NodeMCU

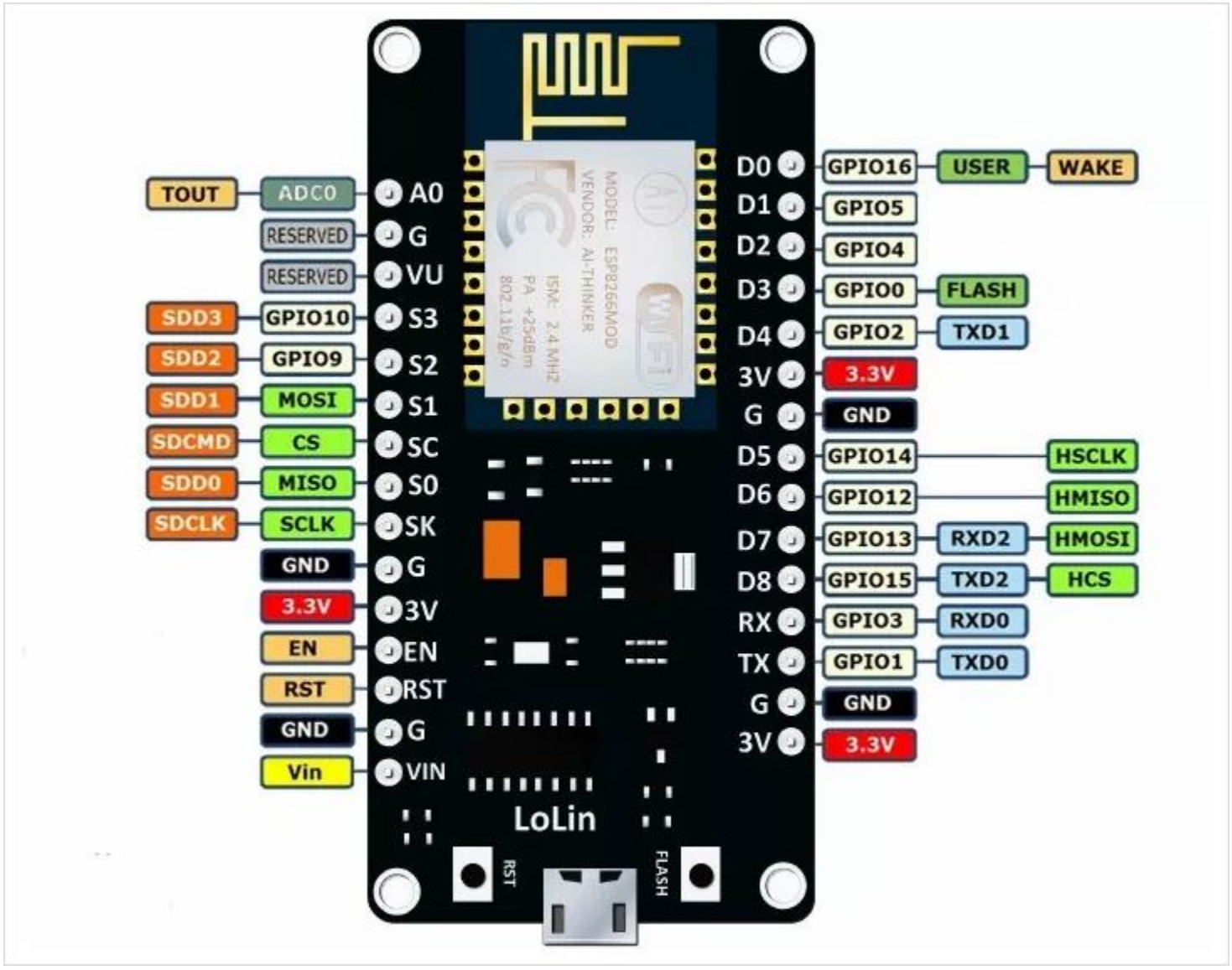
NodeMCU

❖ NodeMCU

- 오픈소스 사물인터넷 (IoT) 플랫폼
- 와이파이 기능이 구현된 MCU 개발보드
- ESPRESSIF사의 ESP8266-12 모듈을 사용

NodeMCU

❖ NodeMCU 핀 배치



NodeMCU

❖ NodeMCU 스펙

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

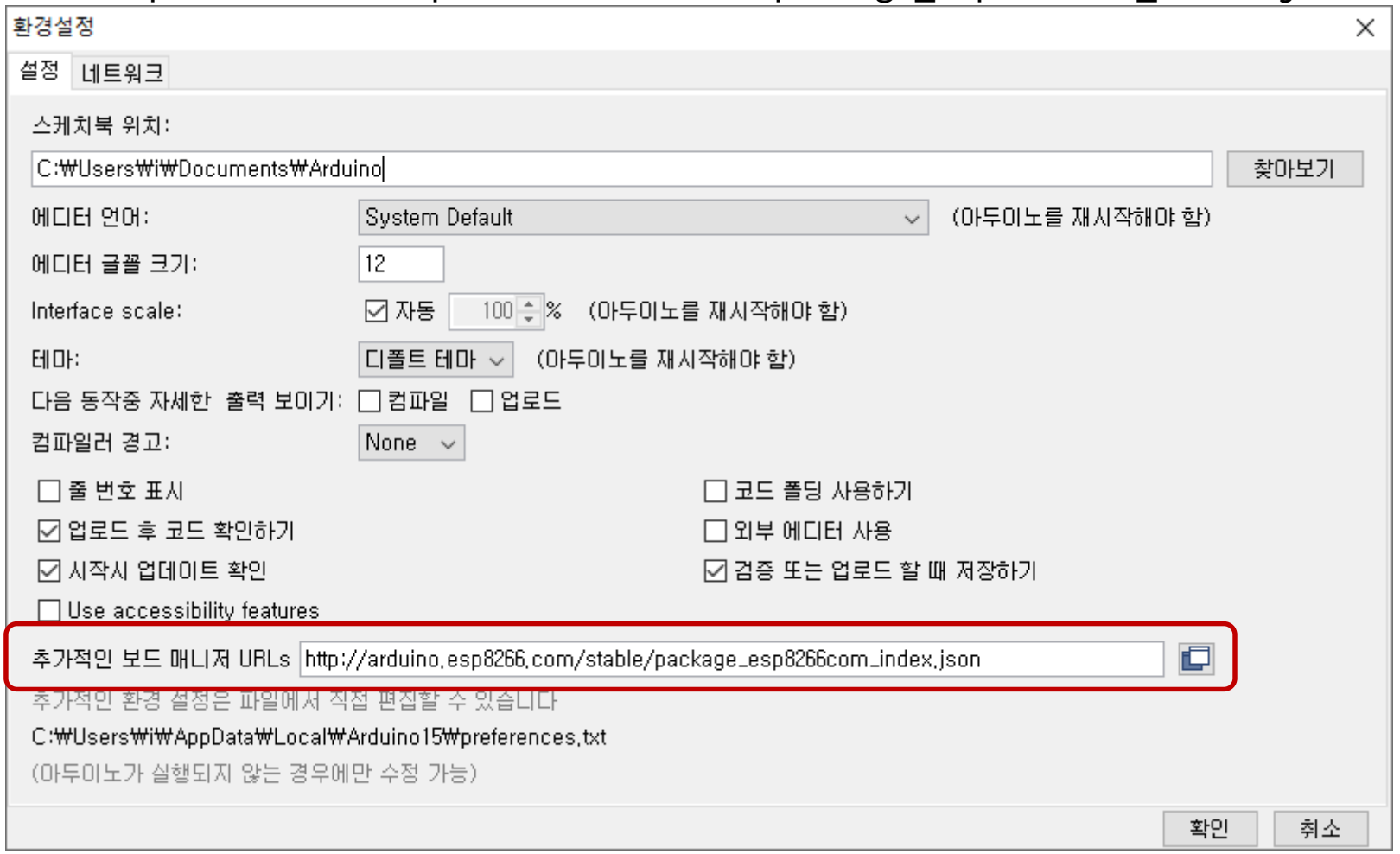
❖ 주요핀

- D3, D4, D8: 부팅모드를 설정하기 위해 내정
- D0: Sleep mode에서 벗어나기 위한 Wake용
- 사용에 제한이 없는 핀: D1, D2, D5, D6, D7
- 내장LED: D4

NodeMCU

❖ 개발환경 설정

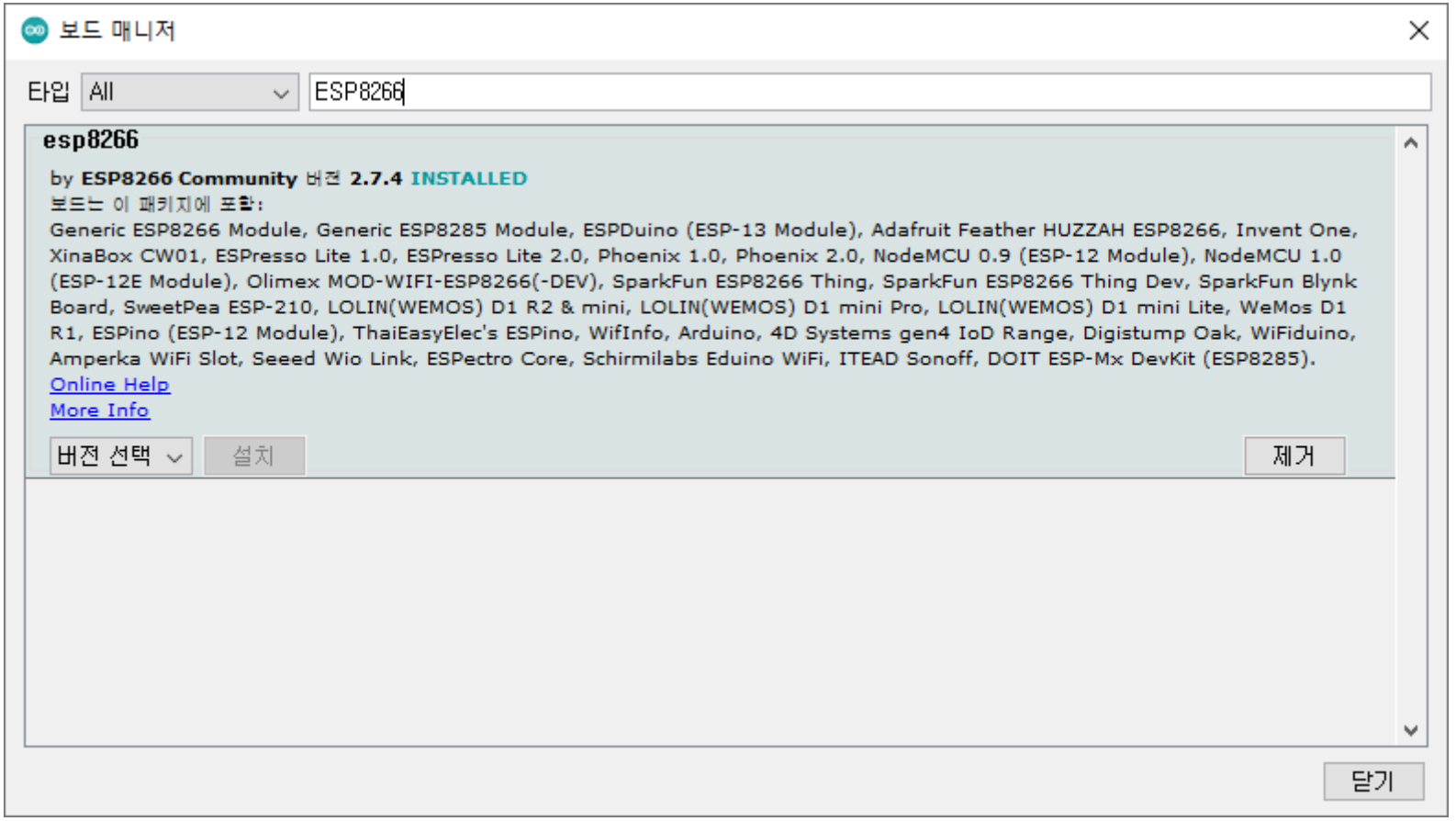
- ㅇ 파일 > 환경 설정
  - ▪ 추가적인 보드 매니저 URLs:
    http://arduino.esp8266.com/stable/package_esp8266com_index.json



NodeMCU

❖ 개발환경 설정

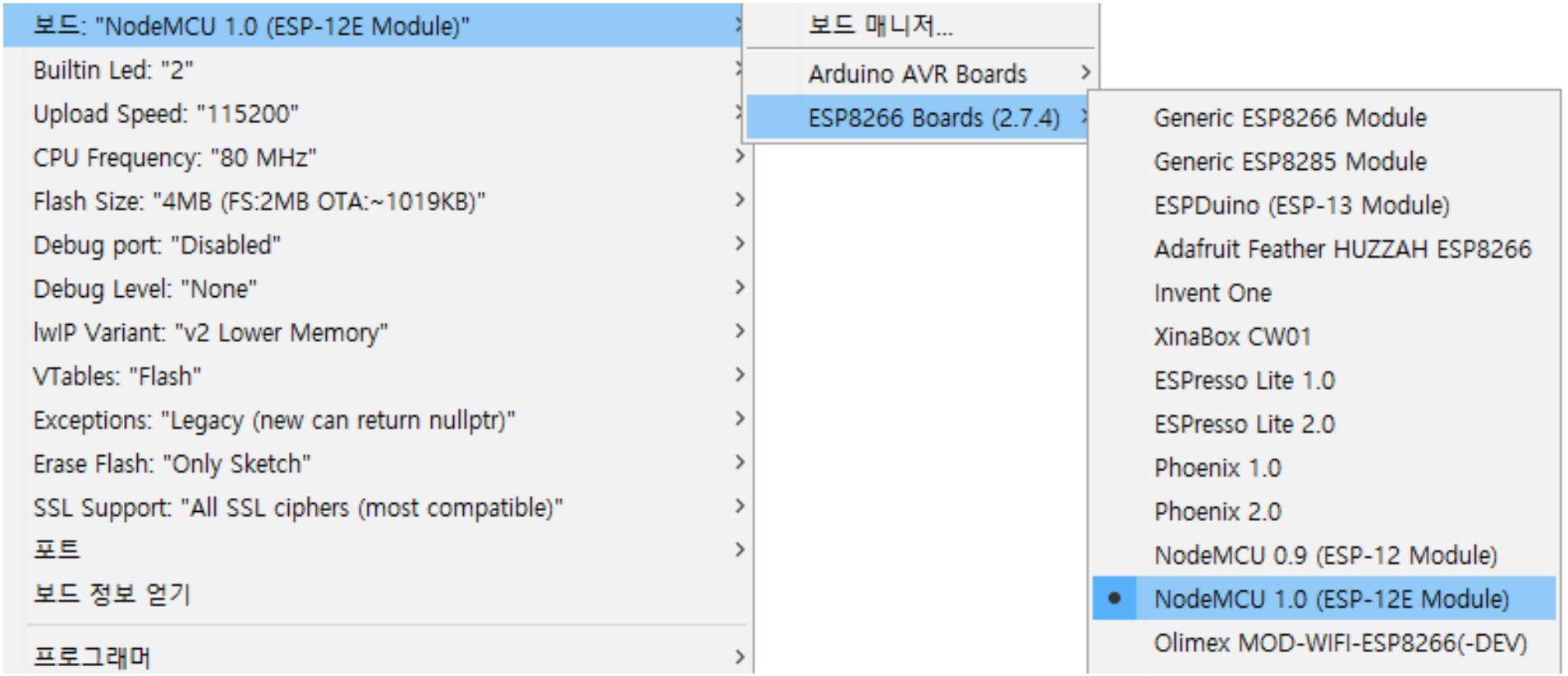
- 툴 > 보드 > 보드매니저...
  - ESP8266 검색 및 설치



NodeMCU

❖ 개발환경 설정

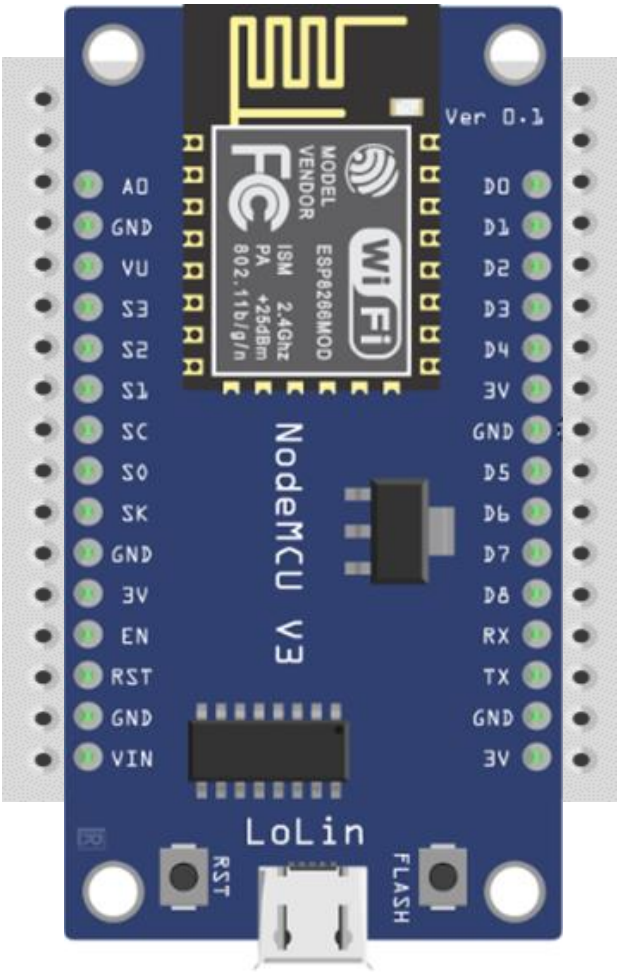
- 툴 > 보드 > ESP8266 Boards(2.7.4)
  - NodeMCU 1.0 (ESP-12E Module) 선택



NodeMCU

❖ 기본예제

- ○ lcd 연결
  - VCC : VIN(5V)
  - GND: GND
  - SCL: D1
  - SDA: D2



NodeMCU

❖ ex01/app.ino

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

const char *ssid = "Campus7_Room4_2.4GHz";
const char *password = "12345678";
MiniCom com;

void wifi_connect() {
    WiFi.begin(ssid, password); // 비밀번호가 없는 경우 NULL
    com.print(0, "try to connect");
    Serial.println();
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }

    com.print(0, "WiFi connected");
    com.print(1, WiFi.localIP().toString().c_str());
    Serial.println();
    Serial.println(WiFi.localIP());
}
```

NodeMCU

❖ ex01/app.ino

```
void setup() {
    com.init();
    wifi_connect();
}

void loop() {
    com.run();
}
```

NodeMCU

❖ WifiMiniCom.h

```
#pragma once

#include <ESP8266WiFi.h>
#include <MiniCom.h>

class WifiMiniCom : public MiniCom {
public:
    WifiMiniCom(int serial_bps=115200, int lcd_addr=0x27);
    void init(const char *ssid, const char *password);
};
```

NodeMCU

❖ WifiMiniCom.cpp

```
#include "WifiMiniCom.h"

WifiMiniCom::WifiMiniCom(int serial_bps, int lcd_addr)
    : MiniCom(serial_bps, lcd_addr) {
}

void WifiMiniCom::init(const char *ssid, const char *password) {
    MiniCom::init();

    WiFi.begin(ssid, password); // 비밀번호가 없는 경우 NULL
    print(0, "try to connect");
    Serial.println();
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }

    print(0, "WiFi connected");
    print(1, WiFi.localIP().toString().c_str());
    Serial.println();
    Serial.println(WiFi.localIP());
}
```

NodeMCU

❖ ex02/app.ino

```
#include <WifiMiniCom.h>

const char *ssid = "Campus7_Room4_2.4GHz";
const char *password = "12345678";

WifiMiniCom com;

void setup() {
    com.init(ssid, password);
}

void loop() {
    com.run();
}
```

NodeMCU

❖ 웹 서버

```
#include <WifiMiniCom.h>

const char *ssid = "Campus7_Room4_2.4GHz";
const char *password = "12345678";

WifiMiniCom com;
WiFiServer server(80);  //80:  Web Server 표준 포트

void setup() {
    com.init(ssid, password);
    server.begin();
}
```

NodeMCU

❖ 웹 서버

```
void loop() {
  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();
```

❖ 웹 서버

```
  // 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("");
}
```