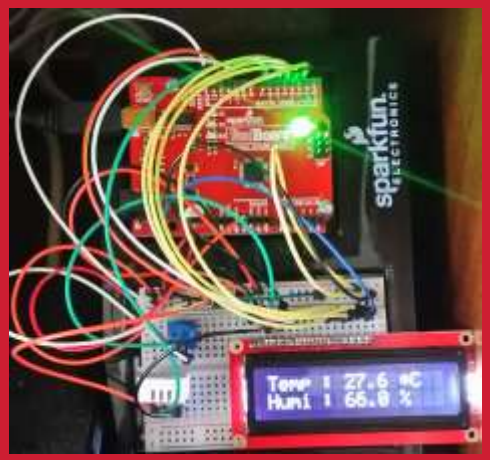
Arduino-IOT

[wk09]



Mid-exam: CdS+DHT22

Arduino + node + plotly

Visualization of Signals using Arduino, Node.js & storing signals in MongoDB & mining data using Python

Drone-IoT-Comsi, INJE University

2nd semester, 2021

Email : chaos21c@gmail.com

My ID

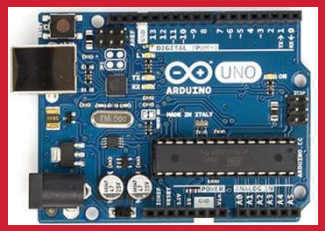
ID를 확인하고 github에 repo 만들기

AA01	김준수	AA13	조재윤
AA02	김현서	AA14	고태승
AA03	박영훈	AA15	이한글
AA04	박윤호	AA16	장세진
AA05	성은지	AA17	장태호
AA06	손윤우	AA18	정지원
AA07	오세윤	AA19	진우태
AA08	우승철	AA20	황혁준
AA09	윤현석	AA21	장이제
AA10	이예주	AA22	박상현
AA11	강지환	AA23	정은성
AA12	성인제	AA24	김경영

위의 id를 이용해서 github에 repo를 만드시오.

Option: 아두이노응용 실습 과제 – AAnn

Public, README.md check



[Review]

- [wk08]
  - RT Data Visualization with node.js
  - Usage of gauge.js
  - Complete your plotly-node project
  - Upload folder: aann-rpt08
  - Use repo “aann” in github

wk08 : Practice : aann-rpt08

- [Target of this week]
  - Complete your works
  - Save your outcomes and upload outputs in github

제출폴더명 : **aann-rpt08**

- 압축할 파일들

① AAnn_DS_30timestamps.png

② AAnn_DS_multiple_axis.png

③ AAnn_cds_gauge.png

④ AAnn_cds_change.png

⑤ AAnn_DS_cds_tmp36.png

⑥ All *.ino

⑦ All *.js

⑧ All *.html

Email : chaos21c@gmail.com

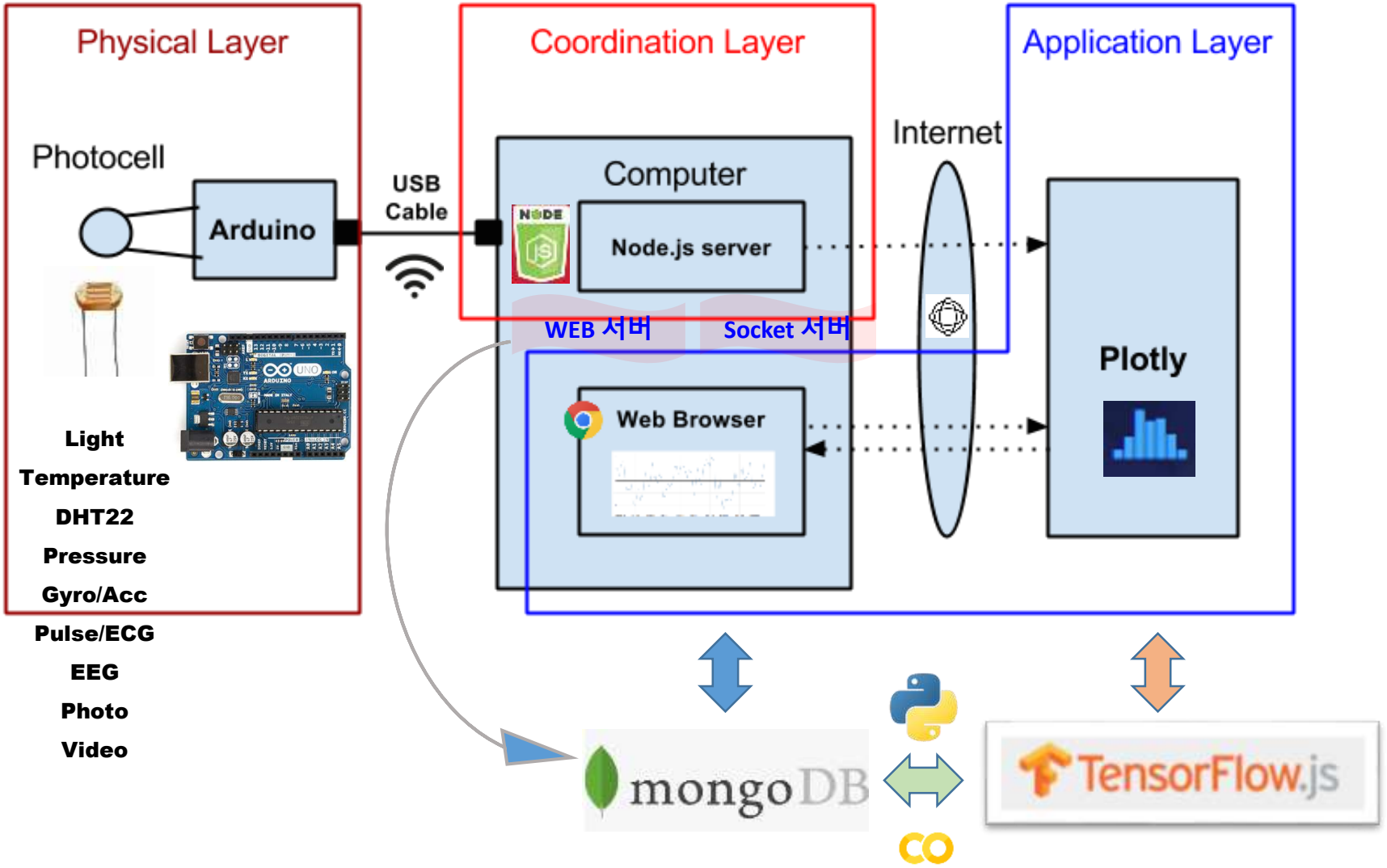
Purpose of AA

주요 수업 목표는 다음과 같다.

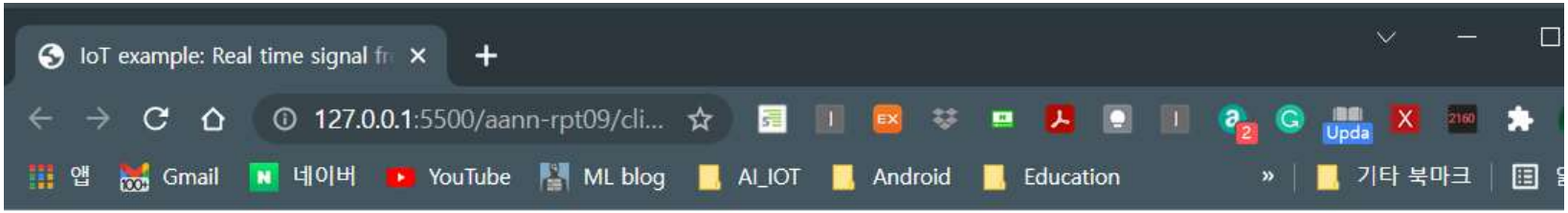
1. Node.js를 이용한 아두이노 센서 신호 처리
2. Plotly.js를 이용한 아두이노 센서 신호 시각화
3. MongoDB에 아두이노 센서 데이터 저장 및 처리

4. 저장된 IoT 데이터의 마이닝 (파이썬 코딩)

Layout [H S C]


Physical Layer
Photocell
Arduino
USB Cable
Light
Temperature
DHT22
Pressure
Gyro/Acc
Pulse/ECG
EEG
Photo
Video
Coordination Layer
Computer
Node.js server
WEB 서버
Socket 서버
Web Browser
Internet
Application Layer
Plotly
mongoDB
TensorFlow.js

on WEB monitoring Arduino data



IoT Signal from Arduino Weather Station

Real-time Signals

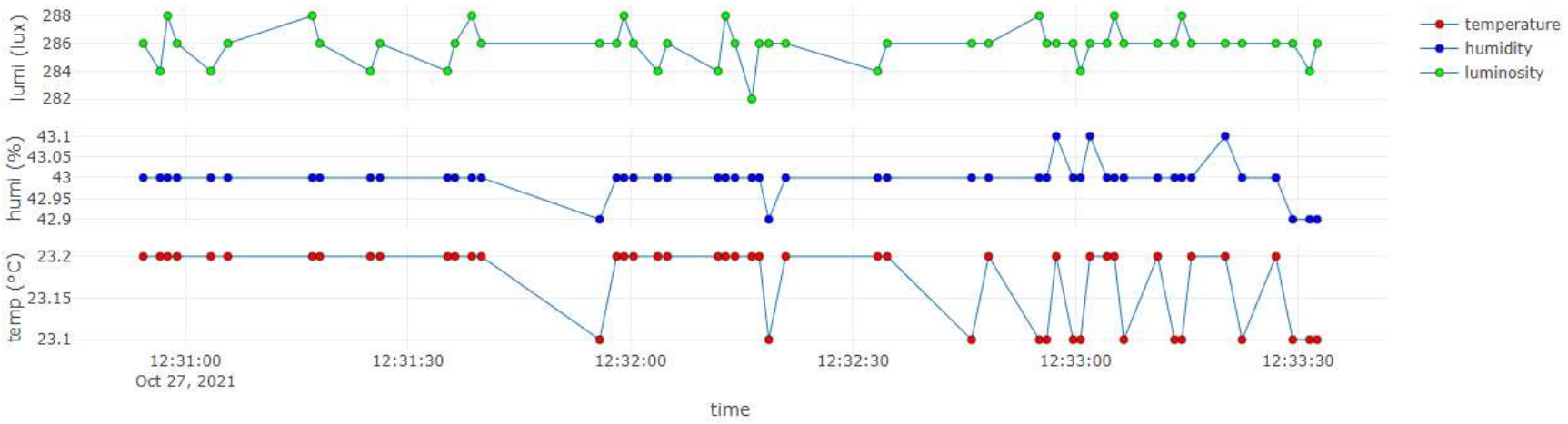
---

on Time: 2021-10-27 11:54:48.997

Signals (온도,습도,조도) : 23.4,42.6,286

Real-time Weather Station from sensors

on Time: 2021-10-27 12:33:32.600



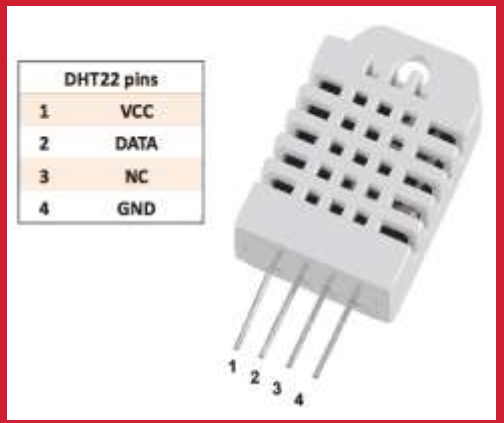
CdS + DHT22

+ plotly.js

Node project

Multi-sensors

DHT22 + CdS



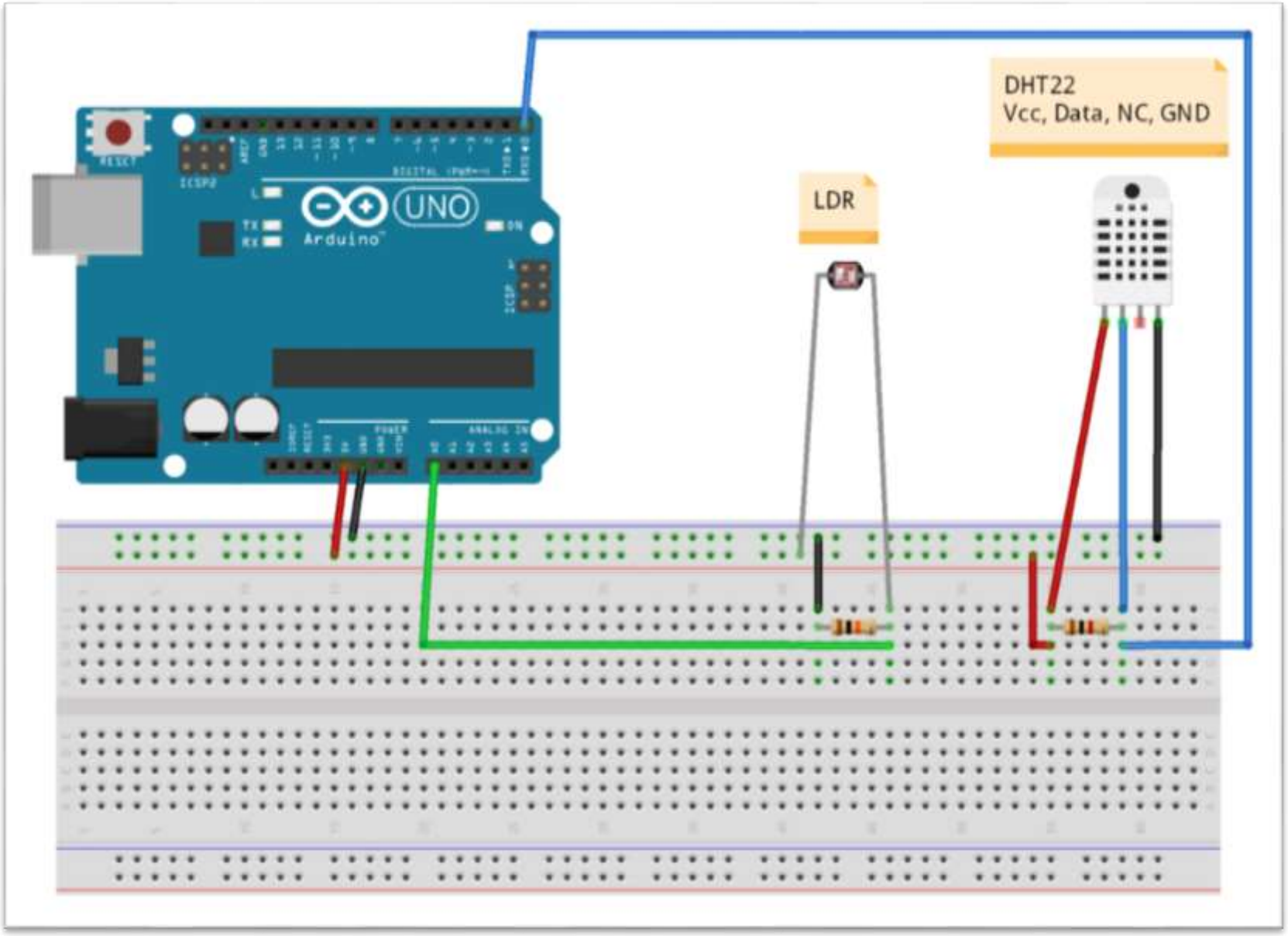
DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND

그림 8-7 DHT22 pin 구조

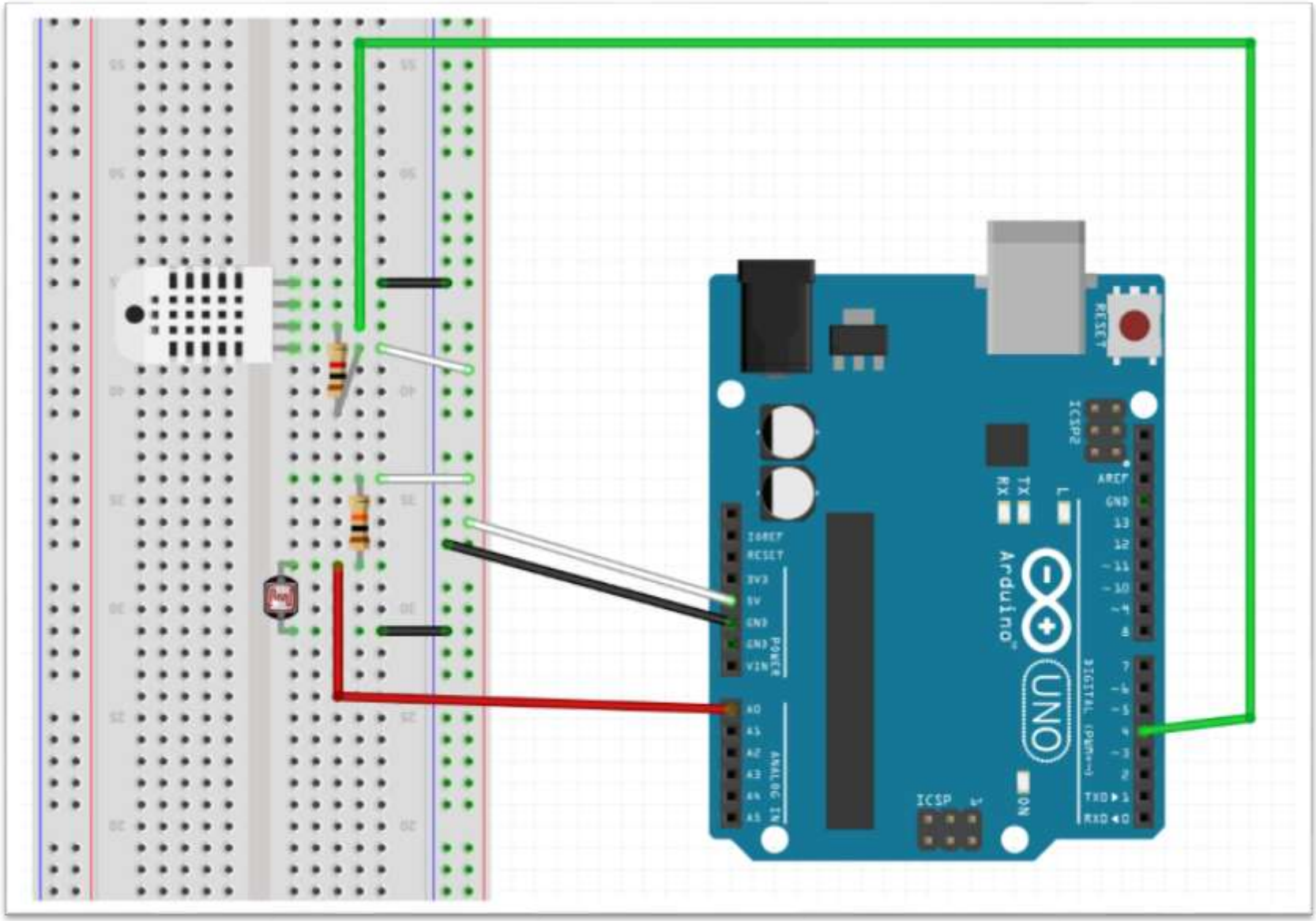
- 3 ~ 5V power and I/O
- 2.5mA max current
- [0-100%] humidity readings with 2-5% accuracy
- [-40 to 80°C] temperature readings ±0.5°C accuracy
- 0.5 Hz sampling rate

https://learn.adafruit.com/dht/overview

A5.7 DHT22 + CdS streaming project



A5.7.1 DHT22 + CdS circuit



DHT22[D4] + 1 kΩ, CdS[A0] + 10 k Ω

A5.7.2 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

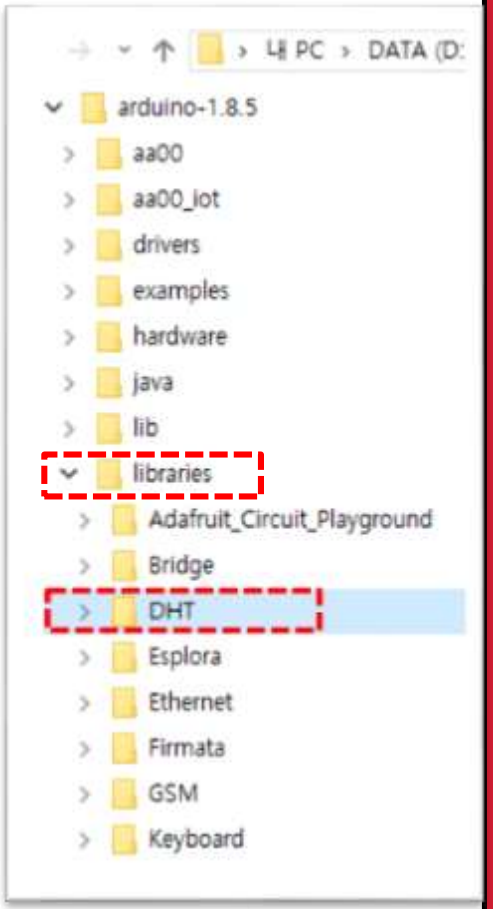
Arduino library for DHT11DHT22, etc Temp & Humidity Sensors http://www.ladyada.net/lea

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



A5.7.4 DHT22 + CdS : circuit

[1] Arduino code: AAnn_CdS_DHT22.ino

AAnn_CdS_DHT22§

```
// DHT22
#include "DHT.h"
#define DHTPIN 4
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
// CdS (LDR)
#define CDS_INPUT 0

void setup() {
  dht.begin();
  Serial.begin(9600);
}
```

```
//Voltage to Lux
double luminosity (int RawADC0){
  double Vout=RawADC0*5.0/1023.0;  // 5/1023
  double lux=(2500/Vout-500)/10;
  // lux = 500 / Rldr,
  // Vout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
  return lux;
}
```

```
void loop() {
  int cds_value, lux;
  float temp, humi;
  // Lux from CdS (LDR)
  cds_value = analogRead(CDS_INPUT);
  lux = int(luminosity(cds_value));
  // Reading temperature or humidity takes a given interval!
  // Sensor readings may also be up to 2 seconds 'old'
  humi = dht.readHumidity();
  // Read temperature as Celsius (the default)
  temp = dht.readTemperature();

  // Check if any reads failed and exit early (to try again).
  if (isnan(humi) || isnan(temp) || isnan(lux)) {
    Serial.println("Failed to read from DHT sensor or CdS!");
    return;
  }
  else {
    Serial.print("AA00,")   // 주석 처리
    Serial.print(temp,1);  // temperature, float
    Serial.print(",");
    Serial.print(humi,1);  // humidity, float
    Serial.print(",");
    Serial.println(lux);   // luminosity, int
  }
  delay(2000);  // 2000 msec, 0.5 Hz
}
```

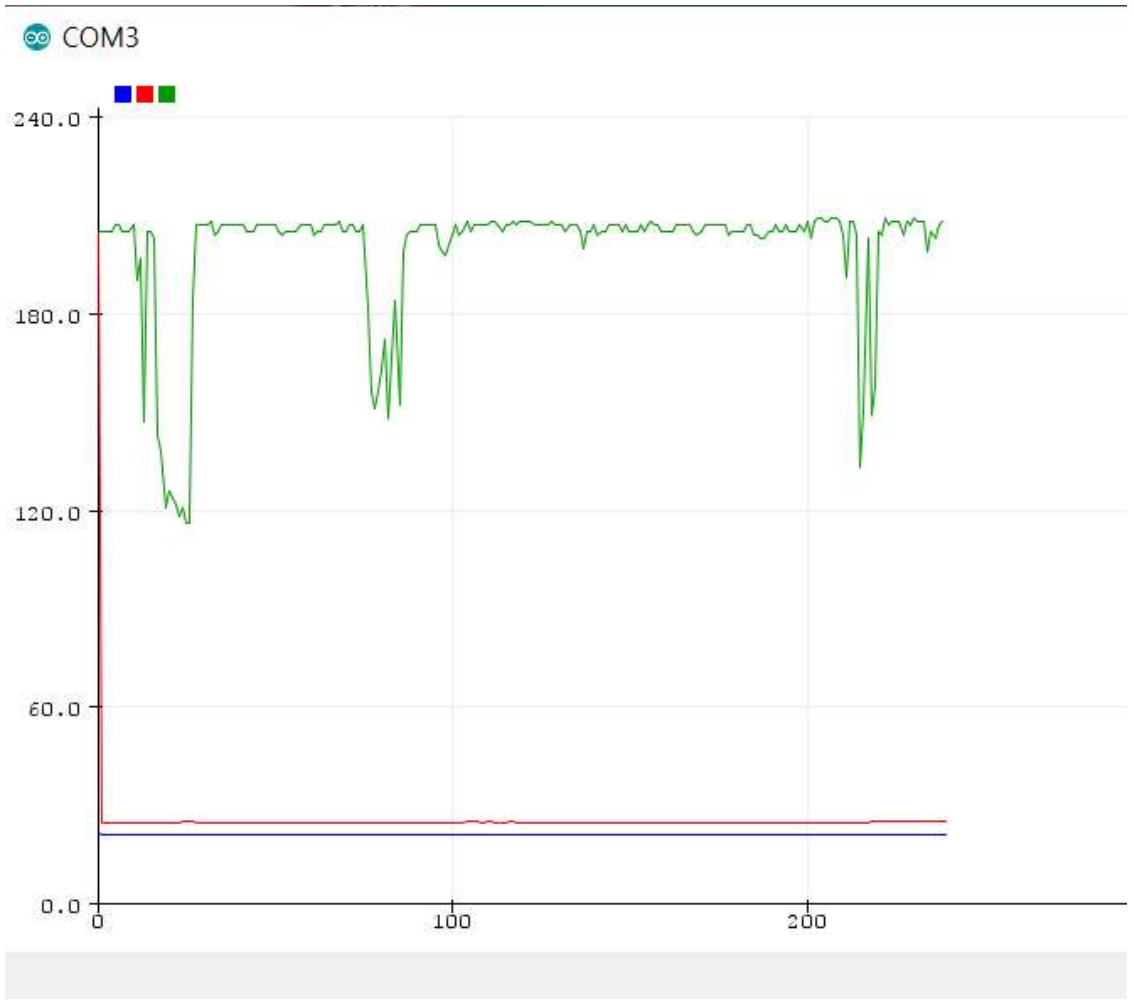
A5.7.5 DHT22 + CdS : Serial monitor

[1] Arduino code: AAnn_CdS_DHT22.ino

COM3

```
21.0,24.7,205
21.0,24.7,207
21.0,24.7,205
21.0,24.7,152
21.0,24.7,167
20.9,24.6,166
20.9,24.6,204
21.0,24.8,204
21.0,24.8,152
21.0,24.8,173
21.0,24.8,191
21.0,24.8,203
21.0,24.8,207
21.0,24.9,204
21.0,24.9,204
```

온도,습도,조도



A5.7.6 DHT22 + CdS + Node.js

[2.1] NodeJS project: "cds-dht22-node project“ → package.json

aann > aann-rpt09 > Node > cds_dht22 > package.json > ...

```
{
  "name": "cds_tmp36",
  "version": "1.0.0",
  "description": "cds-dht22-node project",
  "main": "cds_dht22_node.js",
  ▷ 디버그
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "cds",
    "dht22",
    "node",
    "arduino"
  ],
  "author": "aa00",
  "license": "MIT",
  "dependencies": {
    "serialport": "^9.2.4",
    "socket.io": "^2.4.1"
  }
}
```

A5.7.7 DHT22 + CdS + Node.js

[2.2] NodeJS code: cds_dht22_node.js (← cds_tmp36_node.js를 rename)

```
// cds_dht22_node.js

var serialport = require("serialport");
var portName = "COM3"; // check your COM port!!
var port = process.env.PORT || 3000;

var io = require("socket.io").listen(port);

const Readline = require("@serialport/parser-readline");
// serial port object
var sp = new serialport(portName, {
  baudRate: 9600, // 9600  38400
  dataBits: 8,
  parity: "none",
  stopBits: 1,
  flowControl: false,
  parser: new Readline("\r\n"),
});

const parser = sp.pipe(new Readline({ delimiter: "\r\n" }));

// Read the port data
sp.on("open", () => {
  console.log("serial port open");
});
```

A5.7.8 DHT22 + CdS + Node.js

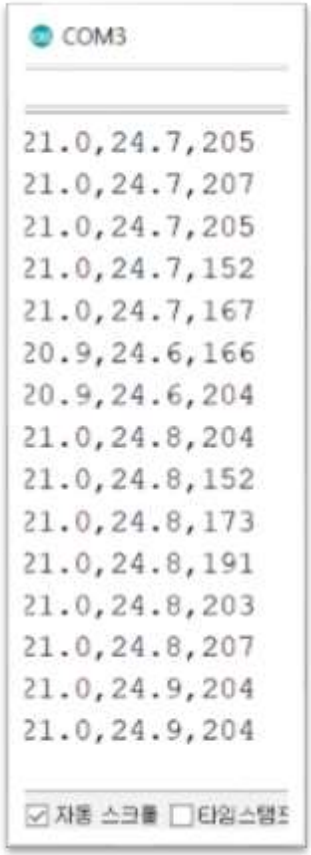
[2.3] NodeJS code: cds_dht22_node.js (Complete your parser code)

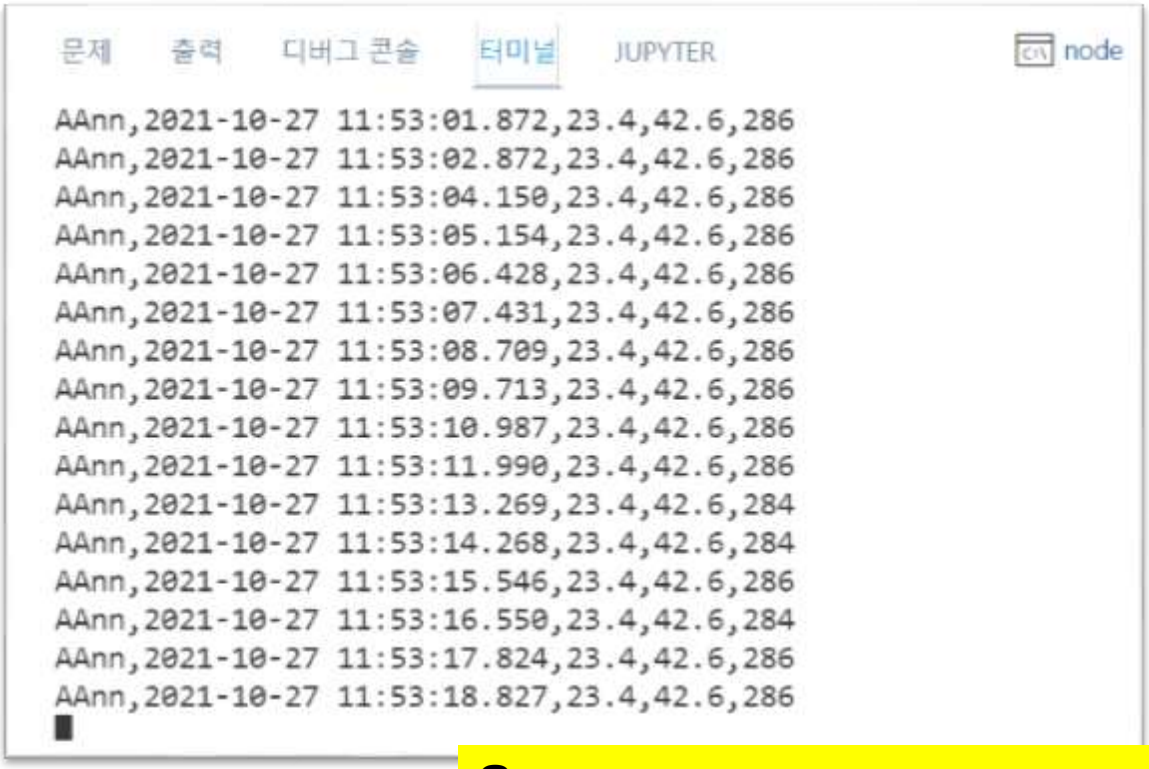
```
var dStr = "";
var readData = ""; //
var temp = "";
var humi = "";
var lux = "";
var mdata = []; // thi
var firstcommaidx = 0;
```

```
parser.on("data", (data) => {
  Complete your parser code!!
                                                    > 0) {
    readData = "";
    dStr = getDateString();
    mdata[0] = dStr; // Date
    mdata[1] = temp; // temperature data
    mdata[2] = humi; // humidity data
    mdata[3] = lux; // luminosity data
    console.log("AAnn," + mdata);
    io.sockets.emit("message", mdata); // send data to all clients
  } else {
    // error
    console.log(readData);
  }
});
```

A5.7.10 DHT22 + CdS + Node.js

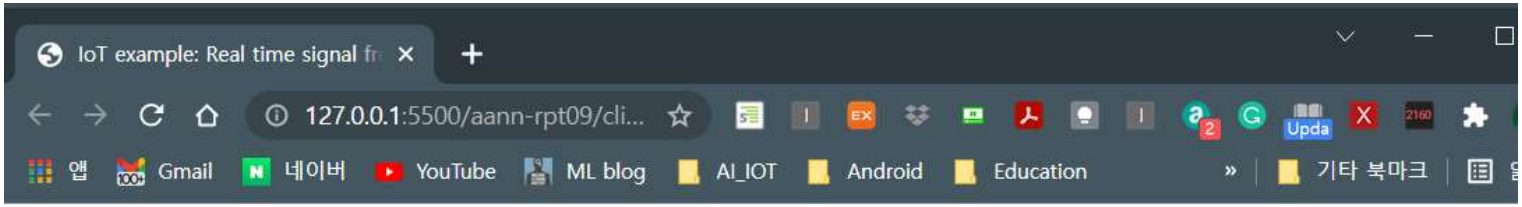
[3] Result: Parsed streaming data from dht22 & CdS (Run in Terminal)





Save as
AAnn_cds_dht22_data.png

Arduino data on network socket



IoT Signal from Arduino Weather Station

Real-time Signals

---

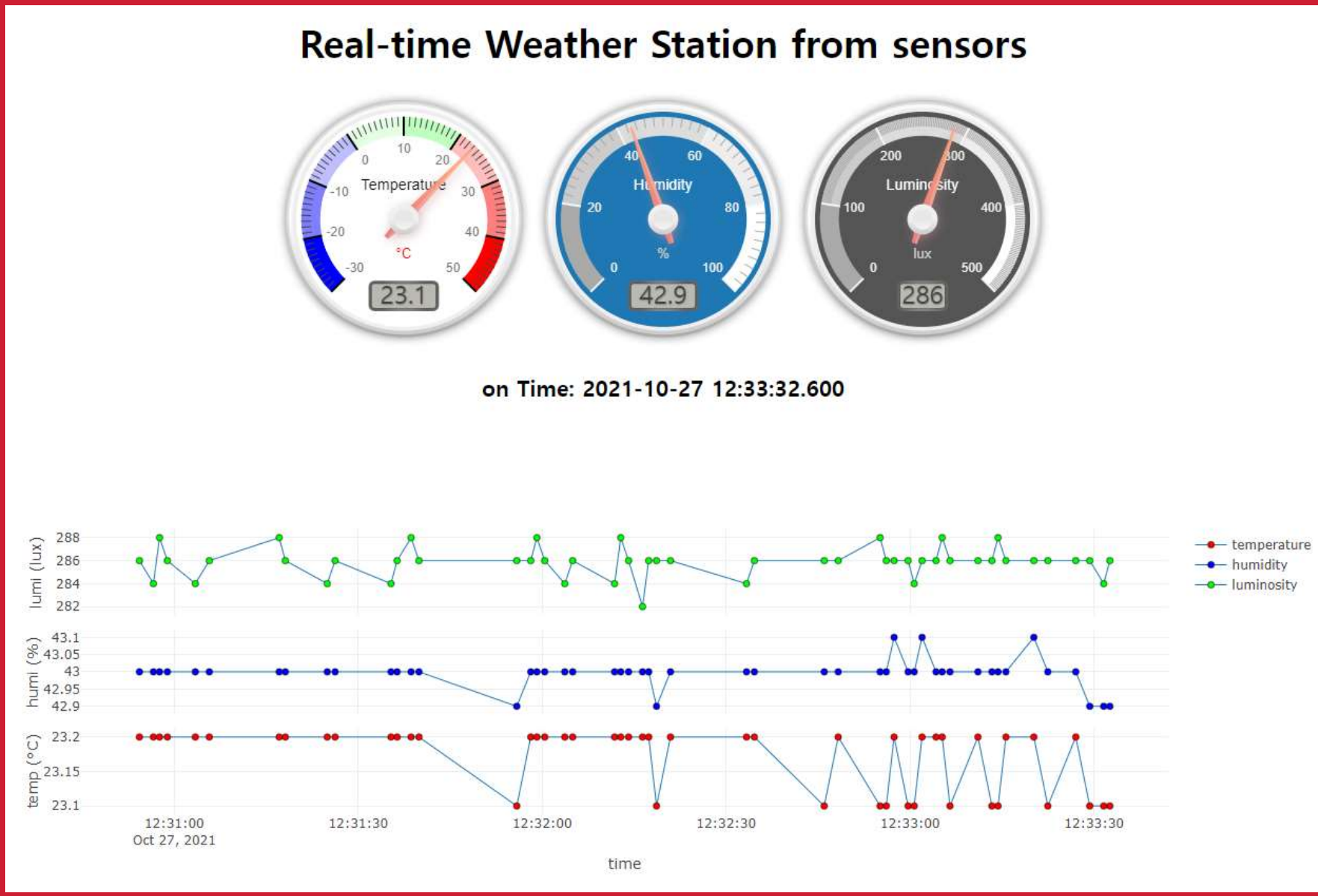
on Time: 2021-10-27 11:54:48.997

Signals (온도,습도,조도) : 23.4,42.6,286

Save as AAnn_signals_cds_dht22.html

Real-time monitoring of signals from Arduino CdS + DHT22 circuit

WEB client： client_cds_dht22.html



A5.8.1 DHT22 + CdS + Node.js

[4.1] WEB client: client_cds_dht22.html

client_CdS_DHT22.html

```
<!DOCTYPE html>
<head>
  <meta charset="utf-8">
  <title>plotly.js Project: Real time signals from multiple sensors</title>
  <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
  <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs
  socket.io.js"></script>

  <script src="gauge.min.js"></script>

  <style>body{padding:0;margin:30;background:#fff}</style>
</head>

<body>  <!-- style="width:100%;height:100%"> -->
    <!-- Plotly chart will be drawn inside this DIV -->
    <h1 align="center">Real-time Weather Station from sensors</h1>
    <!-- 1st gauge -->
    <div align="center">
        <canvas id="gauge1"> </canvas>
        <!-- 2nd gauge -->
        <canvas id="gauge2"> </canvas>
        <!-- 3rd gauge -->
        <canvas id="gauge3"> </canvas>
    </div>
    <!-- <div id="console"> </div> -->
    <h3 align="center"> on Time: <span id="time"> </span> </h3>
    <div id="myDiv"></div>
    <hr>
```

A5.8.2 DHT22 + CdS + Node.js

[4.2] WEB client: client_cds_dht22.html

```
29      <script>
30        /* JAVASCRIPT CODE GOES HERE */
31        var streamPlot = document.getElementById('myDiv');
32        var ctime = document.getElementById('time');
33        var tArray = [], // time of data arrival
34          y1Track = [], //value of sensor 1 : temperature
35          y2Track = [], //value of sensor 2 : humidity
36          y3Track = [], //value of sensor 3 : Luminosity
37          numPts = 50, // number of data points in x-axis
38          dtda = [],  // 1 x 4 array : [date, data1, data2, data3] from sensors
39          preX = -1,
40          preY = -1,
41          preZ = -1,
42          initFlag = true;
```

Check points: tArray

xTrack → y1Track, yTrack → y2Track

& add y3Track & preZ

A5.8.3 DHT22 + CdS + Node.js

[4.3] WEB client: client_cds_dht22.html

```
var socket = io.connect('http://localhost:3000'); // port = 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]);  // temperature
            dtda[2]=parseFloat(msg[2]);  // Humidity
            dtda[3]=parseInt(msg[3]);    // Luminosity
            init();
            initFlag=false;
        }

        dtda[0]=msg[0];
        dtda[1] = parseFloat(msg[1]);
        dtda[2] = parseFloat(msg[2]);
        dtda[3] = parseInt(msg[3]);
```

Update

to include three signals:

temp, humi, lux

A5.8.4 DHT22 + CdS + Node.js

[4.4] WEB client: client_cds_dht22.html

```
// Only when any of data is different from the previous one,
// the screen is redrawed.
if (dtda[1] != preX || dtda[2] != preY || dtda[3] != preZ) {  // any change?
    preX = dtda[1];
    preY = dtda[2];
    preZ = dtda[3];

    // when new data is coming, keep on streaming
    ctime.innerHTML = dtda[0];
    gauge_temp.setValue(dtda[1])  // temp gauge
    gauge_humi.setValue(dtda[2]); // humi gauge
    gauge_lux.setValue(dtda[3]);  // lux gauge
    //nextPt();
    tArray = tArray.concat(dtda[0]);
    tArray.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);
    y3Track = y3Track.concat(dtda[3]);
    y3Track.splice(0, 1);

    var update = {
        x:  [tArray, tArray, tArray],
        y:  [y1Track, y2Track, y3Track]
    }

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

Update

to include three signals:

temp, humi, lux

A5.8.5 DHT22 + CdS + Node.js

[4.5] WEB client: client_dht22_ldr.html → init()

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

    Plotly.plot(streamPlot, data, layout);
}
```

Update

to include three signals:

temp, humi, lux

A5.8.6 DHT22 + CdS + Node.js

[4.6] WEB client: client_cds_dht22.html - data

```
// data
var data = [{
    x : tArray,
    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 : tArray,
    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 : tArray,
    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

A5.8.7 DHT22 + CdS + Node.js

[4.7] WEB client: client_cds_dht22.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]
    }
```

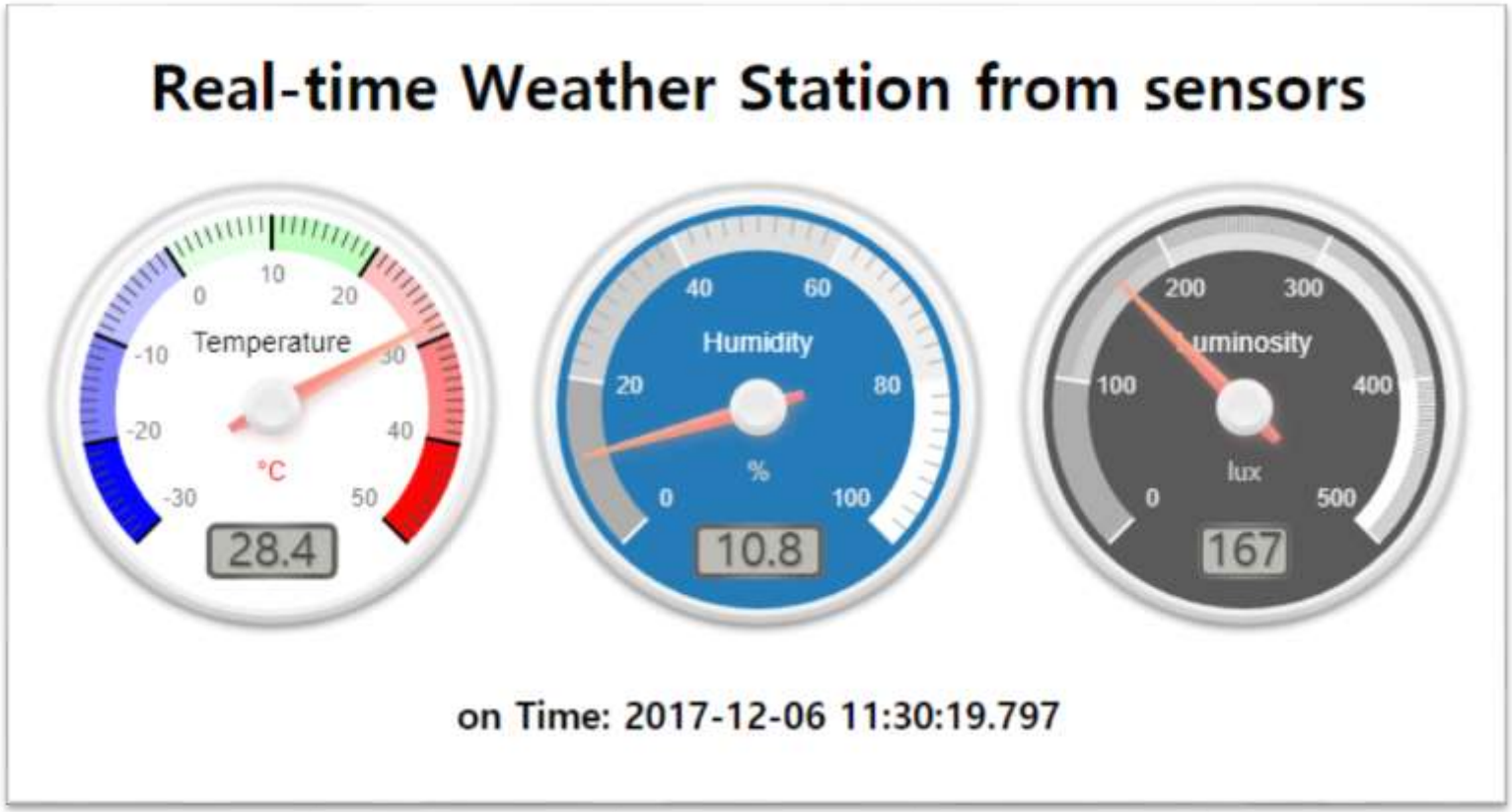
1. Update layout
   to include three signals:
   temp, humi, lux.

2. Check the domain &
   position.

Save the complete code as AAnn_cds_dht22.html

A5.8.8 DHT22 + CdS + Node.js

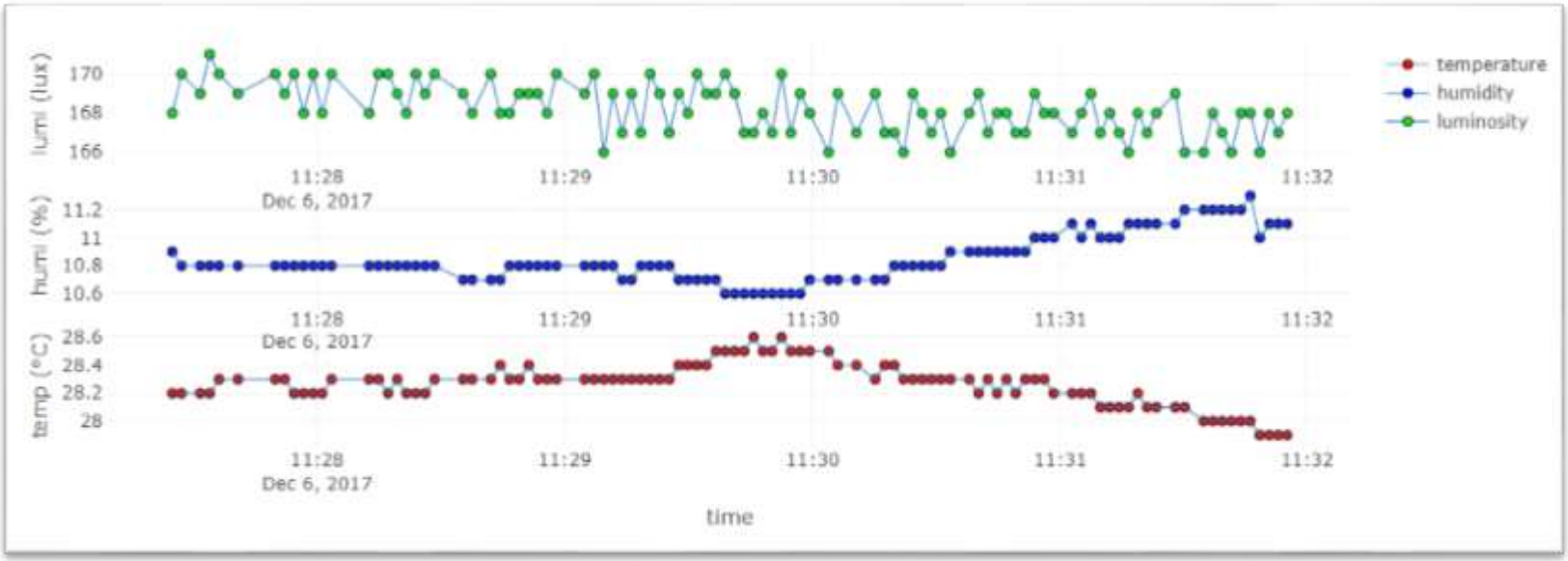
[4.8] WEB client: client_dht22_ldr.html – Design your gauges

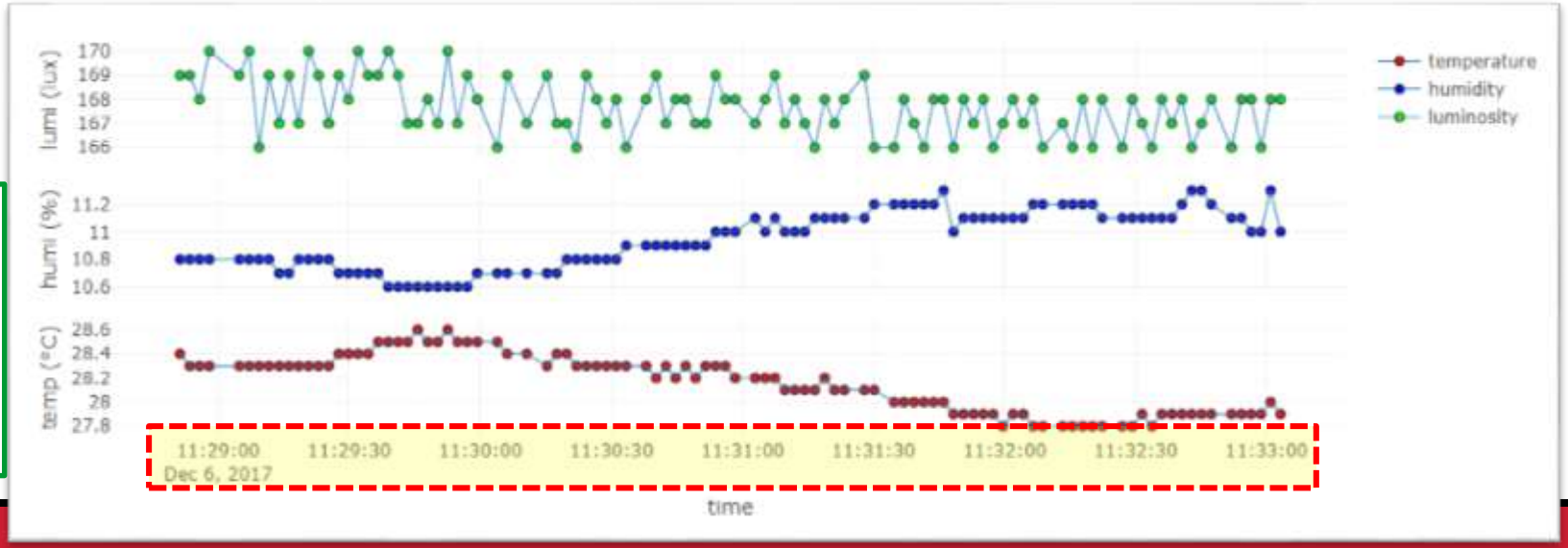


Save the complete code as AAnn_cds_dht22.html

A5.8.9 DHT22 + CdS + Node.js

[4.9] WEB client: Design layout (show date at lower axis)





[Hint]

Plot.ly

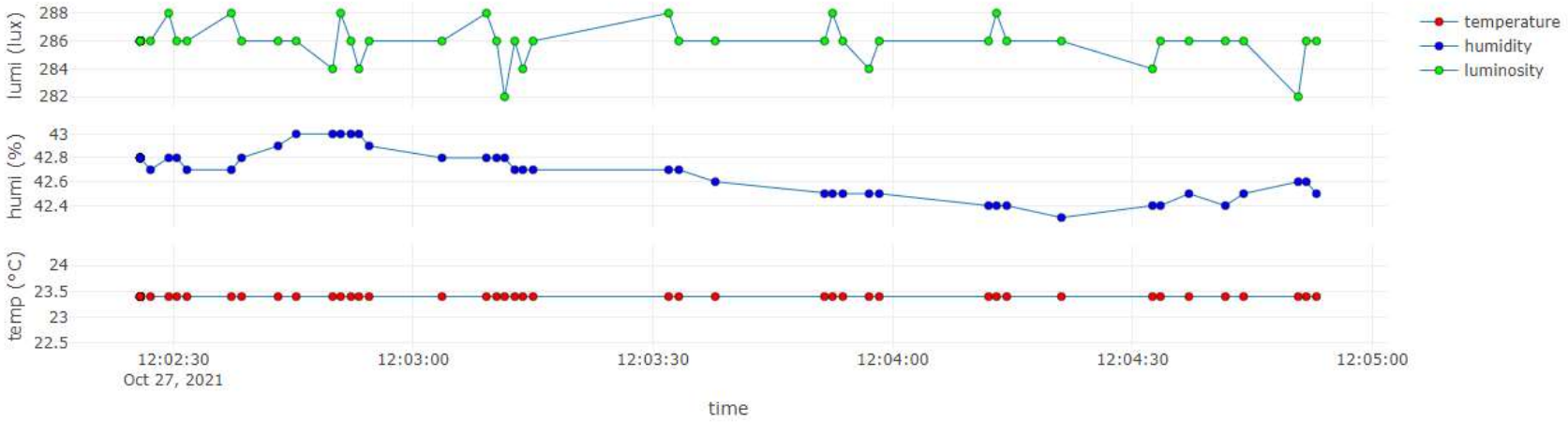
WEB client： client_cds_dht22.html

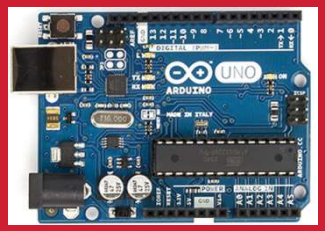
Real-time Weather Station from sensors

on Time: 2021-10-27 12:04:53.016

Save as

AAnn_cds_dht22.png





[Practice]

- [wk09: mid-exam.]
  - RT Data Visualization with node.js
  - Multiple data and Usage of gauge.js
  - Complete your real-time WEB charts
  - Upload folder: aann-rpt09
  - Use repo “aann” in github

wk09 : Practice : aann-rpt09

- [Target of this week]
  - Complete your works : **mid-exam.**
  - **Save your outcomes and upload outputs in github**

제출폴더명 : **aann-rpt09**

- 제출할 파일들

① **AAnn_cds_dht22_data.png**

② **AAnn_signals_cds_dht22.html**

③ **AAnn_cds_dht22.html**

④ **AAnn_cds_dht22.png**

⑤ **All *.ino**

⑥ **All *.js**

⑦ **All *.html**

Lecture materials

References & good sites

- ✓ http://www.arduino.cc Arduino Homepage
- ✓ http://www.nodejs.org/ko Node.js
- ✓ https://plot.ly/ plotly
- ✓ https://www.mongodb.com/ MongoDB
- ✓ http://www.w3schools.com By w3schools
- ✓ http://www.github.com GitHub

Target of this class