22-02-2025 WEEK-5

Aim: write a program to monitor temperature and humidity using Arduino.

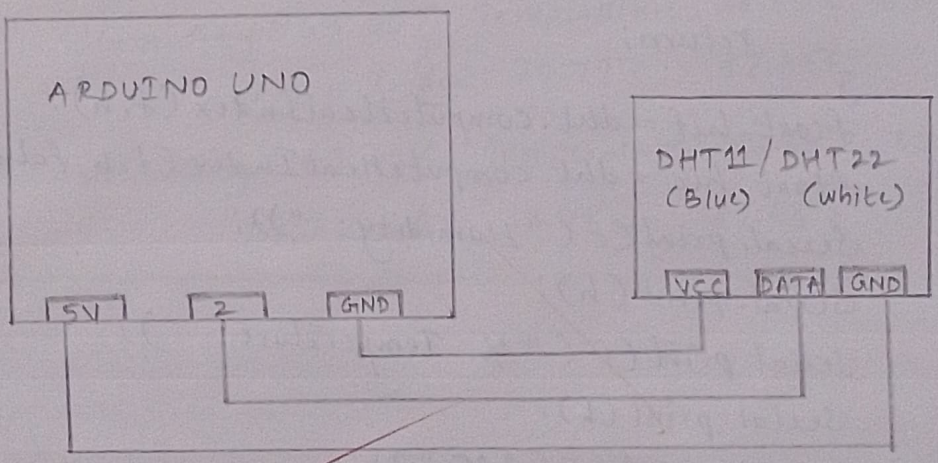
Hardware Requirements:

1. Arduino UNO board
2. DHT11 Temperature and Humidity sensor (3 Pins)
3. Jumper wires
4. Bread Board

Procedure:

- Go to tools select board (Arduino UNO)
- Connect Pin1 (on left) of the sensor to +5V.
- Connect Pin2 of the sensor to whatever your DHT11.
- Connect Pin4 (on right) of the sensor to GROUND.
- Connect a 10K resistor from Pin2 (data) to Pin1 (power) of the sensor.

Pin Diagram:



NOTE: Go to library and click on manage library → Search for DHT sensor library.

Source Code:

```
#include <DHT.h>
#define DHTPIN 8
#define DHTTYPE, DHT22
DHT dht (DHTPIN, DHTTYPE);
void setup() {
    Serial.begin(9600);
    Serial.println(F("DHT test! "));
    dht.begin();
}
void loop() {
    delay(2000);
    float h = dht.readHumidity();
    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;
    }
    float hif = dht.computeHealIndex(f, h);
    float hic = dht.computeHeatIndex(t, h, false);
    Serial.print(F("Humidity: "));
    Serial.print(h);
    Serial.print(F("% Temperature: "));
    Serial.print(t);
    Serial.print(F("°C "));
    Serial.print(f);
    Serial.print(F("°F Heat index: "));
    Serial.print(hic);
    Serial.print(F("°c"));
    Serial.print(hif);
    Serial.println(F("°F"));
}
```

NOTE:

DHT11: It is a basic, ultra low cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and splits out a digital signal on the data pin (no analog input pins needed)

Output:

DHT Test1

Humidity: 63.00% Temperature: 28.60°C 83.48°F
Heat Index: 30.79°C 87.43°F

Humidity: 63.00% Temperature: 28.60°C 83.48°F
Heat Index: 30.79°C 87.43°F

Humidity: 63.00% Temperature: 28.60°C 83.48°F
Heat Index: 30.79°C 87.43°F

Humidity: 63.00% Temperature: 28.60°C 83.48°F
Heat Index: 30.79°C 87.43°F

Humidity: 63.00% Temperature: 28.60°C 83.48°F
Heat Index: 30.79°C 87.43°F

Humidity: 63.00% Temperature: 28.60°C 83.48°F
Heat Index: 30.79°C 87.43°F

01-03-2025 WEEK-6

Aim: write a program to Interface IR sensor using Arduino using IoT cloud application.

The different cloud applications which integrate IoT are:

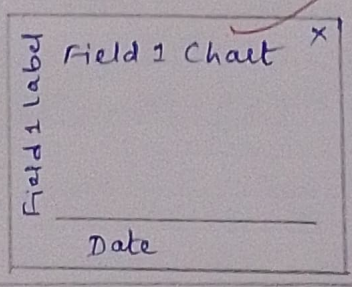
1. Thingspeak
2. Xively

1. Thingspeak: It is IoT analytic service that allows you to aggregate, visualize, analyze live data stream from cloud (in the cloud)

- MATLAB code supports thingspeak to perform online analysis and process the data.
- MATLAB also supports simulation.

- Open chrome, open thingspeak
- Create new user account
- A new account contains a new channel.
- The channel has 8 different fields
- It stores the data and integrates the data with different sensors.
- Channel settings helps us to visualize the data
- There are 2 views
  - (i) Private view
  - (ii) Public view (In Channels)
- It contains 2 API keys (read API key, write API key)
- your private data is protected with API key that you can control

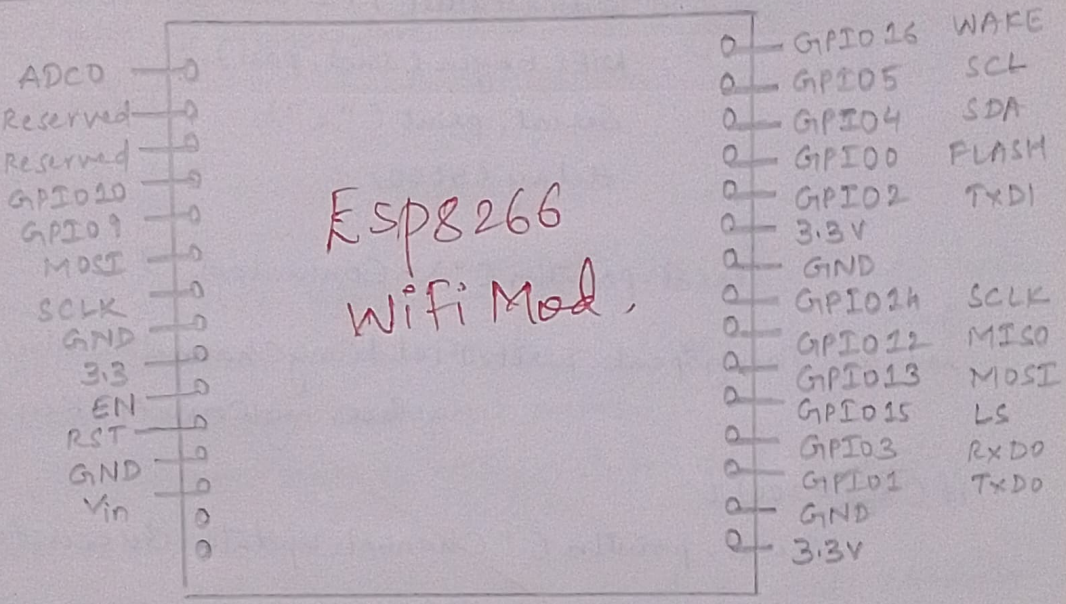
Channel ID: 246713
Author: mwa0000033348795
Access: Private
Private View: Public View Channel Settings Sharing API Keys
Channel Stats



Hardware Requirements: Node MCU is a micro-controller unit used to develop IoT applications. It connects devices. It has built in wifi and programming capability facilitating speedy prototyping and development of IoT applications.

Arduino + WiFi = NodeMCU

Node MCU Pin Diagram:



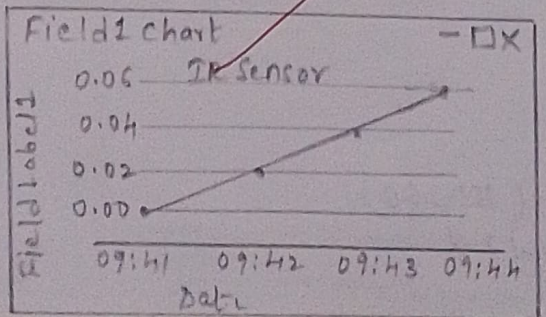
Go to Sketch, click on include. library and search in manage libraries ESP8266 and Thingspeak.

Source Code:

```
#include <ESP8266WiFi.h>
#include "secrets.h"
#include "ThingSpeak.h"
char ssid[] = SECRET_SSID;
char pass[] = SECRET_PASS;
int keyIndex = 0;
WiFiClient client;
unsigned long myChannelNumber = SECRET_CH_ID;
const char myWriteAPIKey = SECRET_WRITE_APIKEY;
int number = 0;
int IRPIN = D3;
void setup() {
    Serial.begin(115200);
    while(!Serial) {
    } ;
```

```
    WiFi.mode(WIFI_STA);
    ThingSpeak.begin(client);
}
void loop() {
    if (WiFi.status() != WL_CONNECTED) {
        Serial.print("Attempting to connect to SSID: ");
        Serial.println(SECRET_SSID);
        while (WiFi.status() != WL_CONNECTED) {
            WiFi.begin(ssid, pass);
            Serial.print(".");
            delay(5000);
        }
        Serial.println("\nConnected.");
    }
    int x = ThingSpeak.writeField(myChannelNumber, 1,
                                  number, myWriteAPIKey);
    if (x == 200) {
        Serial.println("Channel update successful.");
    }
    else {
        Serial.println("Problem updating channel.
                        HTTP error code " + String(x));
    }
    number++;
    if (number > 99) {
        number = 0;
    }
    delay(20000);
}
#define SECRET_SSID    "MySSID"
#define SECRET_PASS    "MyPassword"
#define SECRET_CH_ID   000000
#define SECRET_WRITE_APIKEY "XYZ"
```

Output:



Created: an hour ago
Last entry: 22 min ago
Entries: 4

8/3/25

08-03-2025 WEEK-7

Aim: Write a program to upload temperature and humidity to the cloud using an arduino.

Hardware Requirements

- Arduino UNO board
- NodeMCU ESP8266 Board
- DHT-11 / DHT-22 temperature and humidity sensor
- Jumper wires
- Bread board

Procedure:

1. Download esp8266 zip file → go to librarier → add zipfile
2. Connect Node MCU, Go to tools → change board to Node MCU esp8266 and port numbu.
3. Connect DHT-11 / DHT-22 temperature and humidity sensor to Node MCU.
4. Sign up to Cloud → open thingspeak > create channelr → copy API key to the source code.
5. SSID and password of your WiFi connection should be given in source code.
6. Compile and upload to program and verify the readings in serialmonitor.
7. Go to cloud and verify the temperature and humidity values in graph.

Source Code:

```
#include<DHT.h>
#include<DHT_U.h>
#include<ESP8266WiFi.h>
String apikey = "ZyVPqRYN150EBYRD";
const char *ssid = "surekha"
const char *pass = "sakhusion"
const char *server = "api.thingspeak.com";
#define DHTPIN D3
```

```
DHT dht(DHTPIN, DHT11);
WiFiClient client;
void setup() {
    Serial.begin(115200);
    delay(1000);
    dht.begin();
    Serial.println("Connecting to ");
    Serial.println(ssid);
    WiFi.begin(ssid, pass);
    while(WiFi.status() != WL_CONNECTED) {
        delay(2000);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi Connected");
}
void loop() {
    float h = dht.readHumidity();
    float t = dht.readTemperature();
    if (isnan(h) || isnan(t)) {
        Serial.println("Failed to read from DHT
                        sensor!");
        return;
    }
    if(client.connect(server, 80)){
        String postStr = apiKey;
        postStr += "&field1=";
        postStr += String(t);
        postStr += "&field2=";
        postStr += String(h);
        postStr += "\r\n\r\n";
        client.print("POST/update HTTP/1.1\n");
        client.print("Host: api.thingspeak.com\n");
        client.print("Connection: close\n");
        client.print("X_THINGSPEAKAPIKEY:" +
                     apiKey + "\n");
        Client.print("Content-Type: application
                      /x-www-form-urlencoded\n");
```

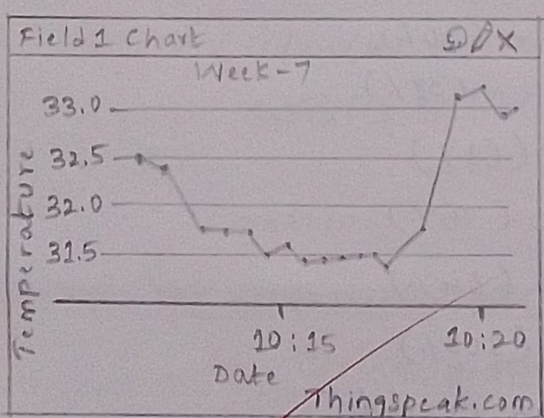
```
client.print("Content-Length: ");
client.print("\n\n");
client.print(*postStr.length());
client.print(postStr);
Serial.print("Temperature: ");
Serial.print(t);
Serial.print("degrees Celsius, Humidity");
Serial.print(h);
Serial.println("%. send to Thingspeak");
```

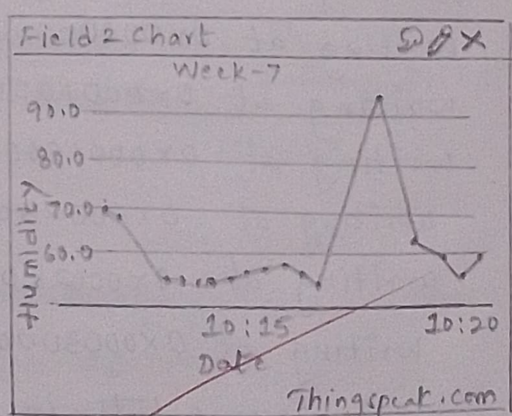
```
}
client.stop();
Serial.println("waiting...");
delay(1000);
}
```

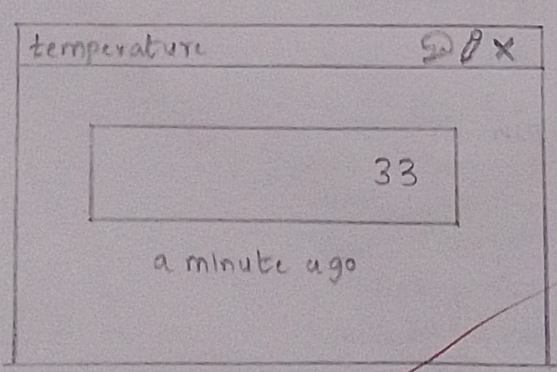
Output:

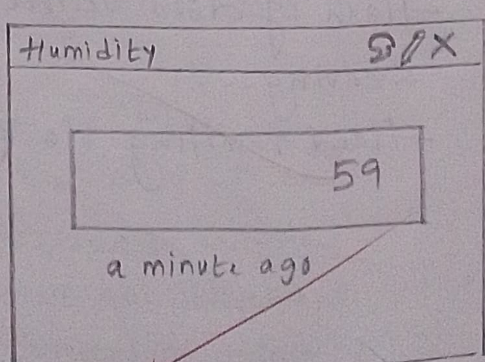
```
uploading stub...
Running stub...
stub running...
writing at 0x0000... (7%)
writing at 0x00004000... (15%)
writing at 0x00008000... (23%)
writing at 0x0000c000... (30%)
writing at 0x00010000... (38%)
writing at 0x0001c000 (61%)
writing at 0x00024000 (76%)
writing at 0x00028000 (84%)
writing at 0x00030000 (100%)
wrote 283216 bytes in 18.5 seconds
Hash of data verified
Leaving...
Hard resetting via RTS pin...
```

Temperature : 31.30°c , Humidity : 54.00% send to Thingspeak
waiting...
Temperature : 31.30°c , Humidity : 54.00% send to Thingspeak
waiting...
Temperature : 31.30°c , Humidity : 54.00% send to Thingspeak
waiting...
Temperature : 31.30°c , Humidity : 54.00% send to Thingspeak
waiting...
Temperature : 31.30°c , Humidity : 54.00% send to Thingspeak
waiting...
Temperature : 31.30°c , Humidity : 54.00% send to Thingspeak
waiting...
Temperature : 31.30°c , Humidity : 54.00% send to Thingspeak
waiting...
Temperature : 31.30°c , Humidity : 54.00% send to Thingspeak
waiting...
Temperature : 31.30°c , Humidity : 54.00% send to Thingspeak
waiting...









15-03-2025 WEEK-8

Aim: Write a program on Arduino to retrieve temperature and humidity data from cloud.

Hardware Requirements:

- Arduino UNO board
- Node MCU ESP8266 Breakout Board
- DHT-11 Temperature and humidity sensor
- Jumper wires
- Bread board

Procedure:

- Download esp8266.zip file → go to libraries → add zip file.
- Connect Node MCU, Go to Tools → change board to Node MCU esp8266 and port number.
- Connect DHT-11 / DHT-22 temperature and humidity sensor to Node MCU
- Sign up to cloud → open Thingspeak > create channels → copy API Key to the source code.
- SSID and password of your WiFi connection should be given in source code.
- Compile and upload the program and verify the readings in serialmonitor.
- Go to cloud and verify the temperature and humidity values in graph.

Source code

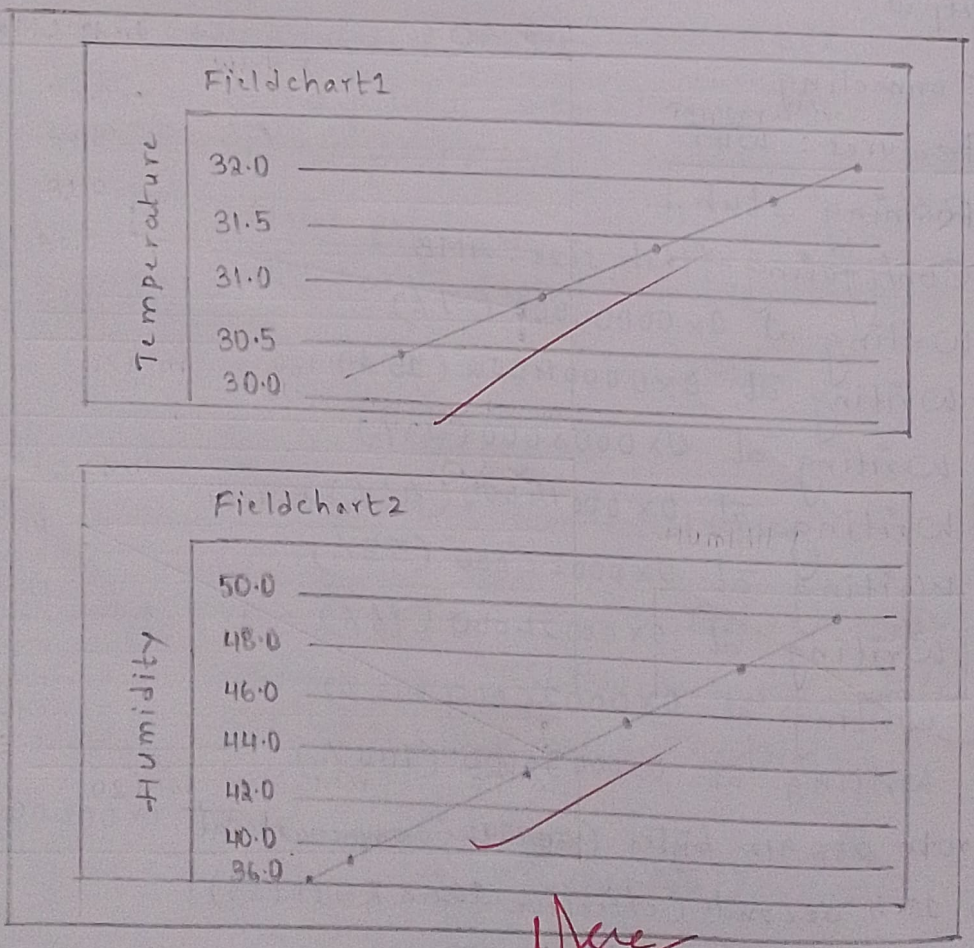
```
#include "ThingSpeak.h"
#include "ESP8266WiFi.h"
#include <DHT.h>
const char ssid[] = "sure";
const char pass[] = "abc";
int statusCode = 0;
WiFiClient client;
unsigned long counterChannelNumber = 2846266;
const char * myCounterReadAPIKey = "SXJMCVLJW1ZMZ";
const int FieldNumber1 = 1;
const int FieldNumber2 = 2;
void setup() {
    Serial.begin(115200);
    WiFi.mode(WIFI_STA);
    ThingSpeak.begin(client);
}
void loop() {
    if(WiFiStatus() != WL_CONNECTED){
        Serial.print("Connecting to ");
        Serial.print(ssid);
        Serial.println("....");
        while(WiFi.Status() != WL_CONNECTED) {
            WiFi.begin(ssid, pass);
            delay(5000);
        }
        Serial.println("Connected to WiFi Successfully");
    }
    long temp = Thingspeak.readLongField(counter
        ChannelNumber, FieldNumber1, myCounterReadAPIKey);
    statusCode = Thingspeak.getLastReadStatus();
    if(statusCode = 200) {
        Serial.print("Temperature");
        Serial.println(temp);
    }
}
```

```
else {
  Serial.println("unable to read channel / No internet connection");
}
delay(100);
long humidity = ThingSpeak.readLongField(counterChannelNumber, FieldNumber2, myCounterReadAPIKey);
statusCode = ThingSpeak.getLastReadStatus();
if(statusCode == 200){
  Serial.println("Humidity: ");
  Serial.println(humidity);
}
else {
  Serial.println("Unable to read channel / No internet connection");
}
delay(100);
}
```

Output:

```
Connecting....
Features: WiFi
Running stub...
Configuring flash size: 4MB
Writing at 0x00000000 (7%)
Writing at 0x00004000 (15%)
Writing at 0x00008000 (23%)
Writing at 0x00014000 (46%)
Writing at 0x0001C000 (53%)
Writing at 0x00024000 (76%)
Writing at 0x0002C000 (92%)
Writing at 0x00030000 (100%)
Wrote 282912 bytes (207372 compressed) at 0x00000000
in 18.4 seconds (effective 122.8 kbits/s)
Hash of data verified
Hard resetting via RTS pin.
```

Temperature : 29
Humidity : 36
Temperature : 29
Humidity : 36
Temperature : 29
Humidity : 36
Temperature : 29
Humidity : 36
Temperature : 29
Humidity : 36
Temperature : 29
Humidity : 36
Temperature : 29
Humidity : 36
Temperature : 29
Humidity : 36



Asha
22/3/25