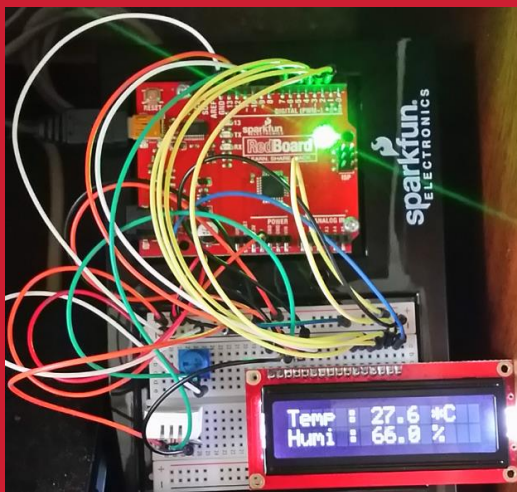
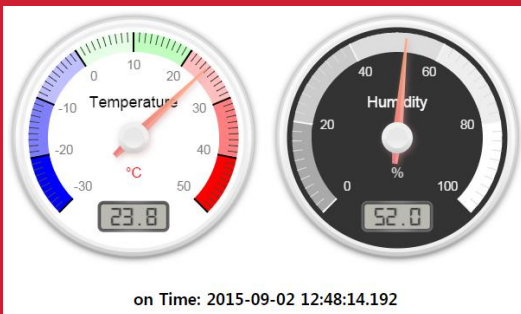




HW-SW-Connectivity

[wk15]

Arduino & NodeJS Project

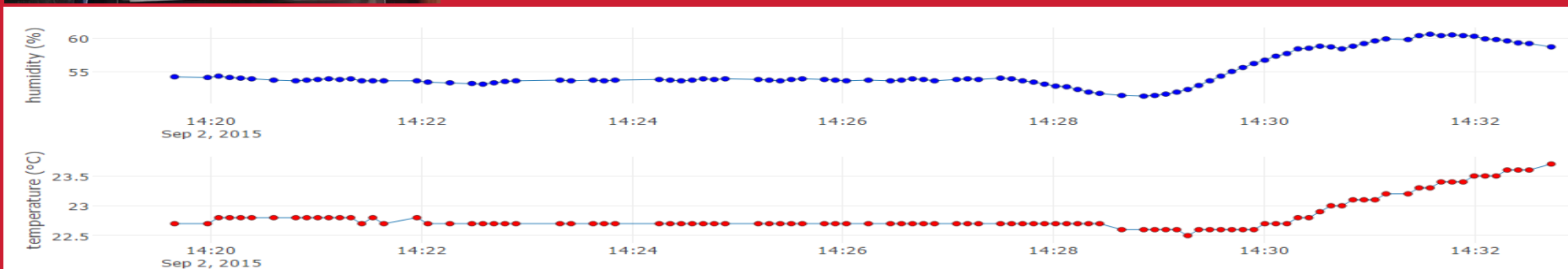


Basic HW and SW Integration using
Arduino & Javascript

COMSI, INJE University

2nd semester, 2017

Email : yish@inje.ac.kr



[DIY]

Real-time

**TMP36 & CdS monitoring
using plot.ly**

→ Data streaming



[DIY] RT sensor-data streaming in Arduino

[5.1] WEB client: [client_tmp36_ldr_update.html](#)
(using plotly streaming without nextPt())

Comment function `nextPt()`

```
// function nextPt() {  
  
  //   xArray.shift();  
  //   xArray.push(dtda[0]);  
  
  //   xTrack.shift();  
  //   xTrack.push(dtda[1]); // sensor 1:temp  
  //   yTrack.shift();  
  //   yTrack.push(dtda[2]); // sensor 2:Lux  
  
  //   Plotly.redraw(streamPlot);  
  // }
```



[DIY] RT sensor-data streaming in Arduino

[5.2] WEB client: [client_tmp36_ldr_update.html](#) (using plotly streaming without nextPt())

```
//nextPt();  
xArray = xArray.concat(dtda[0])  
xArray.splice(0, 1) // remove the oldest data  
xTrack = xTrack.concat(dtda[1])  
xTrack.splice(0, 1) // remove the oldest data
```

```
var update = {  
  x: [xArray],  
  y: [xTrack]  
}
```

Complete this part of the code.

```
Plotly.update(streamPlot, update);
```

```
// function nextPt() {  
  
//   xArray.shift();  
//   xArray.push(dtda[0]);  
  
//   xTrack.shift();  
//   xTrack.push(dtda[1]); // sensor 1:temp  
//   yTrack.shift();  
//   yTrack.push(dtda[2]); // sensor 2:lux  
  
//   Plotly.redraw(streamPlot);  
// }
```

**Save the complete
code as
[AAnn_update.html](#)**



[DIY] RT sensor-data streaming in Arduino

[5.2] WEB client: [client_tmp36_ldr_update.html](#) (using plotly streaming without nextPt())

Stupid

```

if (dtdata[1] != preX || dtdata[2] != preY) {
  preX = dtdata[1];
  preY = dtdata[2];

  gauge_temp.setValue(dtdata[1]);
  gauge_lux.setValue(dtdata[2]);
  ctime.innerHTML = dtdata[0];
  // nextPt();
  xArray = xArray.concat(dtdata[0])
  xArray.splice(0, 1)
  xTrack = xTrack.concat(dtdata[1])
  xTrack.splice(0, 1)
  yTrack = yTrack.concat(dtdata[2])
  yTrack.splice(1, 2)

  var update = {
    x: [xArray],
    y: [xTrack],
    z: [yTrack]
  }

  Plotly.update(streamPlot, update);
}

```

Bug?

```

if (dtdata[1] != preX || dtdata[2] != preY) {
  preX = dtdata[1];
  preY = dtdata[2];

  gauge_temp.setValue(dtdata[1]);
  gauge_lux.setValue(dtdata[2]);
  ctime.innerHTML = dtdata[0];
  //nextPt();

  xArray = xArray.concat(dtdata[0])
  xArray.splice(0,1)
  xTrack = xTrack.concat(dtdata[1])
  xTrack.splice(0,1)
  yTrack = yTrack.concat(dtdata[2])
  yTrack.splice(0,1)

  var update = {
    x: [xArray],
    y: [xTrack,yTrack]
  }

  Plotly.update(streamPlot,update);
}

```

Great

```

if (dtdata[1] != preX || dtdata[2] != preY) {
  preX = dtdata[1];
  preY = dtdata[2];

  // when new data is coming, keep on st
  ctime.innerHTML = dtdata[0];
  gauge_temp.setValue(dtdata[1]) // temp
  gauge_lux.setValue(dtdata[2]); // Lux g
  //nextPt();
  xArray = xArray.concat(dtdata[0])
  xArray.splice(0, 1) // remove the old
  xTrack = xTrack.concat(dtdata[1])
  xTrack.splice(0, 1) // remove the old
  yTrack = yTrack.concat(dtdata[2])
  yTrack.splice(0, 1)

  var update = {
    x: [xArray, xArray],
    y: [xTrack, yTrack]
  }

  Plotly.update(streamPlot, update);
}

```



Arduino



Home

Buy

Download

Products ▼

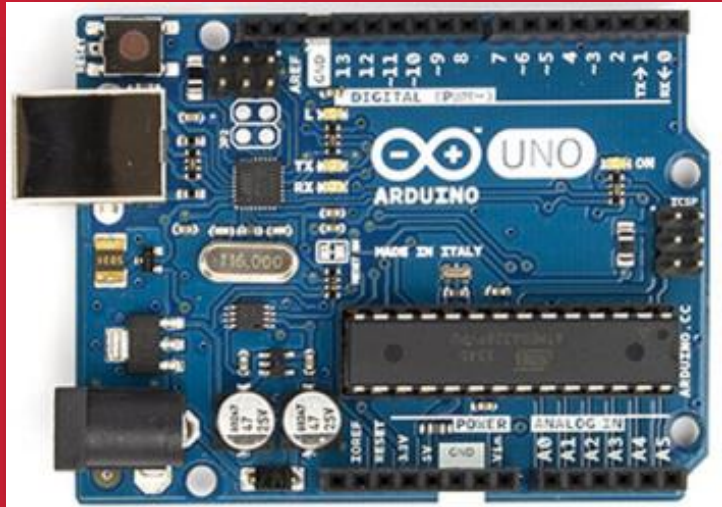
Learning ▼

Forum

Support ▼

Blog

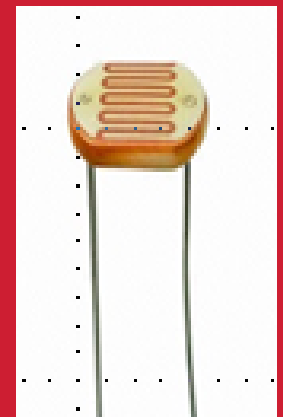
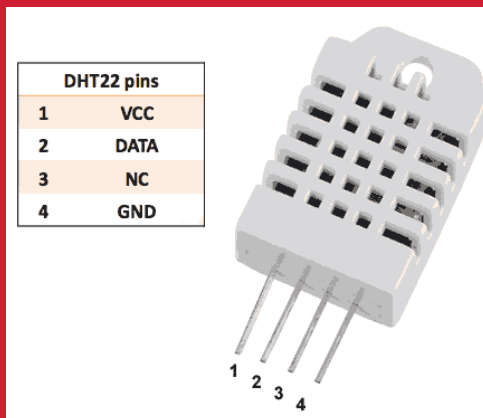
<https://www.arduino.cc/>



Arduino & Node.js

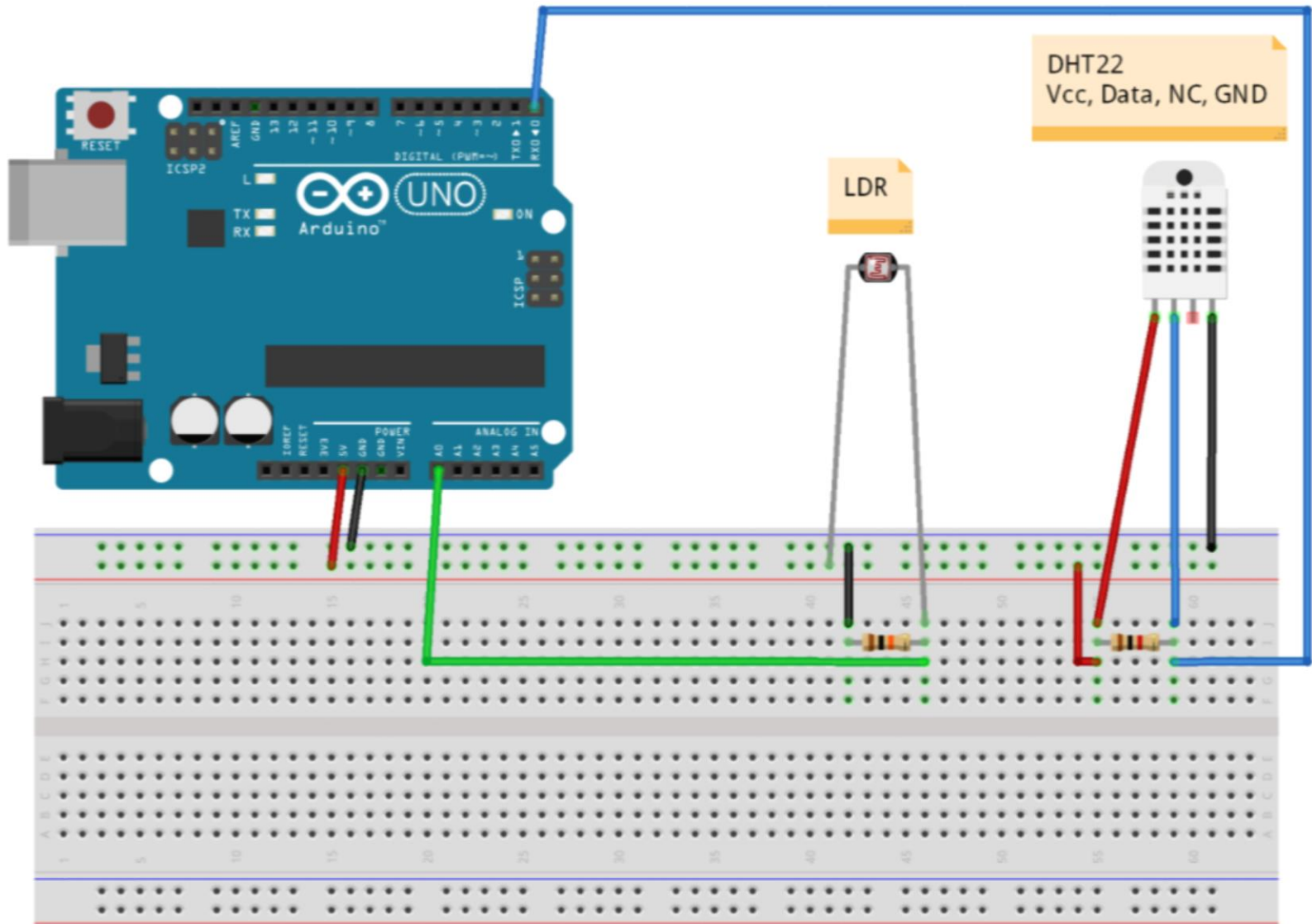
Multi-sensors

DHT22 + CdS



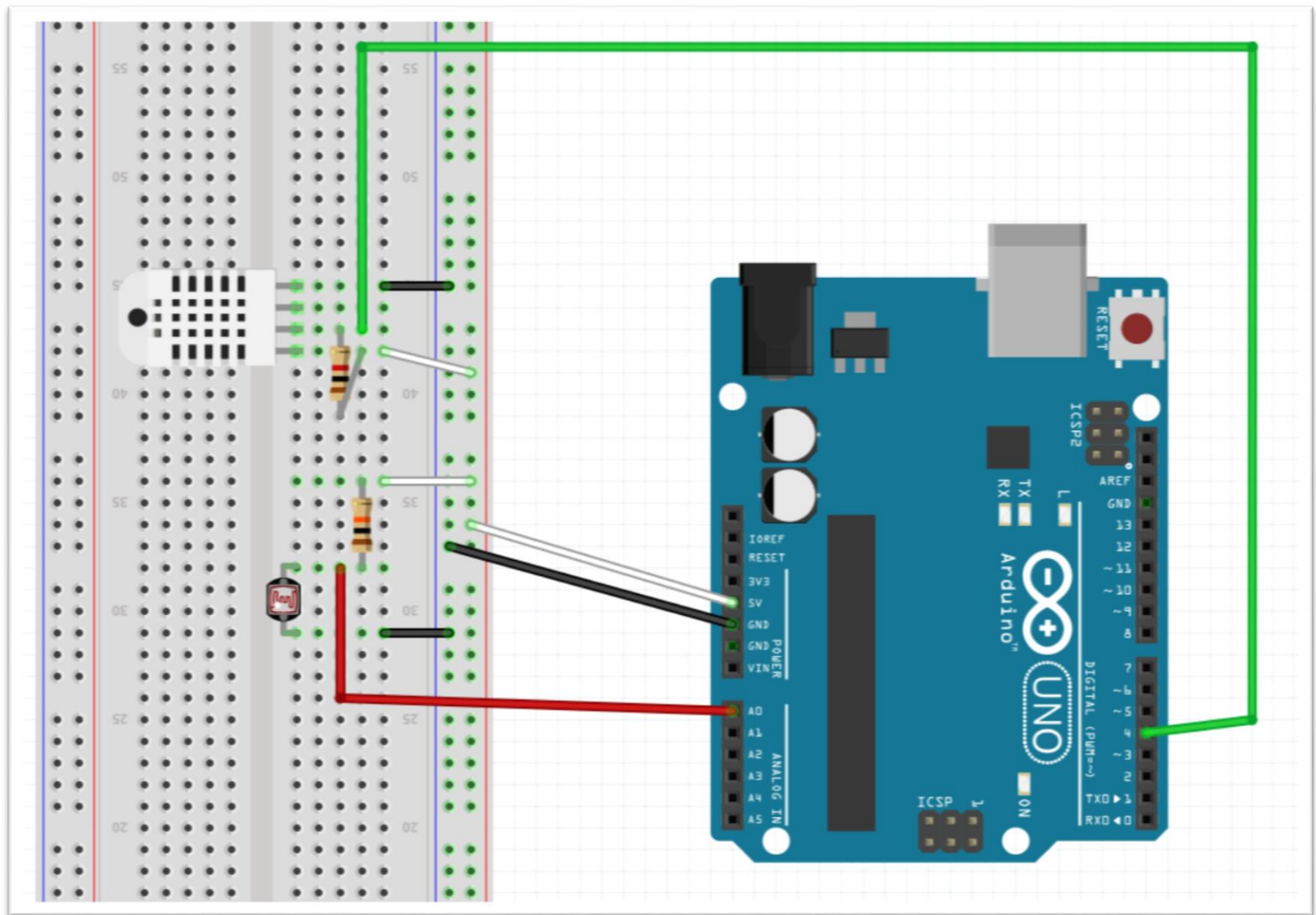


A7.1 DHT22 + CdS : circuit (Google)






A7.1 DHT22 + CdS : circuit





A7.1 DHT22 + CdS : DHT library

 Features Business Explore Marketplace Pricing This repository

adafruit / DHT-sensor-library

<> Code

! Issues 21

🔗 Pull requests 15

📁 Projects 0

📖 Wiki

📊 Insights


Arduino library for DHT11DHT22, etc Temp & Humidity Sensors [http://www.ladyada.net/lea](http://www.ladyada.net/learn/arduino/libraries/dht.html)

📄 54 commits

🌿 1 branch

🔑 8 releases

Branch: master ▾ New pull request

 microbuilder Merged unified and raw libraries

📁 .github	Add GitHub issue template
📁 examples	Merged unified and raw libraries
📄 DHT.cpp	Fix #44 by conditionally excluding unused port and bitmask state on n...
📄 DHT.h	Fix #44 by conditionally excluding unused port and bitmask state on n...
📄 DHT_U.cpp	Merged unified and raw libraries
📄 DHT_U.h	Merged unified and raw libraries



A7.1.1 DHT22 + CdS : circuit

[1] Arduino code: [AAnn_DHT22_CdS.ino](#)

AAnn_DHT22_CdS

```
1 // DHT22
2 #include "DHT.h"
3 #define DHTPIN 4
4 #define DHTTYPE DHT22
5 DHT dht(DHTPIN, DHTTYPE);
6 // CdS (LDR)
7 #define CDS_INPUT 0
8
9 void setup() {
10   dht.begin();
11   Serial.begin(9600);
12 }
```

```
42 //Voltage to LuxLux
43 double luminosity (int RawADC0){
44   double Yout=RawADC0*0.0048828125; // 5/1
45   int lux=(2500/Yout-500)/10;
46   // lux = 500 / Rldr, Yout = Ildr*Rldr = (
47   return lux;
48 }
```

```
14 void loop() {
15   int cds_value, lux;
16   float temp, humi;
17   // Lux from CdS (LDR)
18   cds_value = analogRead(CDS_INPUT);
19   lux = int(luminosity(cds_value));
20   // Reading temperature or humidity takes a given interval!
21   // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
22   humi = dht.readHumidity();
23   // Read temperature as Celsius (the default)
24   temp = dht.readTemperature();
25
26   // Check if any reads failed and exit early (to try again).
27   if (isnan(humi) || isnan(temp) || isnan(lux)) {
28     Serial.println("Failed to read from DHT sensor or CdS!");
29     return;
30   }
31   else {
32     Serial.print("AAnn,");
33     Serial.print(temp,1); // temperature
34     Serial.print(",");
35     Serial.print(humi,1); // humidity
36     Serial.print(",");
37     Serial.println(lux); // luminosity
38   }
39   delay(2000); // 2000 msec, a data per 6 min = 6 * 60 * 1000 = 360000
40 }
```

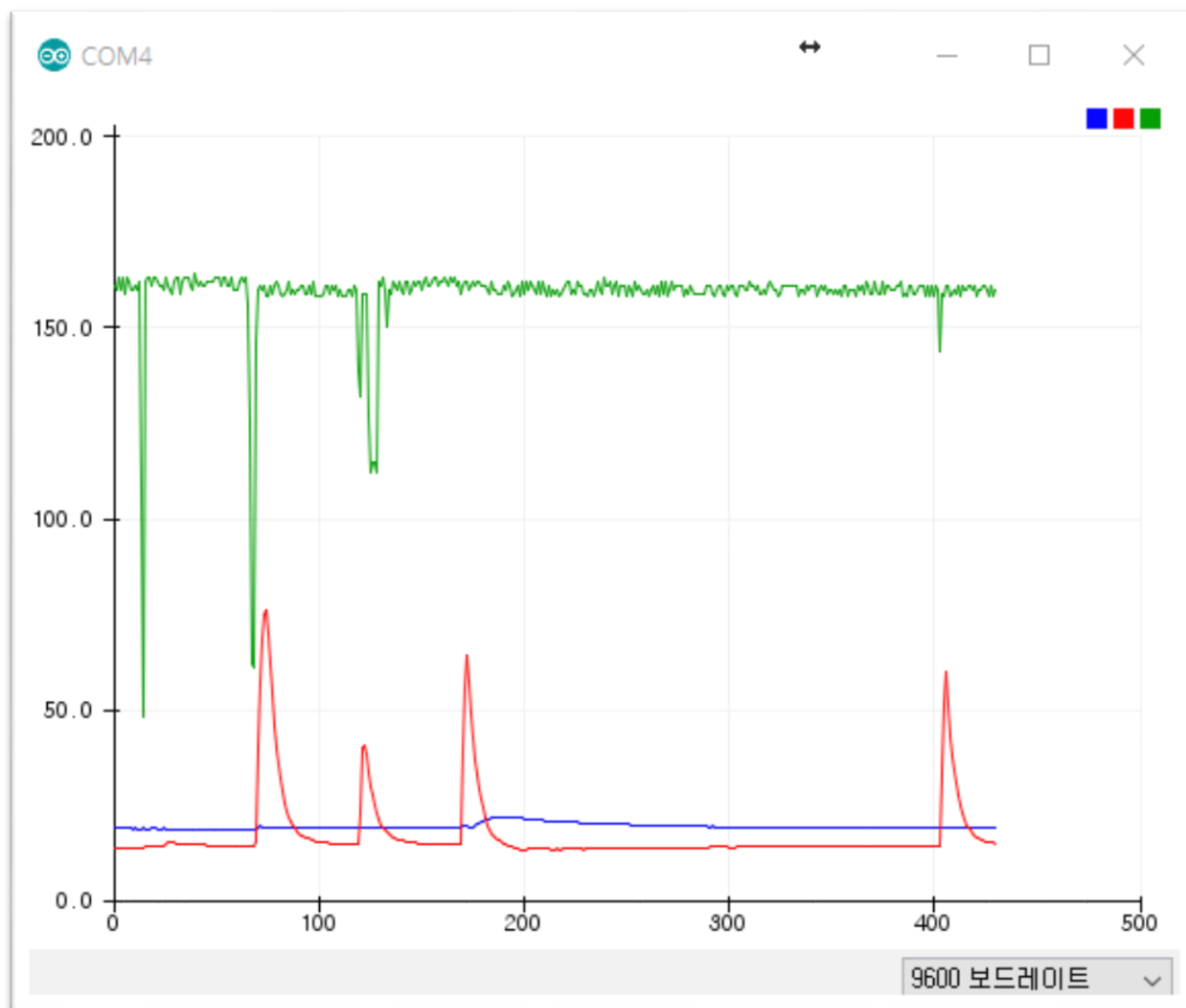


A7.1.2 DHT22 + CdS : Serial monitor

[1] Arduino code: [AAnn_DHT22_CdS.ino](#)

COM4

```
AAnn,21.5,12.1,156  
AAnn,21.5,12.2,158  
AAnn,21.5,12.3,158  
AAnn,21.4,12.3,156  
AAnn,21.4,12.3,157  
AAnn,21.3,12.4,157  
AAnn,21.3,12.5,113  
AAnn,21.3,12.6,41  
AAnn,21.2,12.7,157  
AAnn,21.2,12.7,158  
AAnn,21.2,12.7,157  
AAnn,21.1,12.7,157  
AAnn,21.0,12.6,158  
AAnn,21.0,12.6,158  
AAnn,21.0,12.6,157
```





A7.2.1 DHT22 + CdS + Node.js

[2.1] NodeJS code: [dht22_ldr_node.js](#) (← [tmp36_ldr_node.js](#))

```
1  // tmp36_ldr_node.js
2
3  var serialport = require('serialport');
4  var portName = 'COM4'; // check your COM port!!
5  var port      = process.env.PORT || 3000;
6
7  var io = require('socket.io').listen(port);
8
9  // serial port object
10 var sp = new serialport(portName,{
11     baudRate: 9600, // 9600 38400
12     dataBits: 8,
13     parity: 'none',
14     stopBits: 1,
15     flowControl: false,
16     parser: serialport.parsers.readline('\r\n')
17 });
```



A7.2.2 DHT22 + CdS + Node.js

[2.2] NodeJS code: dht22_ldr_node.js (Complete your parser code)

```
19 var readData = ''; // this stores the buffer
20 var temp = '';
21 var humi = '';
22 var lux = '';
23 var mdata = []; // this array stores date and data from multiple sensors
24 var firstcommaidx = 0;
25
26 sp.on('data', function (data) { // call back when data is received
27   readData = data.toString(); // append data to buffer
28   firstcommaidx = readData.indexOf(','); // string.indexOf(searchvalue,start)
29   //console.log(data);
30
31   // parsing data into signals
32
33   Complete your parser code!!
34
35   //console.log(firstcolonidx + "," + readData.indexOf(':', firstcolonidx+1))
36   readData = '';
37
38   dStr = getDateString();
39   mdata[0]=dStr; // Date
40   mdata[1]=temp; // temperature data
41   mdata[2]=humi; // humidity data
42   mdata[3]=lux; // luminosity data
43   console.log(mdata);
44   io.sockets.emit('message', mdata); // send data to all clients
45
46 } else { // error
47   console.log(readData);
48 }
49
50
51 });
```




A7.2.2 DHT22 + CdS + Node.js

[2.2] NodeJS code: dht22_ldr_node.js (Complete your parser code)

```
19 var readData = ''; // this stores the buffer
20 var temp = '';
21 var humi = '';
22 var lux = '';
23 var mdata = []; // this array stores date and data from multiple sensors
24 var firstcommaidx = 0;
25
26 sp.on('data', function (data) { // call back when data is received
27   readData = data.toString(); // append data to buffer
28   firstcommaidx = readData.indexOf(','); // string.indexOf(searchvalue,start)
29   //console.log(data);
30
31   // parsing data into signals
32   if (readData.lastIndexOf(',') > firstcommaidx && firstcommaidx > 0) {
33     temp = readData.substring(firstcommaidx + 1, readData.indexOf(',',firstcommaidx+1));
34     humi = readData.substring(readData.indexOf(',',firstcommaidx+1) + 1, readData.lastIndexOf(','));
35     lux = readData.substring(readData.lastIndexOf(',')+1);
36     //console.log(firstcolonidx + "," + readData.indexOf(':', firstcolonidx+1))
37     readData = '';
38
39     dStr = getDateString();
40     mdata[0]=dStr; // Date
41     mdata[1]=temp; // temperature data
42     mdata[2]=humi; // humidity data
43     mdata[3]=lux; // luminosity data
44     console.log(mdata);
45     io.sockets.emit('message', mdata); // send data to all clients
46
47   } else { // error
48     console.log(readData);
49   }
50
51 });
```

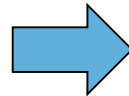


A7.3 DHT22 + CdS + Node.js

[3] Result: Parsed streaming data from dht22 & CdS (Run in Node cmd)

COM4

AAnn,21.5,12.1,156
AAnn,21.5,12.2,158
AAnn,21.5,12.3,158
AAnn,21.4,12.3,156
AAnn,21.4,12.3,157
AAnn,21.3,12.4,157
AAnn,21.3,12.5,113
AAnn,21.3,12.6,41
AAnn,21.2,12.7,157
AAnn,21.2,12.7,158
AAnn,21.2,12.7,157
AAnn,21.1,12.7,157
AAnn,21.0,12.6,158
AAnn,21.0,12.6,158
AAnn,21.0,12.6,157



```
NodeJS - node dht22_ldr_node
D:\Portable\NodeJSPortable\Data\aa00\ldr>node dht22_ldr_node
[ '2017-12-05 17:55:08.320', '18.8', '14.1', '160', '1' ]
[ '2017-12-05 17:55:10.593', '18.8', '14.1', '158', '1' ]
[ '2017-12-05 17:55:12.851', '18.8', '14.1', '160', '1' ]
[ '2017-12-05 17:55:15.125', '18.8', '14.1', '155', '1' ]
[ '2017-12-05 17:55:17.397', '18.8', '14.1', '76', '1' ]
[ '2017-12-05 17:55:19.670', '18.8', '14.1', '158', '1' ]
[ '2017-12-05 17:55:21.943', '19.3', '33.9', '140', '1' ]
[ '2017-12-05 17:55:24.216', '19.5', '50.6', '158', '1' ]
[ '2017-12-05 17:55:26.474', '19.2', '52.5', '160', '1' ]
[ '2017-12-05 17:55:28.747', '19.1', '46.7', '159', '1' ]
[ '2017-12-05 17:55:31.021', '19.1', '40.9', '160', '1' ]
[ '2017-12-05 17:55:33.293', '19.1', '36.3', '159', '1' ]
[ '2017-12-05 17:55:35.566', '19.1', '32.0', '161', '1' ]
[ '2017-12-05 17:55:37.839', '19.1', '28.6', '159', '1' ]
[ '2017-12-05 17:55:40.097', '19.1', '25.6', '158', '1' ]
[ '2017-12-05 17:55:42.370', '19.1', '23.4', '158', '1' ]
[ '2017-12-05 17:55:44.644', '19.1', '21.6', '160', '1' ]
[ '2017-12-05 17:55:46.916', '19.0', '20.0', '158', '1' ]
[ '2017-12-05 17:55:49.188', '19.1', '19.2', '135', '1' ]
[ '2017-12-05 17:55:51.462', '19.0', '18.5', '146', '1' ]
[ '2017-12-05 17:55:53.720', '19.5', '43.0', '146', '1' ]
[ '2017-12-05 17:55:55.993', '19.5', '54.1', '160', '1' ]
[ '2017-12-05 17:55:58.265', '19.3', '46.4', '158', '1' ]
[ '2017-12-05 17:56:00.538', '19.2', '39.5', '160', '1' ]
[ '2017-12-05 17:56:02.812', '19.2', '34.6', '160', '1' ]
[ '2017-12-05 17:56:05.068', '19.2', '31.4', '158', '1' ]
[ '2017-12-05 17:56:07.343', '19.2', '28.1', '159', '1' ]
[ '2017-12-05 17:56:09.616', '19.3', '25.5', '158', '1' ]
[ '2017-12-05 17:56:11.890', '19.5', '23.4', '161', '1' ]
[ '2017-12-05 17:56:14.162', '19.7', '21.7', '159', '1' ]
[ '2017-12-05 17:56:16.436', '20.0', '20.3', '158', '1' ]
[ '2017-12-05 17:56:18.693', '20.2', '19.5', '160', '1' ]
[ '2017-12-05 17:56:20.965', '20.5', '18.7', '161', '1' ]
[ '2017-12-05 17:56:23.238', '20.8', '18.1', '159', '1' ]
```

Save as **AAnn_dht22_ldr_data.png**



A7.4.1 DHT22 + CdS + Node.js

[4.1] WEB client: client_dht22_ldr.html

```
1 <!DOCTYPE html>
2 <head>
3   <meta charset="utf-8">
4   <title>plotly.js Project: Real time signals from multiple sensors</title>
5   <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
6   <script type="text/javascript"
7     src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/1.3.6/socket.io.js"></script>
8
9   <script src="gauge.min.js"></script>
10
11   <style>body{padding:0;margin:30;background:#fff}</style>
12 </head>
13
14 <body> <!-- style="width:100%;height:100%" -->
15   <!-- Plotly chart will be drawn inside this DIV -->
16   <h1 align="center">Real-time Weather Station from sensors</h1>
17   <!-- 1st gauge -->
18   <div align="center">
19     <canvas id="gauge1"> </canvas>
20     <!-- 2nd gauge -->
21     <canvas id="gauge2"> </canvas>
22     <!-- 3rd gauge -->
23     <canvas id="gauge3"> </canvas>
24   </div>
25   <!-- <div id="console"> </div> -->
26   <h3 align="center">on Time: <span id="time"> </span> </h3>
27   <div id="myDiv"></div>
28   <hr>
```



A7.4.2 DHT22 + CdS + Node.js

[4.2] WEB client: client_dht22_ldr.html

```
29  <script>
30    /* JAVASCRIPT CODE GOES HERE */
31    var streamPlot = document.getElementById('myDiv');
32    var ctime = document.getElementById('time');
33    /*var streamToggle = document.getElementById('streamtoggle');
34    */
35    var xArray = [], // time of data arrival
36        y1Track = [], // value of sensor 1 : temperature
37        y2Track = [], // value of sensor 2 : humidity
38        // value of sensor 3 : Luminosity
39        numPts = 100, // number of data points in x-axis
40        dtda = [], // 1 x 4 array : [date, data1, data2, data3]
41        preX = -1,
42        preY = -1,
43        preZ = -1,
44        initFlag = true;
```

Check

xTrack → y1Track, yTrack → y2Track

& add y3Track



A7.4.3 DHT22 + CdS + Node.js

[4.3] WEB client: client_dht22_ldr.html

```
var socket = io.connect('http://localhost:3000');
socket.on('connect', function () {
  socket.on('message', function (msg) {
    // initial plot
    if(msg[0]!='' && initFlag){
      dtda[0]=msg[0];
      dtda[1]=parseFloat(msg[1]); // tempe
      dtda[2]=parseFloat(msg[2]); // Lumin
      init();
      initFlag=false;
    }

    // Convert value to integer
    dtda[0]=msg[0];
    dtda[1] = parseFloat(msg[1]);
    dtda[2] = parseFloat(msg[2]);
```

Update
to include three signals:
temp, humi, lux

```
// Only when any of temperature or Luminosity is di
// the screen is redrawed.
if (dtda[1] != preX || dtda[2] != preY) { // any ch
  preX = dtda[1];
  preY = dtda[2];

  // when new data is coming, keep on streaming
  ctime.innerHTML = dtda[0];
  gauge_temp.setValue(dtda[1]) // temp gauge
  gauge_lux.setValue(dtda[2]); // lux gauge
  //nextPt();
  xArray = xArray.concat(dtda[0])
  xArray.splice(0, 1) // remove the oldest data
  y1Track = y1Track.concat(dtda[1])
  y1Track.splice(0, 1) // remove the oldest data
  y2Track = y2Track.concat(dtda[2])
  y2Track.splice(0, 1)

  var update = {
    x: [xArray, xArray],
    y: [y1Track, y2Track]
  }

  Plotly.update(streamPlot, update);
}
```



A7.4.4 DHT22 + CdS + Node.js

[4.4] WEB client: client_dht22_ldr.html

```
function init() { // initial screen ()  
  // starting point : first data (temp, lux)  
  for ( i = 0; i < numPts; i++) {  
    xArray.push(dtdata[0]); // date  
    y1Track.push(dtdata[1]); // sensor 1 (temp)  
    y2Track.push(dtdata[2]); // sensor 2 (humi)  
    y3Track.push(dtdata[3]); // sensor 3 (lux)  
  }  
  
  Plotly.plot(streamPlot, data, layout);  
}
```

Update
to include three signals:
temp, humi, lux



A7.4.5 DHT22 + CdS + Node.js

[4.5] WEB client: client_dht22_ldr.html - data

```
// data
var data = [{
  x : xArray,
  y : y1Track,
  name : 'temperature',
  mode: "markers+lines", // "
  line: {
    color: "#1f77b4",
    width: 1
  },
  marker: {
    color: "rgb(255, 0, 0)",
    size: 6,
    line: {
      color: "black",
      width: 0.5
    }
  }
},
]
```

```
{
  x : xArray,
  y : y2Track,
  name : 'humidity',
  xaxis: 'x2',
  yaxis : 'y2',
  mode: "markers+lines", // "
  line: {
    color: "#1f77b4",
    width: 1
  },
  marker: {
    color: "rgb(0, 0, 255)",
    size: 6,
    line: {
      color: "black",
      width: 0.5
    }
  }
},
]
```

```
{
  x : xArray,
  y : y3Track,
  name : 'luminosity',
  xaxis: 'x3',
  yaxis : 'y3',
  mode: "markers+lines", // "
  line: {
    color: "#1f77b4",
    width: 1
  },
  marker: {
    color: "rgb(0, 255, 0)",
    size: 6,
    line: {
      color: "black",
      width: 0.5
    }
  }
},
]};
```

Update **data**
to include three signals:
temp, humi, lux



A7.4.6 DHT22 + CdS + Node.js

[4.6] WEB client: client_dht22_ldr.html - layout

```
var layout = {  
  xaxis : {  
    title : 'time',  
    domain : [0, 1]  
  },  
  yaxis : {  
    title : 'temp (°C)',  
    domain : [0, 0.3],  
    range : [-30, 50]  
  },  
  xaxis2 : {  
    title : '',  
    domain : [0, 1],  
    position : 0.35  
  },  
  yaxis2 : {  
    title : 'humi (%)',  
    domain : [0.35, 0.65],  
    range : [0, 100]  
  },  
  xaxis3 : {  
    title : '',  
    domain : [0, 1],  
    position : 0.7  
  },  
  yaxis3 : {  
    title : 'lumi (lux)',  
    domain : [0.7, 1],  
    range : [0, 500]  
  }  
}
```

1. Update **layout**
to include three signals:
temp, humi, lux.
2. Check the domain &
position.

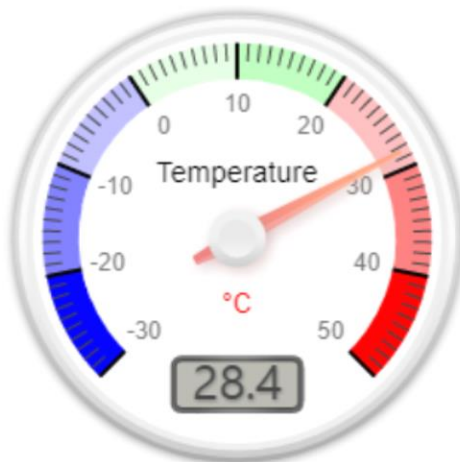
**Save the complete
code as
AAnn_dht22_ldr.html**



A7.5.1 DHT22 + CdS + Node.js

[5.1] WEB client: client_dht22_ldr.html – [Design your gauges](#)

Real-time Weather Station from sensors

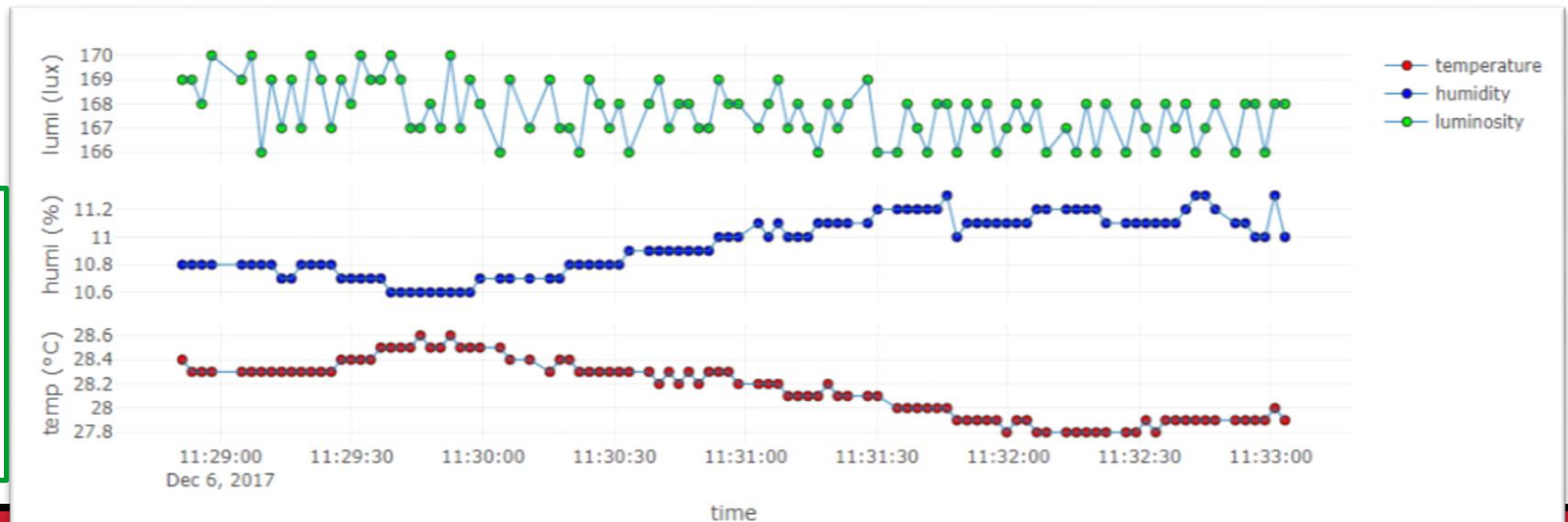
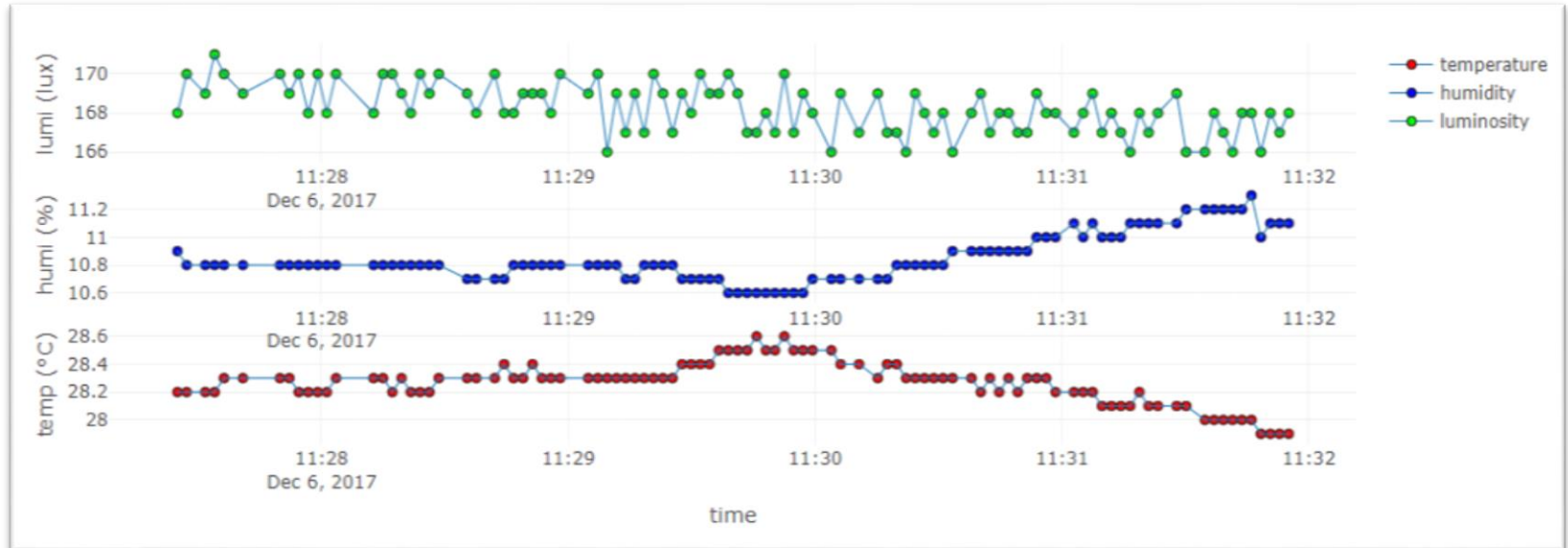


on Time: 2017-12-06 11:30:19.797



A7.5.2 DHT22 + CdS + Node.js

[5.2] WEB client: client_dht22_ldr.html – Design layout



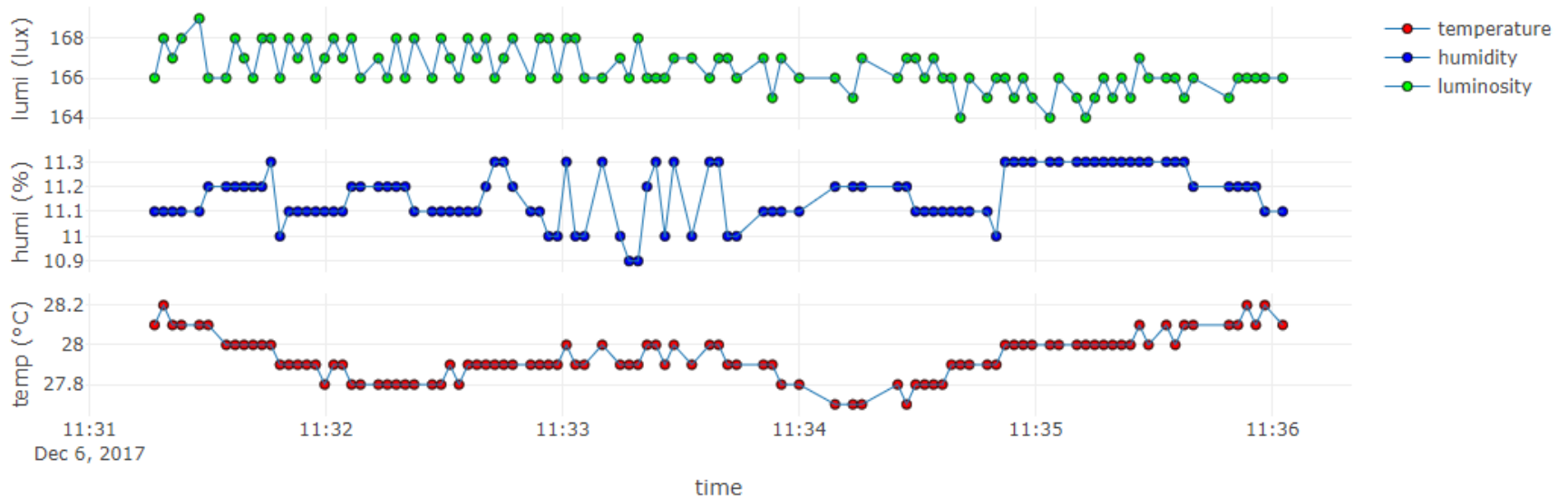
[Hint]

[Plot.ly](https://plot.ly)

Real-time Weather Station from sensors



on Time: 2017-12-06 11:36:02.639



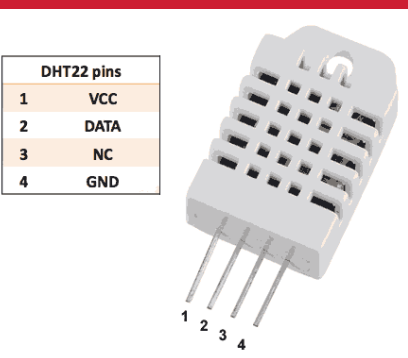


[Practice]



◆ [wk15 - 10점 기말 실기]

- RT Data Visualization with node.js
- DHT22 + CdS sensors
- Complete your real-time charts
- AAnn_Rpt12.zip



◆ [Target of this week]

- Complete your plots of real-time streaming of DHT22 & CdS
- Design your own gauge and layout.
- Save your outcomes and compress them.

제출파일명 : **AAnn_Rpt12.zip**

▪ 압축할 파일들

① **AAnn_dht22_ldr_data.png**

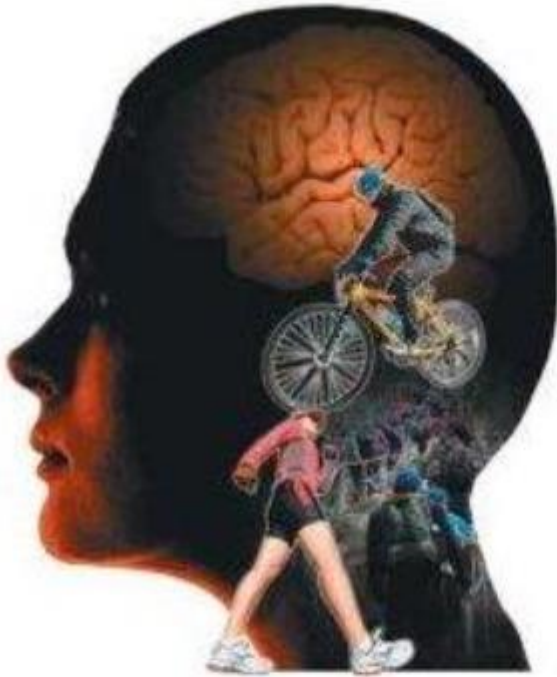
② **AAnn_dht22_ldr.html**

Email : chaos21c@gmail.com

필기시험

1. 일시: 12월 13일(수) 오후 2시 ~ 3시
2. 장소: E323
3. 20문제 (객관식 및 단답형)
4. 범위 : **node.js + arduino** (중간고사 이후 **code** 출제)

Further study to store data to MongoDB



Redwoods Yi

Redwoods

Block or report user

Overview

Repositories 5

Stars 2

Followers

Pinned repositories

dht22-iot-project

lot project to monitor data streaming from DHT22 wired at Arduino.

● HTML

arduino-nodejs-plotly-streaming

This repo introduces a simple and efficient way to plot the streaming data from Arduino with Easy Pulse ppg sensor or DHT11 sensor.



[Tip] Using WEB browser in SB text3

[Tool] Sublime Text - 현재 작업 중인 파일을 웹브라우저로 열기

1. Tool -> New Plugin을 실행 한 후 아래 내용으로 덮어 씌운 후 '**open_browser**'으로 저장한다.

```
import sublime, sublime_plugin
import webbrowser

class OpenBrowserCommand(sublime_plugin.TextCommand):
    def run(self,edit):
        url = self.view.file_name()
        webbrowser.open_new(url)
```

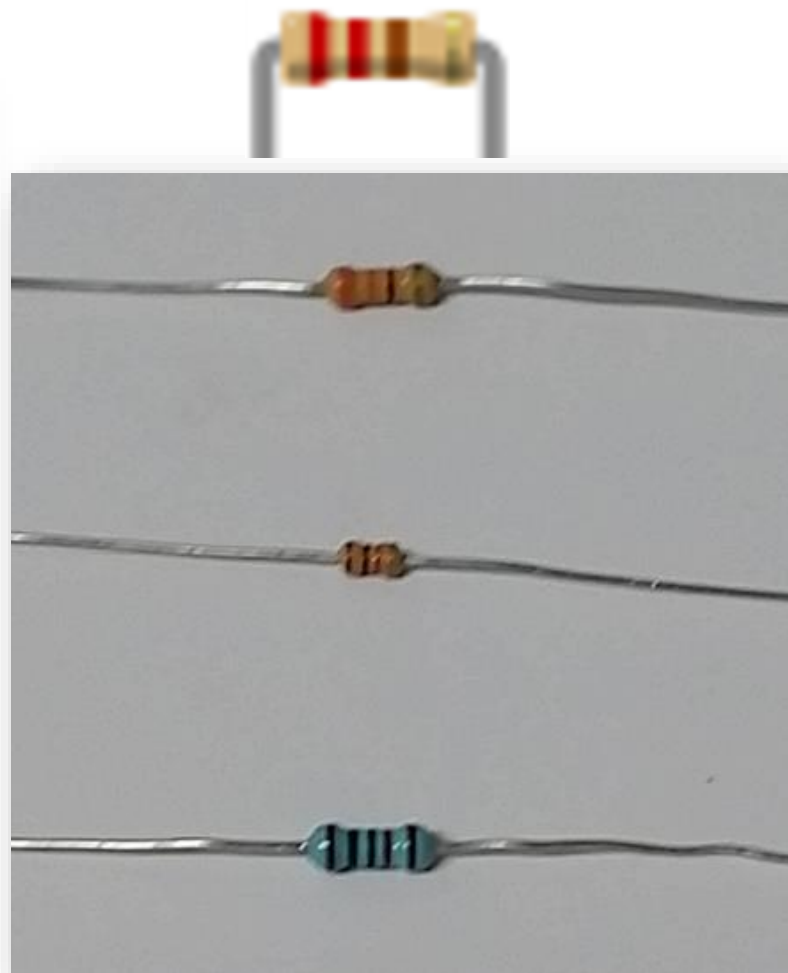
2. Preferences -> Key Bindings - User로 이동한 후 단축키를 할당한다.

```
{ "keys": ["f10"], "command": "open_browser" }
```

[참고 : 저항 값 읽기]



Color	First	Second	Third	Multiplier	Tolerance
Black	0	0	0	x1	
Brown	1	1	1	x10	1%
Red	2	2	2	x100	2%
Orange	3	3	3	x1000	
Yellow	4	4	4	x10 000	
Green	5	5	5	x100 000	0,50%
Blue	6	6	6	x1 000 000	0,25%
Violette	7	7	7	x10 000 000	0,10%
Gray	8	8	8		
White	9	9	9		
Silver				x0,01	10%
Gold				x0,1	5%



● References & good sites

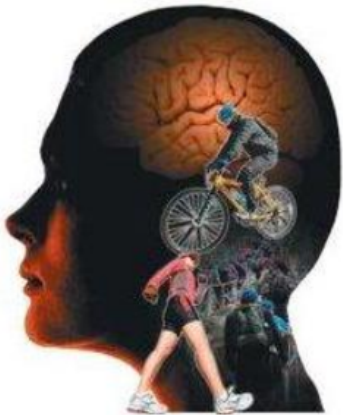
- ✓ <http://www.nodejs.org/ko> Node.js
- ✓ <http://www.arduino.cc> Arduino Homepage
- ✓ <http://www.w3schools.com> By w3schools
- ✓ <http://www.github.com> GitHub
- ✓ <http://www.google.com> Googling



[Features](#) [Business](#) [Explore](#) [Marketplace](#) [Pricing](#)

Search GitHub

[Sign in](#) or [Sign up](#)



Redwoods Yi

Redwoods

Block or report user

📍 GimHae, Republic of Korea

Overview

Repositories 5

Stars 2

Followers 0

Following 0

Pinned repositories

dht22-iot-project

lot project to monitor data streaming from DHT22 wired at Arduino.

● HTML

Lec

All lectures by Redwoods in Inje University

arduino-nodejs-plotly-streaming

This repo introduces a simple and efficient way to plot the streaming data from Arduino with Easy Pulse ppg sensor or DHT11 sensor.

● HTML

hw-coding

Resource for lecture of Hardware Programming (2017, Inje university)

● Arduino

