

NodeMCU로 시작하는 사물인터넷 DIY

2020. 05.

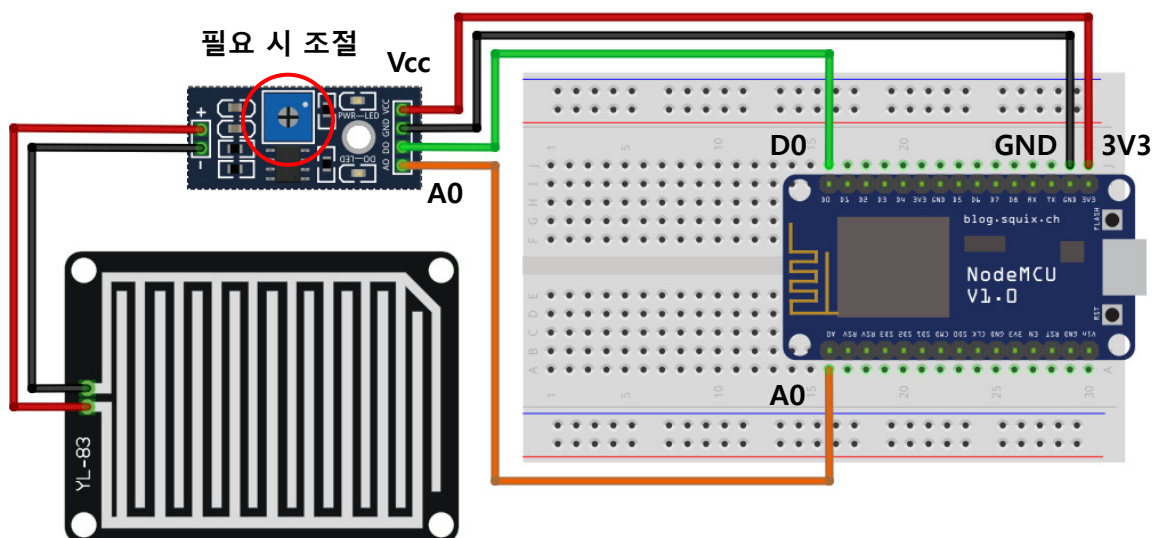
김 학 용



강우센서(Raindrop) 이용하기

◆ 강우센서의 기본 동작 확인하기

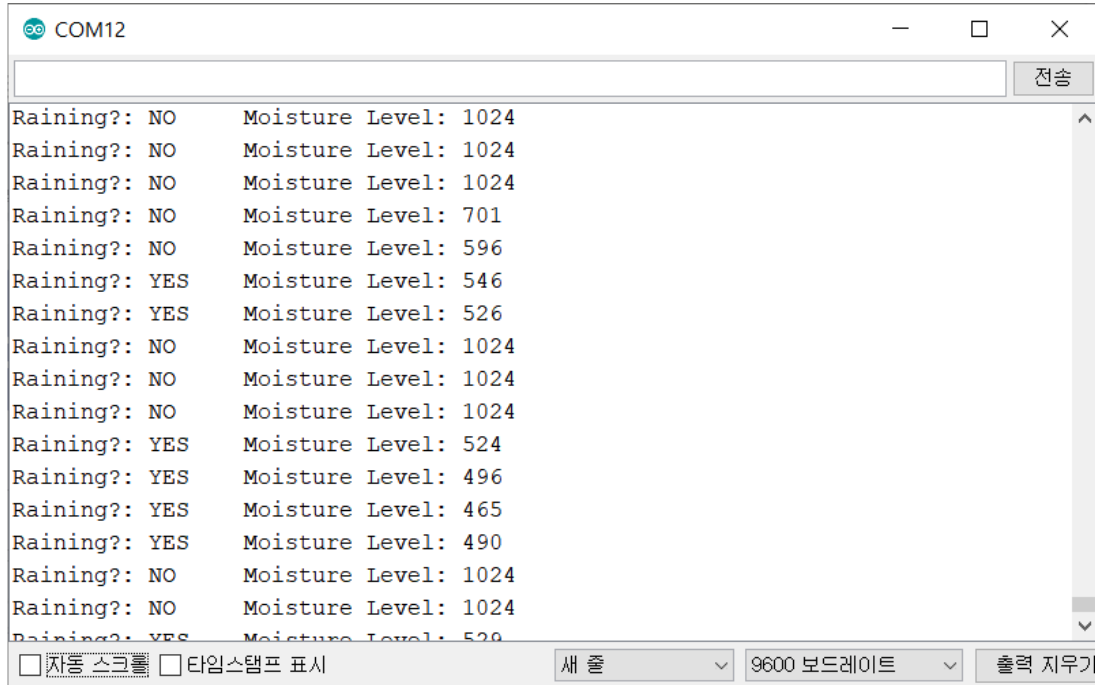
- 아래와 같이 회로를 구성 (A0는 A0에, D0는 D0에 연결)
- 아두이노 IDE에서 다운로드 받은 Raindrop_Sensor.ino를 불러서 컴파일 & 업로드
 - 컴파일&업로드 전에 아두이노 IDE의 '툴' 메뉴에서 '시리얼 모니터' 선택



강우센서(Raindrop) 이용하기

◆ 결과 확인

- 힘을 조절해가며 물에 적신 휴지를 센서 위에 올렸다 떼어보며 데이터값 확인
 - 아날로그 센서 값이 약 550 이하일 때 비가 온다고 판단 (개별 조절 필요)



강우센서(Raindrop) 이용하기

◆ 코드 이해

- analogRead() 함수를 이용해서 강우 수준을 측정 (수분이 없을 때 1024)
- digitalRead() 함수를 이용해서 강우 수준을 Yes/No로 결정

```
int AsensorPin = A0;
int DsensorPin = D0;
int sensorValue;
boolean isRaining = false;
String strRaining;

void setup() {
  Serial.begin(9600);
  pinMode(D0, INPUT);
}

void loop() {
  sensorValue = analogRead(AsensorPin);
  isRaining = !(digitalRead(DsensorPin));

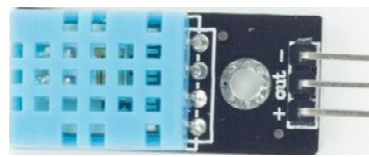
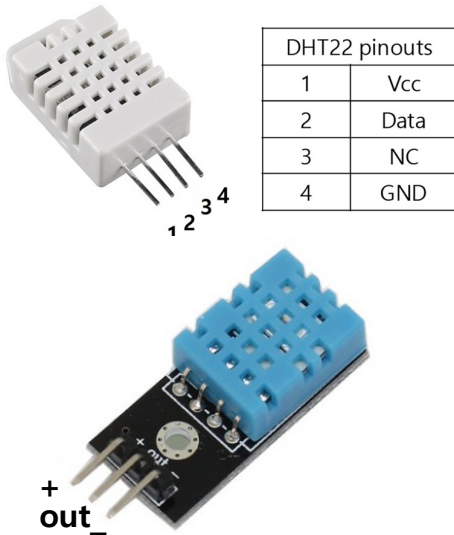
  if(isRaining){
    strRaining = "YES";
  } else{
    strRaining = "NO";
  }

  Serial.print("Raining?: ");
  Serial.print(strRaining);
  Serial.print("\t Moisture Level: ");
  Serial.println(sensorValue);

  delay(200);
}
```

스마트 온습도 측정기 (1)

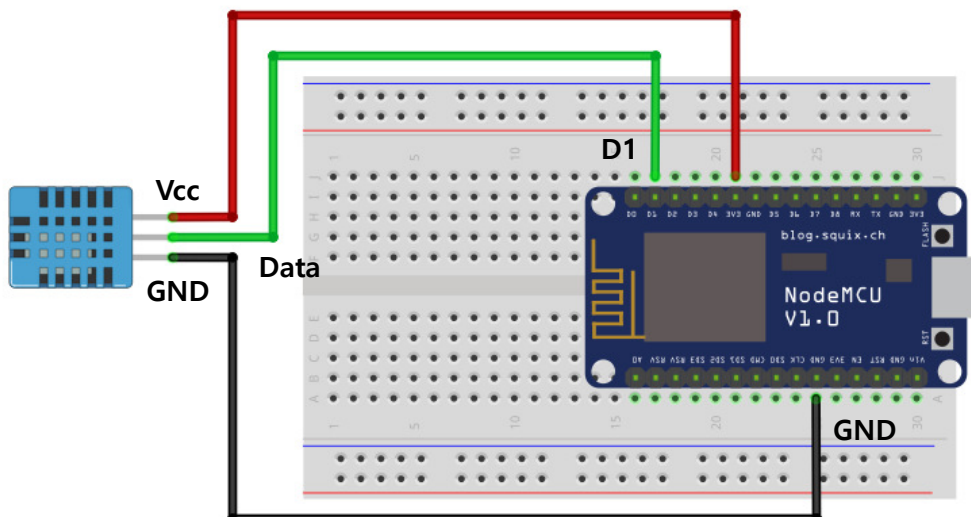
- ◆ 온도와 습도를 동시에 측정할 수 있는 DHT11 센서(파란색) 이용
 - 온도와 습도를 개별적으로 측정하는 센서들도 존재함
 - DHT11/22은 원래 4핀으로 구성되어 있으나, 실제로는 3개의 핀만 이용



GND 핀에 연결
D1에 연결
3V3 혹은 Vcc에 연결

스마트 온습도 측정기 (1)

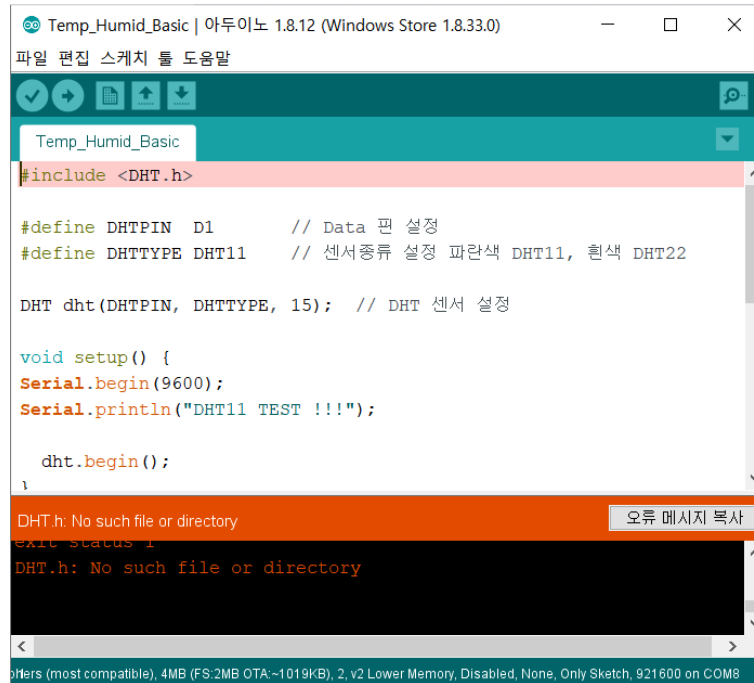
- ◆ 그림과 같이 회로를 구성 (DHT11의 핀 위치가 다를 수 있으므로 주의!)
- ◆ 아두이노 IDE에서 다운로드 받은 Temp_Humid_Basic.ino 열기
- ◆ DHT 라이브러리 다운로드 후 컴파일 및 업로드



스마트 온습도 측정기 (1)

◆ DHT 라이브러리 설치

- DHT11 센서를 이용하기 위해서는 미리 해당 라이브러리를 설치해야 함
→ 그렇지 않으면 DHT.h No such file or directory 라는 에러가 발생



```
Temp_Humid_Basic | 아두이노 1.8.12 (Windows Store 1.8.33.0)
파일 편집 스케치 툴 도움말

Temp_Humid_Basic
#include <DHT.h>

#define DHTPIN D1 // Data 핀 설정
#define DHTTYPE DHT11 // 센서종류 설정 파란색 DHT11, 흰색 DHT22

DHT dht(DHTPIN, DHTTYPE, 15); // DHT 센서 설정

void setup() {
  Serial.begin(9600);
  Serial.println("DHT11 TEST !!!");

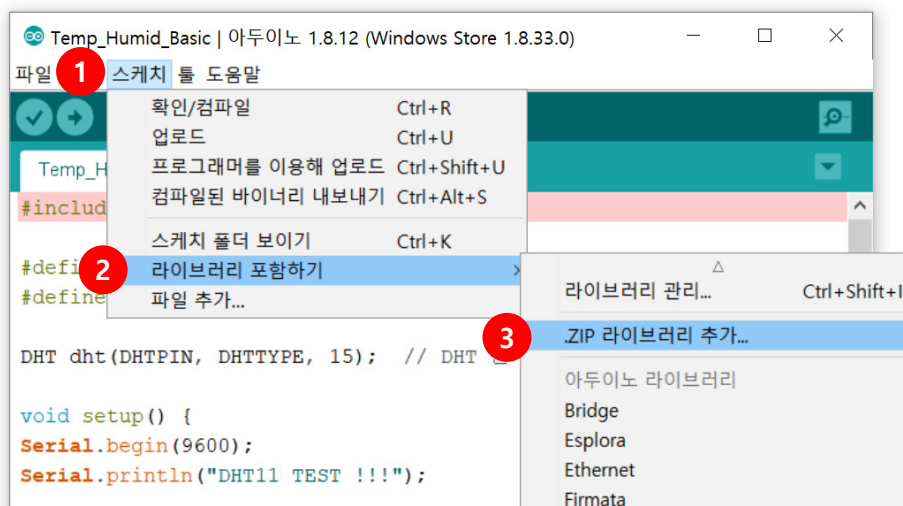
  dht.begin();
}

DHT.h: No such file or directory
exit status 1
DHT.h: No such file or directory
```

스마트 온습도 측정기 (1)

◆ DHT 라이브러리 설치

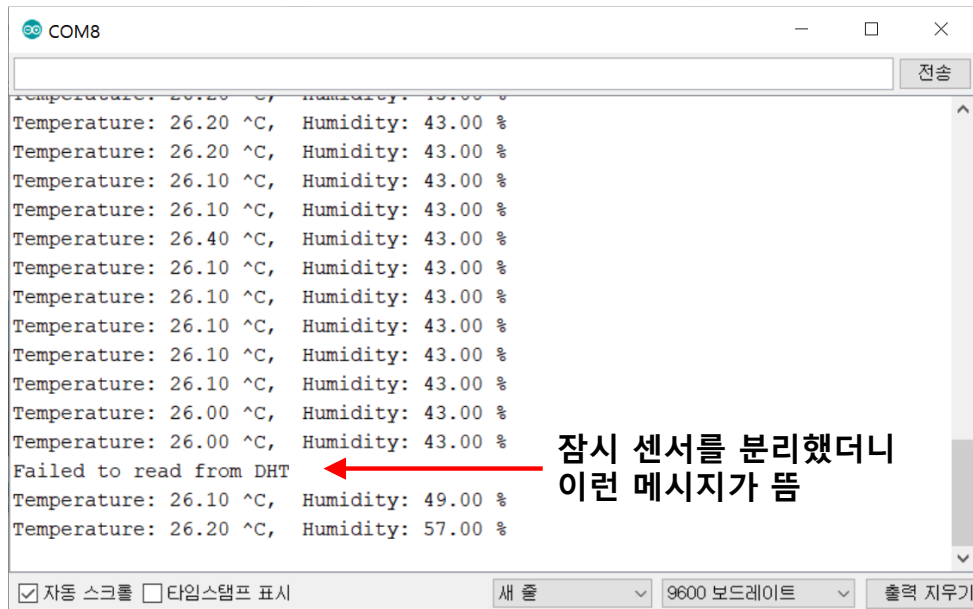
- 아두이노 IDE의 (1) 스케치 메뉴에서 (2) 라이브러리 포함하기 선택 후
(3) .ZIP 라이브러리 추가 → GitHub에서 다운로드 후 압축해제한 디렉토리에서
Zipped Library 폴더를 찾고 DHT-sensor-library-master.zip 파일을 선택하여 추가
- 같은 방법으로 Adafruit_Sensor_master.zip 파일을 선택하여 추가



스마트 온습도 측정기 (1)

◆ 컴파일 및 실행

- 2개의 라이브러리를 성공적으로 추가했으면 (1) 툴 메뉴에서 (2) 시리얼 모니터 선택
- ➔ 버튼을 클릭해서 컴파일 및 실행 후 시리얼 모니터에 나오는 결과 확인
- 문제가 있거나 센서를 잘못 연결하면 Failed to read from DHT라는 메시지가 출력됨



스마트 온습도 측정기 (2)

◆ 코드 리뷰

- `#include <DHT.h>` 와 같은 방식으로 필요한 라이브러리 추가
- `#define DHTPIN D1` 과 같은 방식으로 전역 변수 정의
- `isnan(var)` 함수는 var 값이 숫자가 아니면 1 값을 회신하는 함수 (is not a number)
- `Serial.print()` 함수를 이용해서 시리얼 모니터에 결과값 출력

```
#include <DHT.h>

#define DHTPIN D1          // Data 핀 설정
#define DHTTYPE DHT11     // 센서종류 설정  파란색 DHT11

DHT dht(DHTPIN, DHTTYPE, 15); // DHT 센서 설정

void setup() {
  Serial.begin(9600);
  Serial.println("DHT11 TEST !!!");

  dht.begin();

  delay(3000);
}

void loop() {
  float t = dht.readTemperature();
  float h = dht.readHumidity();

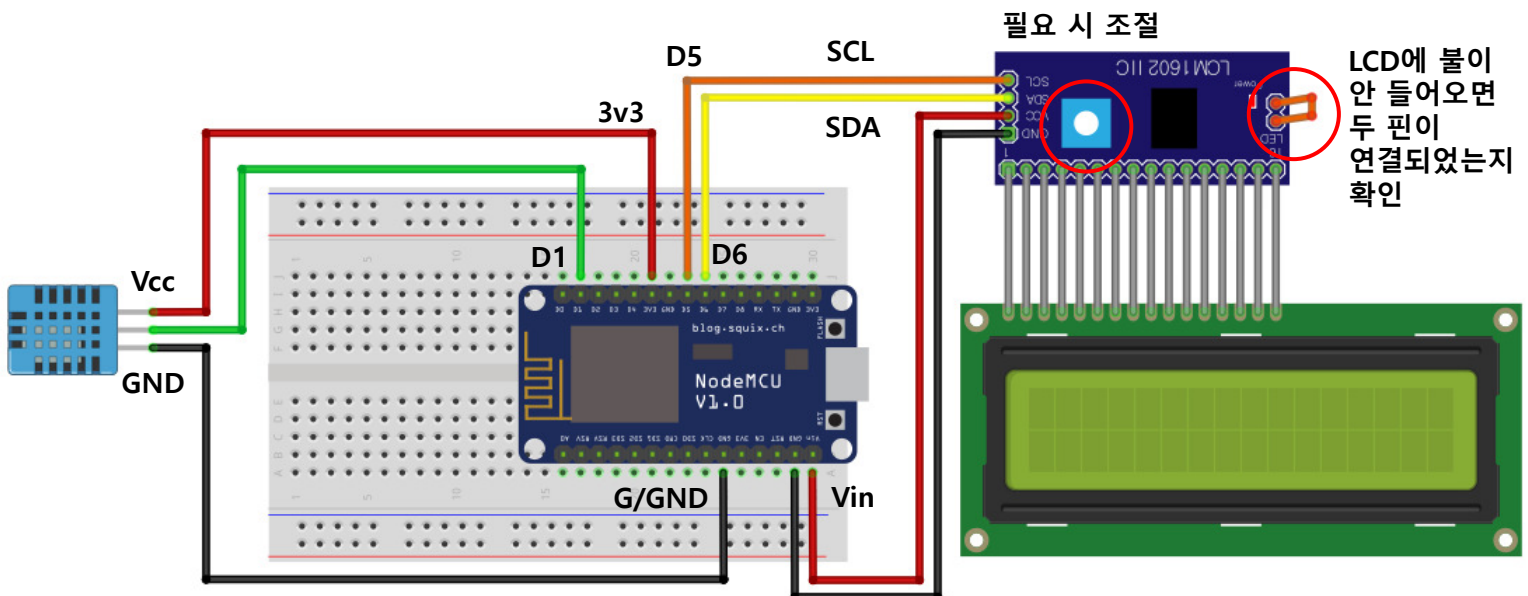
  if (isnan(t) || isnan(h)) {
    Serial.println("Failed to read from DHT");
  } else {
    Serial.print("Temperature: ");
    Serial.print(t);
    Serial.print(" ^C,\t");
    Serial.print("Humidity: ");
    Serial.print(h);
    Serial.println(" %");
  }
  delay(3000);
}
```

I2C LCD 이용하기

- ◆ LCD 1602는 한 줄에 16글자씩 2줄로 표시하는 디스플레이 모듈
 - 화면에 글자를 표시하기 위해서는 16개 선을 연결해야만 함
 - 또한, LCD를 이용하기 위한 라이브러리를 추가해 주어야 함
 - 화면에 표시할 글자를 시리얼 통신(I2C, Inter-Integrated Circuit) 방식으로 전송하면 전원(Vcc, GND) 빼고 2개의 연결만 있으면 됨 (모두 4개의 연결)
- ◆ I2C 통신을 하기 위해서 필요한 일
 - NodeMCU와 I2C 통신을 하기 위해서는 전용 라이브러리를 설치해야 함
 - IDE의 스케치 메뉴에서 '라이브러리 추가'를 선택하고 Zipped 라이브러리 추가를 선택한 후 LiquidCrystal_I2C-master.zip 파일을 선택

온습도 센서와 I2C LCD를 함께 이용하기

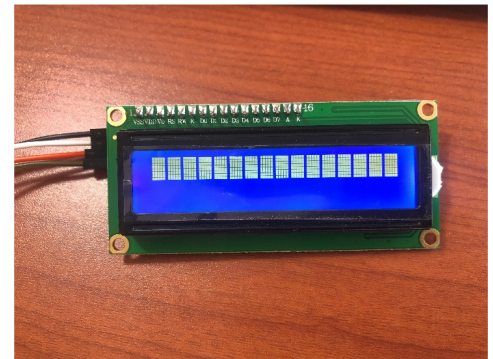
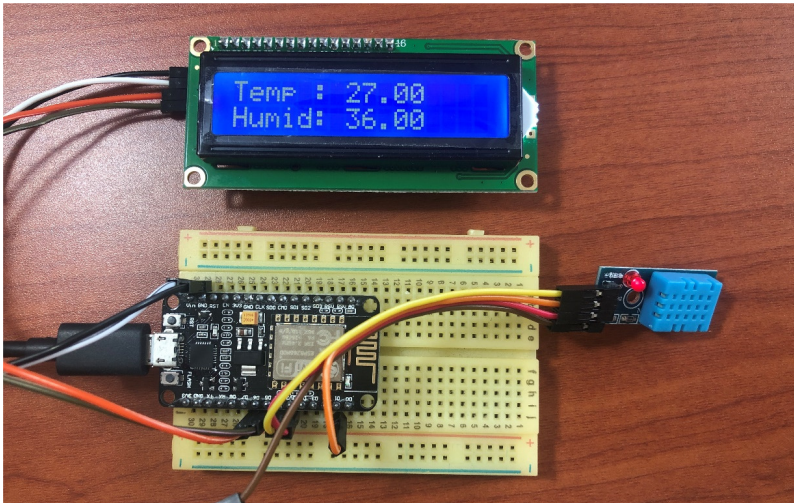
- ◆ 이전과 동일한 회로에 I2C LCD만 추가로 연결
 - I2C LCD의 Vcc는 Vin에 연결, SDA는 D6, SCL은 D5에 연결
- ◆ IDE에서 Temp_Humid_I2C_LCD.ino 열고 코드 확인
 - DHTPIN을 D1으로 설정, DHTTYPE을 DHT11로 설정 (파란색)



온습도 센서와 I2C LCD를 함께 이용하기

◆ 결과 확인 및 오류 해결

- 정상적으로 동작하는 경우, 아래 사진처럼 LCD에 온도/습도값이 출력됨
- LCD에 오른쪽 사진처럼 표시되는 경우 코드의 LiquidCrystal_I2C lcd(0x27, 16, 2); 부분에서 0x27을 0x3F로 변경해서 다시 컴파일
- 글자가 안 보이는 경우에는 드라이버를 이용해서 앞의 ○ 표시 부분을 조절



이렇게 보이는 경우에는
0x27을 0x3F로 바꾼다.

온습도 센서와 I2C LCD를 함께 이용하기

◆ 코드 설명

```
#include <LiquidCrystal_I2C.h>  ← LCD를 I2C 방식으로 이용하도록 하는 라이브러리 포함
#include <DHT.h>                ← 온습도센서인 DHT11을 이용하도록 하는 라이브러리
#include <Wire.h>

#define DHTPIN D1                // Data 핀 설정
#define DHTTYPE DHT11           // 센서종류 설정

DHT dht(DHTPIN, DHTTYPE, 15);   // DHT 센서 설정

LiquidCrystal_I2C lcd(0x27, 16, 2); ← I2C 방식으로 통신하는 주소(0x27)과 LCD의 크기 16x2
//LiquidCrystal_I2C lcd(0x3F, 16, 2);

void setup() {
    Serial.begin(9600);
    Serial.println("DHT11 TEST !!!");

    dht.begin();

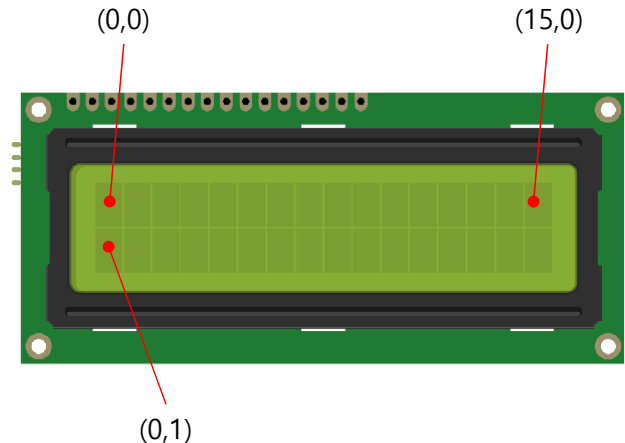
    lcd.begin(D6, D5); //LCD의 SDA, SCL에 각각 연결 ← LCD를 구동시키며 통신 채널은 NodeMCU의 D6와 D5.
    lcd.backlight();   ← D6/D5를 변경함으로써 I2C 통신 핀을 변경할 수 있음.
                       LCD의 백라이트를 켜줌

    lcd.clear();       ← 이전에 LCD에 표시했던 글자를 지워줌
    lcd.setCursor(0, 0); ← LCD에 표시할 글자의 위치
    lcd.print("DHT11 Test!!"); ← LCD에 DHT11 Test!! 를 표시
    delay(2000);
}
```

온습도 센서와 I2C LCD를 함께 이용하기

◆ 코드 설명

```
void loop() {  
    float t = dht.readTemperature();  
    float h = dht.readHumidity();  
  
    if (isnan(t) || isnan(h)) {  
        Serial.println("Failed to read from DHT");  
        lcd.clear();  
        lcd.setCursor(0,0);  
        lcd.print("Failed to Read!!");  
    } else {  
        Serial.print("Temperature: ");  
        Serial.print(t);  
        Serial.print(" ^C,\t");  
        Serial.print("Humidity: ");  
        Serial.print(h);  
        Serial.println(" %");  
        lcd.clear();  
        lcd.setCursor(0,0);  
        lcd.print("Temp : ");  
        lcd.print(t);  
        lcd.setCursor(0,1);  
        lcd.print("Humid: ");  
        lcd.print(h);  
    }  
    delay(3000);  
}
```



네트워크 시계 만들기

- ◆ 이전 회로 그대로 유지 (온습도 센서도 분리하지 마세요)
- ◆ 아두이노 IDE에서 NTP_I2C_LCD.ino 열어 코드 수정
 - IDE의 스케치 메뉴에서 라이브러리 추가하기 선택 후 Zipped 라이브러리 추가하기
 - 다운로드 받은 폴더의 Zipped Library 폴더에서 Time-master.zip 선택
 - 코드에서 ssid와 password 부분을 자신의 와이파이 이름과 비밀번호로 변경

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

- LCD에 시간과 날짜가 표시되지 않는 경우 0x27을 0x3F로 변경

```
LiquidCrystal_I2C lcd(0x27, 16, 2); // 0x27 or 0x3F
```

- ◆ 이번에는 아두이노 IDE에서 NTP_Temp_I2C_LCD.ino 열어 테스트
 - 이전과 동일한 방식으로 코드 확인 및 수정
 - DHTPIN과 DHTTYPE이 D1 및 DHT11인지 확인하고 수정

네트워크 시계 만들기

◆ 결과 확인하기 (오른쪽은 네트워크 시계 + 온습도 센서)

