NodeMCU 입문하기

목차

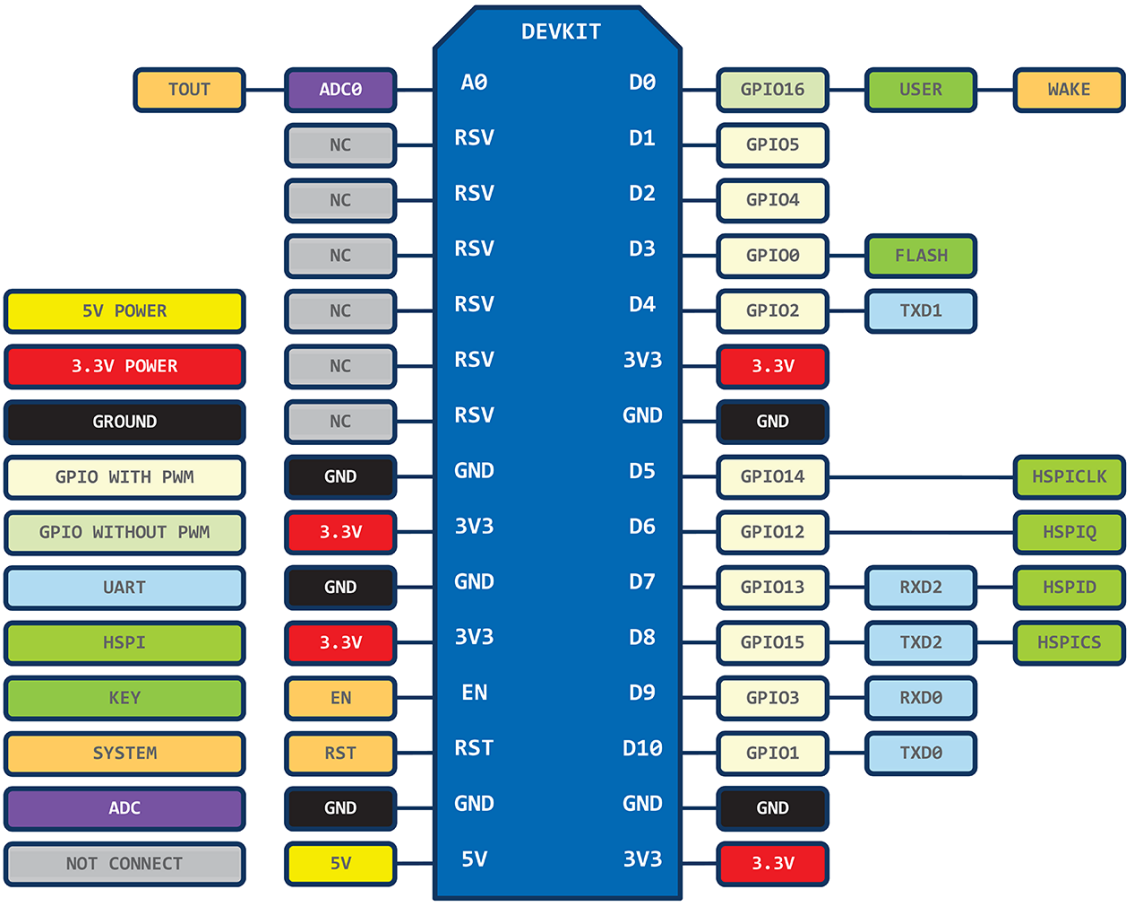
1. NodeMCU란?

NodeMCU (노드엠씨유)는 오픈소스 사물인터넷 (IoT) 플랫폼으로 와이파이 기능이 구현된 MCU 개발보드라고 생각할 수 있습니다. 아두이노 및 라즈베리파이를 사용해보신 분이라면 한번쯤 들어보셨을 ESP8266 와이파이 모듈을 개발한 ESPRESSIF사의 ESP8266-12 모듈을 사용합니다. 이름에서 의미하는 것처럼 사물인터넷 노드용 MCU이며, 작은 크기와 저렴한 가격으로 네트워크 기능이 구현된 아두이노라고 생각할 수 있습니다.

메카솔루션에서 판매하고 있는 NodeMCU V1.0은 ESP-12E를 사용하며, USB 신호를 UART 신호로 변경하기 위해서 CP2012 칩셋을 사용하고 있습니다. 마이크로USB의 앞쪽에 있는 조그만 칩이 CP2012이고, 오른쪽의 안테나와 알루미늄 쉴드가 ESP8266-12E SOC이며, 중앙의 검정색 칩이 LM1117로 5V 전압을 3.3V로 변환합니다.

Specifications of ESP-12E WiFi Module

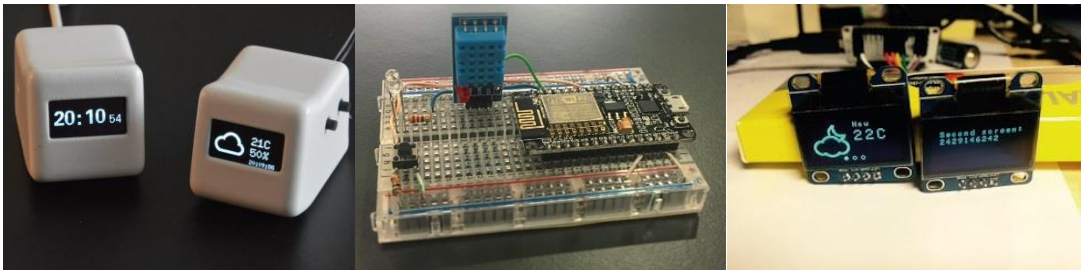
Wireless Standard	IEEE 802.11 b/g/n
Frequency Range	2.412 - 2.484 GHz
Power Transmission	802.11b : +16 ± 2 dBm (at 11 Mbps) 802.11g : +14 ± 2 dBm (at 54 Mbps) 802.11n : +13 ± 2 dBM (at HT20, MCS7)
Receiving Sensitivity	802.11b : -93 dBm (at 11 Mbps, CCK) 802.11g : -85 dBm (at 54 Mbps, OFDM) 802.11n : -82 dBm (at HT20, MCS7)
Wireless Form	On-board PCB Antenna
IO Capability	UART, I2C, PWM, GPIO, 1 ADC
Electrical Characteristic	3.3 V Operated 15 mA output current per GPIO pin 12 - 200 mA working current Less than 200 uA standby current
Operating Temperature	-40 to +125 ºC
Serial Transmission	110 - 921600 bps, TCP Client 5
Wireless Network Type	STA / AP / STA + AP
Security Type	WEP / WPA-PSK / WPA2-PSK
Encryption Type	WEP64 / WEP128 / TKIP / AES
Firmware Upgrade	Local Serial Port, OTA Remote Upgrade
Network Protocol	IPv4, TCP / UDP / FTP / HTTP
User Configuration	AT + Order Set, Web Android / iOS, Smart Link APP



[NodeMCU V1.0의 핀 배치도]

2. NodeMCU로 할 수 있는 프로젝트

아두이노와 와이파이 모듈로 할 수 있었던 원격 제어, 원격 모니터링이 주로 NodeMCU로 할 수 있는 프로젝트입니다.



3D 프린터를 활용하여 현재 시간과 날씨 습도 등의 데이터를 웹을 통해 받은 후 디스플레이를

할 수도 있으며, 저렴한 DHT11 센서를 통해 원격으로 온습도 센싱을 한 후 디스플레이를 할 수도 있습니다.

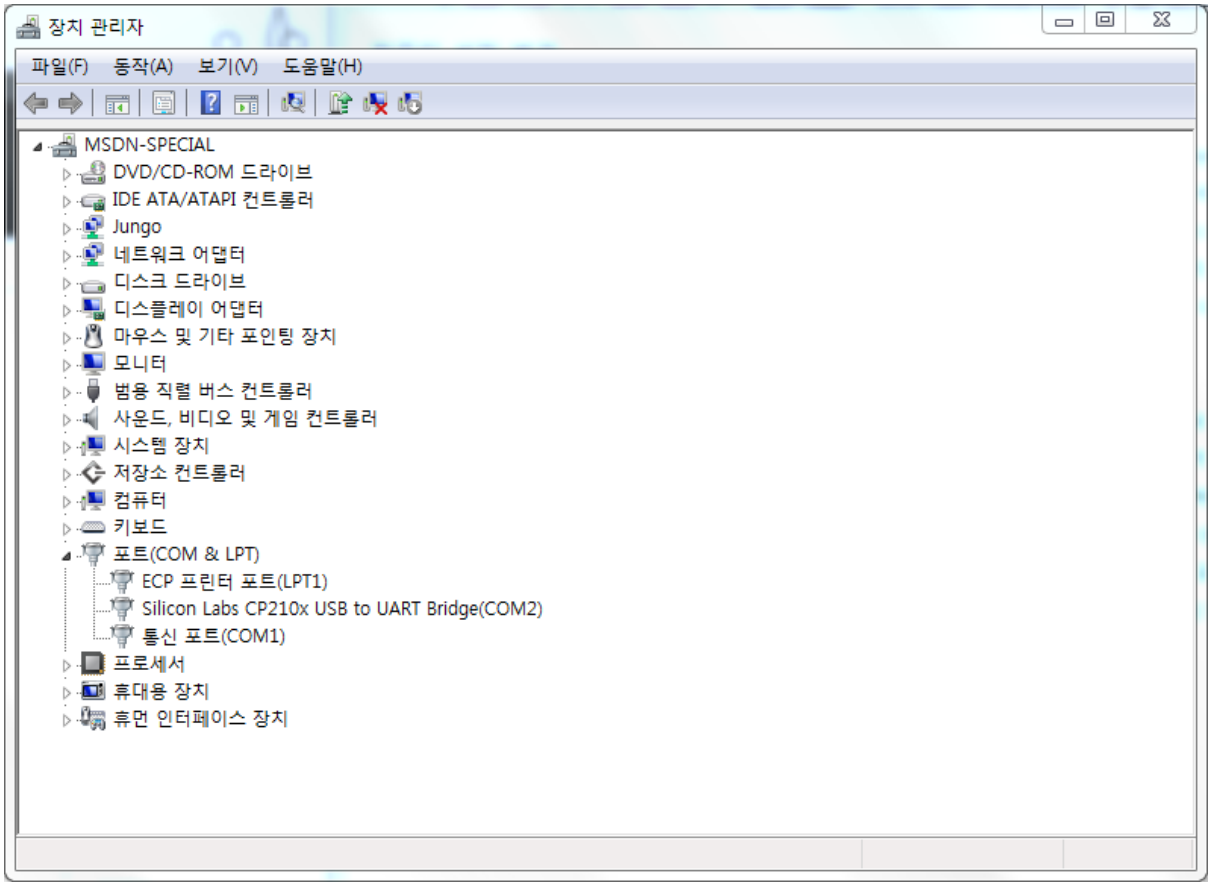
뿐만 아니라, 스마트플러그를 통한 전력 센싱과 전력 제어, 그리고 비닐하우스 내에서 광량 조절 및 온습도 조절 등 다양한 분야에서도 활용할 수 있습니다.

3. NodeMCU 개발환경 구축하기

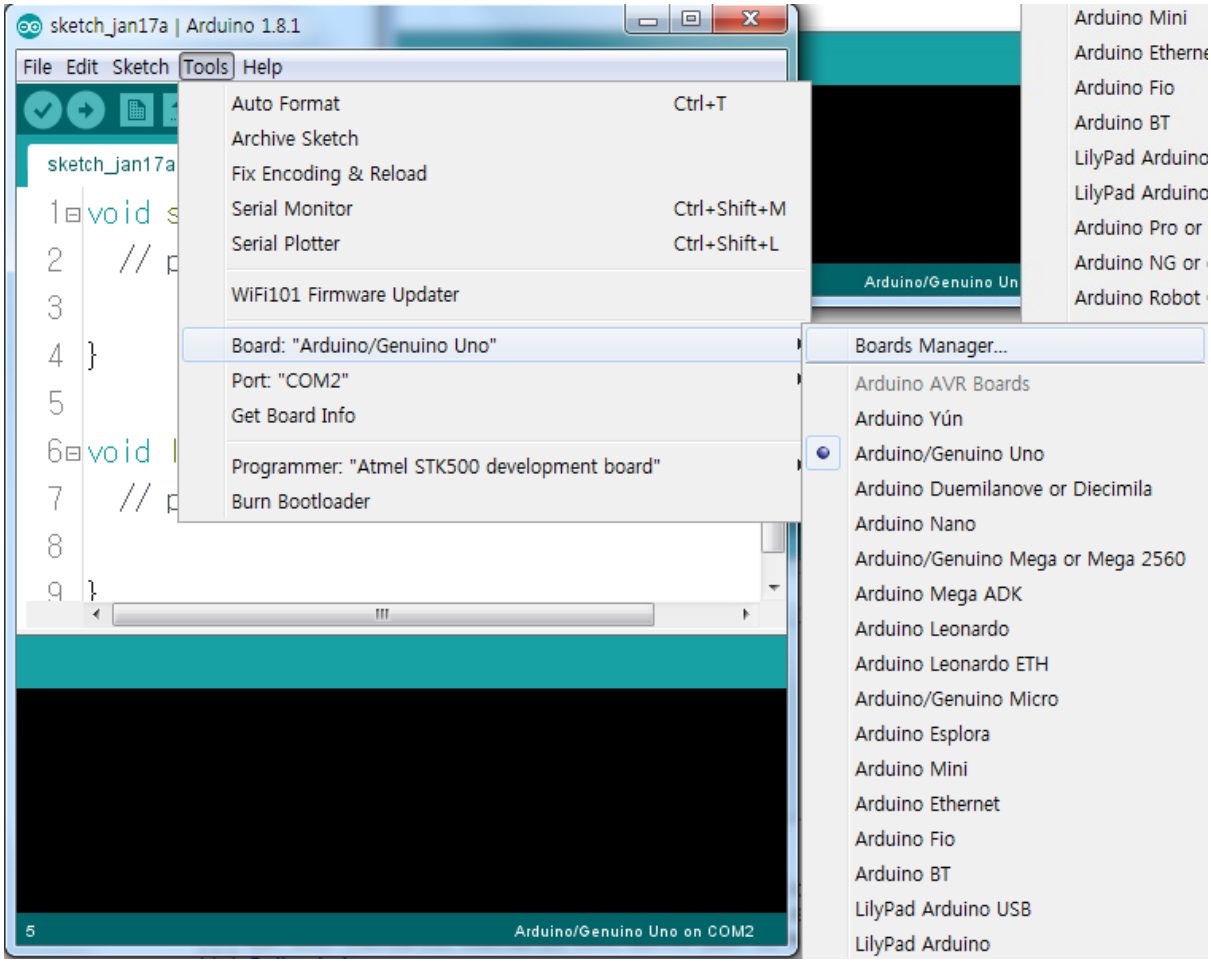
NodeMCU는 Lua라는 프로그램 언어를 통해 개발할 수 있습니다. 하지만, 많은 사람들이 아두이노 IDE를 통해 프로그래밍을 하면서 보다 많은 자료가 공유되고, 편리하게 사용되고 있습니다. 본 챕터에서는 아두이노 IDE를 통한 NodeMCU 개발환경을 구축하고, 기본 예제인 Blink 프로그램을 업로드 해보도록 하겠습니다.

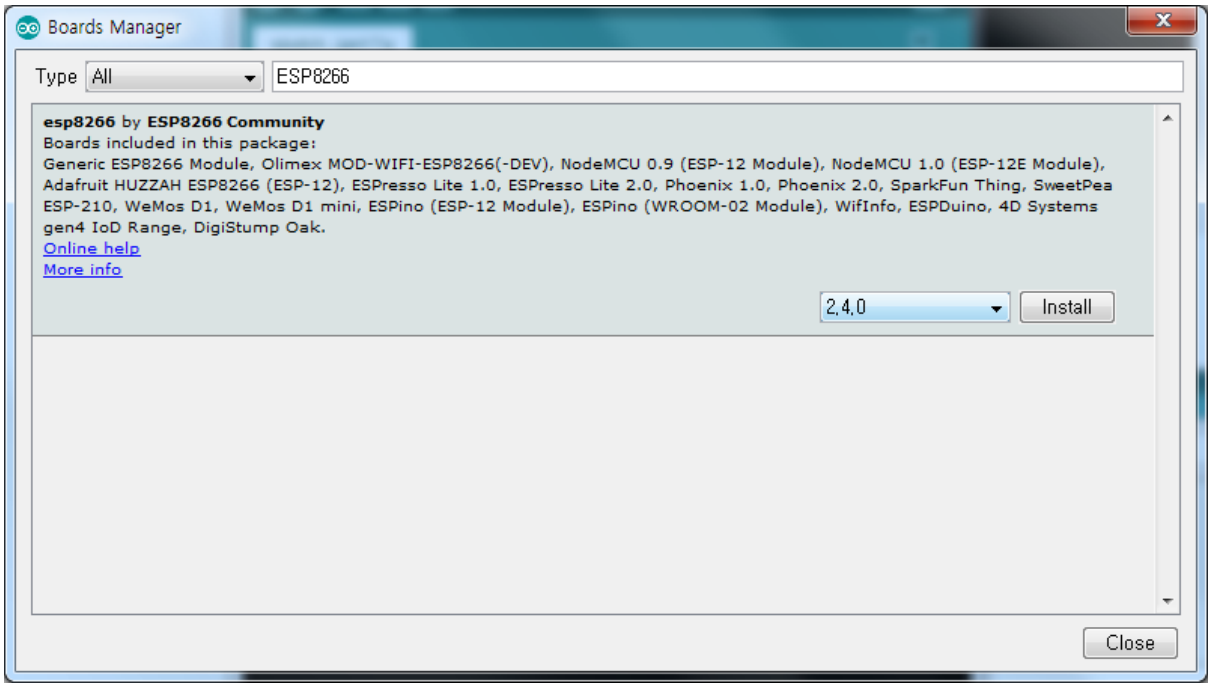
먼저, NodeMCU에 USB 마이크로케이블을 통해 PC에 연결합니다. 그러면 다음과 같이 CP2012 드라이버가 설치되는 것을 윈도우즈 오른쪽 하단에서 확인할 수 있습니다.

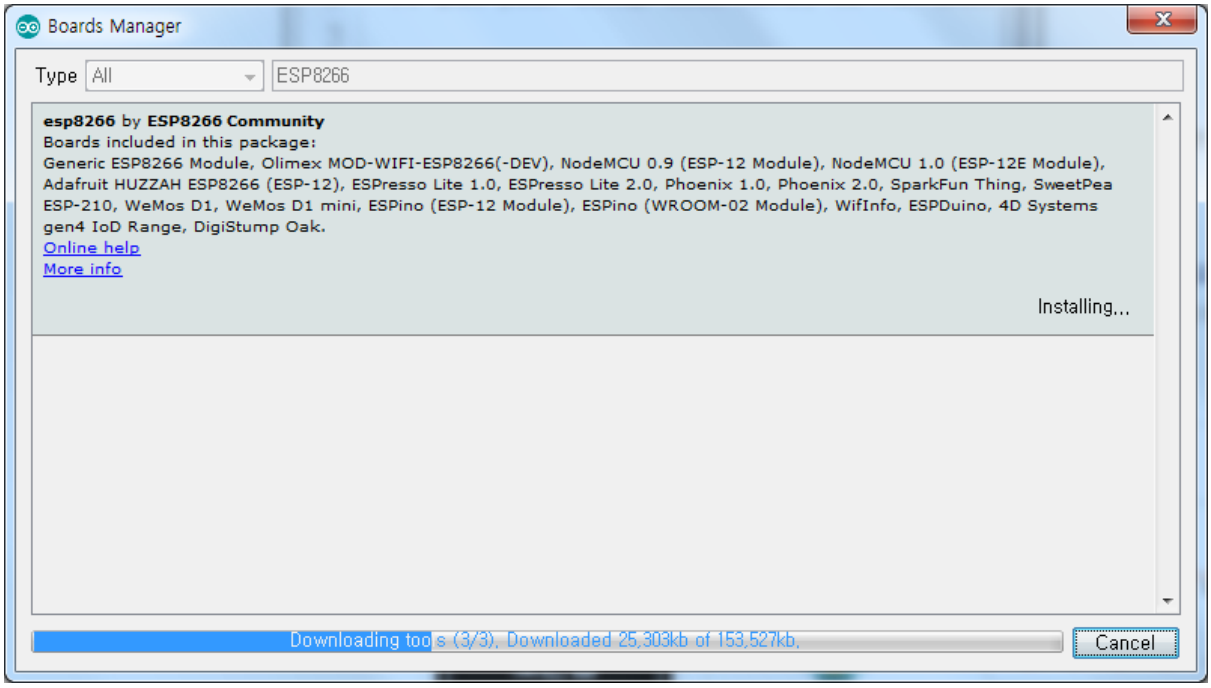
제대로 설치되었는지 보기 위해서는 PC의 장치관리자의 포트를 통해 재확인할 수 있습니다.

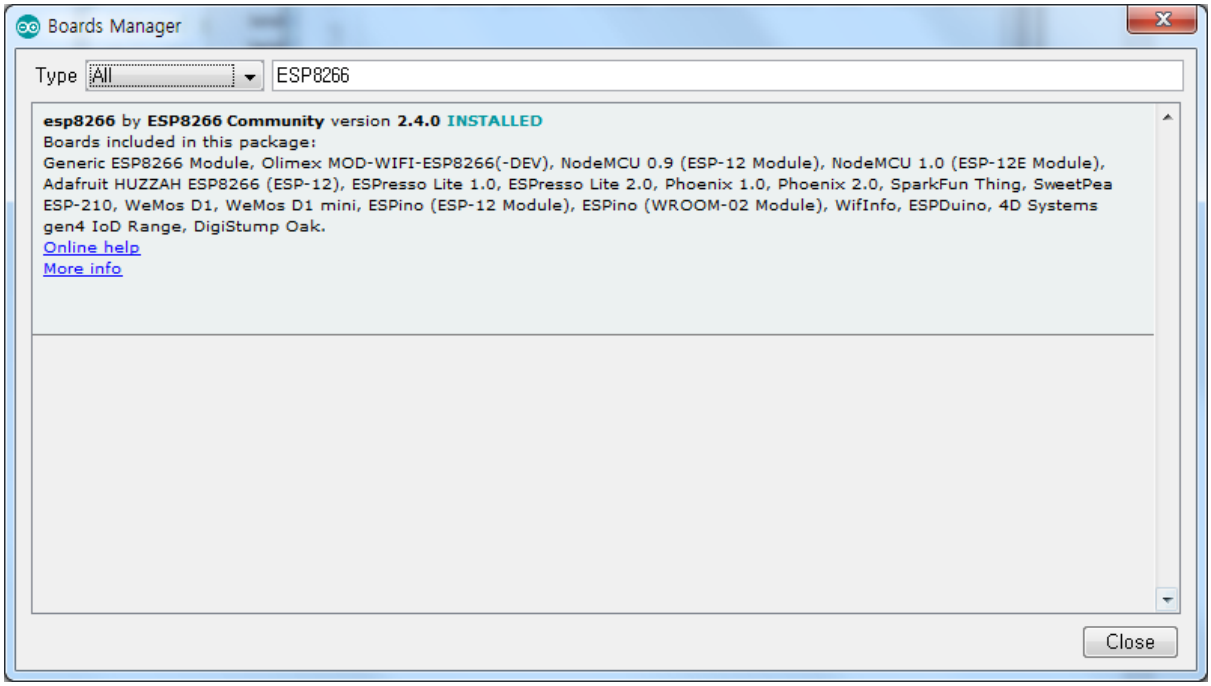


아두이노 IDE를 설치했다면, Tools (도구) – Boards Manager (보드매니저)를 통해 ESP8266 개발툴킷을 설치합니다.

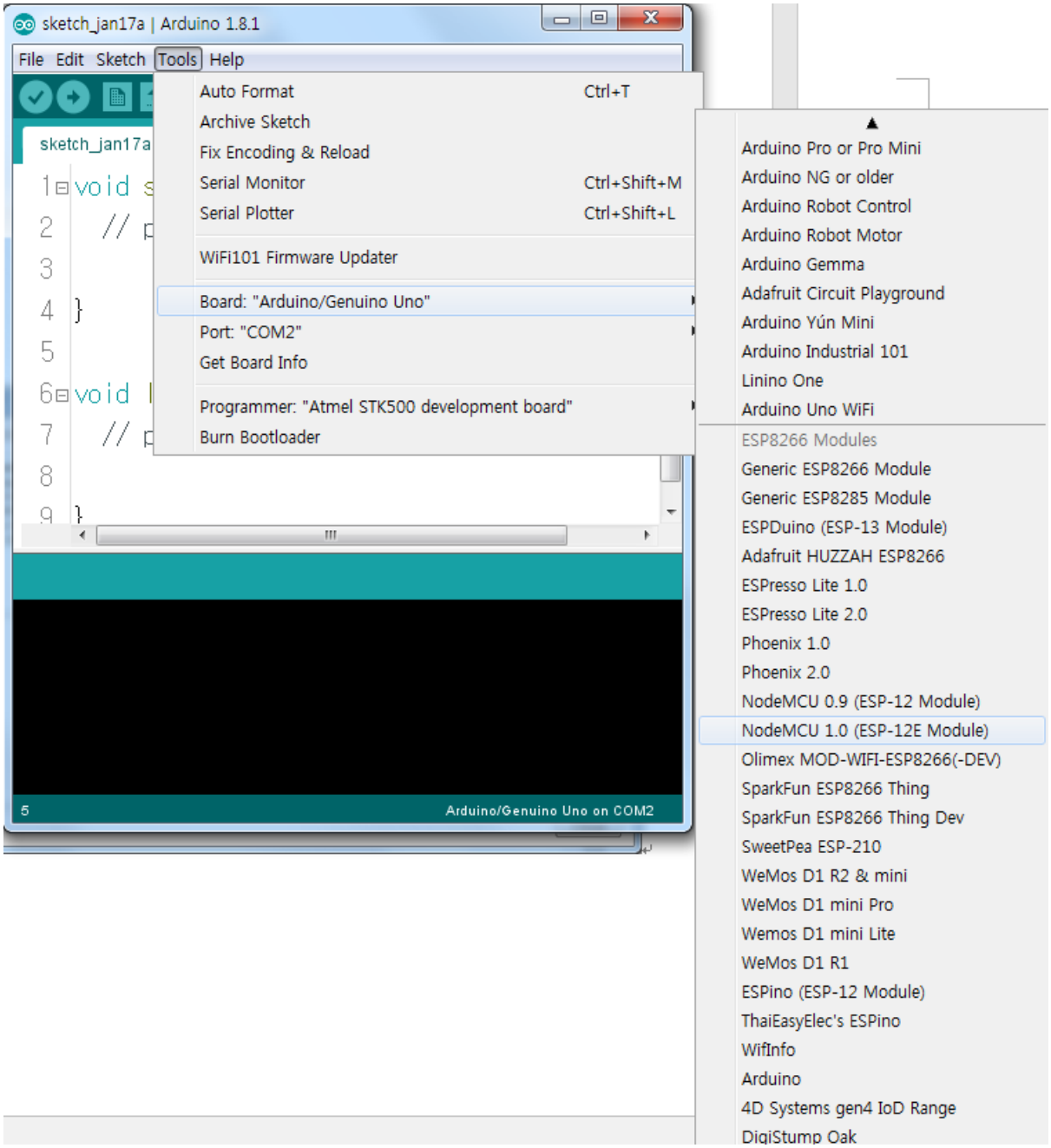

sketch_jan17a | Arduino 1.8.1
File Edit Sketch Tools Help
sketch_jan17a
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno"
Port: "COM2"
Get Board Info
Programmer: "Atmel STK500 development board"
Burn Bootloader
Boards Manager...
Arduino AVR Boards
Arduino Yún
Arduino/Genuino Uno
Arduino Duemilanove or Diecimila
Arduino Nano
Arduino/Genuino Mega or Mega 2560
Arduino Mega ADK
Arduino Leonardo
Arduino Leonardo ETH
Arduino/Genuino Micro
Arduino Esplora
Arduino Mini
Arduino Ethernet
Arduino Fio
Arduino BT
LilyPad Arduino USB
LilyPad Arduino
Arduino/Genuino Uno on COM2


Boards Manager
Type All
ESP8266
esp8266 by ESP8266 Community
Boards included in this package:
Generic ESP8266 Module, Olimex MOD-WIFI-ESP8266(-DEV), NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Adafruit HUZZAH ESP8266 (ESP-12), ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, SparkFun Thing, SweetPea ESP-210, WeMos D1, WeMos D1 mini, ESPino (ESP-12 Module), ESPino (WROOM-02 Module), WifInfo, ESPDuino, 4D Systems gen4 IoD Range, DigiStump Oak.
Online help
More info
2.4.0
Install
Close

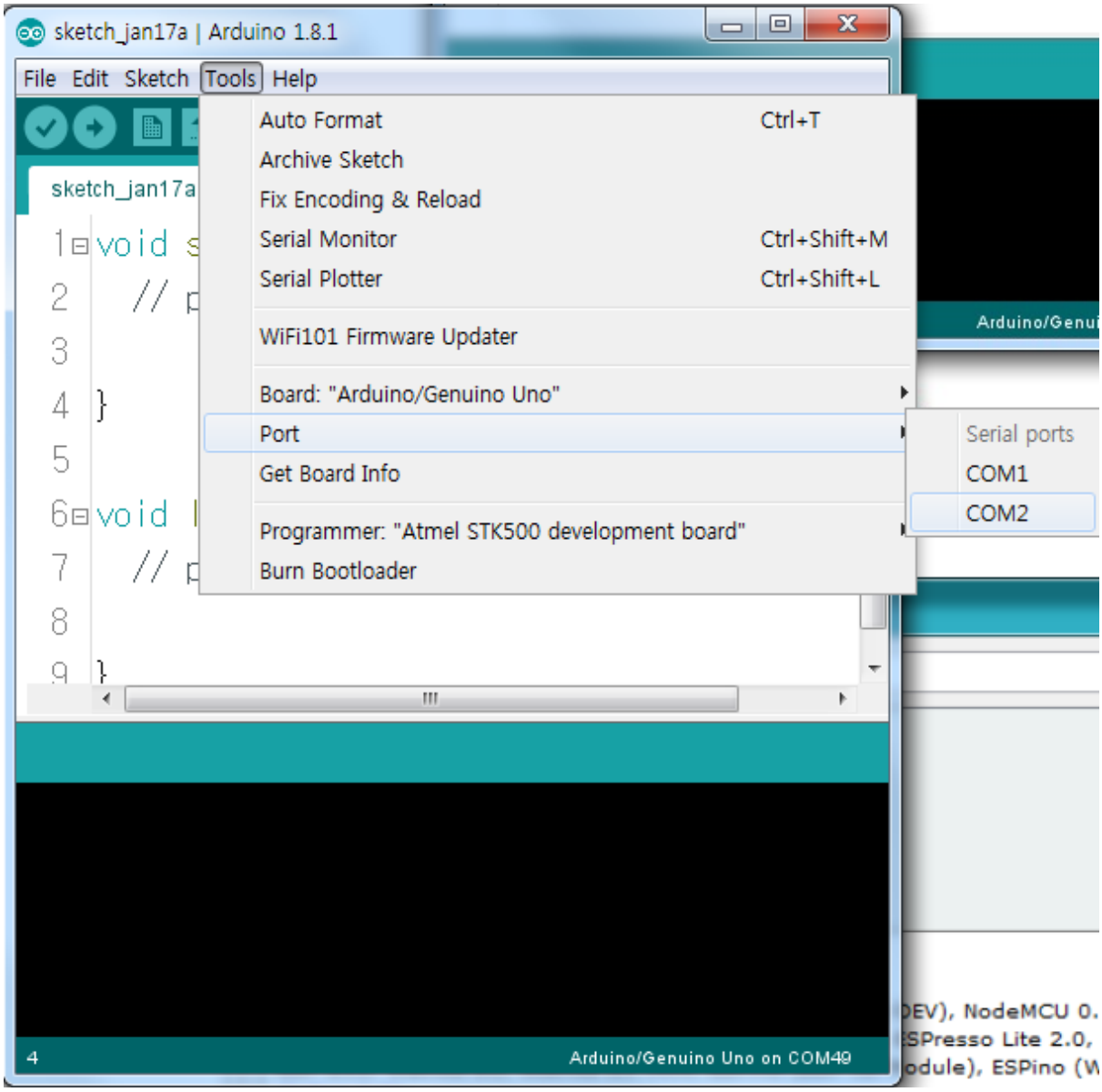


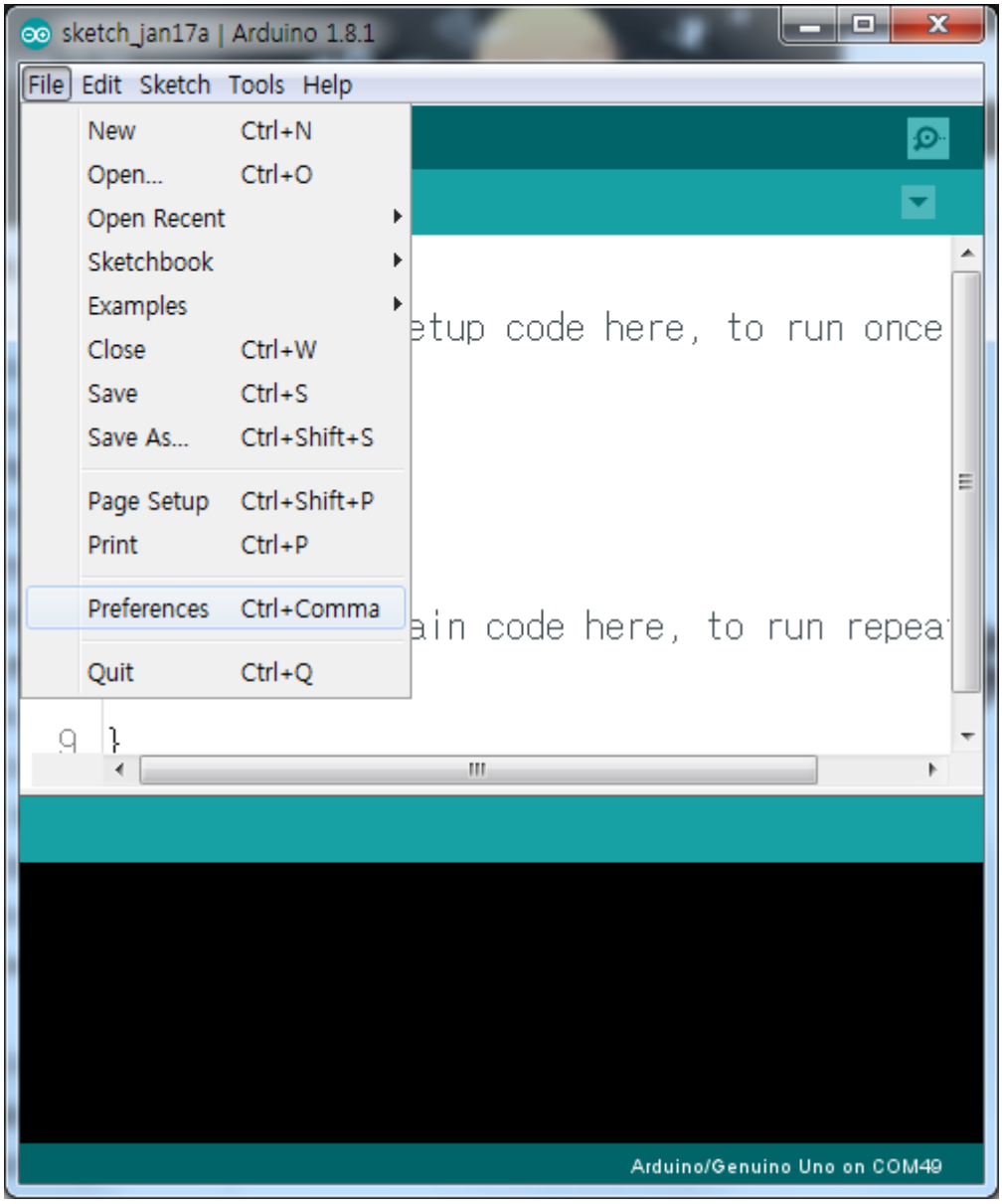


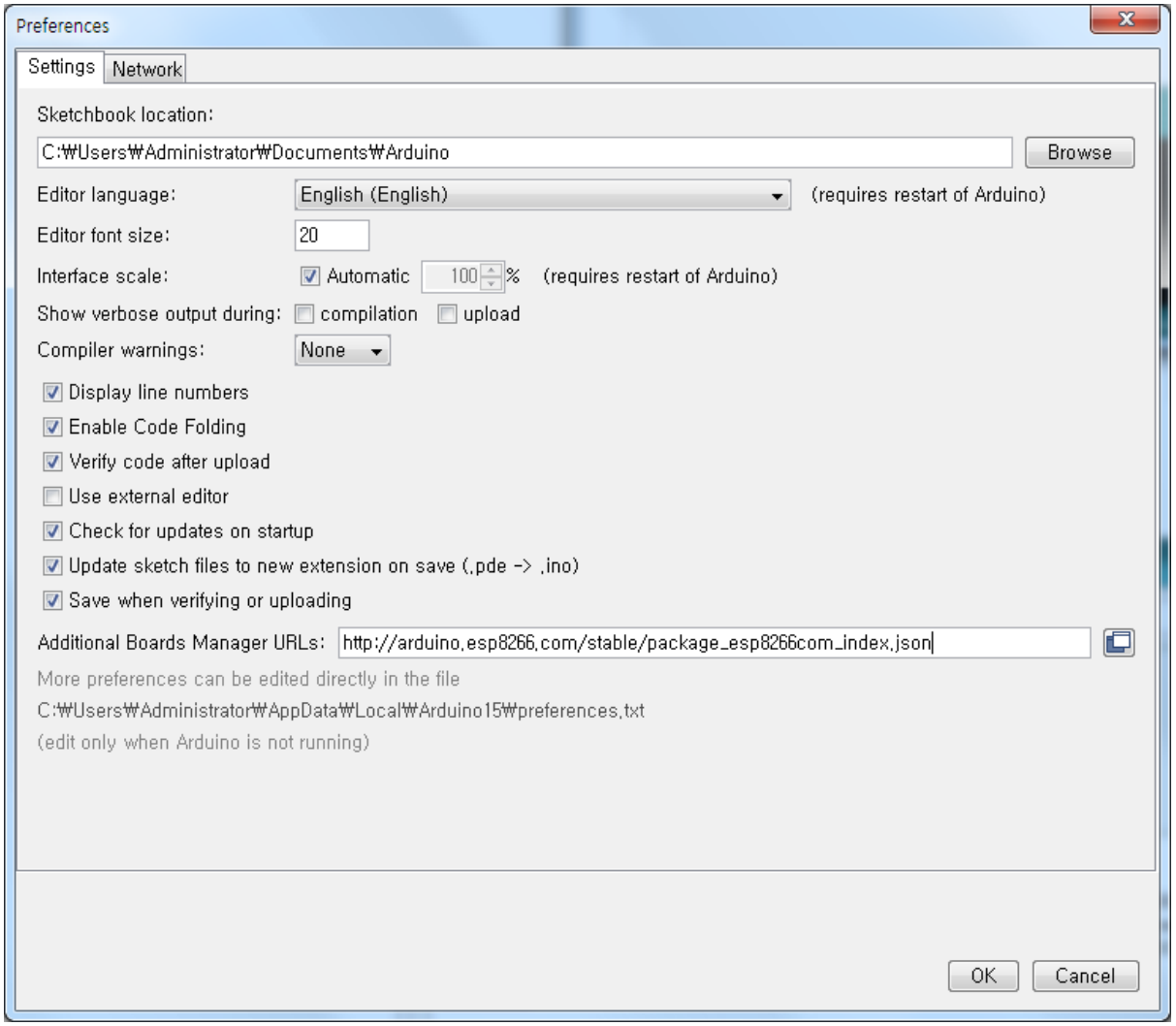
설치가 완료된 후, 아두이노 IDE의 Tools (도구) – Board (보드) 설정에서 "NodeMCU 1.0 (ESP-12E Module)"이라는 보드를 찾을 수 있습니다. 클릭을 해서 지정을 합니다.



그 다음에는 NodeMCU가 연결된 포트를 Tools (도구) – Port (포트)를 통해 설정합니다. 보드 선택할 때와 마찬가지로 클릭을 해서 지정을 하게 되는데, 여러개의 COM 포트가 있다면 연결된 NodeMCU를 PC에서 뺐다가 다시 연결하면서 변화가 있는 COM 포트 번호를 선택할 수 있습니다.


sketch_jan17a | Arduino 1.8.1
File Edit Sketch Tools Help
sketch_jan17a
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno"
Port
Get Board Info
Programmer: "Atmel STK500 development board"
Burn Bootloader
Serial ports
COM1
COM2
4
Arduino/Genuino Uno on COM49


sketch_jan17a | Arduino 1.8.1
File Edit Sketch Tools Help
New Ctrl+N
Open... Ctrl+O
Open Recent
Sketchbook
Examples
Close Ctrl+W
Save Ctrl+S
Save As... Ctrl+Shift+S
Page Setup Ctrl+Shift+P
Print Ctrl+P
Preferences Ctrl+Comma
Quit Ctrl+Q
etup code here, to run once
ain code here, to run repea
}
Arduino/Genuino Uno on COM49

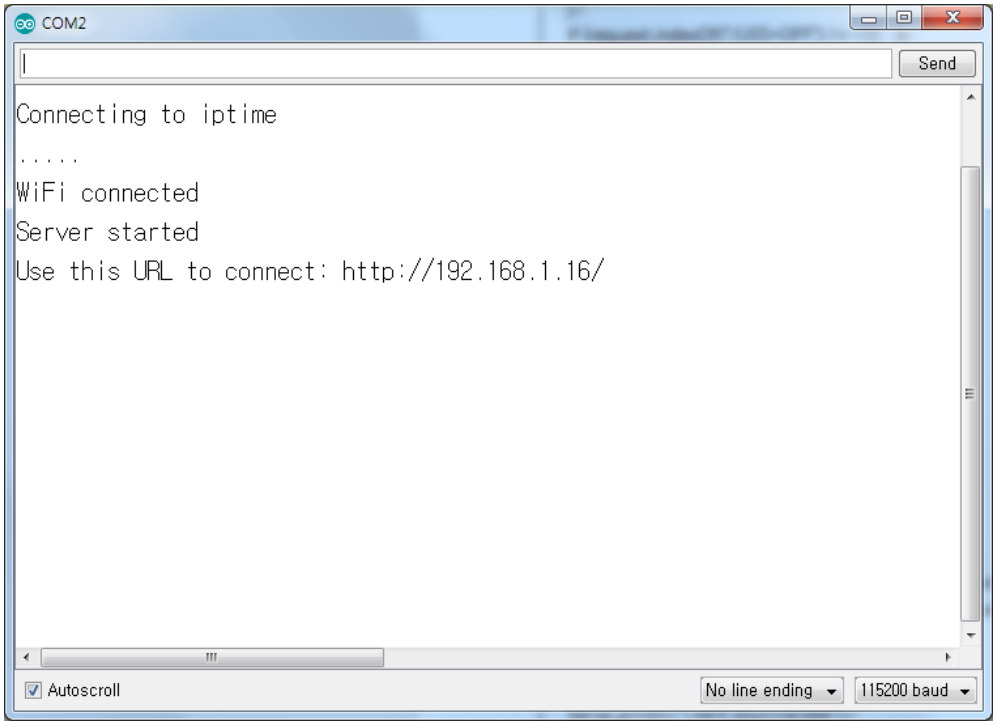

Preferences
Settings
Network
Sketchbook location:
C:₩Users₩Administrator₩Documents₩Arduino
Browse
Editor language:
English (English)
(requires restart of Arduino)
Editor font size:
20
Interface scale:
Automatic
100
%
(requires restart of Arduino)
Show verbose output during:
compilation
upload
Compiler warnings:
None
Display line numbers
Enable Code Folding
Verify code after upload
Use external editor
Check for updates on startup
Update sketch files to new extension on save (.pde -> .ino)
Save when verifying or uploading
Additional Boards Manager URLs:
http://arduino.esp8266.com/stable/package_esp8266com_index.json
More preferences can be edited directly in the file
C:₩Users₩Administrator₩AppData₩Local₩Arduino15₩preferences.txt
(edit only when Arduino is not running)
OK
Cancel

4. NodeMCU를 사용하여 “HELLO WORLD!” 웹에 출력하기

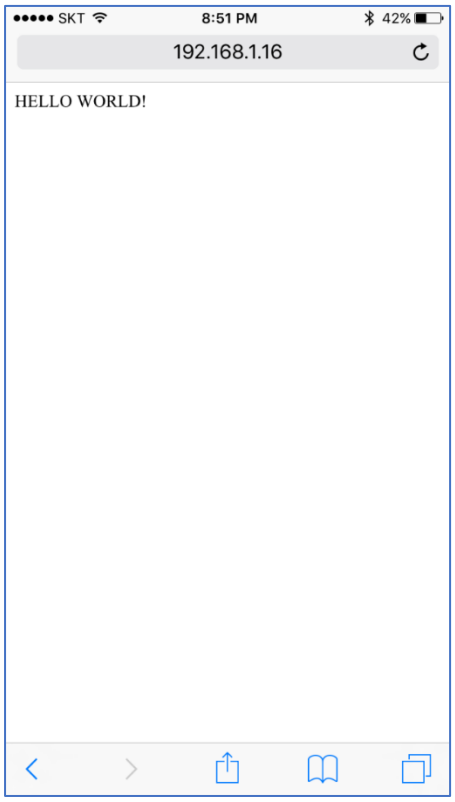
```
#include <ESP8266WiFi.h>

const char* ssid = "iptime";
const char* password = "";

WiFiServer server(80);

void setup() {
  Serial.begin(115200);
  delay(10);


  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");

  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");
```

```
}

void loop() {
  // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
    return;
  }

  // Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){
    delay(1);
  }

  // Read the first line of the request
  String request = client.readStringUntil('₩r');
  Serial.println(request);
  client.flush();

  // Return the response
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println(""); //  do not forget this one
  client.println("<!DOCTYPE HTML>");
  client.println("<html>");
  client.print("HELLO WORLD!");
  client.println("</html>");

  delay(1);
  Serial.println("Client disonnected");
  Serial.println("");

}
```

업로드가 완료된 후에, 오른쪽 상단의 시리얼 모니터링 버튼을 클릭하고 NodeMCU의 리셋버튼을 누르면 원격 접속하기 위한 URL을 확인할 수 있습니다.



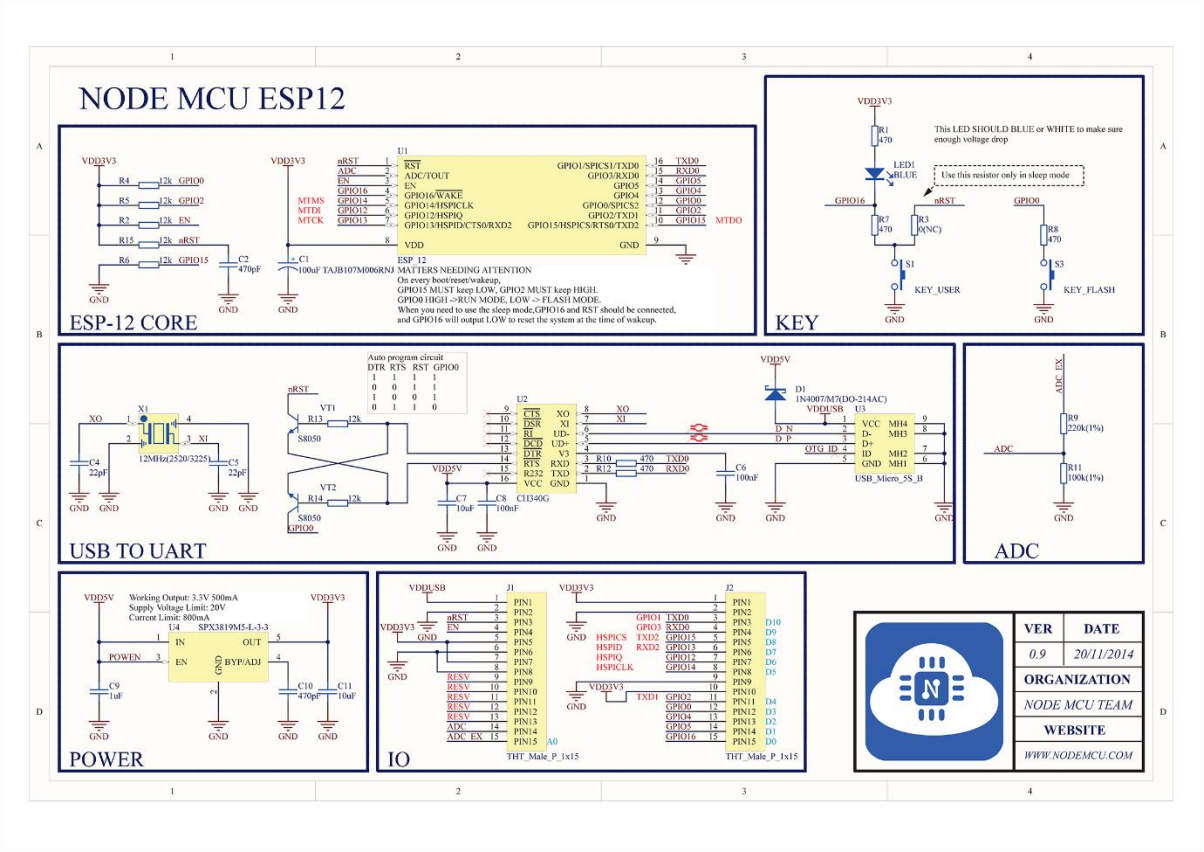
http://192.168.1.16을 스마트폰에서 접속하면 HELLO WORLD!라는 텍스트를 다음과 같이 확인할 수 있습니다.



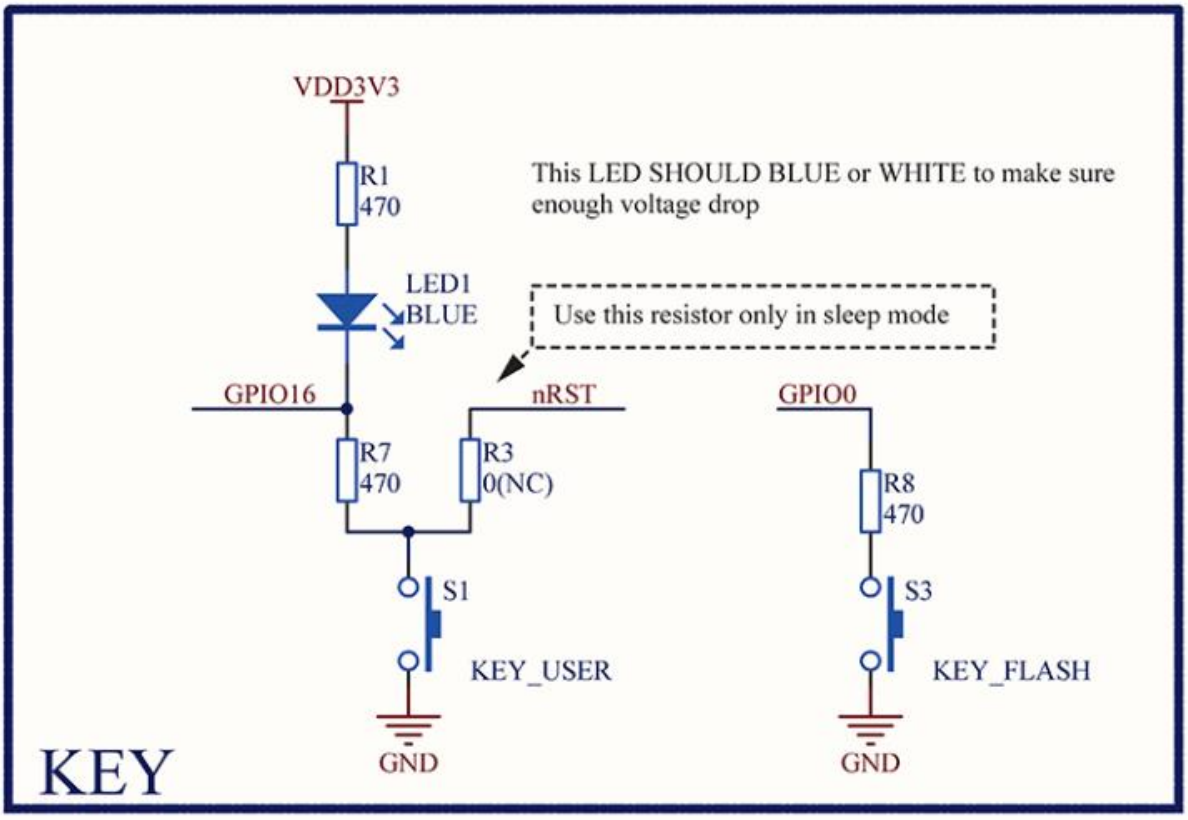
5. NodeMCU로 Blink 예제 실행하기

NodeMCU를 사용하여 Blink 예제를 사용해보면 다른 점을 발견할 수 있는데, 바로 내장된 LED이다.

digitalWrite(LED, HIGH);를 실행하면 LED가 켜져야한다고 생각하는데, NodeMCU에서는 이 부분이 반대로 되어 있다. 그 이유는 다음의 NodeMCU의 스케메틱을 통해 확인해볼 수 있다.



NodeMCU에서 제공하는 스케메틱을 보면, 오른쪽 상단의 블럭에서 내장 LED와 연결된 GPIO16번이 LED의 (-)극에 연결이 되어 있다. 확대해서 살펴보면 다음과 같다.



GPIO16이 LOW인 상태이면, LED는 3V3을 입력받고 불이 켜지게 되며, GPIO16이 상태가 HIGH가 되면, LED의 양단의 전위차가 0이 되어 불이 꺼지게 된다. 일반적인 아두이노의 13번 내장 LED와는 방향이 반대이기 때문에 같은 예제를 사용하더라도 반대로 LED가 작동하는 것을 확인할 수 있다.

때문에, NodeMCU를 사용하여 내장된 LED를 제어한다면 다음과 같은 코드를 사용할 수 있다.

```
int LED_pin = 16;
int turn_on = 0;
int turn_off = 1;

void setup() {
  // put your setup code here, to run once:
  pinMode(LED_pin, OUTPUT);
  digitalWrite(LED_pin, turn_off);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(LED_pin, turn_on);
  delay(1000);
```

```
    digitalWrite(LED_pin, turn_off);
    delay(1000);
}
```

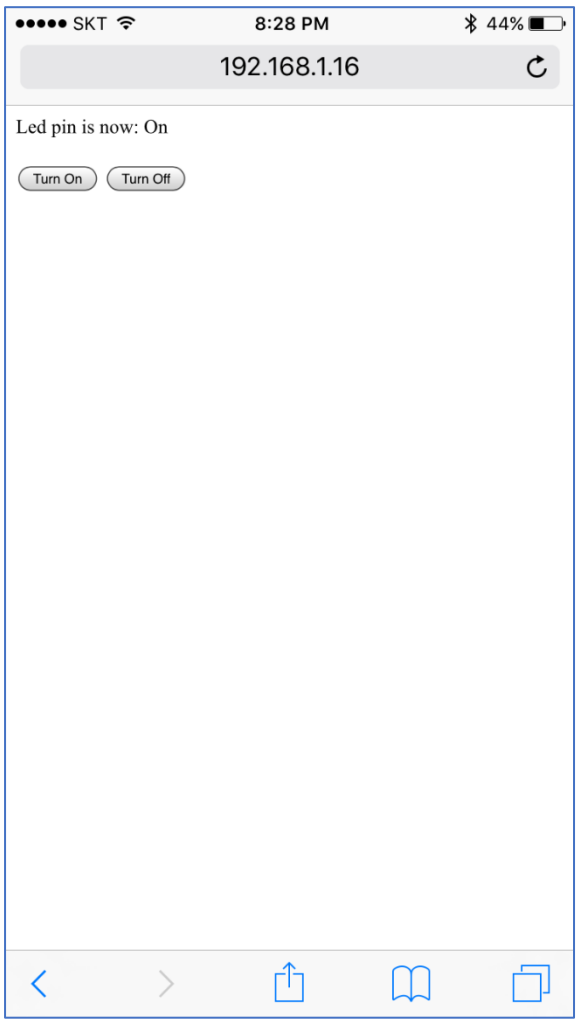
6. 원격으로 LED 제어하기

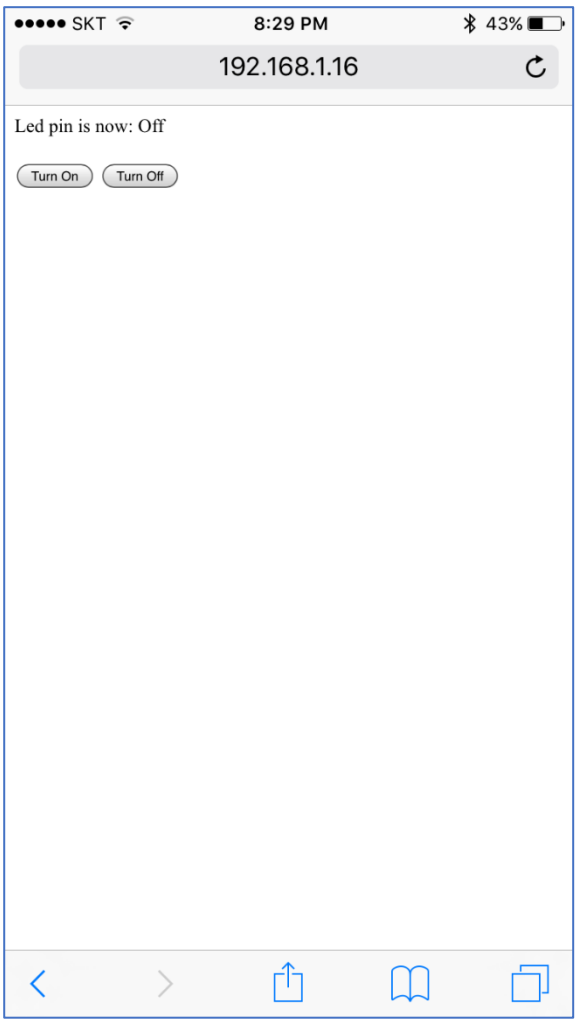
```
#include <ESP8266WiFi.h>

const char* ssid = "iptime";
const char* password = "";

WiFiServer server(80);

// 변수 지정
int LED_pin = 16;
int turn_on = 0;
int turn_off = 1;

void setup() {
  Serial.begin(115200);
  delay(10);

  pinMode(LED_pin, OUTPUT);
  digitalWrite(LED_pin, turn_off);

  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
```

```
  server.begin();
  Serial.println("Server started");

  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");

}

void loop() {
  // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
    return;
  }

  // Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){
    delay(1);
  }

  // Read the first line of the request
  String request = client.readStringUntil('₩r');
  Serial.println(request);
  client.flush();

  // Match the request

  int value = turn_off;
  if (request.indexOf("/LED=ON") != -1)  {
    digitalWrite(LED_pin, turn_on);
    value = turn_on;
  }
  if (request.indexOf("/LED=OFF") != -1)  {
    digitalWrite(LED_pin, turn_off);
    value = turn_off;
```

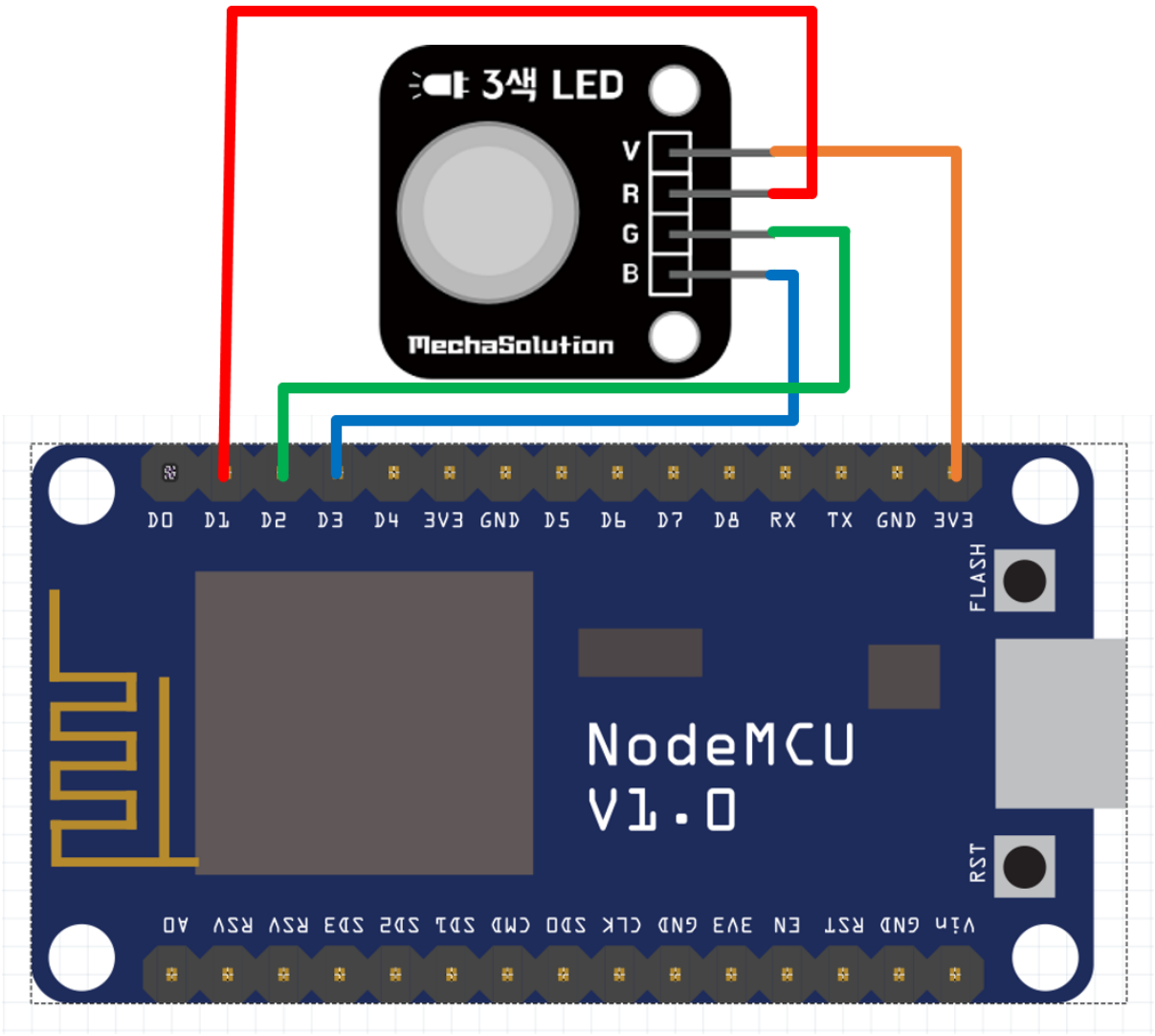
```
  }


  // Return the response
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println(""); //  do not forget this one
  client.println("<!DOCTYPE HTML>");
  client.println("<html>");

  client.print("Led pin is now: ");

  if(value == turn_on) {
    client.print("On");
  } else {
    client.print("Off");
  }
  client.println("<br><br>");
  client.println("<a href=₩"/LED=ON₩"₩"><button>Turn On </button></a>");
  client.println("<a href=₩"/LED=OFF₩"₩"><button>Turn Off </button></a><br />");
  client.println("</html>");

  delay(1);
  Serial.println("Client disonnected");
  Serial.println("");

}
```

스마트폰에서 접속을 한 후에, 버튼을 클릭하게 되면 다음과 같이 원격으로 LED가 제어되는 것을 확인할 수 있습니다.


SKT
8:28 PM
44%
192.168.1.16
Led pin is now: On
Turn On
Turn Off


SKT
8:29 PM
43%
192.168.1.16
Led pin is now: Off
Turn On
Turn Off

7. 원격으로 RGB LED 제어하기



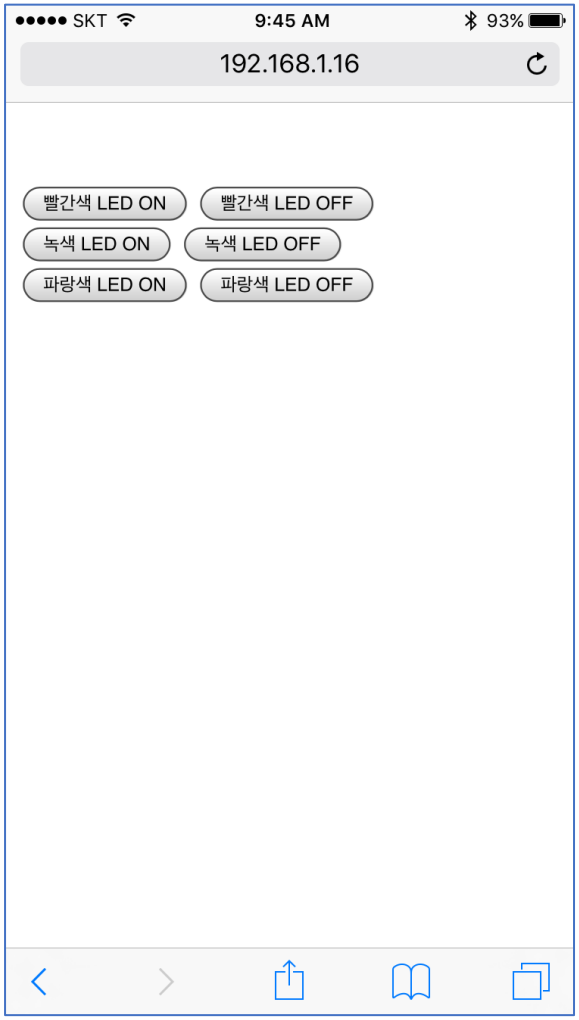

```
#include <ESP8266WiFi.h>

const char* ssid = "iptime";
const char* password = "";

WiFiServer server(80);

// 변수 지정
int LED_pin_R = 5; // D1
int LED_pin_G = 4; // D2
```

```
int LED_pin_B = 0; // D3

// Anode
int turn_on = 0;
int turn_off = 1;

// Cathode
// int turn_on = 1;
// int turn_off = 0;

void setup() {
  Serial.begin(115200);
  delay(10);

  pinMode(LED_pin_R, OUTPUT);
  pinMode(LED_pin_G, OUTPUT);
  pinMode(LED_pin_B, OUTPUT);
  digitalWrite(LED_pin_R, turn_off);
  digitalWrite(LED_pin_G, turn_off);
  digitalWrite(LED_pin_B, turn_off);

  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");
```

```
  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");

}

void loop() {
  // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
    return;
  }

  // Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){
    delay(1);
  }

  // Read the first line of the request
  String request = client.readStringUntil('₩r');
  Serial.println(request);
  client.flush();

  // Match the request

  int value_R = turn_off;
  int value_G = turn_off;
  int value_B = turn_off;

  if (request.indexOf("/LED_R=ON") != -1)  {
    digitalWrite(LED_pin_R, turn_on);
    value_R = turn_on;
  }
  if (request.indexOf("/LED_R=OFF") != -1)  {
    digitalWrite(LED_pin_R, turn_off);
```

```
    value_R = turn_off;
  }

  if (request.indexOf("/LED_G=ON") != -1)  {
    digitalWrite(LED_pin_G, turn_on);
    value_G = turn_on;
  }
  if (request.indexOf("/LED_G=OFF") != -1)  {
    digitalWrite(LED_pin_G, turn_off);
    value_G = turn_off;
  }

  if (request.indexOf("/LED_B=ON") != -1)  {
    digitalWrite(LED_pin_B, turn_on);
    value_B = turn_on;
  }
  if (request.indexOf("/LED_B=OFF") != -1)  {
    digitalWrite(LED_pin_B, turn_off);
    value_B = turn_off;
  }

  // Return the response
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println(""); //  do not forget this one
  client.println("<!DOCTYPE HTML>");
  client.println("<html>");

  client.println("<meta http-equiv='Content-Type' content='text/html' charset='utf-8'/>");

  client.println("<br><br>");
  client.println("<a href=₩"/LED_R=ON₩"₩"><button> 빨간색 LED ON </button></a>");
  client.println("<a href=₩"/LED_R=OFF₩"₩"><button> 빨간색 LED OFF
</button></a><br/>");
  client.println("<a href=₩"/LED_G=ON₩"₩"><button> 녹색 LED ON </button></a>");
  client.println("<a href=₩"/LED_G=OFF₩"₩"><button> 녹색 LED OFF </button></a><br/>");
  client.println("<a href=₩"/LED_B=ON₩"₩"><button> 파랑색 LED ON </button></a>");
  client.println("<a href=₩"/LED_B=OFF₩"₩"><button> 파랑색 LED OFF
</button></a><br/>");
```

```
    client.println("</html>");

    delay(1);
    Serial.println("Client disonnected");
    Serial.println("");

}
```



8. 원격으로 온습도 모니터링하기

HELLO WORLD! 예제에서는 NodeMCU 웹서버에 접속한 브라우저에 HTML로 텍스트를 표현해 보았습니다. 이번에는 텍스트 뿐만 아니라 온도와 습도를 센서를 사용하여 모니터링하고, 이를 웹서버의 브라우저에 표현해보도록 하겠습니다. 그리고, 스마트폰을 통해 원격으로 접속하여 확인하는 프로그램을 작성해보겠습니다. 사용할 온습도 센서는 DHT11 기반의 온습도 센서로 다음과 같습니다.

사양 (Specification)

작동 전압: DC 3.3V ~ 5V

온도 범위: 0 ~ 50℃ / 정밀도 ± 2℃

습도 범위: 0~90% RH / 정밀도 ± 5%

디지털 인터페이스

크기: 30 x 21 mm

먼저, DHT11 센서를 사용하기 위해서 다음의 링크를 참고해볼 수 있습니다.

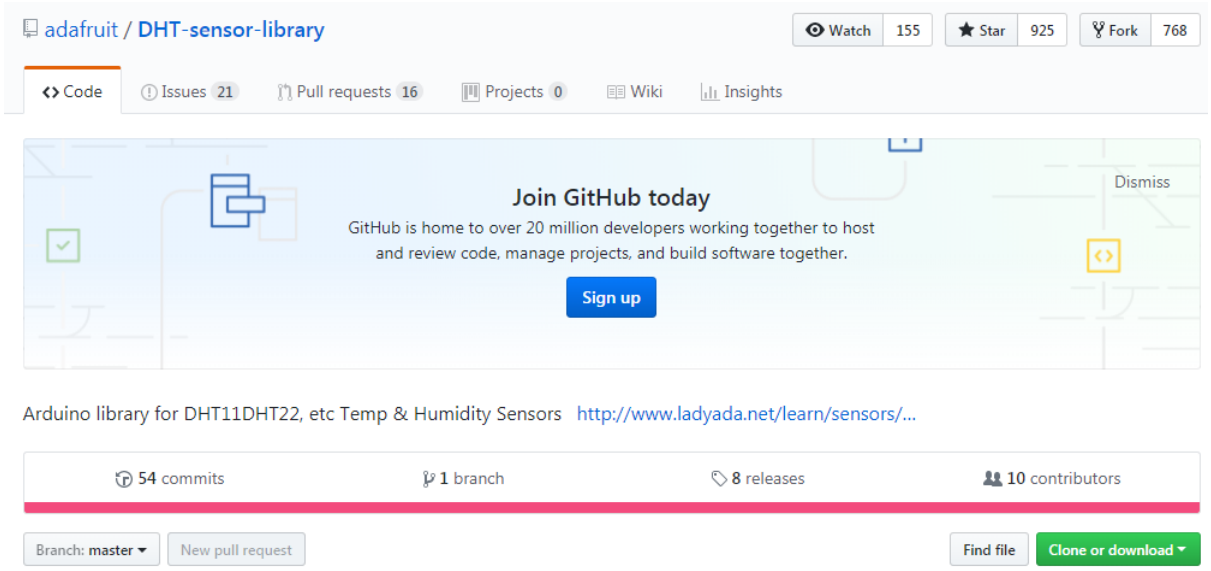
https://blog.naver.com/roboholic84/221186233842

상단의 링크에 라이브러리에 대한 사용법 및 링크가 있지만, 직접 라이브러리 압축파일은 다음의 링크에서 다운로드 받습니다.

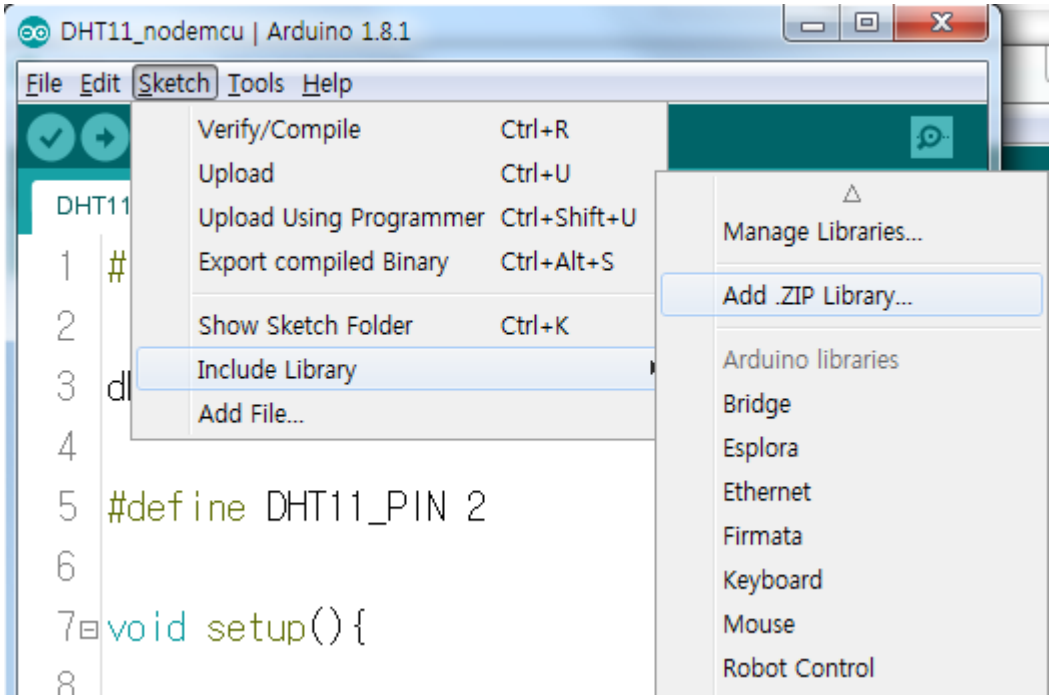
https://github.com/adafruit/Adafruit_Sensor

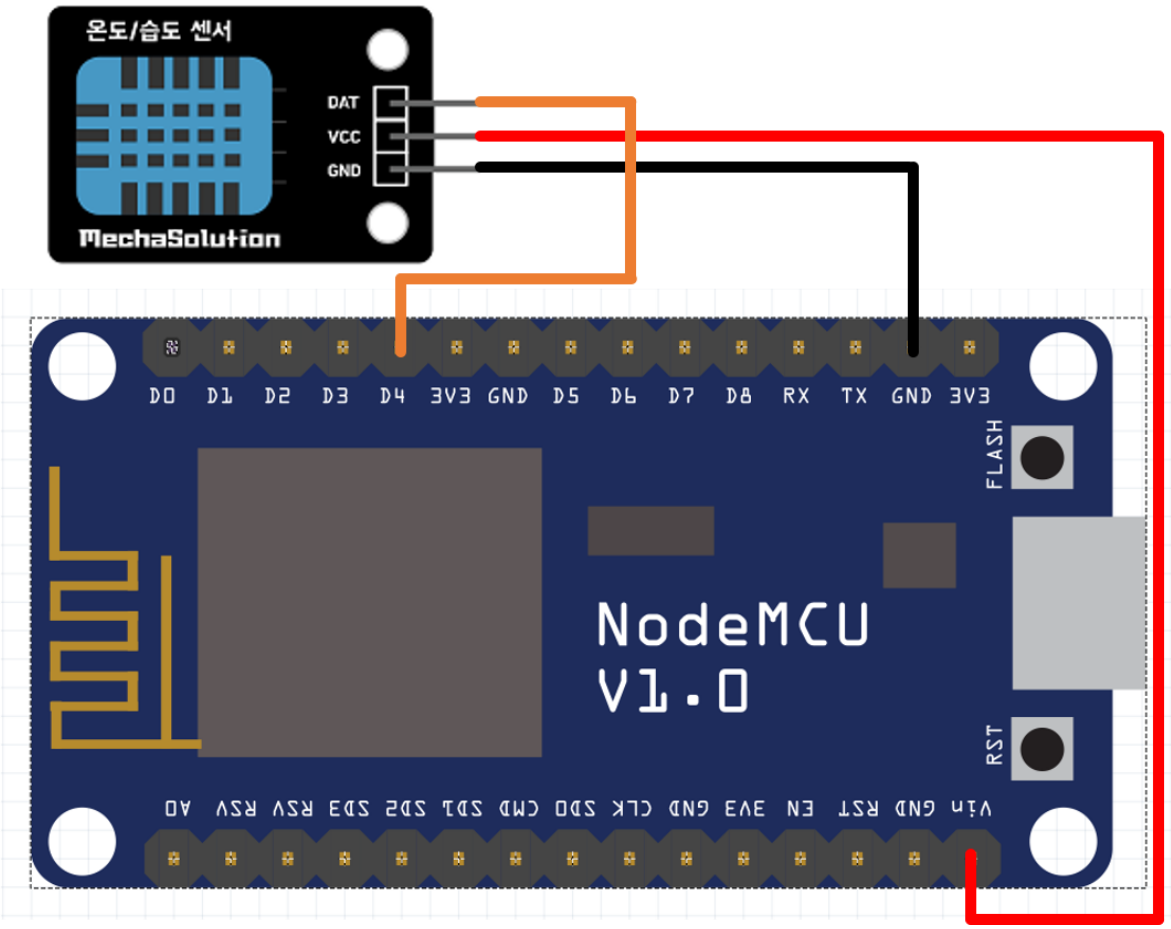
https://github.com/adafruit/DHT-sensor-library

그리고, 창의 오른쪽에 녹색버튼인 "Clone or download"을 클릭해서 압축파일을 받습니다.



두 개의 압축파일을 받은 후에, 아두이노 IDE의 Sketch (스케치) – Include Library – Add .ZIP Library...를 통해 추가합니다.





먼저, 원격이 아닌 온습도만 모니터링하는 프로그램은 다음과 같습니다.

```
#include "DHT.h"

#define DHTPIN 2        // DHT11이 연결된 핀

#define DHTTYPE DHT11     // DHT 11, DHT시리즈중 11을 선택합니다.

DHT dht(DHTPIN, DHTTYPE);

void setup() {
   Serial.begin(9600);
   Serial.println("DHTxx test!");
```

```
  dht.begin();
}

void loop() {
  delay(2000);

  float h = dht.readHumidity();// 습도를 측정합니다.
  float t = dht.readTemperature();// 온도를 측정합니다.
  float f = dht.readTemperature(true);// 화씨 온도를 측정합니다.

  // 값 읽기에 오류가 있으면 오류를 출력합니다.
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  // 보정된 화씨 값을 가져옵니다.
  float hif = dht.computeHeatIndex(f, h);
  // 보정된 섭씨 값을 가져옵니다.
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print("Humidity: ");
  Serial.print(h);
  Serial.print(" %₩t");
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.print(" *C ");
  Serial.print(f);
  Serial.print(" *F₩t");
  Serial.print("Heat index: ");
  Serial.print(hic);
  Serial.print(" *C ");
  Serial.print(hif);
  Serial.println(" *F");
}
```

시리얼 모니터링을 통해, 온습도가 잘 출력되는 것을 확인한 후 원격으로 데이터를 모니터링하기 위해 다음의 코드를 업로드 해봅니다.

```
#include <ESP8266WiFi.h>
const char* ssid = "iptime";
const char* password = "";

#include "DHT.h"
#define DHTPIN 2        // DHT11이 연결된 핀
#define DHTTYPE DHT11    // DHT 11, DHT시리즈중 11을 선택합니다.
DHT dht(DHTPIN, DHTTYPE);



WiFiServer server(80);

void setup() {
  Serial.begin(115200);
  delay(10);
  Serial.println("DHTxx test!");
  dht.begin();

  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");

  // Print the IP address
```

```
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");
}

void loop() {
    // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
    return;
  }

  // Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){
    delay(1);
  }

  // Read the first line of the request
  String request = client.readStringUntil('₩r');
  Serial.println(request);
  client.flush();



  delay(2000);

  float h = dht.readHumidity();// 습도를 측정합니다.
  float t = dht.readTemperature();// 온도를 측정합니다.
  float f = dht.readTemperature(true);// 화씨 온도를 측정합니다.

  // 값 읽기에 오류가 있으면 오류를 출력합니다.
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  // 보정된 화씨 값을 가져옵니다.
  float hif = dht.computeHeatIndex(f, h);
```

```
  // 보정된 섭씨 값을 가져옵니다.
  float hic = dht.computeHeatIndex(t, h, false);



  // Return the response
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println(""); //   do not forget this one
  client.println("<!DOCTYPE HTML>");
  client.println("<html>");
  client.print("Humidity: ");
  client.print(h);
  client.print(" %₩t");
  client.print("Temperature: ");
  client.print(t);
  client.print(" *C ");
  client.print(f);
  client.print(" *F₩t");
  client.print("Heat index: ");
  client.print(hic);
  client.print(" *C ");
  client.print(hif);
  client.println(" *F");
  client.println("</html>");

  delay(1);
  Serial.println("Client disonnected");
  Serial.println("");

}
```

시리얼 모니터링을 위해 브라우저에서 http://192.168.1.16을 접속해봅니다.


●●●●● SKT
9:40 PM
39%
192.168.1.16
Humidity: 21.00 % Temperature: 25.00 *C 77.00 *F Heat index: 24.10 *C 75.39 *F

9. 원격으로 화분의 수분량 모니터링하기

사양 (Specification)

토양 수분센서

아날로그 인터페이스

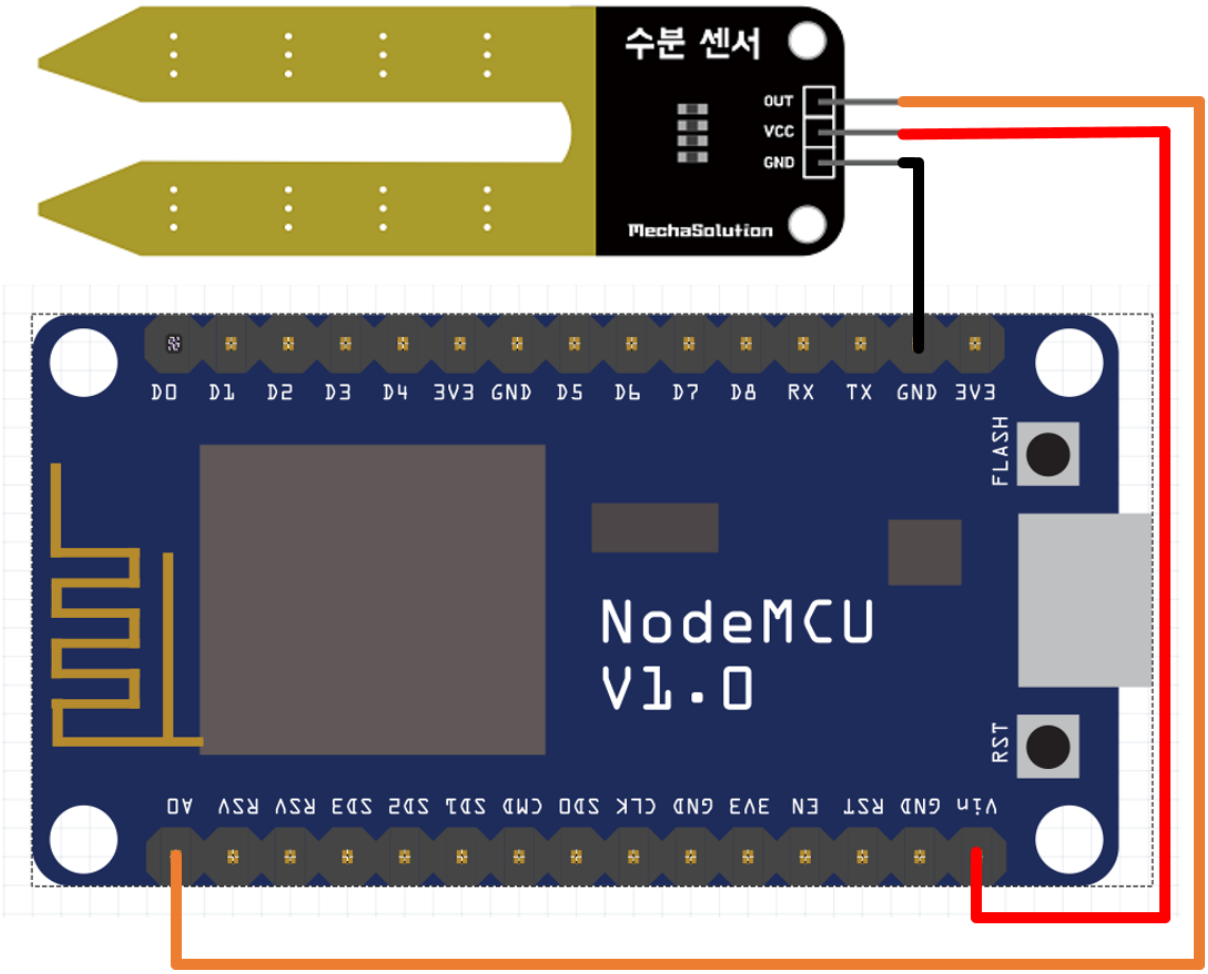
센싱부 표면 금도금

입력 전압: 3.3~5V

출력 전압: 0~3.6V

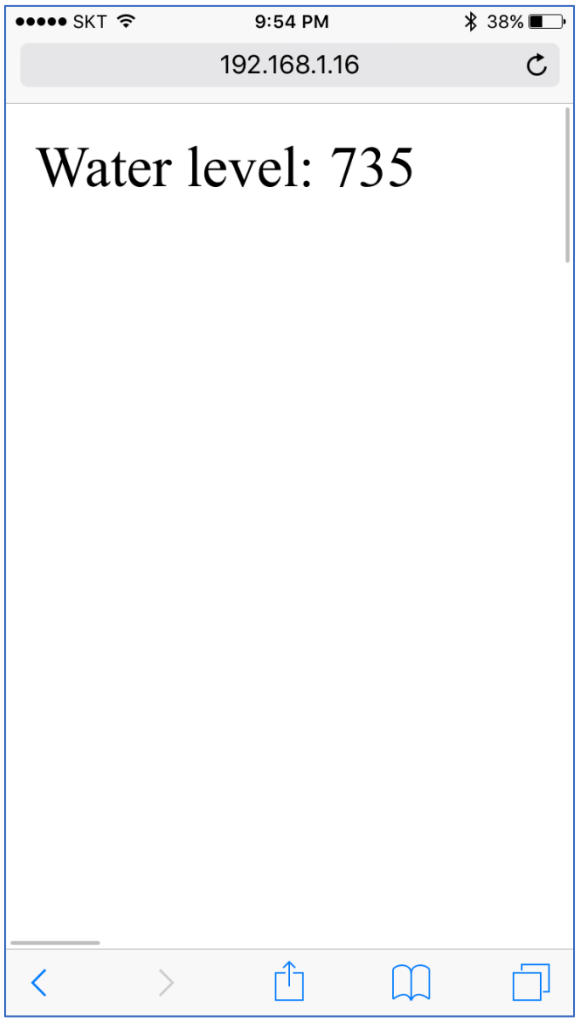
전체 크기: 60 x 20mm

센싱부 크기: 40 x 20mm



```
#include <ESP8266WiFi.h>
const char* ssid = "iptime";
const char* password = "";

WiFiServer server(80);

void setup() {
  Serial.begin(115200);
  delay(10);

  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");

  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");
}

void loop() {
    // Check if a client has connected
  WiFiClient client = server.available();
```

```
  if (!client) {
    return;
  }

  // Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){
    delay(1);
  }

  // Read the first line of the request
  String request = client.readStringUntil('₩r');
  Serial.println(request);
  client.flush();

  delay(2000);

  int waterlevel = analogRead(0);

  // Return the response
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println(""); //  do not forget this one
  client.println("<!DOCTYPE HTML>");
  client.println("<html>");
  client.print("Water level: ");
  client.print(waterlevel);
  client.println("</html>");

  delay(1);
  Serial.println("Client disonnected");
  Serial.println("");

}
```



동일한 방법으로 아날로그 센서를 사용하는 조도센서, 사운드센서, 수위센서, 가스센서 등에 사용할 수 있습니다.

사양 (Specification)

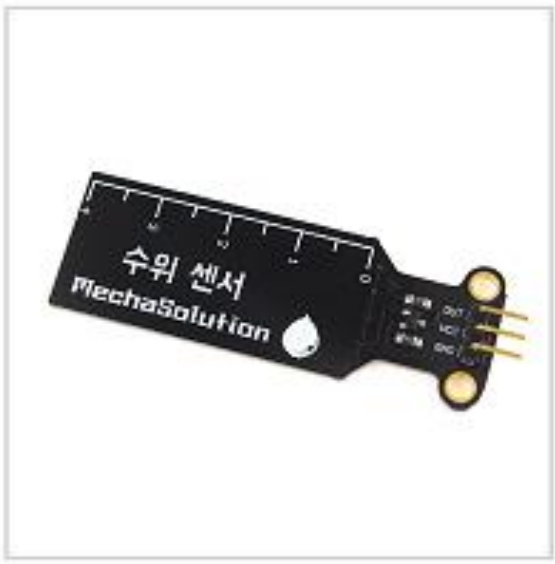
CDS 황화 카드뮴 광센서

아날로그 인터페이스

작동 전압: DC 3.3V ~ 5V

크기: 23 x 21 mm

	**사양 (Specification)** LM386 기반 사운드 센서 아날로그 인터페이스 작동 전압: DC 5V 크기: 26 x 21 mm 무게: 5g

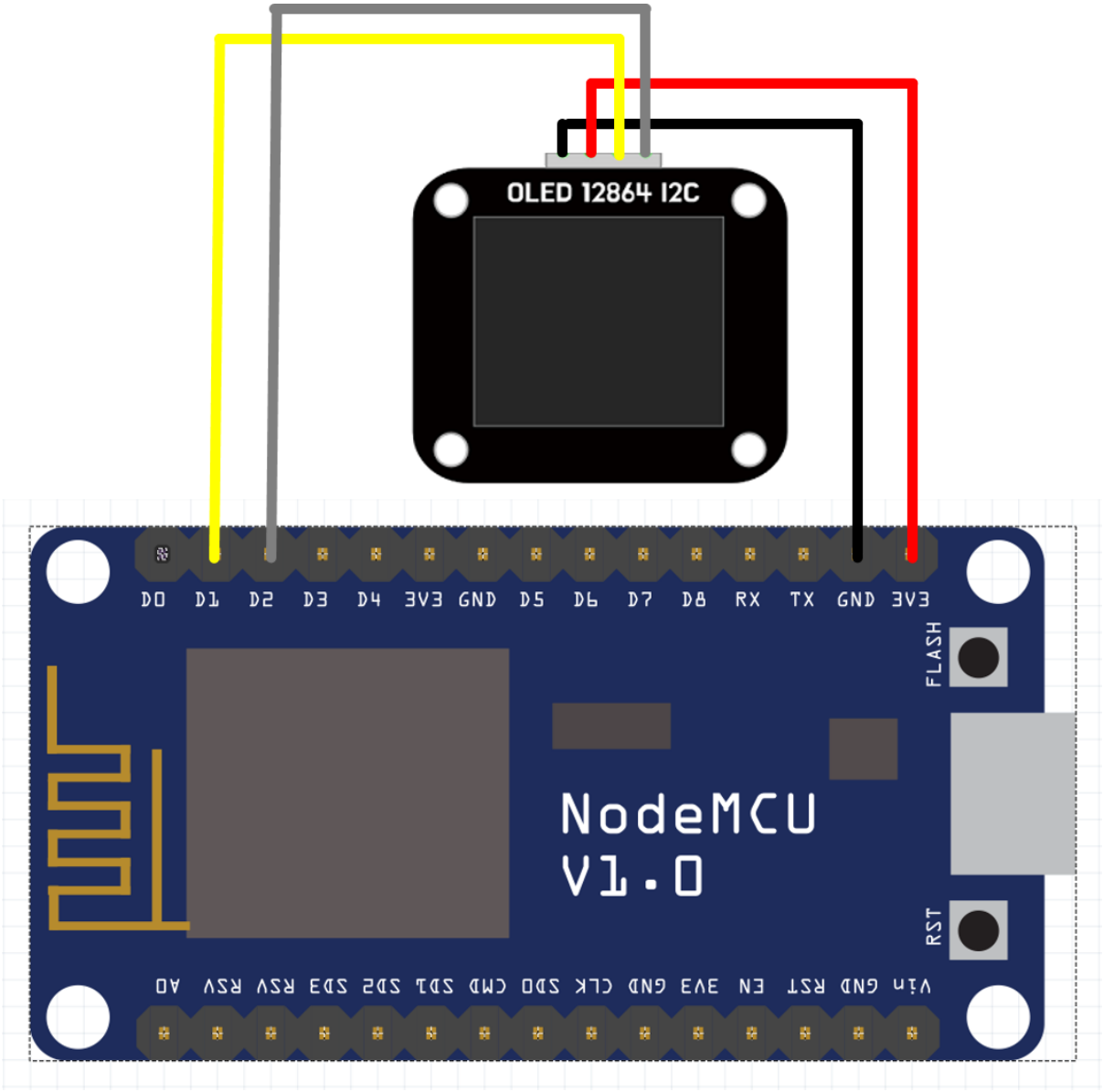
	**사양 (Specification)** 아날로그 인터페이스 작동 전압: 3.3~5V 크기: 60 x 40mm 무게: 4g

	**사양 (Specification)** MQ 시리즈 가스센서 아날로그 인터페이스 작동 전압: 5V 크기: 38 x 21mm 측정 가스: 알코올

10. 128X62 LCD에 원격으로 텍스트 출력하기

https://github.com/squix78/esp8266-oled-ssd1306

	**사양 (Specification)** 0.96 인치 OLED 디스플레이 SSD1306 드라이버칩 사용 I2C 인터페이스 I2C 주소: 0x3C 해상도: 128x64 컬러: 검정/배경, 흰색/글씨 크기: 38 x 28 x 9mm 무게: 16g

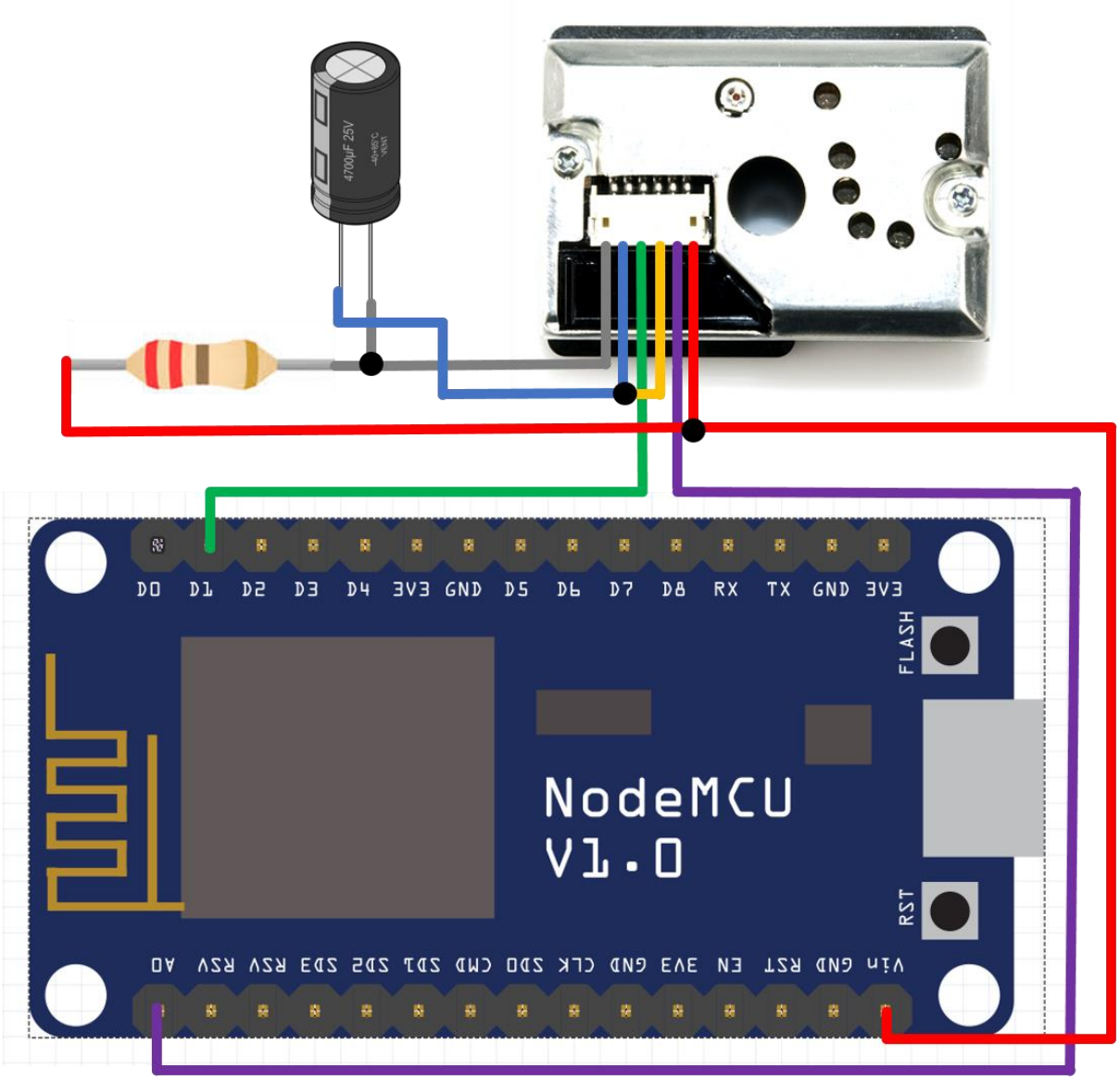


OLED 12864 I2C	NodeMCU 보드
GND	GND
VCC	3V3
SDA	D2
SCL	D1

```
/* Hello World OLED Test */
#include "SSD1306.h" // alias for `#include "SSD1306Wire.h"`
```

```
SSD1306   display(0x3c, 5, 4); // Initialise the OLED display using Wire library
void setup()
{
  Serial.begin(115200);
  display.init(); // Initialising the UI will init the display too.
  display.flipScreenVertically();

  display.clear();
  drawHelloWorld();
  display.display();
}
void loop()
{

}
void drawHelloWorld()
{
      display.setTextAlignment(TEXT_ALIGN_LEFT);
      display.setFont(ArialMT_Plain_10);
      display.drawString(0, 0, "Mechasolution");
      display.setFont(ArialMT_Plain_16);
      display.drawString(0, 10, "Mechasolution");
      display.setFont(ArialMT_Plain_24);
      display.drawString(0, 26, "Mechasolution");
}
```

11. 미세먼지 모니터링




```
int measurePin = 0;
int ledPower = 5;

unsigned int samplingTime = 280;
unsigned int deltaTime = 40;
unsigned int sleepTime = 9680;

float voMeasured = 0;
float calcVoltage = 0;
float dustDensity = 0;
```

```
void setup(){
  Serial.begin(9600);
  pinMode(ledPower,OUTPUT);
}

void loop(){
  digitalWrite(ledPower,LOW);
  delayMicroseconds(samplingTime);

  voMeasured = analogRead(measurePin);

  delayMicroseconds(deltaTime);
  digitalWrite(ledPower,HIGH);
  delayMicroseconds(sleepTime);

  calcVoltage = voMeasured*(5.0/1024);
  dustDensity = 0.17*calcVoltage-0.1;

  if ( dustDensity < 0)
  {
    dustDensity = 0.00;
  }

  Serial.println("Raw Signal Value (0-1023):");
  Serial.println(voMeasured);

  Serial.println("Voltage:");
  Serial.println(calcVoltage);

  Serial.println("Dust Density:");
  Serial.println(dustDensity);

  delay(1000);
}
```

원격 미세먼지 모니터링

```
#include <ESP8266WiFi.h>
const char* ssid = "iptime";
```

```
const char* password = "";

// 먼지센서 모니터링용 변수
int measurePin = 0;
int ledPower = 5;

unsigned int samplingTime = 280;
unsigned int deltaTime = 40;
unsigned int sleepTime = 9680;

float voMeasured = 0;
float calcVoltage = 0;
float dustDensity = 0;

WiFiServer server(80);

void setup() {
  Serial.begin(115200);
  delay(10);
  pinMode(ledPower,OUTPUT);

  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");
```

```
  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");
}

void loop() {
    // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
    return;
  }

  // Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){
    delay(1);
  }

  // Read the first line of the request
  String request = client.readStringUntil('₩r');
  Serial.println(request);
  client.flush();

  delay(1000);

  digitalWrite(ledPower,LOW);
  delayMicroseconds(samplingTime);

  voMeasured = analogRead(measurePin);

  delayMicroseconds(deltaTime);
  digitalWrite(ledPower,HIGH);
  delayMicroseconds(sleepTime);

  calcVoltage = voMeasured*(5.0/1024);
  dustDensity = 0.17*calcVoltage-0.1;
```

```
  if ( dustDensity < 0)
  {
    dustDensity = 0.00;
  }


  // Return the response
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println(""); //   do not forget this one
  client.println("<!DOCTYPE HTML>");
  client.println("<html>");
  client.print("Raw Signal Value (0-1023): ");
  client.println(voMeasured);
  client.print("Voltage:");
  client.println(calcVoltage);
  client.print("Dust Density:");
  client.println(dustDensity);
  client.println("</html>");

  delay(1);
  Serial.println("Client disonnected");
  Serial.println("");

}
```