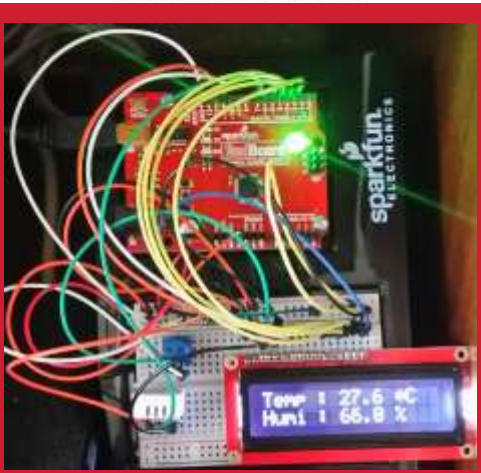
on Time: 2015-09-02 12:48:14.192



Arduino-IOT

[wk11]

Data Visualization - plotly.js

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

Drone-IoT-Comsi, INJE University

2nd semester, 2020

Email : chaos21c@gmail.com

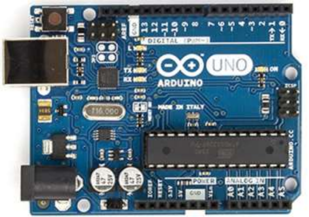
My ID

1분반-목요일 (2학년)

- AA1-01: 강서현
- AA1-02: 강태민
- AA1-03: 김세은
- AA1-04: 여수민
- AA1-05: 정영훈
- AA1-06: 차혁준
- AA1-07: 하태헌
- AA1-08: 김경욱
- AA1-09: 김민욱
- AA1-10: 김민성
- AA1-11: 김민준
- AA1-12: 김인수
- AA1-13: 김현식
- AA1-14: 장성운
- AA1-15: 전승진
- AA1-16: 정희철
- AA1-17: 조동현
- AA1-18: 전동빈
- AA1-19: 신종원

2분반-수요일 (3학년)

- AA2-01: 강민수
- AA2-02: 구병준
- AA2-03: 김종민
- AA2-04: 박성철
- AA2-05: 이승현
- AA2-06: 이창호
- AA2-07: 손성빈
- AA2-08: 안예찬
- AA2-09: 유종인
- AA2-10: 이석민
- AA2-11: 이정문
- AA2-12: 이주원
- AA2-13: 정재영
- AA2-14: 하태성
- AA2-15: 김경미
- AA2-16: 김규년
- AA2-17: 김유빈
- AA2-18: 송다은
- AA2-19: 정주은
- AA2-20: 권준표



[Review]

- [wk10]
  - RT Data Visualization with node.js
  - Usage of gauge.js
  - Complete your plotly-node project
  - Upload folder: aax-nn-rpt08
  - Use repo “aax-nn” in github

wk10 : Practice : aax-nn-rpt08

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

제출폴더명 : aax-nn-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

IOT: HSC

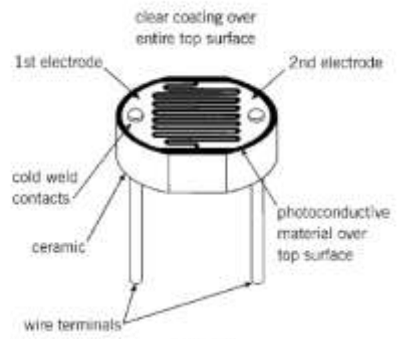
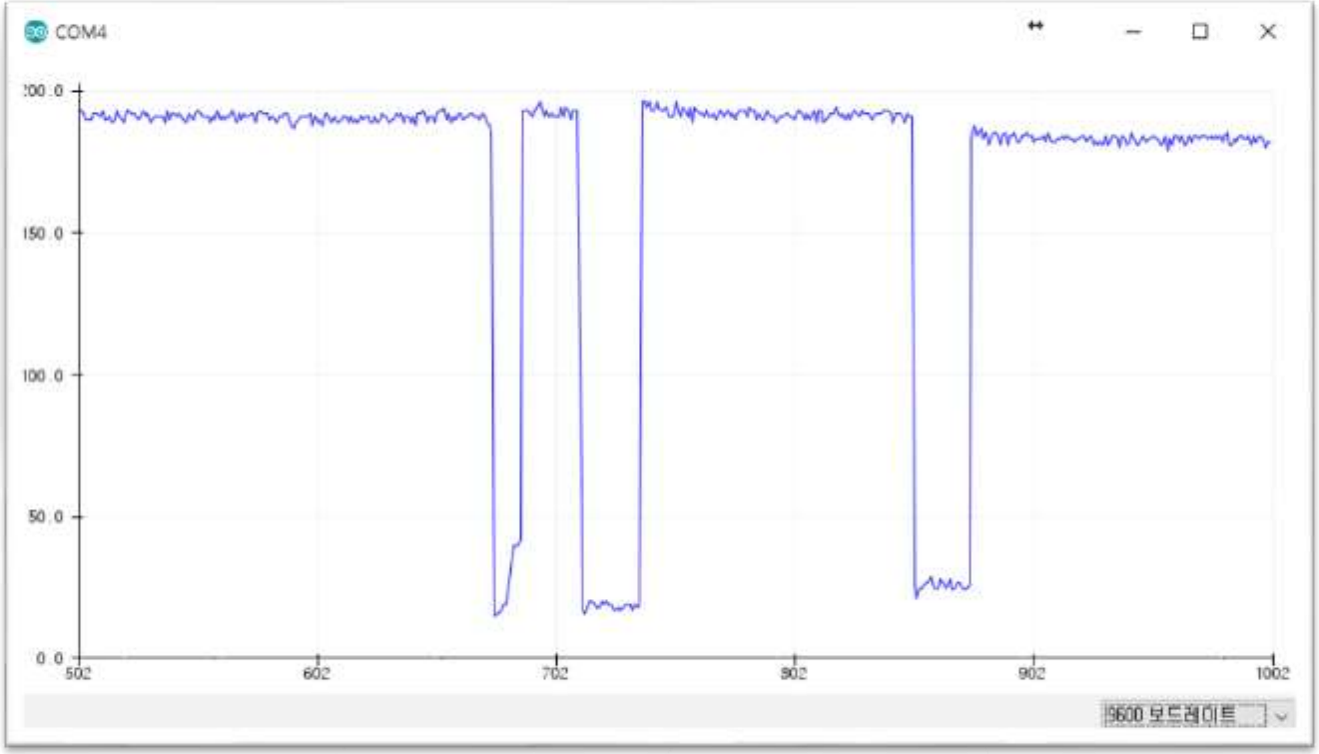
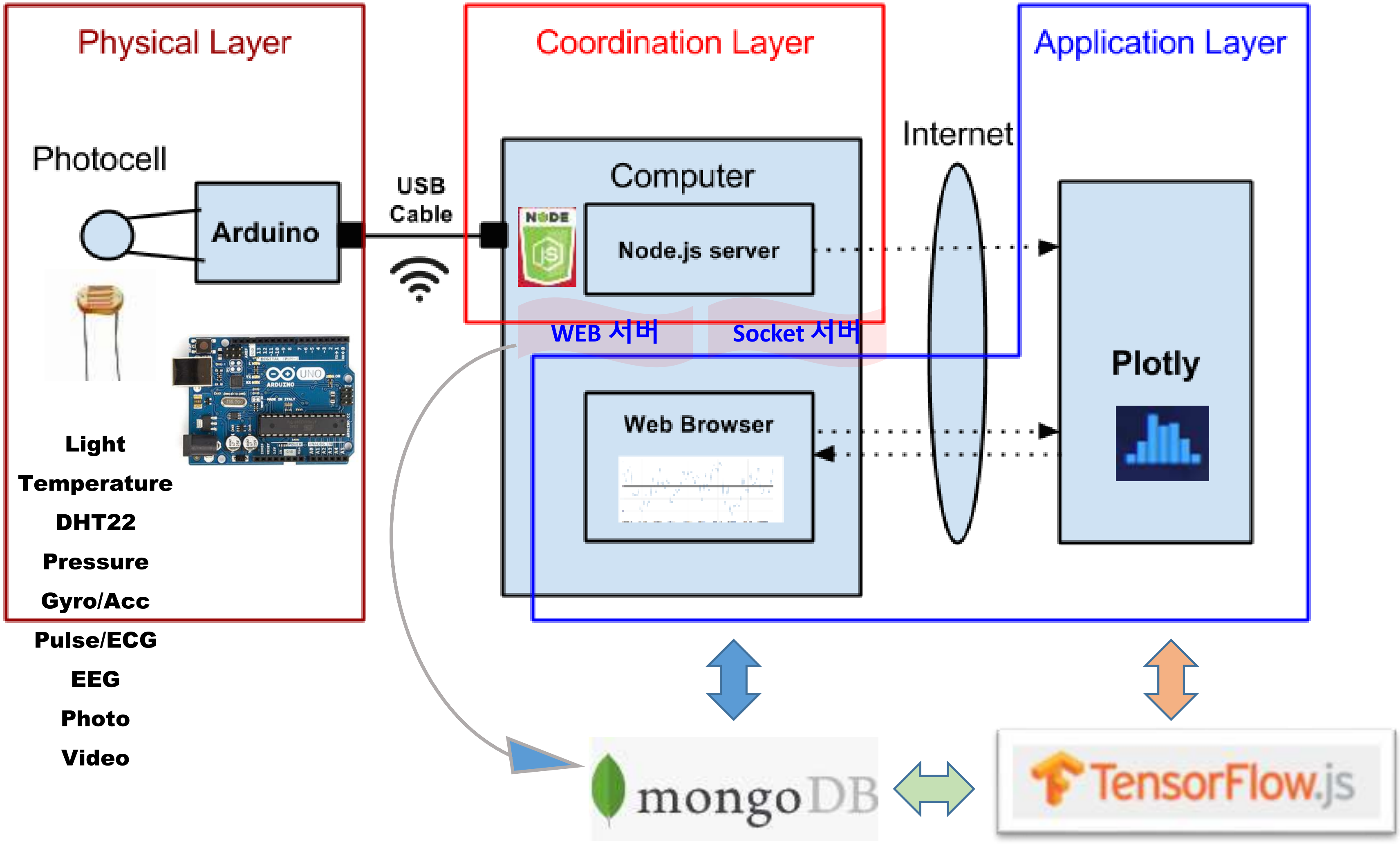


Figure 3
Typical Construction of a Plastic Coated Photocell

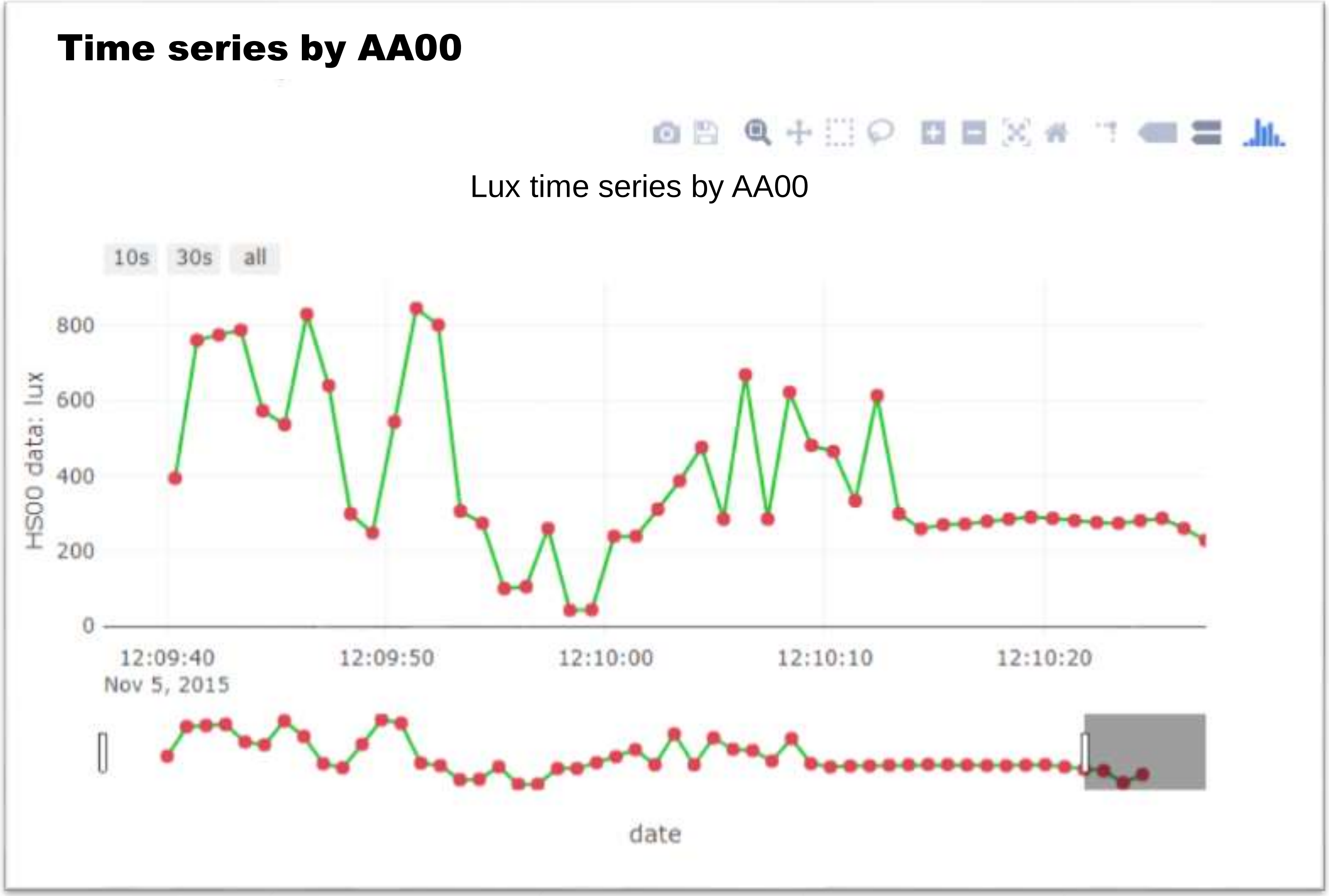


9600 보드레이트

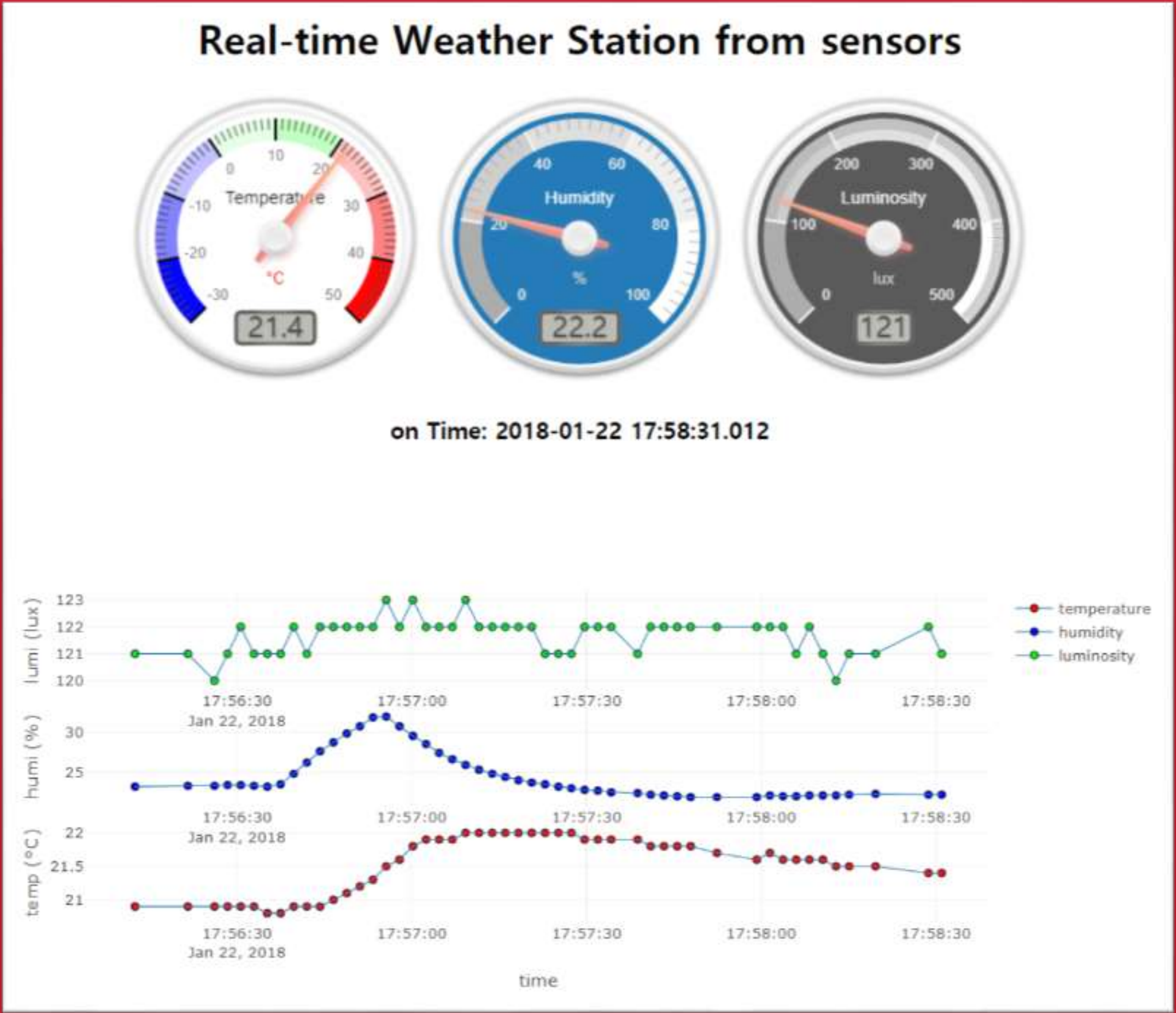
Layout [H S C]


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

Arduino data + plotly

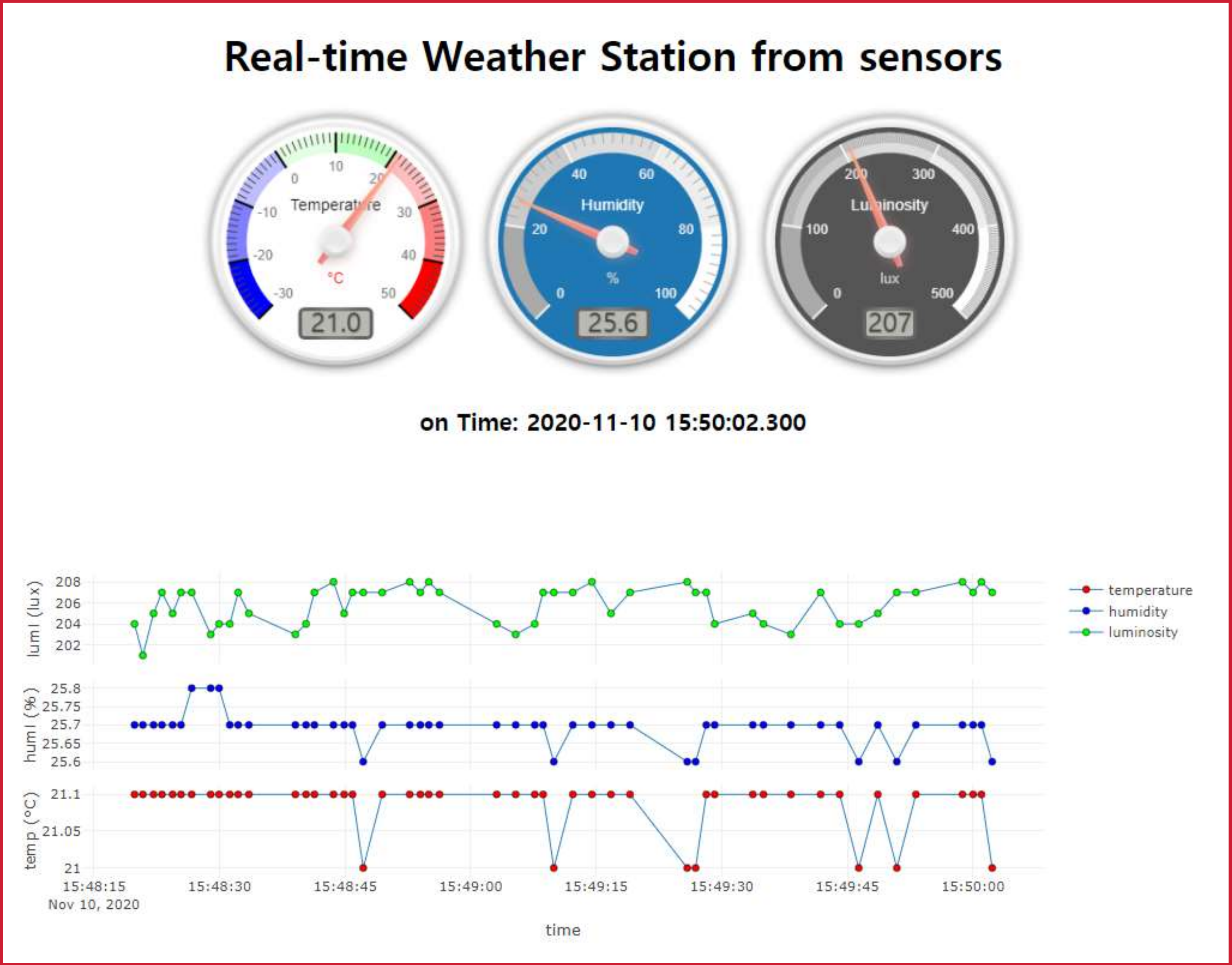


Time series by AA00



Real-time Weather Station from sensors

on Time: 2018-01-22 17:58:31.012



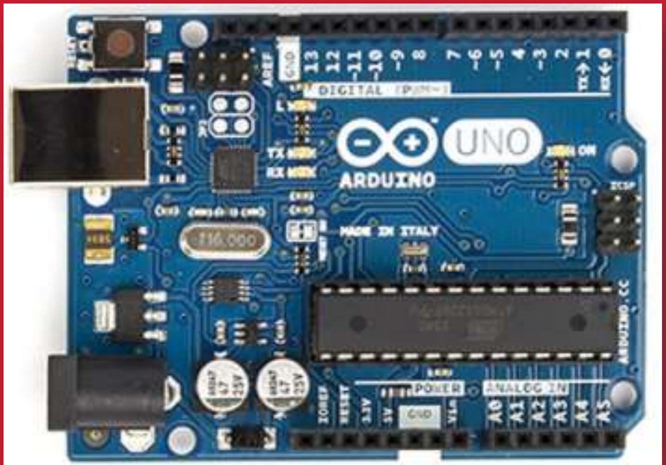
Real-time Weather Station from sensors

on Time: 2020-11-10 15:50:02.300

CdS + DHT22

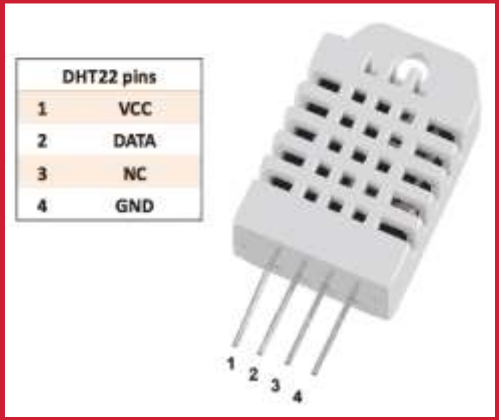
+ plotly.js

Node project

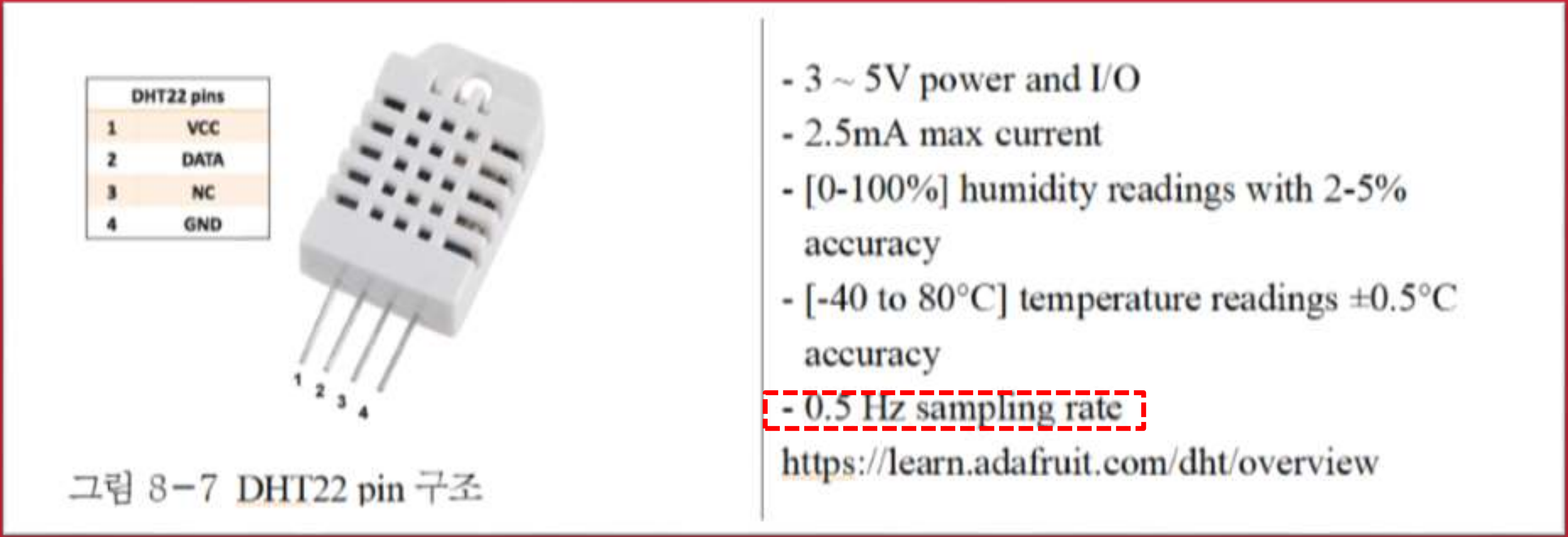


Multi-sensors

DHT22 + CdS



DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND



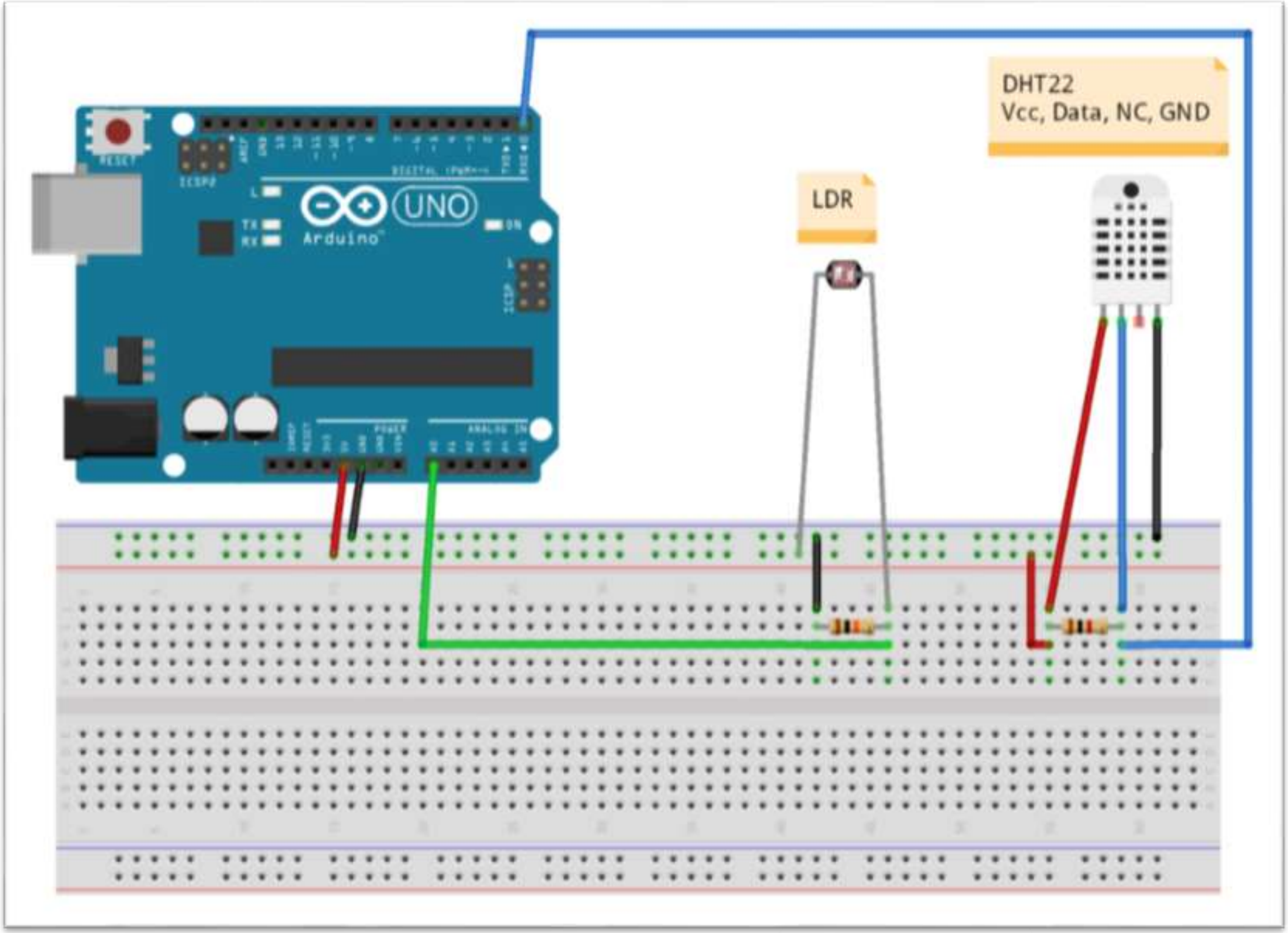
DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND

그림 8-7 DHT22 pin 구조

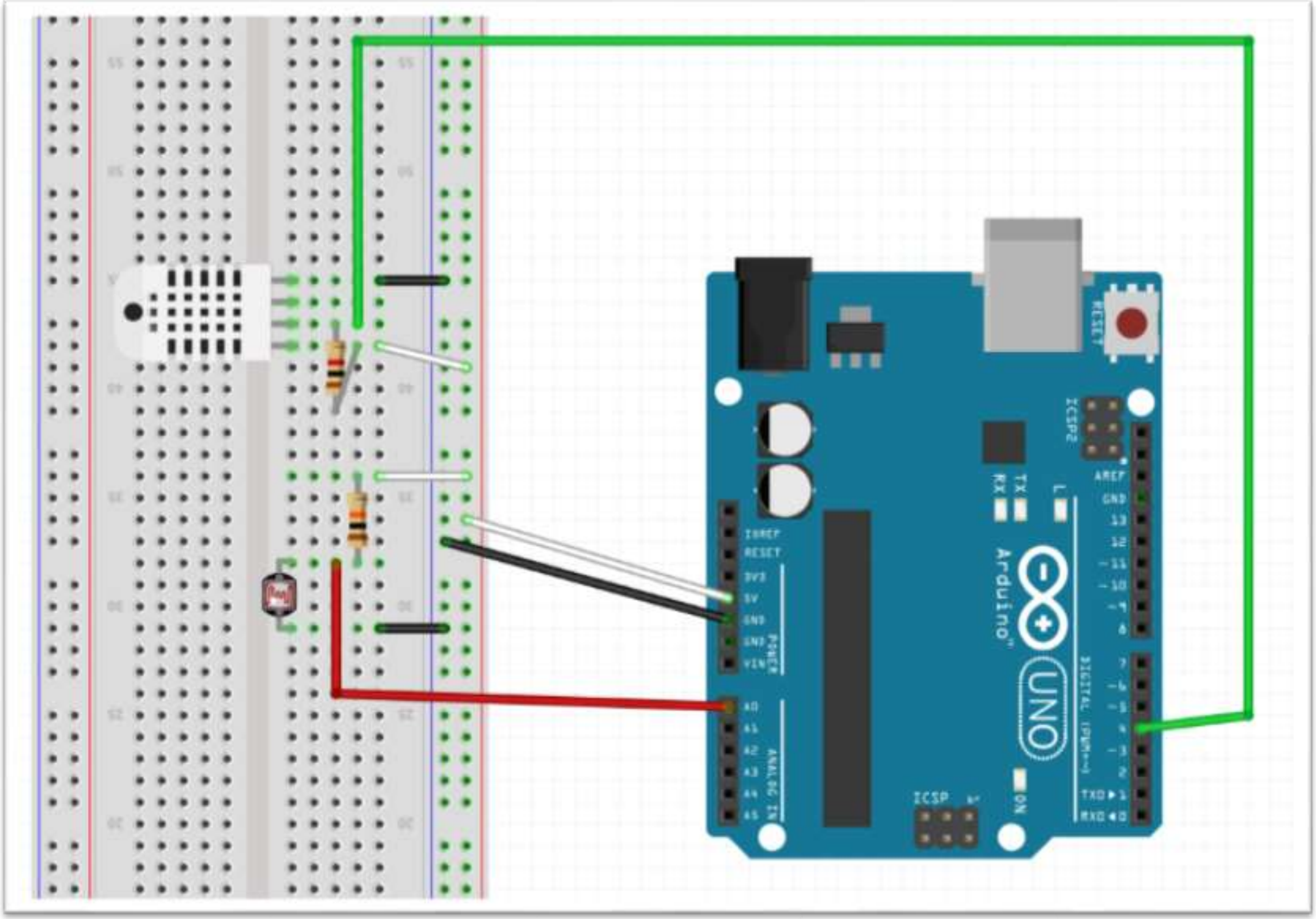
- 3 ~ 5V power and I/O
- 2.5mA max current
- [0-100%] humidity readings with 2-5% accuracy
- [-40 to 80°C] temperature readings ±0.5°C accuracy
- 0.5 Hz sampling rate

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

A5.7 DHT22 + CdS streaming project

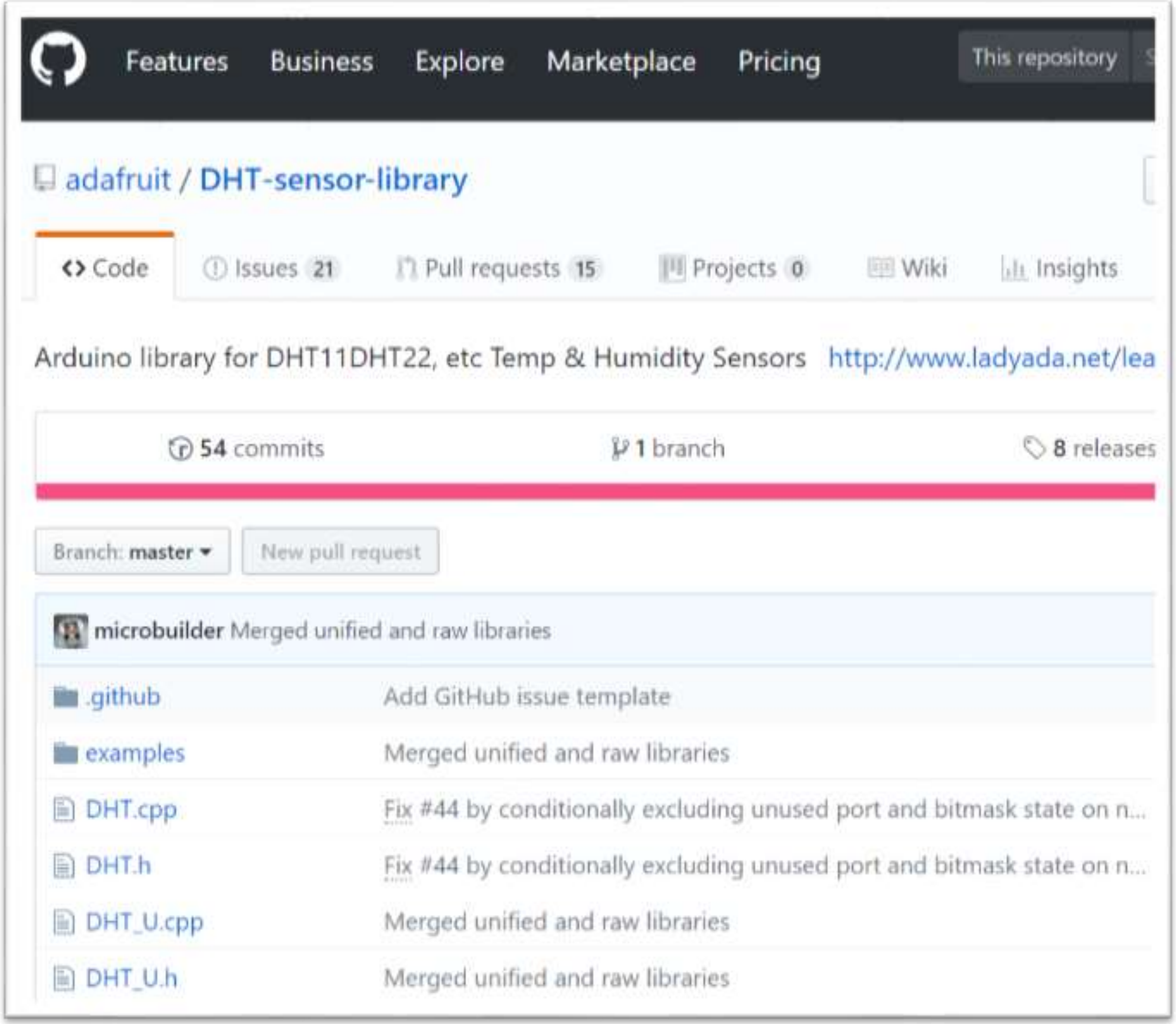


A5.7.1 DHT22 + CdS circuit



DHT22 + 1 kΩ,   CdS + 10 k Ω

A5.7.2 DHT22 + CdS : DHT library



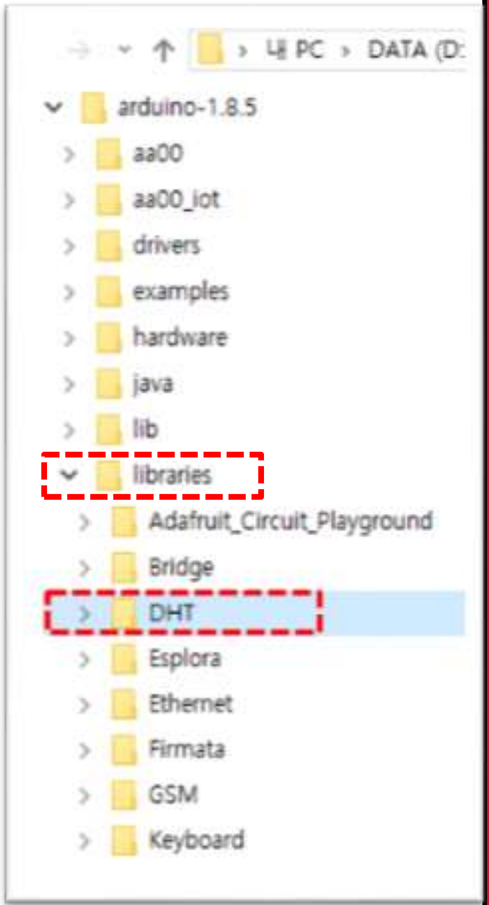
adafruit / DHT-sensor-library

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

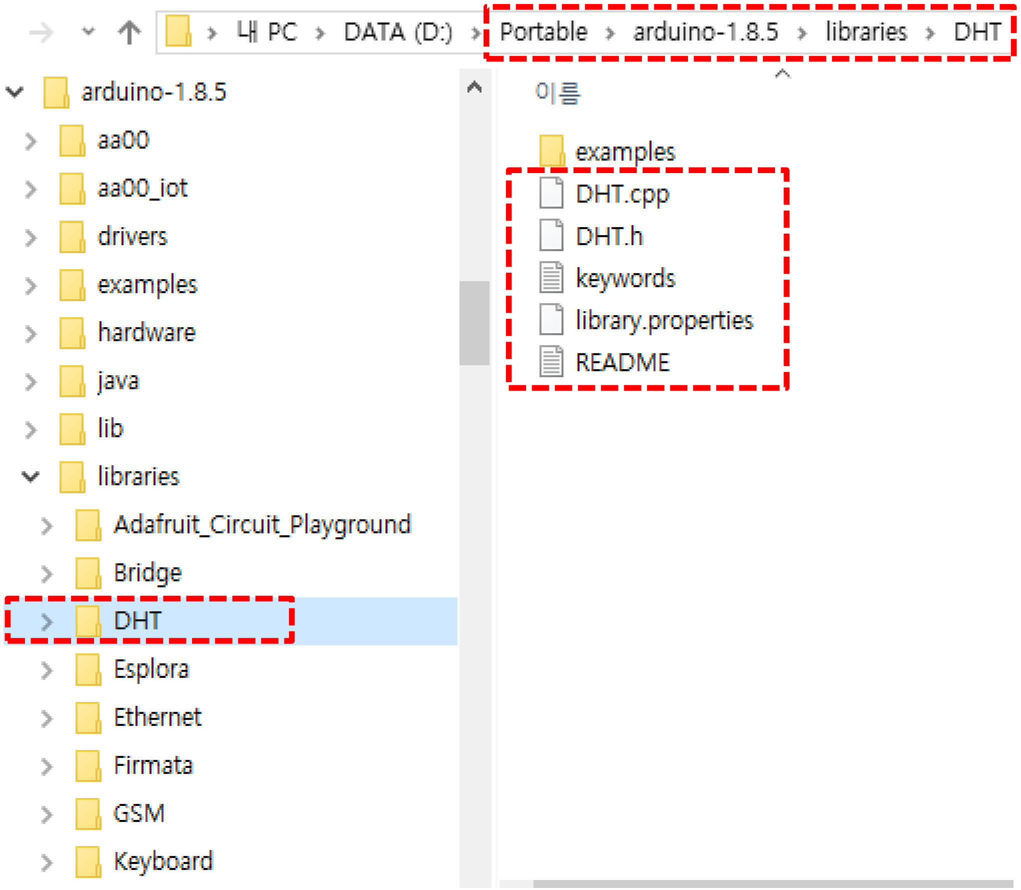
54 commits | 1 branch | 8 releases

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

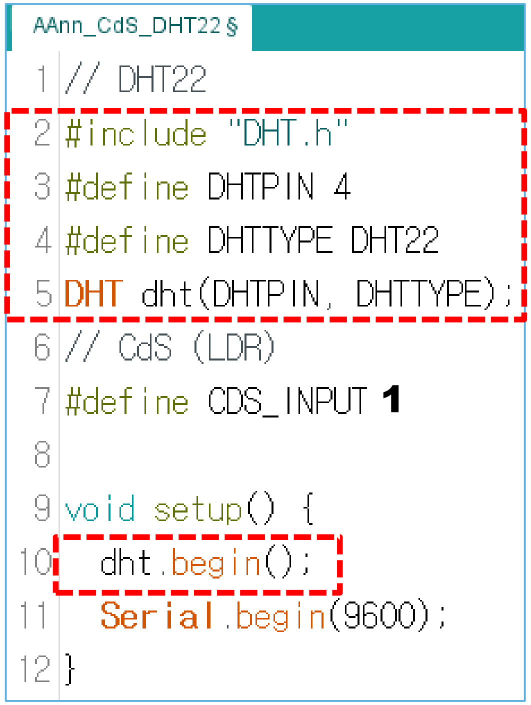


A5.7.3 DHT22 + CdS : DHT library

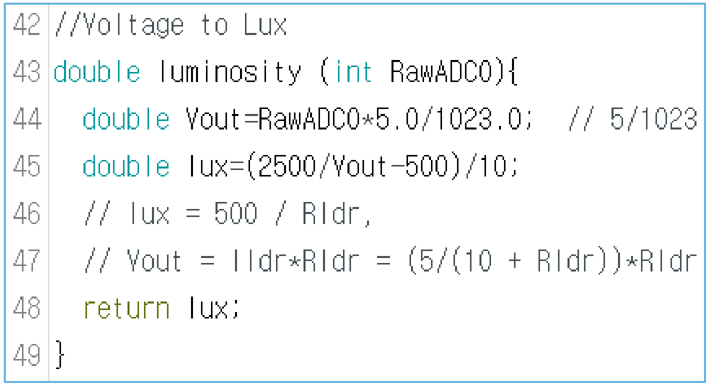


A5.7.4 DHT22 + CdS : circuit

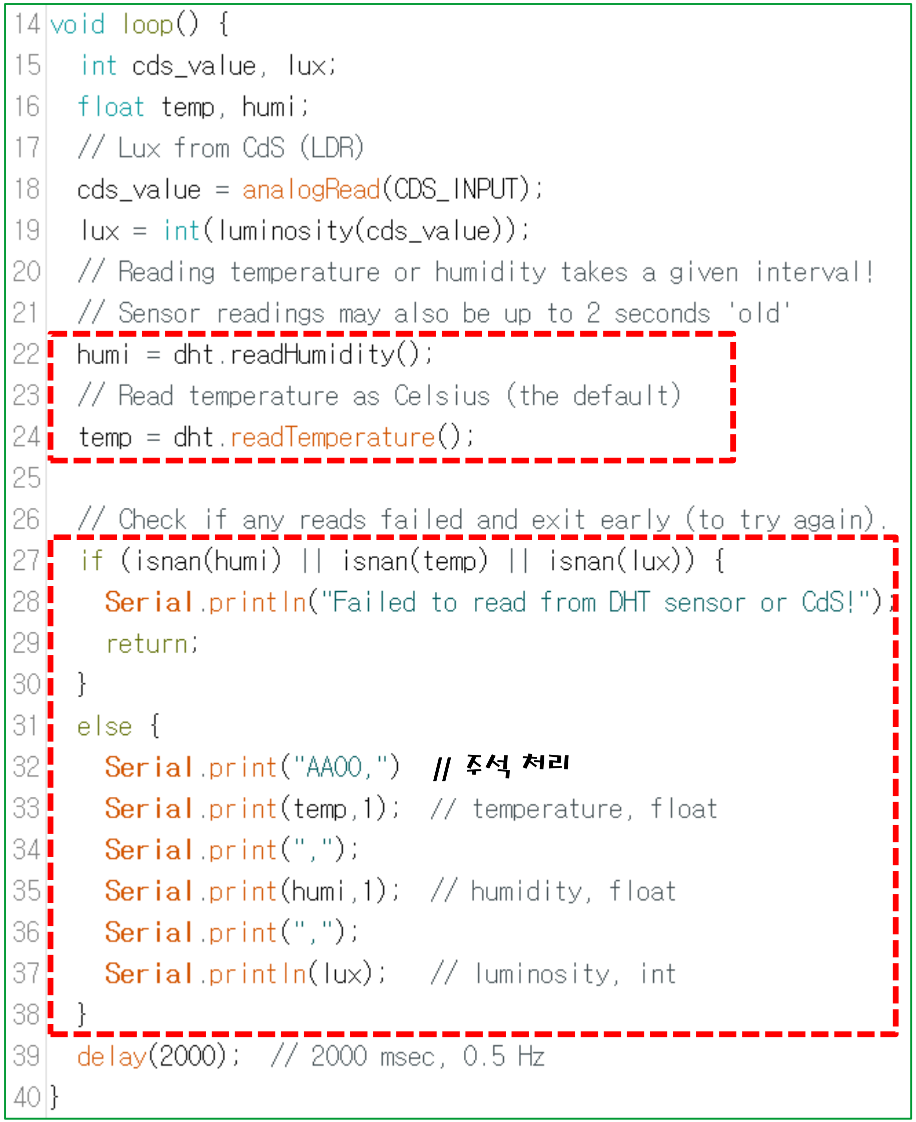
[1] Arduino code: AAnn_CdS_DHT22.ino



```
// DHT22
#include "DHT.h"
#define DHTPIN 4
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
// CdS (LDR)
#define CDS_INPUT 1

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



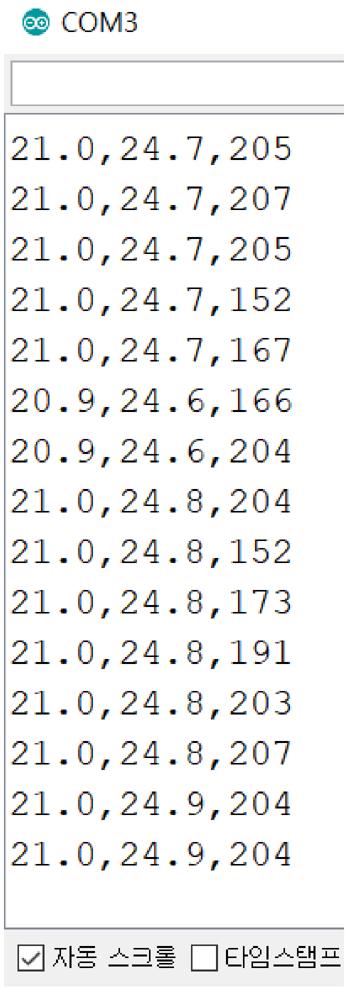
```
//Voltage to Lux
double luminosity (int RawADC0){
  double Vout=RawADC0*5.0/1023.0;  // 5/1023
  double lux=(2500/Vout-500)/10;
  // lux = 500 / Rldr,
  // Vout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
  return lux;
}
```



```
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

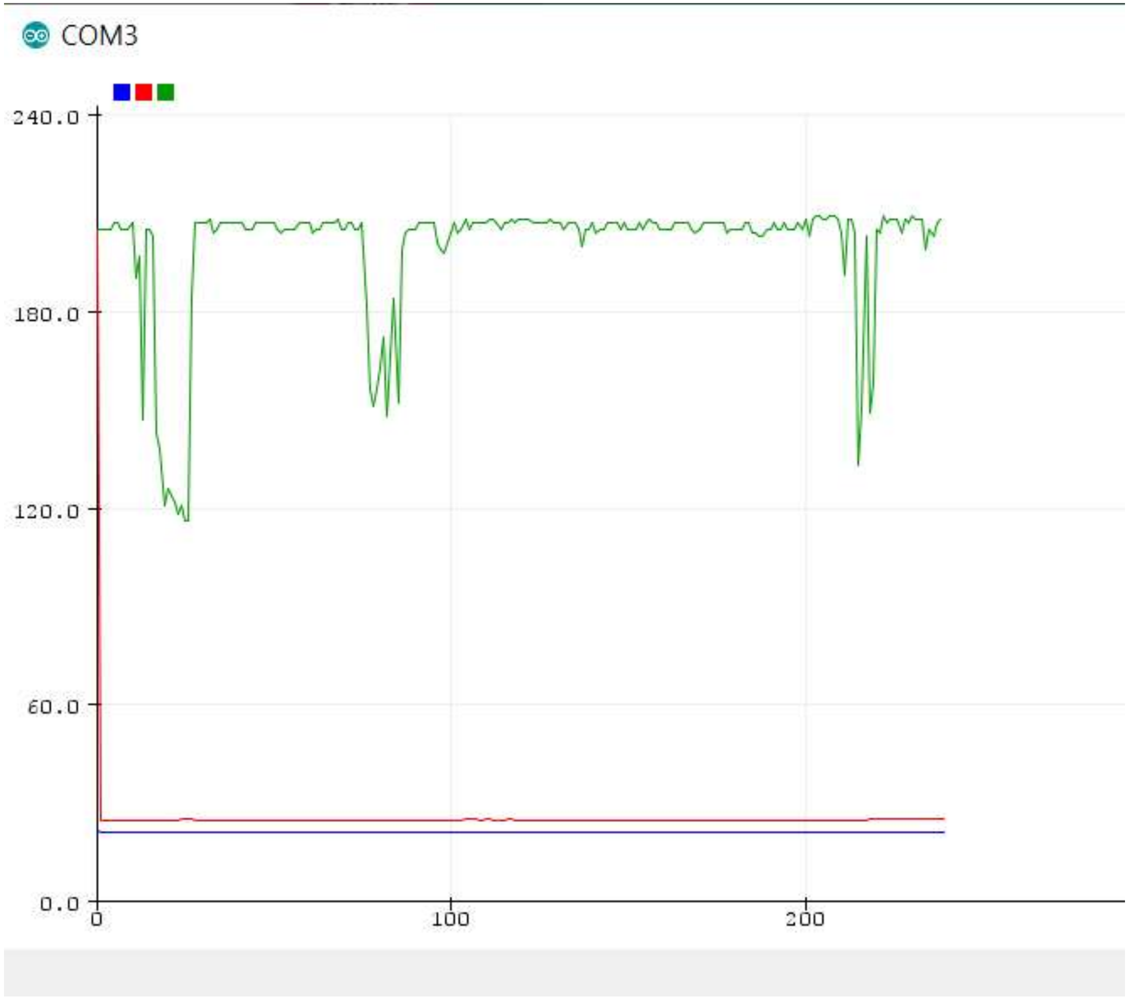
[1] Arduino code: AAnn_CdS_DHT22.ino



COM3

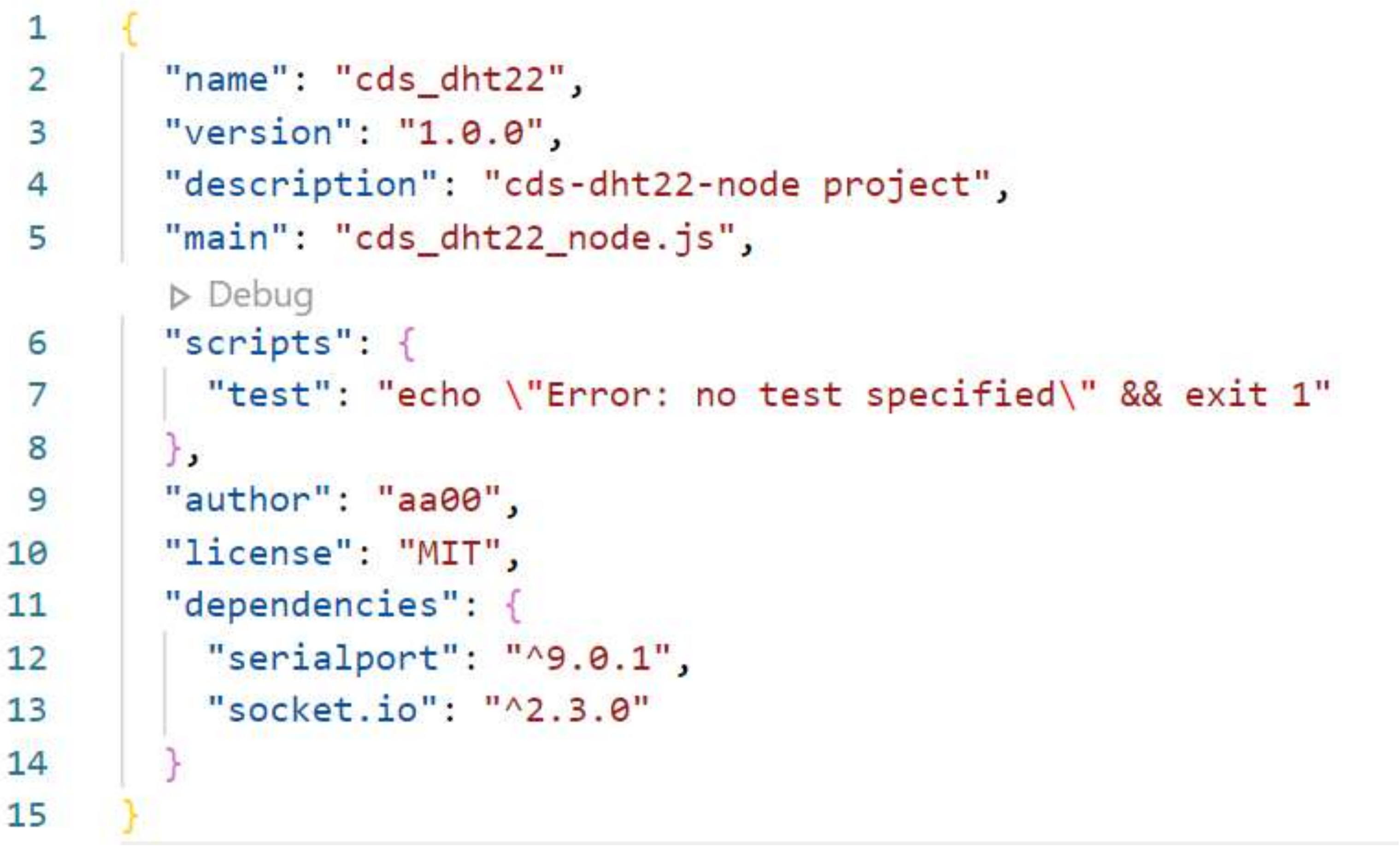
```
21.0,24.7,205
21.0,24.7,207
21.0,24.7,205
21.0,24.7,152
21.0,24.7,167
20.9,24.6,166
20.9,24.6,204
21.0,24.8,204
21.0,24.8,152
21.0,24.8,173
21.0,24.8,191
21.0,24.8,203
21.0,24.8,207
21.0,24.9,204
21.0,24.9,204
```

☑ 자동 스크롤 □ 타임스탬프



A5.7.6 DHT22 + CdS + Node.js

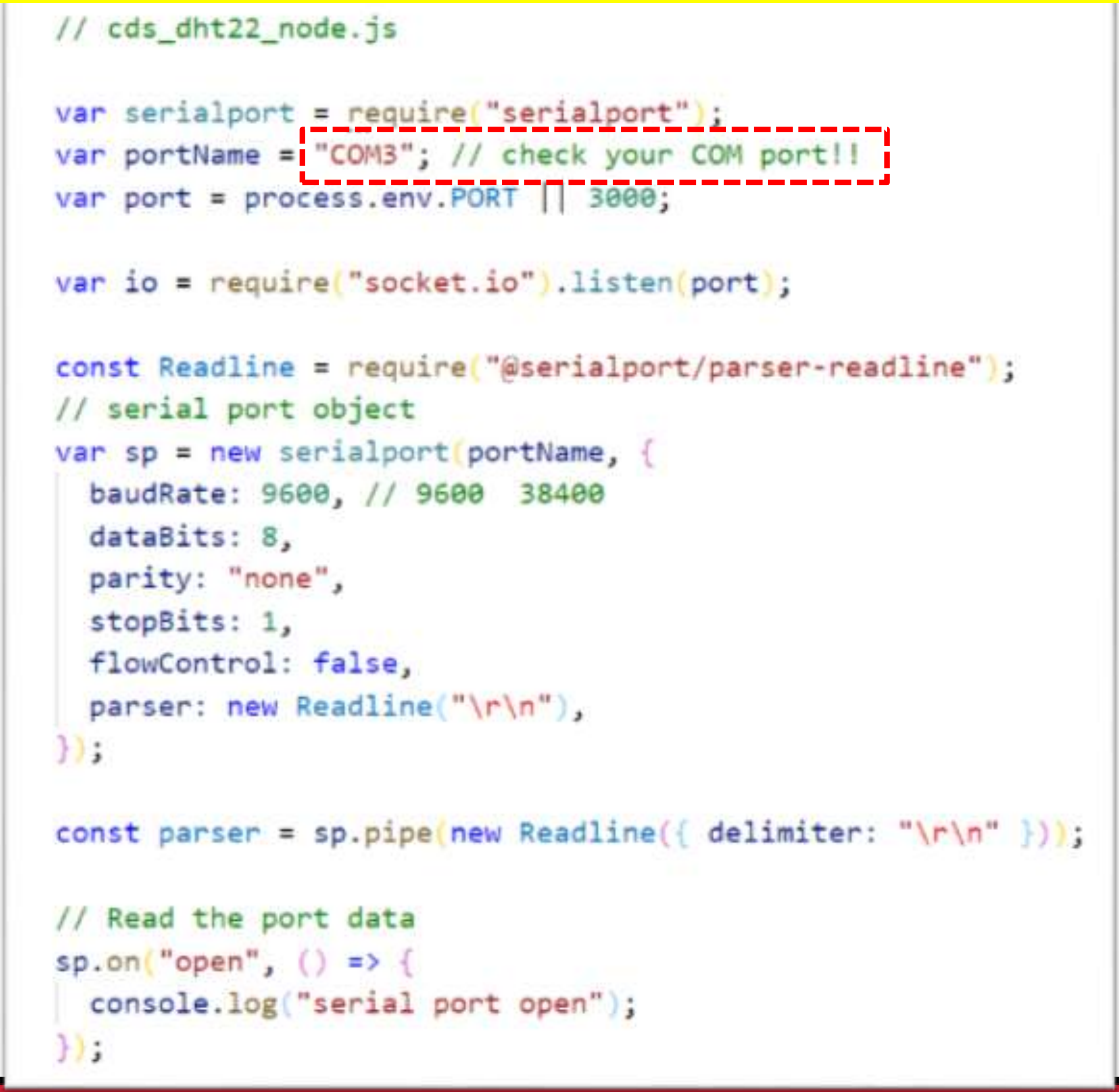
[2.1] NodeJS project: "cds-dht22-node project“ → package.json



```json
{
  "name": "cds_dht22",
  "version": "1.0.0",
  "description": "cds-dht22-node project",
  "main": "cds_dht22_node.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "aa00",
  "license": "MIT",
  "dependencies": {
    "serialport": "^9.0.1",
    "socket.io": "^2.3.0"
  }
}
```

A5.7.7 DHT22 + CdS + Node.js

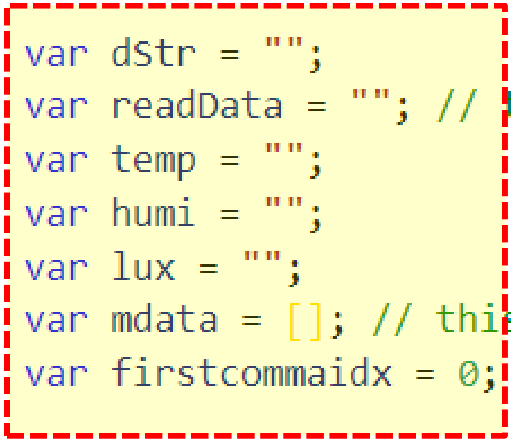
[2.2] NodeJS code: cds_dht22_node.js (← cds_tmp36_node.js를 rename)



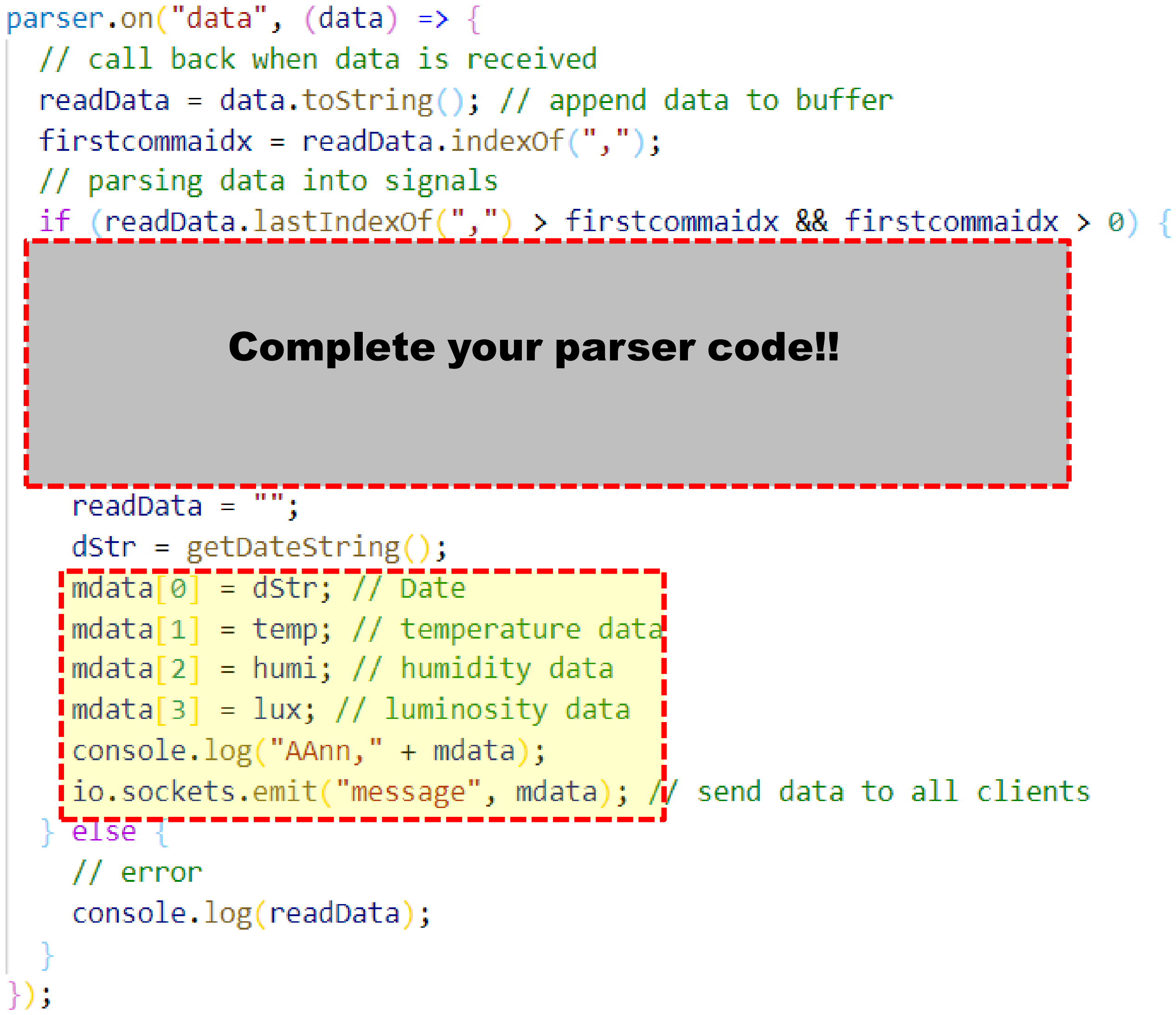
```
// cds_dht22_node.js

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;
```



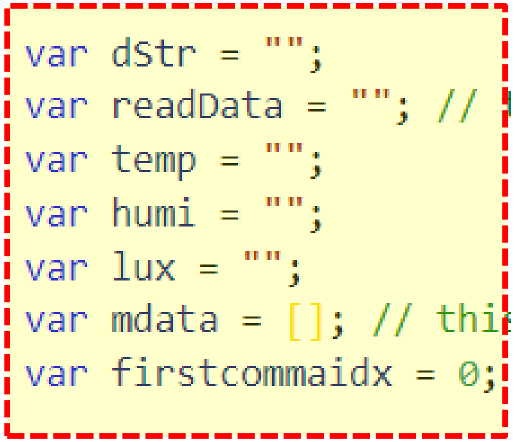
```
parser.on("data", (data) => {
  // call back when data is received
  readData = data.toString(); // append data to buffer
  firstcommaidx = readData.indexOf(",");
  // parsing data into signals
  if (readData.lastIndexOf(",") > firstcommaidx && firstcommaidx > 0) {
```

Complete your parser code!!

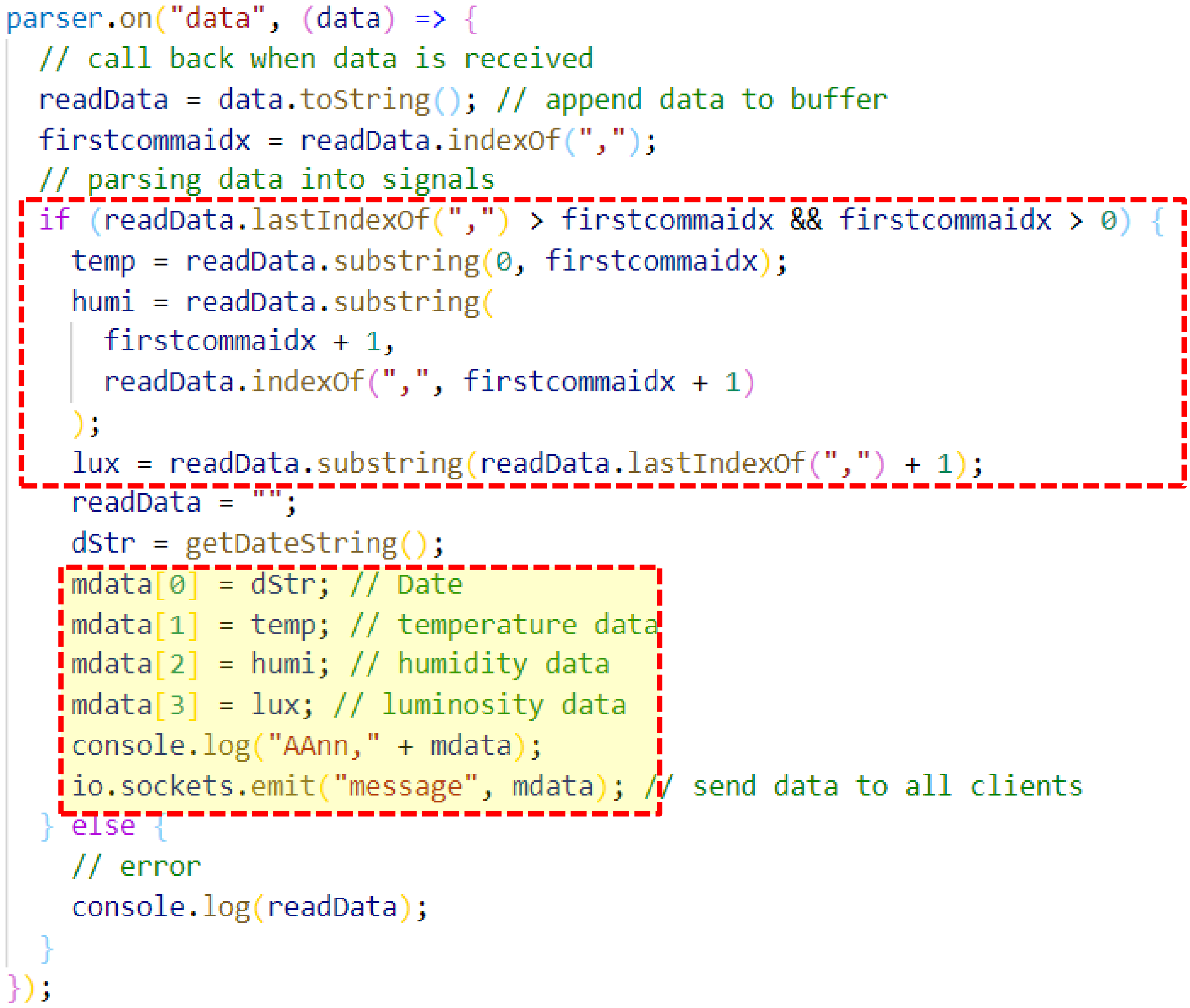
```
    readData = "";
    dStr = getDateString();
    mdata[0] = dStr; // Date
    mdata[1] = temp; // temperature data
    mdata[2] = humi; // humidity data
    mdata[3] = lux; // luminosity data
    console.log("AAnn," + mdata);
    io.sockets.emit("message", mdata); // send data to all clients
 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

A5.7.9 DHT22 + CdS + Node.js

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



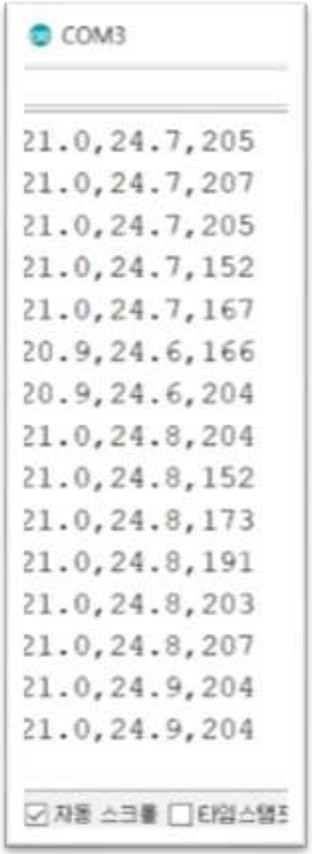
```
var dStr = "";
var readData = ""; //
var temp = "";
var humi = "";
var lux = "";
var mdata = []; // thi
var firstcommaidx = 0;
```



```
parser.on("data", (data) => {
  // call back when data is received
  readData = data.toString(); // append data to buffer
  firstcommaidx = readData.indexOf(",");
  // parsing data into signals
  if (readData.lastIndexOf(",") > firstcommaidx && firstcommaidx > 0) {
    temp = readData.substring(0, firstcommaidx);
    humi = readData.substring(
      firstcommaidx + 1,
      readData.indexOf(",", firstcommaidx + 1)
    );
    lux = readData.substring(readData.lastIndexOf(",") + 1);
    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

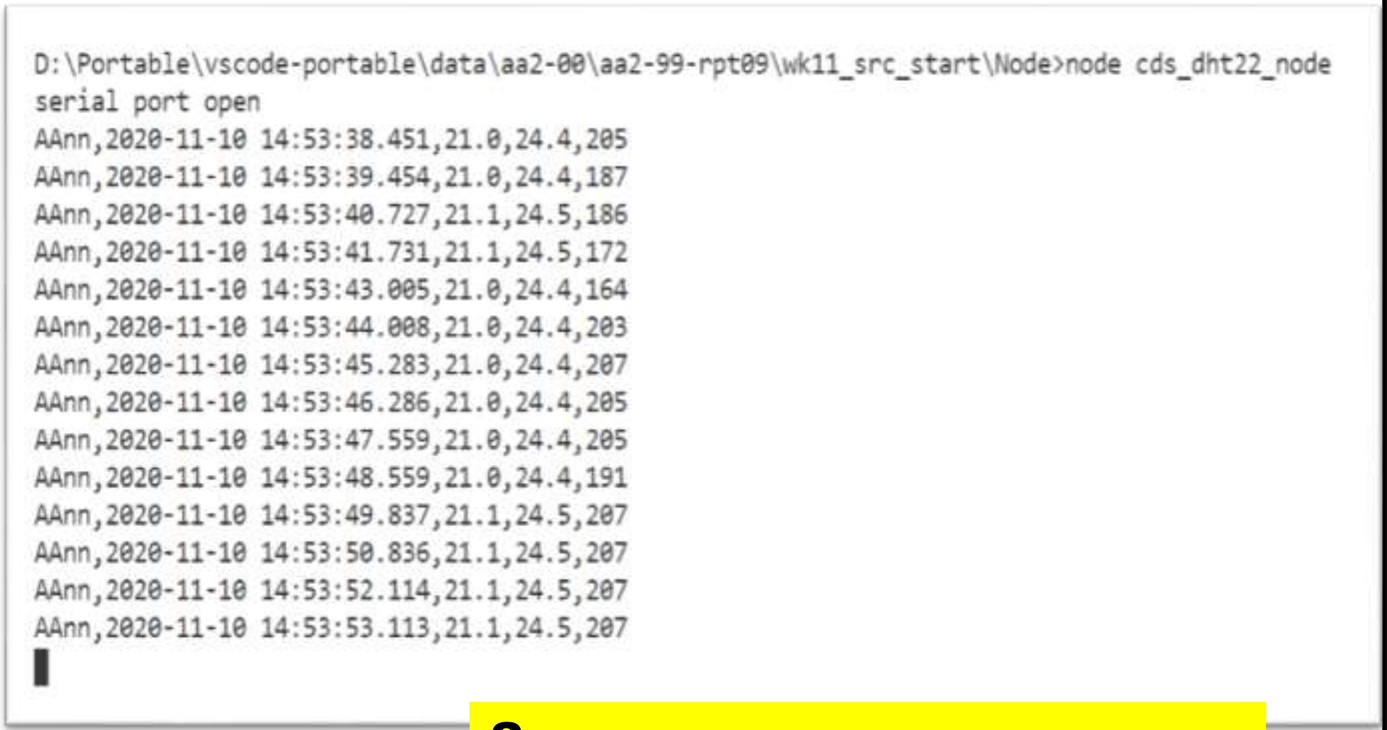
[3] Result: Parsed streaming data from dht22 & CdS (Run in Terminal)



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
```

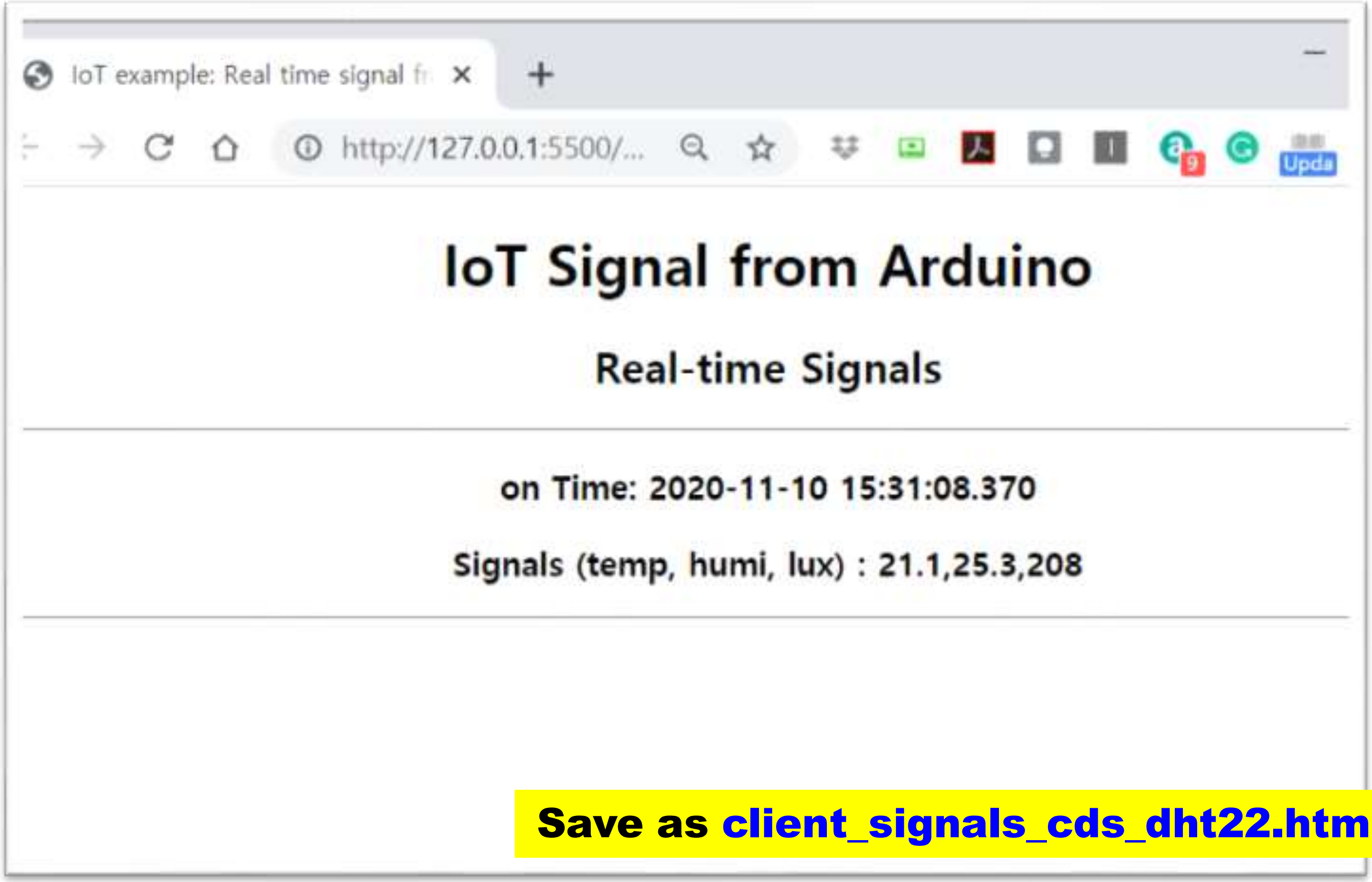
☑ 자동 스크롤 □ 타임스탬프



```
D:\Portable\vscode-portable\data\aa2-00\aa2-99-rpt09\wk11_src_start\Node>node cds_dht22_node
serial port open
AAnn,2020-11-10 14:53:38.451,21.0,24.4,205
AAnn,2020-11-10 14:53:39.454,21.0,24.4,187
AAnn,2020-11-10 14:53:40.727,21.1,24.5,186
AAnn,2020-11-10 14:53:41.731,21.1,24.5,172
AAnn,2020-11-10 14:53:43.005,21.0,24.4,164
AAnn,2020-11-10 14:53:44.008,21.0,24.4,203
AAnn,2020-11-10 14:53:45.283,21.0,24.4,207
AAnn,2020-11-10 14:53:46.286,21.0,24.4,205
AAnn,2020-11-10 14:53:47.559,21.0,24.4,205
AAnn,2020-11-10 14:53:48.559,21.0,24.4,191
AAnn,2020-11-10 14:53:49.837,21.1,24.5,207
AAnn,2020-11-10 14:53:50.836,21.1,24.5,207
AAnn,2020-11-10 14:53:52.114,21.1,24.5,207
AAnn,2020-11-10 14:53:53.113,21.1,24.5,207
```

Save as
AAnn_cds_dht22_data.png

Arduino data on network socket



IoT Signal from Arduino

Real-time Signals

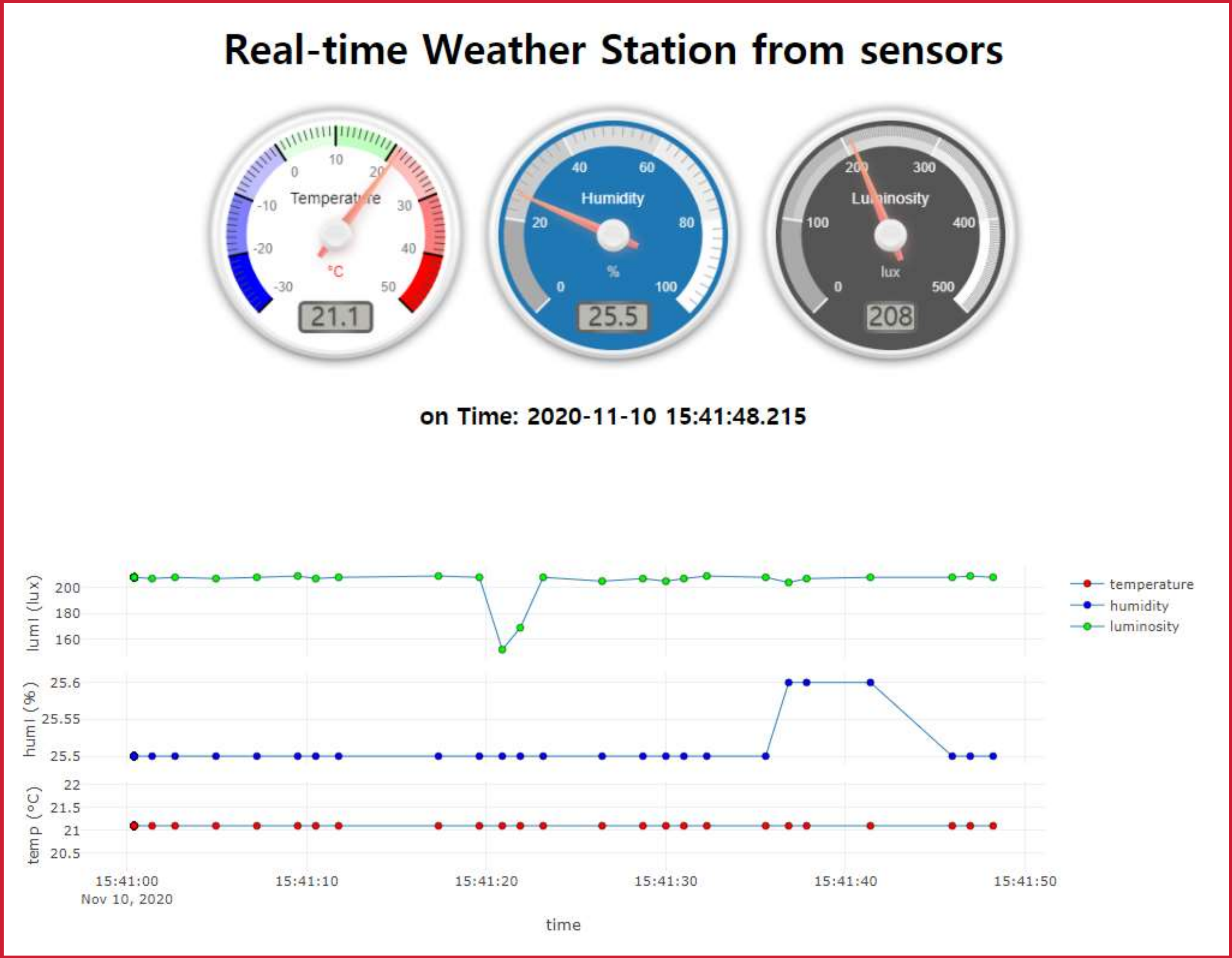
on Time: 2020-11-10 15:31:08.370

Signals (temp, humi, lux) : 21.1,25.3,208

Save as client_signals_cds_dht22.html

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

WEB client： client_cds_dht22.html

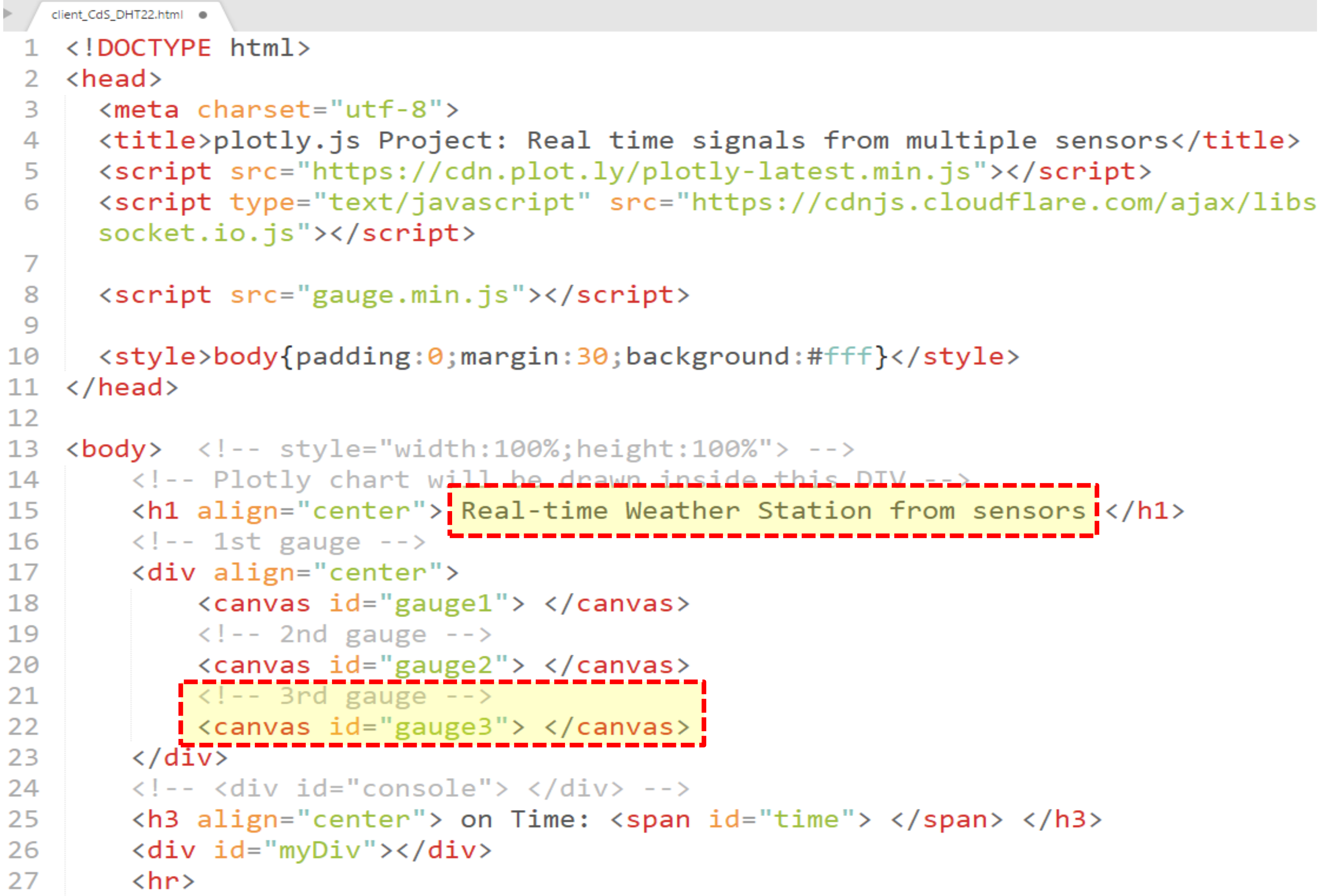


Real-time Weather Station from sensors

on Time: 2020-11-10 15:41:48.215

A5.8.1 DHT22 + CdS + Node.js

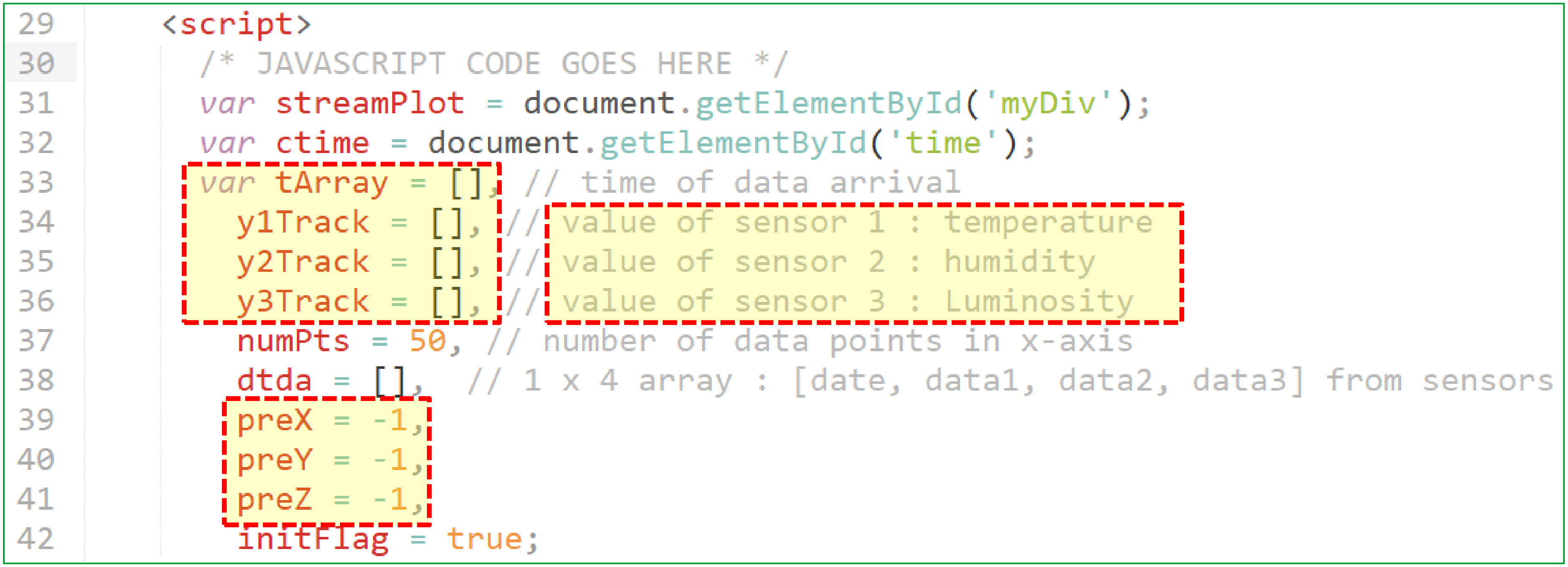
[4.1] WEB client: client_cds_dht22.html



```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



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

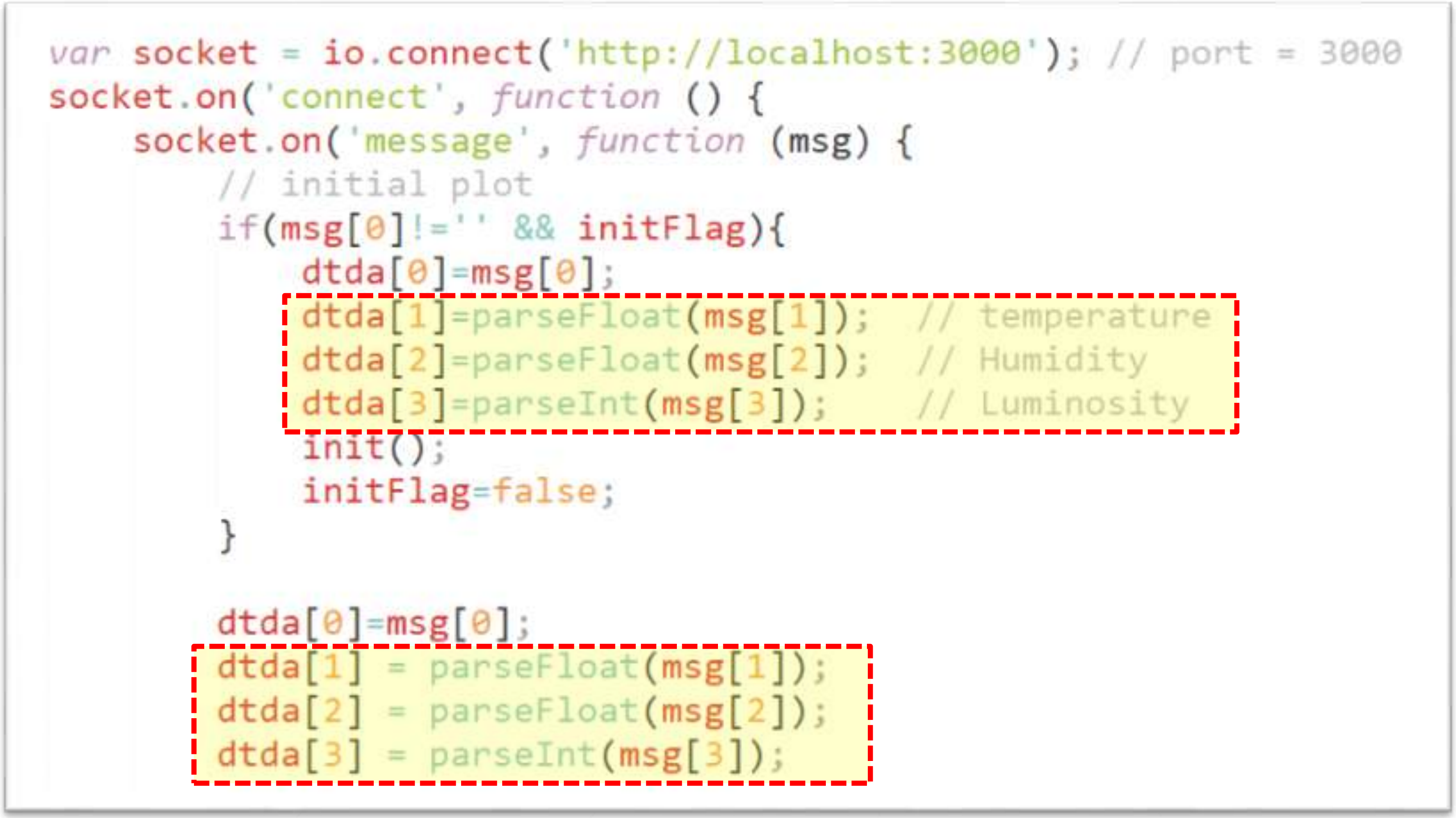
Check points: tArray

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]);
```

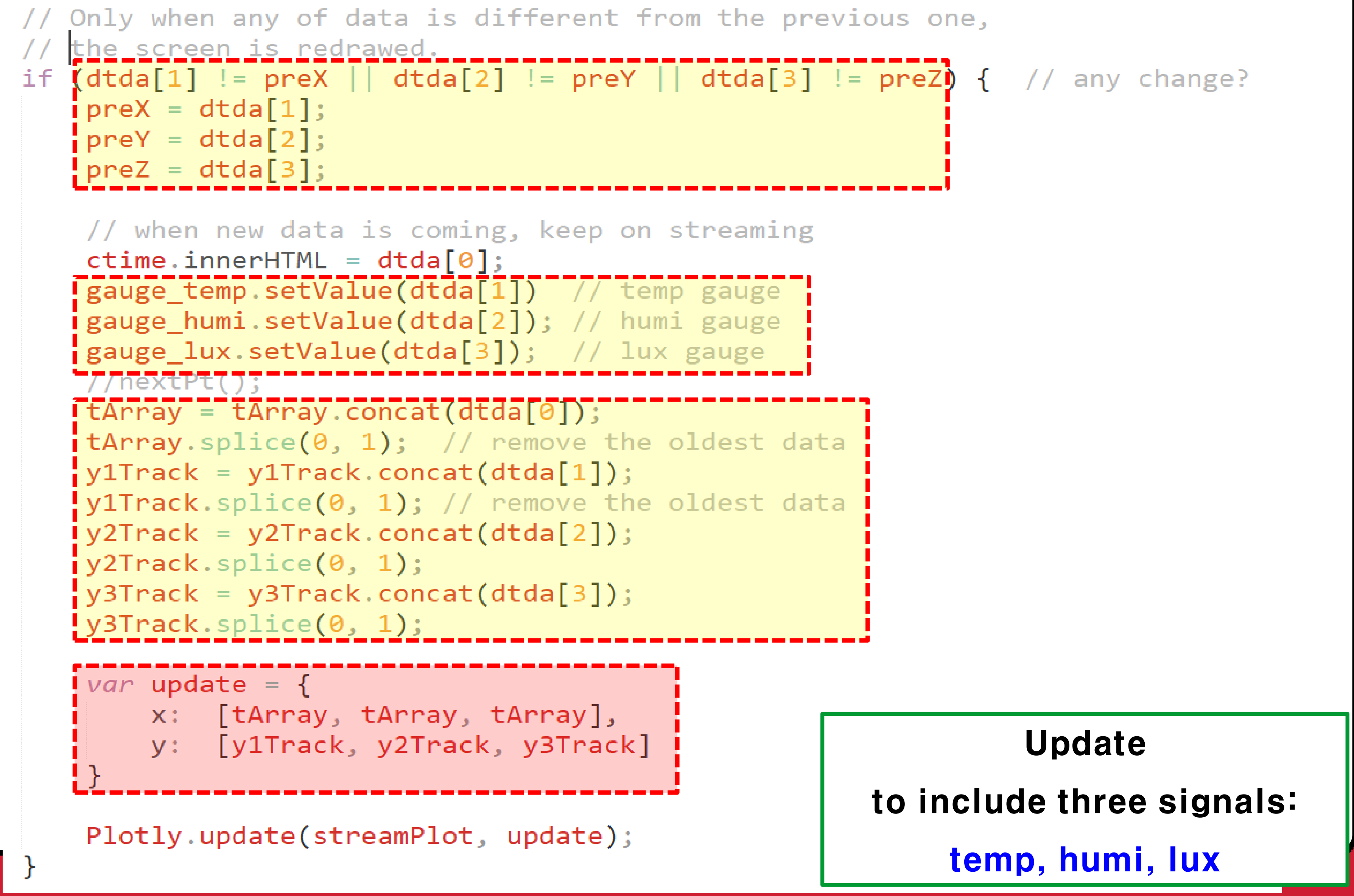
Update

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);
}
```

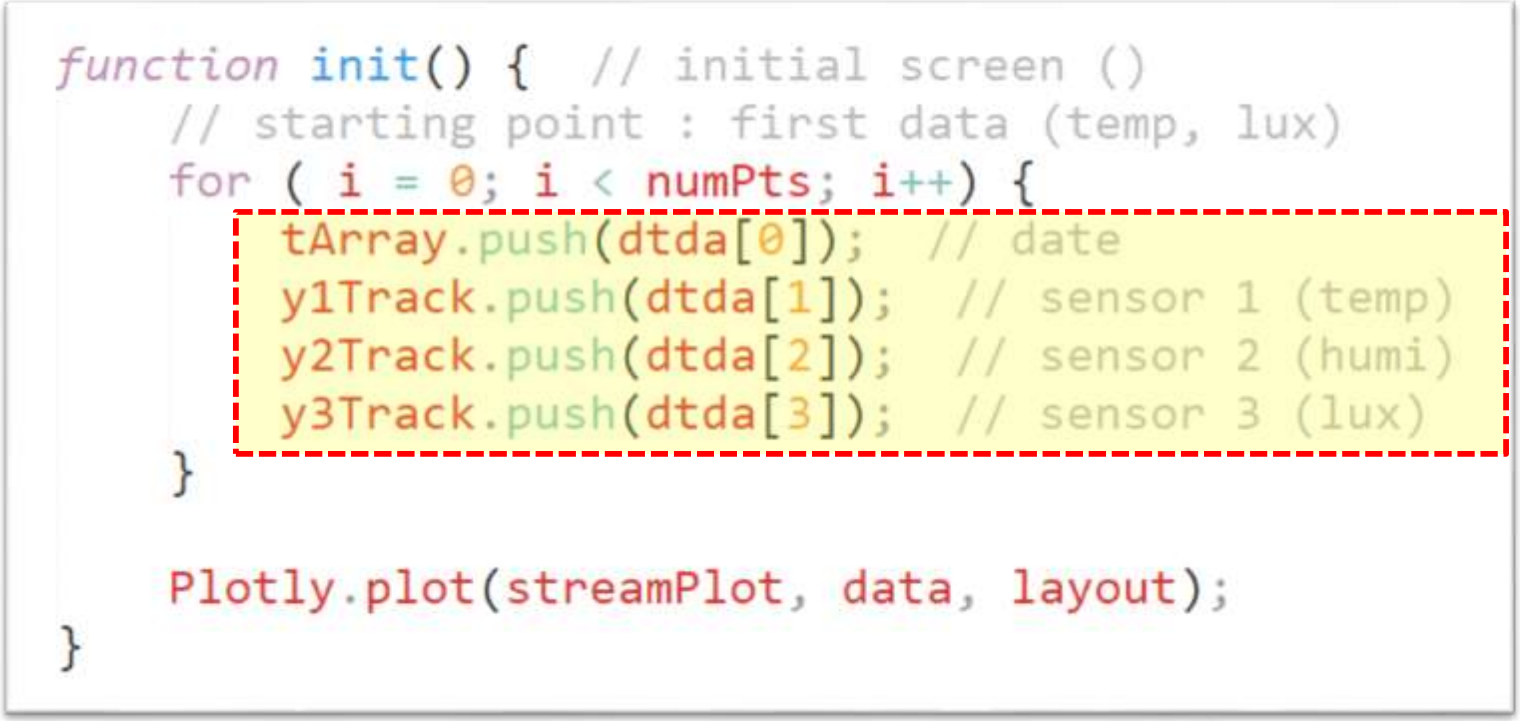
Update

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);
}
```

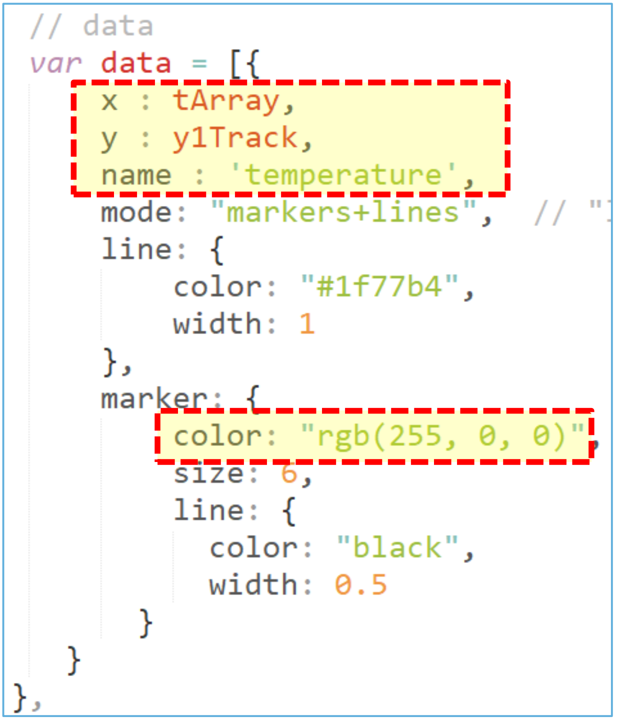
Update

to include three signals:

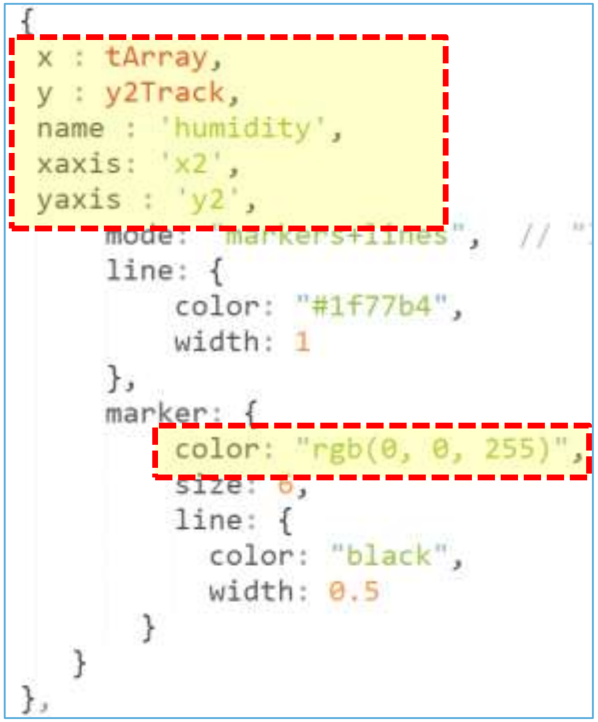
temp, humi, lux

A5.8.6 DHT22 + CdS + Node.js

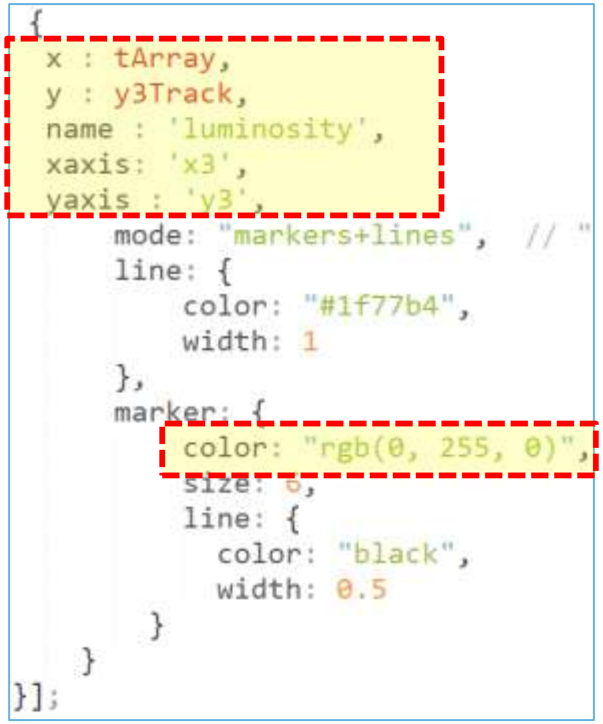
[4.6] WEB client: client_cds_dht22.html - data



```
// 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
        }
    }
}];
```

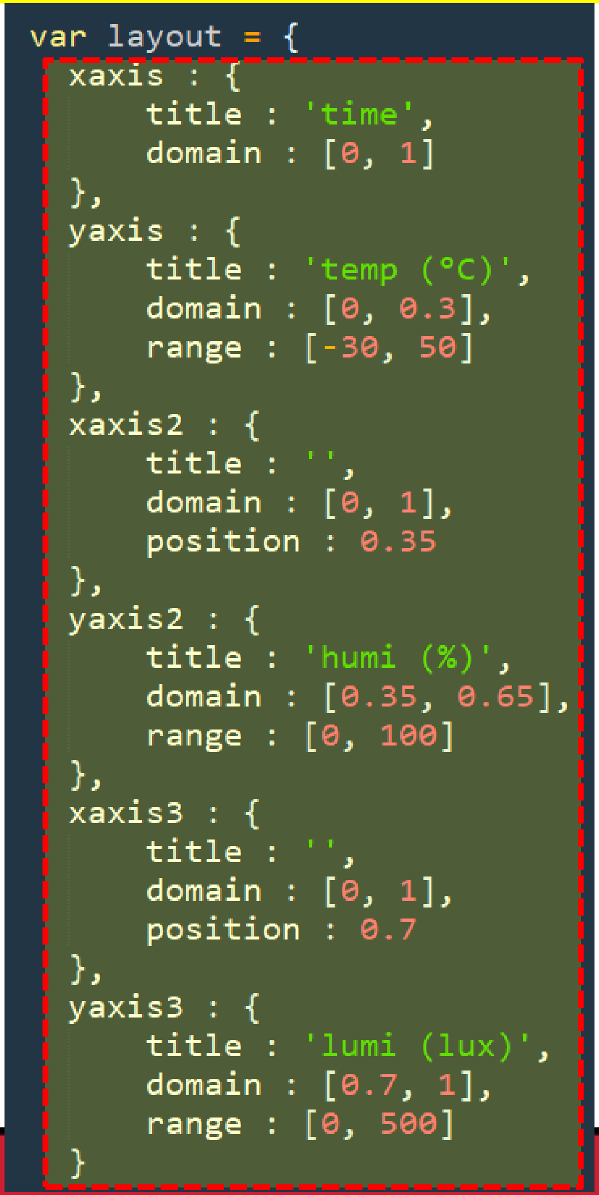
Update data

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 &

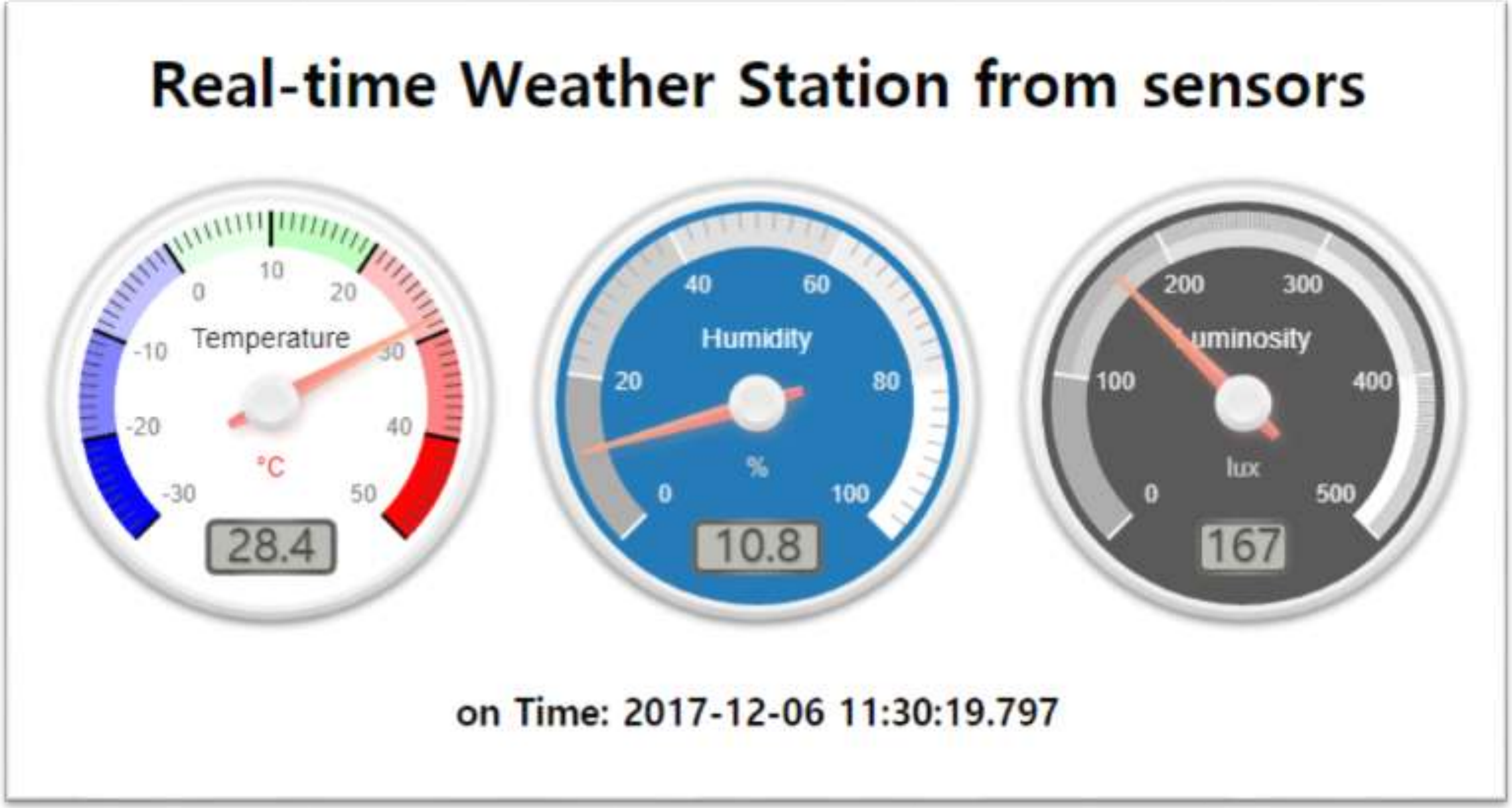
   position.

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



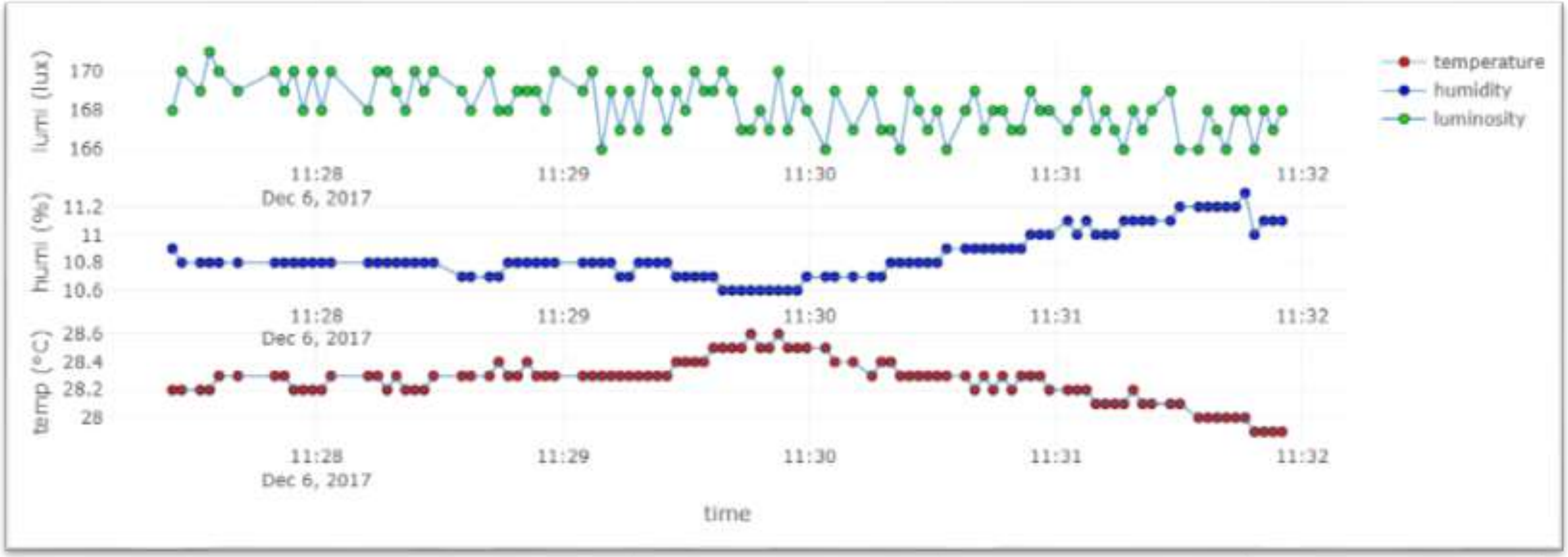
Real-time Weather Station from sensors

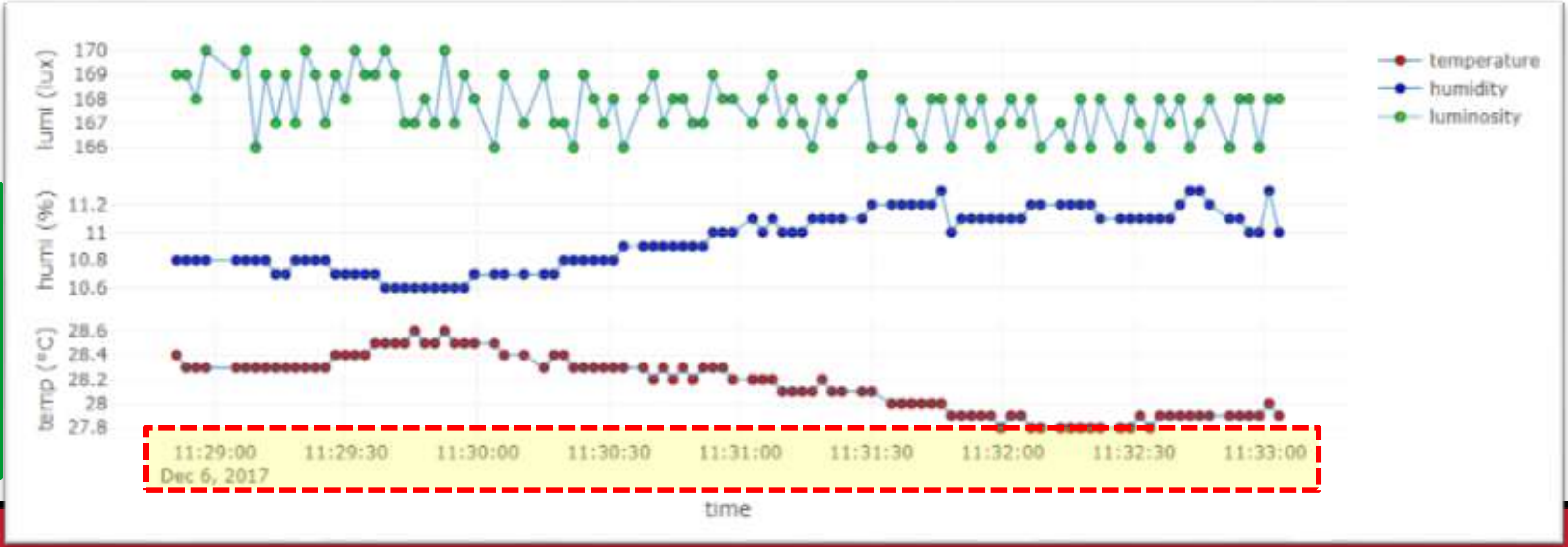
on Time: 2017-12-06 11:30:19.797

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)

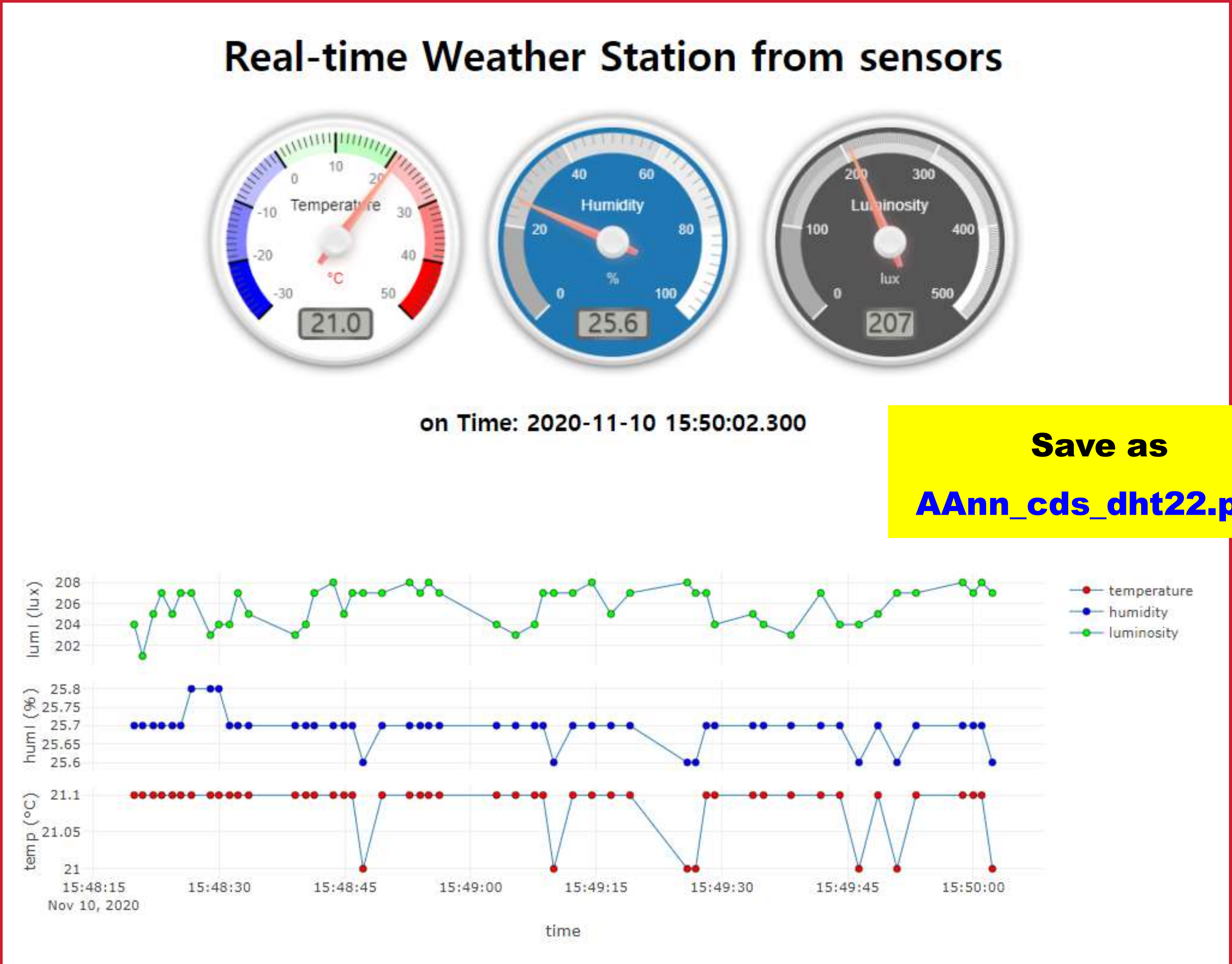




[Hint]

Plot.ly

WEB client： client_cds_dht22.html

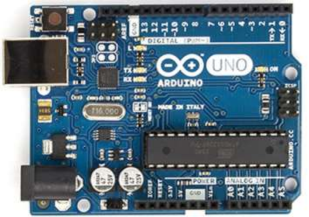


Real-time Weather Station from sensors

on Time: 2020-11-10 15:50:02.300

Save as

AAnn_cds_dht22.png



[Practice]

- [wk11]
  - RT Data Visualization with node.js
  - Multiple data and Usage of gauge.js
  - Complete your real-time WEB charts
  - Upload folder: aax-nn-rpt09
  - Use repo “aax-nn” in github

wk11 : Practice : aax-nn-rpt09

[Target of this week]

- Complete your works
- Save your outcomes and upload outputs in github

제출폴더명 : **aax-nn-rpt09**

- 제출할 파일들

① **AAnn_DS_cds_tmp36.png**

② **AAnn_cds_dht22_data.png**

③ **AAnn_cds_dht22.html**

④ **AAnn_cds_dht22.png**

⑤ **All *.ino**

⑥ **All *.js**

⑦ **All *.html**

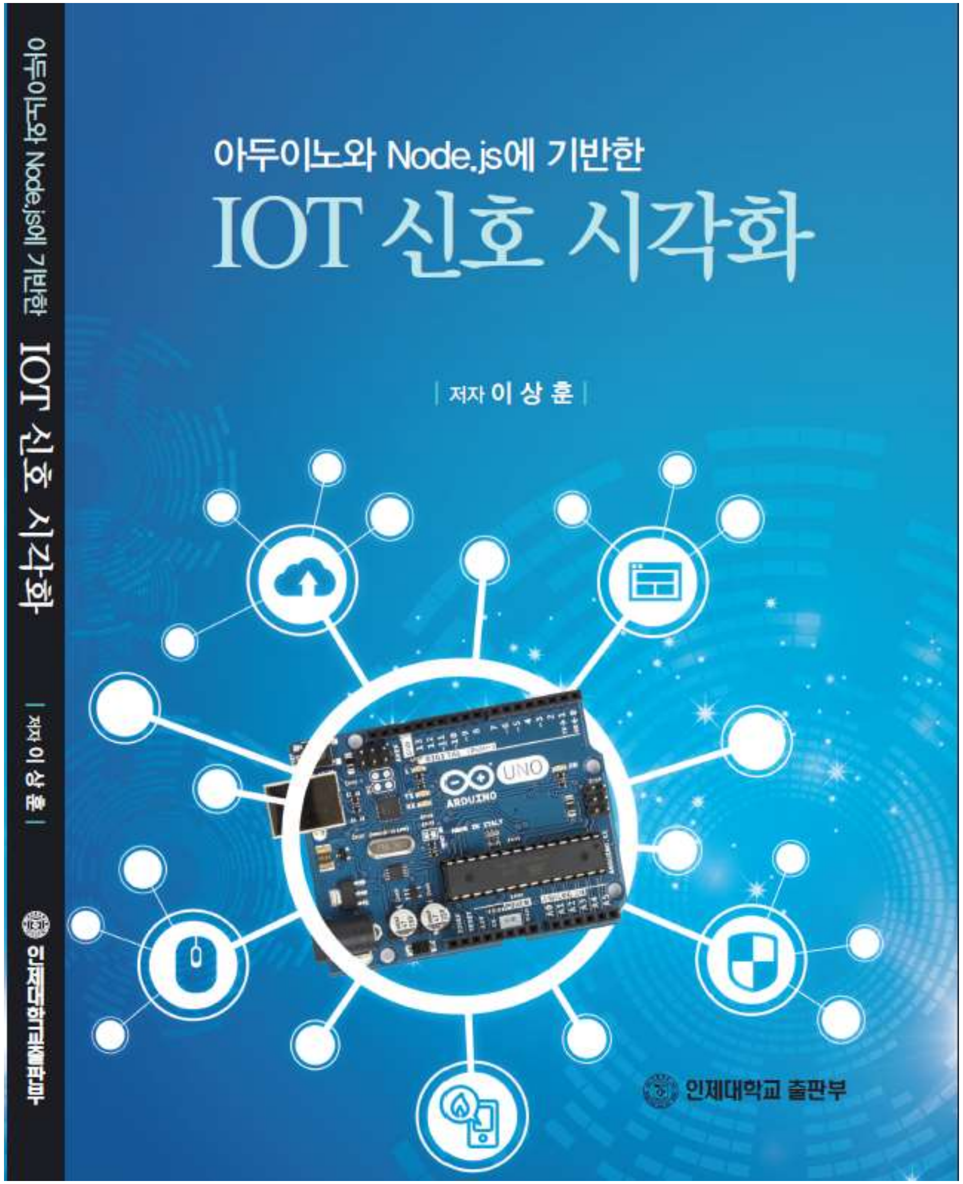
Email : chaos21c@gmail.com

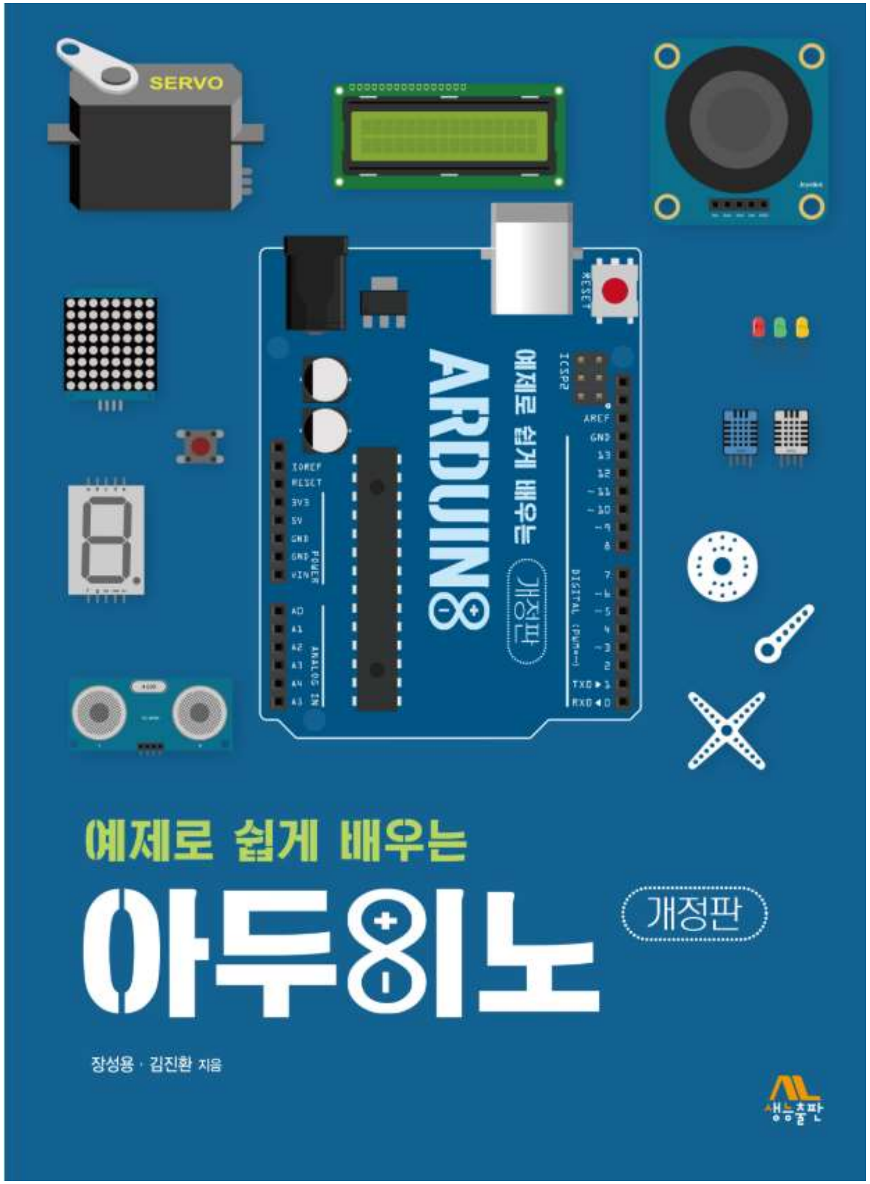
Lecture materials

References & good sites

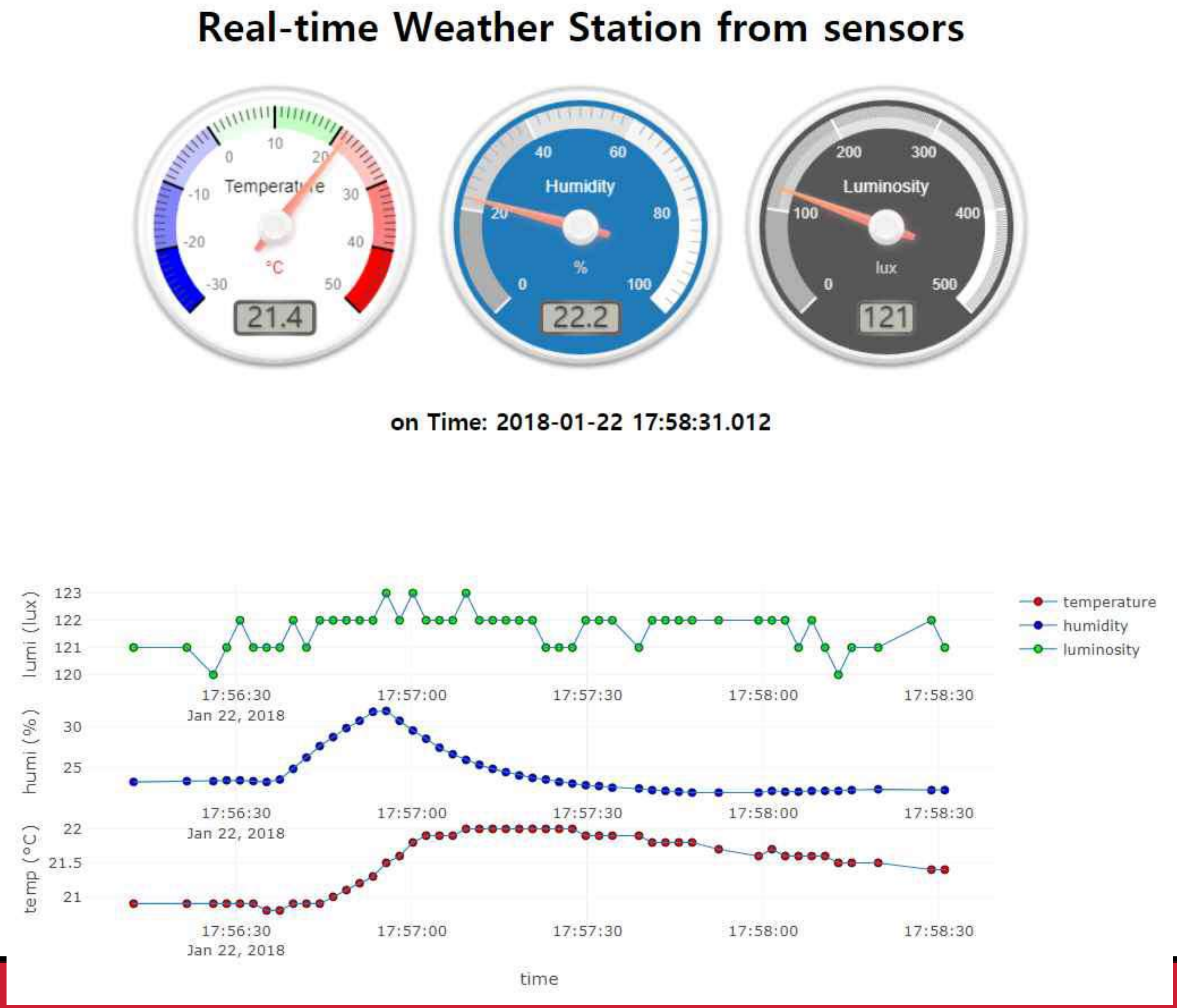
- ✓ http://www.arduino.cc Arduino Homepage
- ✓ http://www.nodejs.org/ko Node.js
- ✓ https://plot.ly/ plotly
- ✓ https://www.mongodb.com/ MongoDB
- ✓ http://www.w3schools.com By w3schools
- ✓ http://www.github.com GitHub

주교재 및 참고도서


아두이노와 Node.js에 기반한
IOT 신호 시각화
| 저자 이 상 훈 |
인제대학교 출판부
아두이노와 Node.js에 기반한 IOT 신호 시각화 | 저자 이 상 훈 | 인제대학교출판부


SERVO
예제로 쉽게 배우는
아두이노
개정판
장성용 · 김진환 지음
생능출판
예제로 쉽게 배우는 ARDUINO 개정판

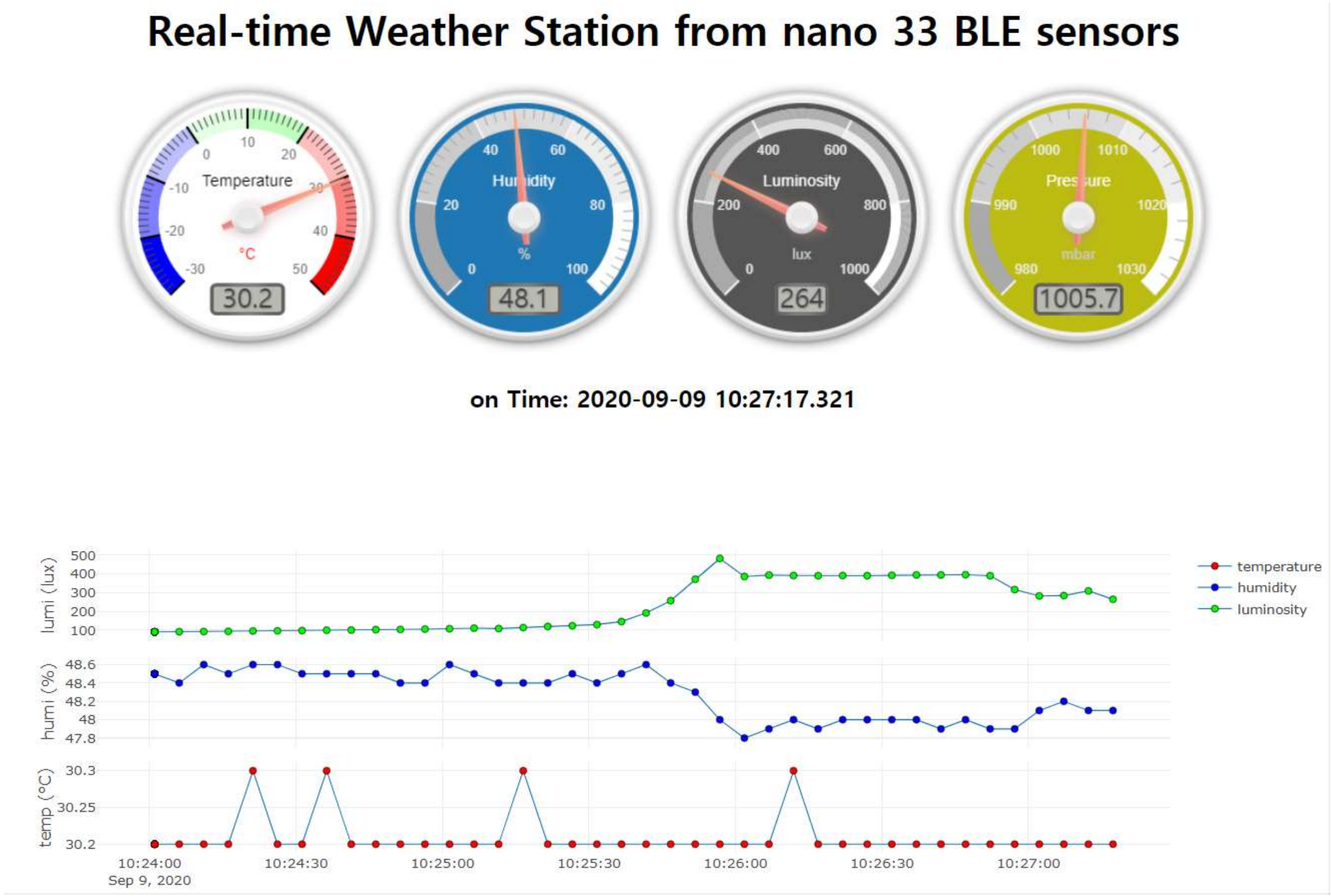
Target of this class



Real-time Weather Station from sensors

on Time: 2018-01-22 17:58:31.012

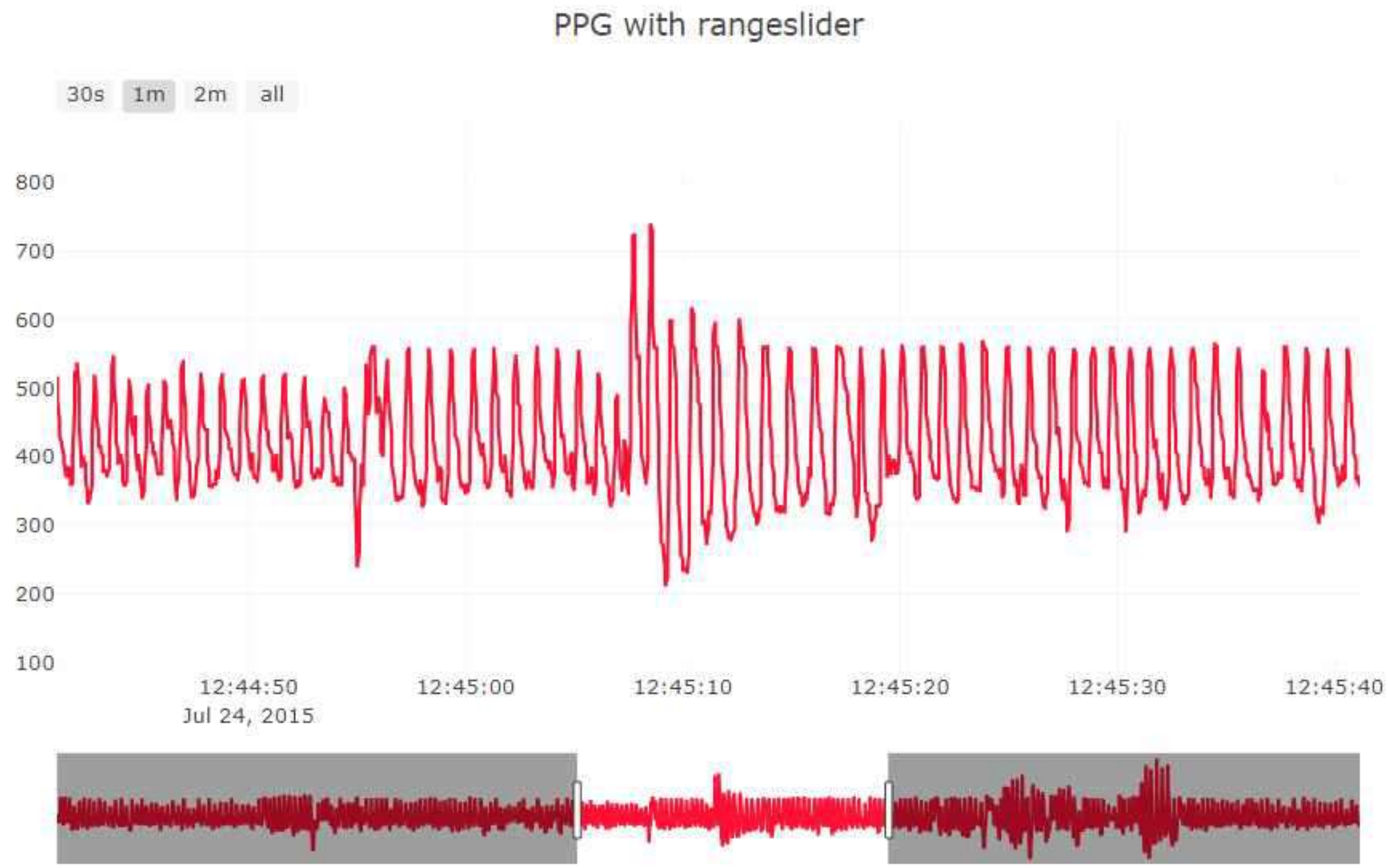
Target of this class



Real-time Weather Station from nano 33 BLE sensors

on Time: 2020-09-09 10:27:17.321

Another target of this class


PPG with rangeslider
30s
1m
2m
all
12:44:50
Jul 24, 2015
12:45:00
12:45:10
12:45:20
12:45:30
12:45:40