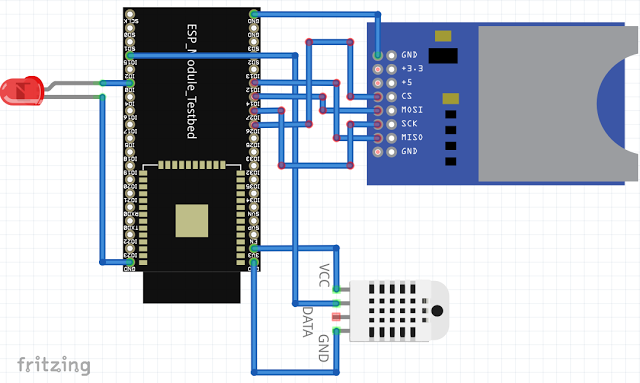
# **[Demo 13: How to display temperature/humidity using Google Chart/Jquery and control LED through Arduino ESP32 Web Server](http://www.iotsharing.com/2017/05/how-to-display-dht11-google-chart-jquery-led-control-esp32-webserver.html)**

**1. Introduction**  
In this tutorial,we will combine previous tutorial to make a simple project. This project combines previous tutorials:  
- Turn the ESP32 into Web Server ([Demo 12: How to turn the ESP32 into a Web Server](http://www.iotsharing.com/2017/05/how-to-turn-esp32-into-web-server.html" \t "http://www.iotsharing.com/2017/05/_blank))  
- Load the html chart page from Sdcard and respond to web browser ([Demo 7: How to use Arduino ESP32 to store data to sdcard](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-store-data-to-sdcard.html" \t "http://www.iotsharing.com/2017/05/_blank))  
- Measure temperature/humidity using DHT22 ([Demo 3: How to use Arduino ESP32 to read temperature/humidity from DHT11/DHT22](http://www.iotsharing.com/2017/05/how-to-arduino-esp32-dht11-dht22-temperature-humidity-sensor.html" \t "http://www.iotsharing.com/2017/05/_blank))  
- Control LED ([Demo 1: Blinky - a Hello World on Arduino ESP32](http://www.iotsharing.com/2017/05/blinky-hello-world-on-arduino-esp32.html" \o "Demo 1: Blinky - a Hello World on Arduino ESP32" \t "http://www.iotsharing.com/2017/05/_blank))  
- mDNS for HTTP service ([Demo 9: How to use mDNS to resolve host names to ESP32 IP addresses](http://www.iotsharing.com/2017/05/how-to-use-mdns-to-resolve-hostname-esp32-ipaddress.html" \t "http://www.iotsharing.com/2017/05/_blank))   
So when browser access **"http://esp32.local"**a chart of temperature/humidity will occur along with a button name **"LED"** to control LED on/off. This button have green color when led is on and red color when led is off.  
**2. Hardware**  
You need modules below:  
- A module micrSD with memory card  
- A DHT22 sensor  
- A LED  
- ESP32 board  
Connections:  
Connect VCC and GND of microSD (VCC=5V), DHT22 (VCC=3.3V) to VCC (5V-3.3V) and GND of ESP32.  
[ESP32 IO15 - DHT22 DATA]  
[ESP32 IO2 - LED ANODE]  
[ESP32 GND - LED CATHODE]  
[ESP32 IO26 - microSD CS]  
[ESP32 IO27 - microSD SCK]  
[ESP32 IO14 - microSD MOSI]  
[ESP32 IO12 - microSD MISO]

[](https://1.bp.blogspot.com/-Tlv2_AA2g0o/WSMGl9clAUI/AAAAAAAAD6M/6CUY0k4LGSQpPUgSNpi6dz2nMC2pEyMKQCLcB/s1600/esp32webser-chart-dht2-led.png)

**Figure: ESP32 connect to microSD, DHT22, LED**

**3. Software**

- Create a **chrt.html** file in memory card at **root "/"**with content:

|  |
| --- |
| <!DOCTYPE html>  <head>  <meta charset="utf-8" />  <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>  <script type="text/javascript" src="https://www.google.com/jsapi?autoload={'modules':[{'name':'visualization','version':'1.1','packages':['corechart','gauge','line','imagelinechart']}]}"></script>  <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>  <style>  .btn {  background-color: #4CAF50;  border: none;  color: white;  padding: 15px 32px;  text-align: center;  text-decoration: none;  display: inline-block;  font-size: 16px;  margin: 4px 2px;  cursor: pointer;  }  .red {background-color: #f44336;}  </style>  </head>  <body>  <div align="center"><h1>IoTSharing.com Chart Demo</h1></div>  <div id="htchart"></div>  <p><button id="led" class="btn red">LED</button></p>  <script>  google.charts.setOnLoadCallback(draw);  var data, chart;  var cnt = 0;  var points = 14;  var got = 0;  var led = 0;  var opt = {  hAxis: {  title:'Time'  },  vAxis: {  title:'Temp/Hum'  }  };  function draw(){  data = new google.visualization.DataTable();  chart = new google.visualization.LineChart(document.getElementById('htchart'));  data.addColumn('string','Time');  data.addColumn('number','temp');  data.addColumn('number','hum');  data.addRows([  [' ', 0, 0]  ]);  chart.draw(data, opt);  }  $("#led").click(function() {  if(led == 0){  $.get("/led1", function(d, s){  led = 1;  $("#led").removeClass("btn red");$("#led").addClass("btn");  });  }else{led=0;  $.get("/led0", function(d, s){  led = 0;  $("#led").removeClass("btn");$("#led").addClass("btn red");  });  }  });  function update() {  if(got == 0){  got = 1;  $.get("/dht22", function(j, s){  cnt ++;  var cur = new Date();  data.addRows([[cur.getMinutes() + ":" + cur.getSeconds(), j.temp, j.hum]]);  if(cnt == points){  for (i = 0; i < points/2; i++) {  data.removeRow(i);  }  cnt = i;  }  got = 0;  chart.draw(data, opt);  });  }else{  got ++;  if(got == points/3){  got = 0;  }  }  }  setInterval(update, 1000);  </script>  </body>  </html> |

The html code using **Google chart API** for drawing the values of temperature/humidity under l**ine chart style** and Jquery to query the value of DHT22 (through update() function every 1 second) "setInterval(update, 1000)".In the update() function we use Jquery "$.get("/dht22", function(j, s)" to query "/dht22" temperature/humidity. We also create a "led" button and process "click" action when user press it through "$("#led").click(function()". In click processing function we check if current status of LED is 0 we create a Jquery to request the LED on through "$.get("/led1", function(d, s)" and vice versa. About Google chart code, whenever we got new data from DHT22 we add it into data table and re-draw the chart.

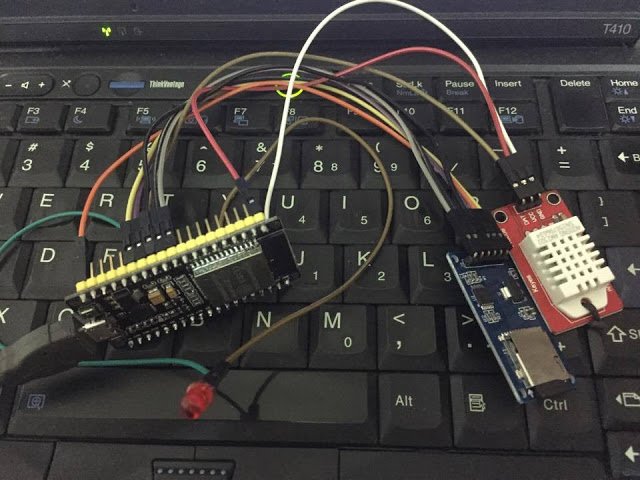
- In this project we use Json style to respond the dht22 request from client. So we download Json library at: [https://github.com/bblanchon/ArduinoJson.git](https://github.com/bblanchon/ArduinoJson.git" \t "http://www.iotsharing.com/2017/05/_blank). The unzipping the downloaded folder and copy it to Arduino/libraries.

- Then we create an **Arduino** project and Save as **esp32chart** with code:

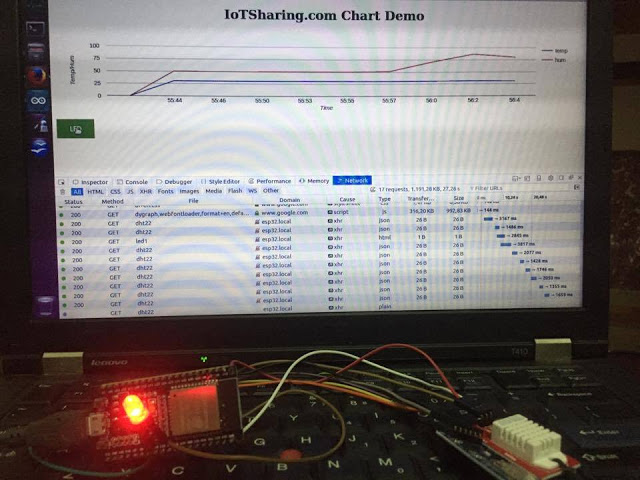
**Note:** I explain the code by detail comments in the code, so I will not mention more here.

|  |
| --- |
| #include <WiFiClient.h>  #include <ESP32WebServer.h>  #include <WiFi.h>  #include <ESPmDNS.h>  #include "DHT.h"  #include <ArduinoJson.h>  #include <SPI.h>  #include <mySD.h>  #define DHTPIN 15  #define DHTTYPE DHT22  DHT dht(DHTPIN, DHTTYPE);  const char\* ssid = "dd-wrt";  const char\* password = "0000000000";  ESP32WebServer server(80);  float humidity = 0;  float temperature = 0;  const int led = 2;  void handleRoot() {  /\* we load the chart.html from microSD \*/  File myFile = SD.open("CHRT~1.HTM");  if (myFile) {  /\* respond the content of file to client by calling streamFile()\*/  size\_t sent = server.streamFile(myFile, "text/html");  /\* close the file \*/  myFile.close();  } else {  Serial.println("error opening test.txt");  }  }  void handleLed1() {  digitalWrite(led, 1);  server.send(200, "text/html", "1");  }  void handleLed0() {  digitalWrite(led, 0);  server.send(200, "text/html", "0");  }  void handleDHT(){  /\* read the data from dht22 sensor \*/  humidity = dht.readHumidity();  temperature = dht.readTemperature();  StaticJsonBuffer<100> jsonBuffer;  JsonObject& root = jsonBuffer.createObject();  /\* failed to read the data then return HTTP status code 503 sensor failed \*/  if (isnan(humidity) || isnan(temperature)) {  server.send(503, "text/html", "sensor failed");  } else {  /\* we use Json "{"hum":#value, "temp": #value}" to respond the /dht22 request \*/  root["hum"] = humidity;  root["temp"] = temperature;  char jsonChar[100];  root.printTo((char\*)jsonChar, root.measureLength() + 1);  /\* we respond Json style so content-type is text/json \*/  server.send(200, "text/json", jsonChar);  }  }  /\* cannot handle request so return 404 \*/  void handleNotFound(){  digitalWrite(led, 1);  String message = "File Not Found\n\n";  server.send(404, "text/plain", message);  digitalWrite(led, 0);  }  void setup(void){  pinMode(led, OUTPUT);  digitalWrite(led, 0);  Serial.begin(115200);  WiFi.begin(ssid, password);  Serial.println("");  // Wait for connection  while (WiFi.status() != WL\_CONNECTED) {  delay(500);  Serial.print(".");  }  Serial.println("");  Serial.print("Connected to ");  Serial.println(ssid);  Serial.print("IP address: ");  Serial.println(WiFi.localIP());  if (MDNS.begin("esp32")) {  Serial.println("MDNS responder started");  }  /\* register the callbacks to process client request \*/  /\* root request we will read the memory card to get  the content of chrt.html and respond that content to client \*/  server.on("/", handleRoot);  /\* this callback handle /dht22 request,  read the sensor form json string and respond\*/  server.on("/dht22", handleDHT);  /\* this callback handle /led1 request, just turn on led \*/  server.on("/led1", handleLed1);  /\* this callback handle /led0 request, just turn off led \*/  server.on("/led0", handleLed0);  server.onNotFound(handleNotFound);  server.begin();  dht.begin();  Serial.println("HTTP server started");  Serial.print("Initializing SD card...");  /\* initialize microSD \*/  if (!SD.begin(26, 14, 12, 27)) {  Serial.println("initialization failed!");  return;  }  Serial.println("initialization done.");  }  void loop(void){  server.handleClient();  } |

**4.Result**

[](https://4.bp.blogspot.com/-uxd33jqxBQ8/WSMP3snFIGI/AAAAAAAAD6g/pM8i3-8pJYcV-zAC88ryYGB_lCGddpPAwCLcB/s1600/esp32-dht22-webserver-led-chart-jquery.jpg)

**Figure: esp32 webserver + dht22 + led + google chart + jquery**

**[](https://1.bp.blogspot.com/-50jYnRGq910/WSMP3RU2pwI/AAAAAAAAD6c/V1TzSuAJ7P4xvSzLQrE3MBwAsfdgR_OfQCEw/s1600/esp32-dht22-webserver-led-chart-jquery-demo.jpg)  
 Figure: esp32 webserver + dht22 + led + google chart + jquery demo**