International Institute of Information Technology

Research Teaser Internship '21
- Internet of Things

Overview: In this lab session, you will be interfacing a DHT11 temperature-humidity sensor with the nodemcu and write the outputs to the serial monitor. Once that is done, you will be working to communicate the temperature-humidity values from the nodemcu with the sensor to the arduino UNO using I2C interface.

Part 1) DHT11 Temperature & Humidity sensor on NodeMCU using Arduino IDE

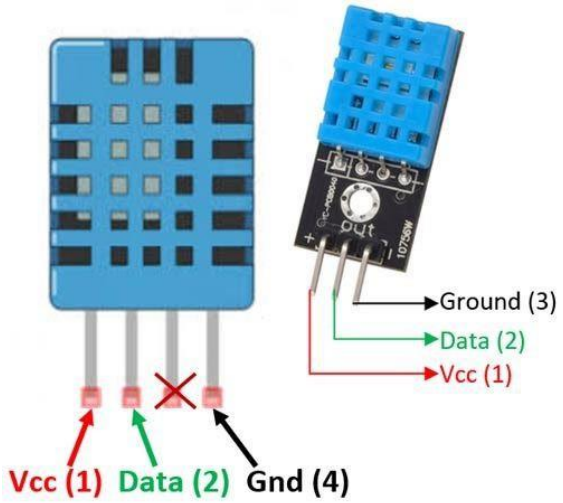
a) Introduction and understanding the DHT11 sensor:

i) The DHT11 is a commonly used Temperature and humidity sensor. The sensor comes with a dedicated NTC to measure temperature and an 8-bit microcontroller to output the values of temperature and humidity as serial data.

ii) The DHT11 is chosen because it is lab calibrated, accurate and stable and its signal output is digital.

iii) Most important of all, it is relatively inexpensive for the given performance.

iv) Below is the pinout diagram of the DHT11 sensor



v) DHT11 Specifications:

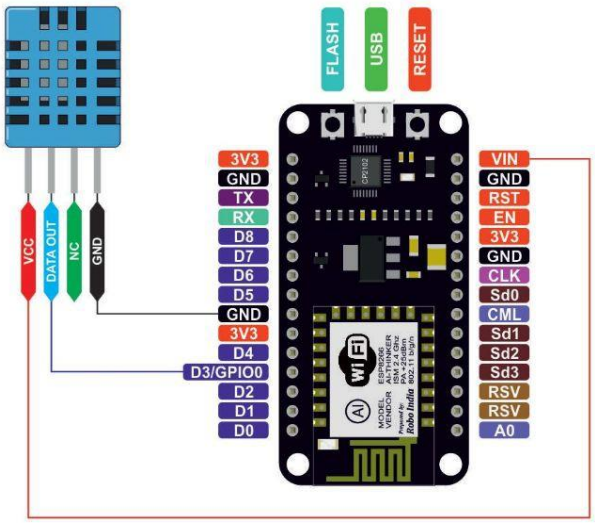
1) Operating Voltage: 3.5V to 5.5V
2) Operating current: 0.3mA (measuring) 60uA (standby)
3) Output: Serial data
4) Temperature Range: 0°C to 50°C
5) Humidity Range: 20% to 90%
6) Resolution: Temperature and Humidity both are 16-bit
7) Accuracy: ±1°C and ±1%

vi) Please refer to the data sheet for more information about the DHT11 sensor. Link to the data sheet here.

b) Components required for this exercise:

i) DHT11 sensor
ii) nodemcu
iii) Breadboard
iv) Few jumper wires

c) Circuit Diagram:



d) **Library Files:** Following two libraries will be required to run this code. Download the zip file, extract the same and copy this to your Arduino library folder.
   i) **Library 1**: You may download library file from here.
   ii) **Library 2**: You may download library file from here.

e) **Pseudocode**:
   i) Include the library of the DHT11 temperature and humidity sensor using #include “DHT.h”
   ii) Define the DHTtype and digital pin, in this case type is DHT11 and digital pin is GPIO 0/D3 using #define DHTTYPE and #define dht_dpin.
   iii) Instantiate the DHT dht(dht_dpin, DHTTYPE)
   iv) In the setup block, use dht.begin(), Serial.begin(<baud rate>) to begin the serial communication. Use delay(<delay in milliseconds>) to give an appropriate delay.
   v) In the loop block, use float variables to read the temperature and humidity values from the sensor using dht.readHumidity(), dht.readTemperature(); Serial.print()/Serial.println() to print the values on the serial monitor and give an appropriate delay using delay(<delay in milliseconds>) according to the frequency of sensor.
   vi) You can refer to the example codes in the library you already downloaded for the syntaxes as well as some additional points.

f) **Expected output**:
   i) The output values should be the temperature and humidity values measured by the DHT11 sensor at each instant it is sensing corresponding to your delay.
   ii) Compare your values with the room temperature and verify if the output temperature value is not too far from it, else your sensor is not working.
   iii) Try playing around with the sensor a little like blowing on to it etc. which should increase the temperature and verify.

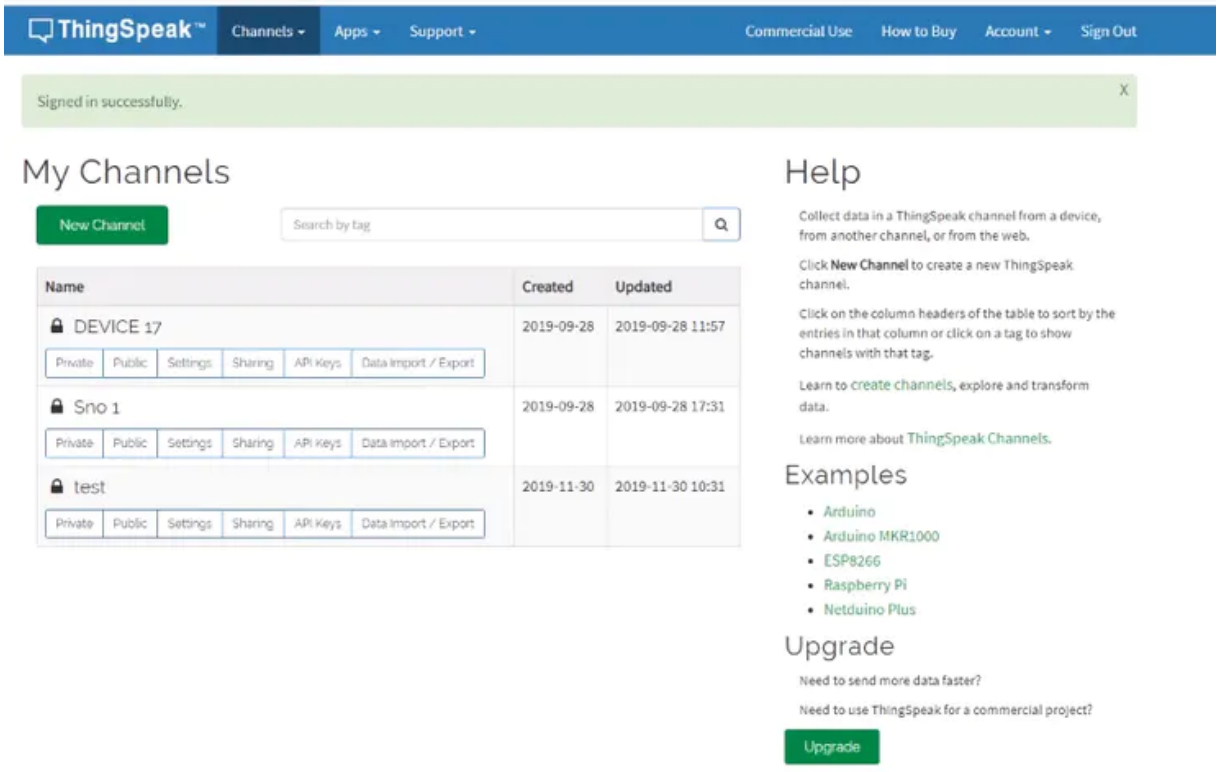
Note: The above tutorial is available on the net. Please refrain from copying the code directly, if anyone is seen copying from the net blindly, he/she will be penalized.

Part 2) Posting the Temperature values to Thingspeak.

a) Introduction:

The platform is primarily aimed towards IoT Projects and data analytics using visuals.

To get started with the free services of Thingspeak you will first need to Sign Up using your email ID, once that is done along with the email verification you will be greeted with a similar-looking page:



Now looking at some terminologies that might help you understand this better and make your work with web servers like these smoother:

1) **Reading/Downloading Data:** Getting data on your ESP8266/ESP32 from the server is a read operation.

2) **Writing/Uploading Data**: Sending data from your ESP8266/ESP32 to the server is a write operation.

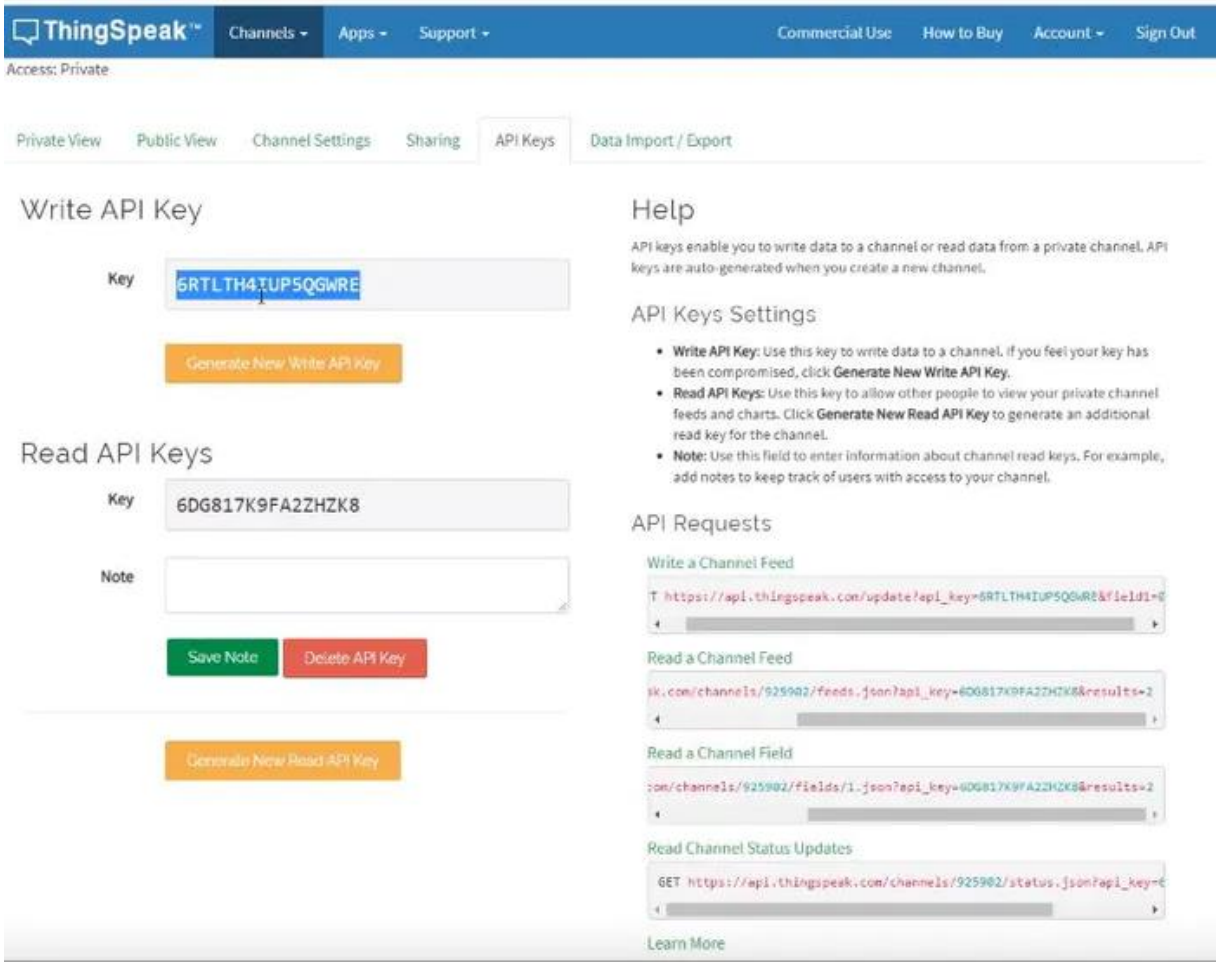
3) **API Key**: To have data security and to prevent anyone randomly from reading/writing data to your server there needs to be some sort of security/password and the API Key is

something intended towards this. API Key is a long alphanumeric key which is needed to read/data to the server. There are separate keys for reading and writing data.

4) **Channel**: A channel in thingspeak is a software counterpart of an IoT hardware device that you connect to Thingspeak, in our case an ESP8266 will utilise one entire channel of our bandwidth. In a free account of thingspeak, you can have a maximum of 4 channels.

5) **Field**: Each channel has 8 fields. A field is a variable and stores/shares a data type, for example when we send temperature and humidity from our device to the server, both the parameters will use one field each of the channel.

That's pretty much it about thingspeak!



Copy and keep the Write API Key, we will need it later while testing out the link to Thingspeak.

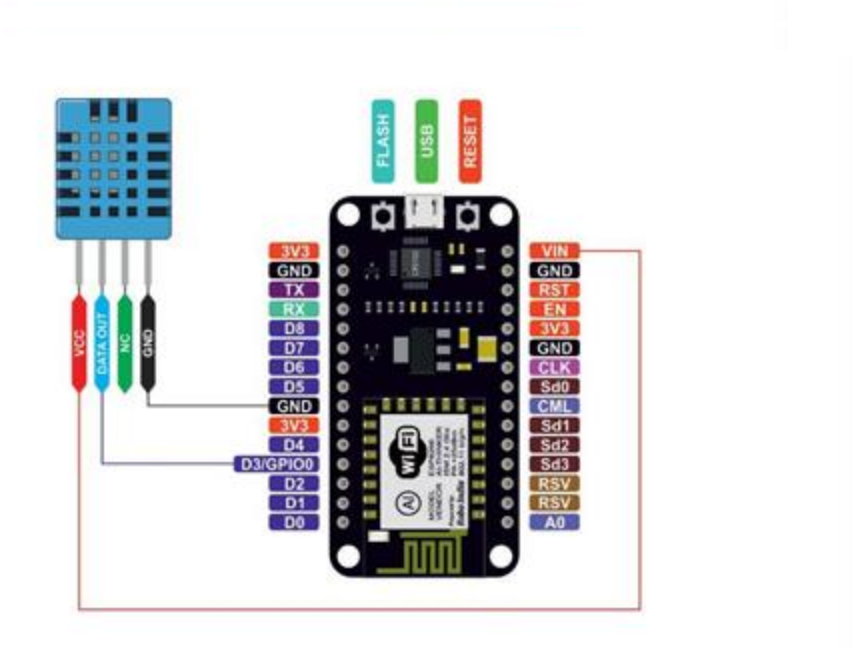
b) Components required:

NodeMCU DHT11

sensor Breadboard

Few jumper wires

c) Circuit Diagram: Make these new connections apart from the earlier ones in part I. (Make these new connections to the already existing circuit)



d) Pseudocode:

1. Include the ESP8266WiFi.h file to use the Wi-Fi library.

2. Create a string ssid = “Your SSID” and another password = “Your Password” .

3. Instantiate a WifiServer object and set the port as 80. Port 80 is commonly used for HTTP requests : WiFiServer my_server(80)

4. Include the library of the DHT11 temperature and humidity sensor using #include “DHT.h”

5.Define the DHTtype and digital pin, in this case type is DHT11 and digital pin is GPIO 0/D3 using #define DHTTYPE and #define dht_dpin.

6.Instantiate the DHT dht(dht_dpin, DHTTYPE)

7. In the setup function

a. Initiate Serial communication.

b. Connect to Wi-Fi using WiFi.begin(ssid,password).

c. Wait until the board is connected to Wi-Fi. When connected WiFi.status() should equate to WL_CONNECTED.

d. Print a debug statement stating that the board is now connected to the WiFi.

e. Print the local IP of the server using WiFi.localIP() .

8. In the loop function,

a. use float variables to read the temperature and humidity values from the sensor using dht.readHumidity(), dht.readTemperature(); Serial.print()/Serial.println() to print the values on the serial monitor and give an appropriate delay using delay(<delay in milliseconds>) according to the frequency of sensor.

b. An if statement inside the loop then determines whether the specified amount of time has passed and whether the client is disconnected. If so, use the function updateThingSpeak sends the string values of temperature and humidity to fields 1 and 2 respectively.

c. Finally the connection is disconnected.

9. Upload the code to your NodeMCU and wait for it to connect to the WiFi. Once connected it will display the local IP on the Serial Monitor. Open the IP on your web browser. This should open a webpage with two buttons : one for turning the LED on and one for off.

e) Expected output:

Post the temperature and humidity values to the thingspeak channel in field1 and field2 respectively.