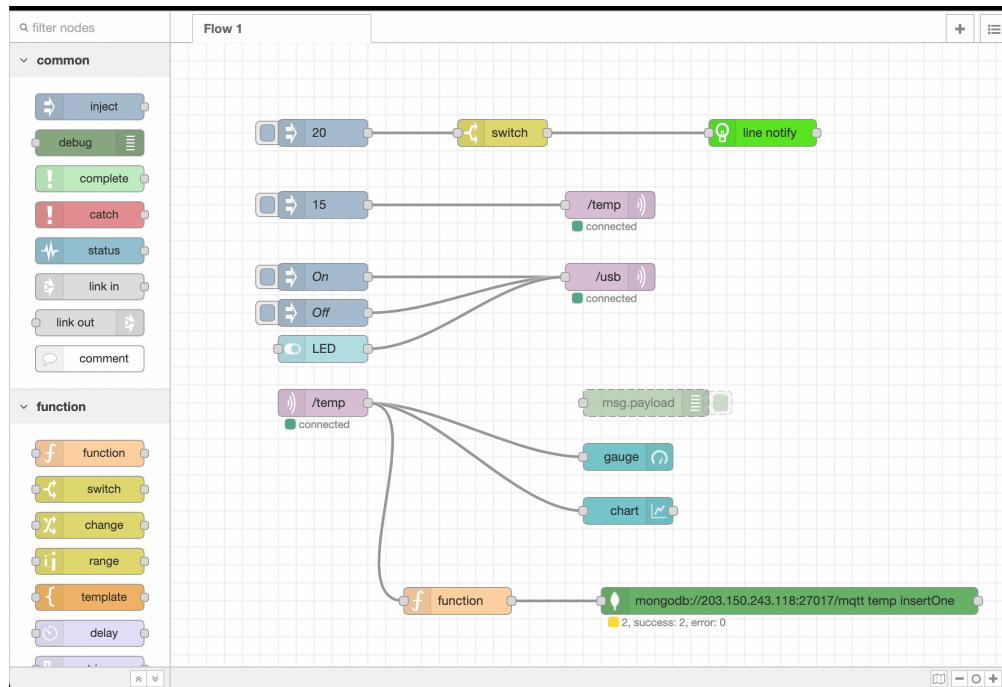
Click to add token

# Test Line notification



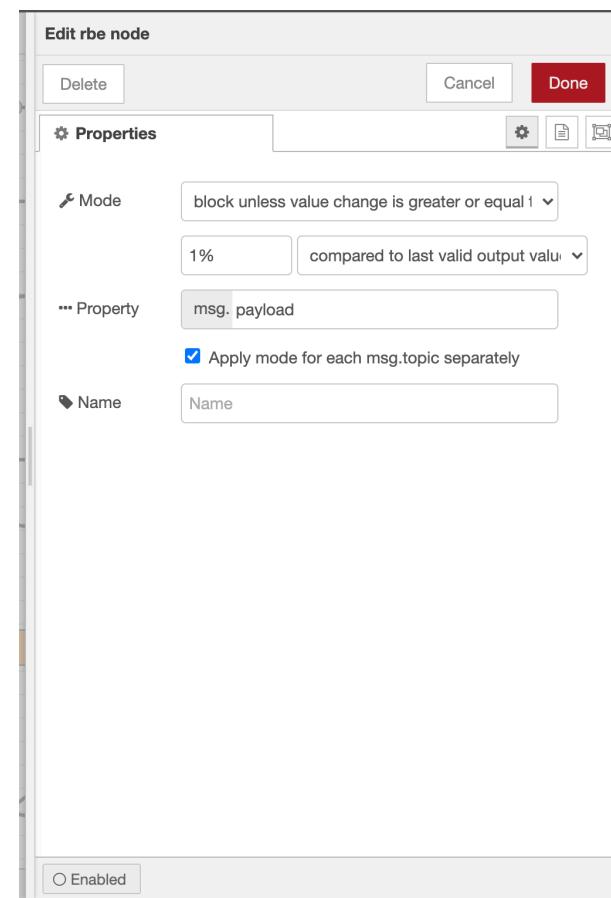
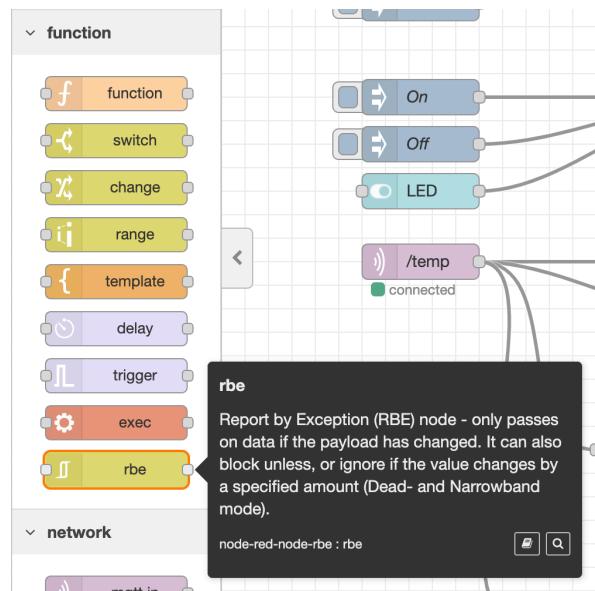
# Add switch to compare value

## If the value is greater than 20 then sent message



# Exercise : check temperature from KidBright

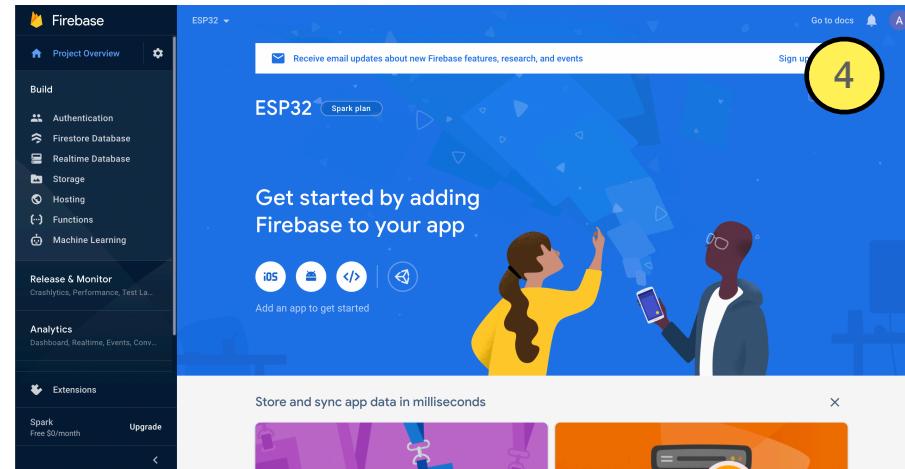
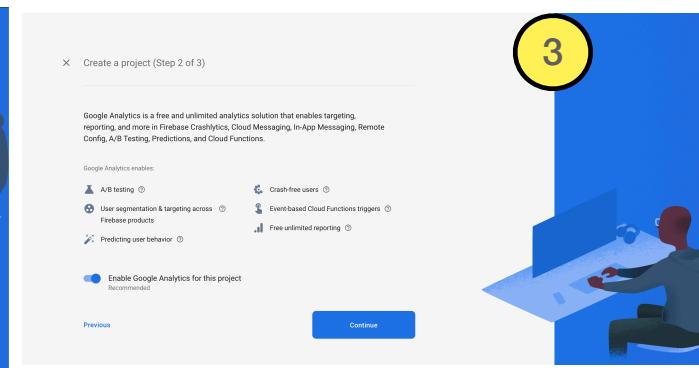
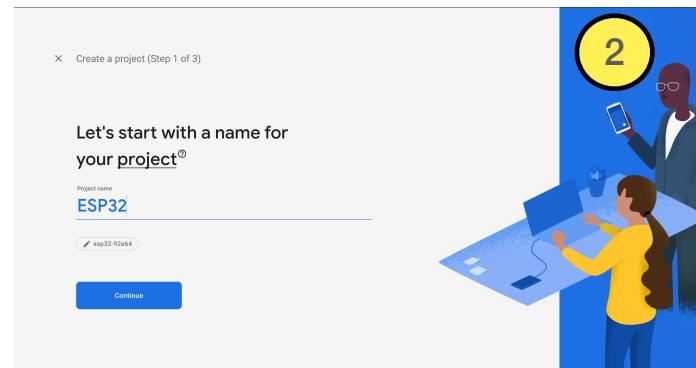
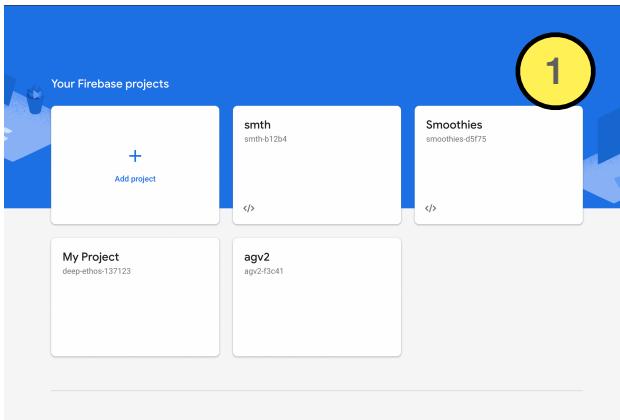
Check temperature change and notify when it is higher than a threshold



**Firebase : a real time database**

# Create a Firebase project

Go to <https://console.firebaseio.google.com/>



# Set up Authentication

## Enable *Anonymous Sign In*

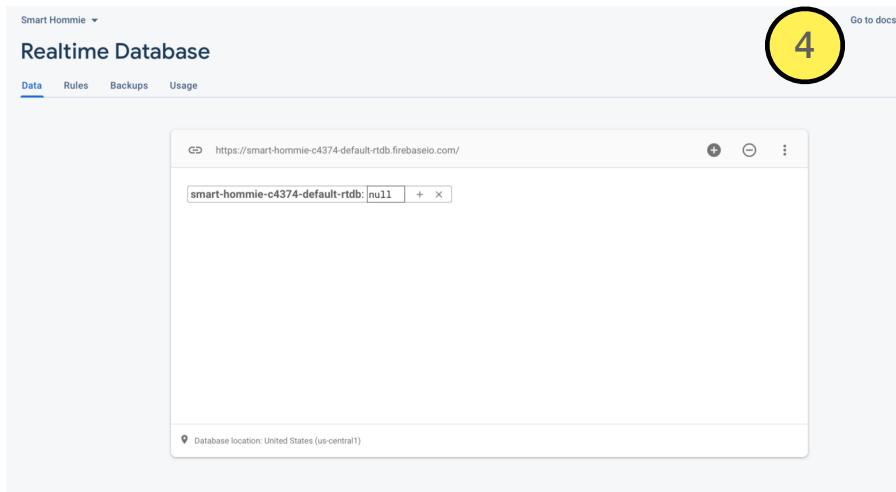
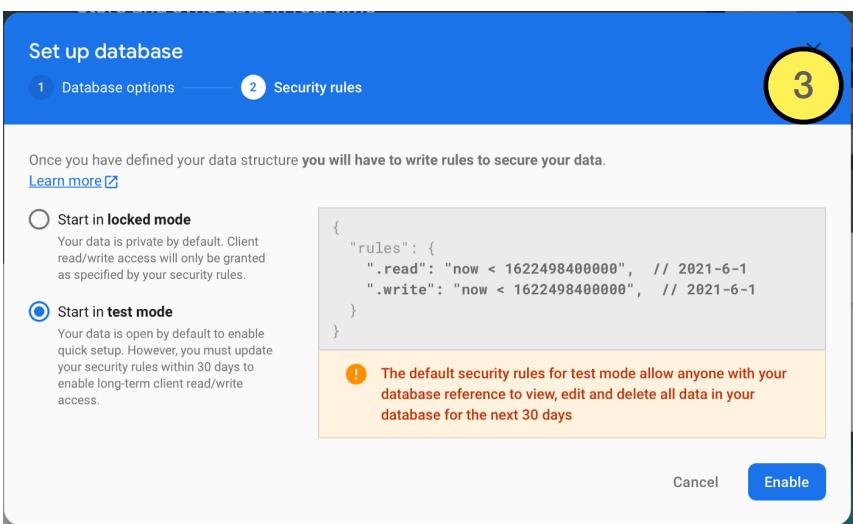
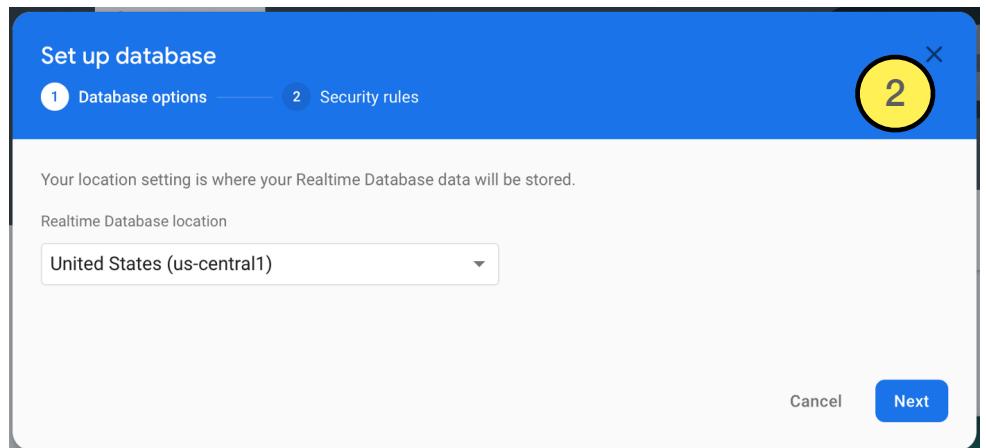
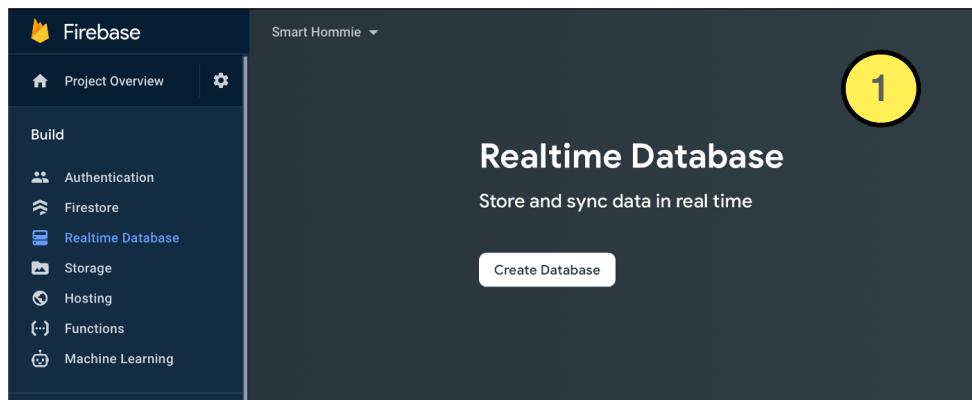
The screenshot shows the Firebase Project Overview page for a project named "ESP32". The left sidebar includes options for Authentication, Firestore Database, Realtime Database, Storage, Hosting, Functions, and Machine Learning. The main content area is titled "Authentication" and contains a sub-section titled "Learn more" with links to "How do I get started?" and "How does Authentication work?". A video thumbnail titled "Introducing Firebase Authentication" is also present.

The guide consists of four numbered steps:

- 1**: Shows the "Sign-in method" tab selected in the Firebase Authentication settings. It lists various providers: Email/Password, Phone, Google, Play Games, Game Center, Facebook, Twitter, GitHub, Yahoo, Microsoft, Apple, and Anonymous. All are currently disabled.
- 2**: Shows the "Anonymous" provider row in the list. The "Status" column indicates it is "Disabled".
- 3**: Shows the "Anonymous" provider row again, but now with the "Status" column showing "Enabled".
- 4**: Shows a detailed view of the "Anonymous" provider settings. It includes a toggle switch labeled "Enable" which is turned on, and a note: "Enable anonymous guest accounts in your application, which lets you enforce user-specific Security and Firebase rules without requiring credentials from your users." At the bottom are "Cancel" and "Save" buttons.

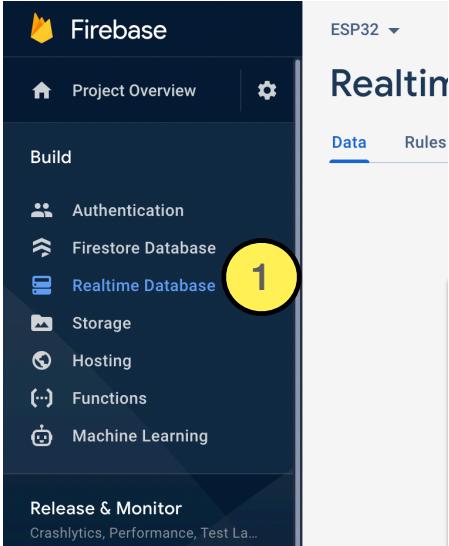
# Create a database

## create a database for our data

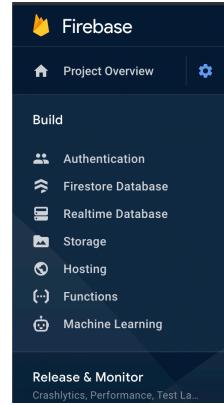


# Get API Key and database URL for ESP32

## API key and database URL will be used for ESP32 authentication



1



2



2

**API key**

**Database URL**

The screenshots illustrate the steps to retrieve the API key and database URL from a Firebase project:

- Project Overview:** In the left sidebar, the "Realtime Database" icon is highlighted with a yellow circle labeled "1".
- Project settings:** Under the "General" tab, the "Web API Key" field is highlighted with a yellow circle labeled "2".
- Realtime Database:** On the "Data" tab, the database URL is shown in the browser address bar, highlighted with a yellow circle labeled "2".

# Firebase ESP32 publish data

## ESP32 sets values in database

### Define authentication API\_KEY and DATABASE\_URL

```
// Your Firebase Project Web API Key  
#define API_KEY "AIzaSyARWZzmjIfRLHXU-C5xX0NkqSArJCBmDxQ"  
// Your Firebase Realtime database URL  
#define DATABASE_URL "https://esp32-92e64-default-rtdb.firebaseio.com"
```

```
Serial.printf("Set int... %s\n", Firebase.RTDB.setFloat(&fbdo, "/sensor/temp", readTemperature()) ? "ok" : fbdo.errorReason().c_str());
```

# ESP32 the first Firebase program

## fisrFirebase.ino

```
// Your Firebase Project Web API Key
#define API_KEY "AIzaSyCG59xub0TJe2gMqg1hz2NmclvY_0AK-_M"
// Your Firebase Realtime database URL
#define DATABASE_URL "https://esp32-b74d2-default-rtdb.firebaseio.com/"

const char *ssid = "CAKE";
const char *password = "0891300633";
```

Set authentication

```
void firebase_init() {
    // configure firebase API Key
    config.api_key = API_KEY;
    // configure firebase realtime database url
    config.database_url = DATABASE_URL;
    // Enable WiFi reconnection
    Firebase.reconnectWiFi(true);
    Serial.println("-----");
    Serial.println("Sign up new user...");
    // Sign in to firebase Anonymously
    if (Firebase.signUp(&config, &auth, "", "")) {
        Serial.println("Success");
        isAuthenticated = true;
        // Set the database path where updates will be loaded for this device
        databasePath = "/";
        uid = auth.token.uid.c_str();
    } else {
        Serial.printf("Failed, %s\n", config.signer.signupError.message.c_str());
        isAuthenticated = false;
    }
    // Assign the callback function for the long running token generation task,
    // see addons/TokenHelper.h
    config.token_status_callback = tokenStatusCallback;
    // Initialise the firebase library
    Firebase.begin(&config, &auth);
}
```

Connect to firebase

```
void loop() {
    if (Firebase.ready() && (millis() - sendDataPrevMillis > 15000 || sendDataPrevMillis == 0))
    {
        sendDataPrevMillis = millis();

        Serial.printf("Set int... %s\n", Firebase.RTDB.setInt(&fbdo, "/test/int", count) ? "ok" : fbdo.errorReason().c_str());

        Serial.printf("Get int... %s\n", Firebase.RTDB.getInt(&fbdo, "/test/int") ? String(fbdo.intData()).c_str() : fbdo.errorReason().c_str());

        FirebaseJson json;
        json.add("value", count);

        Serial.printf("Push json... %s\n", Firebase.RTDB.pushJSON(&fbdo, "/test/push", &json) ? "ok" : fbdo.errorReason().c_str());

        json.set("value", count + 100);
        Serial.printf("Update json... %s\n", Firebase.RTDB.updateNode(&fbdo, String("/test/push/" + fbdo.pushName()).c_str(), &json) ? "ok" : fbdo.errorReason().c_str());

        count++;
    }
}
```

Send data to firebase

# Observe data in the database

ESP32 ▾

## Realtime Database

Data Rules Backups Usage

Protect your Realtime Database resources from abuse, such as billing fraud or phishing Configure App Check

https://esp32-b74d2-default-rtdb.firebaseio.com/

esp32-b74d2-default-rtdb

- test
  - int: 7
  - push

Database location: Singapore (asia-southeast1)

ESP32 ▾

## Realtime Database

Data Rules Backups Usage

Protect your Realtime Database resources from abuse, such as billing fraud or phishing Configure App Check

https://esp32-b74d2-default-rtdb.firebaseio.com/

esp32-b74d2-default-rtdb

- test
  - int: 4
  - push

Database location: Singapore (asia-southeast1)

# ESP32 publish temperature to firebase

## sendTempToFirebase.ino

```
float readTemperature() {
    Wire1.beginTransmission(LM73_ADDR);
    Wire1.write(0x00); // Temperature Data Register
    Wire1.endTransmission();

    uint8_t count = Wire1.requestFrom(LM73_ADDR, 2);
    float temp = 0.0;
    if (count == 2) {
        byte buff[2];
        buff[0] = Wire1.read();
        buff[1] = Wire1.read();
        temp += (int)(buff[0]<<1);
        if (buff[1]&0b10000000) temp += 1.0;
        if (buff[1]&0b01000000) temp += 0.5;
        if (buff[1]&0b00100000) temp += 0.25;
        if (buff[0]&0b10000000) temp *= -1.0;
    }
    return temp;
}

void firebase_init() {
    // configure firebase API Key
    config.api_key = API_KEY;
    // configure firebase realtime database url
    config.database_url = DATABASE_URL;
    // Enable WiFi reconnection
    Firebase.reconnectWiFi(true);
    Serial.println("-----");
    Serial.println("Sign up new user...");
    // Sign in to firebase Anonymously
    if (Firebase.signUp(&config, &auth, "", "")) {
        Serial.println("Success");
        isAuthenticated = true;
        // Set the database path where updates will be loaded for this device
        databasePath = "/";
        fuid = auth.token.uid.c_str();
    } else {
        Serial.printf("Failed, %s\n", config.signer.signupError.message.c_str());
        isAuthenticated = false;
    }
    // Assign the callback function for the long running token generation task,
    // see addons/TokenHelper.h
    config.token_status_callback = tokenStatusCallback;
    // Initialise the firebase library
    Firebase.begin(&config, &auth);
}

void setup() {
    // Configure pin
    pinMode(led_pin_1, OUTPUT);
    pinMode(led_pin_2, OUTPUT);

    Serial.begin(115200);
    Wire1.begin(4, 5);

    WiFi.begin(ssid, password);

    while (WiFi.status() != WL_CONNECTED) {
        digitalWrite(led_pin_1, HIGH);
        delay(500);
        digitalWrite(led_pin_1, LOW);
        delay(500);
    }
    digitalWrite(led_pin_1, LOW);
    Serial.println("Connected to the WiFi network");
    firebase_init();
}

void loop() {
    if (Firebase.ready() && (millis() - sendDataPrevMillis > update_interval || sendDataPrevMillis == 0))
    {
        Serial.println(readTemperature());
        sendDataPrevMillis = millis();

        Serial.printf("Set int... %s\n", Firebase.RTDB.setFloat(&fbdo, "/sensor/temp", readTemperature()) ? "ok" :
        fbdo.errorReason().c_str());

        Serial.printf("Get int... %s\n", Firebase.RTDB.getFloat(&fbdo, "/sensor/temp") ? String(fbdo.intData()).c_str() :
        fbdo.errorReason().c_str());
    }
}
```

ESP32 ▾

## Realtime Database

Data Rules Backups Usage

Protect your Realtime Database resources from abuse, such as billing fraud or phishing.

<https://esp32-b74d2-default-rtbd.firebaseio.com/>

```
esp32-b74d2-default-rtbd
  ↘ sensor
    ↘ temp: 33
  ↘ test
    ↘ int: 53
    + push
```

Database location: Singapore (asia-southeast1)

ESP32 ▾

## Realtime Database

Data Rules Backups Usage

Protect your Realtime Database resources from abuse, such as billing fraud or phishing [Configure App Check](#).

<https://esp32-b74d2-default-rtbd.firebaseio.com/>

**esp32-b74d2-default-rtbd**

```
  ↘ sensor
    ↘ temp: 33
  ↘ test
    ↘ int: 53
    + push
```

Database location: Singapore (asia-southeast1)

# Firebase ESP32 listens to data firebaseDataChangeCallback.ino

```
void firebase_init() {
    // configure firebase API Key
    config.api_key = API_KEY;
    // configure firebase realtime database url
    config.database_url = DATABASE_URL;
    // Enable WiFi reconnection
    Firebase.reconnectWiFi(true);
    Serial.println("-----");
    Serial.println("Sign up new user...");
    // Sign in to firebase Anonymously
    if (Firebase.signUp(&config, &auth, "", "")) {
        Serial.println("Success");
        isAuthenticated = true;
        // Set the database path where updates will be loaded for this device
        databasePath = "/";
        uid = auth.token.uid.c_str();
    } else {
        Serial.printf("Failed, %s\n", config.signer.signupError.message.c_str());
        isAuthenticated = false;
    }
    // Assign the callback function for the long running token generation task,
    // see addons/TokenHelper.h
    config.token_status_callback = tokenStatusCallback;
    // Initialise the firebase library
    Firebase.begin(&config, &auth);

    if (!Firebase.RTDB.beginStream(&stream, "/status"))
        Serial.printf("stream begin error, %s\n", stream.errorReason().c_str());

    Firebase.RTDB.setStreamCallback(&stream, streamCallback, streamTimeoutCallback);
}
```

```
void streamCallback(FirebaseStream data)
{
    Serial.printf("stream path, %s\nevent path, %s\ndata type, %s\nevent type, %s\n\n",
                  data.streamPath().c_str(),
                  data.dataPath().c_str(),
                  data.dataType().c_str(),
                  data.eventType().c_str());
    printResult(data); //see addons/RTDBHelper.h
    Serial.println();

    if(data.intData() == 1) {
        digitalWrite(usb_pin,LOW);
    } else if(data.intData() == 0){
        digitalWrite(usb_pin, HIGH);
    }
}

void streamTimeoutCallback(bool timeout)
{
    if (timeout)
        Serial.println("stream timeout, resuming...\n");
}
```

# Firebase ESP32 listens to data

## Create new data value

The image consists of three side-by-side screenshots of the Firebase Realtime Database console, illustrating the process of creating a new data value.

**Screenshot 1:** Shows the database structure at `https://esp32-b74d2-default.firebaseio.com/.json`. It contains a node `esp32-b74d2-default-rtdb` with children `sensor`, `status: 0`, and `test`. A yellow callout bubble with the text "Click" points to the `+ Add child` button at the top of the list.

**Screenshot 2:** Shows a modal dialog for adding a new child. The path is `https://esp32-b74d2-default-rtdb.firebaseio.com/.json`. The modal has fields for "Name" (set to "status") and "Value" (set to "1"). Below the fields are "Cancel" and "Add" buttons. A yellow callout bubble with the text "Click" points to the "Add" button.

**Screenshot 3:** Shows the updated database structure at `https://esp32-b74d2-default.firebaseio.com/.json`. The `status` child under `esp32-b74d2-default-rtdb` now has a value of "1". The `sensor`, `status: 0`, and `test` nodes remain as they were.

# Test by changing status value

## Observe USB light

<https://esp32-b74d2-default-rtdb.firebaseio.com/>

esp32-b74d2-default-rtdb

- + sensor
- status: 0 Click to change value
- + test

Database location: Singapore (asia-southeast1)

<https://esp32-b74d2-default-rtdb.firebaseio.com/>

esp32-b74d2-default-rtdb

- + sensor
- status:  x
- + test

Database location: Singapore (asia-southeast1)

# Web app dashboard

The image shows the Firebase web app dashboard. On the left, the main navigation bar includes Project Overview, Authentication, Firestore Database, and a selected item, Firebase. The Firebase section has sub-options like Project Overview, Build, Authentication, Firestore Database, Realtime Database, Storage, Hosting, Functions, Machine Learning, Release & Monitor, Analytics, Engage, Extensions, and Spark. A modal window titled "Add Firebase to your web app" is open over the main content. The modal has two steps: 1. Register app, where the app nickname is set to "esp32". Step 2. Add Firebase SDK, which displays a large amount of JavaScript code for initializing the Firebase SDK. Three yellow callout boxes with the word "Click" point to the "Register app" button, the "Add Firebase SDK" button, and the "Continue" button at the bottom of the modal.

Project settings

Users and permissions

Usage and billing

Project name: ESP32

Project ID: esp32-b74d2

Project number: 265654270299

Default GCP resource location: Not yet selected

Web API Key: AlzaSyCG59xubOTJe2gMqg1hz2NmclLvY\_

Public settings

These settings control instances of your project shown to the public

Public-facing name: project-265654270299

Support email: Not configured

Your apps

There are no apps in your project

Select a platform to get started

iOS Android </> Web

Register app

Also set up Firebase Hosting for this app

Hosting can also be set up later. It's optional.

Register app

Add Firebase SDK

```
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/8.7.1.firebaseio-app.js"></script>

<!-- TODO: Add SDKs for Firebase products that you want to use
      https://firebase.google.com/docs/web/setup/available-libraries -->
<script src="https://www.gstatic.com/firebasejs/8.7.1.firebaseio-analytics.js"></script>

<script>
  // Your web app's Firebase configuration
  // For Firebase JS SDK v7.20.0 and later, measurementId is optional
  var firebaseConfig = {
    apiKey: "AIzaSyCG59xubOTJe2gMqg1hz2NmclLvY_0AK-M",
    authDomain: "esp32-b74d2.firebaseioapp.com",
    databaseURL: "https://esp32-b74d2.firebaseio.rtdb.firebaseio.com",
    projectId: "esp32-b74d2",
    storageBucket: "esp32-b74d2.appspot.com",
    measurementId: "265654270299",
    appId: "1:265654270299:web:c088adfcbebf180ac827e",
    measurementId: "G-SV7SY10B8K"
  };
  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
  firebase.analytics();
</script>
```

Learn more about Firebase for web: [Get Started](#) [Web SDK](#)

Are you using npm or a bundler like webpack or Rollup? [Read our documentation](#)

Continue

# Get Firebase configuration

The image shows two screenshots of the Firebase console. The left screenshot is the 'Project Overview' page for a project named 'ESP32'. The right screenshot is the 'Project settings' page for the same project. A yellow callout box with the text 'Copy configuration to clipboard' points from the left screenshot to the configuration code in the right screenshot.

**Project Overview (Left):**

- Project Overview
- Build
- Authentication
- Firebase Database

**Project settings (Right):**

- Project Overview
- Build
- Authentication
- Firebase Database
- Realtime Database
- Storage
- Hosting
- Functions
- Machine Learning

**Release & Monitor**  
Crashlytics, Performance, Test Lab...

**Analytics**  
Dashboard, Realtime, Events, Conv...

**Engage**  
Predictions, A/B Testing, Cloud M...

**Extensions**

Spark Free \$0/month Upgrade

**Copy configuration to clipboard**

Copy and paste these scripts into the bottom of your <body> tag, but before you use any Firebase services:

```
<!-- The core Firebase JS SDK is always required and must be listed
<script src="https://www.gstatic.com/firebasejs/8.7.1.firebaseio.js">
```

```
<!-- TODO: Add SDKs for Firebase products that you want to use
          https://firebase.google.com/docs/web/setup#available-libraries
<script src="https://www.gstatic.com/firebasejs/8.7.1/firebase-analytics.js">
```

```
<script>
  // Your web app's Firebase configuration
  // For Firebase JS SDK v7.20.0 and later, measurementId is optional
  var firebaseConfig = {
    apiKey: "AIzaSyCG59xub0TJe2gMqg1hz2NmclVY_0AK-_M",
    authDomain: "esp32-b74d2.firebaseio.com",
    databaseURL: "https://esp32-b74d2-default-rtdb.firebaseio.com",
    projectId: "esp32-b74d2",
    storageBucket: "esp32-b74d2.appspot.com",
    messagingSenderId: "265654270299",
    appId: "1:265654270299:web:c4888a6dfcebf180ac827e",
    measurementId: "G-SV7SY10B8K"
  };
  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
  firebase.analytics();
</script>
```

Are you using npm and a bundler like webpack or Rollup? Check out the [modular SDK](#), currently in beta.

# HTML file onoff.html

```
<html>
  <head>
    <script
      type="text/javascript"
      src="https://www.gstatic.com/charts/loader.js"
    ></script>
    <!-- The core Firebase JS SDK is always required and must be listed first -->
    <script src="https://www.gstatic.com/firebasejs/8.6.5.firebaseio-app.js"></script>

    <!-- TODO: Add SDKs for Firebase products that you want to use
    https://firebase.google.com/docs/web/setup#available-libraries -->
    <script src="https://www.gstatic.com/firebasejs/8.6.5/firebase-analytics.js"></script>
    <script src="https://www.gstatic.com/firebasejs/8.6.5.firebaseio-database.js"></script>

    <script>
      // Your web app's Firebase configuration
      // For Firebase JS SDK v7.20.0 and later, measurementId is optional

      var firebaseConfig = {
        apiKey: "AIzaSyCG59xub0TJe2gMqg1hz2NmclvY_0AK-_M",
        authDomain: "esp32-b74d2.firebaseioapp.com",
        databaseURL:"https://esp32-b74d2-default.firebaseio.asia-southeast1.firebaseio.app",
        projectId: "esp32-b74d2",
        storageBucket: "esp32-b74d2.appspot.com",
        messagingSenderId: "265654270299",
        appId: "1:265654270299:web:81685cf929e28a4dac827e",
        measurementId: "G-0F2YTRG1GX",
      };
      // Initialize Firebase
      firebase.initializeApp(firebaseConfig);
      firebase.analytics();

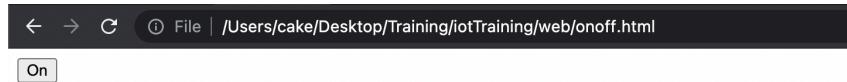
      if (!firebase.apps.length) {
        firebase.initializeApp(firebaseConfig);
      }
      this.database = firebase.database();

      function onoff() {
        currentvalue = document.getElementById("onoff").value;
        if (currentvalue == "Off") {
          document.getElementById("onoff").value = "On";
          firebase.database().ref("status").set(0);
        } else {
          document.getElementById("onoff").value = "Off";
          firebase.database().ref("status").set(1);
        }
      }
    </script>
  </head>
  <body>
    <input type="button" value="On" id="onoff" onclick="onoff()" />
  </body>
</html>
```

Paste clipboard

# Load onoff.html using Chrome

## Test on/off switch



# Appendix

## KidBright IDE

