Embedded System Software and IoT(Internet of Things)

2020. 3. 24

Division of Electrical Engineering
Hanyang University, ERICA Campus

Contents

Embedded Systems

Modern Computers

■ Concept of modern computers

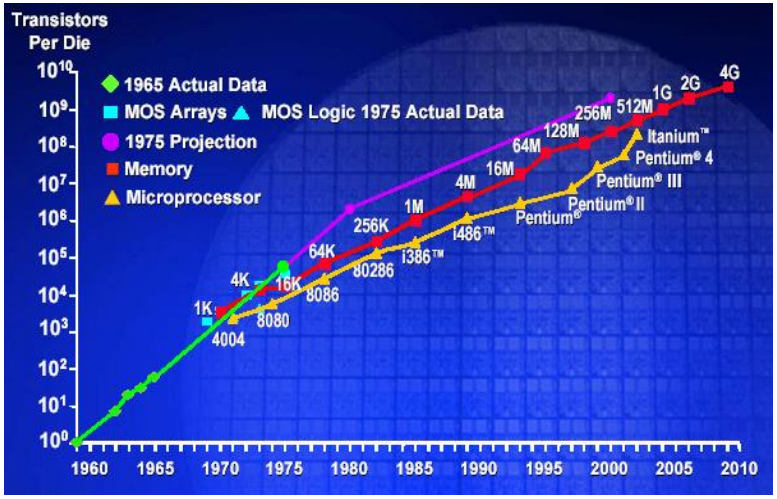
- The principle of the modern computer was proposed by **Alan Turing** in his seminal 1936 paper, *On Computable Numbers*. Turing proposed a simple device that he called "Universal Computing machine" and that is now known as a universal Turing machine. He proved that such a machine is capable of computing anything that is computable by executing instructions (program) stored on tape, allowing the machine to be programmable. The fundamental concept of Turing's design is the stored program, where all the instructions for computing are stored in memory. **Von Neumann** acknowledged that the central concept of the modern computer was due to this paper. Turing machines are to this day a central object of study in theory of computation. Except for the limitations imposed by their finite memory stores, modern computers are said to be Turing-complete, which is to say, they have algorithm execution capability equivalent to a universal Turing machine.

-Wikipedia

Early Personal Computers

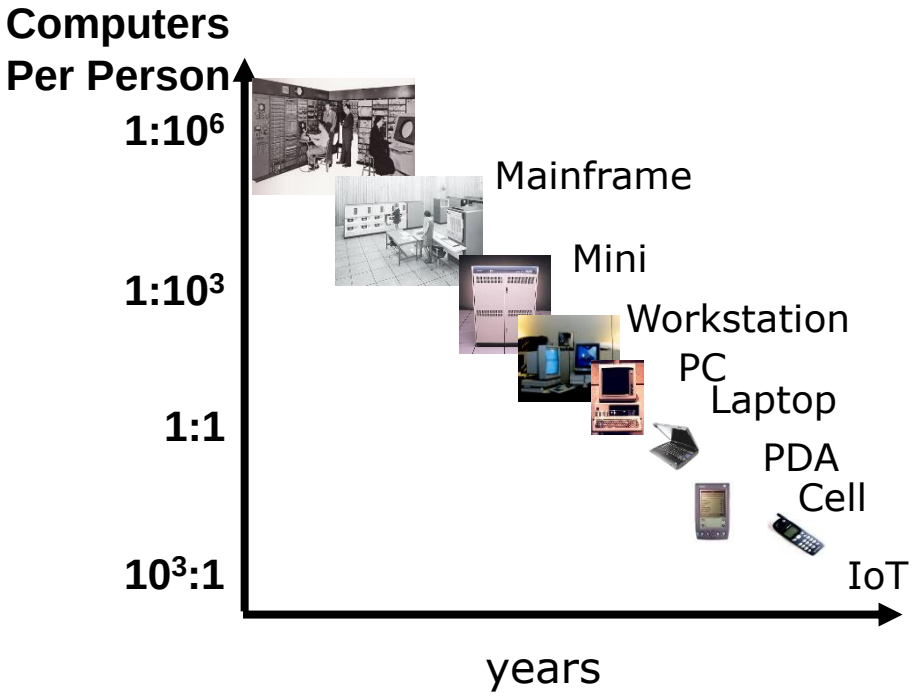
Technology Trends

Moore's Law: # transistors on cost-effective chip doubles every 18 months



Today: 1 million transistors per $

Bell's Law: a new computer class emerges every 10 years



전자공학부의 컴퓨터 관련 교과목

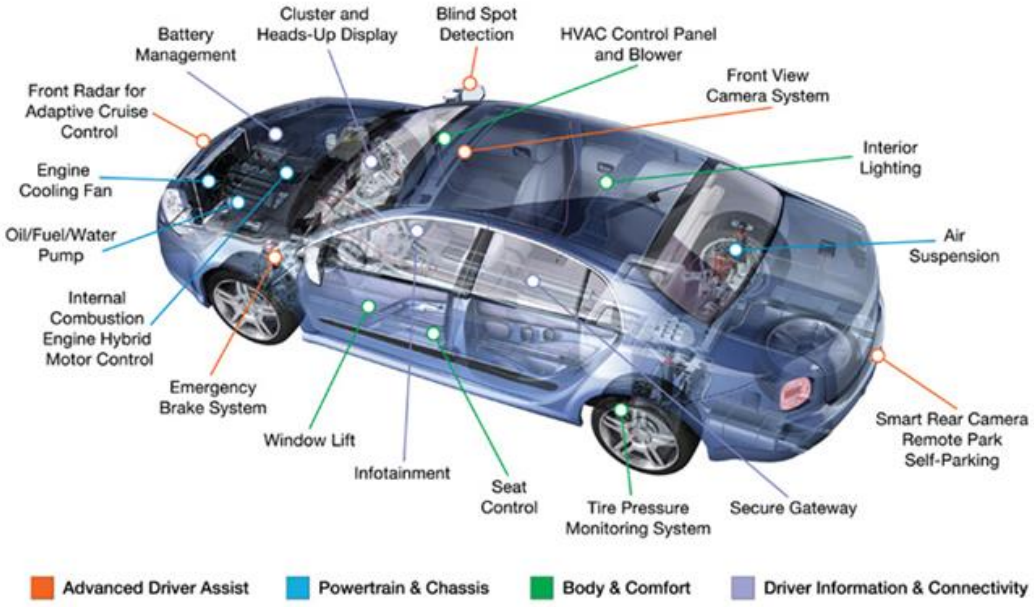
- 프로그래밍실습(S)
- 디지털논리설계(H)
- 데이터구조론(S)
- 논리회로실험(H)
- 디지털시스템(H)
- 컴퓨터아키텍쳐(H)
- 마이크로프로세서응용(S)
- 컴퓨터네트워크1,2
- 마이크로프로세서설계(H)
- 임베디드시스템설계(S)
- 알고리즘응용(S)

(S: Software, H: Hardware)

Definition of Embedded Systems

- "Any sort of device which includes a programmable computer but itself is not intended to be a general-purpose computer"- Marilyn Wolf

- Embedded System = Computers inside a product



Embedded Systems Overview

- **Computing systems are everywhere**
- **Most of us think of "desktop" computers**
  - PC's
  - Laptops
  - Mainframes
  - Servers
- **But there's another type of computing system**
  - Far more common…

- **Embedded computing systems**
  - Computing systems embedded within electronic devices
  - Hard to define. Nearly any computing system other than a desktop computer
  - Billions of units produced yearly, versus millions of desktop units
  - Perhaps 50 per household and per automobile

Slide credit Vahid/Givargis, Embedded Systems Design: A Unified Hardware/Software Introduction, 2000

How many do we use?

- **Average middle-class home has 40 to 50 embedded processors in it**
  - Microwave, washer, dryer, dishwasher, TV, VCR, stereo, hair dryer, coffee maker, remote control, humidifier, heater, toys, etc.
- **Luxury cars have over 80 embedded processors**
  - Brakes, steering, windows, locks, ignition, dashboard displays, transmission, mirrors, etc.
- **Personal computers have over 10 embedded processors**
  - Graphics accelerator, mouse, keyboard, hard-drive, CD-ROM, bus interface, network card, etc.

- Mike Schulte

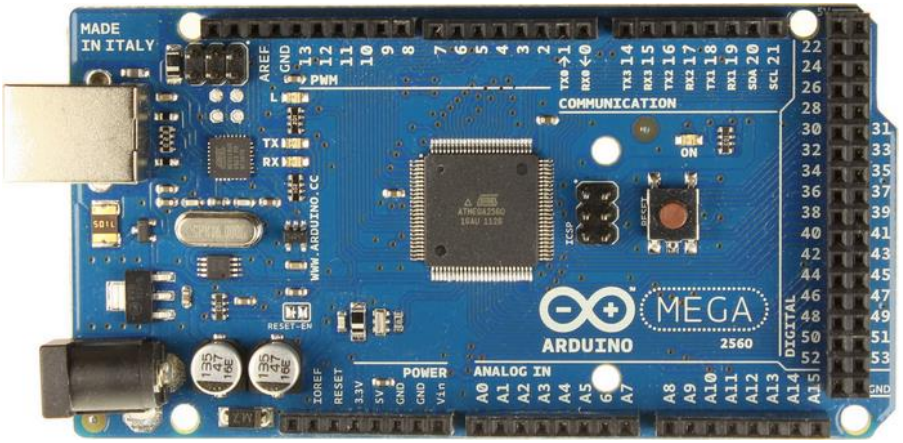
Types of Embedded Systems

- **General Computing**
  - Applications similar to desktop computing, but in an embedded package
  - Video games, set-top boxes, wearable computers, ATM
- **Control Systems**
  - Closed-loop feedback control of real-time system
  - Vehicle engines, chemical processes, flight control
- **Signal Processing**
  - Radar, sonar, video compression
- **Communications & Networking**
  - Telephone system, internet

Slide credit P Koopman, CMU

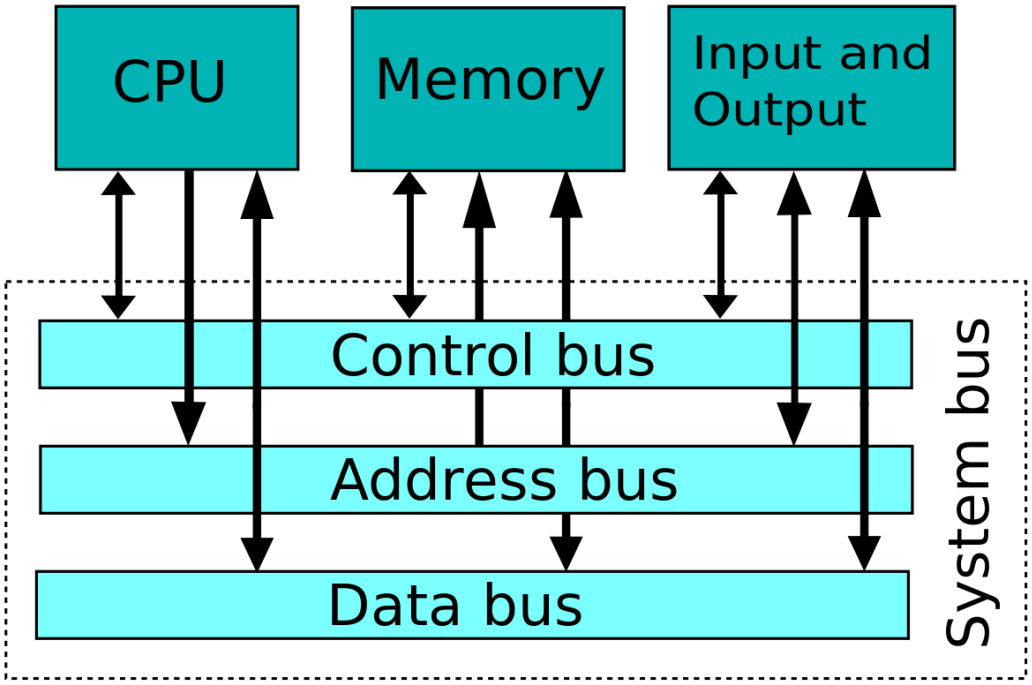
Processors

- **Microprocessors for PCs**
- **Embedded processors or Microcontrollers for embedded systems**
  - Often with lower clock speeds
  - Integrated with memory and
  - I/O devices e.g. A/D D/A PWM CAN
  - Higher environmental specs

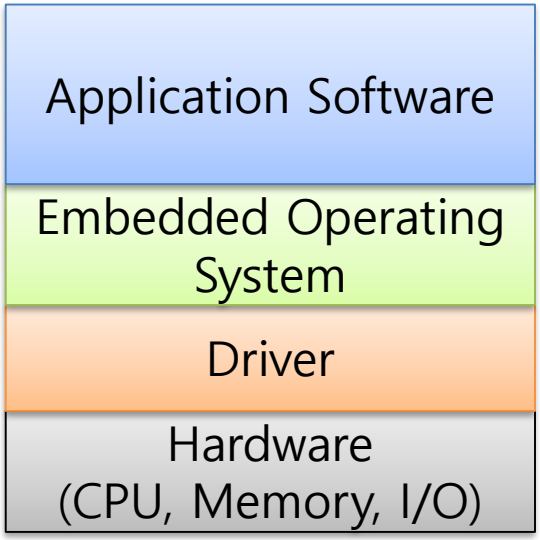


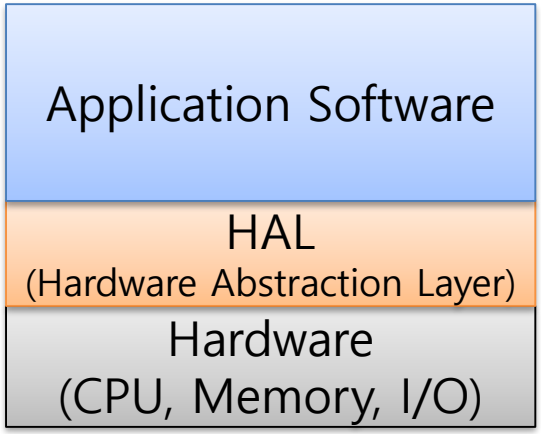
Microcomputer System

■ Microprocessor and Microcontroller



Embedded System Software





- **Embedded Operating Systems: Embedded Linux, Windows CE, VxWorks, pSOS, QNX, Nucleus, FreeRTOS, etc.**
- **Characteristics: Real-Time, Portability, Limited Resource, Reliability**

What is IoT?

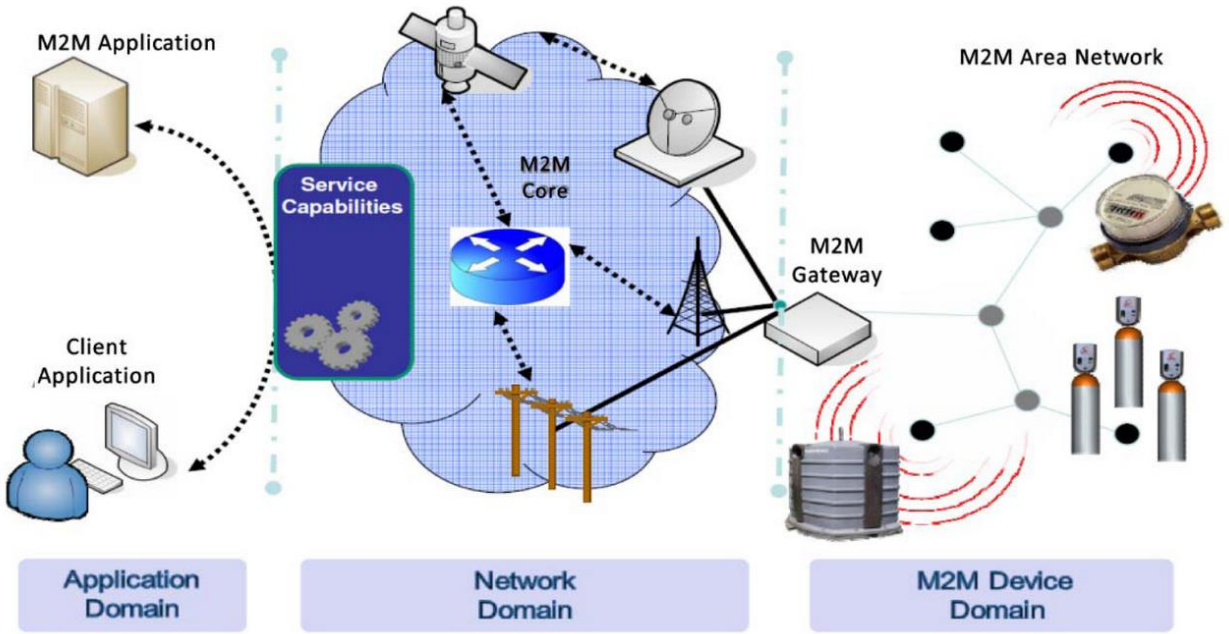
- The Internet of things (IoT) is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and network connectivity which enables these objects to connect and exchange data. Each thing is uniquely identifiable through its embedded computing system but is able to inter-operate within the existing Internet infrastructure.

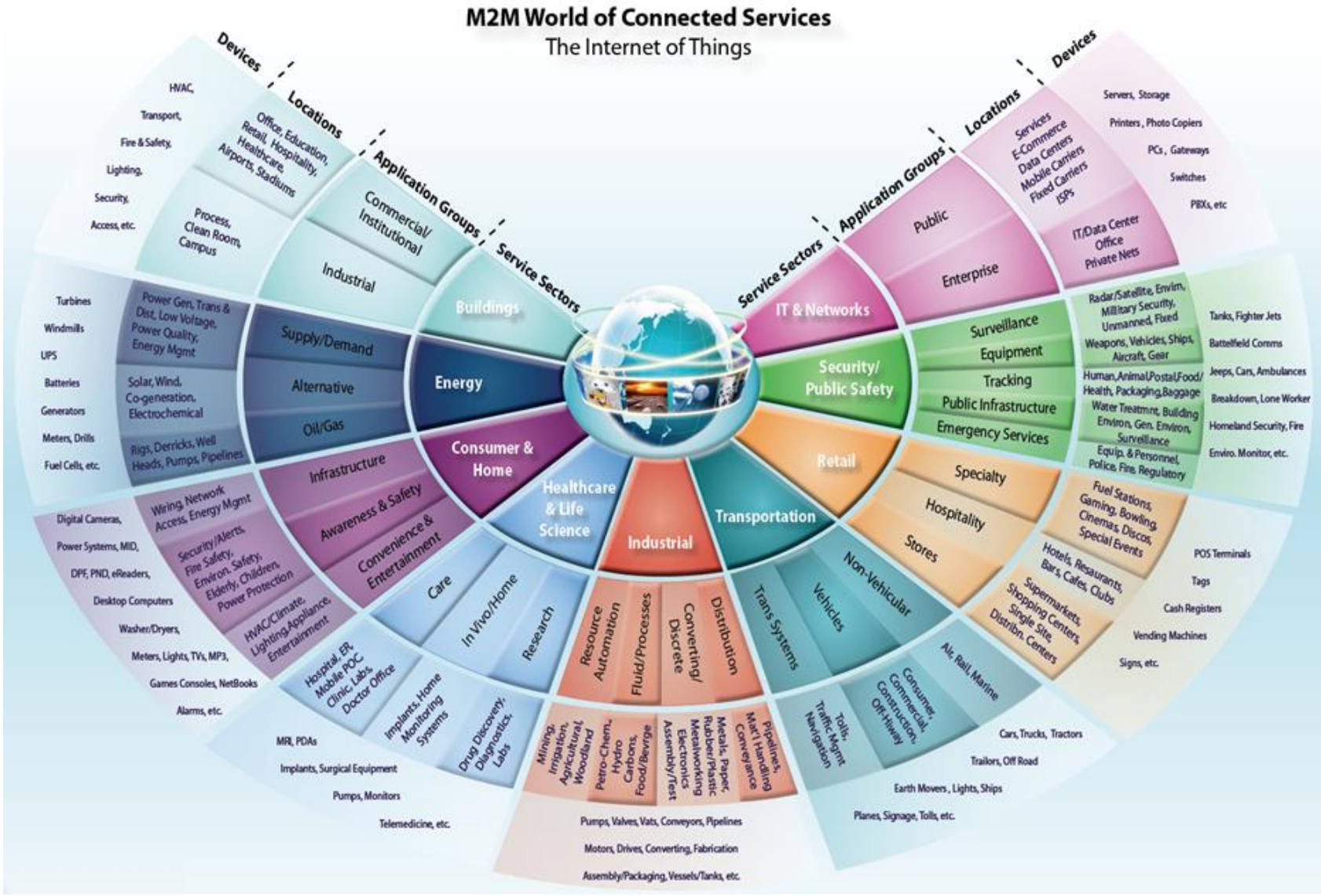
- Experts estimate that the IoT will consist of about 30 billion objects by 2020. It is also estimated that the global market value of IoT will reach $7.1 trillion by 2020.

-Wikipedia

Various Names

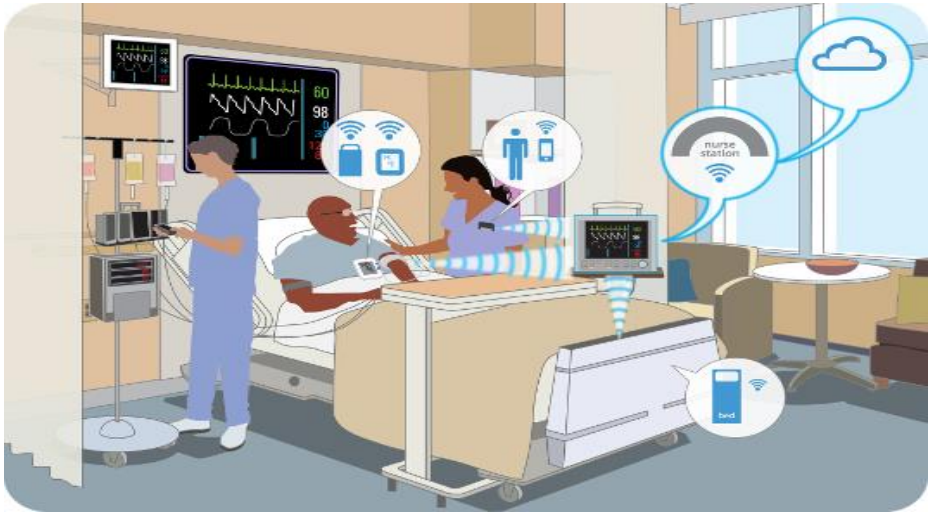
- M2M (Machine to Machine)
- “Internet of Everything” (Cisco Systems)
- “World Size Web” (Bruce Schneier)
- “Skynet”




M2M World of Connected Services
The Internet of Things
Devices
Locations
Application Groups
Service Sectors
HVAC,
Transport,
Fire & Safety,
Lighting,
Security,
Access, etc.
Office, Education, Retail, Hospitality, Healthcare, Airports, Stadiums
Process, Clean Room, Campus
Commercial/ Institutional
Industrial
Buildings
Energy
Consumer & Home
Healthcare & Life Science
Industrial
Transportation
Retail
Security/ Public Safety
IT & Networks
Public
Enterprise
Services
E-Commerce
Data Centers
Mobile Carriers
Fixed Carriers
ISPs
Servers, Storage
Printers, Photo Copiers
PCs, Gateways
Switches
PBXs, etc
IT/Data Center
Office
Private Nets
Turbines
Windmills
UPS
Batteries
Generators
Meters, Drills
Fuel Cells, etc.
Power Gen, Trans & Dist, Low Voltage, Power Quality, Energy Mgmt
Solar, Wind, Co-generation, Electrochemical
Rigs, Derricks, Well Heads, Pumps, Pipelines
Supply/Demand
Alternative
Oil/Gas
Digital Cameras,
Power Systems, MID,
DPF, PND, eReaders,
Desktop Computers
Washer/Dryers,
Meters, Lights, TVs, MP3,
Games Consoles, NetBooks
Alarms, etc.
Wiring, Network Access, Energy Mgmt
Security/Alerts, Fire Safety, Environ. Safety, Elderly, Children, Power Protection
HVAC/Climate, Lighting,Appliance, Entertainment
Infrastructure
Awareness & Safety
Convenience & Entertainment
MR, PDAs
Implants, Surgical Equipment
Pumps, Monitors
Telemedicine, etc.
Hospital, ER, Mobile POC, Clinic, Labs, Doctor Office
Implants, Home Monitoring Systems
Drug Discovery, Diagnostics, Labs
Care
In Vivo/Home
Research
Pumps, Valves, Vats, Conveyors, Pipelines
Motors, Drives, Converting, Fabrication
Assembly/Packaging, Vessels/Tanks, etc.
Mining, Irrigation, Agricultural, Woodland
Petro-Chem, Hydro Carbons, Food/Bevrge
Electronics Assembly/Test
Metals, Paper, Rubber/Plastic Metalworking
Matl Handling Conveyance
Pipelines
Resource Automation
Fluid/Processes
Converting/ Discrete
Distribution
Planes, Signage, Tolls, etc.
Earth Movers, Lights, Ships
Trailors, Off Road
Cars, Trucks, Tractors
Tolls, Traffic Mgmt, Navigation
Consumer, Commercial, Construction, Off-Hiway
Air, Rail, Marine
Trans Systems
Vehicles
Non-Vehicular
Signs, etc.
Vending Machines
Cash Registers
Tags
POS Terminals
Hotels, Resaurants, Bars, Cafes, Clubs
Supermarkets, Shopping Centers, Single Site, Distribn. Centers
Fuel Stations, Gaming, Bowling, Cinemas, Discos, Special Events
Specialty
Hospitality
Stores
Radar/Satellite, Envirn, Military Security, Unmanned, Fixed
Weapons, Vehicles, Ships, Aircraft, Gear
Human,Animal,Postal,Food/ Health, Packaging,Baggage
Water Treatmnt, Building Environ, Gen. Environ, Surveillance
Equip. & Personnel, Police, Fire, Regulatory
Surveillance
Equipment
Tracking
Public Infrastructure
Emergency Services
Tanks, Fighter Jets
Battelfield Comms
Jeeps, Cars, Ambulances
Breakdown, Lone Worker
Homeland Security, Fire
Enviro. Monitor, etc.

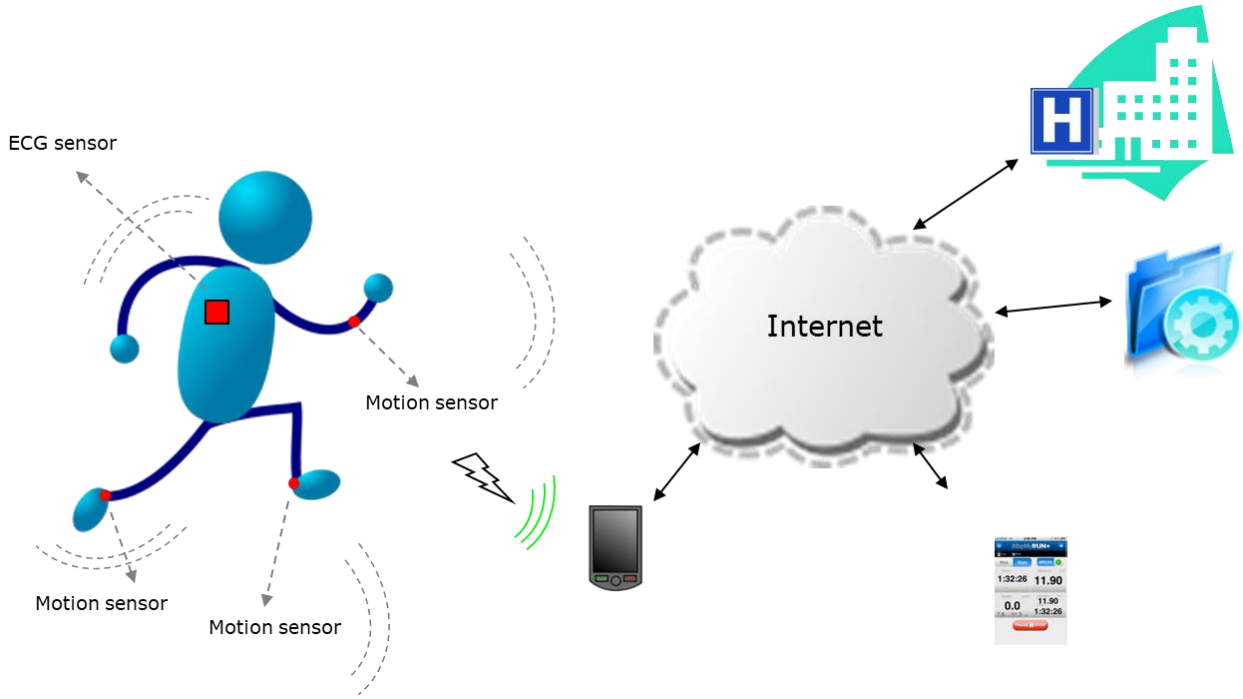
Where is IoT?

■ Everywhere



Trends

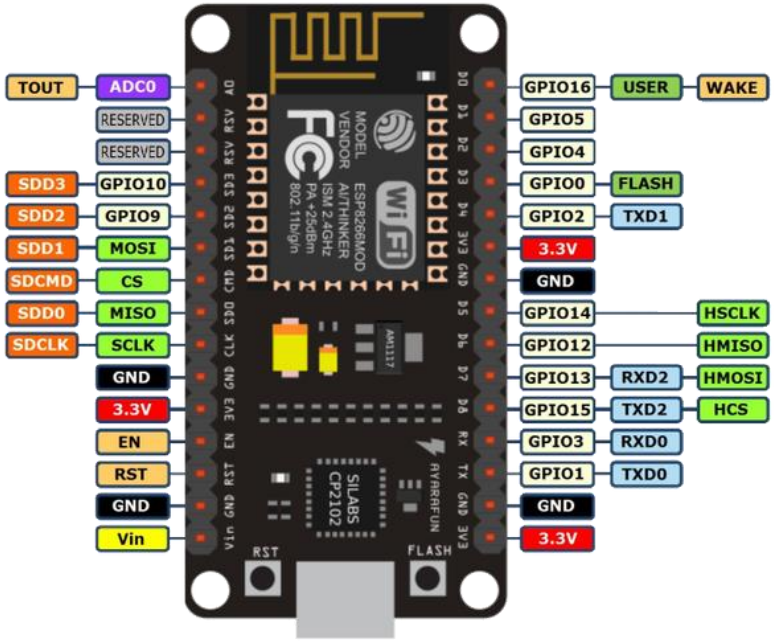
- Sensor devices are becoming widely available
- More “Things” are being connected
- People connecting to Things
- Things connecting to Things

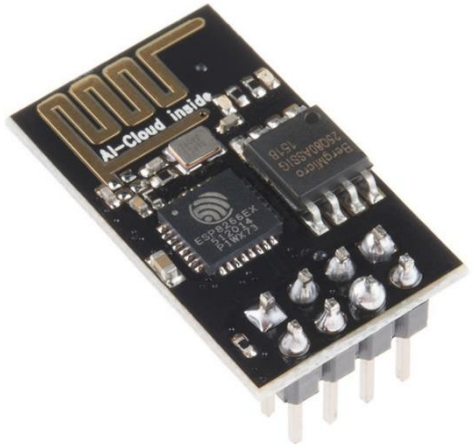


Lab1: NodeMCU를 이용한 IoT 실습

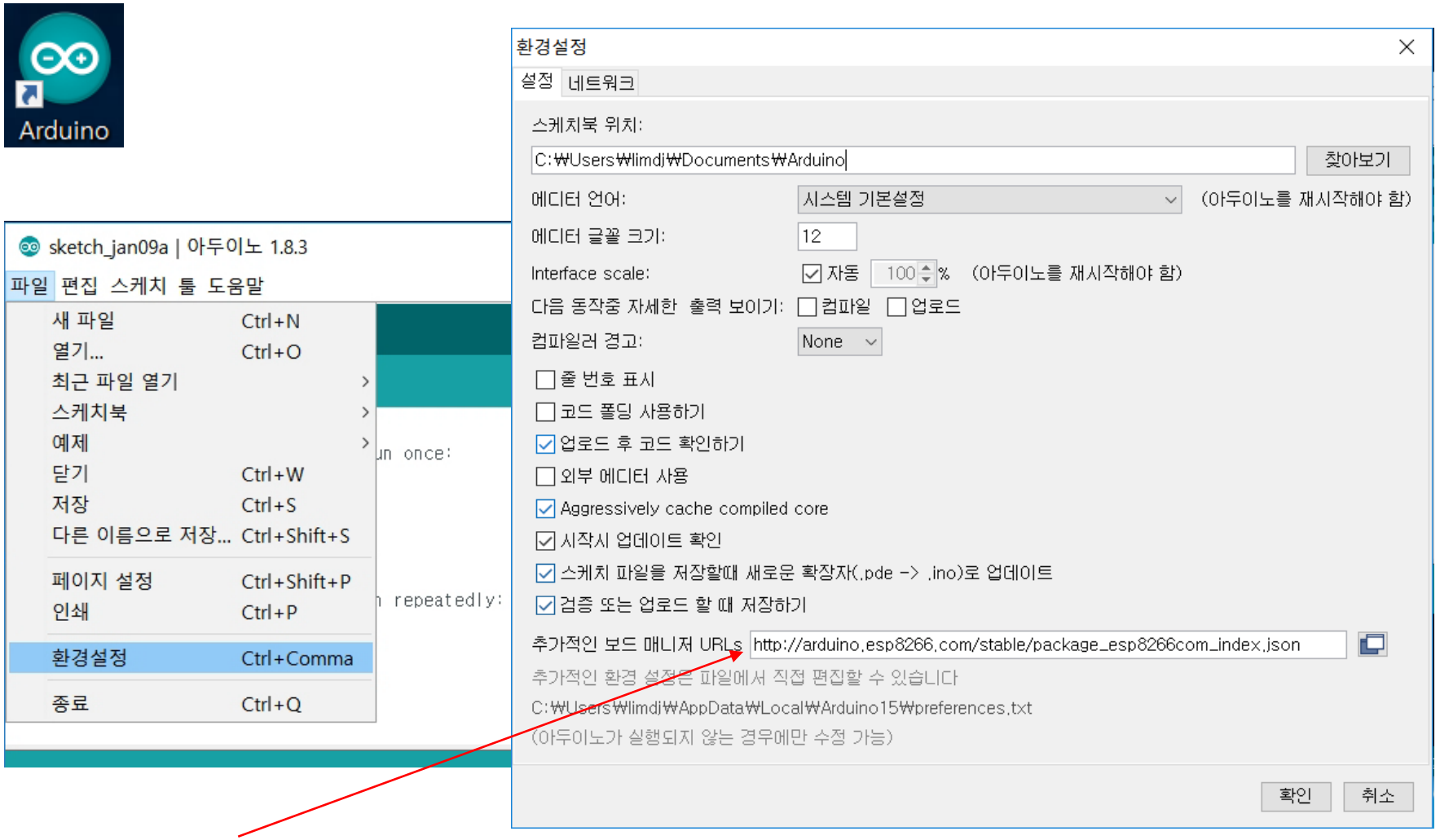
NodeMCU

- An open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.
- ESP8266: a low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability produced by Shanghai-based Chinese manufacturer, Espressif Systems.





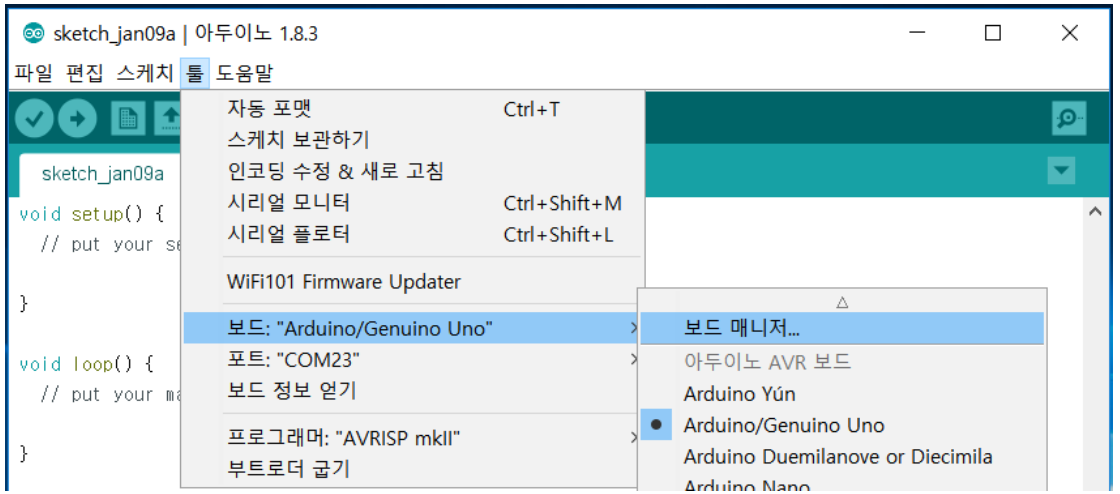
Arduino Programming for NodeMCU



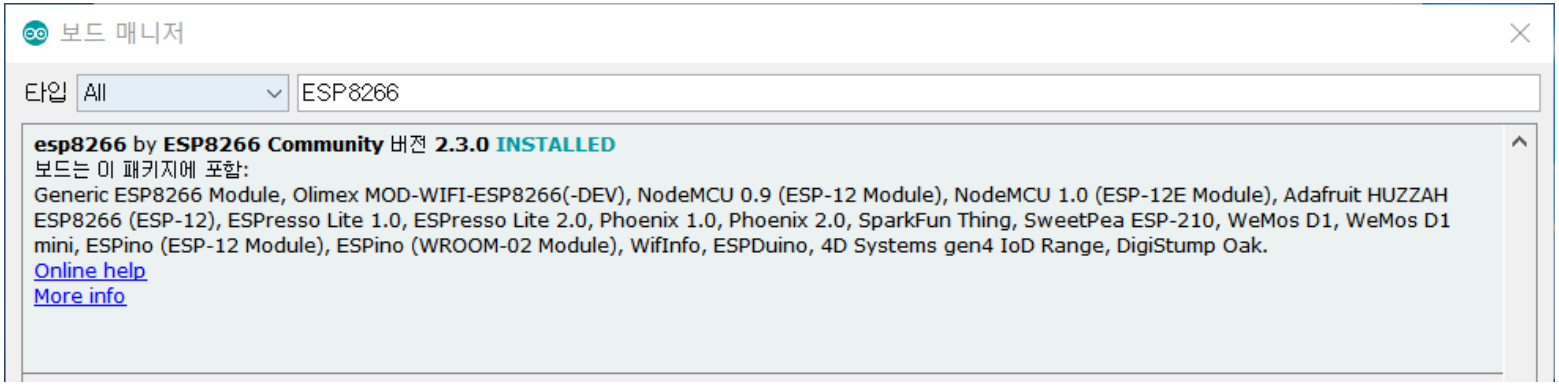
http://arduino.esp8266.com/stable/package_esp8266com_index.json

Add ESP8266 Board

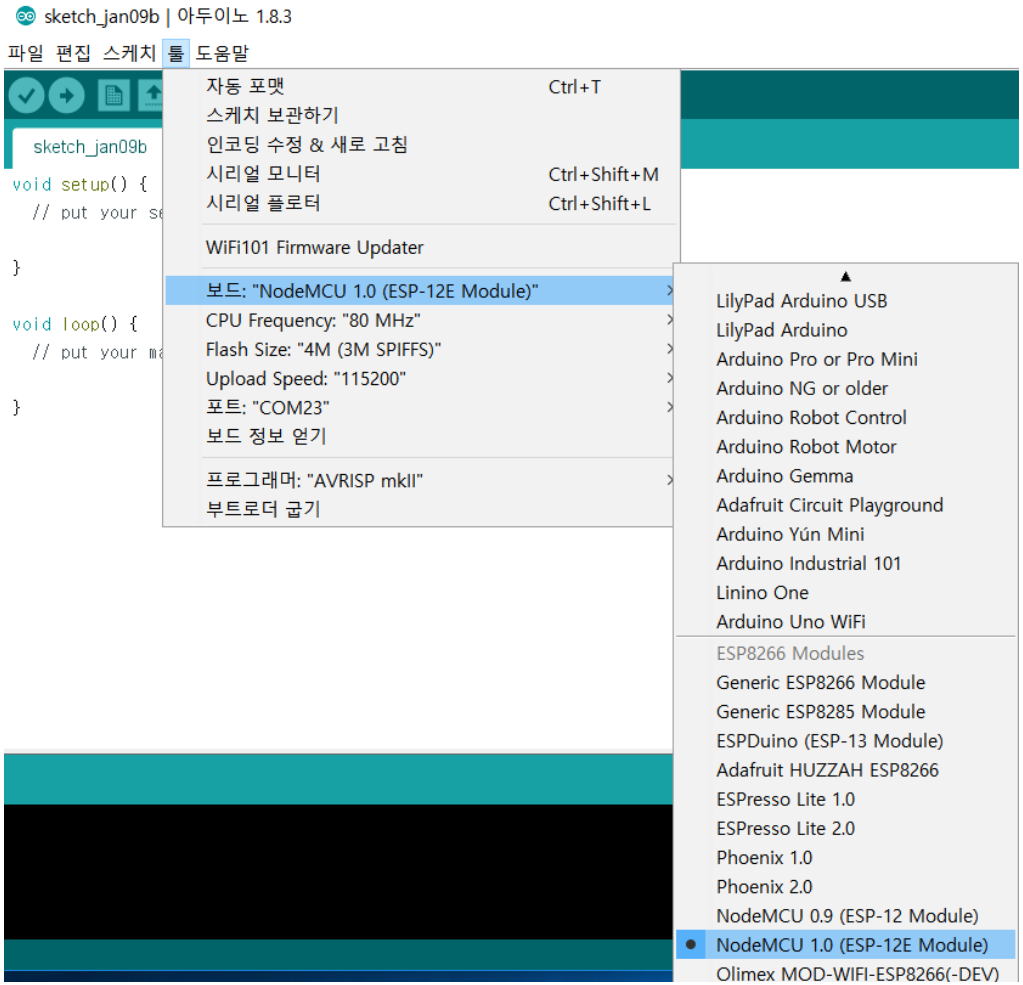
- Open Board Manager



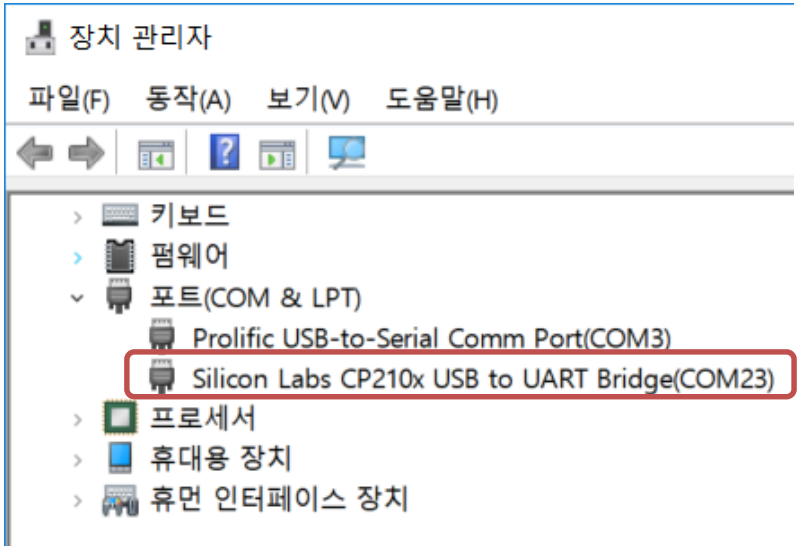
- Search ESP8266
- If ESP8266 is not installed, click More info and install the board

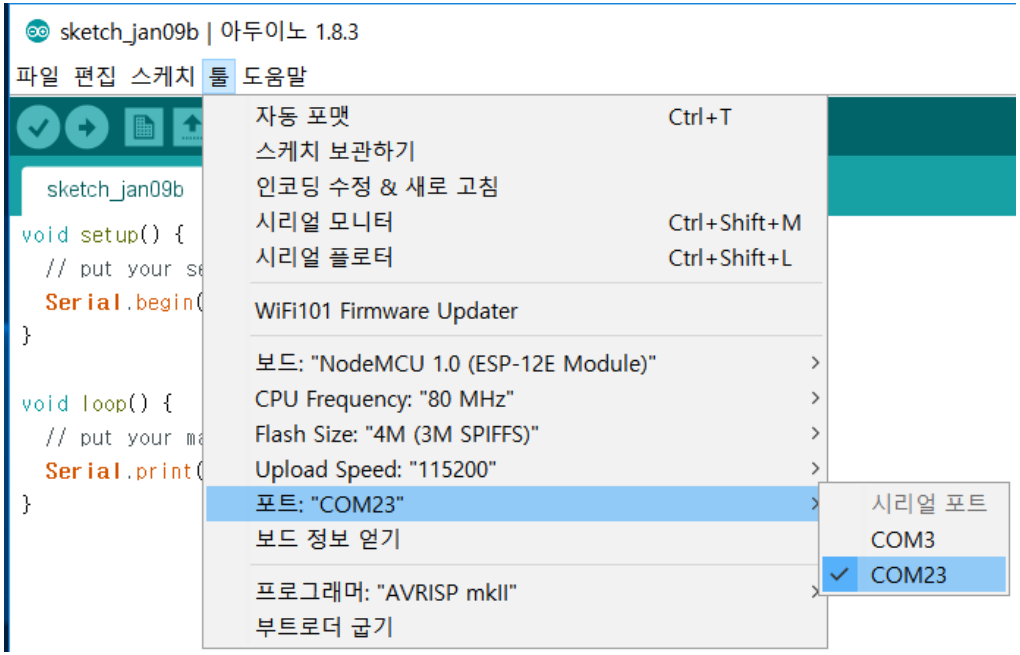


■ Select NodeMCU 1.0 (ESP-12E Module) from Board Manager



- NodeMCU와 PC를 microUSB 케이블을 이용해서 연결
- 장치 관리자를 열어서 CP210x 장치의 COM 번호 확인 후, 아두이노 환경에서 포트 COM 번호 설정



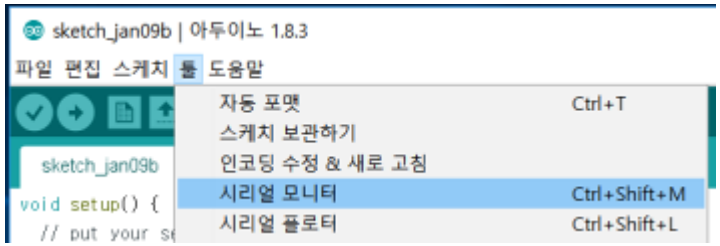


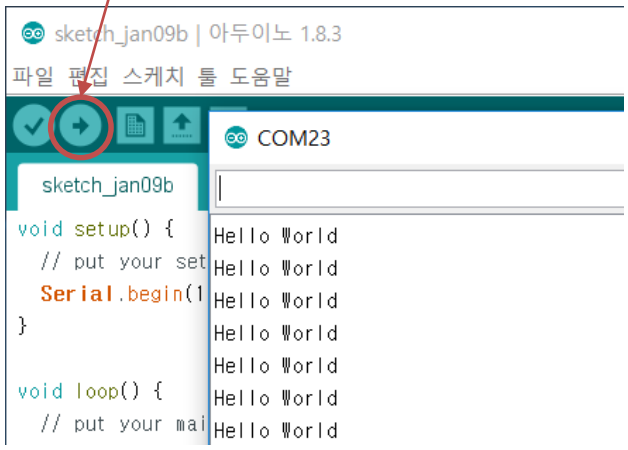
Hello World 예제

■ 아래의 코드를 입력

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
}

void loop() {
  // put your main code here, to run repeatedly:
  Serial.print("Hello World₩n");
}
```

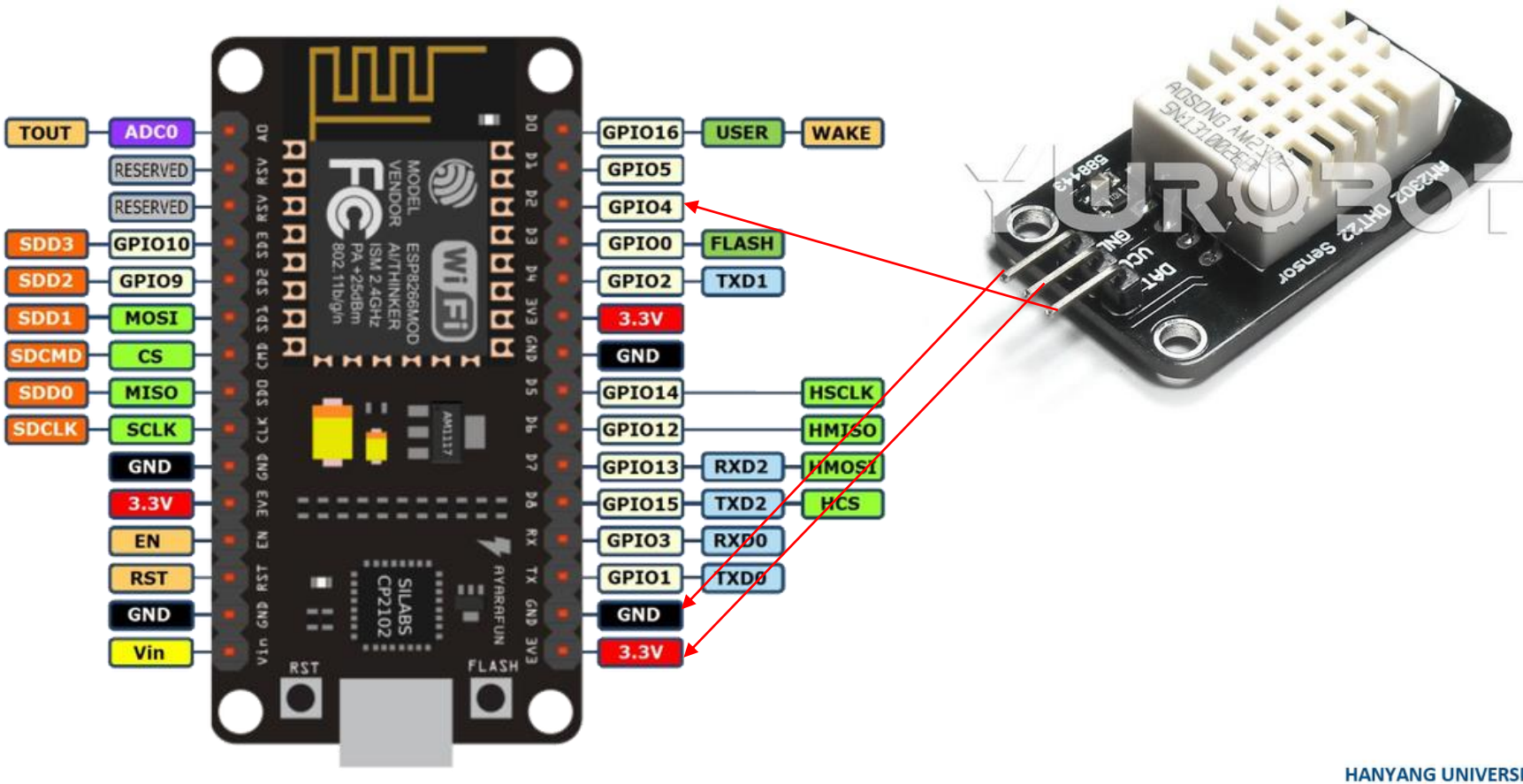
■ 시리얼 모니터를 열고, 업로드 버튼을 누른 후 결과 확인





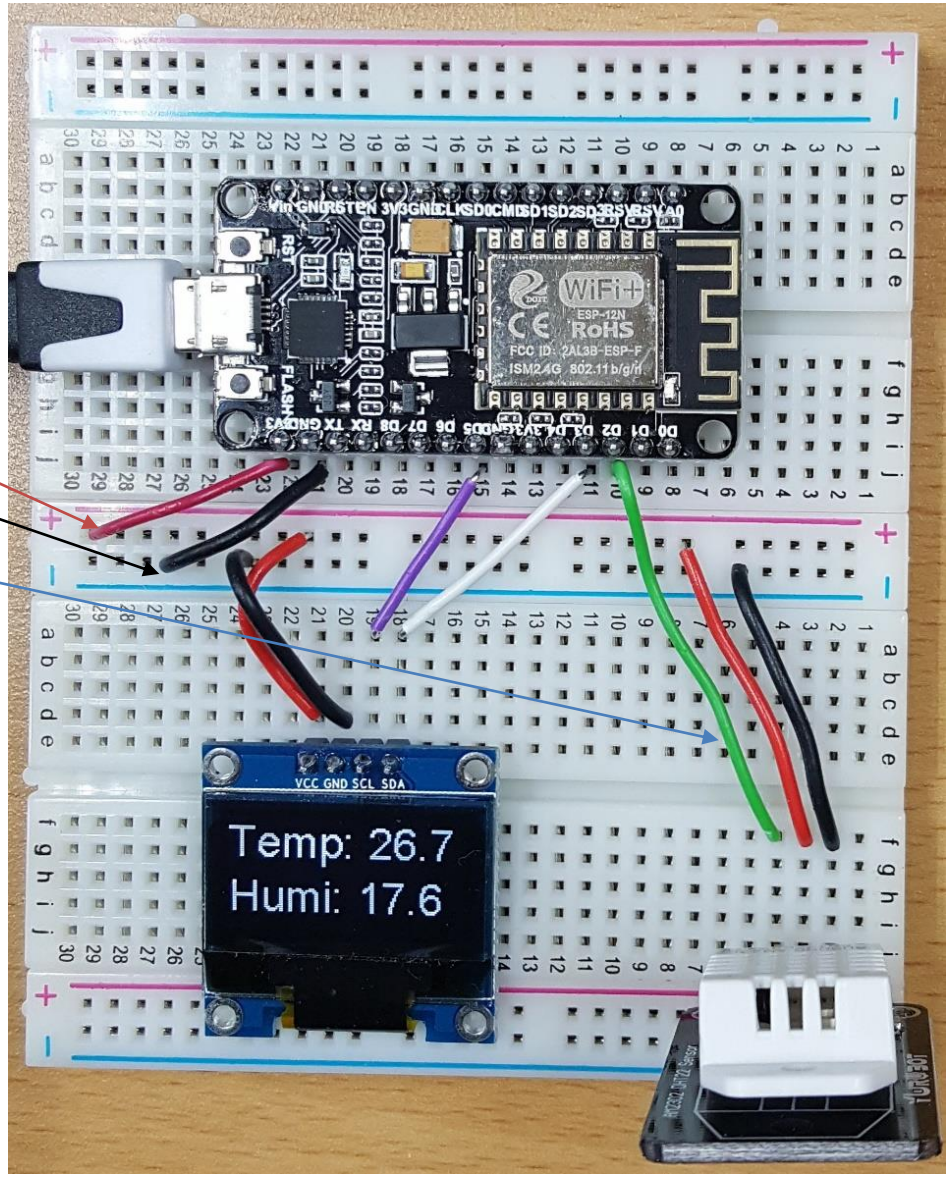
온습도 센서 DHT22 연결

■ VCC->3.3V, GND->GND, DAT->D2(GPIO4)



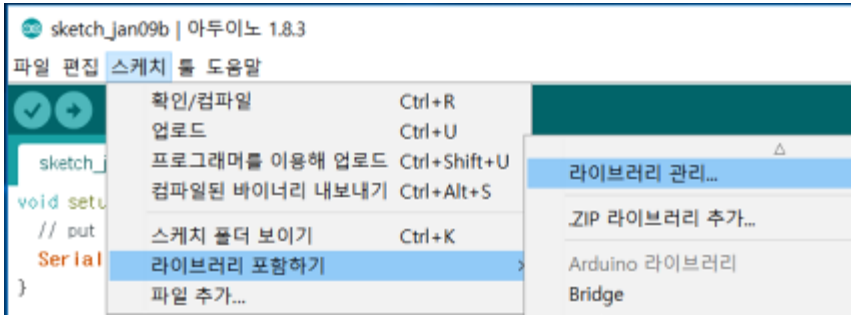
브레드 보드

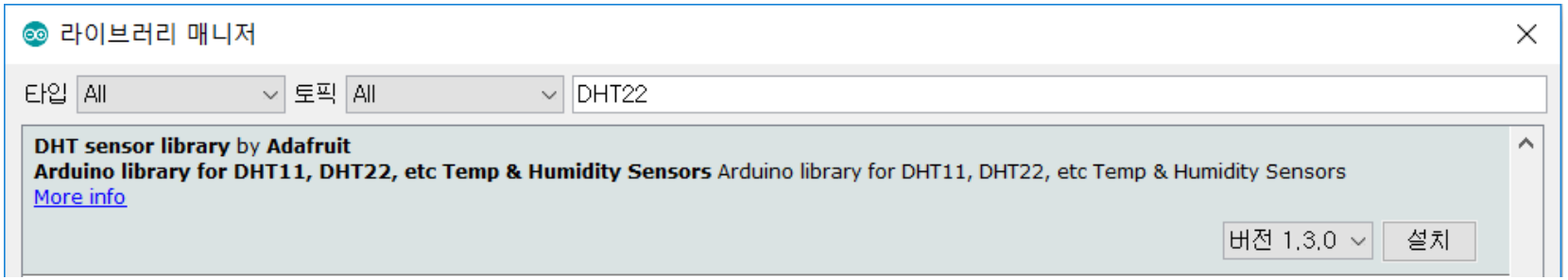
- **먼저 DHT22 연결, LCD는 다음에 연결 예정**
  - 3.3V 연결
  - GND 연결
  - Serial data 연결

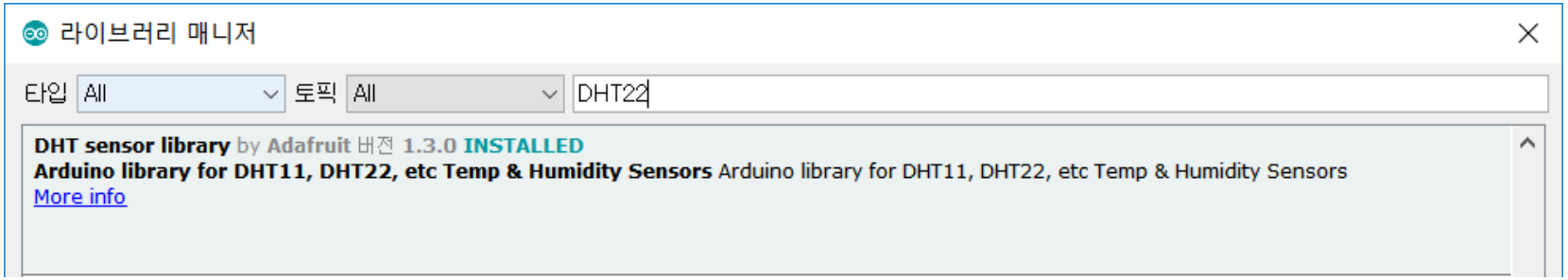


DHT Sensor Library

■ 라이브러리 매니저 열어서 DHT22로 검색 후, DHT Sensor library 설치 확인.



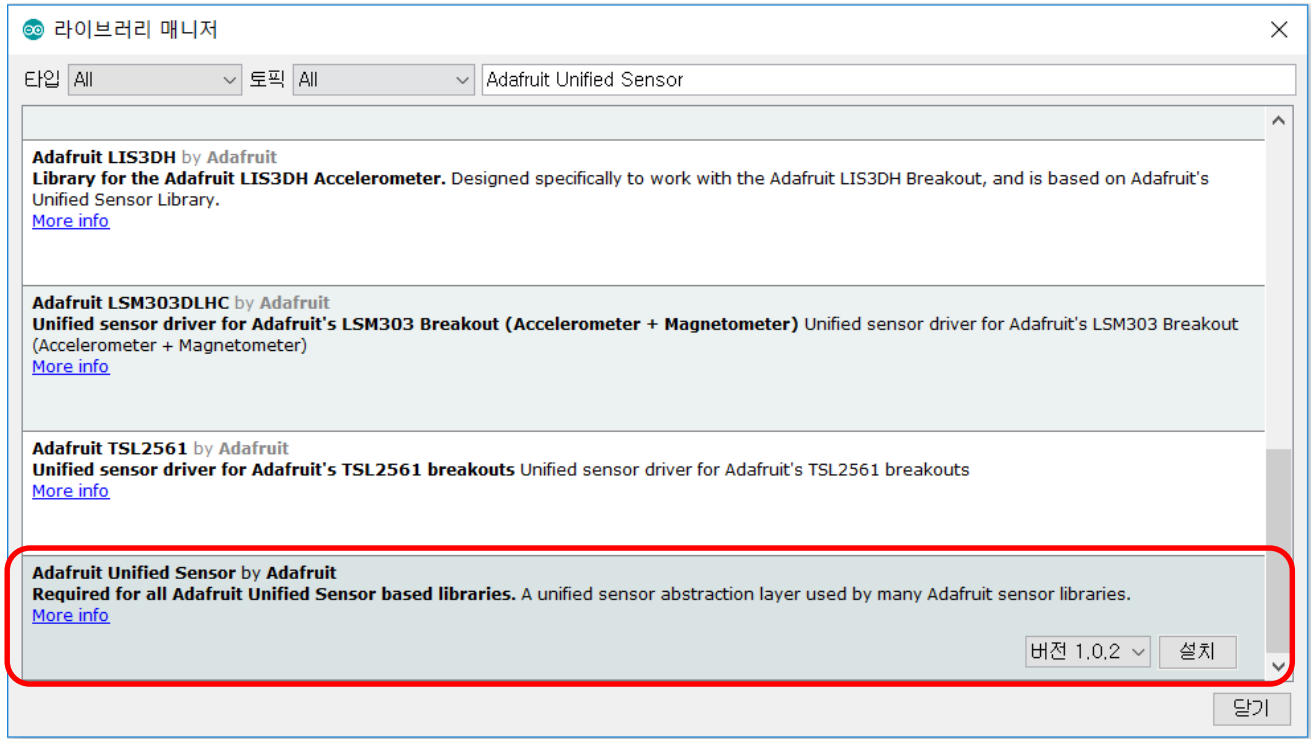




필요한 추가 라이브러리 설치

■ 라이브러리 매니저 열어서 Adafruit Unified Sensor로 검색 후, Adafruit Unified Sensor library 설치 확인.

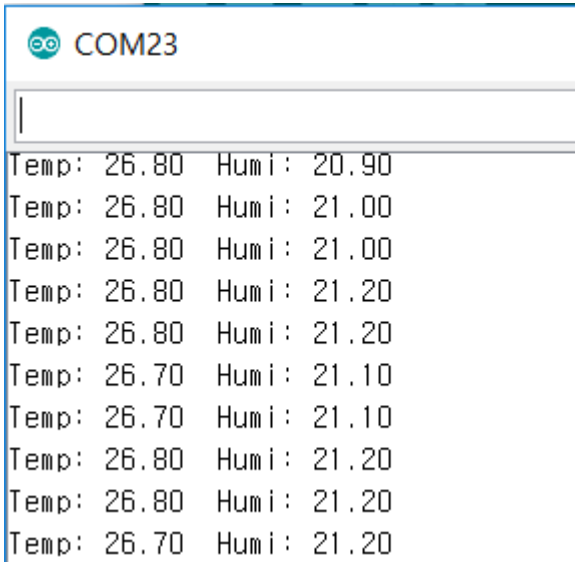
Adafruit Unified Sensor by Adafruit 버전 1.0.2 INSTALLED
Required for all Adafruit Unified Sensor based libraries. A unified sensor abstraction layer used by many Adafruit sensor libraries.
More info



온습도 센서 읽기

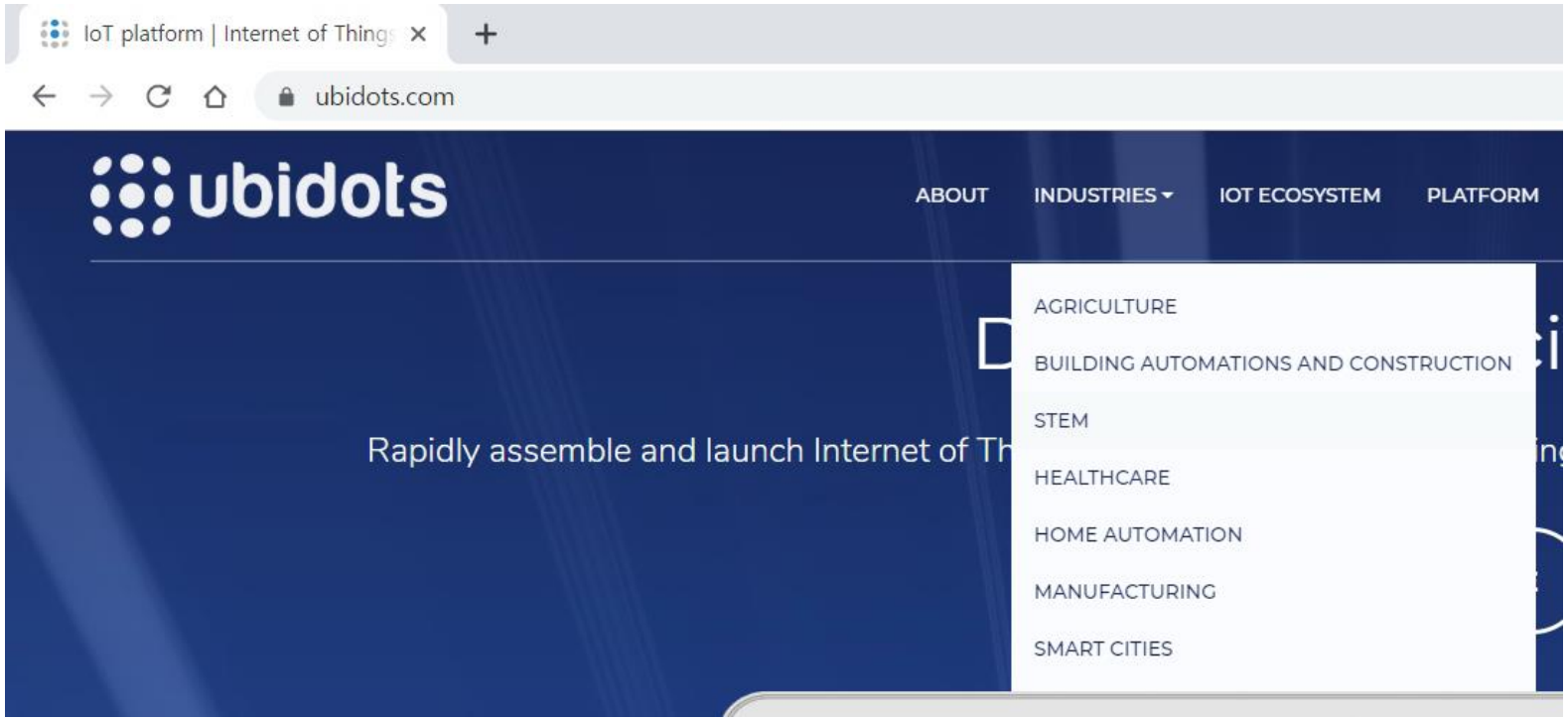
■ 아래의 코드를 입력, 업로드 후 실행 확인

```
#include <DHT.h>
#define DHTPIN D2
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
void setup() {
  Serial.begin(115200);
  dht.begin();
}
void loop() {
  float temp=dht.readTemperature();
  float humi=dht.readHumidity();
  Serial.print("Temp: ");
  Serial.print(temp);
  Serial.print("   Humi: ");
  Serial.println(humi);
  delay(1000);
}
```

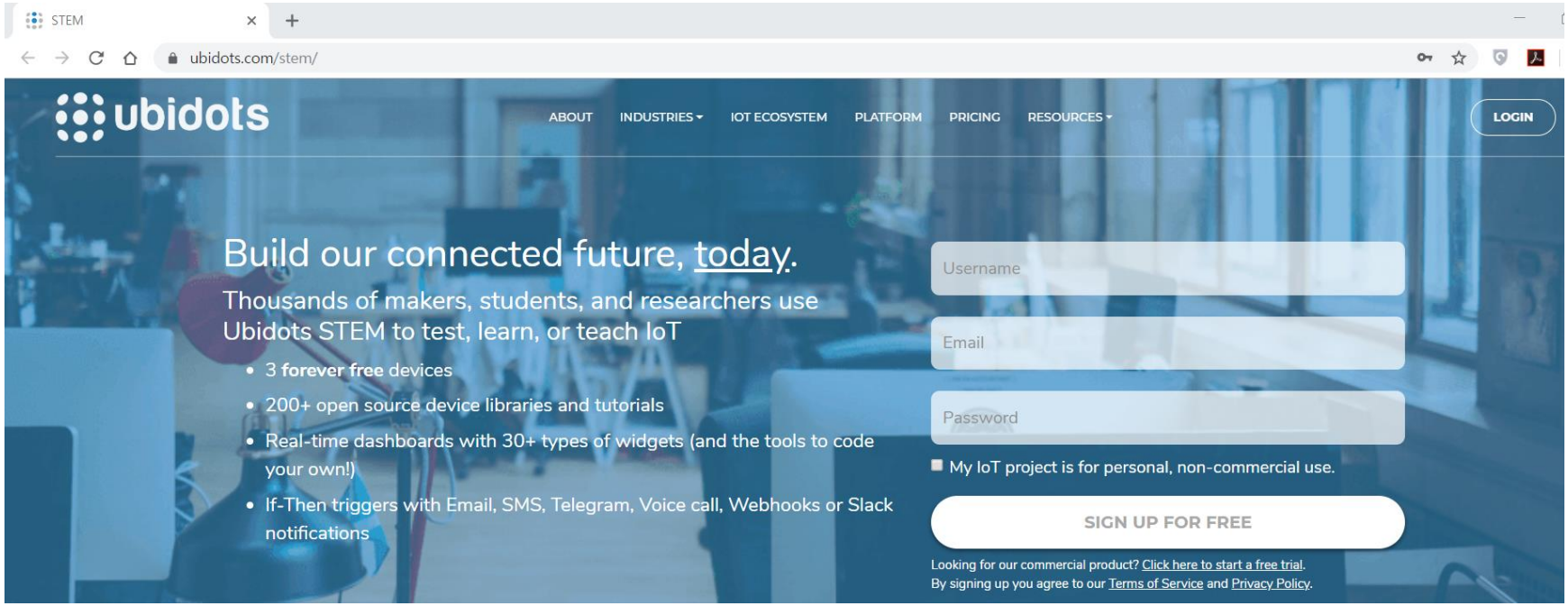


IoT 서비스 웹사이트 가입

ubidots.com
Select STEM from INDUSTRIES menu

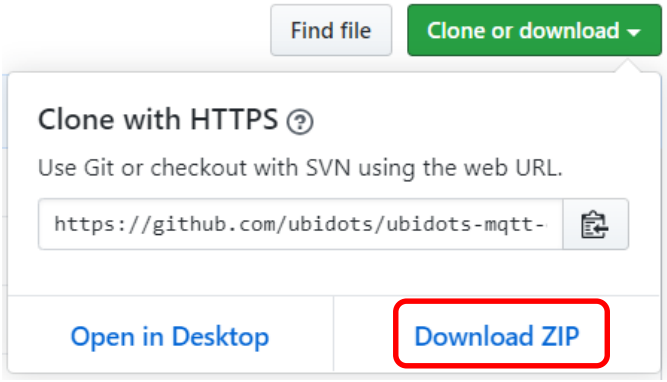


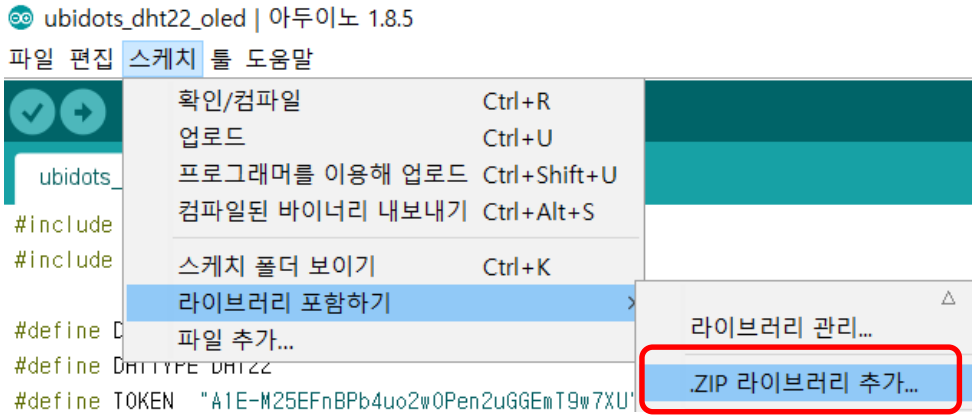
Sign up or login



Ubidots Library

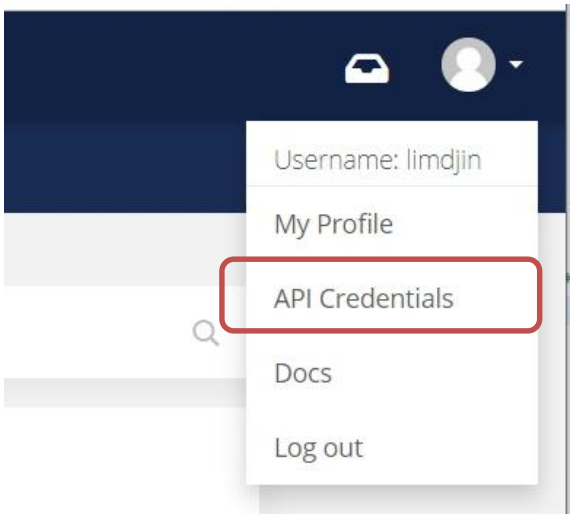
- Download the UbidotsMicroESP8266 library here https://github.com/ubidots/ubidots-mqtt-esp
- Select Download ZIP

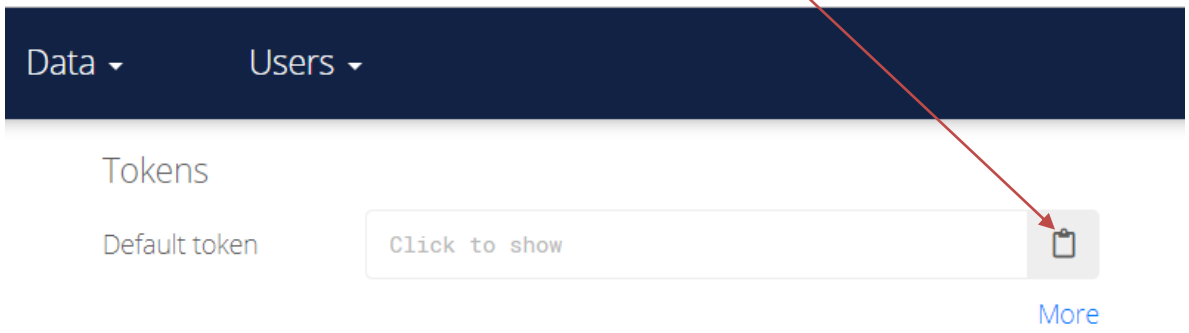




- Now, click on Sketch -> Include Library -> Add .ZIP Library
- Select ubidots-mqtt-esp-master.zip and then “Accept” or “Choose”
- Close the Arduino IDE and open it again.

Copy your Ubidots Token





온습도 센서 값 전송(1)

■ 아래의 코드를 입력, 업로드 및 실행.

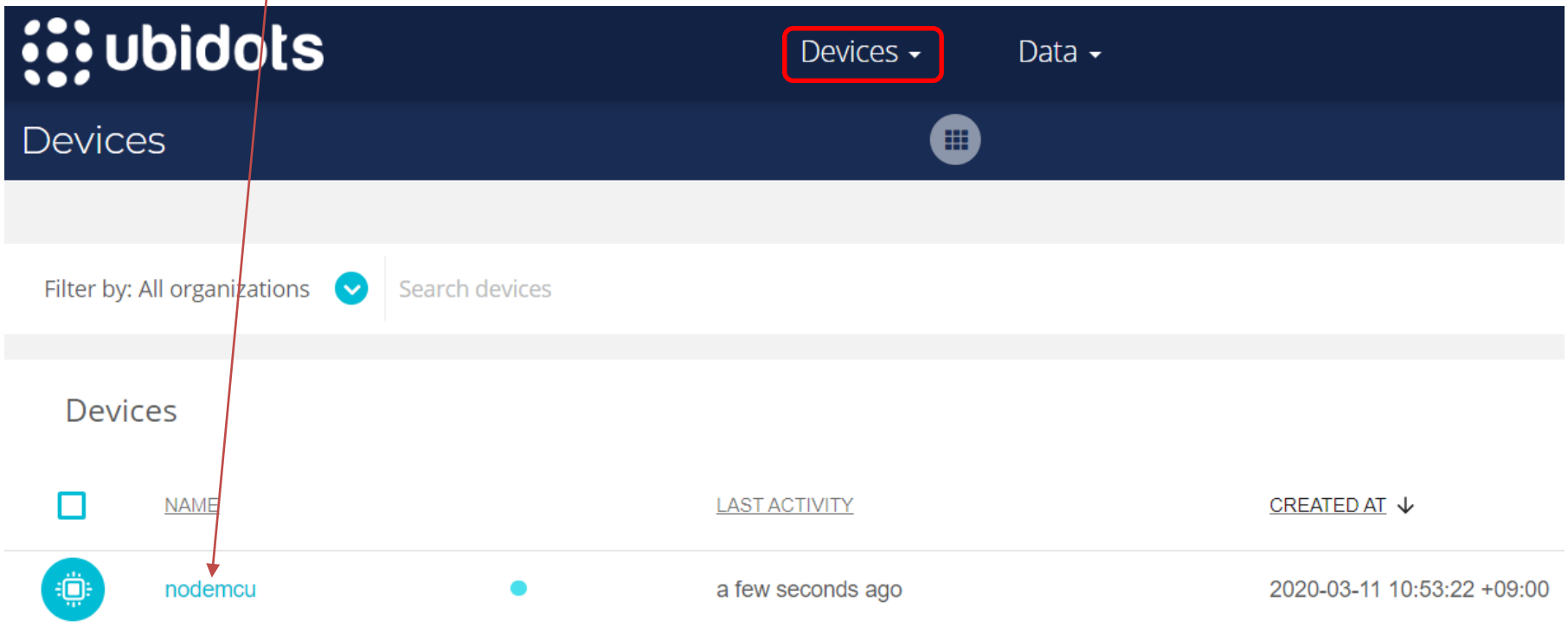
```
#include <DHT.h>
#include "UbidotsESPMQTT.h"

#define DHTPIN D2
#define DHTTYPE DHT22
#define TOKEN  "xxxxxxxxxxxxxx"  // Put here your Ubidots TOKEN
#define WIFI_SSID "xxxxxxxx"
#define WIFI_PASS "xxxxxxxx"

DHT dht(DHTPIN, DHTTYPE);
Ubidots client(TOKEN);

void callback(char* topic, byte* payload, unsigned int length) {
}
void setup() {
  Serial.begin(115200);
  dht.begin();
  delay(10);
  client.wifiConnection(WIFI_SSID, WIFI_PASS);
  client.begin(callback);
}
```

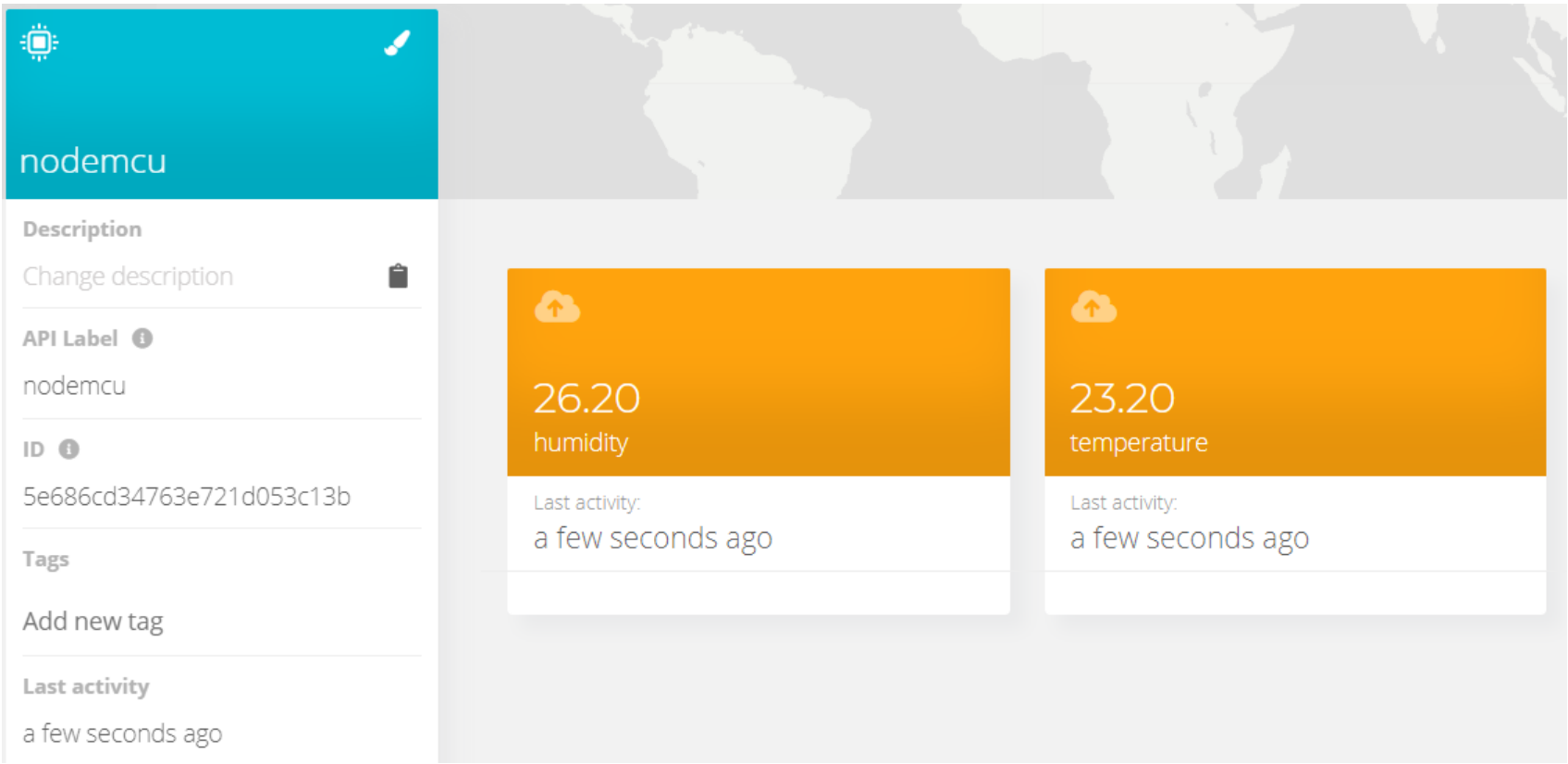
온습도 센서 값 전송(2)

```
void loop() {
  float temp=dht.readTemperature();
  float humi=dht.readHumidity();
  if(!client.connected()){
      client.reconnect();
  }
  client.add("Temperature", temp);
  client.add("Humidity", humi);
  client.ubidotsPublish("Nodemcu");

  Serial.print("Temp: ");
  Serial.print(temp);
  Serial.print("   Humi: ");
  Serial.println(humi);

  delay(1000*10);
}
```

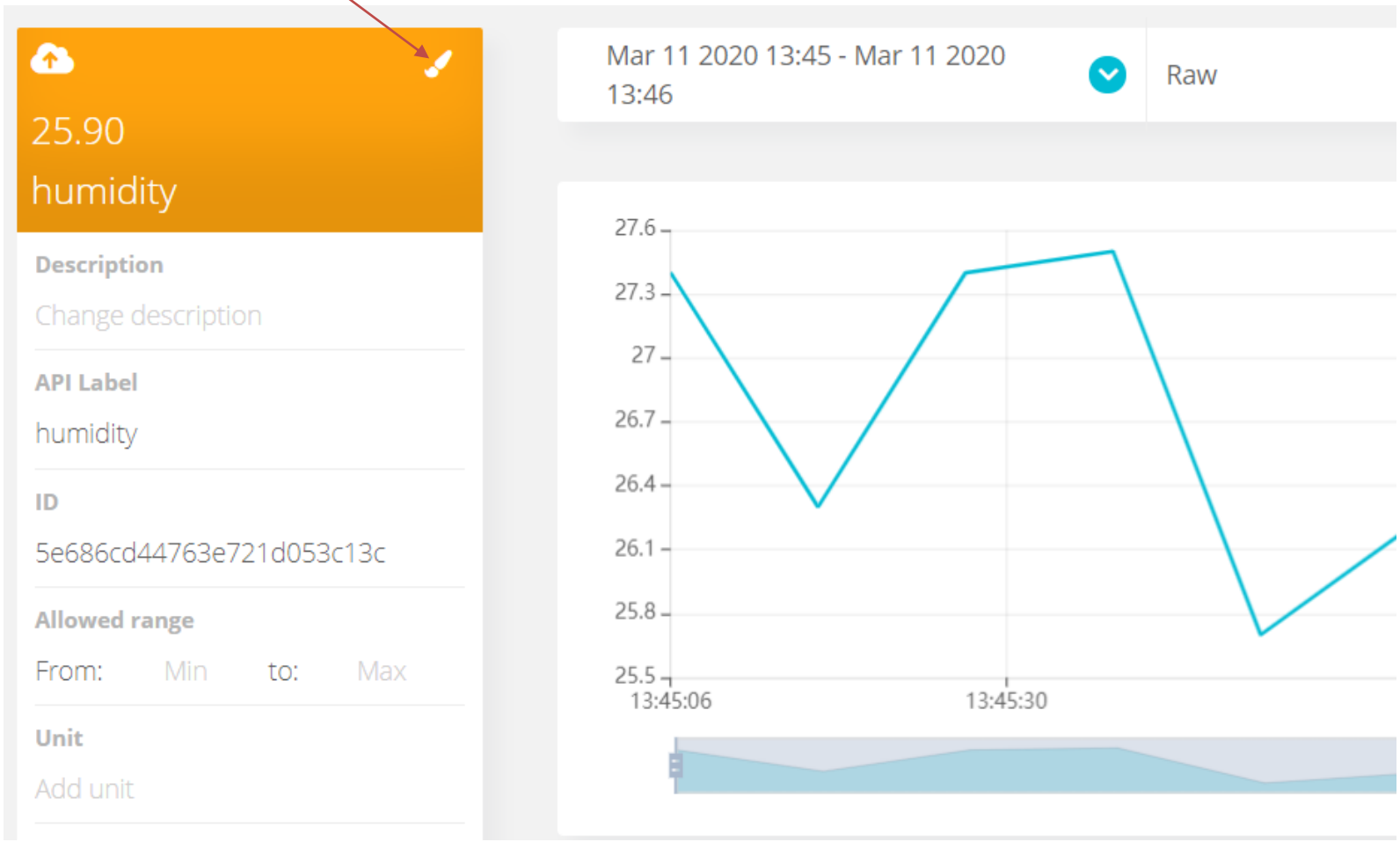
Devices

- Wait for the device to appear
- Double click the device

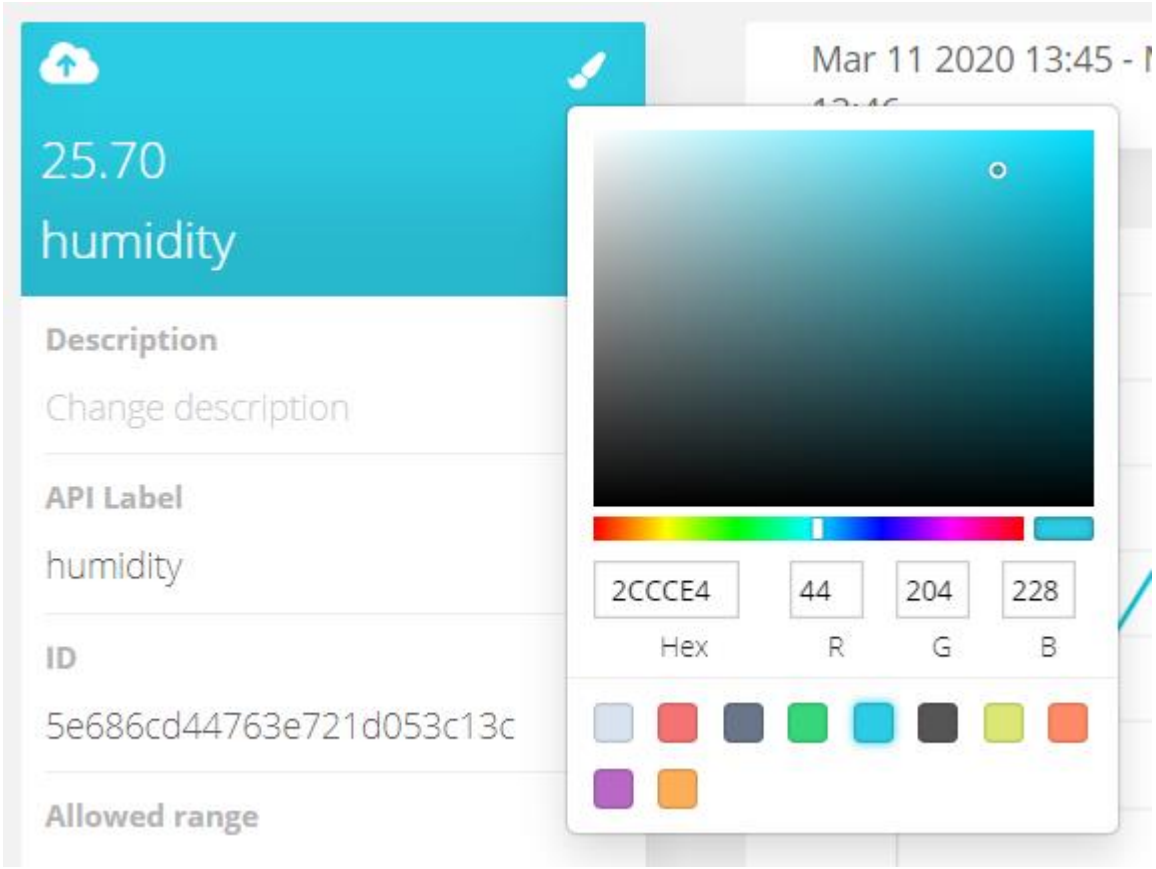


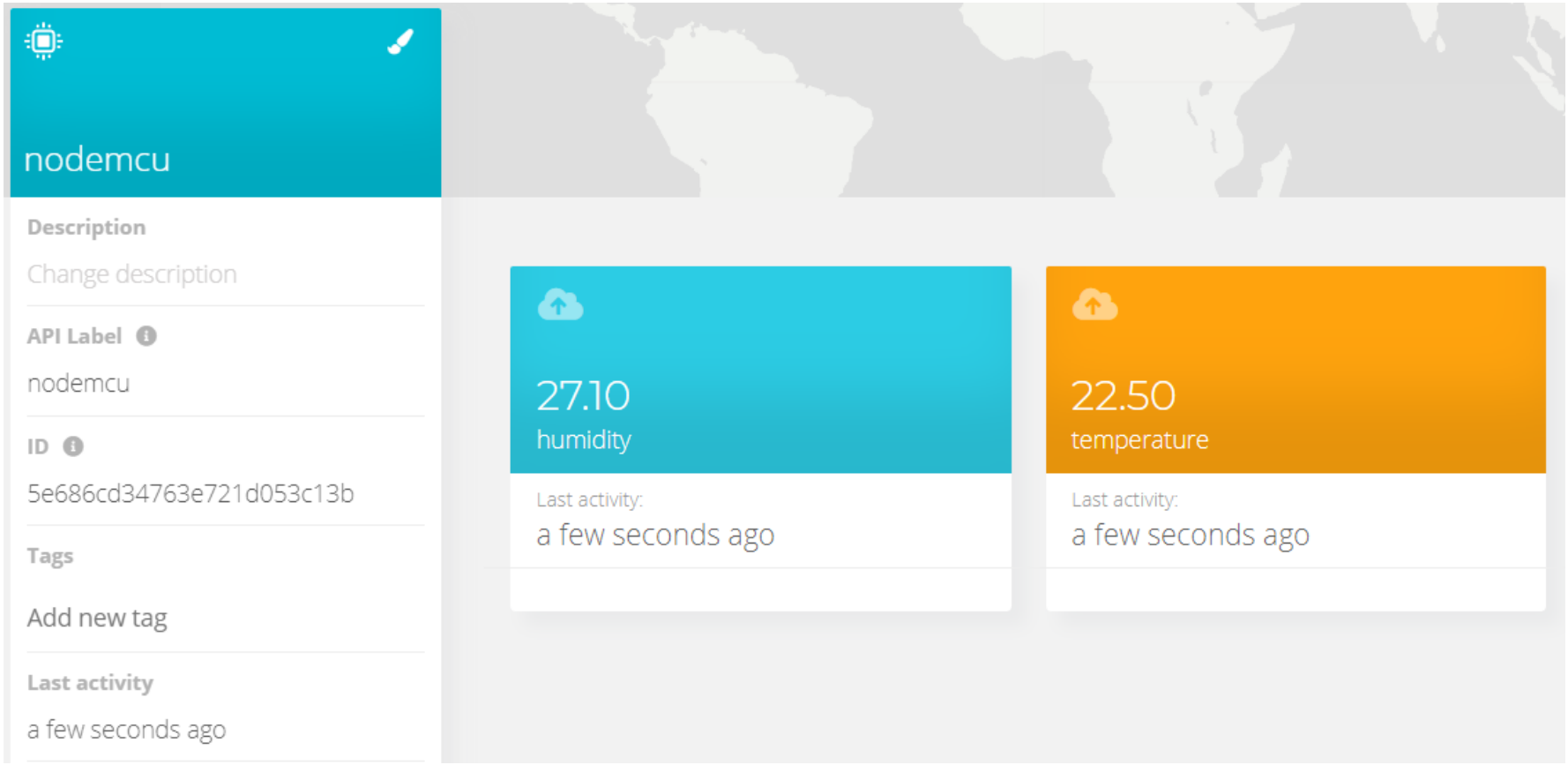

nodemcu
Description
Change description
API Label
nodemcu
ID
5e686cd34763e721d053c13b
Tags
Add new tag
Last activity
a few seconds ago
26.20
humidity
Last activity:
a few seconds ago
23.20
temperature
Last activity:
a few seconds ago

■ Click brush

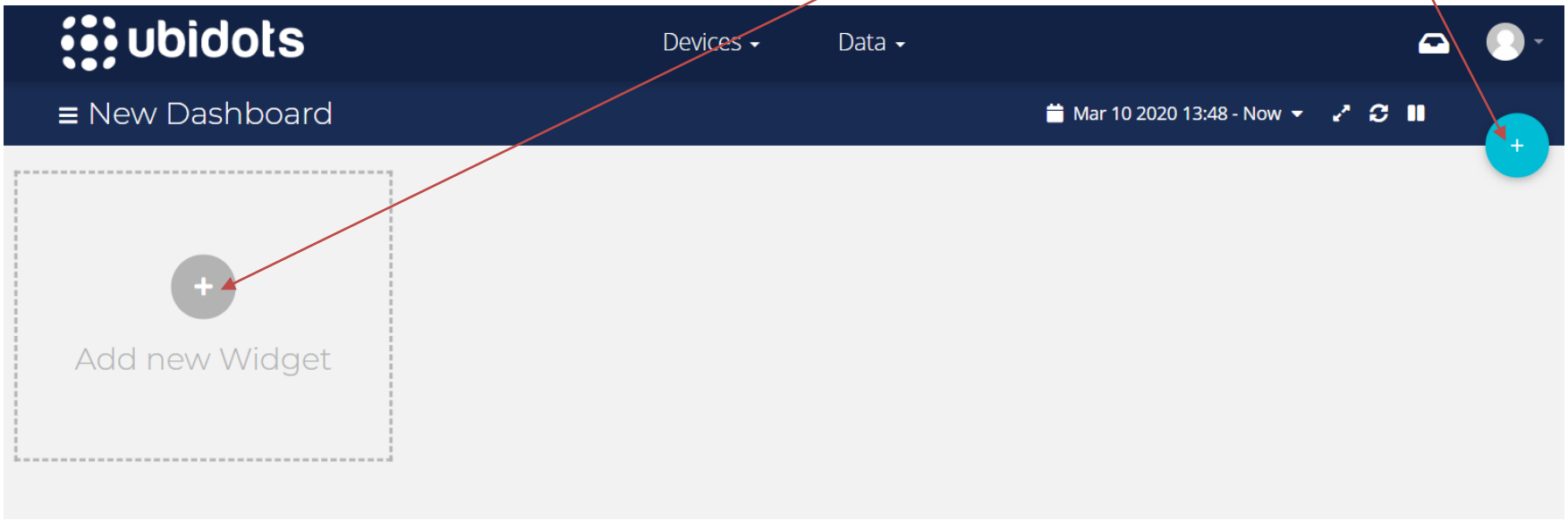

25.90
humidity
Description
Change description
API Label
humidity
ID
5e686cd44763e721d053c13c
Allowed range
From: Min to: Max
Unit
Add unit
Mar 11 2020 13:45 - Mar 11 2020 13:46
Raw
27.6
27.3
27
26.7
26.4
26.1
25.8
25.5
13:45:06
13:45:30

■ **Click desired color and click ←**

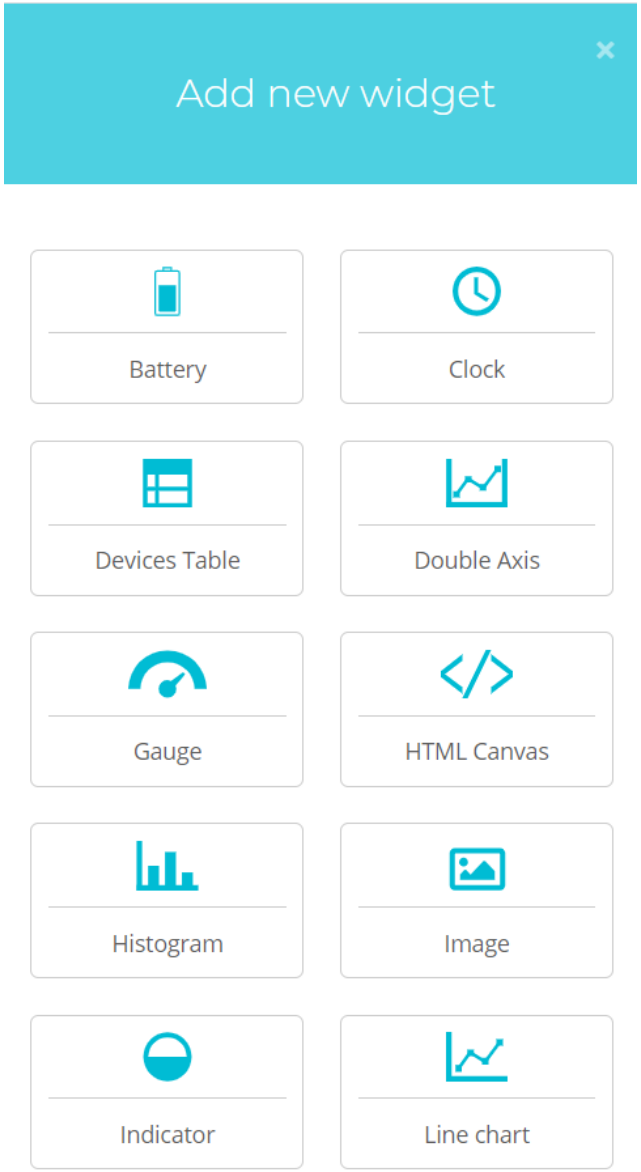



nodemcu
Description
Change description
API Label
nodemcu
ID
5e686cd34763e721d053c13b
Tags
Add new tag
Last activity
a few seconds ago
27.10
humidity
Last activity:
a few seconds ago
22.50
temperature
Last activity:
a few seconds ago

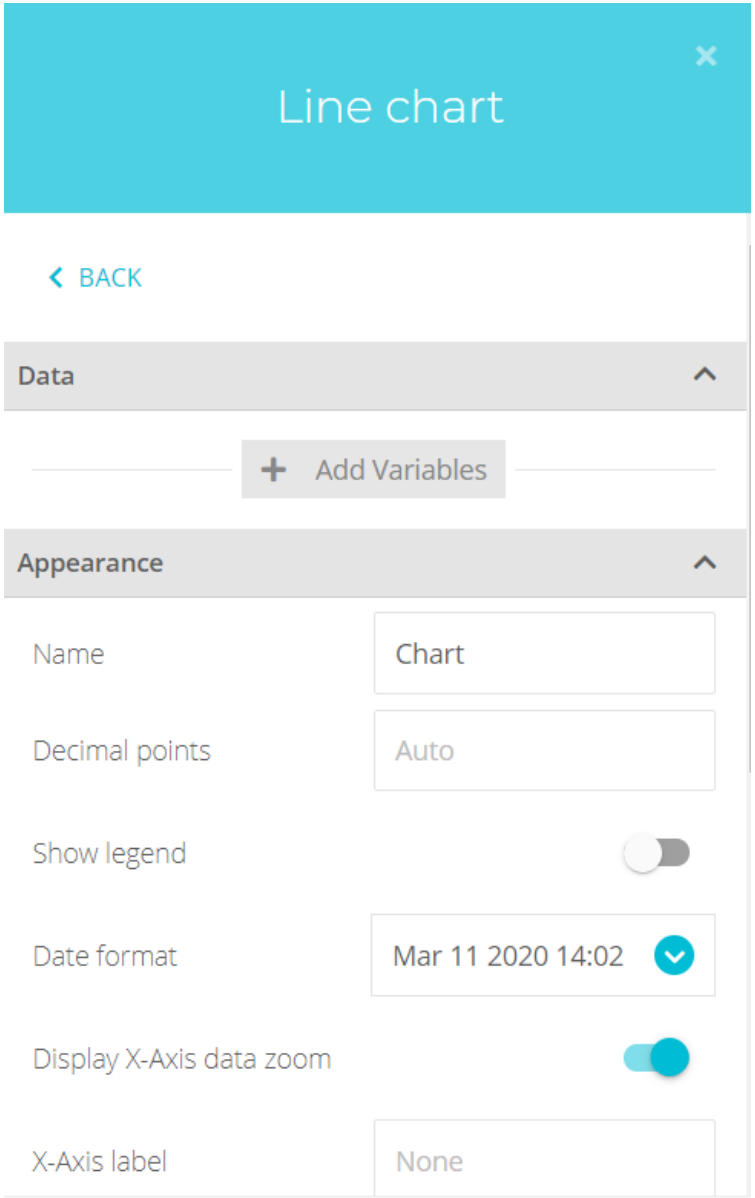
■ **Select Dashboards from Data menu and click “Add new Widget”.**



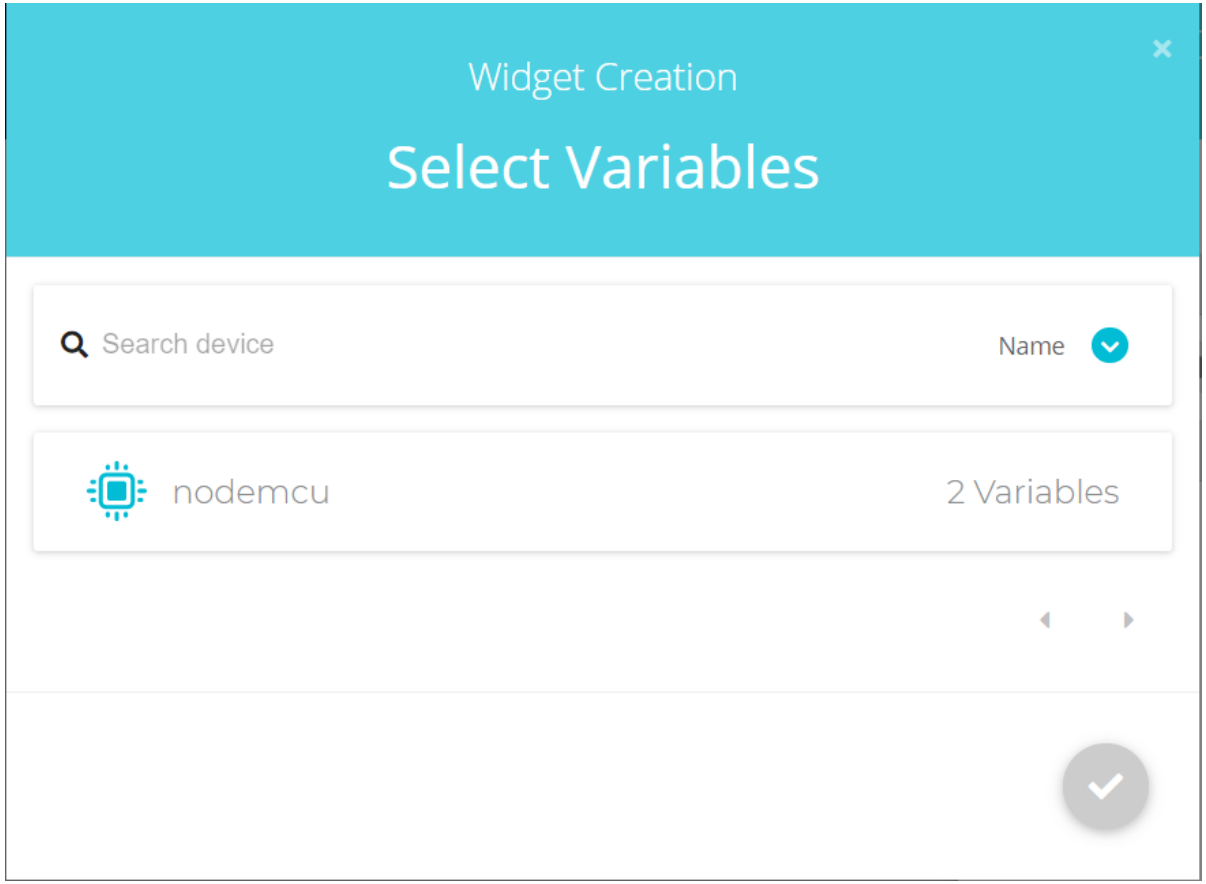
■ Select Line chart



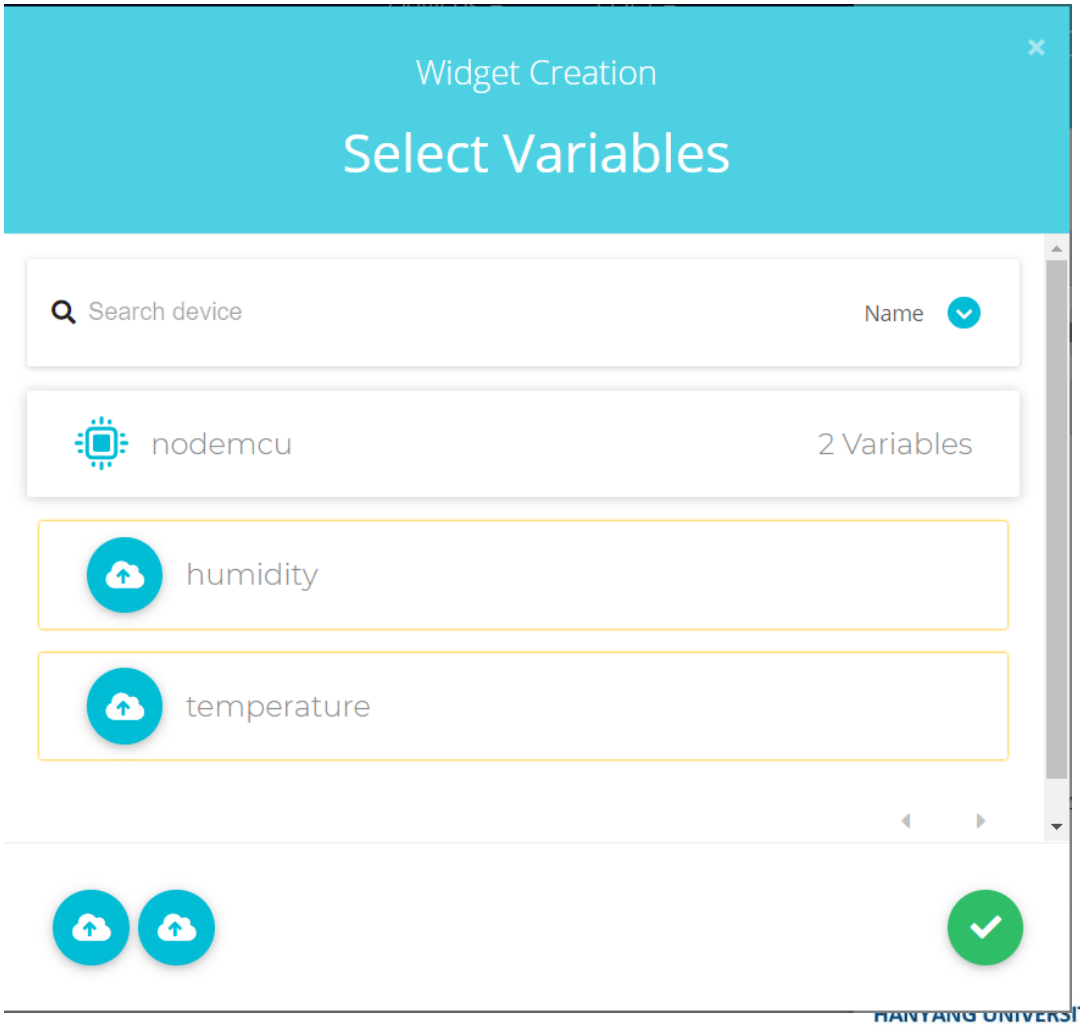
■ Add Variables



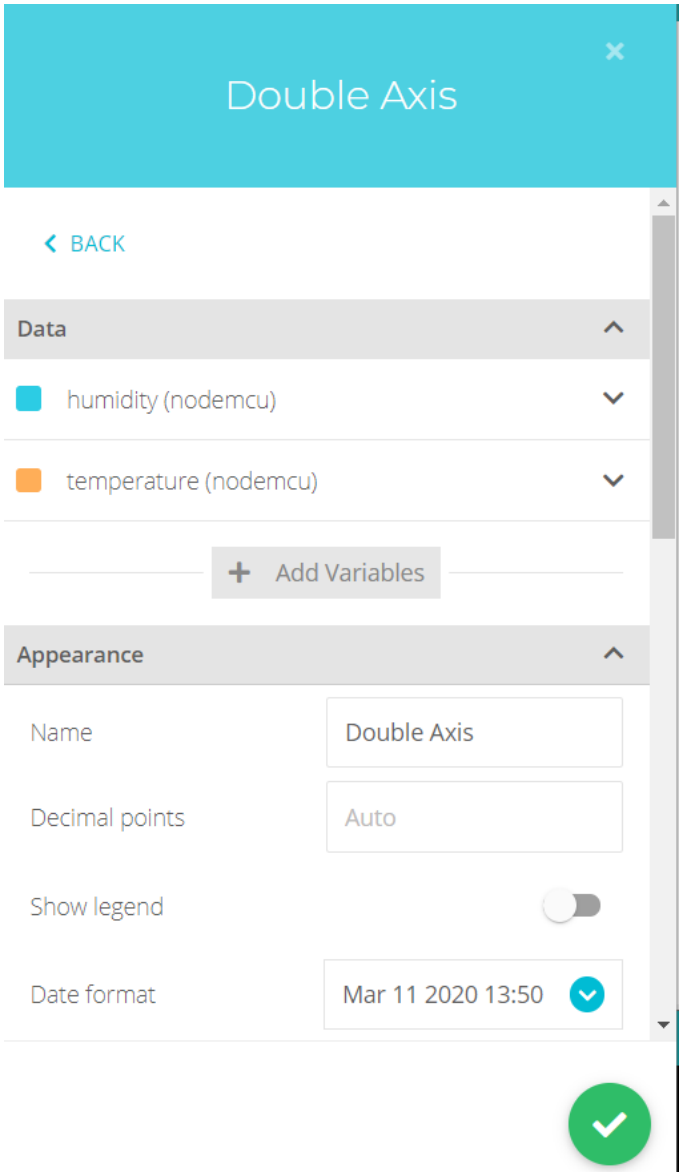
■ Click nodemcu

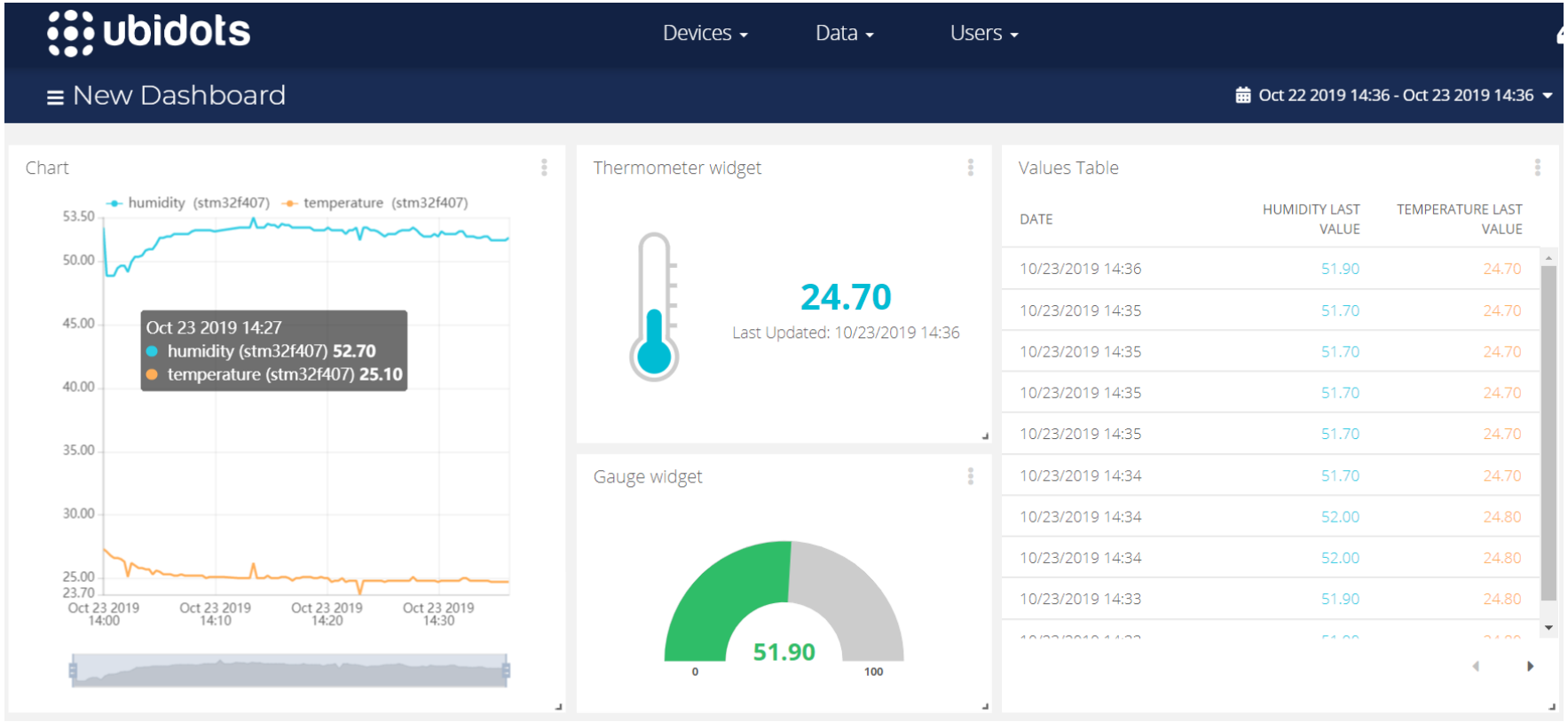


- Click humidity & temperature
- Click check mark

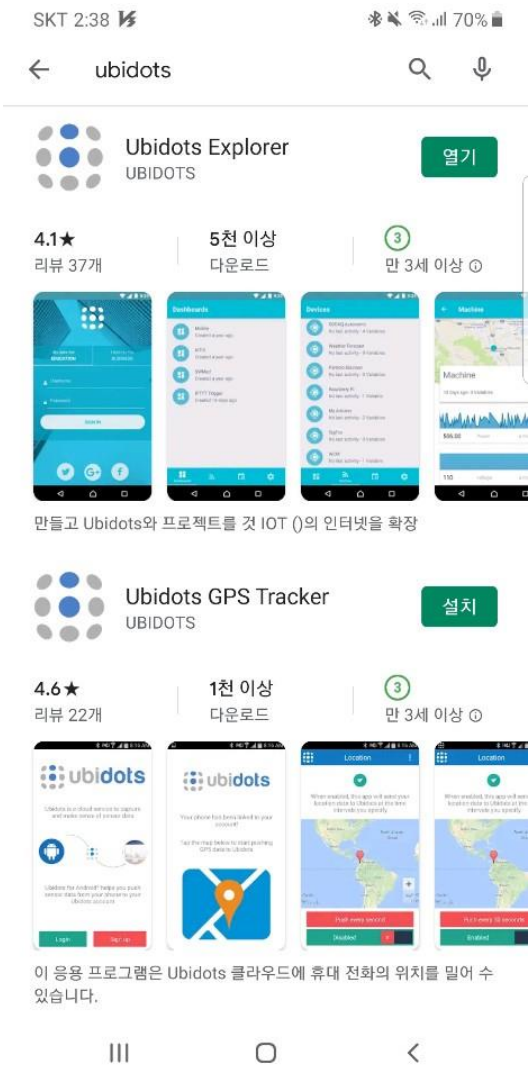


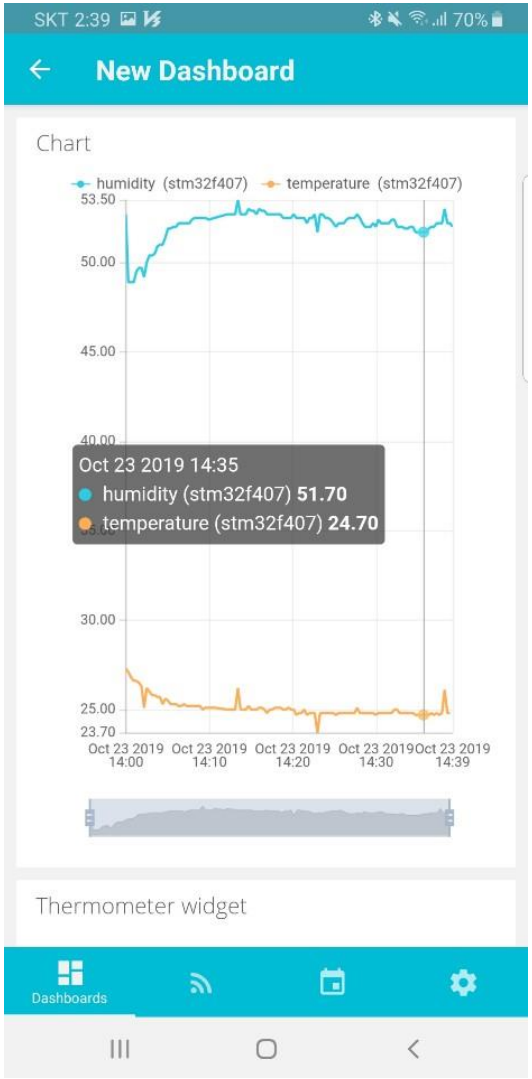
■ Click check mark




ubidots
Devices
Data
Users
New Dashboard
Oct 22 2019 14:36 - Oct 23 2019 14:36
Chart
humidity (stm32f407)
temperature (stm32f407)
53.50
50.00
45.00
40.00
35.00
30.00
25.00
23.70
Oct 23 2019 14:27
humidity (stm32f407) 52.70
temperature (stm32f407) 25.10
Oct 23 2019 14:00
Oct 23 2019 14:10
Oct 23 2019 14:20
Oct 23 2019 14:30
Thermometer widget
24.70
Last Updated: 10/23/2019 14:36
Gauge widget
51.90
0
100
Values Table
DATE
HUMIDITY LAST VALUE
TEMPERATURE LAST VALUE
10/23/2019 14:36 51.90 24.70
10/23/2019 14:35 51.70 24.70
10/23/2019 14:35 51.70 24.70
10/23/2019 14:35 51.70 24.70
10/23/2019 14:35 51.70 24.70
10/23/2019 14:34 51.70 24.70
10/23/2019 14:34 52.00 24.80
10/23/2019 14:34 52.00 24.80
10/23/2019 14:33 51.90 24.80

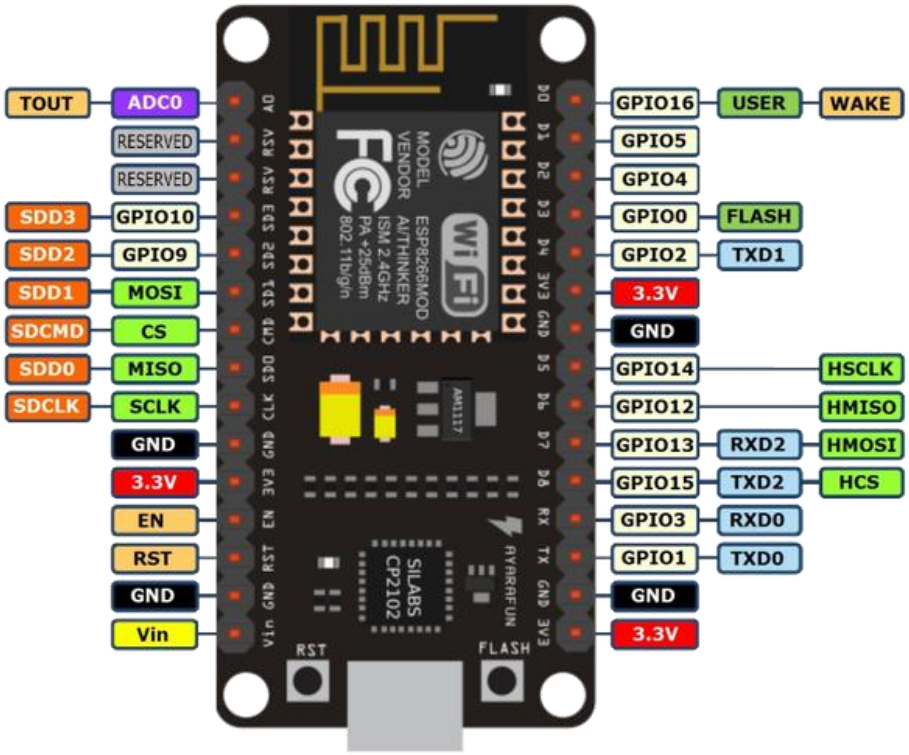
Smartphone App

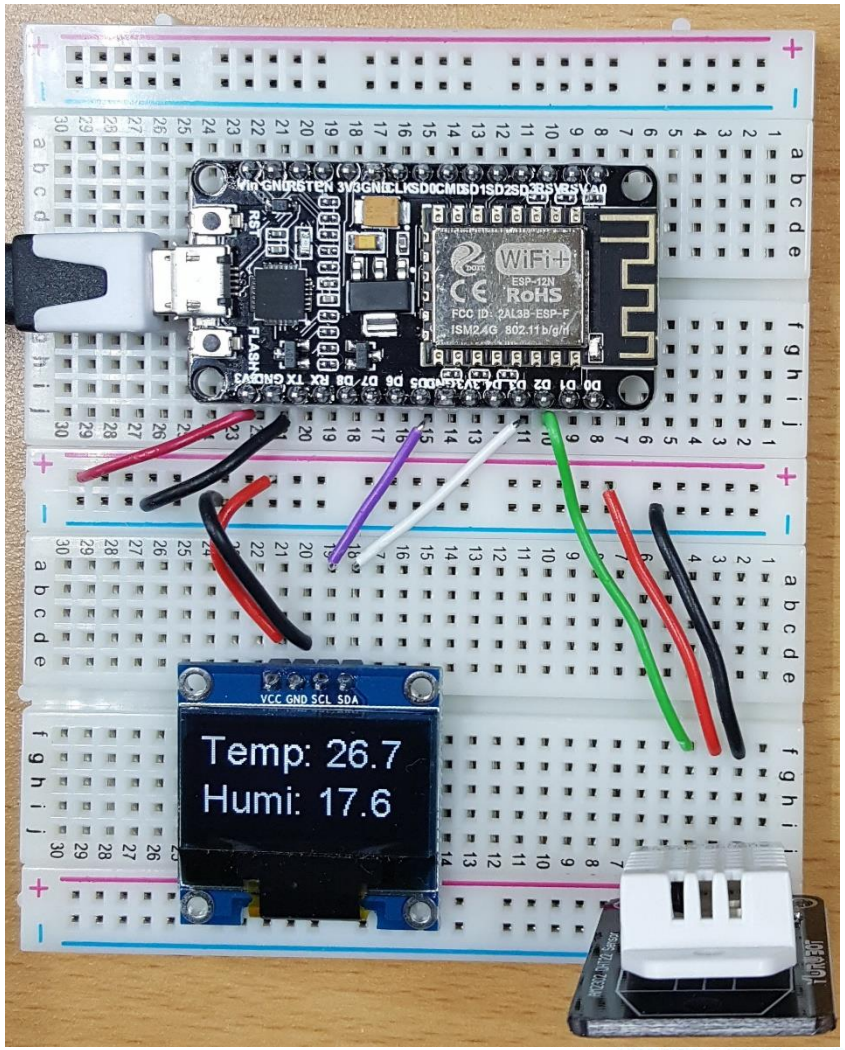




0.96inch OLED Display 연결(I2C serial communication)

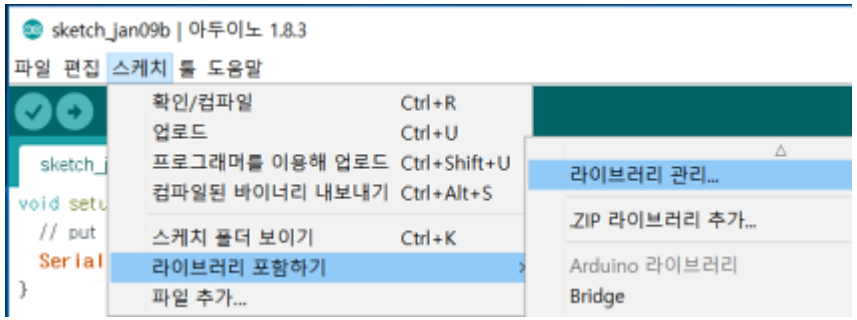
■ VCC->3.3V, GND->GND, SCL->D5, SDA->D3

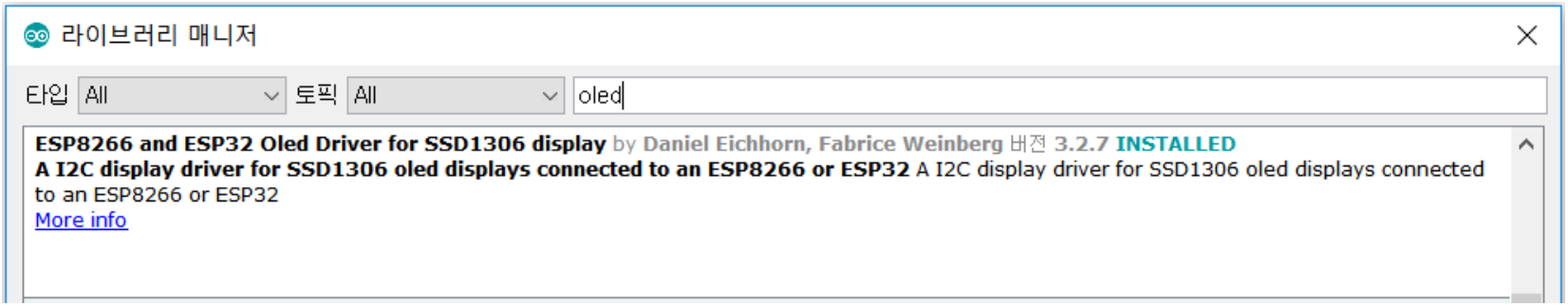


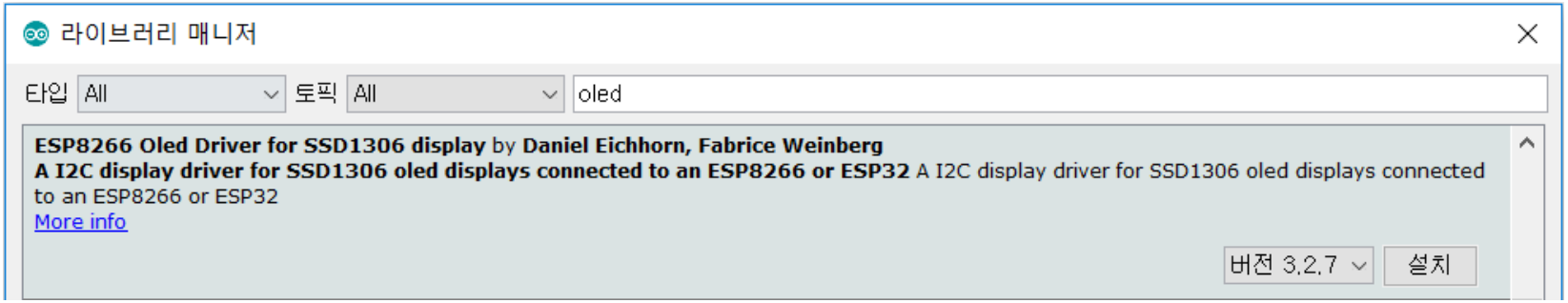


ESP8266 Oled Display Library

■ 라이브러리 매니저 열어서 oled로 검색 후, ESP8266 용 Oled Display library 설치 확인.







Oled Display Test

■ 아래의 코드를 입력, 업로드 후 실행 확인

```
#include "SSD1306.h"
SSD1306  display(0x3c, D3, D5);

void setup() {
  display.init();
  display.flipScreenVertically();
}
void drawFontFaceDemo() {
    display.setTextAlignment(TEXT_ALIGN_LEFT);
    display.setFont(ArialMT_Plain_10);
    display.drawString(0, 0, "Hello world");
    display.setFont(ArialMT_Plain_16);
    display.drawString(0, 10, "Hello world");
    display.setFont(ArialMT_Plain_24);
    display.drawString(0, 26, "Hello world");
}
void loop() {
  display.clear();
  drawFontFaceDemo();
  display.display();
}
```

온습도 값 Display(1)

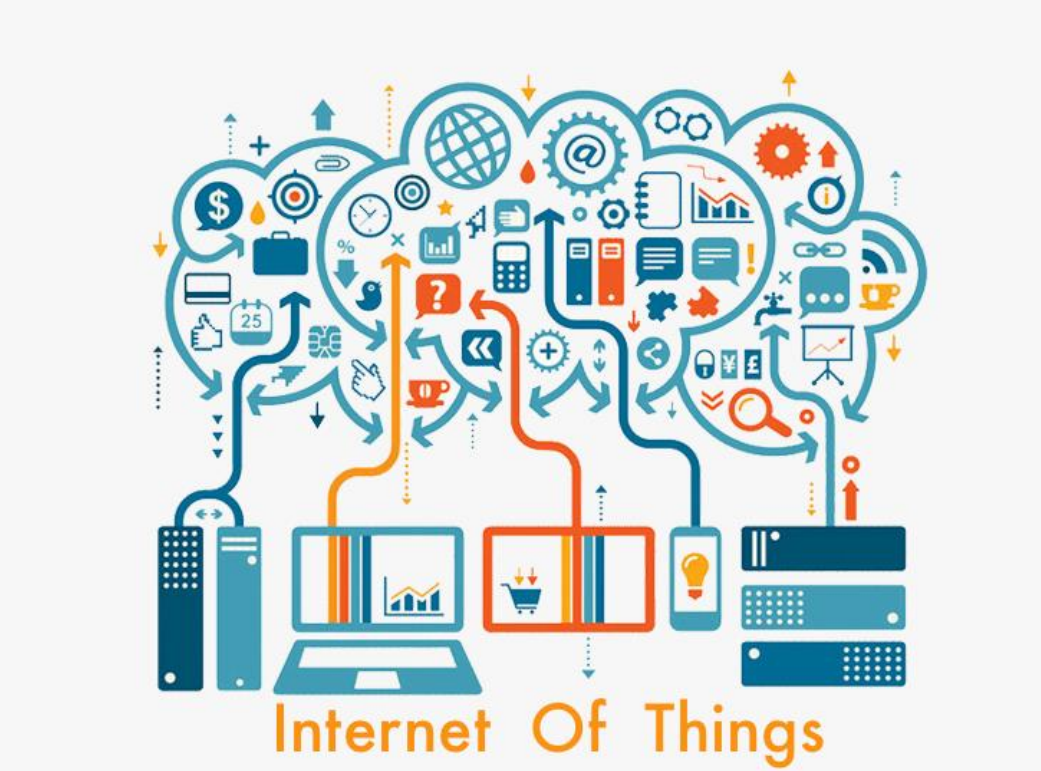
■ 아래의 코드를 입력, 업로드 후 실행 확인

```
#include <DHT.h>
#include "UbidotsESPMQTT.h"

#define DHTPIN D2
#define DHTTYPE DHT22
#define TOKEN  "xxxxxxxxxxxxxxxxxx"  // Put here your Ubidots TOKEN
#define WIFI_SSID "xxxxxxxxx"
#define WIFI_PASS "xxxxxxxx"
#include "SSD1306.h"

DHT dht(DHTPIN, DHTTYPE);
Ubidots client(TOKEN);
SSD1306  display(0x3c, D3, D5);
void callback(char* topic, byte* payload, unsigned int length) {
}
void setup() {
  Serial.begin(115200);
  dht.begin();
  delay(10);
  client.wifiConnection(WIFI_SSID, WIFI_PASS);
  client.begin(callback);
  display.init();
  display.flipScreenVertically();
}
```

온습도 값 Display(2)

```
void loop() {
  float temp=dht.readTemperature();
  float humi=dht.readHumidity();
  if(!client.connected()){
      client.reconnect();
  }
  client.add("Temperature", temp);
  client.add("Humidity", humi);
  client.ubidotsPublish("Nodemcu");

  Serial.print("Temp: ");
  Serial.print(temp);
  Serial.print("   Humi: ");
  Serial.println(humi);

  char string[10];
  display.clear();
  display.setTextAlignment(TEXT_ALIGN_LEFT);
  display.setFont(ArialMT_Plain_24);
  dtostrf(temp, 4,1, string);
  String body="Temp: ";
  body += string;
  display.drawString(0, 12, body);
  dtostrf(humi, 4,1, string);
  body="Humi: ";
  body += string;
  display.drawString(0, 40, body);
  display.display();
  delay(1000*10);
}
```

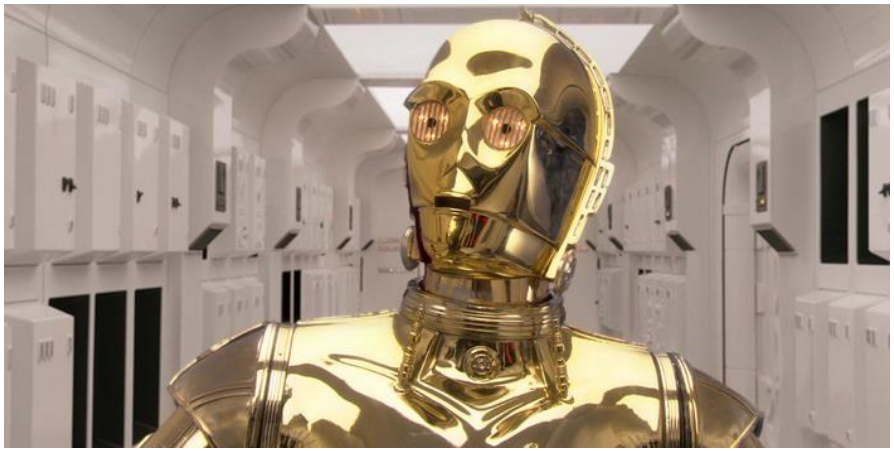
IoT Protocols



Communication Protocol

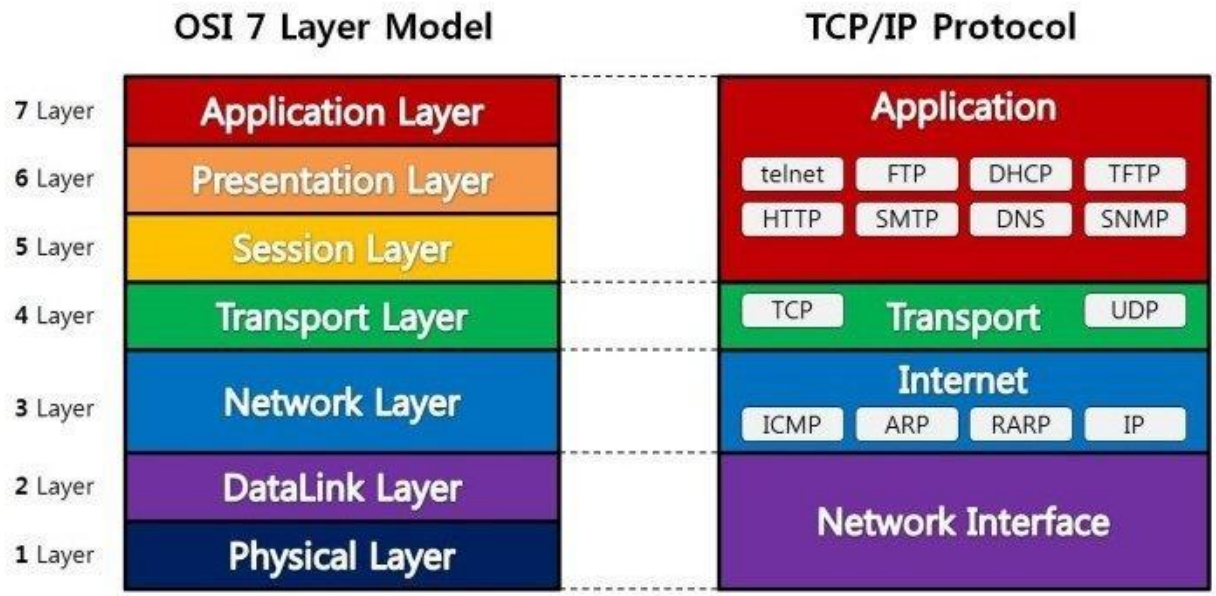
- In telecommunication, a communication protocol is a system of rules that allow two or more entities of a communications system to transmit information via any kind of variation of a physical quantity. The protocol defines the rules syntax, semantics and synchronization of communication and possible error recovery methods. Protocols may be implemented by hardware, software, or a combination of both.[1]
- Communicating systems use well-defined formats (protocol) for exchanging various messages.

-Wikipedia

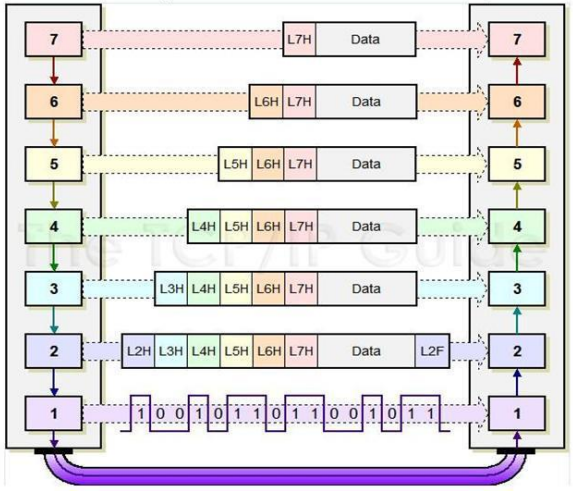


C3PO: Protocol Droid

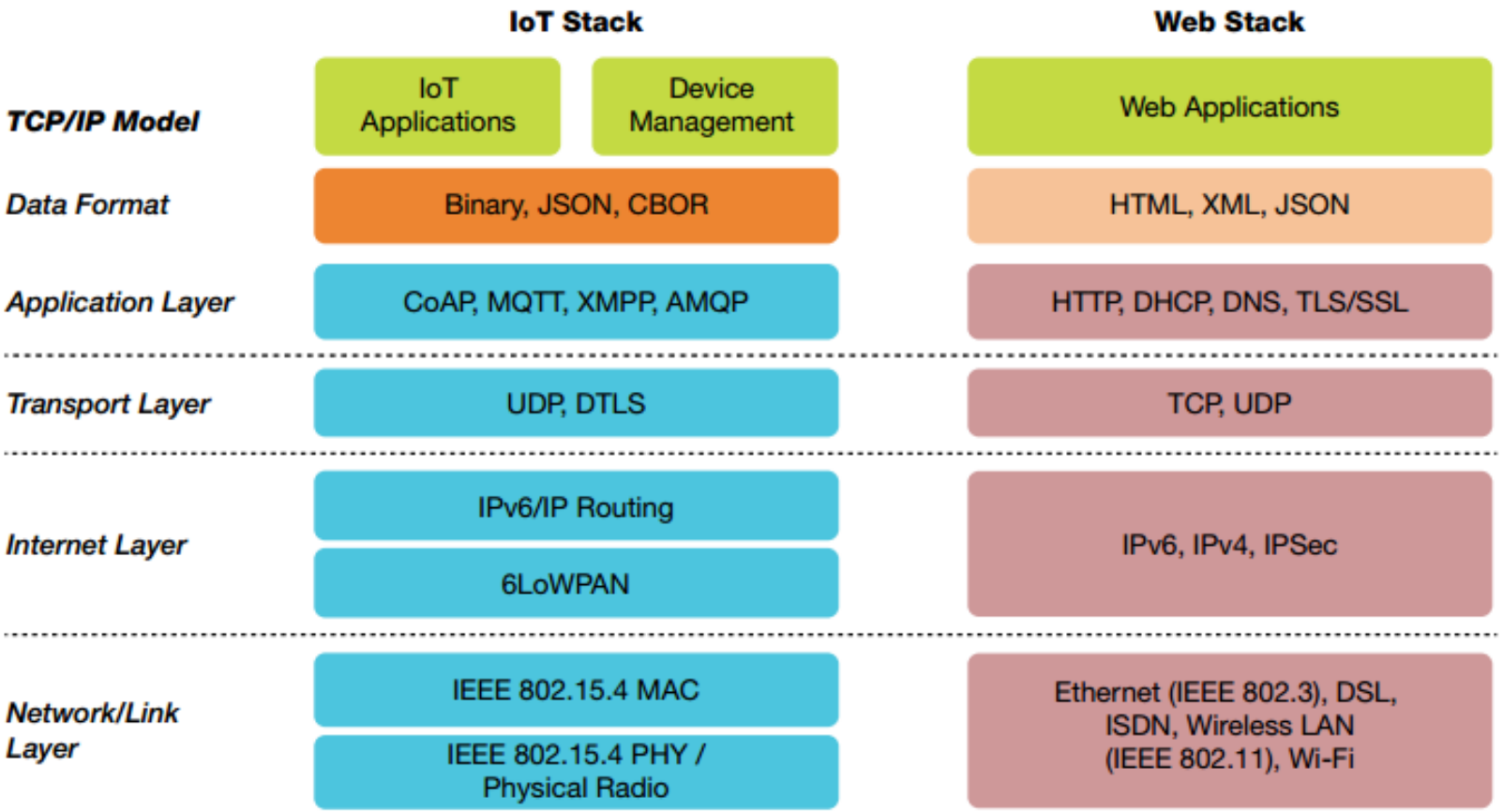
OSI Model (Open Systems Interconnection Model)



Data encapsulation & de-encapsulation



IoT Protocol Stack

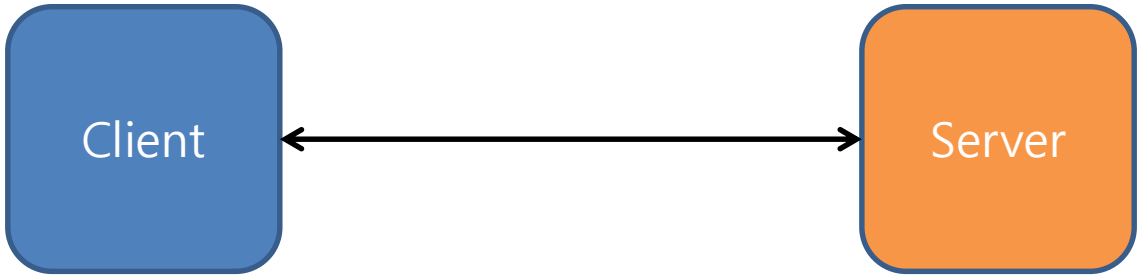


Standardization

- **HTTP**
  - IETF standard (RFC 2616 is HTTP/1.1)
- **CoAP**
  - IETF standard (RFC 7252)
- **MQTT**
  - OASIS standard (v3.1.1)
- **AMQP**
  - OASIS and ISO 19464 standard (1.0)

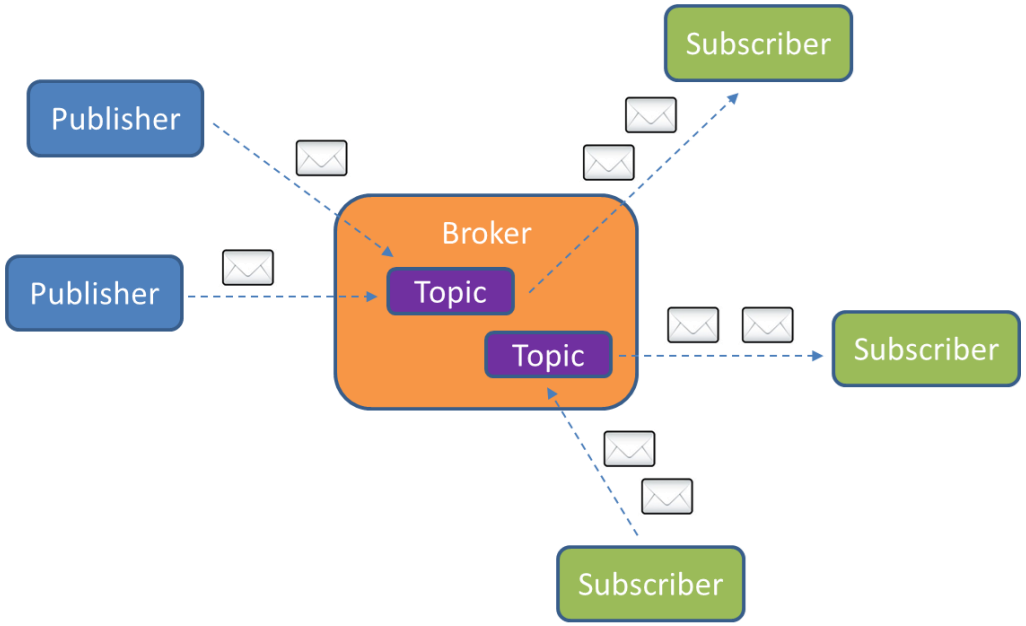
Architecture: HTTP & CoAP

- **Client/Server**
  - request/response
    - HTTP : synchronous
    - CoAP : (also) asynchronous
- **HTTP is ASCII based**
- **CoAP is binary based**



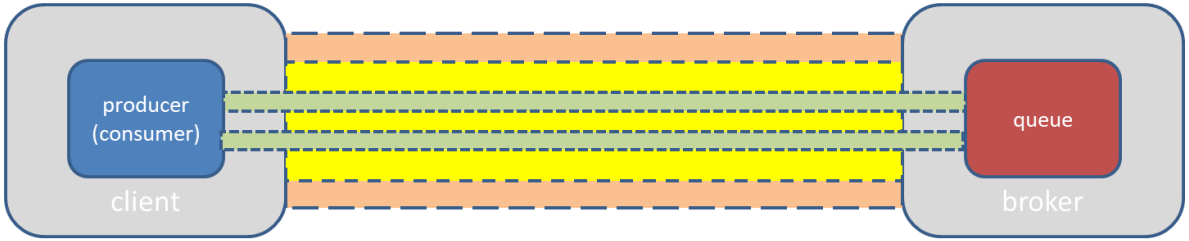
Architecture: MQTT

- **Broker and connected Clients**
  - broker receives subscriptions from clients on topics
  - broker receives messages and forward them
  - clients subscribe/publish on topics
- **Brokers bridge configuration**



Architecture: AMQP(1.0)

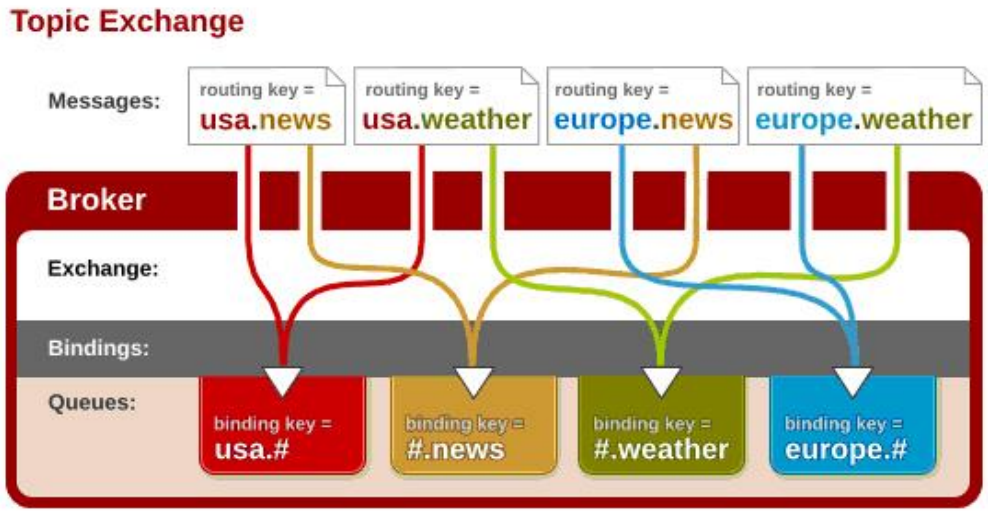
■ Multiplexing frames on sessions and links



container
node

connection
session
link

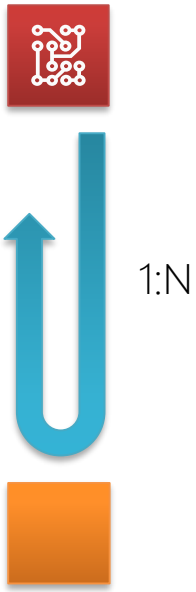
container
node



IoT Communication Patterns

Telemetry

Information flows from device to other systems for conveying status changes in the device



Inquiries

Requests from devices looking to gather required information or asking to initiate activities

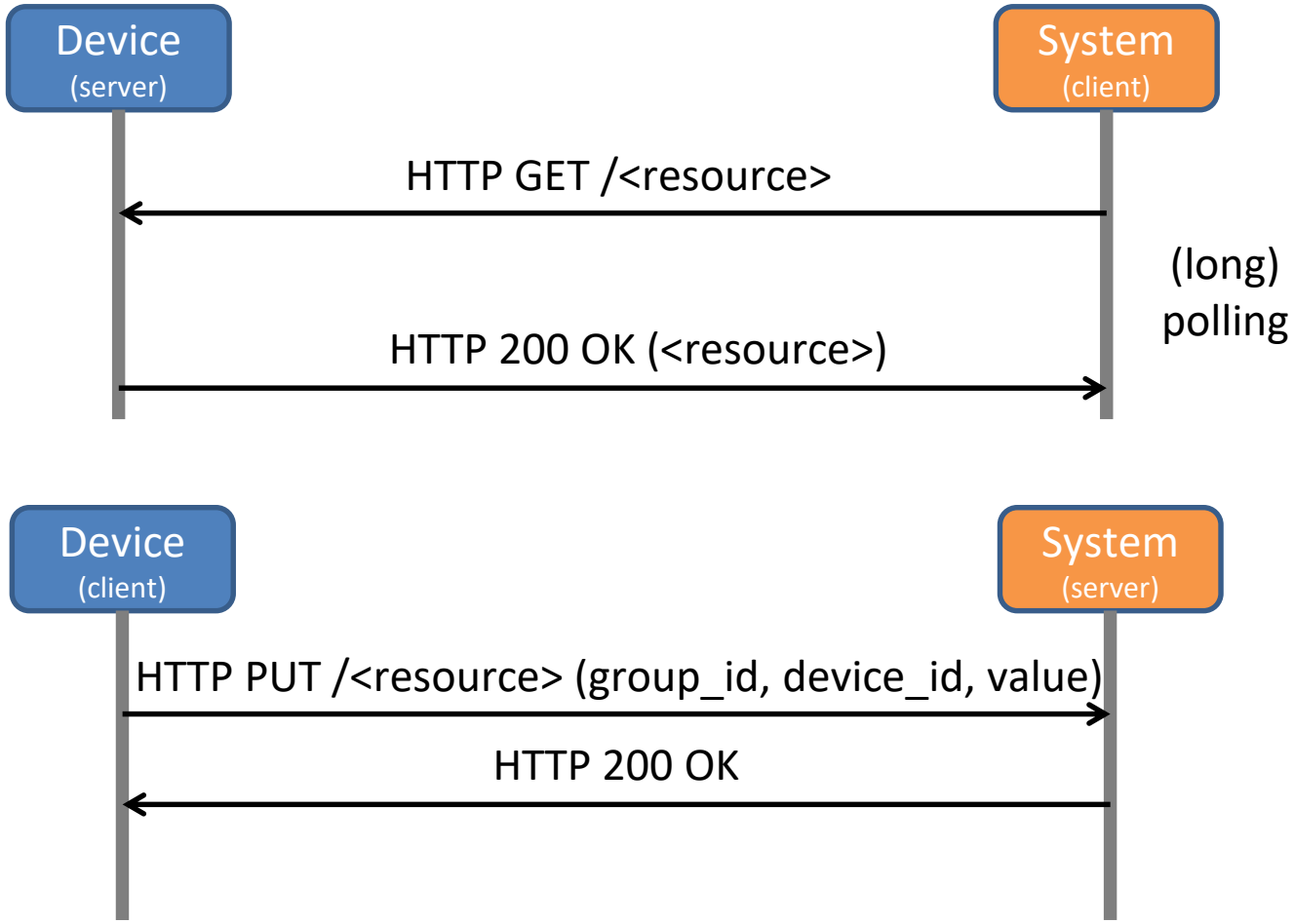
Commands

Commands from other systems to a device or a group of devices to perform specific activities

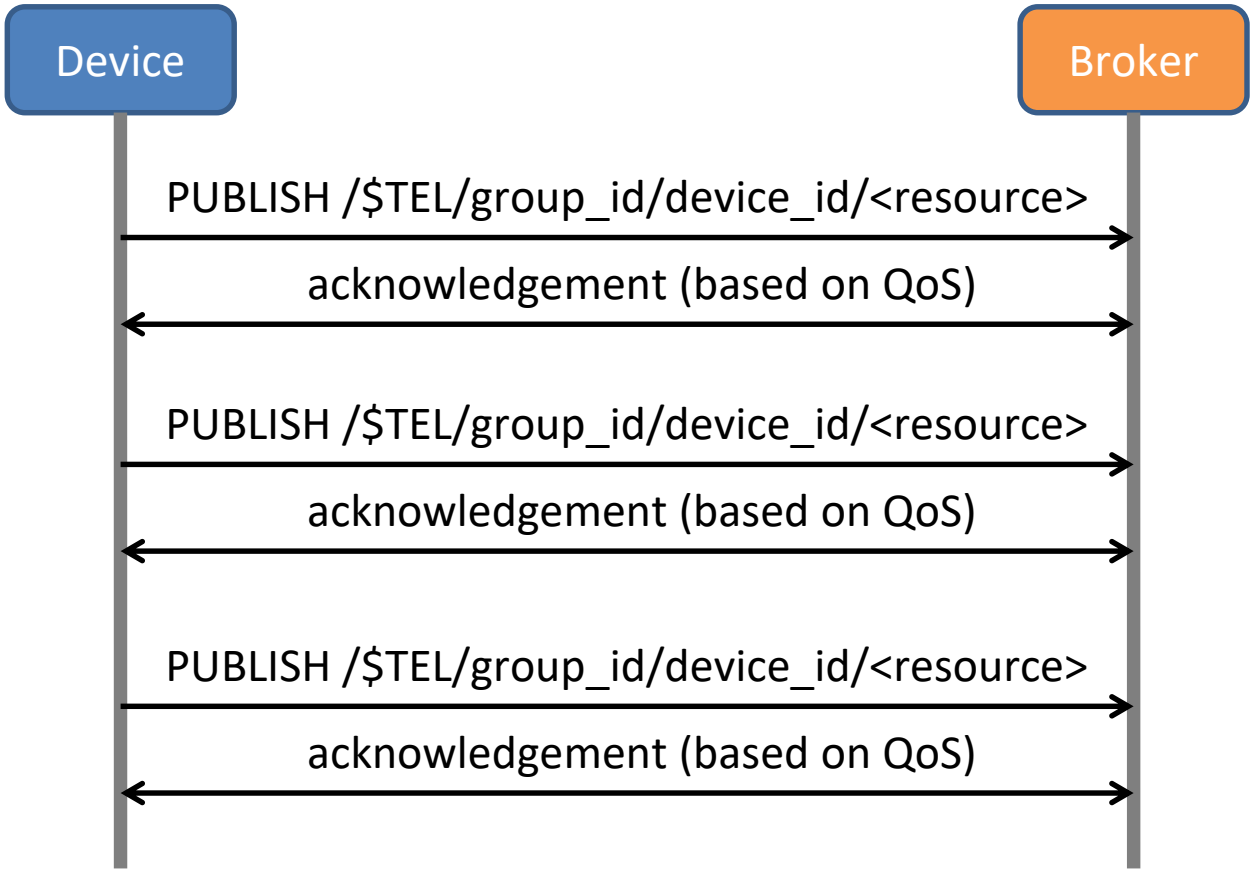
Notifications

Information flows from other systems to a device (-group) for conveying status changes in the world

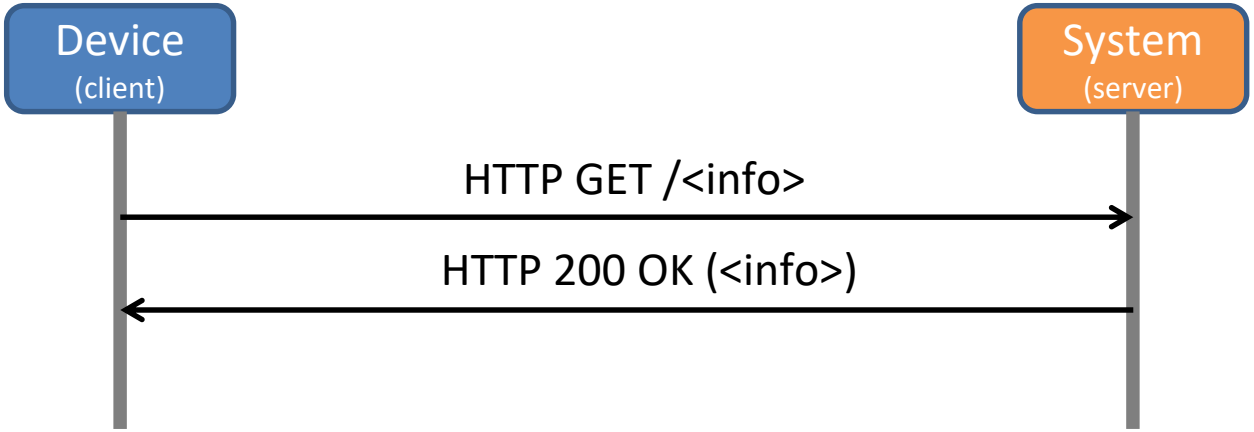
Telemetry: HTTP



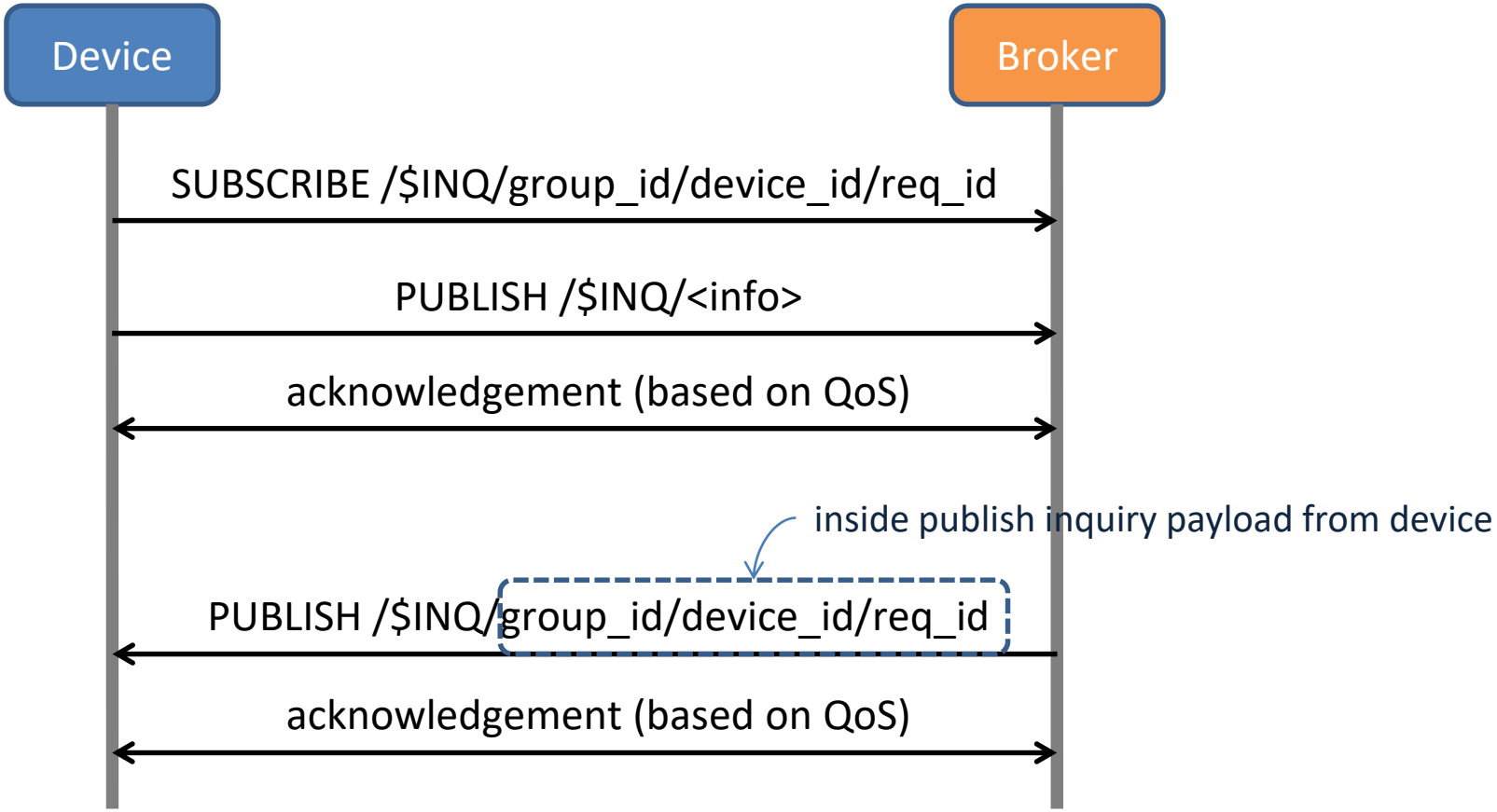
Telemetry: MQTT



Inquiry: HTTP

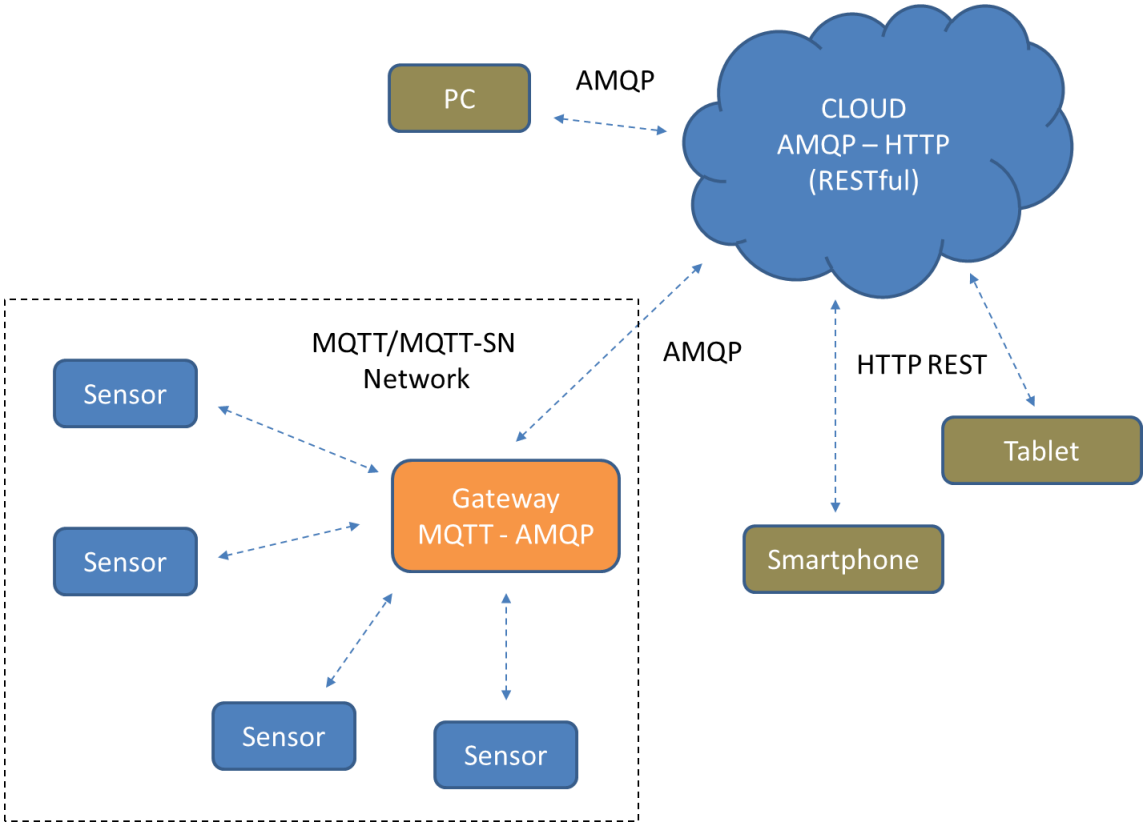


Inquiry: MQTT

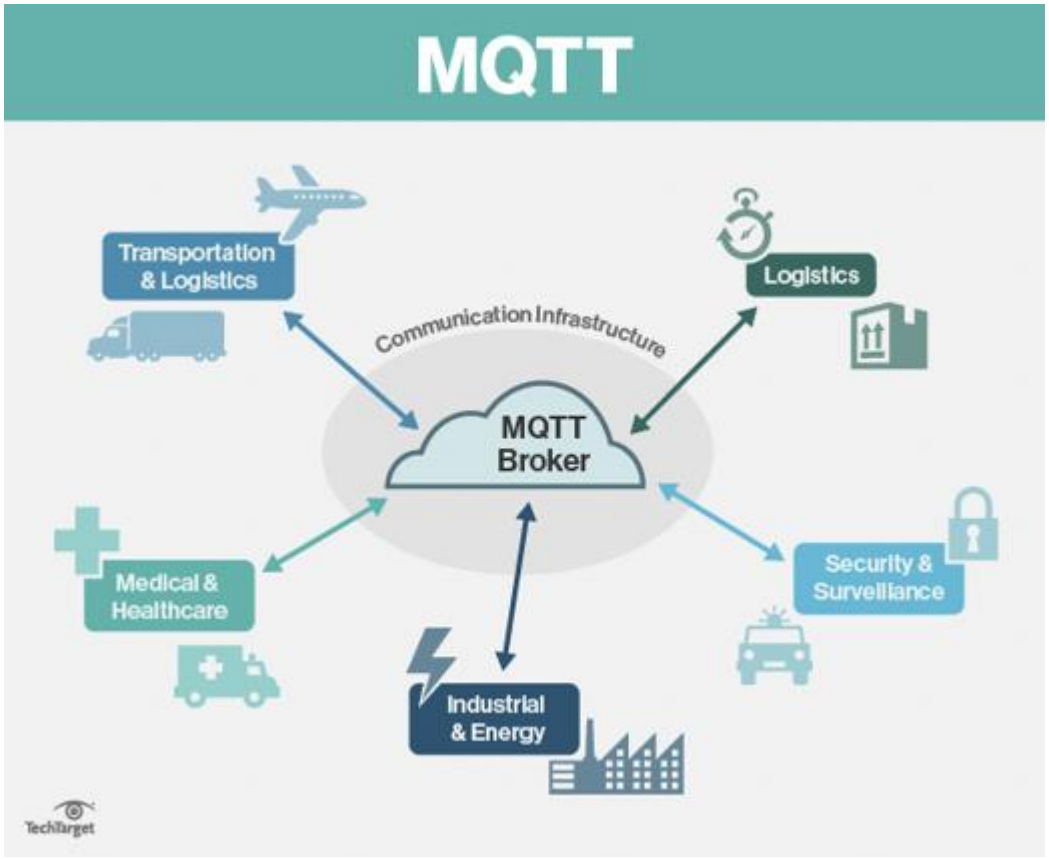


IoT Protocol Conclusion

- protocol choice depends on scenario
- some protocols have more features than other
- a complex system can use more protocols



Lab2: MQTT Protocol 실습



MQTT Protocol

■ Publish/Subscribe Model

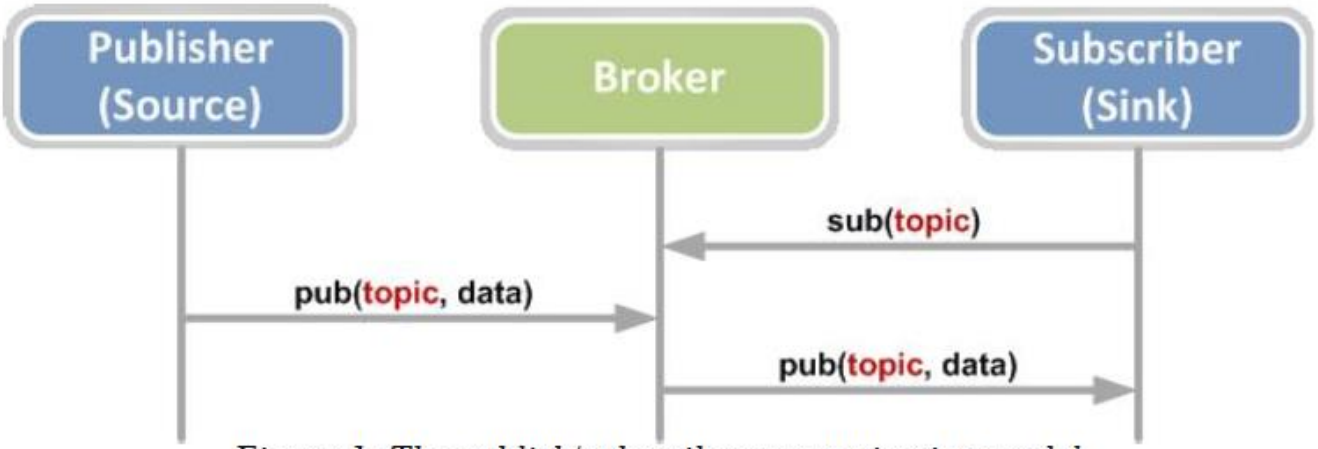
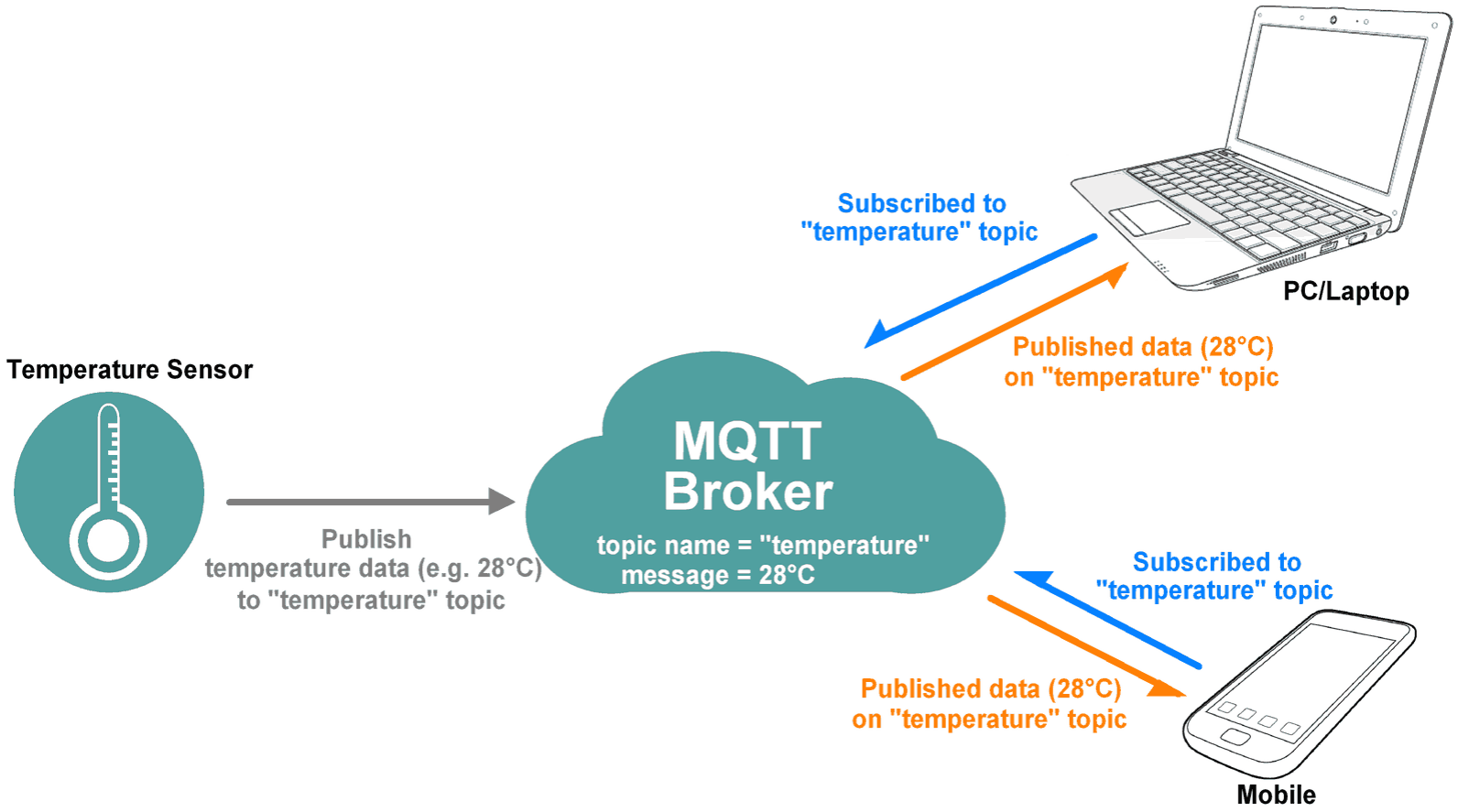
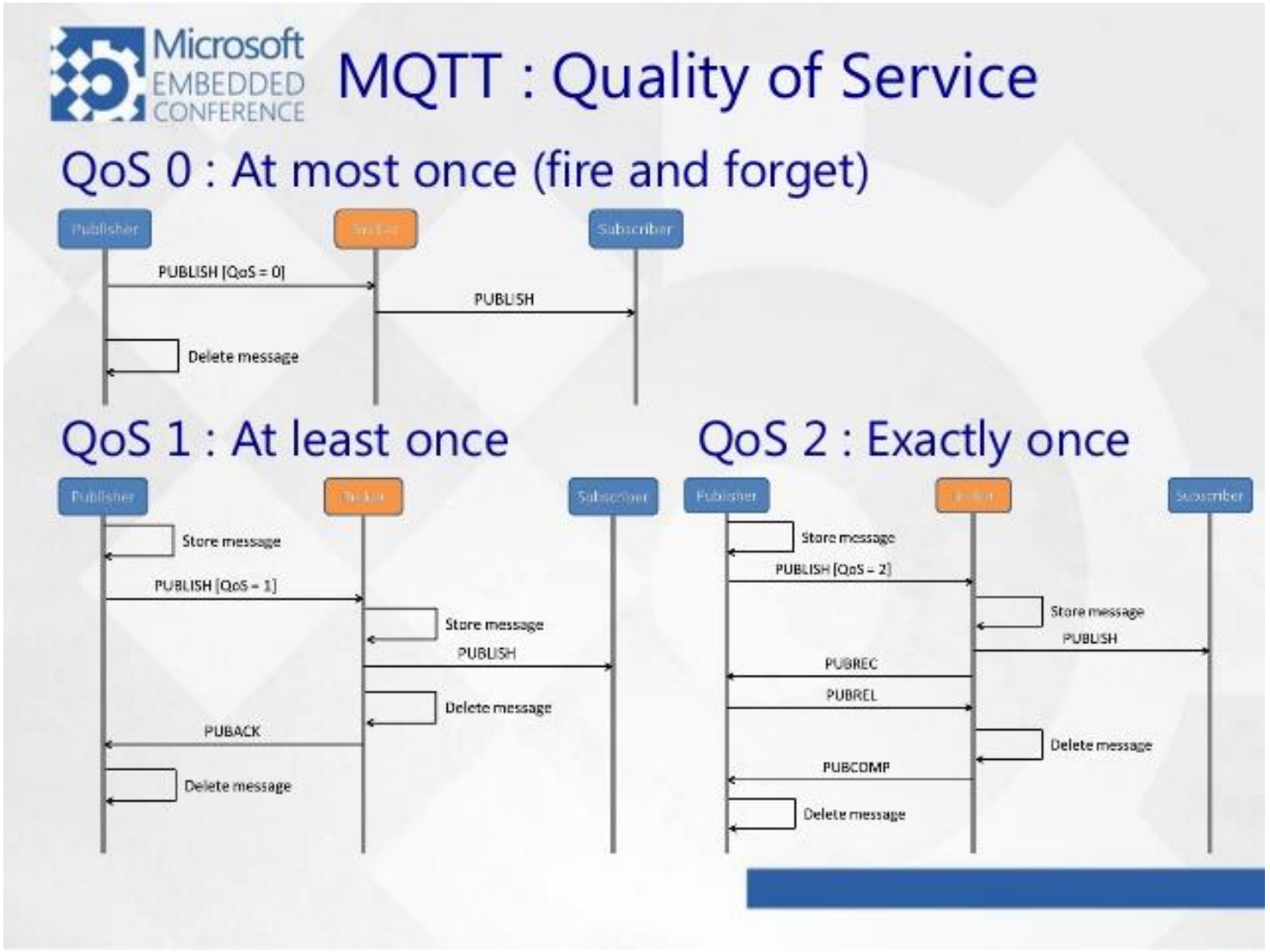


Figure 1: The publish/subscribe communication model

MQTT Protocol 사용 예

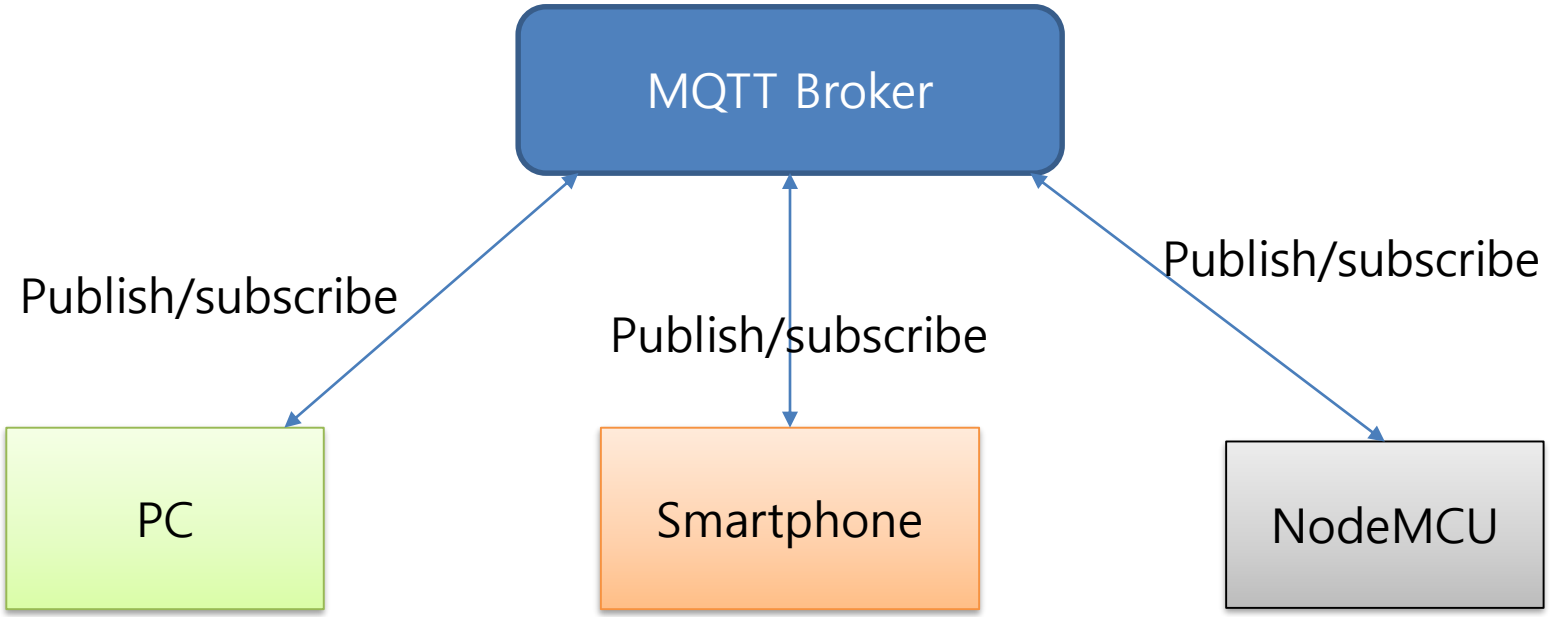


MQTT: Quality of Service

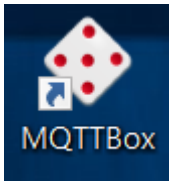


실습 구성 및 순서

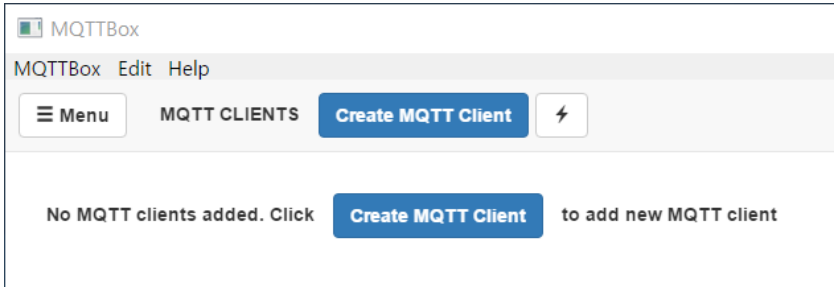
1. **PC and MQTT Broker**
2. **Smartphone, PC and MQTT Broker**
3. **NodeMCU, Smartphone, PC and MQTT Broker**



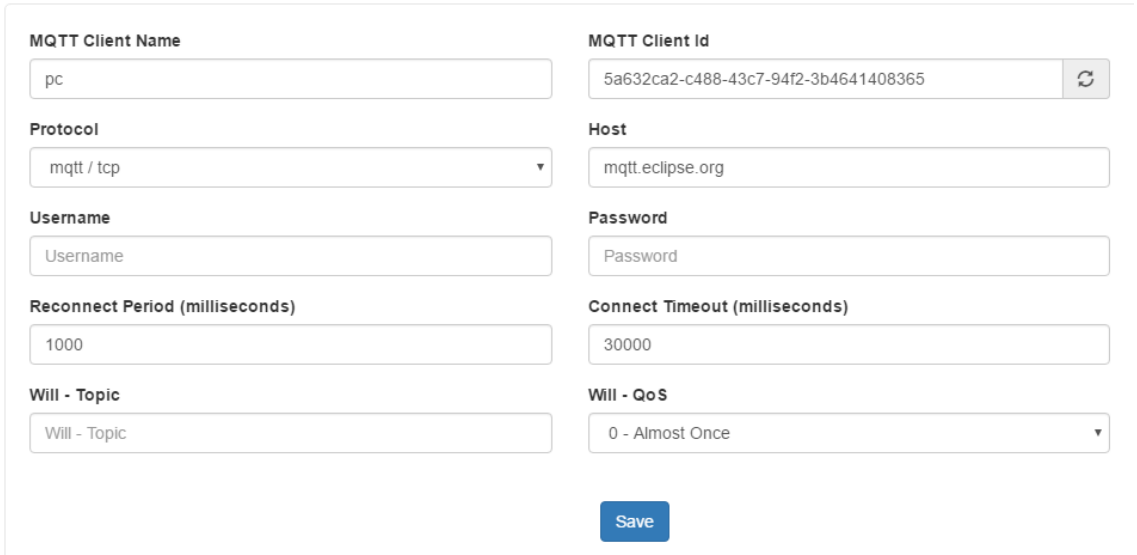
Windows MQTT Client



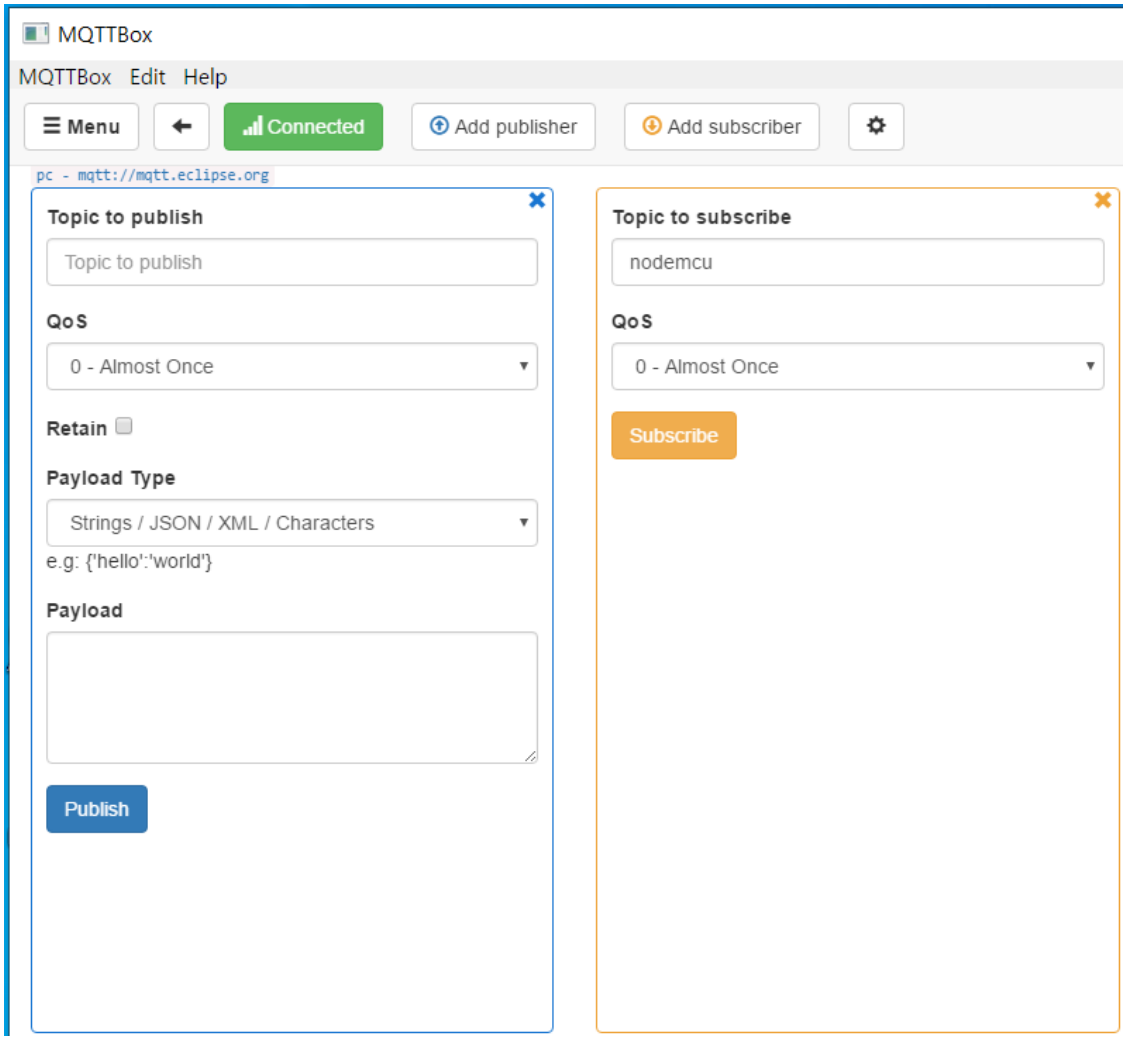
- Open MQTTBox and Create MQTT Client



- Host: mqtt.eclipse.org
- Protocol: mqtt/tcp
- MQTT Client Name: pc
- Click SAVE

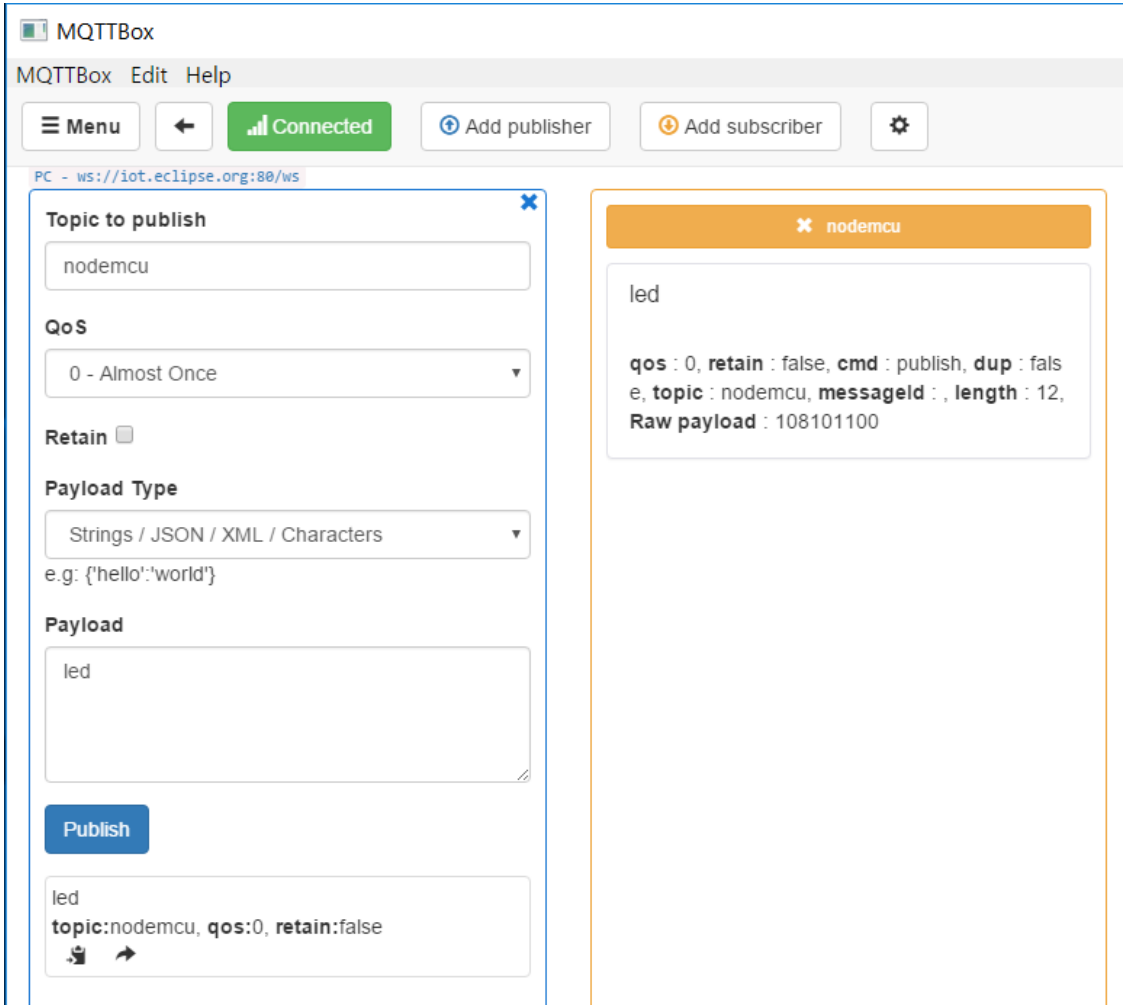


■ Topic to subscribe 에 입력(예:nodemcu) 후 click Subscribe 버튼



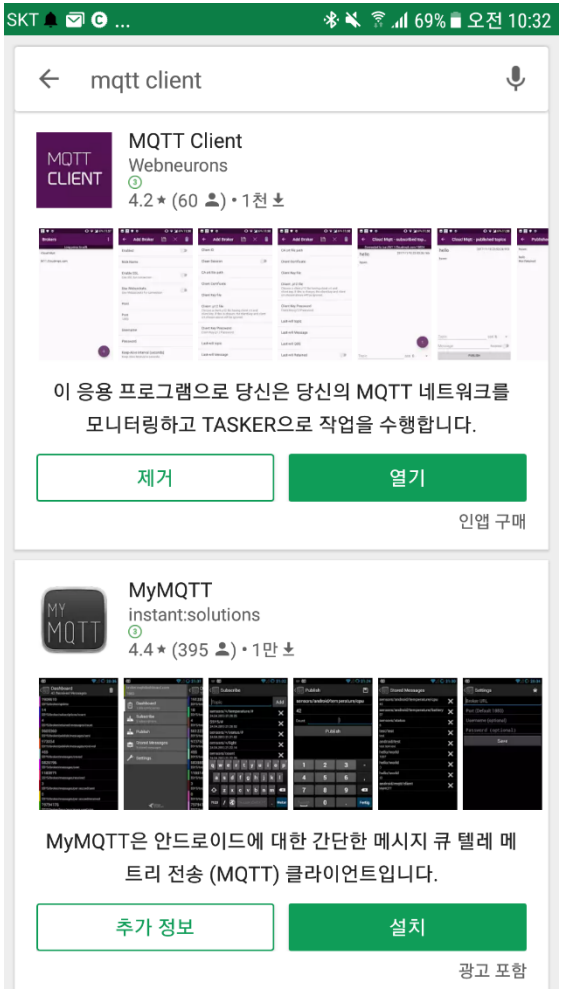
- 실습 시 주의 사항: 모든 학생들이 동일한 Topic을 사용할 경우 중복될 수 있으므로, 실습 시 Topic은 “nodemcuXXXX”(XXXX는 자신의 생일 4자리)으로 정하거나, 자신 만의 고유한 Topic을 사용하여 다른 사람과 Topic이 중복되지 않도록 한다.

■ Topic to publish(예: nodemcu)와 Payload(예: led)에 입력 후, Publish 버튼을 누르면 subscriber에 topic과 메시지가 나오는 것을 확인.



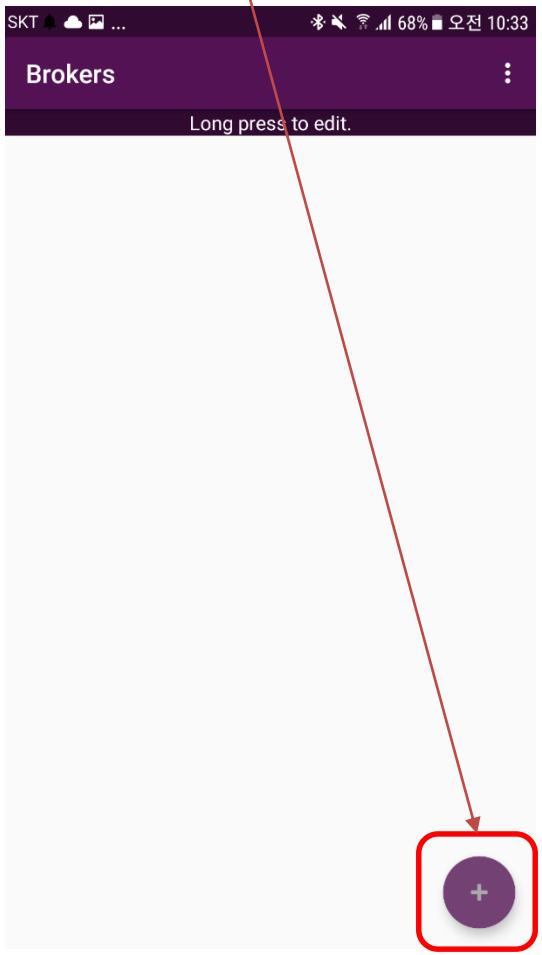
스마트폰에 mqtt client app 설치

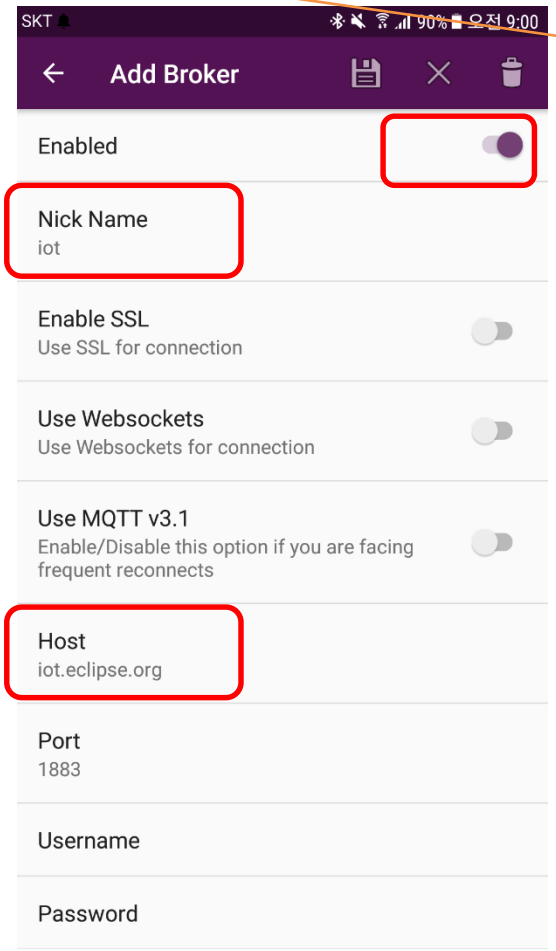
■ 앱스토어에서 mqtt 로 검색하여 유사한 앱 설치

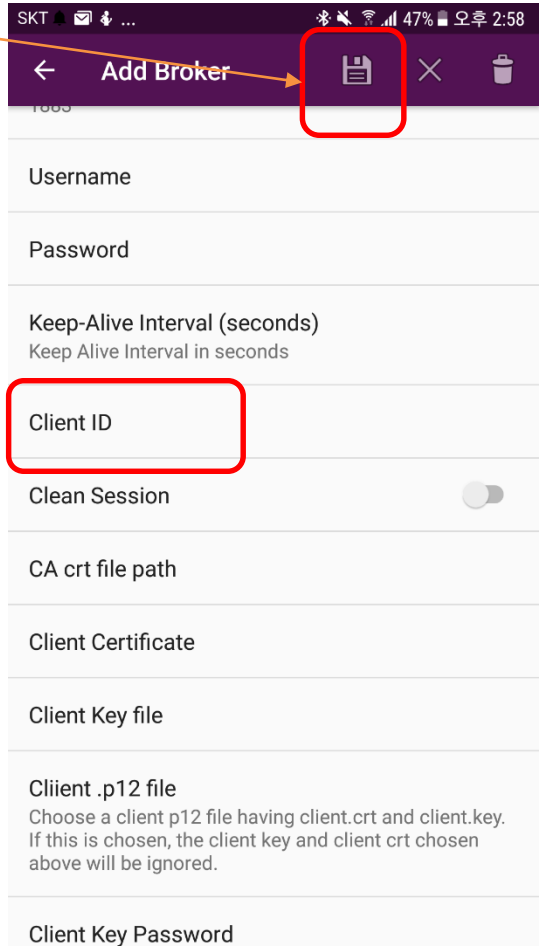


MQTT Client: Add Broker

■ Add Broker button을 눌러서 Enabled ON, Nick Name, Host(iot.eclipse.org), Client ID 입력 후 저장 버튼을 누름.

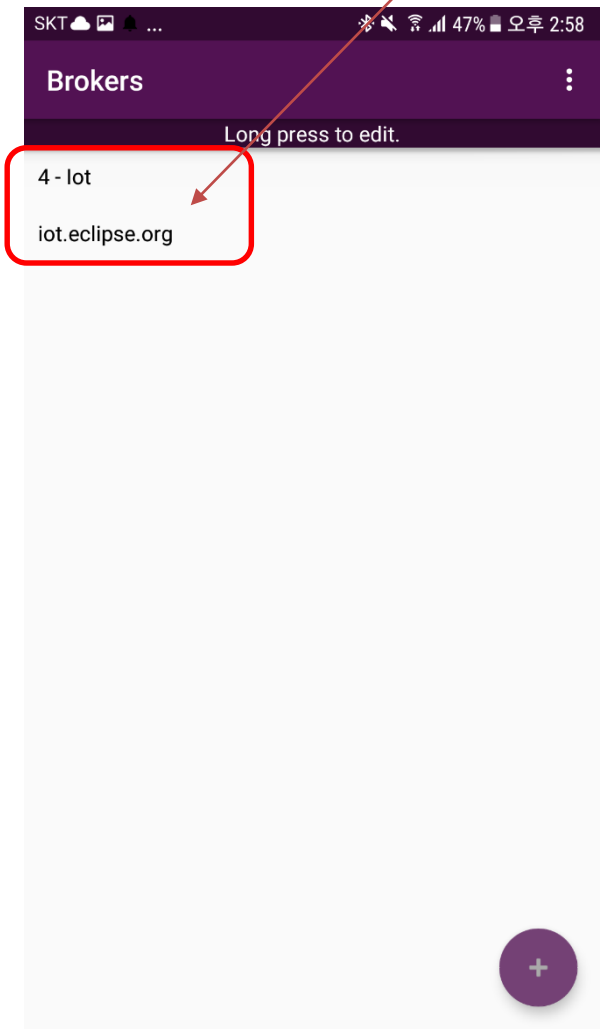






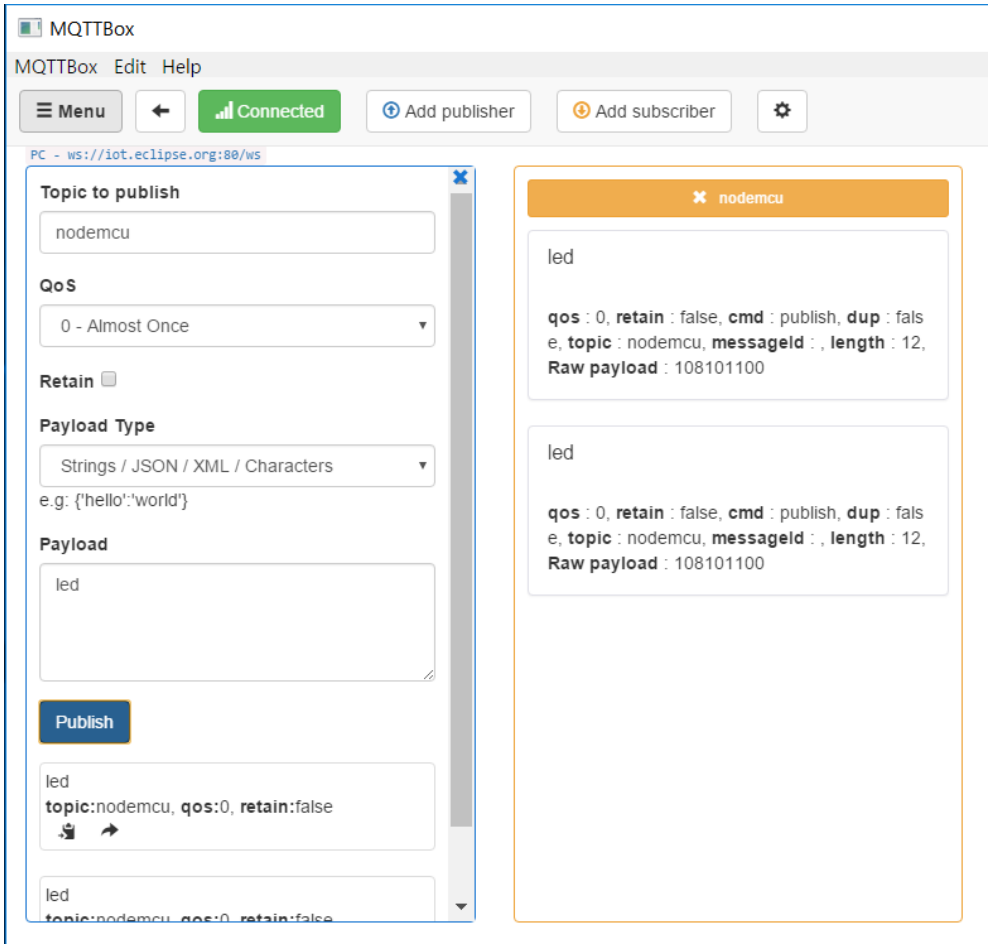
MQTT Client: Add Topics

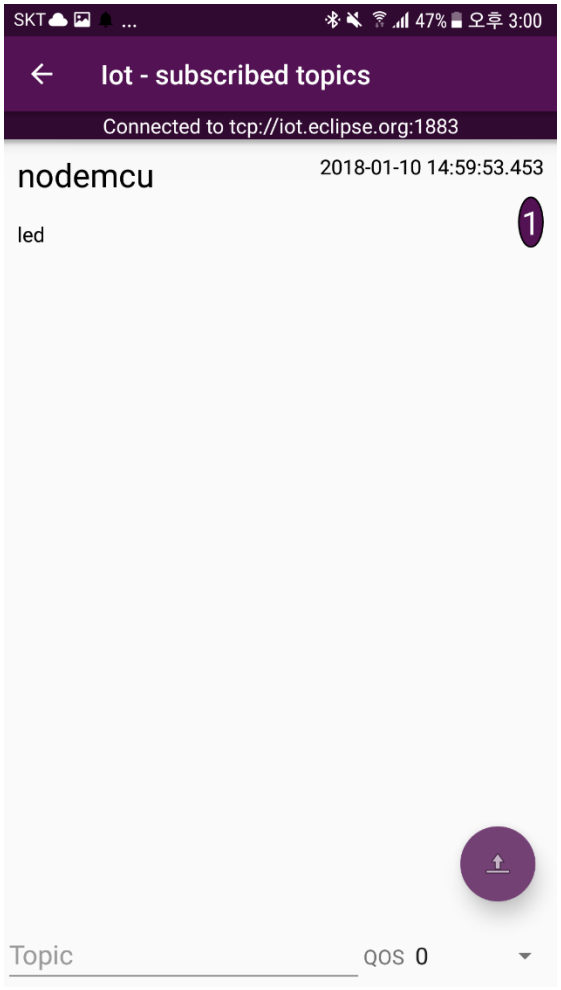
■ 새로 추가된 Broker를 눌러서 Subscribed topics에 nodemcu 입력



MQTT Client

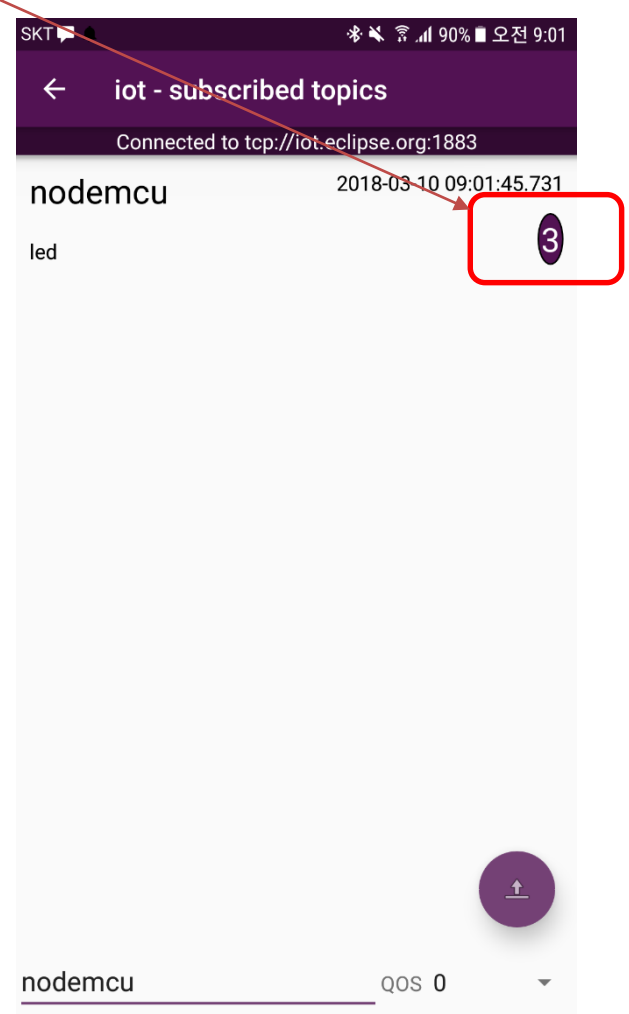
■ PC Client에서 Topic을 Publish 후 스마트폰 Client에서 subscribed 확인

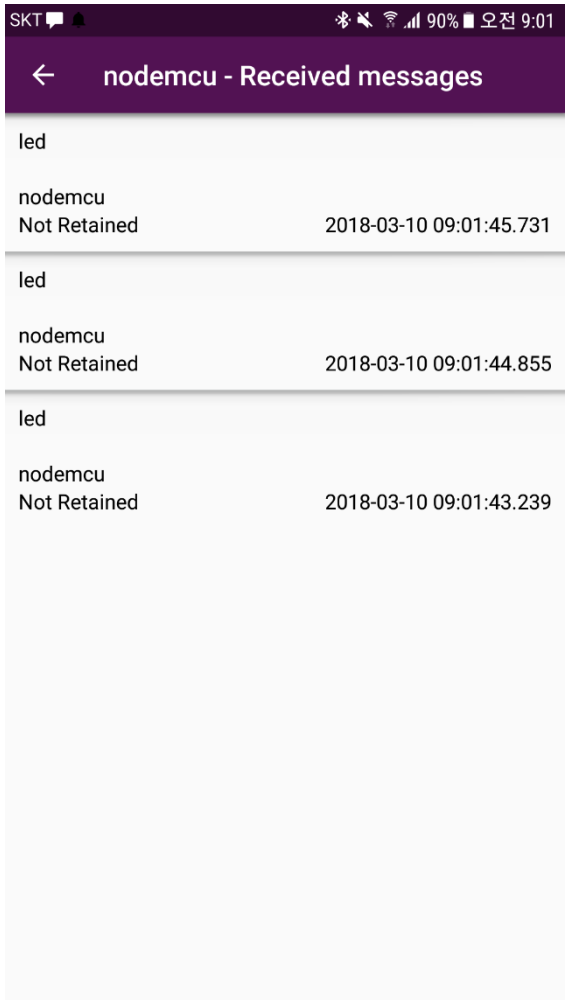




MQTT Client

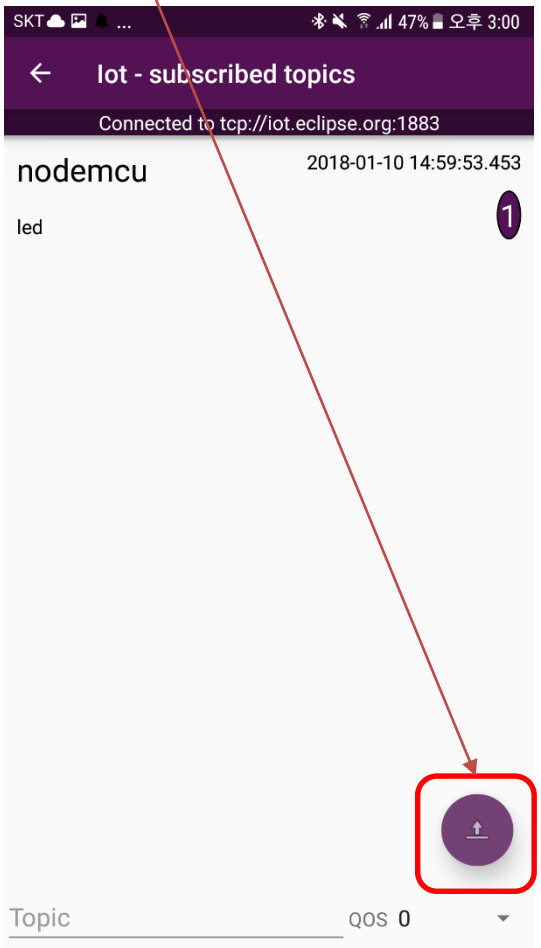
■ 번호를 누르면 수신된 메시지 확인 가능

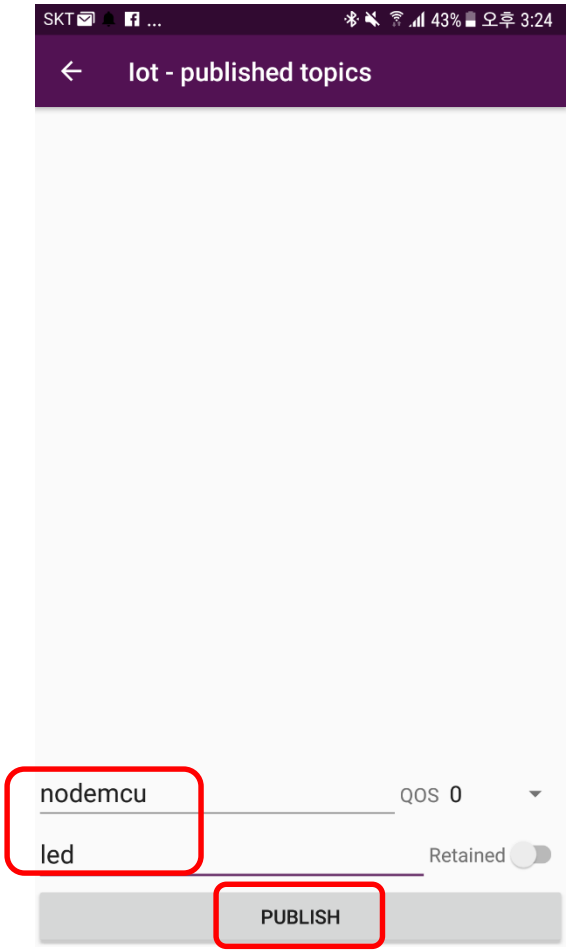




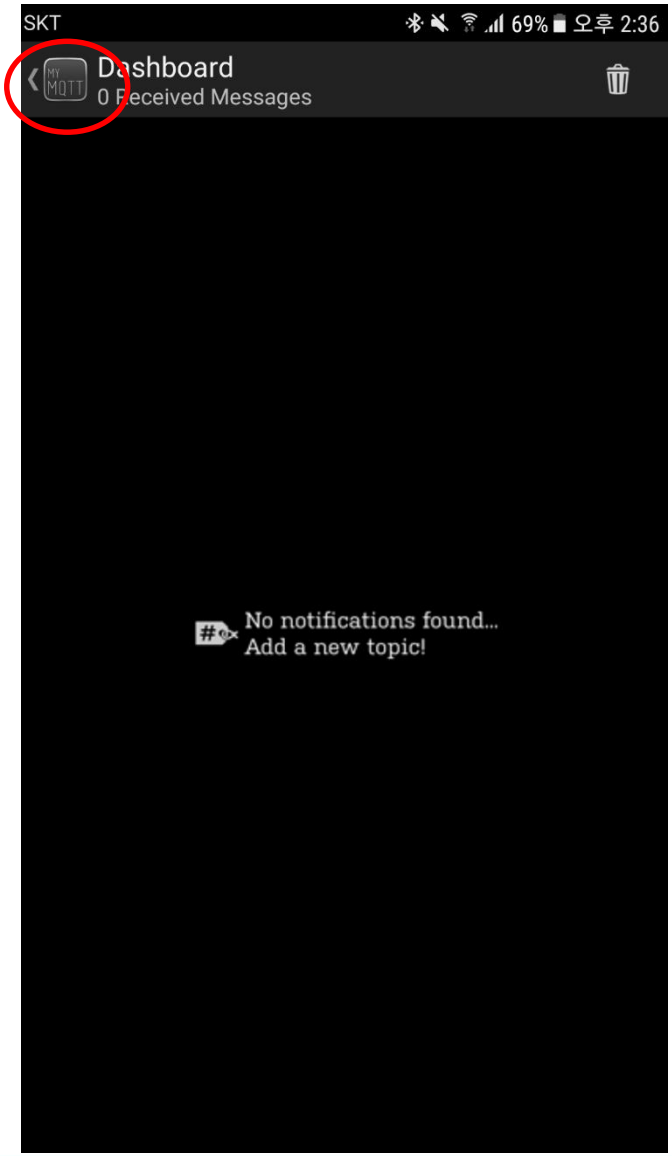
MQTT Client

■ Publish button을 눌러서 publish topic과 message를 입력 후 publish 버튼을 누른 후, PC Client에서 확인.

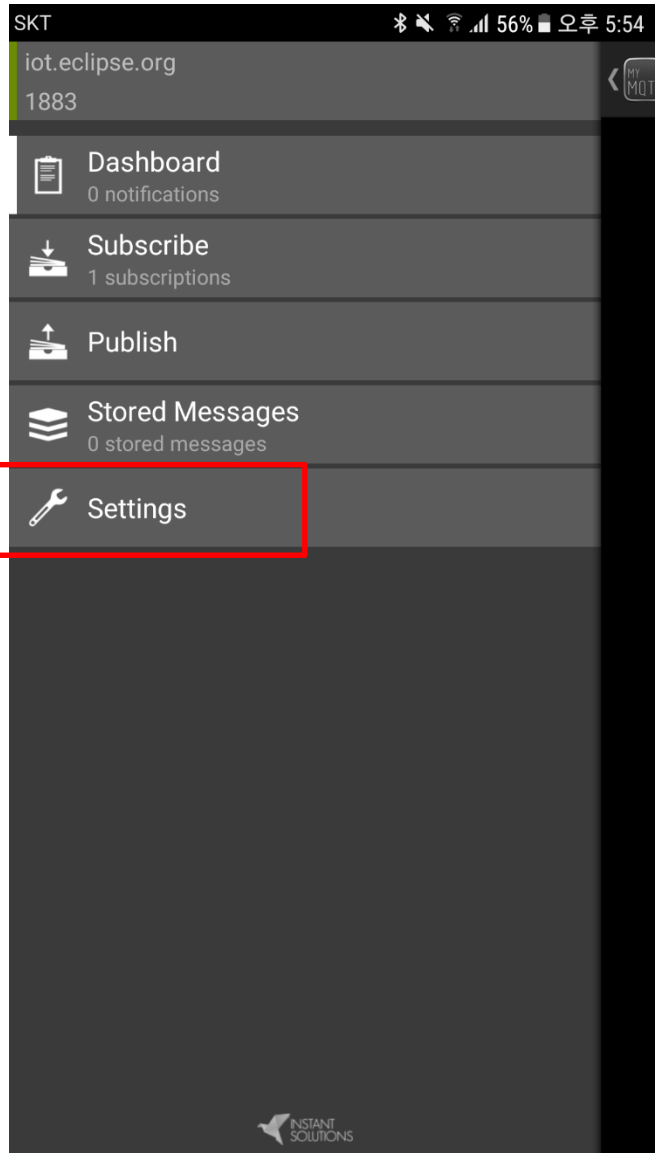


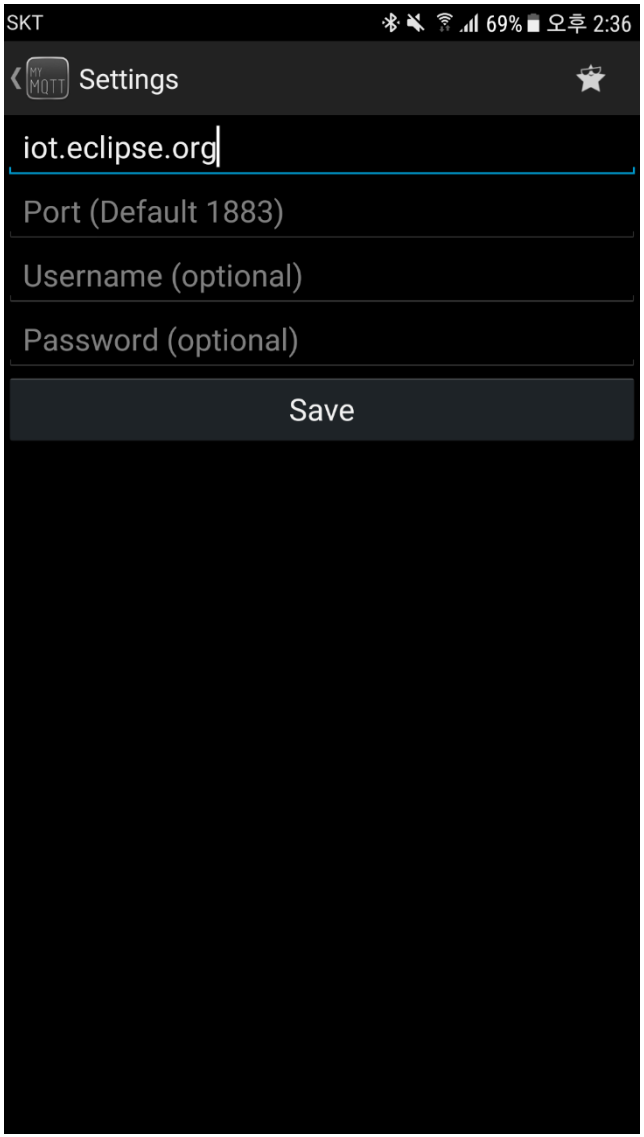


MyMQTT

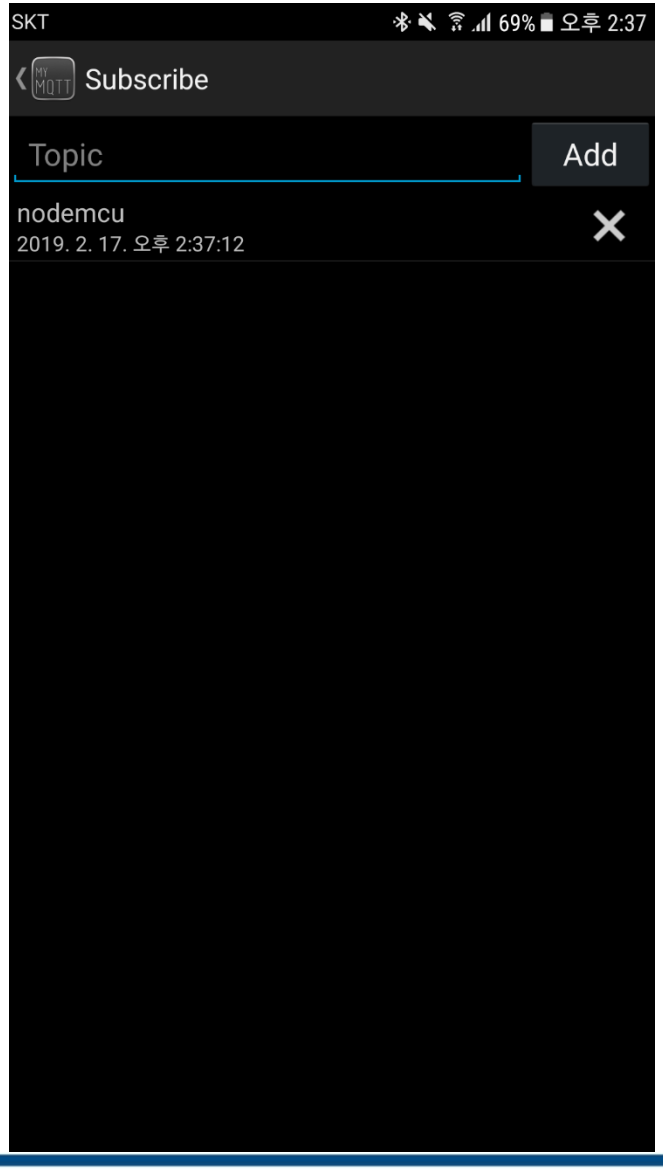

SKT
69% 오후 2:36
Dashboard
0 Received Messages
No notifications found...
Add a new topic!

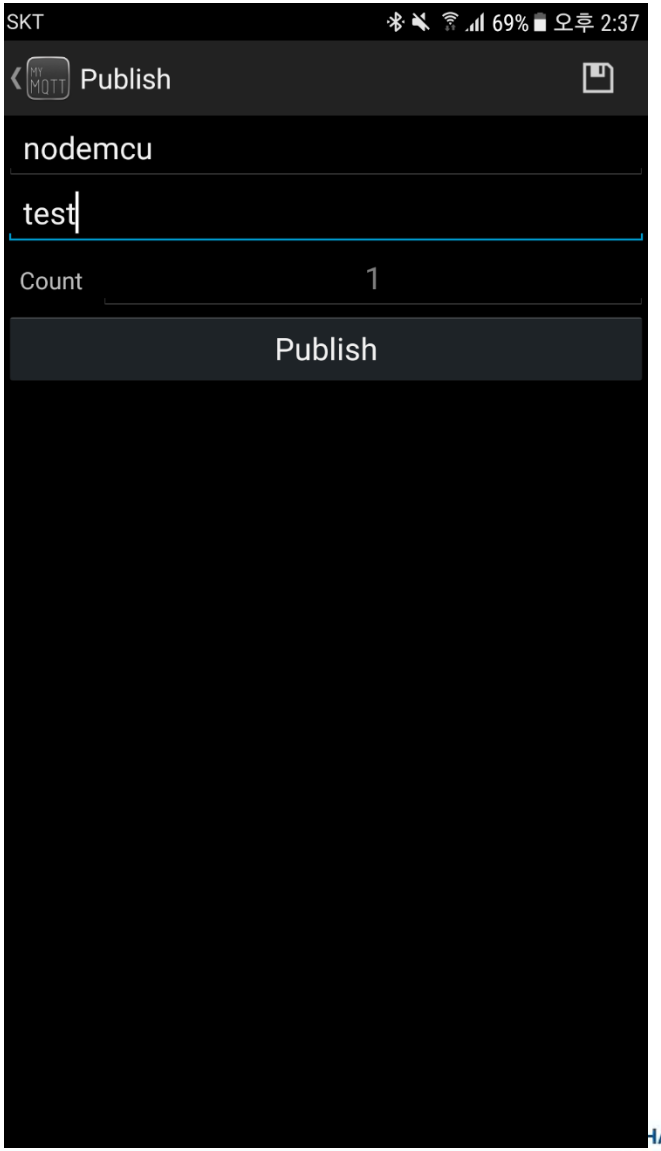
MyMQTT


SKT
56% 오후 5:54
iot.eclipse.org
1883
Dashboard
0 notifications
Subscribe
1 subscriptions
Publish
Stored Messages
0 stored messages
Settings
INSTANT SOLUTIONS

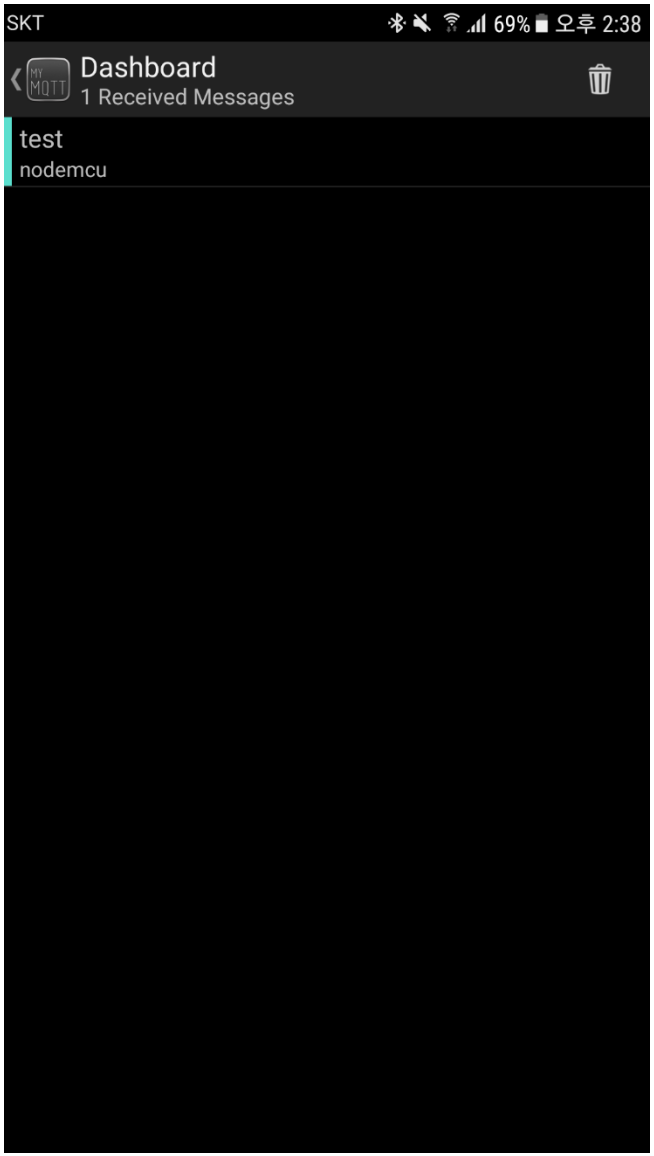

SKT
69% 오후 2:36
Settings
iot.eclipse.org
Port (Default 1883)
Username (optional)
Password (optional)
Save

MyMQTT: Subscribe, Publish

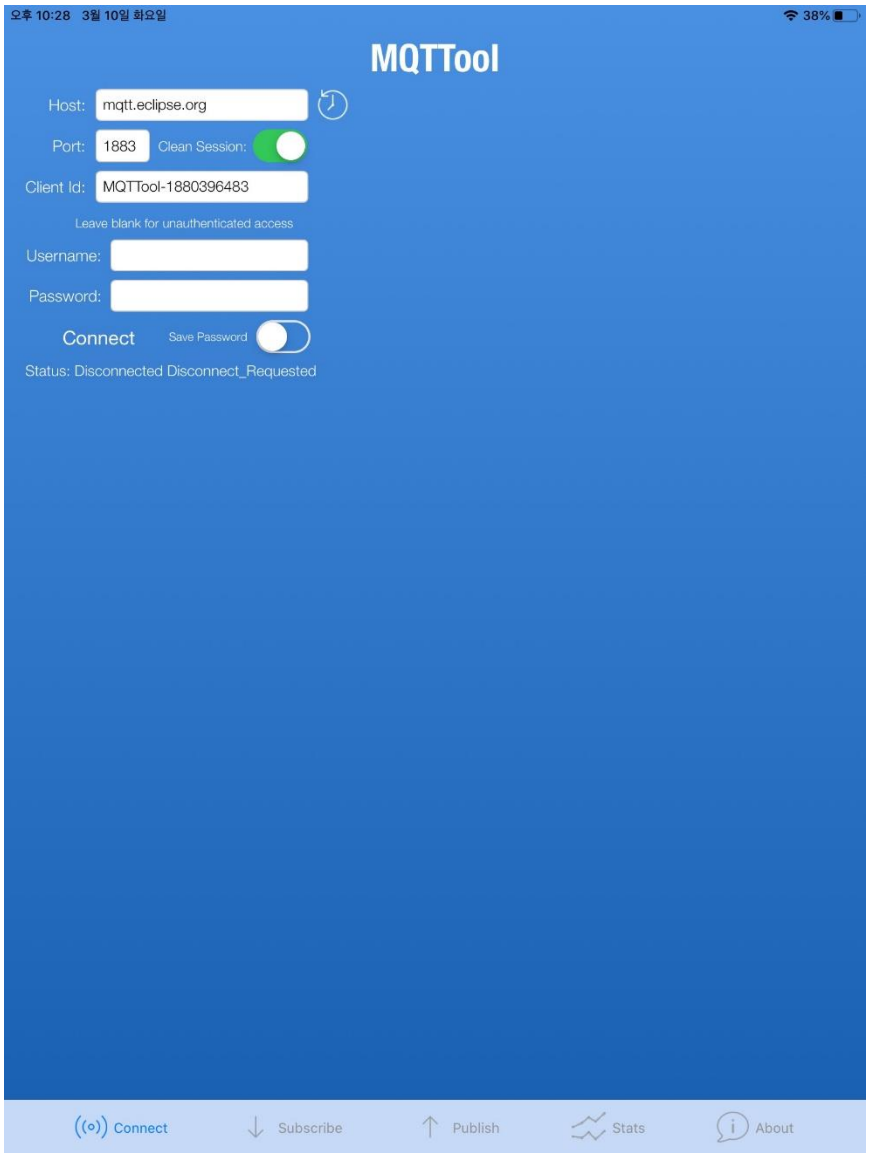


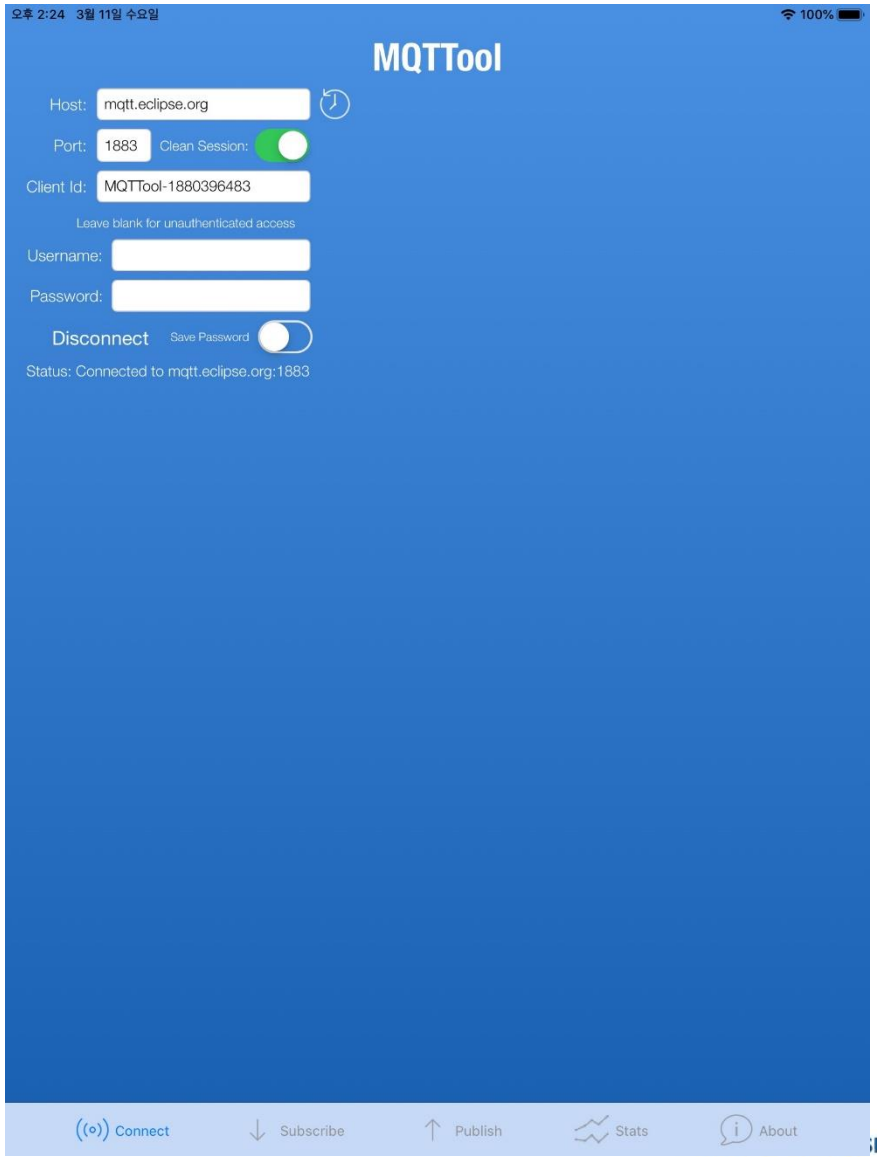


MyMQTT: Dashboard

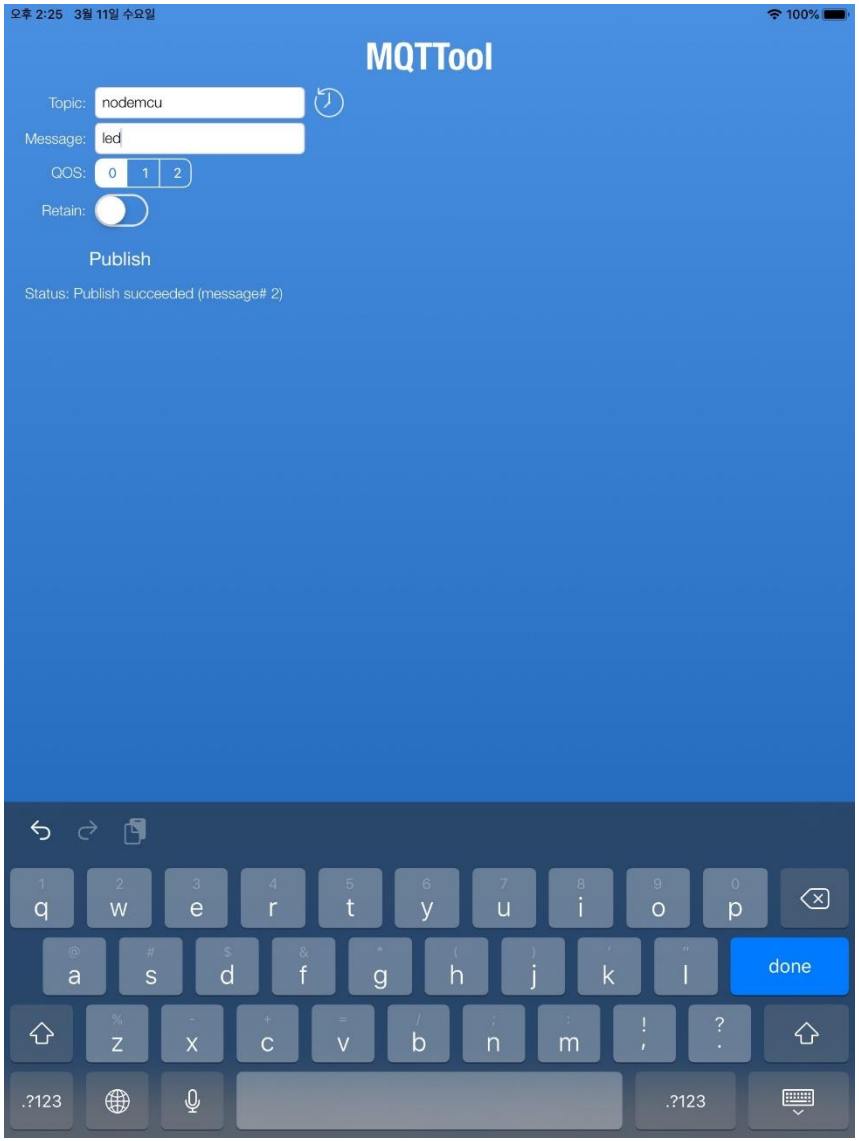


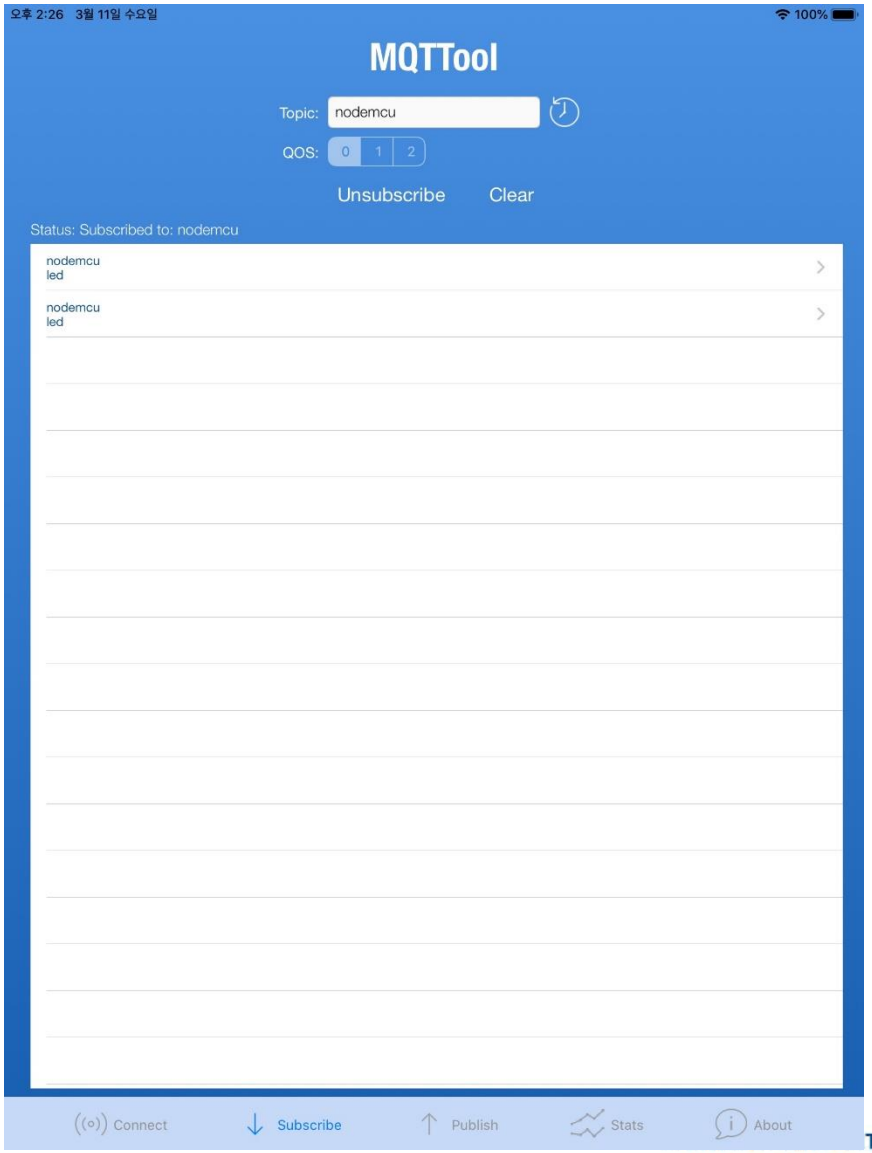
iphone or ipad





iphone or ipad

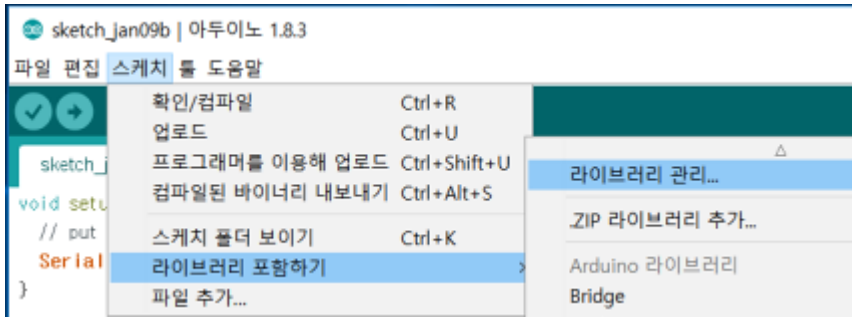


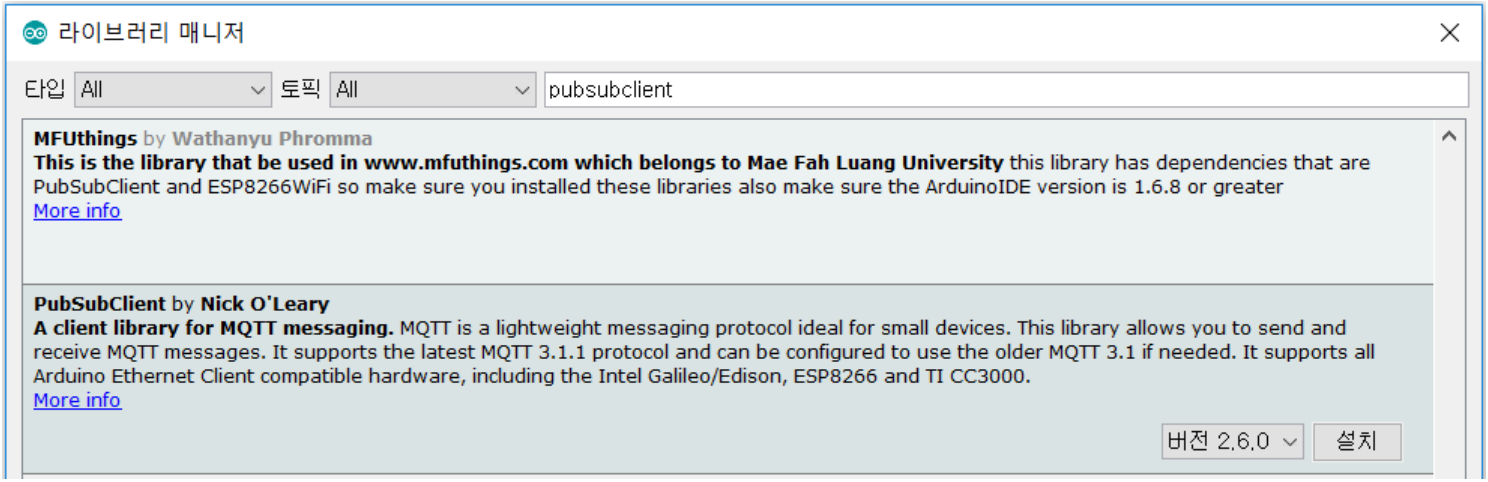


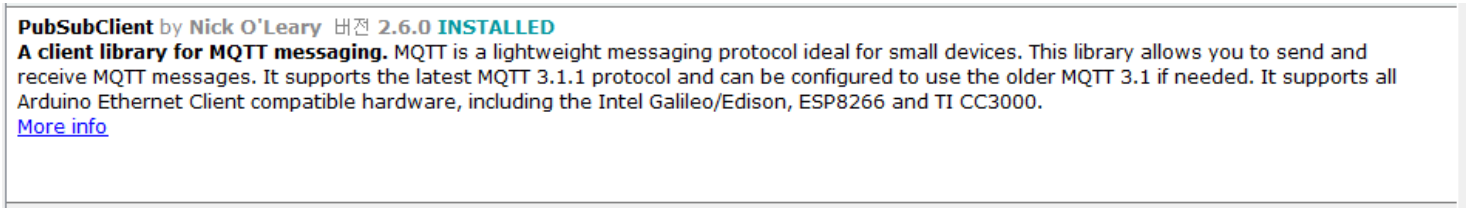
TY

MQTT Client Library

■ 라이브러리 매니저 열어서 pubsubclient로 검색 후, PubSubClient library 설치 확인.







Nodemcu MQTT Client(1)

■ 아래의 코드를 입력, 업로드 후 실행 확인

Topic: ***nodemcu***
Message: ***led***
가 수신될 때마
다 led 가 토글

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

const char* ssid = "iptime_ccrs515";
const char* password = "";
char* clientID = "nodeMCUxxxx";
char* topic = "nodemcuxxxx";
char* server = "iot.eclipse.org"; /* MQTT broker address */
char message_buff[100];
WiFiClient wifiClient;
void callback(char* topic, byte* payload, unsigned int length);
PubSubClient client(server, 1883, callback, wifiClient);

void setup() {
  Serial.begin(115200);
  delay(10);
  pinMode(D0, OUTPUT);
  digitalWrite(D0, LOW);
  Serial.println();
  Serial.println("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
```

수정이 필요할 수 있음
xxxx는 자신의 생일 4자리

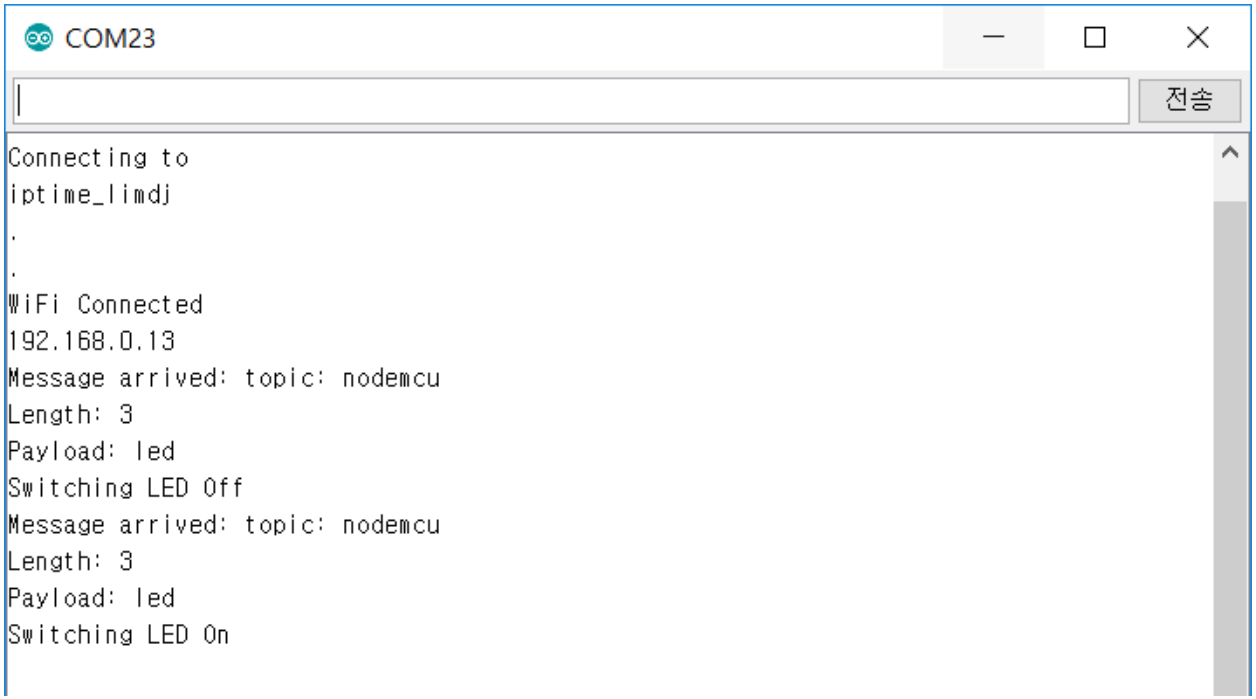
Nodemcu MQTT Client(2)

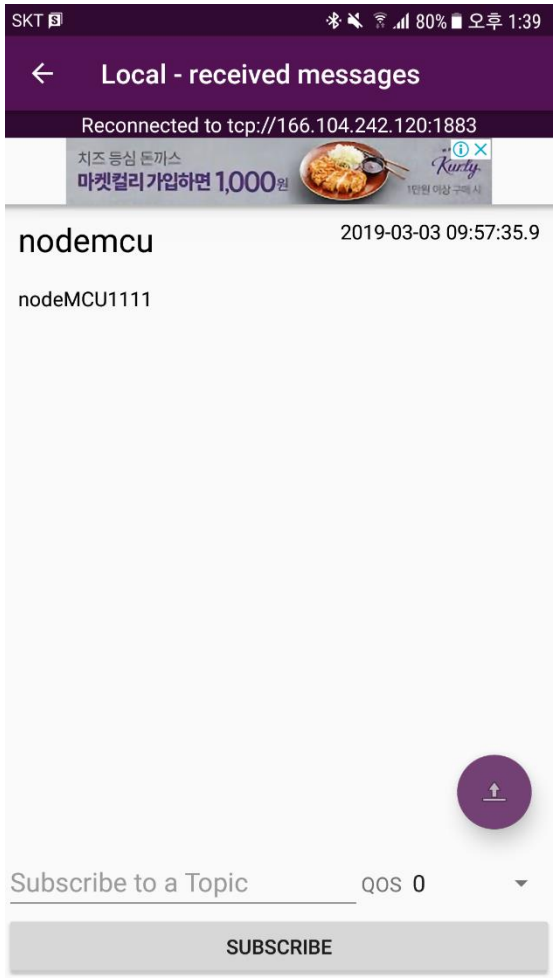
```
 while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.println(".");
  }
  Serial.println("WiFi Connected");
  Serial.println(WiFi.localIP());
  Serial.println(clientID);
  if (client.connect(clientID")){
    client.publish("nodemcu", clientID);
    client.subscribe(topic);
  }
}
void loop() {
  client.loop();
}
void callback(char* topic, byte* payload, unsigned int length) {
  int i = 0;
  Serial.println("Message arrived: topic: " + String(topic));
  Serial.println("Length: "+ String(length,DEC));
```

Nodemcu MQTT Client(3)

```
 for(i=0; i<length; i++){
    message_buff[i] = payload[i];
  }
  message_buff[i]= '₩0';
  String msgString = String(message_buff);
  Serial.println("Payload: "+ msgString);
  int state = digitalRead(D0);
  if (msgString == "led"){
    digitalWrite(D0, !state);
    if(state){
      Serial.println("Switching LED On");
    }
    else{
      Serial.println("Switching LED Off");
    }
  }
}
```

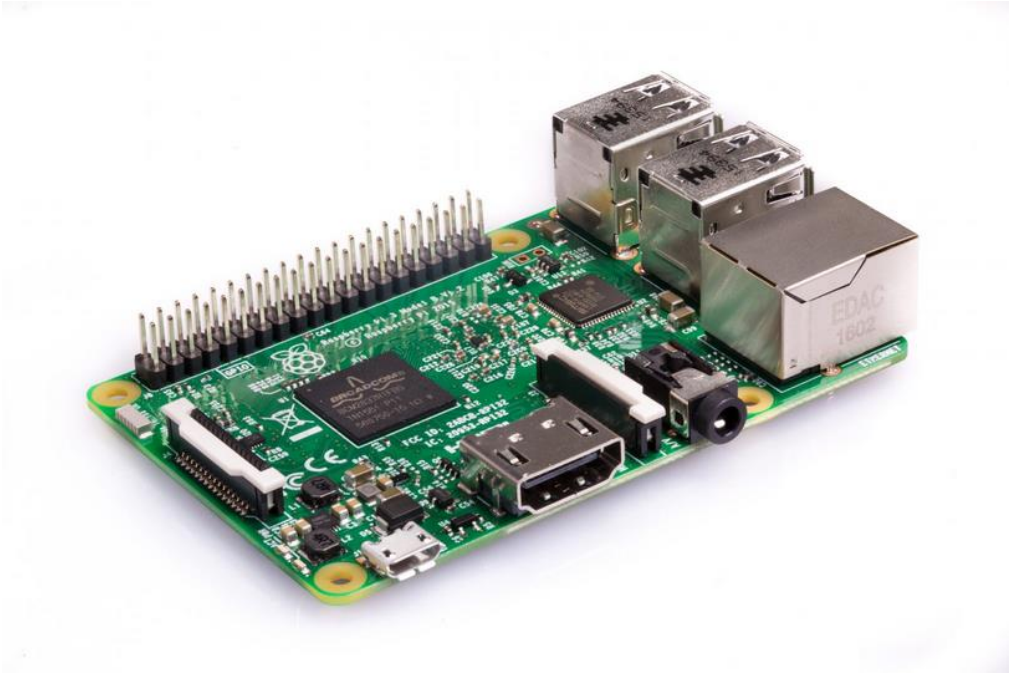
■ PC 혹은 스마트폰의 MQTT Client에서 publish 할 때마다 led가 켜지거나 꺼지는 것을 확인. 또한 시리얼 모니터 열어서 수신 되는 메시지 확인.

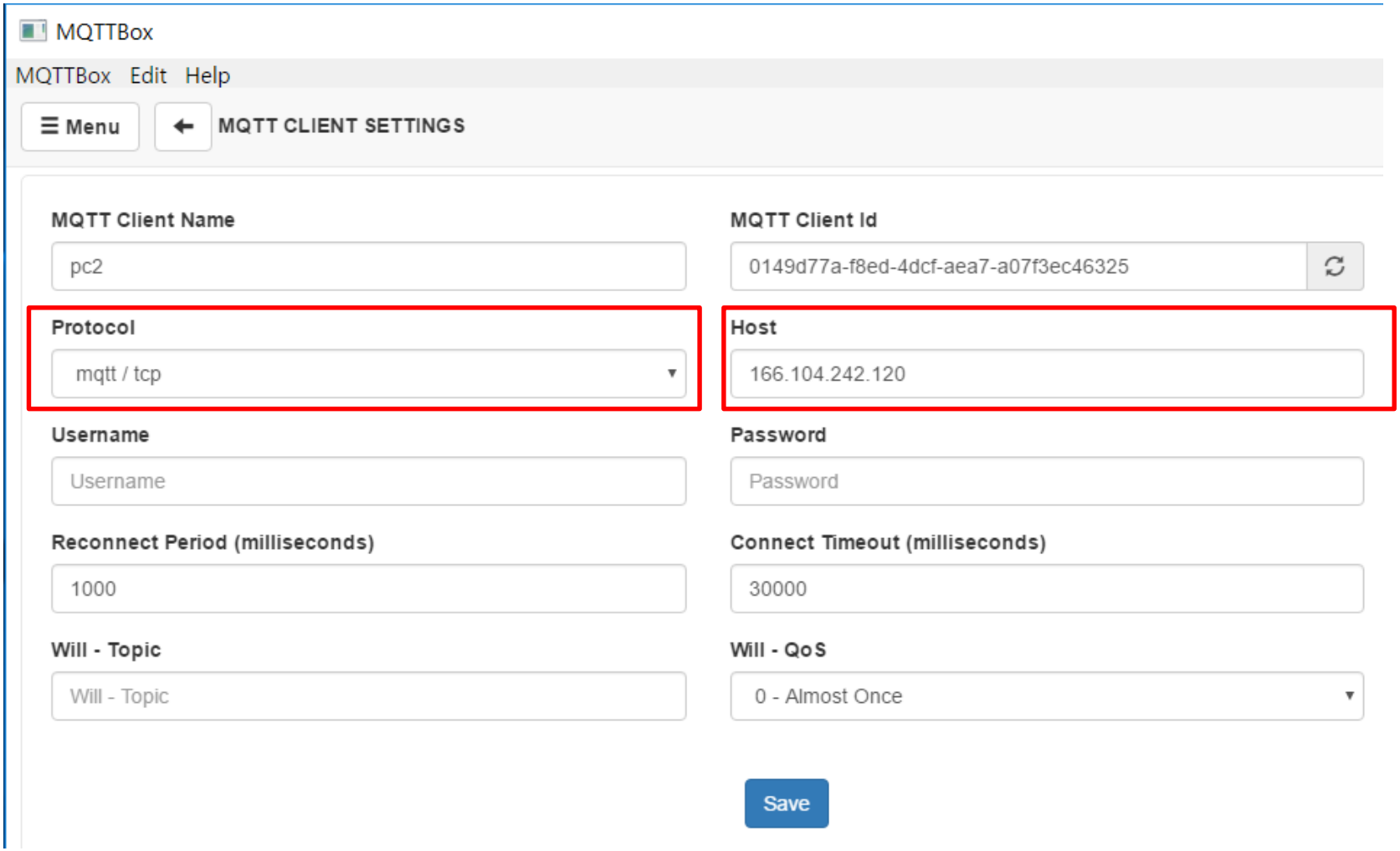




Raspberry PI Local MQTT Broker

■ 라즈베리 파이로 local MQTT broker를 운영할 경우 broker 주소 iot.eclipse.org 대신 라즈베리파이의 local ip 주소(166.104.242.120)를 입력하여 동일한 실습을 진행. 이때, PC용 client, 스마트폰용 client, Nodemcu 용 client 프로그램의 broker 주소를 모두 변경해야 함.

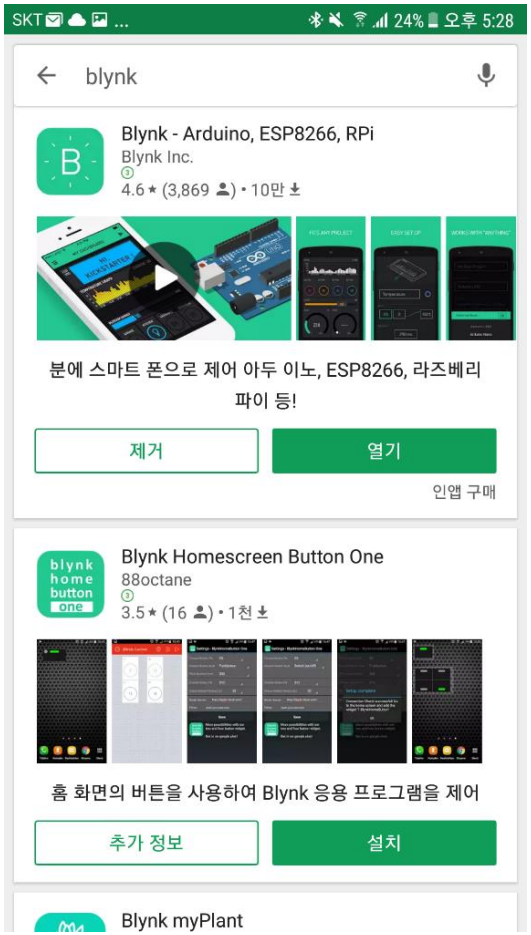



MQTTBox
MQTTBox Edit Help
Menu
MQTT CLIENT SETTINGS
MQTT Client Name
pc2
MQTT Client Id
0149d77a-f8ed-4dcf-aea7-a07f3ec46325
Protocol
mqtt / tcp
Host
166.104.242.120
Username
Username
Password
Password
Reconnect Period (milliseconds)
1000
Connect Timeout (milliseconds)
30000
Will - Topic
Will - Topic
Will - QoS
0 - Almost Once
Save

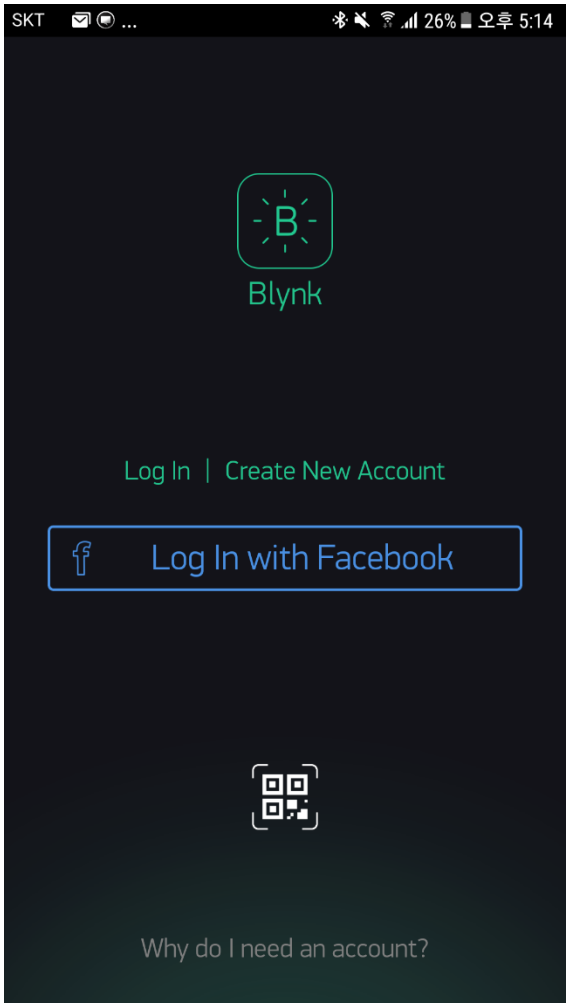
초등학생도 할 수 있는 IoT: Blynk

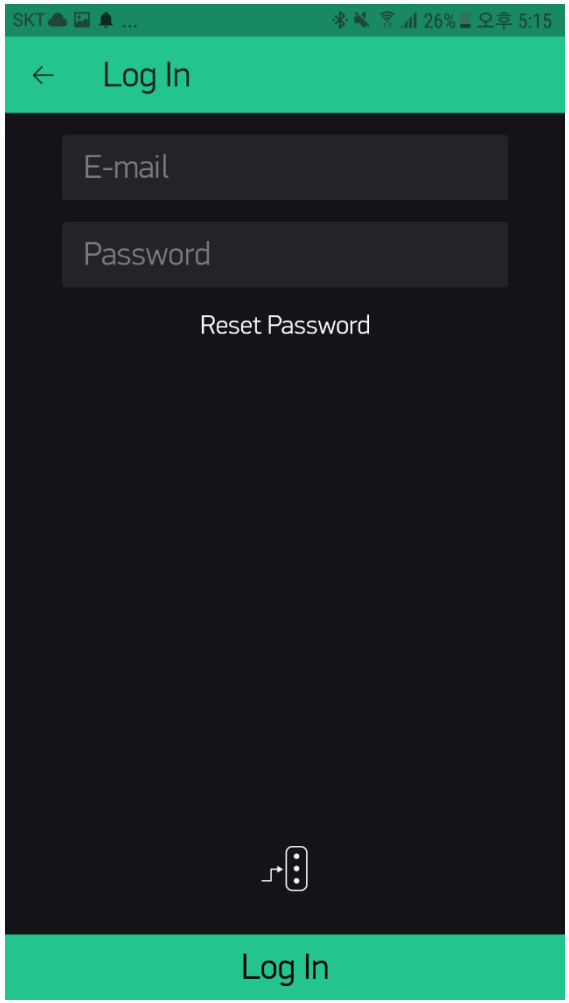
스마트폰에 Blynk 앱 설치

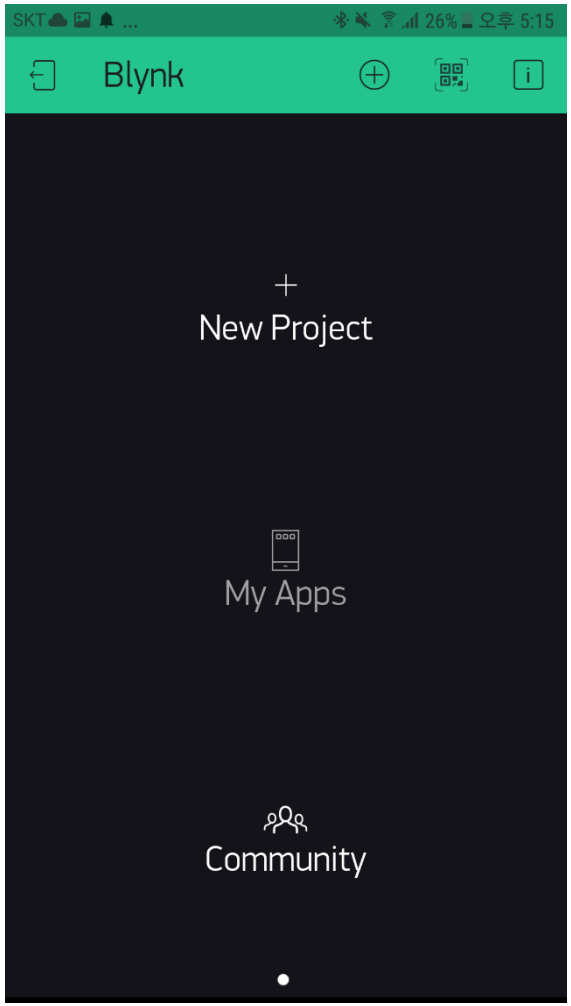
- 스마트폰의 앱 스토어에서 Blynk 를 검색해서 설치
- 가입을 위해서 E-mail 주소가 필요하고, 작업 중 PC에서 E-mail을 열어볼 수 있어야 함



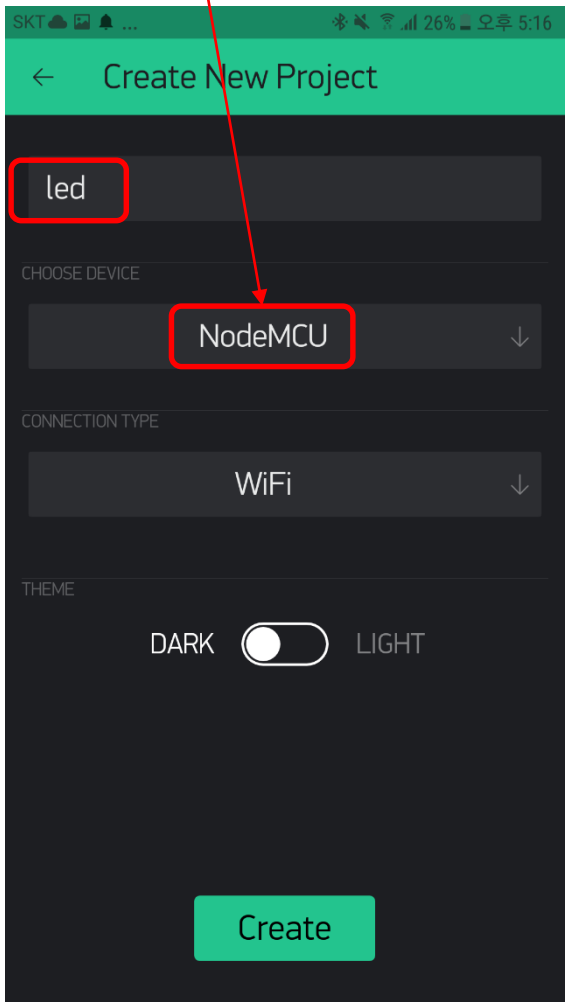
Log In and Create a New Project

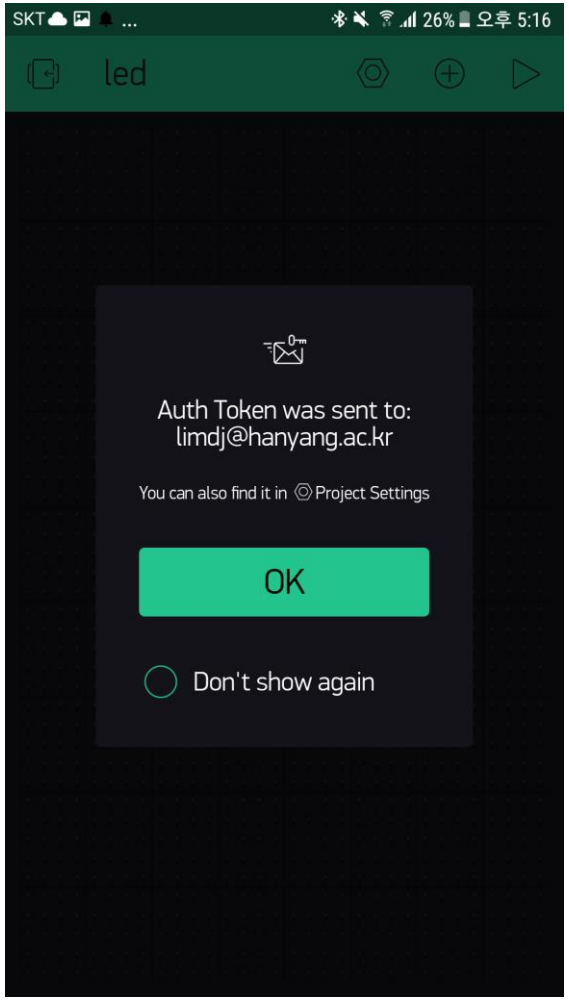


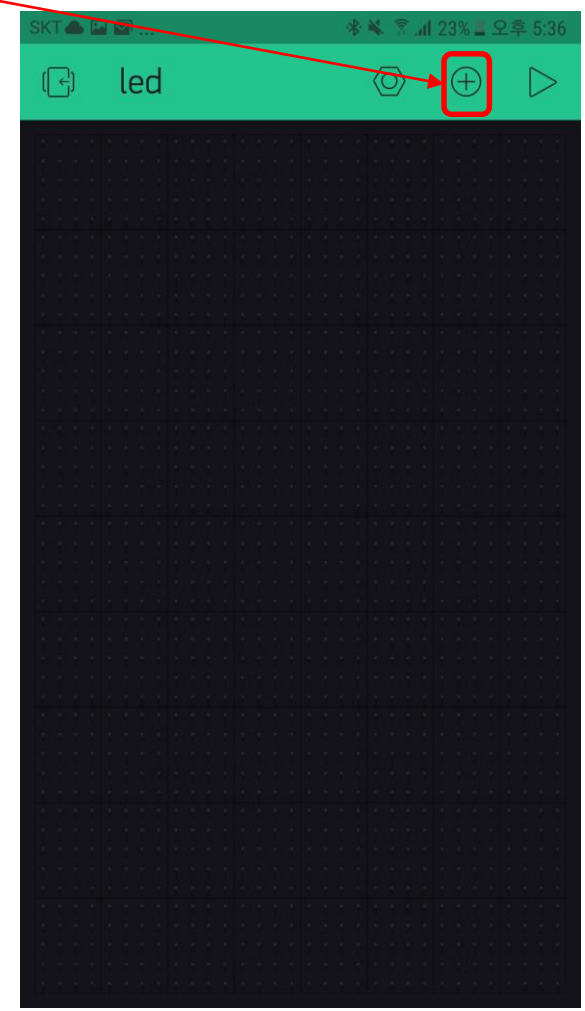




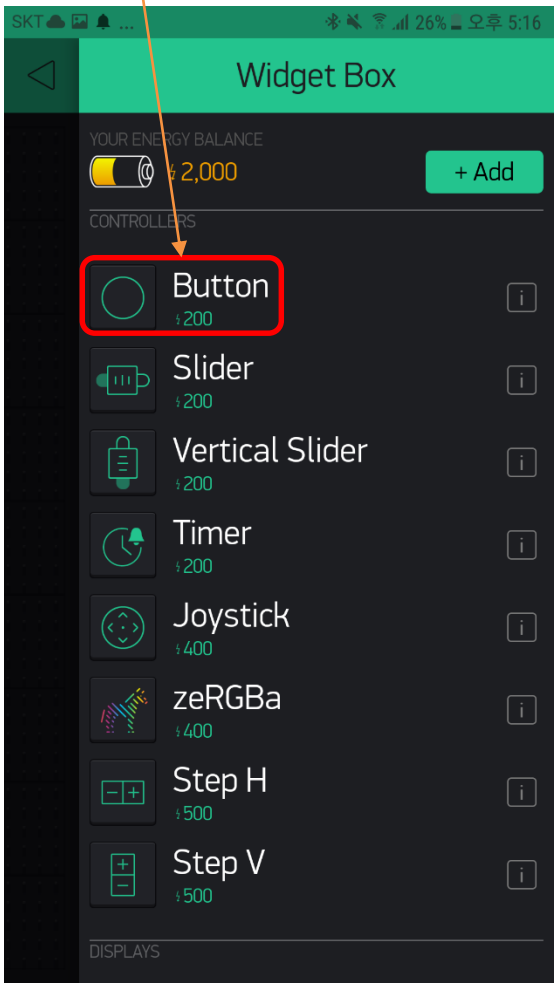
Choose Device and Add Widget

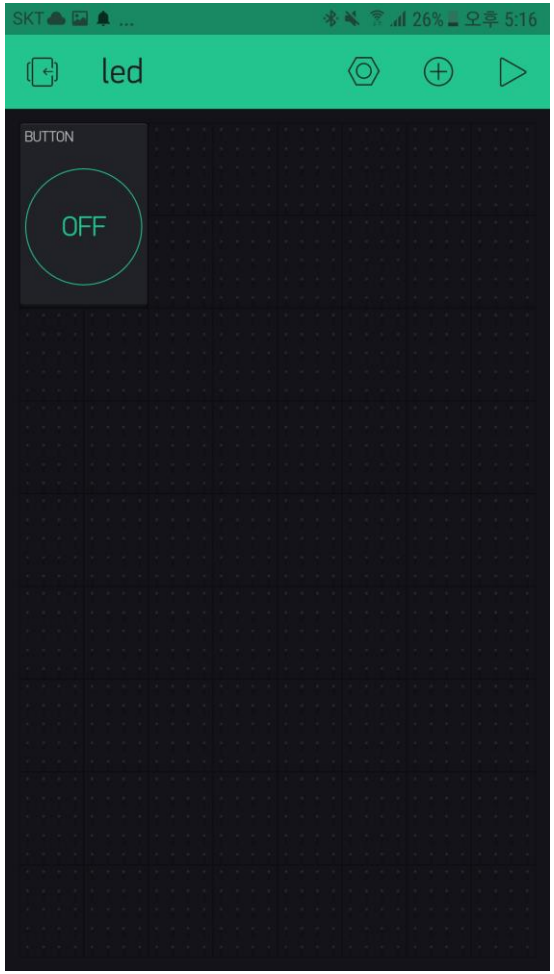


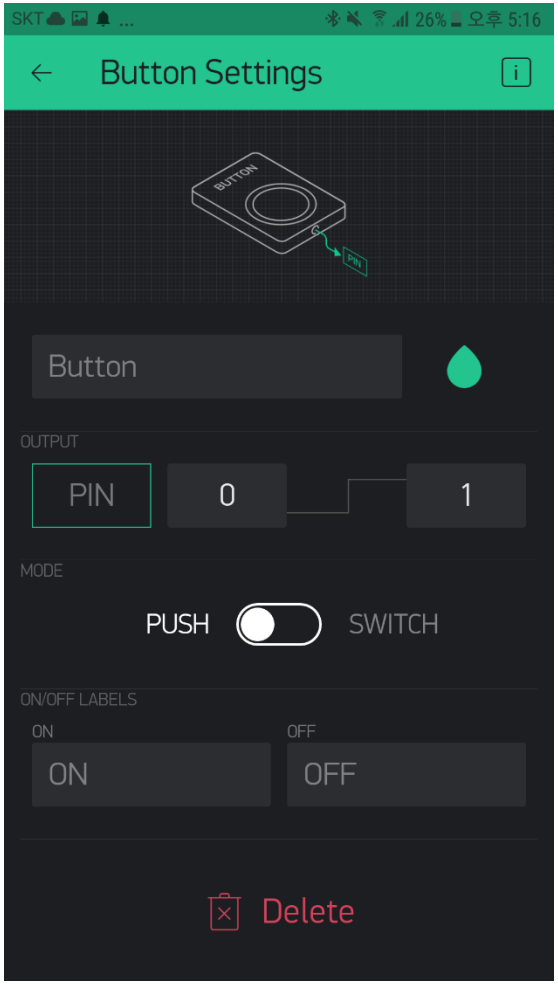




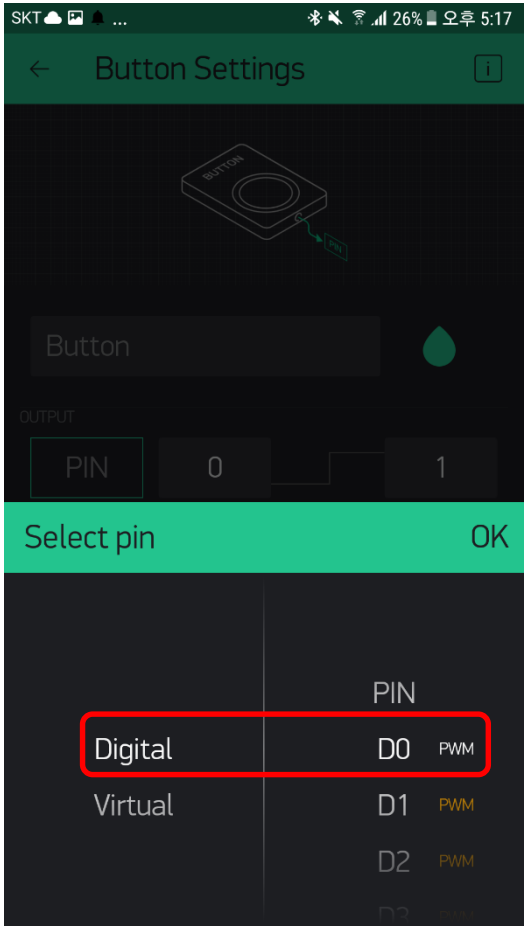
Add Button

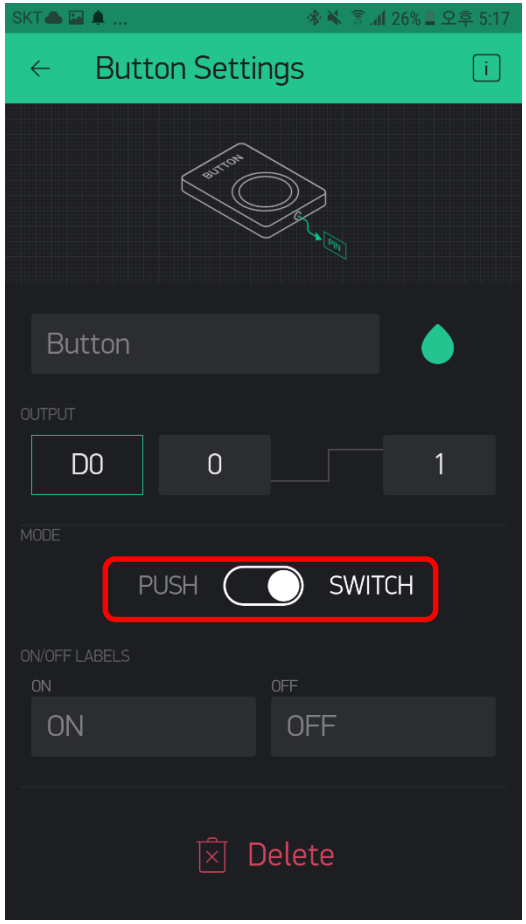

Widget Box
YOUR ENERGY BALANCE
2,000
+ Add
CONTROLLERS
Button
200
Slider
200
Vertical Slider
200
Timer
200
Joystick
400
zeRGBa
400
Step H
500
Step V
500
DISPLAYS

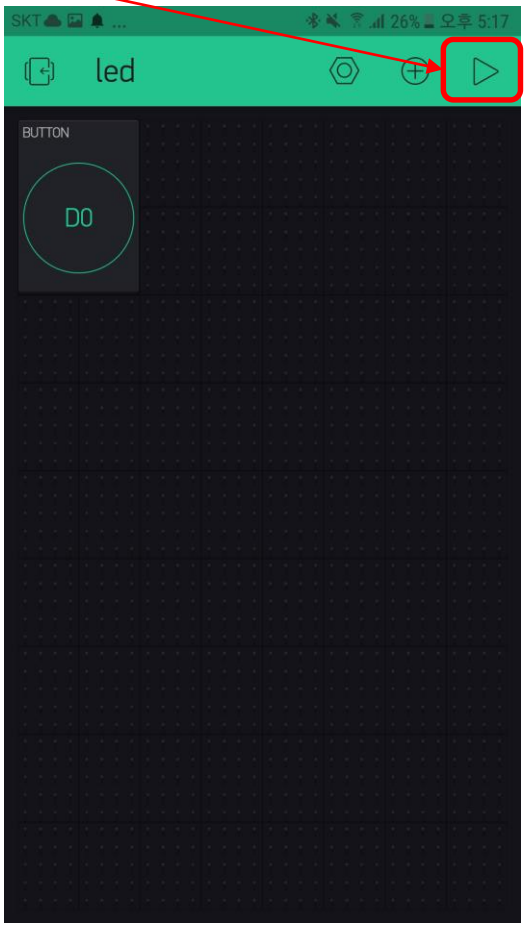

led
BUTTON
OFF


Button Settings
Button
OUTPUT
PIN
0
1
MODE
PUSH
SWITCH
ON/OFF LABELS
ON
OFF
ON
OFF
Delete

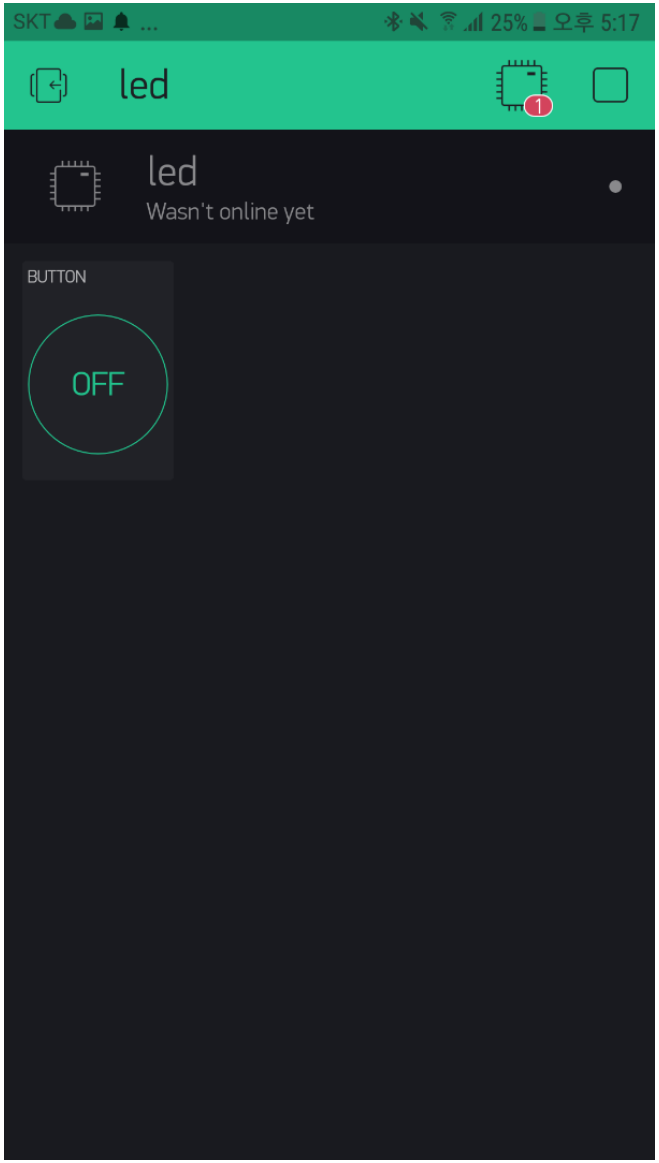
Change settings and press Play button





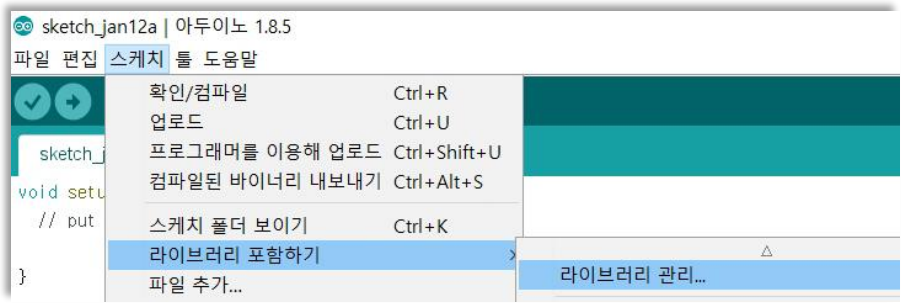


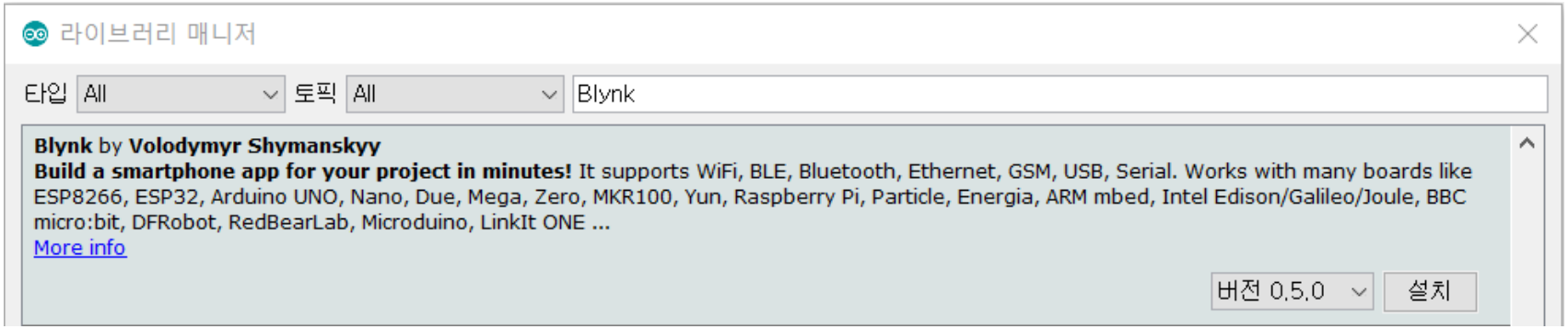
Wait for the device to be online

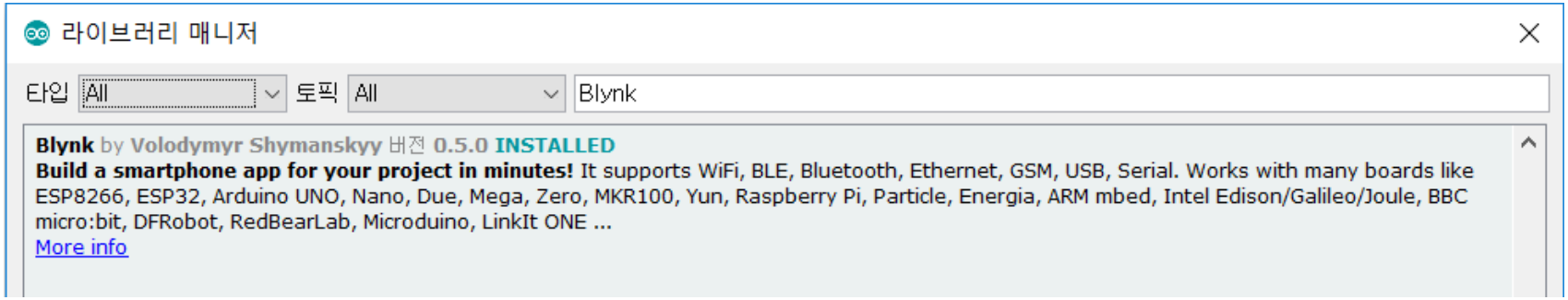


Open Arduino IDE

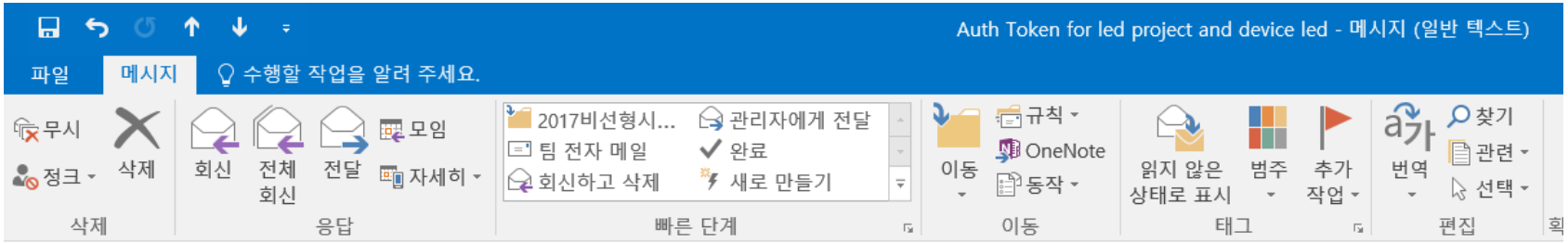
■ 라이브러리 매니저에서 Blynk library 설치







■ Open the e-mail for Token



2018-01-10 (수) 오후 5:16

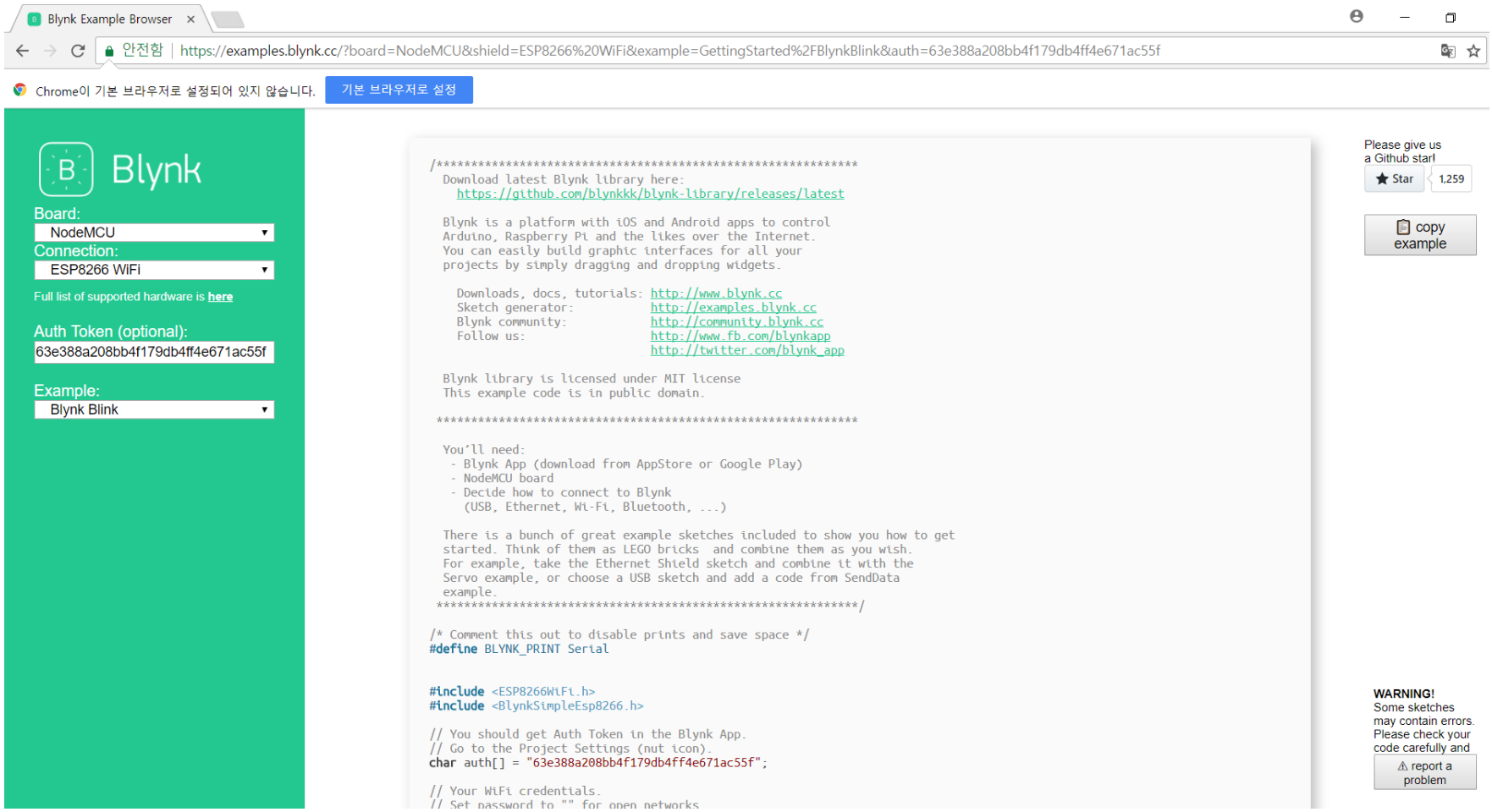
Blynk <dispatcher@blynk.io>

Auth Token for led project and device led

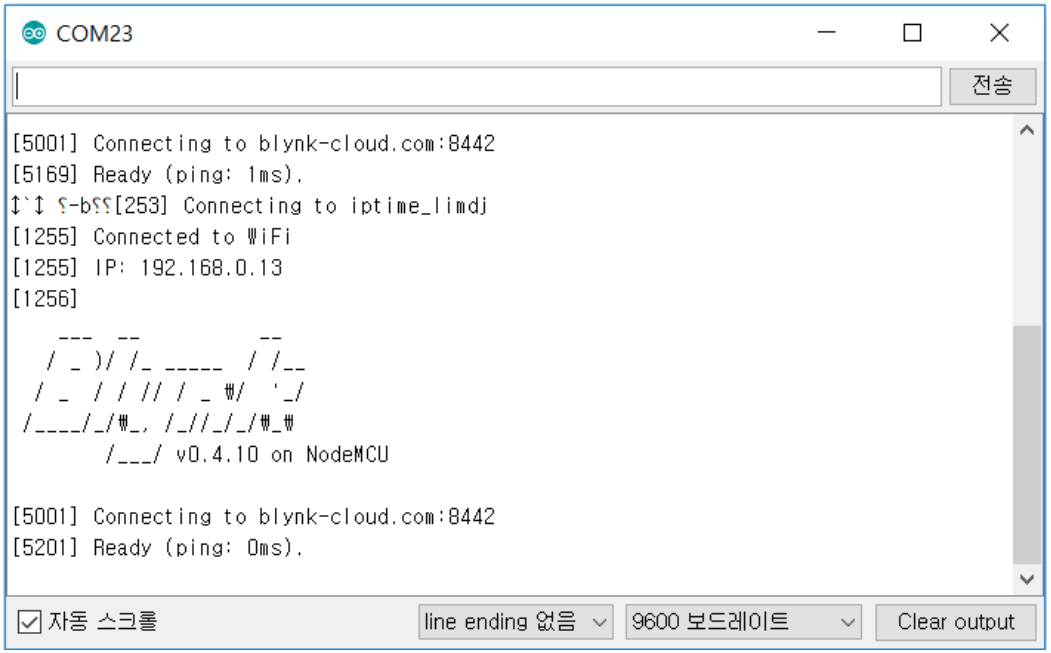
받는 사람 limdj@hanyang.ac.kr

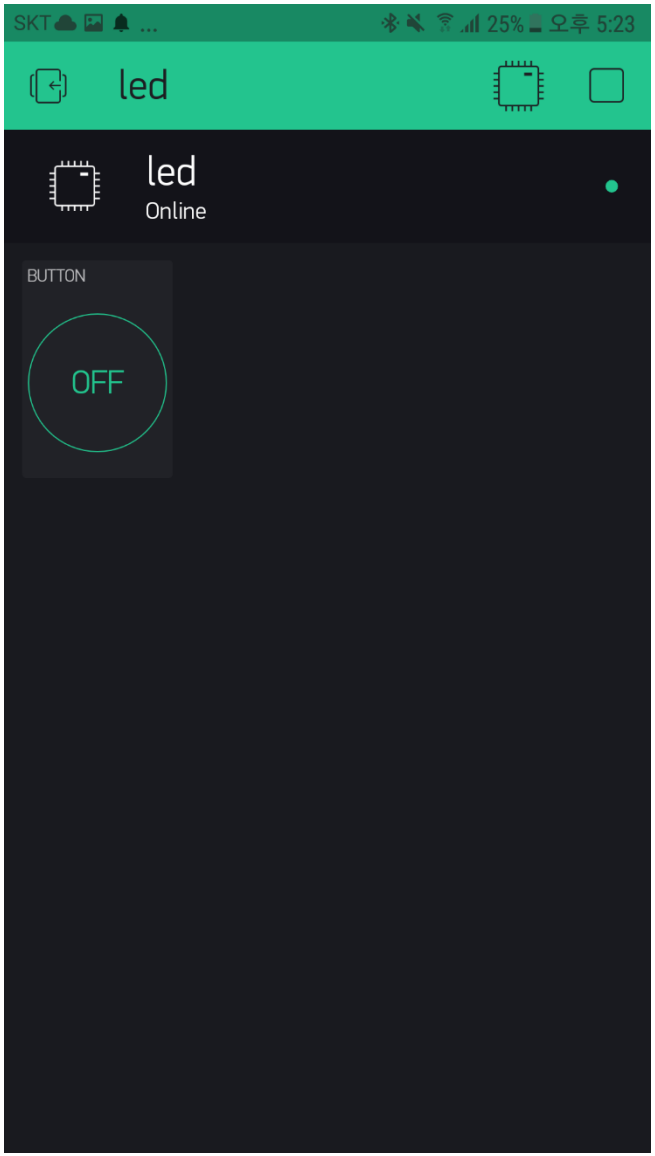
```
Auth Token : 63e388a208bb4f179db4ff4e671ac55f

Happy Blynking!
-
Getting Started Guide -> https://www.blynk.cc/getting-started
Documentation -> http://docs.blynk.cc/
Sketch generator -> https://examples.blynk.cc/

Latest Blynk library -> https://github.com/blynkkk/blynk-library/releases/download/v0.4.10/Blynk Release v0.4.10.zip
Latest Blynk server -> https://github.com/blynkkk/blynk-server/releases/download/v0.29.1/server-0.29.1.jar
-
https://www.blynk.cc
twitter.com/blynk_app
www.facebook.com/blynkapp
```

- Open in Chrome and copy/paste Arduino code
- Change auth[]="XXX" and ssid[]="XXX" in the code

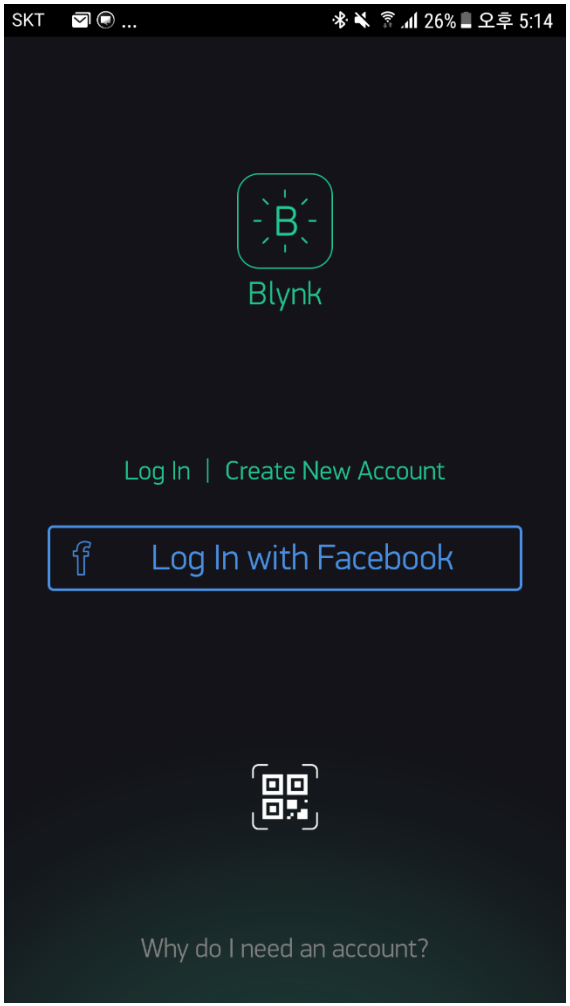


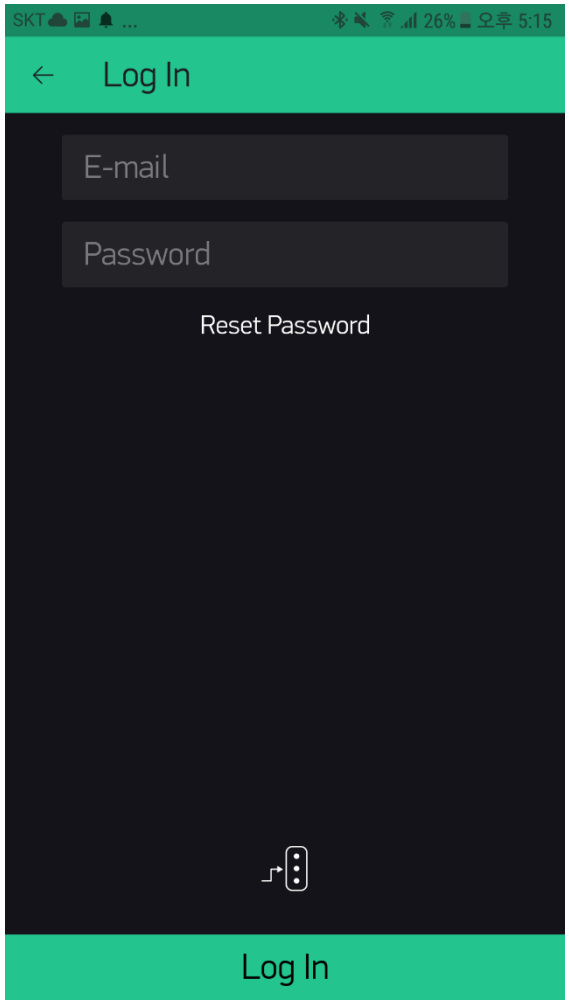
Build and Run

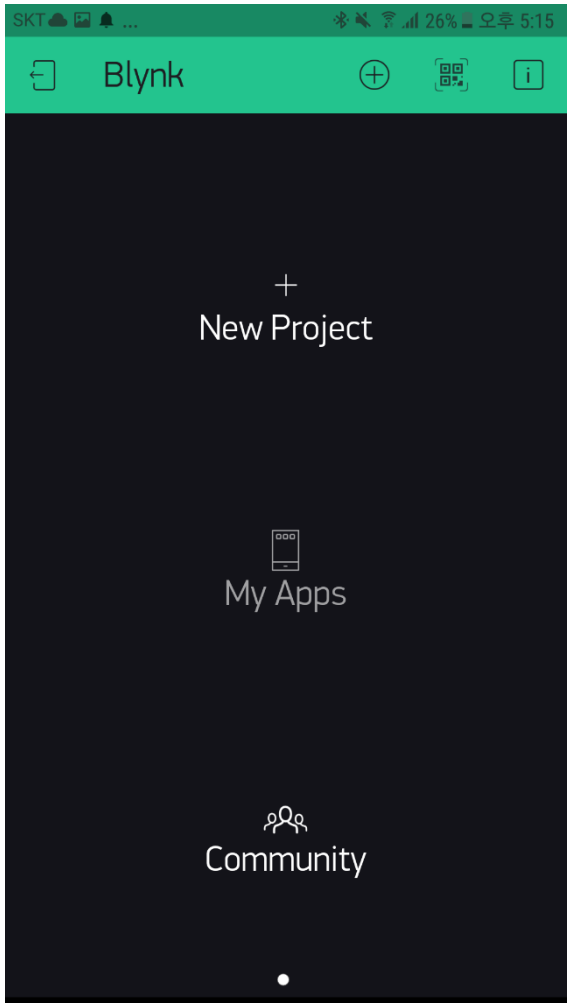




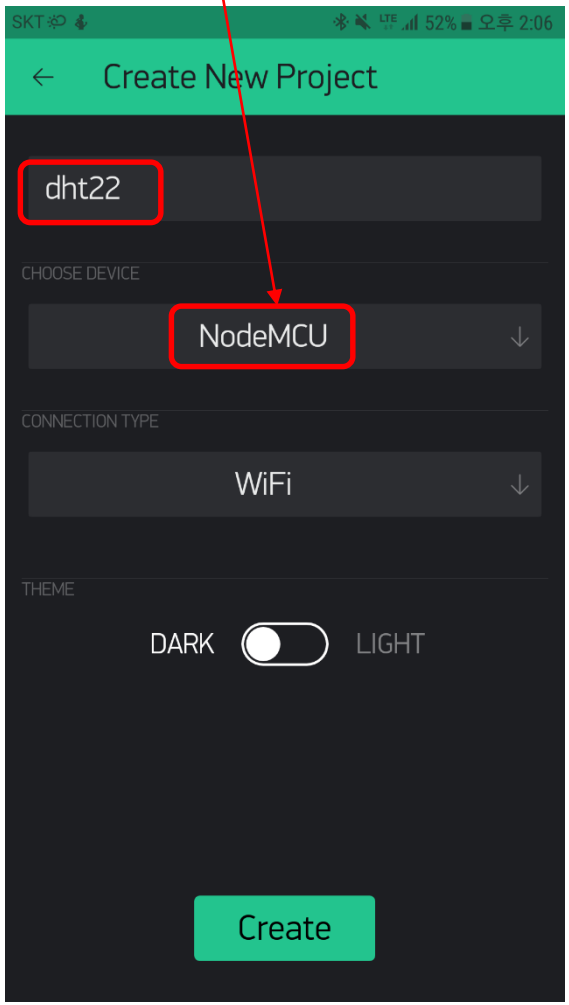
Log In and Create DHT22 Project

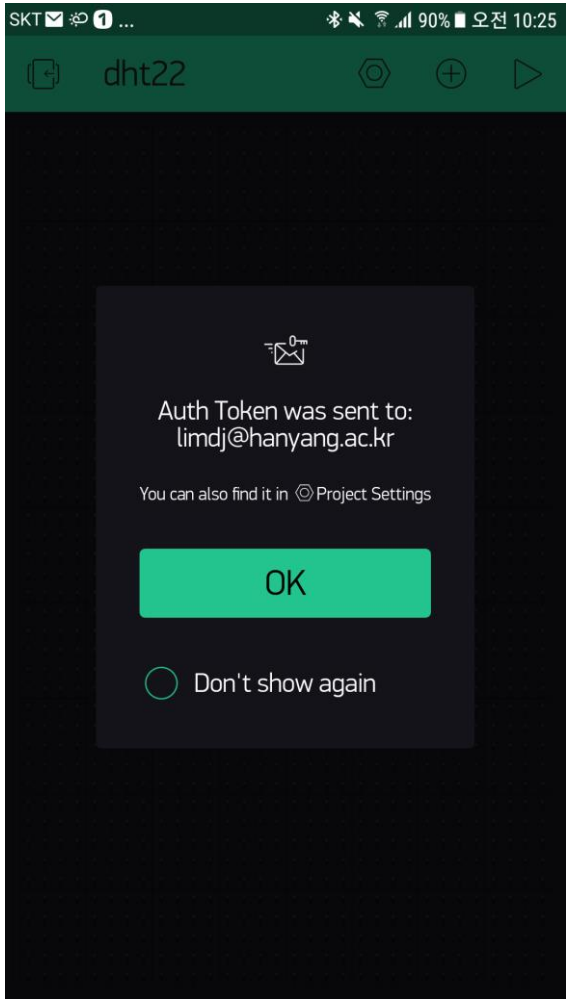


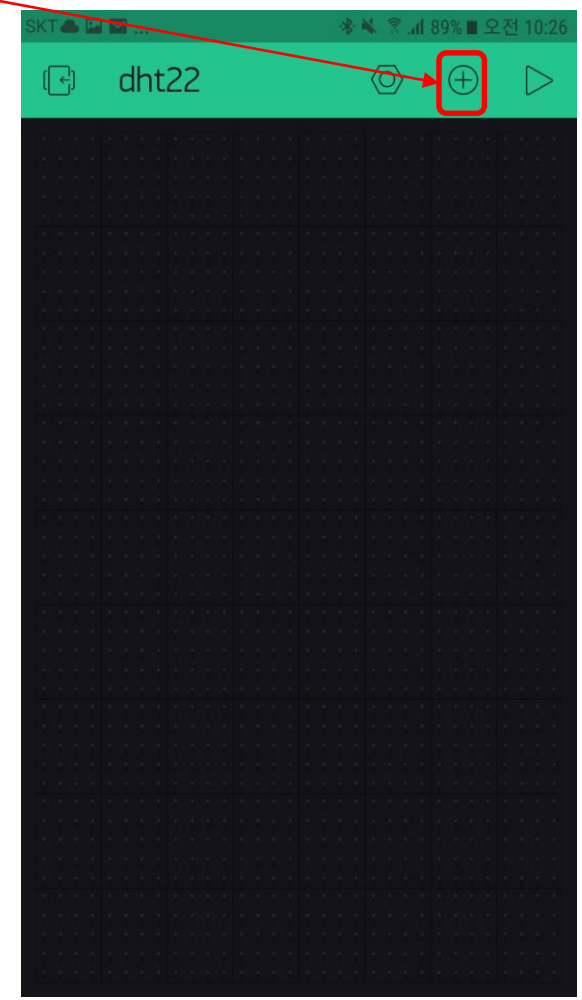




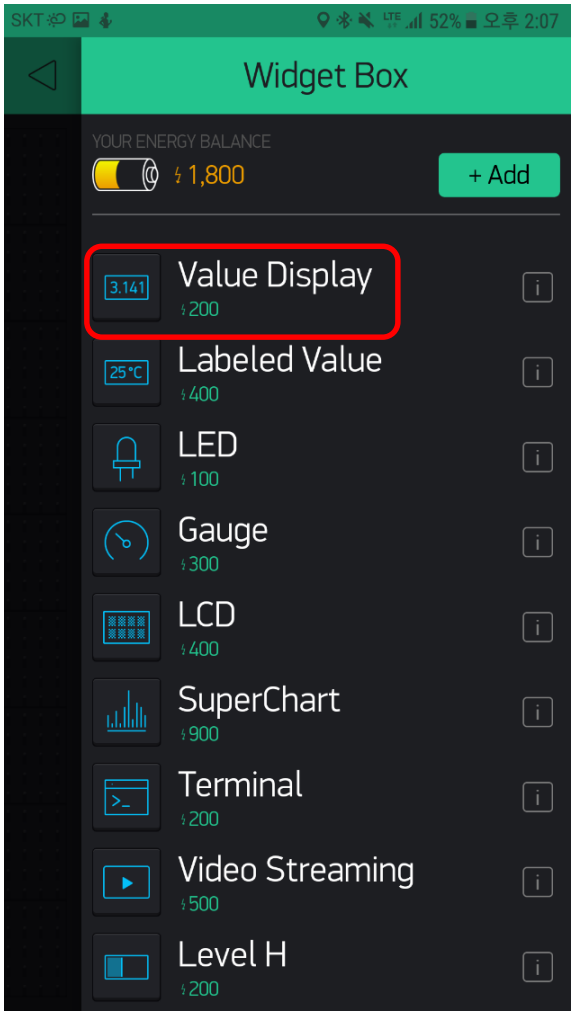
Choose Device and Add Widget

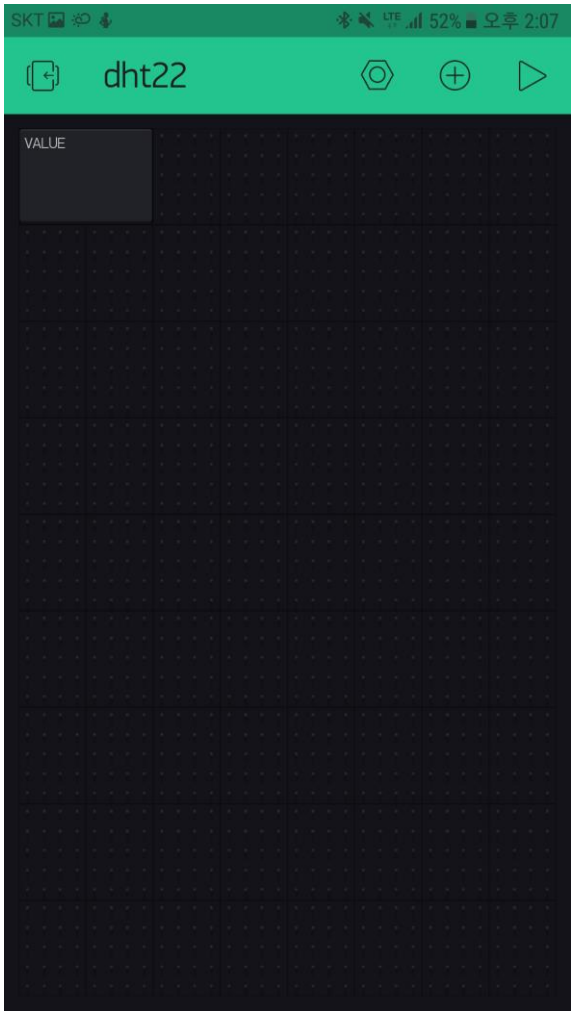


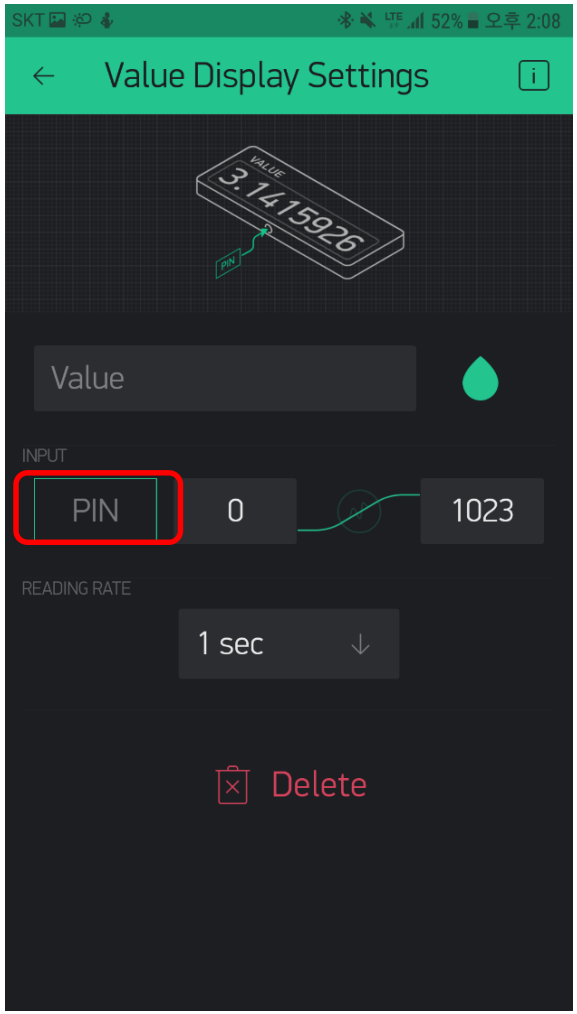




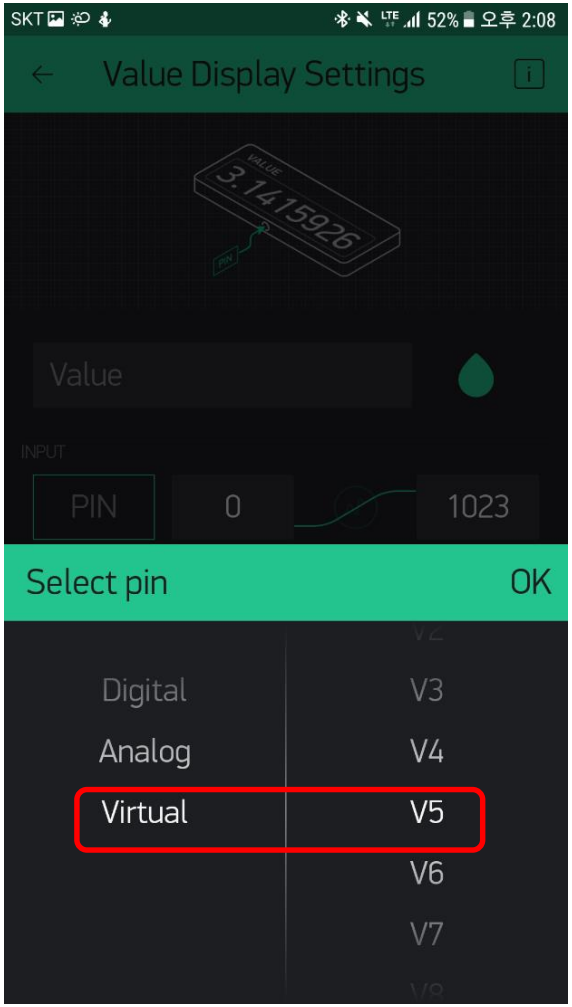
Add Value Display

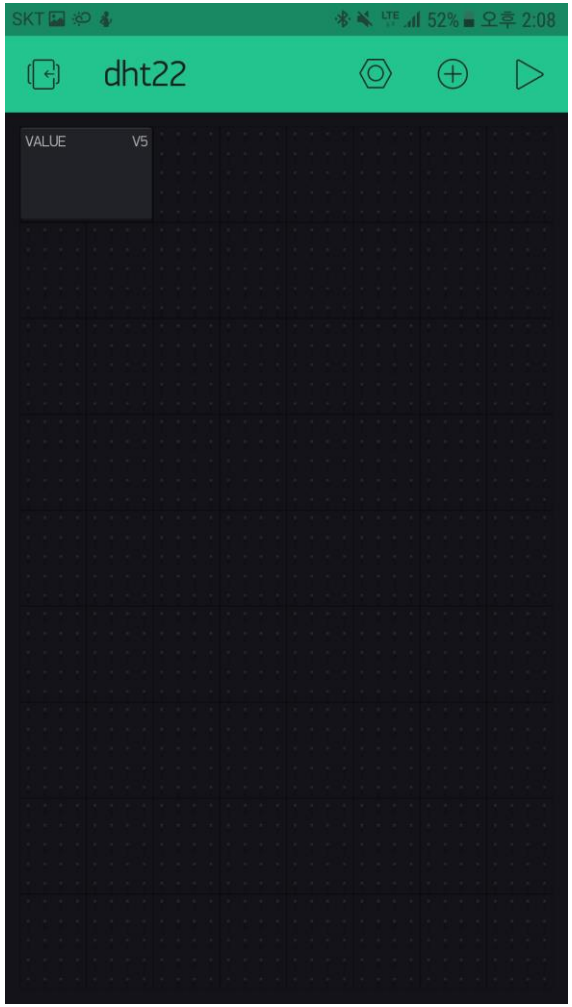


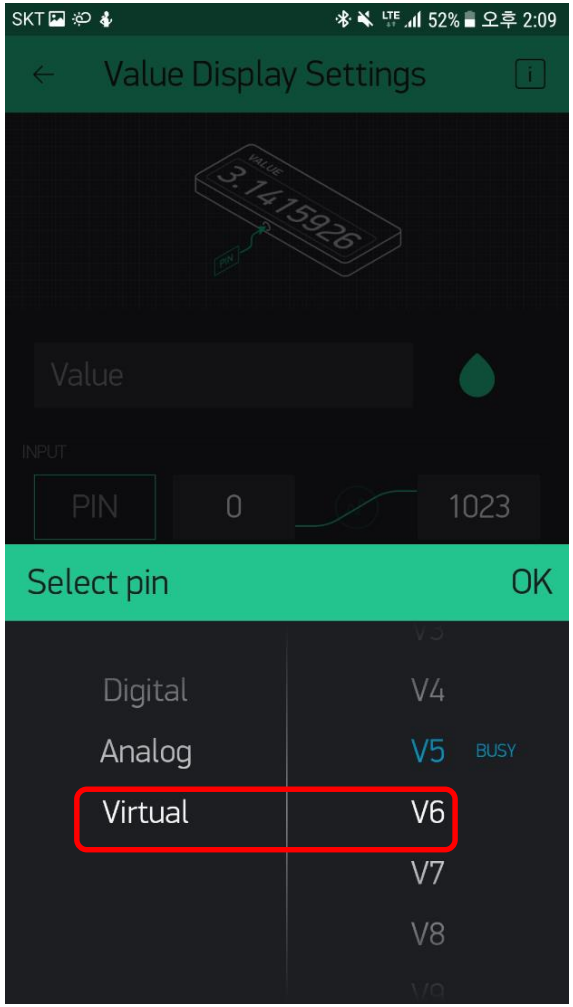




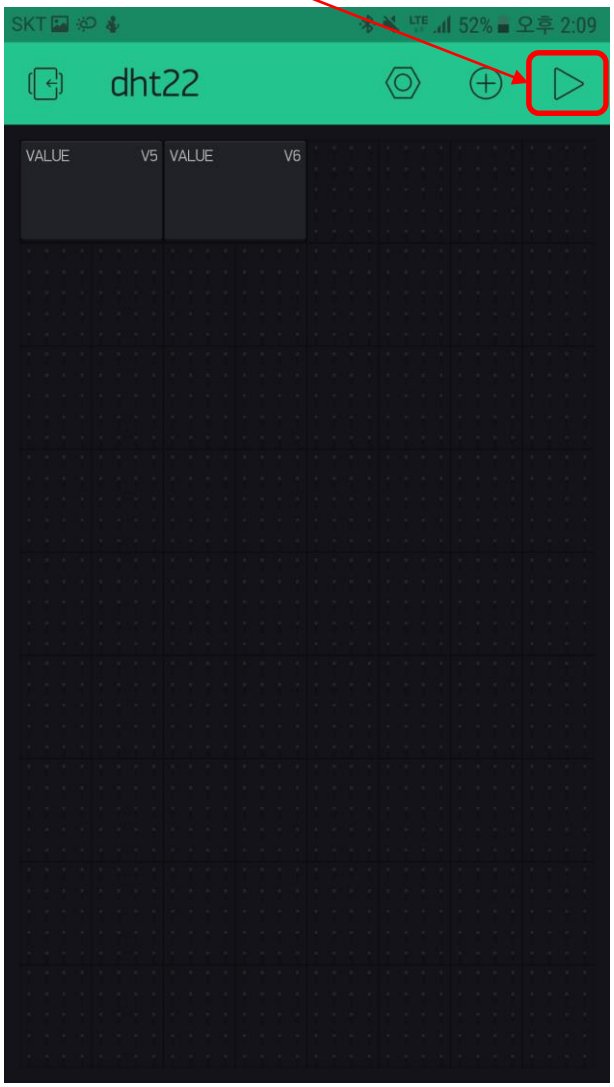
Add another Value Display

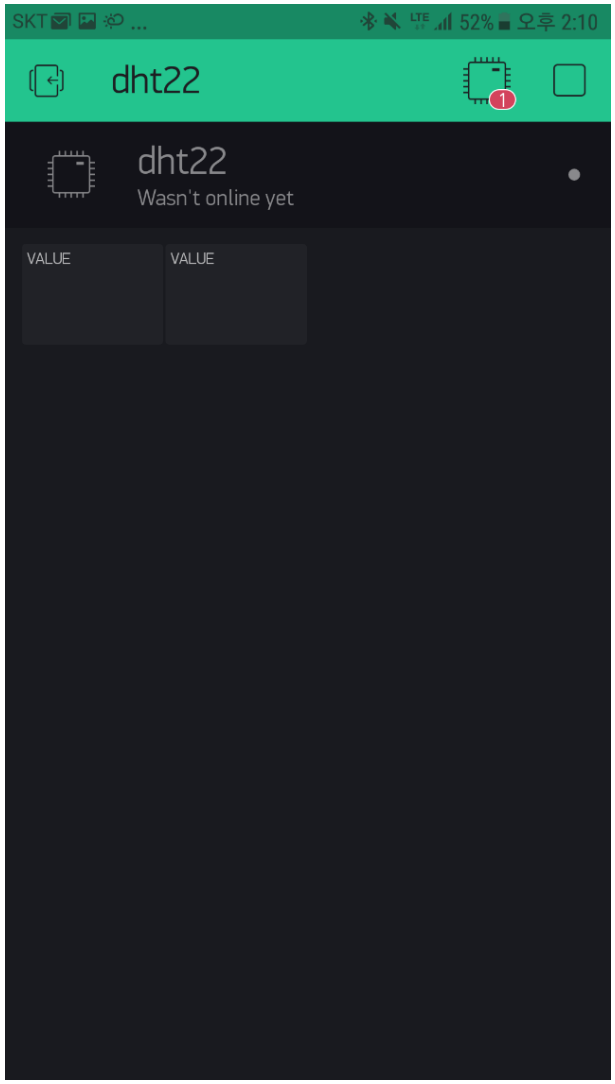




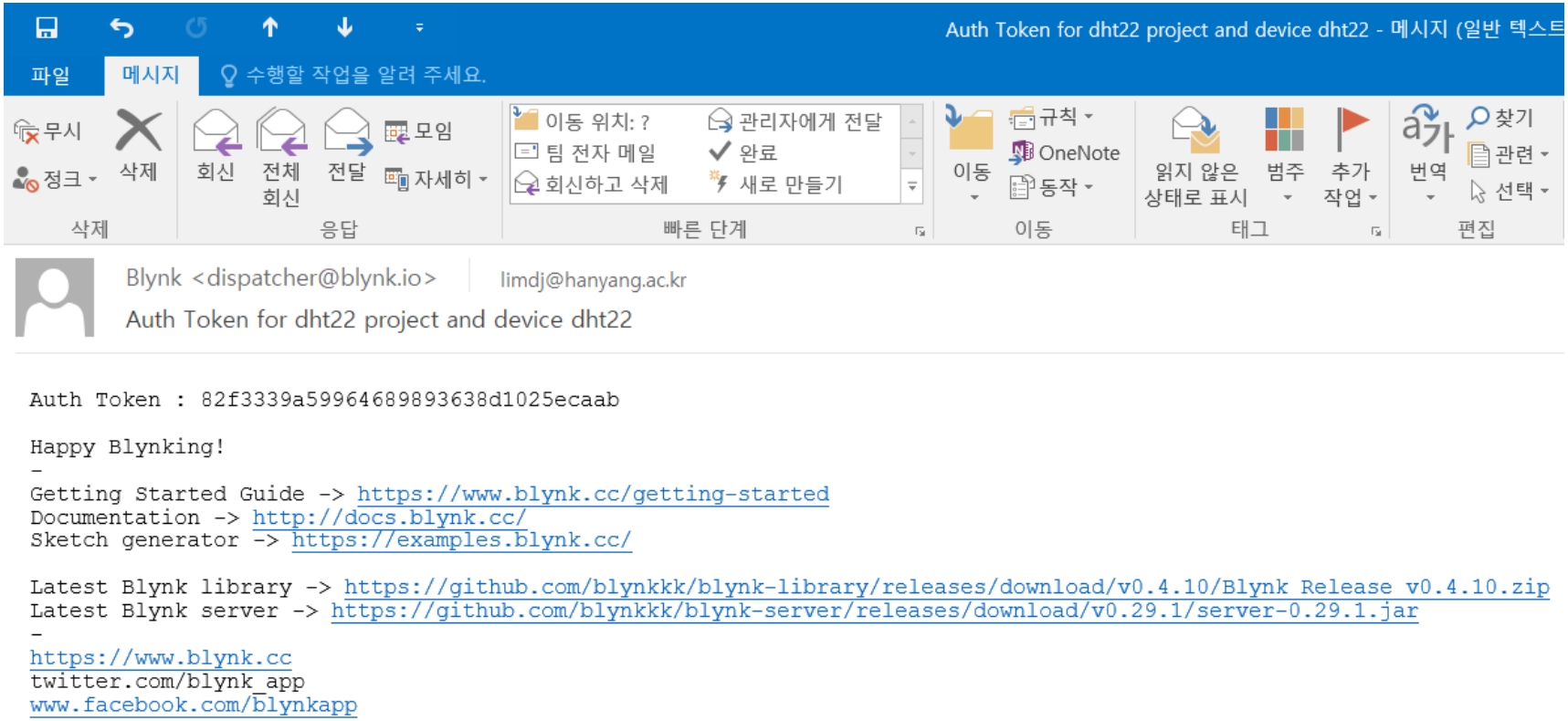


Press play button and wait for the device to be online

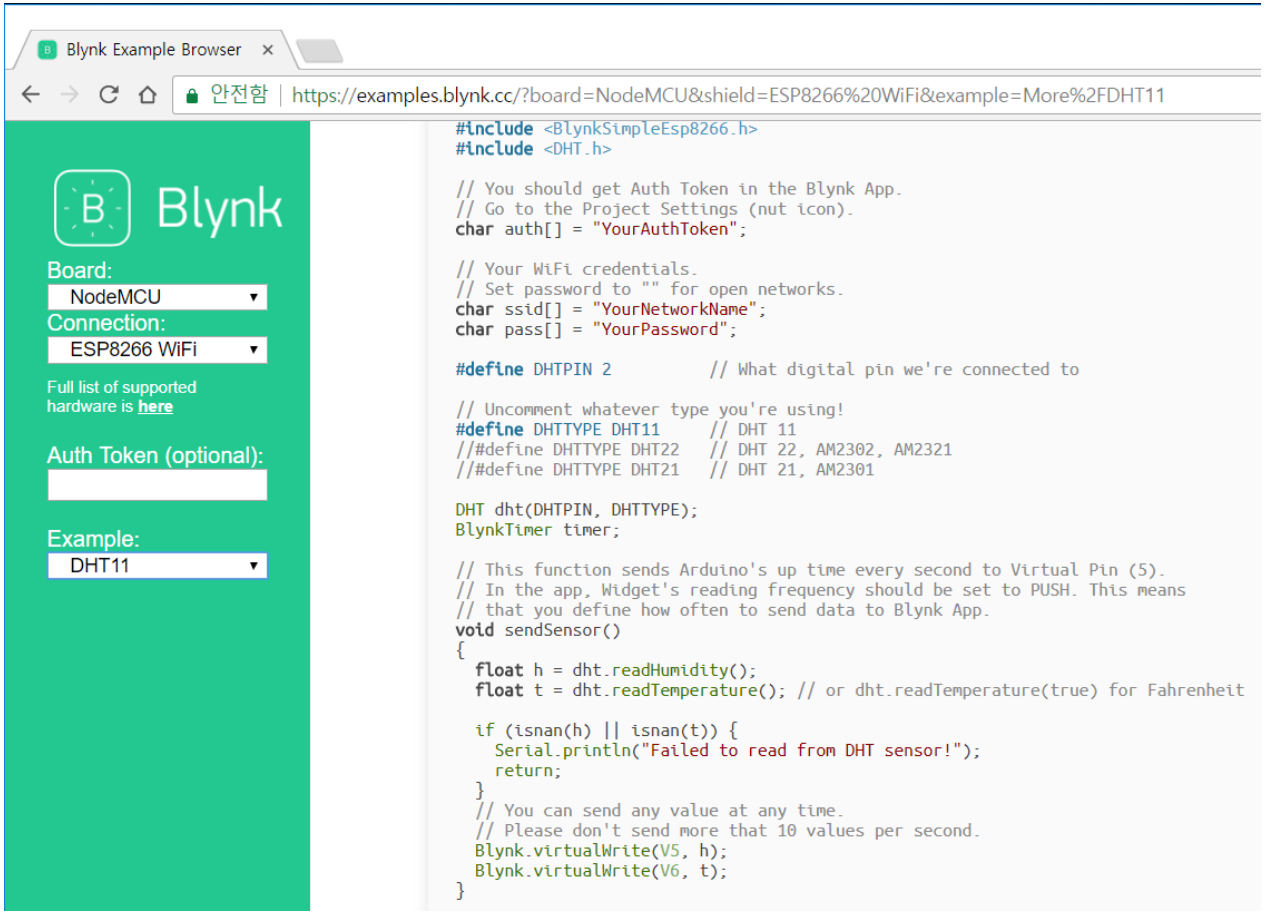




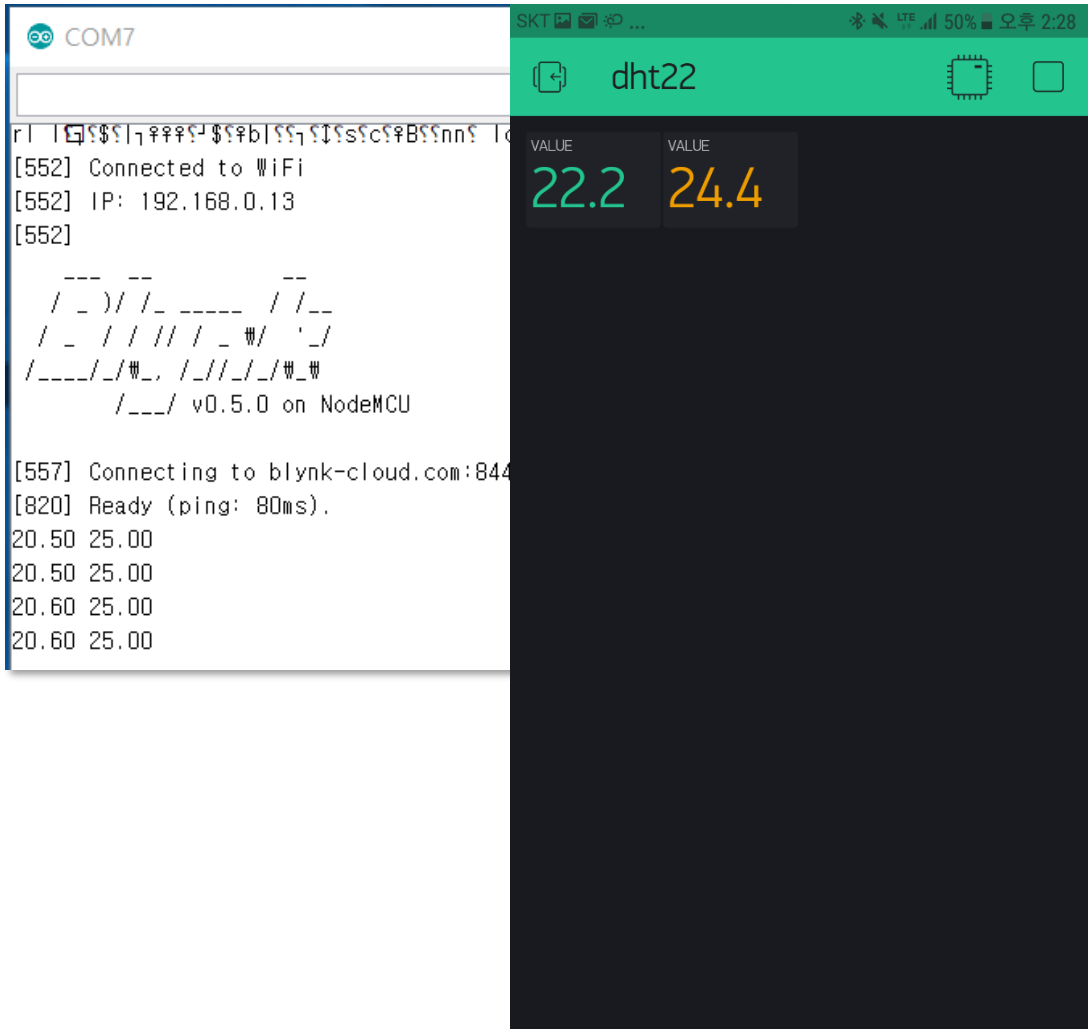
■ Open the e-mail for Token

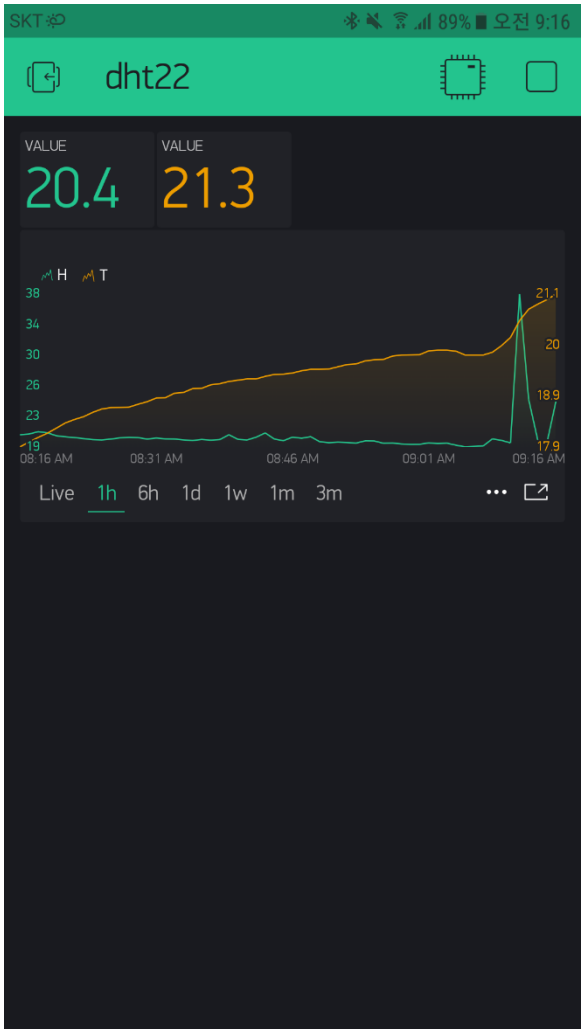


- Open in Chrome and copy/paste Arduino code
- Change auth[]="XXX" and ssid[]="XXX" in the code
- Change to #define DHTTYPE DHT22, #define DHTPIN D2



Build and Run





Thank you.