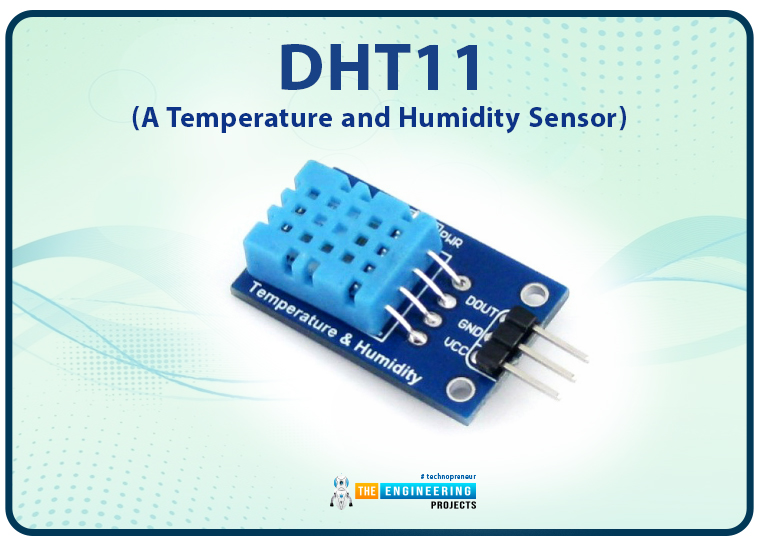
**EXPERIMENT NO: 2: PART B**

**Objectives: Design an IOT system based on ESP32 using DHT11 Interfacing and uploading Temperature and Humidity data to ThingSpeak Web Server**

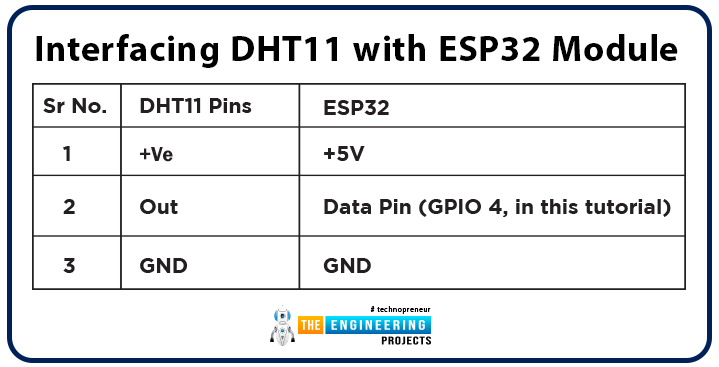
In this experiment you will learn how to interface DHT11 (temperature and humidity sensor) as shown in Figure 1 with the ESP32 and how to share the sensor readings obtained from the DHT11 sensor to a ThingSpeak web- server. The DHT11 interfacing with ESP32 as shown in Table 1 and the wire connections are shown in Figure 2

[](https://images.theengineeringprojects.com/image/webp/2022/03/1-5.jpg.webp?ssl=1)

**Figure 1: DHT11 sensor**

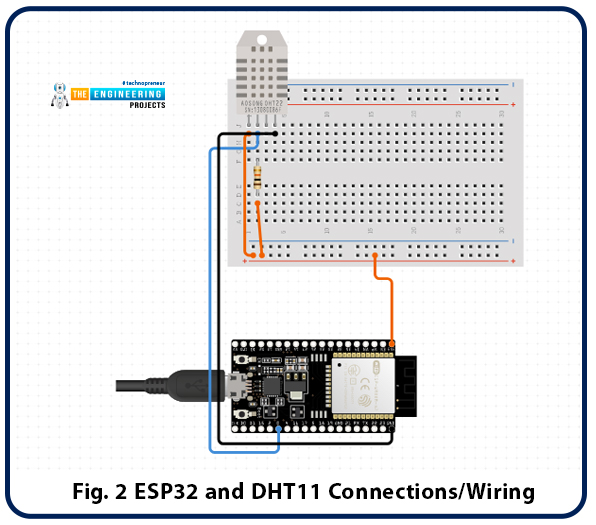
**Components required**

* ESP32 development board
* DHT11 sensor
* 10K resistor
* Connecting wires
* Breadboard

[](https://images.theengineeringprojects.com/image/webp/2022/03/2-5.jpg.webp?ssl=1)

**Table: 1**

**Note:**Connect a 10K resistor between data and power (+5V) pin of DHT11 sensor module.

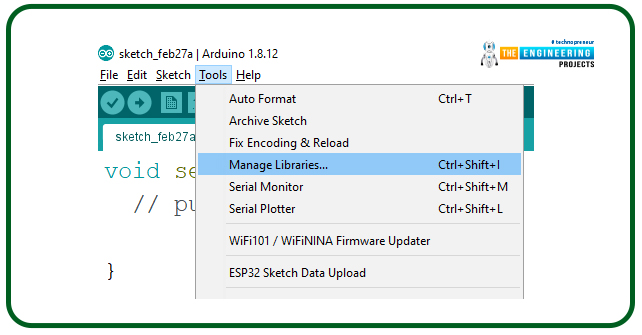
[](https://images.theengineeringprojects.com/image/webp/2022/03/3-6.jpg.webp?ssl=1)

**Figure 2: ESP32 and DHT11 connections/wiring**

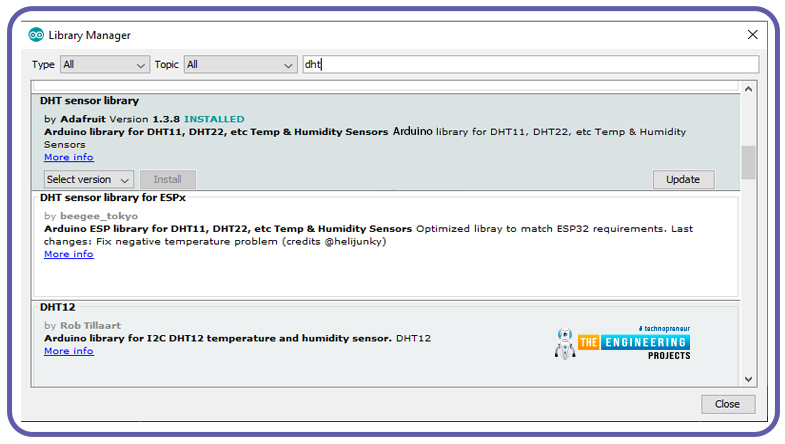
**Arduino Programming**

The Arduino IDE compile the code and upload into ESP32 module. In order to interface DHT11 sensor with ESP32 module it is required to add necessary libraries. To install the DHT11 sensor library;

* Go to **Tools >> Manage Libraries.**

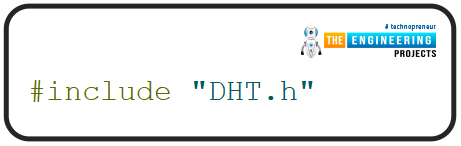
[](https://images.theengineeringprojects.com/image/webp/2022/03/4-6.jpg.webp?ssl=1)

* Type **DHT**in the search bar and install the **DHT sensor library**as shown below.

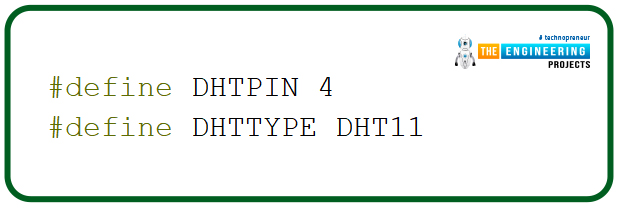
[](https://images.theengineeringprojects.com/image/webp/2022/03/5-7.jpg.webp?ssl=1)

**------e Description**

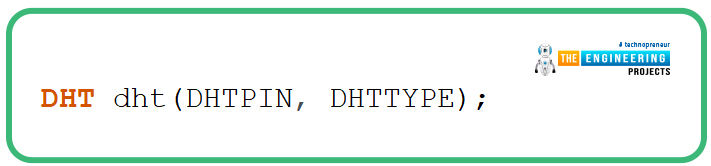
* Add the necessary header files required to interface the DHT11 sensor.

[](https://images.theengineeringprojects.com/image/webp/2022/03/6-6.jpg.webp?ssl=1)

* The next step is the declaration of variables for the DHT11 sensor, the first one is the **DHTPIN**to store the GPIO number receiving input from the DHT11 sensor and another variable is to define the type of DHT (i.e., whether DHT11 or DHT22).

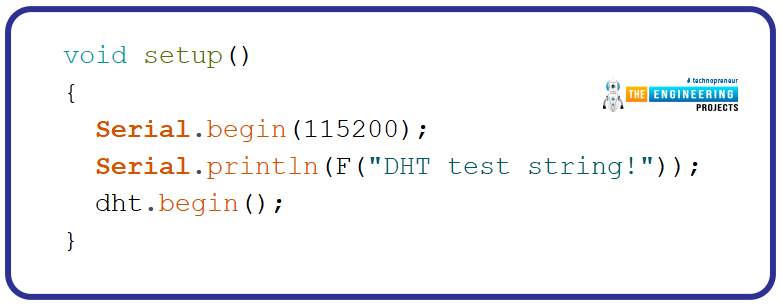
[](https://images.theengineeringprojects.com/image/webp/2022/03/7-5.jpg.webp?ssl=1)

* Next, we are creating a DHT object called **dht**in the DHT sensor type (defined earlier) and the DHT pin.

[](https://images.theengineeringprojects.com/image/webp/2022/03/8-4.jpg.webp?ssl=1)

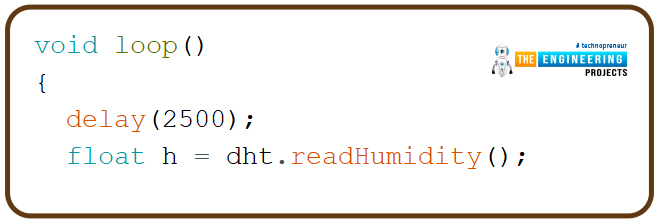
**)**

* Inside the setup function, the first task is initializing the serial monitor at a 115200 baud rate for debugging purposes.
* Initialize the DHT sensor using **begin()**function.

[](https://images.theengineeringprojects.com/image/webp/2022/03/9-5.jpg.webp?ssl=1)

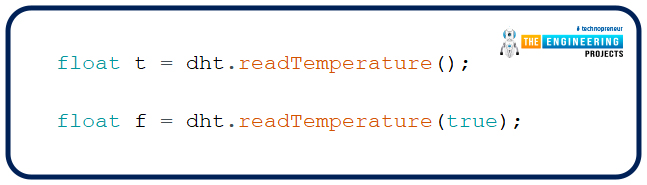
**)**

* DHT11 is a very slow sensor. It takes almost 250ms to read temperature and humidity.
* So it is preferred to wait a few seconds before a new measurement or updated sensor reading.
* Next, we are defining a float type variable **‘h’**to store humidity measured from the DHT11 sensor.
* **readHumidity()**function is used to observe the humidity value.

[](https://images.theengineeringprojects.com/image/webp/2022/03/10-4.jpg.webp?ssl=1)

**Fig. 9**

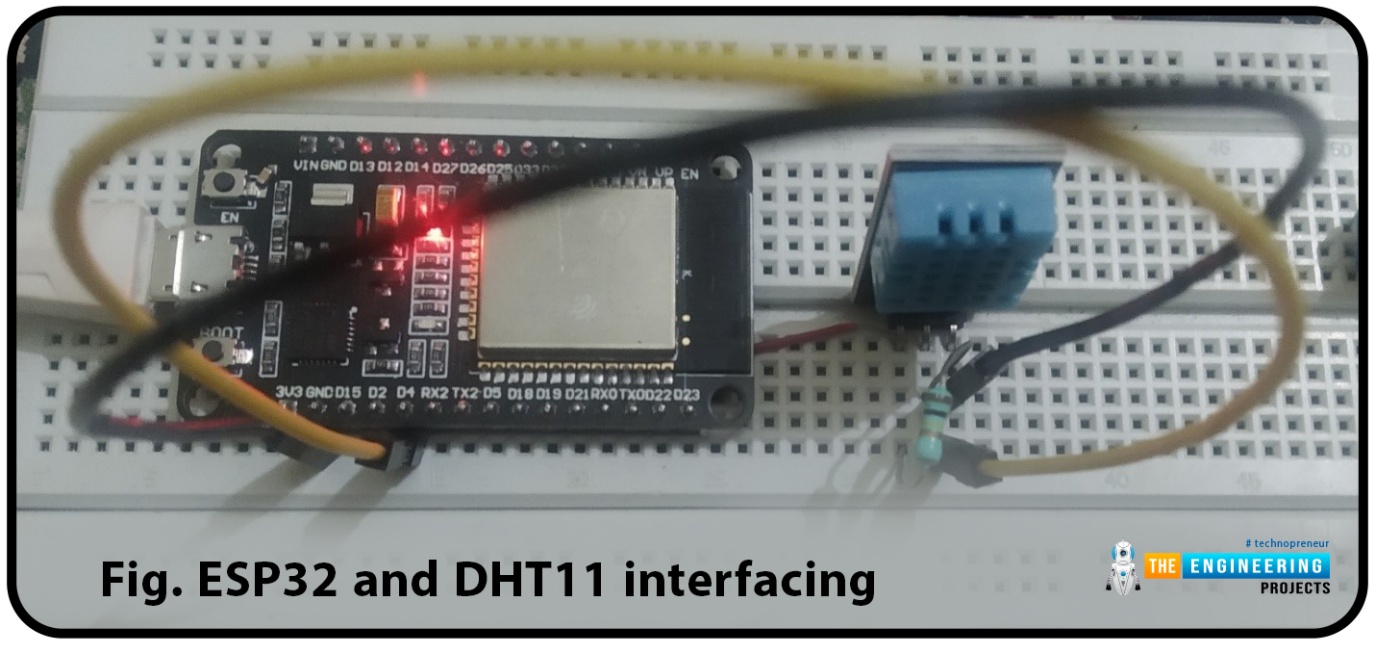
* **readTemperature()**function is used to read the surrounding temperature with DHT11 sensor.

[](https://images.theengineeringprojects.com/image/webp/2022/03/11-3.jpg.webp?ssl=1)

**Fig. 10**

**Results**

* Open the Arduino IDE and paste the above code.
* Compile and upload the program after selecting the correct development board and COM port.
* Connect the DHT11 sensor with ESP32 board as per the given circuit instructions.

[](https://images.theengineeringprojects.com/image/webp/2022/03/14-3.jpg.webp?ssl=1)

* Open the serial monitor at 115200 baud rate and press the enable (EN) button from the ESP32 development board.
* You should see the temperature, humidity readings printed on the serial monitor.

**Uploading DHT11 sensor reading to ThingSpeak Server**

The IoT is the interconnection of physical objects or devices with sensors and software accessing capabilities to communicate data or information over the internet.

To build an IoT network, we need an interface medium that can fetch, control, and communicate data between sender and receiver electronics devices or servers.

Espressif Systems created the ESP32 Wi-Fi chip series. The ESP32 module is equipped with a 32-bit Tensilica microcontroller, 2.4GHz Wi-Fi connectivity, an antenna, memory, and power management modules, and much more. All of these built-in features of this ESP32 module make it ideal for IoT applications.

**ThingSpeak web server**

It is an open data platform for the Internet of Things (Internet of Things). ThingSpeak is a MathWorks web service that allows us to send sensor readings/data to the cloud. We can also visualise and act on the data (calculate the data) sent to ThingSpeak by the devices. Data can be stored in both private and public channels.

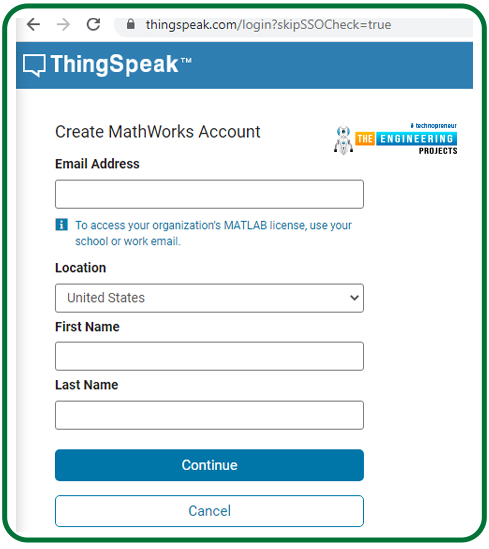
ThingSpeak is commonly used for internet of things prototyping and proof of concept systems requiring analytics.

**Getting Started with ThingSpeak**

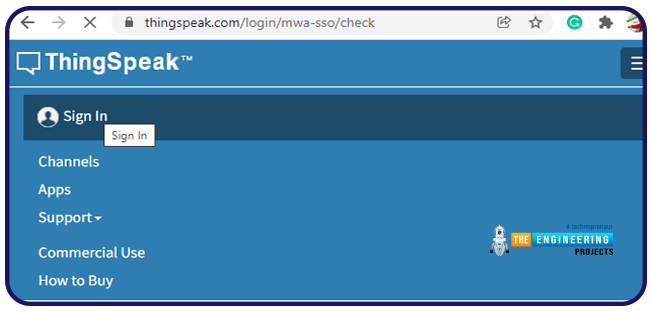
* To create and account or log in to ThingSpeak (operated by MathWorks) server follow the link: <https://thingspeak.com/>
* Click on **Get Started for free.**

[](https://images.theengineeringprojects.com/image/webp/2022/03/16-3.jpg.webp?ssl=1)

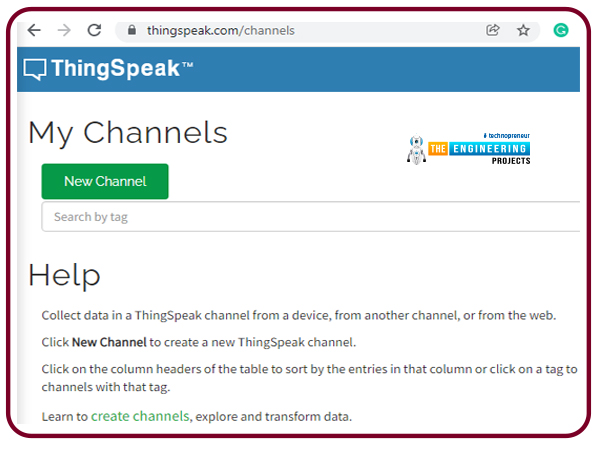
* Enter your details to create a **MathWorks** account as shown below:

[](https://images.theengineeringprojects.com/image/webp/2022/03/17-2.jpg.webp?ssl=1)

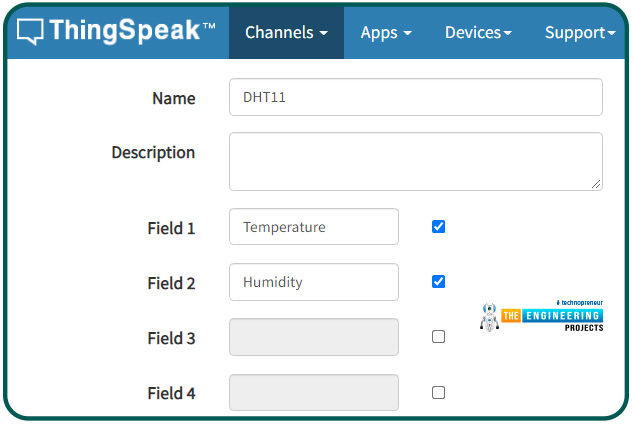
* If you have already created a MathWorks account, then click on **Sign in.**

[](https://images.theengineeringprojects.com/image/webp/2022/03/18-2.jpg.webp?ssl=1)

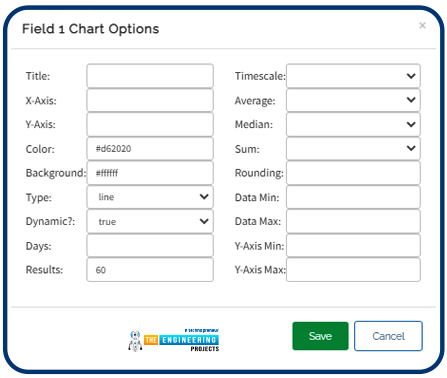
* Create a channel on MathWorks server by clicking on the **New Channel**
* ThingSpeak web service allows its user to create and save maximum of 4 channels for free.
* If you are want access to more channels then you need to make payment for that.

[](https://images.theengineeringprojects.com/image/webp/2022/03/19-2.jpg.webp?ssl=1)

* Enter the respective details in the channel.

[](https://images.theengineeringprojects.com/image/webp/2022/03/20-2.jpg.webp?ssl=1)

* Here we are creating two fields. First one represents the temperature and another one is to represent the humidity measured using DHT11 sensor. You can also add more fields as per your requirements.
* A new URL containing the channel details and channel **Stats**will open, once you have successfully created the channel. On the same page/url, API keys are available for both read and write services.
* Go to **API Keys**and copy the write API key and paste in your Arduino IDE code. So that ESP32 can send or write the DHT sensor readings to the MathWorks server.
* In **Private view**your can also customize your chart. To edit the chart, click on the icon present on the top right corner of field chart.
* Edit the details as per your requirements and click on the **Save**

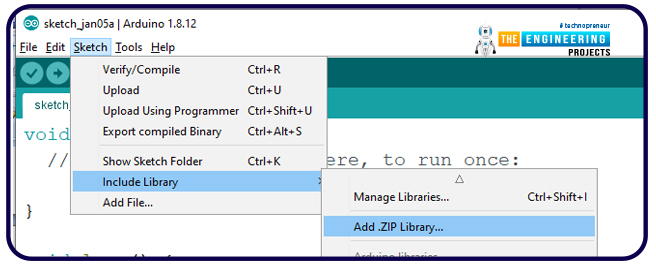
[](https://images.theengineeringprojects.com/image/webp/2022/03/21-1.jpg.webp?ssl=1)

**Arduino IDE programming**

Follow the link attached below to download the **ThingSpeak Arduino library**:

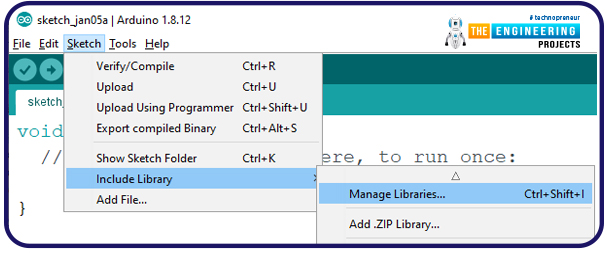
[**https://github.com/mathworks/thingspeak-arduino**](https://github.com/mathworks/thingspeak-arduino)

* Open the Arduino IDE.
* Go to **Sketch >> Include Library >> Add .ZIP Library**and select the downloaded zip file.

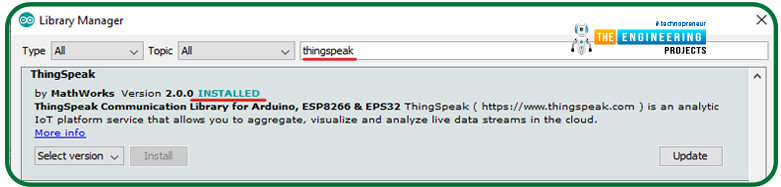
[](https://images.theengineeringprojects.com/image/webp/2022/03/22-1.jpg.webp?ssl=1)

**To check whether the library is successfully added or not:**

* Go to **Sketch >> Include Library >> Manage Libraries**

[](https://images.theengineeringprojects.com/image/webp/2022/03/23-1.jpg.webp?ssl=1)

* Type **thingspeak**in the search bar.

[](https://images.theengineeringprojects.com/image/webp/2022/03/24-1.jpg.webp?ssl=1)

* The ThingSpeak library by **MathWorks**has been successfully downloaded.

CODE FOR THE PROJECT ----------------------

//------style guard ----

#ifdef \_\_cplusplus

extern "C" {

#endif

uint8\_t temprature\_sens\_read();

#ifdef \_\_cplusplus

}

#endif

uint8\_t temprature\_sens\_read();

#include <WiFi.h>

#include <DHT.h>

#include <ThingSpeak.h>

char\* ssid = "TP-Link\_799D"; //enter SSID

char\* passphrase = "93740487"; // enter the password

WiFiServer server(80);

WiFiClient client;

unsigned long myChannelNumber = 1848106;

const char\* myWriteAPIKey = "RU40VZHVK6XGHMPP";

unsigned long lastTime = 0;

unsigned long timerDelay = 1000;

#define DHTPIN 23 // Digital pin connected to the DHT sensor

#define DHTTYPE DHT11 // DHT 11

DHT dht(DHTPIN, DHTTYPE); // Initializing the DHT11 sensor.

void setup() {

Serial.begin(115200); //Initialize serial

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, passphrase);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected.");

Serial.println("IP address: ");

Serial.println(WiFi.localIP());

server.begin();

dht.begin();

ThingSpeak.begin(client); // Initialize ThingSpeak

}

void loop() {

if ((millis() - lastTime) > timerDelay)

{

delay(2500);

// Reading temperature or humidity takes about 250 milliseconds!

float h = dht.readHumidity();

// Read temperature as Celsius (the default)

float t = dht.readTemperature();

float f = dht.readTemperature(true);

if (isnan(h) || isnan(t) || isnan(f)) {

Serial.println(F("Failed to read from DHT sensor!"));

return;

}

Serial.print("----------------------");

Serial.print("Temperature (ºC): ");

Serial.print(t);

Serial.println("ºC");

Serial.print("Humidity");

Serial.println(h);

ThingSpeak.setField(1, h);

ThingSpeak.setField(2, t);

// Write to ThingSpeak. There are up to 8 fields in a channel, allowing you to store //up to 8 different pieces of information in a channel. Here, we write to field 1.

int x = ThingSpeak.writeFields(myChannelNumber,myWriteAPIKey);

if(x == 200){

Serial.println("Channel update successful.");

}

//else{

//Serial.println("Problem updating channel. HTTP error code " + String(x));

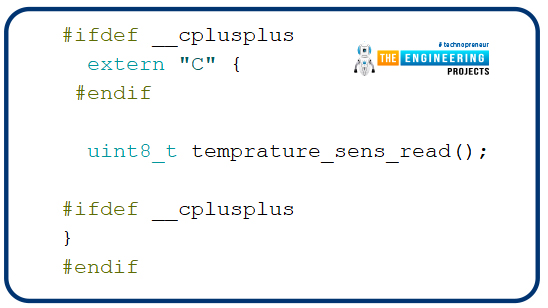
//}

lastTime = millis();

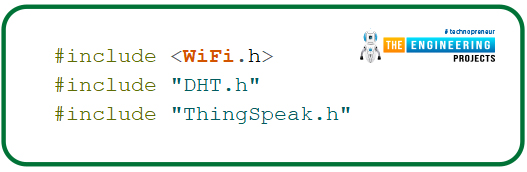
}}

**Code Description**

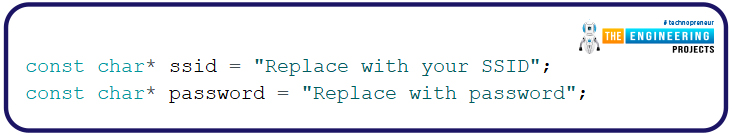
* The style guards are used at the beginning of the program to declare some function to be of **“C”** linkage, instead of **“C++”** Basically, to allow C++ code to interface with C code.

[](https://images.theengineeringprojects.com/image/webp/2022/03/25-1.jpg.webp?ssl=1)

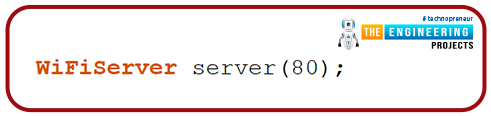
* Add the required header files. In this example we are using three libraries, Wi-Fi.h, DHT.h, ThingSpeak.
* We have already discussed above how to download and add the **DHT** and **ThingSpeak**library files to Arduino IDE.

[](https://images.theengineeringprojects.com/image/webp/2022/03/26-1.jpg.webp?ssl=1)

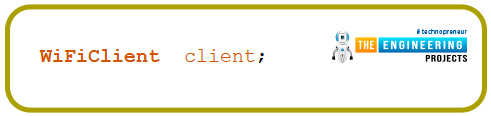
* Enter the network credentials (SSID and Password) of the access point to which your ESP device is supposed to connect for internet connectivity.

[](https://images.theengineeringprojects.com/image/webp/2022/03/27-1.jpg.webp?ssl=1)

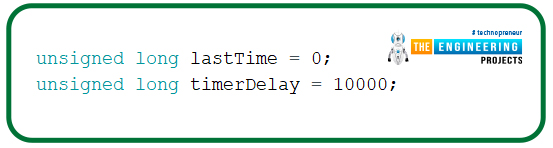
* To access the created web server we also need to assign a port and usually port 80 is used for local web server.

[](https://images.theengineeringprojects.com/image/webp/2022/03/28-1.jpg.webp?ssl=1)

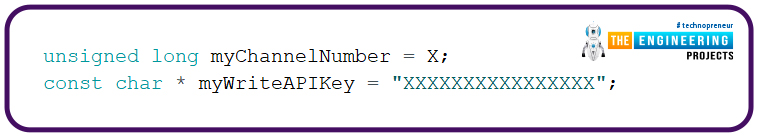
* A Wi-Fi client is created to connect with ThingSpeak.

[](https://images.theengineeringprojects.com/image/webp/2022/03/29-1.jpg.webp?ssl=1)

* Global declaration of timer variables.

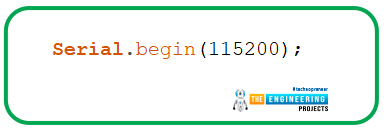
[](https://images.theengineeringprojects.com/image/webp/2022/03/30-1.jpg.webp?ssl=1)

* Add the channel number and API (Write) Key. If you have created only one channel then the channel number will be ‘1’.

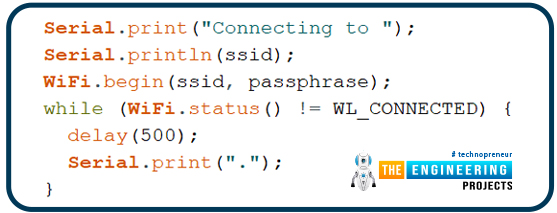
[](https://images.theengineeringprojects.com/image/webp/2022/03/31-1.jpg.webp?ssl=1)

**Setup()**

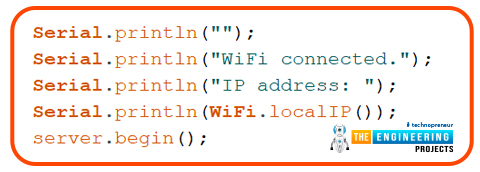
* + Initialize the Serial monitor with a 115200 baud rate for debugging purposes.

[](https://images.theengineeringprojects.com/image/webp/2022/03/32-1.jpg.webp?ssl=1)

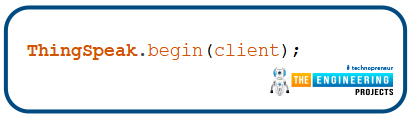
* Set ESP32 Wi-Fi module in station mode using **mode()** function.
* Enable ESP32’s Wi-Fi module using **begin()** function which is passing two arguments SSID and password.
* Wait until the ESP32 is not connected with the wifi network.

[](https://images.theengineeringprojects.com/image/webp/2022/03/33.jpg.webp?ssl=1)

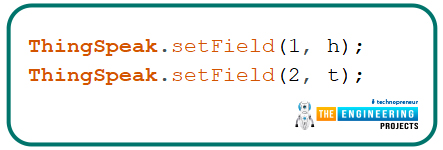
* Once ESP32 is successfully connected to Wi-Fi network, the **localIP()**function will fetch the IP address of the device.
* **begin()**function is used to initialize the server.

[](https://images.theengineeringprojects.com/image/webp/2022/03/34.jpg.webp?ssl=1)

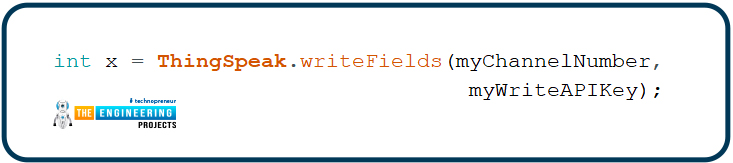
* Initialize the ThingSpeak server using **begin()**function that is passing **client**(globally created) as an argument.

[](https://images.theengineeringprojects.com/image/webp/2022/03/35.jpg.webp?ssl=1)

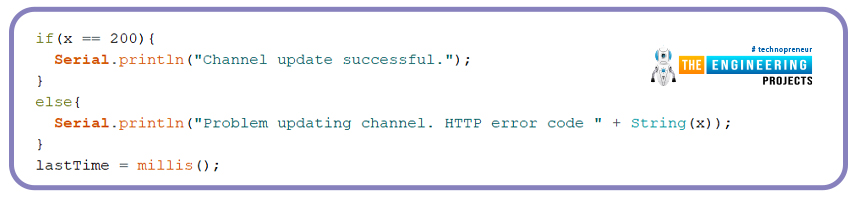
* Set the number of fields you have created to the ThingSpeak server. We are adding only two fields. First one represents the humidity measured by the sensor from its surrounding and the 2nd field represents the temperature in degree Celsius.
* You can also add further fields like for temperature in Fahrenheit, heat index etc.
* ThingSpeak allow the user to add up to maximum of 8 fields for different readings.

[](https://images.theengineeringprojects.com/image/webp/2022/03/36.jpg.webp?ssl=1)

* **writeFields()**function is used to write data to the ThingSpeak server. This function is using the channel number and API key as an argument.

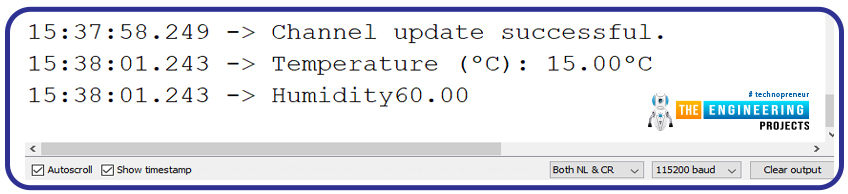
[](https://images.theengineeringprojects.com/image/webp/2022/03/37.jpg.webp?ssl=1)

* Return the code 200 if the sensor readings are successfully published to ThingSpeak server and print the respective results on the serial monitor.

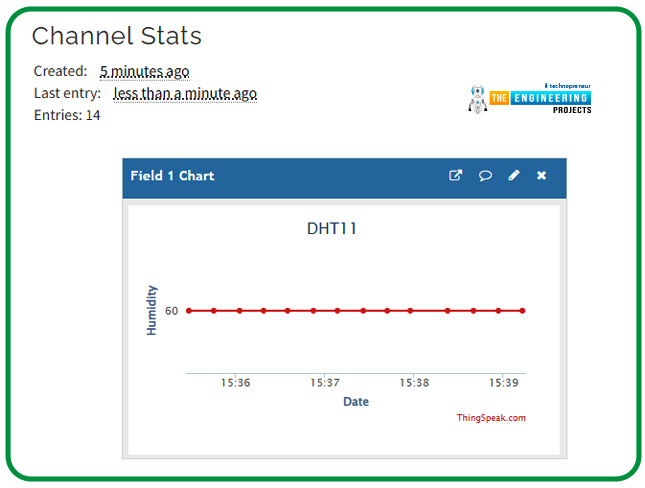
[](https://images.theengineeringprojects.com/image/webp/2022/03/38.jpg.webp?ssl=1)

**Results**

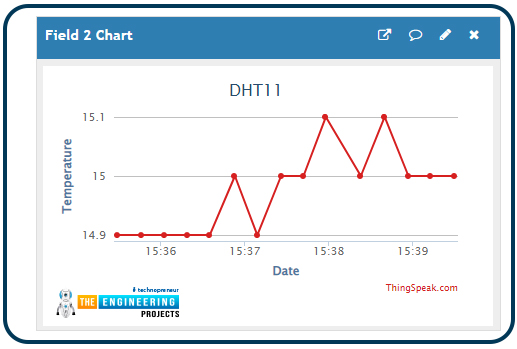
* Copy the above code and paste it into your Arduino IDE.
* Make the required changes in the above code and the changes required includes, network credentials (SSID and Password), API key, Channel number etc.
* Compile and upload the above program into ESP32 development after selecting the correct development board and COM port.
* Make sure the Access Point (Wi-Fi) is ON to which your ESP device is supposed to connect.
* Open the serial monitor at a 115200 baud rate and press the EN button from the ESP32 development board.

[](https://images.theengineeringprojects.com/image/webp/2022/03/39.jpg.webp?ssl=1)

* Open the channel you have created on the ThingSpeak server.
* You should see the charts updated with the latest temperature and humidity readings.

[](https://images.theengineeringprojects.com/image/webp/2022/03/40.jpg.webp?ssl=1)

**Figure 1: Displaying Humidity on ThingSpeak server**

[](https://images.theengineeringprojects.com/image/webp/2022/03/41.jpg.webp?ssl=1)

**Figure 2: Displaying Temperature on ThingSpeak server**

Top of Form

**Top PCB Des**