



[wk08]

Data Visualization I

- plotly.js charts

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







2nd semester, 2023

Email: chaos21c@gmail.com

Drone-IoT-Comsi, INJE University



My ID

ID를 확인하고 github에 repo 만들기

ID	성명
AA01	강동하
AA02	고서진
AA03	김민재
AA04	김예원
AA05	김주호
AA06	김창욱
AA07	김현서
AA08	박종혁
AA09	서명진
AA10	유동기
AA11	
AA12	이근보
AA13	정호기

위의 id를 이용해서 github에 repo를 만드시오.

Option: ^{아두이노}응용 실습 과제 – AAnn

Public, README.md check





[Mid-Exam]

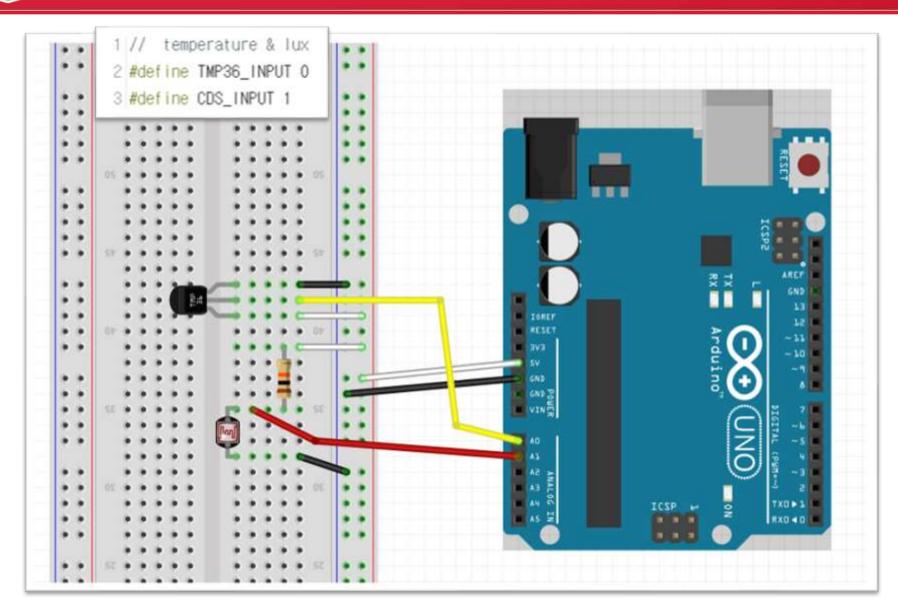
◆ [wk07]

- cds + tmp36 회로 구성 → 조도,온도
- > 습도(humi)를 40~90 범위로 simulation (cds_tmp36_humi.ino)
- 조도,습도,온도 순서로 3개의 신호 처리 (cds_tmp36_humi.js)
- > 조도,습도,온도 웹 모니터링 (client_cds_tmp36_humi.html)





A4.3.1 TMP36 + CdS: circuit







A4.5.6 multi-signals + Node project: WEB

[Web monitoring] client_cds_tmp36_humi.html



IoT Signal from Arduino

Real-time Signals

on Time: 2021-10-05 14:27:23.536

Signals (조도,습도,온도) : 161,41,22

Save as AAnn_cds_tmp36_humi_WEB.png





[Practice]

- ♦ [wk07]
- Arduino sensors + Node.js
- Complete your project
- Upload folder: aann-rpt07
- Use repo "aann" in github

wk07: Mid-Exam: aann-rpt07





- **◆** [Target of this week]
 - Complete your works & update your repo.
 - Save your outcomes and upload outputs in github repo.

제출폴더명: aann-rpt07 - 압축할 파일들 ① cds_tmp36_humi.ino ② cds_tmp36_humi.js ③ cds_tmp36_humi_WEB.png ④ client_cds_tmp36_humi.html ⑤ NO node_modules folder



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)



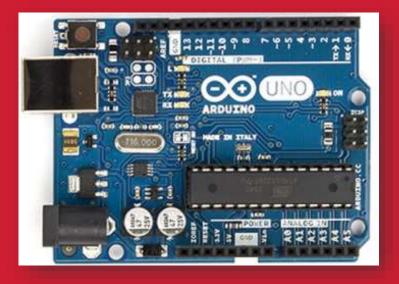






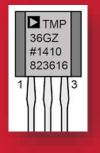


Arduino

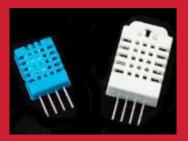


Sensors

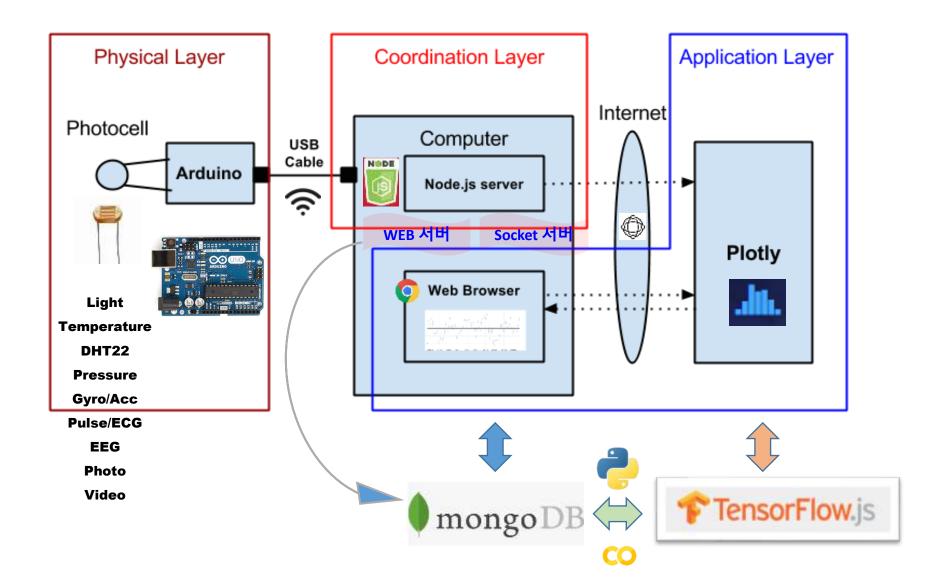
+ Node.js







Layout [H S C]



on WEB monitoring Arduino data

IoT Signal from Arduino

Real-time Signals

on Time: 2021-10-06 09:49:49.818

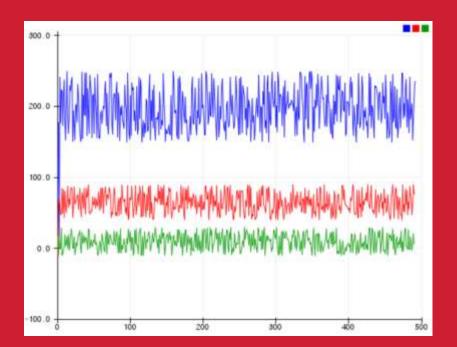
Signals (조도,습도,온도): 166,60,-5



[DIY] Multi-signals

다중신호 시뮬레이션

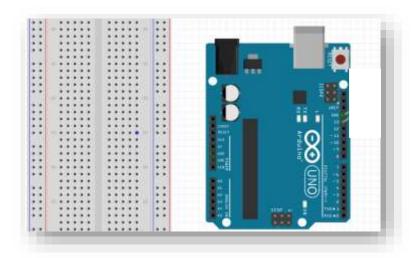
+ node.js







DIY - 스케치



아두이노에서 LED와 저항을 모두 제거하고 USB만 컴퓨터와 연결한다.

전자 소자 연결 없이 마구잡이 수 생성 함수를 이용해서 조도, 습도, 온도에 해당되는 3개의 신호를 만든다.

온도는 값의 범위를 -10 ~ 30, 습도는 40 ~ 90, 그리고 조도는 150 ~ 250 으로 가상적 으로 설정한다.

직렬통신 모니터링을 이용해서 세 개의 신호의 변화를 모니터링 하는 코드를 만들어 결과를 확인한다.

▶ 스케치 구성

- 1.3 개의 신호를 담을 변수를 초기화한다.
- 2. setup()에서 직렬 통신 속도를 9600 bps 로 설정하고 컴퓨터와 연결한다.
- 3. loop()에서 마구잡이 수를 세 개 발생시켜서 직렬 통신으로 3 개의 pwm 값을 각각 컴퓨터로 전송한다.





DIY - code

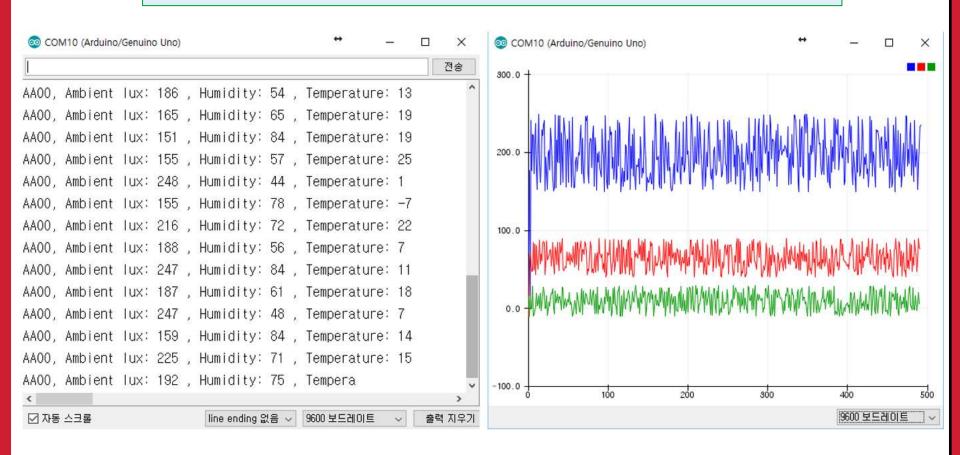
```
10 // the setup routine runs once when you press reset:
11 void setup() {
    // initialize serial communication at 9600 bits per second:
   Serial.begin(9600);
13
14 }
15
16 // the loop routine runs over and over again forever:
17 void loop() {
18 // Multi signals
19 humi = random(40,90);
20 temp = random(-10, 30);
21 lux = random(150,250);
   Serial.print("AA00, Ambient lux: ");
    Serial.print(lux);
24
    Serial.print(" , Humidity: ");
    Serial print (humi);
    Serial.print(" , Temperature: ");
    Serial.println(temp);
    delay(500); // delay in between reads for stability
28
29 }
```



DIY - result

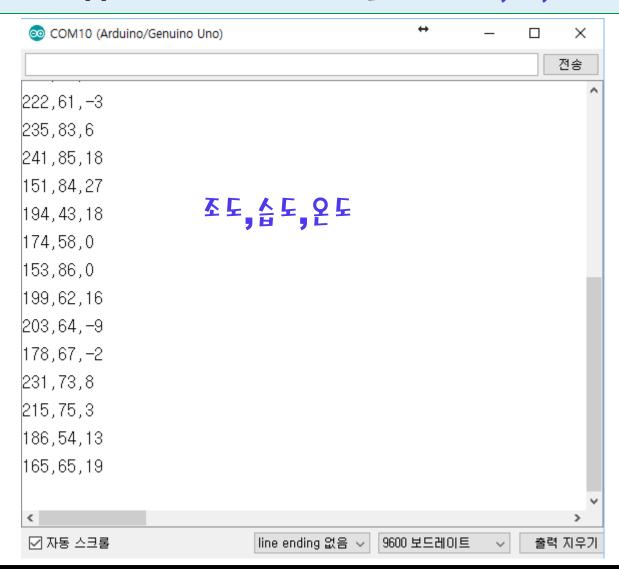
DIY 결과

가상적인 세 개의 센서신호 시뮬레이션:조도(위), 습도(중간), 온도(아래).





DIY 결과 [1] : 가상적인 세 개의 센서신호 시뮬레이션 → 조도, 습도, 온도







DIY 결과 [2] : 가상적인 세 개의 센서신호 시뮬레이션 → 조도, 습도, 온도를 Node.js로 처리

[1 단계] Node cmd

- 1. Make multi_signals node project
- md multi_signals in iot folder
- cd multi_signals
- 2. Go to multi_signals subfolder
- > npm init

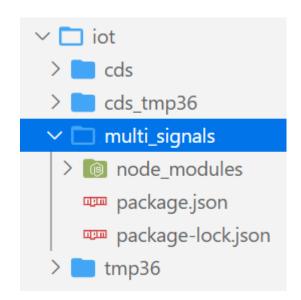
name: multi_signals

description: multi-signals-node project

entry point : aann_multi_signals.js

author: aann

- 3. Install node modules
- npm install –save serialport@9.2.4
- npm install –save socket.io@2.4.1





```
DIY 결과 [2] : 가상적인 세 개의 센서신호 시뮬레이션 → 조도, 습도, 온도를 Node.js로 처리
```

```
Recycling code:
Save cds_tmp36_node.js as
aann_multi_signals.js in multi_signals subfolder
Update code
```

```
var dStr ='';
var readData='';
var temp='';
var humi='';
var lux='';
var mdata=[];
var firstcommaidx = 0;
var secondcommaidx= 0;
```



DIY 결과 [2] : 가상적인 세 개의 센서신호 시뮬레이션 → 조도, 습도, 온도를 Node.js로 처리

```
parser.on("data", (data) => {
 // call back when data is received
 readData = data.toString();
 firstcommaidx = readData.indexOf(",");
 secondcommaidx = readData.indexOf(",", firstcommaidx + 1);
 if (firstcommaidx > 0) 
    아두이노가 직렬통신으로 전송하는 2 개의 comma (,)로 구분된
    조도, 습도, 온도 데이터 메시지를 parsing 하여 mdata 배열에 담는 코드를
               하셔하시요.
    substring() 함수에서 firstcommaidx, secondcommaidx를 잘 이용하시오.
   console.log("AA00," + mdata);
   io.sockets.emit("message", mdata); // send data to all clients
 } else {
   console.log(readData);
```



DIY 결과 [2] : 가상적인 세 개의 센서신호 시뮬레이션 → 조도, 습도, 온도를 Node.js로 처리

```
parser.on("data", (data) => {
 // call back when data is received
 readData = data.toString();
 firstcommaidx = readData.indexOf(",");
 secondcommaidx = readData.indexOf(",", firstcommaidx + 1);
  if (firstcommaidx > 0)
   lux = readData.substring(0, firstcommaidx);
   humi = readData.substring(firstcommaidx + 1, secondcommaidx);
   temp = readData.substring(secondcommaidx + 1);
   readData = "";
   dStr = getDateString();
   mdata[0] = dStr; //date
   mdata[1] = lux; //data
   mdata[2] = humi;
   mdata[3] = temp;
   console.log("AA00," + mdata.toString());
   io.sockets.emit("message", mdata); // send data to all clients
  } else {
   console.log(readData);
```



DIY 결과 [2] : 가상적인 세 개의 센서신호 시뮬레이션 → 조도, 습도, 온도를 Node.js로 처리

```
D:\aann\aann-rpt06\iot\multi_signals>node aann_multi_signals
serial port open
AA00,2021-10-05 14:21:10.805,223,47,-1
AA00,2021-10-05 14:21:11.804,222,48,0
AA00,2021-10-05 14:21:12.808,173,84,28
AA00,2021-10-05 14:21:13.811,215,49,-10
AA00,2021-10-05 14:21:14.811,237,82,-8
                                          ID,시간,조도,습도,온도
AA00,2021-10-05 14:21:15.815,179,43,-3
AA00,2021-10-05 14:21:16.814,153,80,2
AA00,2021-10-05 14:21:17.818,207,59,19
AA00,2021-10-05 14:21:18.817,249,50,3
AA00,2021-10-05 14:21:19.821,185,68,6
AA00,2021-10-05 14:21:20.820,162,87,16
```





A4.5.6 multi-signals + Node project: WEB

[Web monitoring] client_multi_signals.html



IoT Signal from Arduino

Real-time Signals

on Time: 2021-10-05 14:27:23.536

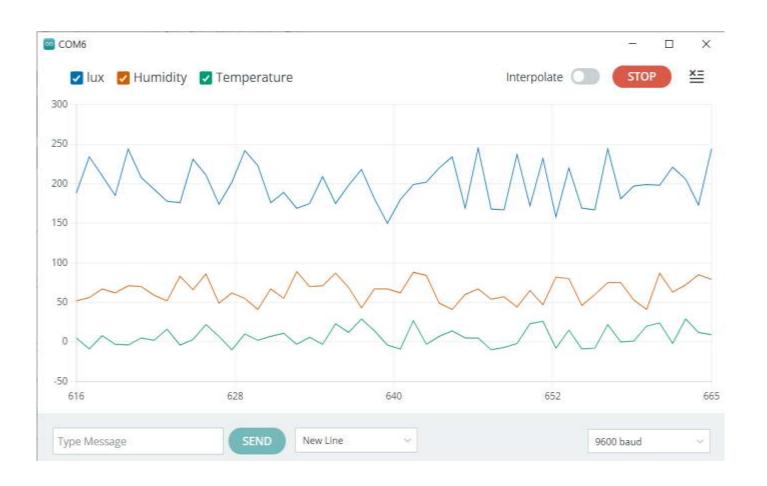
Signals (조도,습도,온도) : 161,41,22





multi-signals + Node project : WEB streaming

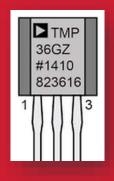
[Web monitoring] client_multi_signals_streaming.html





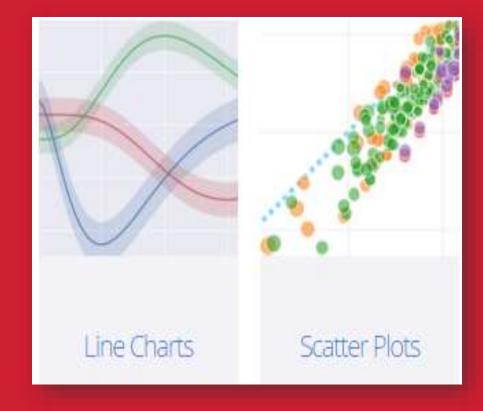








Data visualization using plot.ly

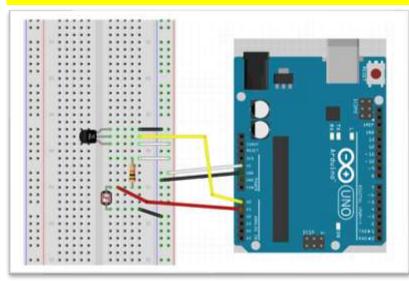






Network socket emitting data

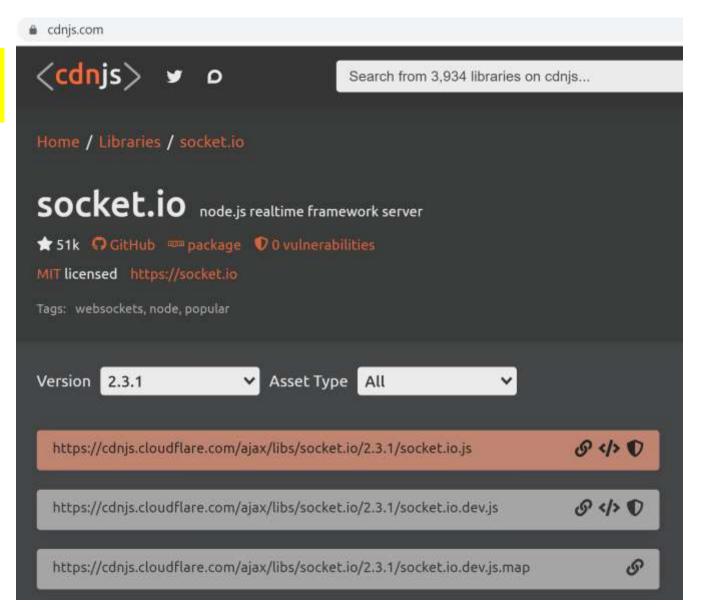
tmp36 + CdS circuit



```
AA00 2020-10-17 11:41:30.533 25.27,245
AA00 2020-10-17 11:41:31.535 25.27,243
AA00 2020-10-17 11:41:32.535 25.27,158
AA00 2020-10-17 11:41:33.534 24.29,40
AA00 2020-10-17 11:41:34.538 24.29,33
AA00 2020-10-17 11:41:35.537 24.78,86
AA00 2020-10-17 11:41:36.541 25.27,249
AA00 2020-10-17 11:41:37.540 25.76,245
AA00 2020-10-17 11:41:38.543 25.76,243
AA00 2020-10-17 11:41:39.543 25.27,245
```

```
var readData = "";
var temp = "";
var lux = "";
var mdata = [];
var firstcommaidx = 0;
parser.on("data", (data) => {
 // call back when data is received
 readData = data.toString();
 firstcommaidx = readData.indexOf(",");
 if (firstcommaidx > 0) {
    temp = readData.substring(0, firstcommaidx);
    lux = readData.substring(firstcommaidx + 1);
    readData = "";
    dStr = getDateString();
   mdata[0] = dStr; //date
    mdata[1] = temp; //data
                                  시간,온도,조도
    mdata 2 = lux;
    console.log("AA00," + mdata);
    io.sockets.emit("message", mdata); // send data
   else
    console.log(readData);
```

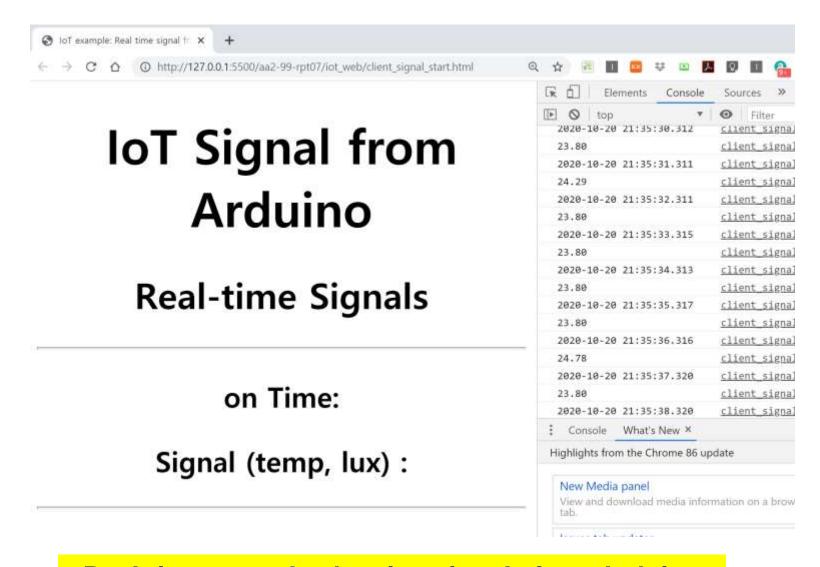
Google search socket.io.js cdn



```
<!DOCTYPE html>
                                                                              client_signal_start.html
     <head>
       <meta charset="utf-8">
       <title>IoT example: Real time signal from Arduino</title>
 5
     <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/2.3.1/socket.io.js"></script>
 6
       <!-- <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/1.3.6/socket.io.js"></scr
       <style>body padding:0;margin:30;background: □ #fff </style>
 9
     </head>
10
     <body> <!-- style="width:100%;height:100%"> -->
11
12
     <h1 align="center"> IoT Signal from Arduino </h1>
13
14
15
     <h2 align="center"> Real-time Signals </h2>
16
17
     (hr)
18
     <h3 align="center"> on Time: <span id="time"> </span> </h3>
19
20
     <h3 align="center"> Signal (temp, lux) : <span id="data"> </span> </h3>
21
22
```

Google search: socket.io.js cdn

```
<script>
                                                   client_signal_start.html
25
       /* JAVASCRIPT CODE GOES HERE */
26
         var ctime = document.getElementById('time');
27
28
         var cdata = document.getElementById('data');
29
30
         // Get message from network socket
31
         var socket = io.connect('http://localhost:3000'); // port = 3000
32
         socket.on('connect', function () {
             socket.on('message', function (msg) {
33
                 // message on Socket (TCP server)
34
35
                 // Check the console by F12 in Chrome browser
36
                 console.log(msg[0]); // mdata[0] => dStr
                 console.log(msg[1]); // mdata[1] => msg[1]
37
38
                 console.log(msg[2]);
39
                 //console.log(msg[3]);
40
41
                 // Display signal on web browser
                 ctime.innerHTML = msg[0];
42
                 cdata.innerHTML = msg[1] + "," + msg[2]; // + "," + msg[3];
43
44
             });
45
         });
46
47
       </script>
```



Real-time console showing signals from Arduino in Chrome browser

[Web monitoring] client_signal_cds_tmp36.html



IoT Signal from Arduino

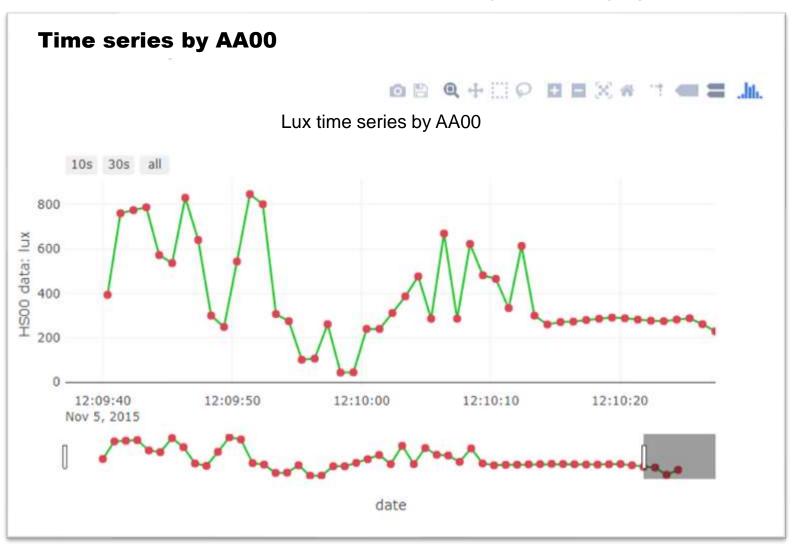
Real-time Signals

on Time: 2021-10-05 14:02:26.657

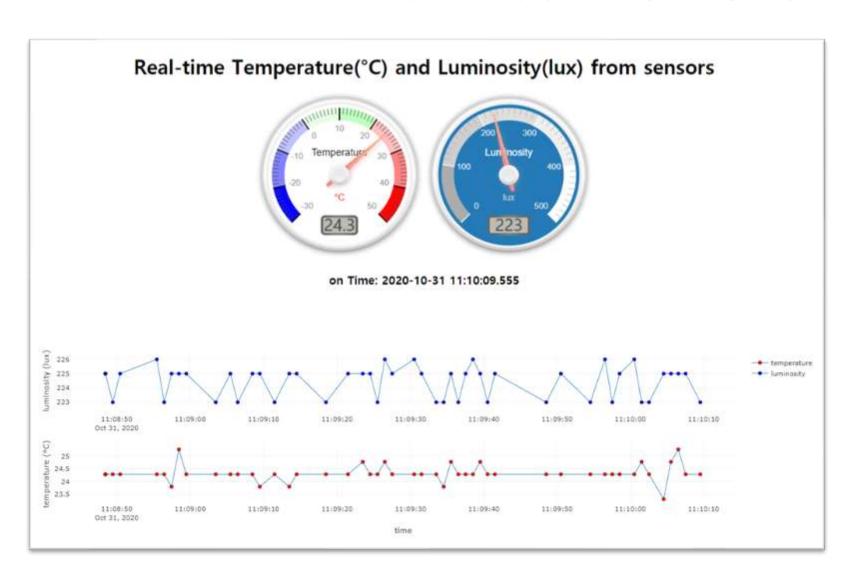
Signal (temp,lumi): 25.27,84

Real-time monitoring of signals from Arduino tmp36 + CdS circuit

Arduino data + plotly.js



Arduino data + plotly.js + gauge.js





A5. Introduction to visualization

System (Arduino, sDevice, ...)



Data (signal, image, sns, ...)



Visualization & monitoring



Data storaging & mining

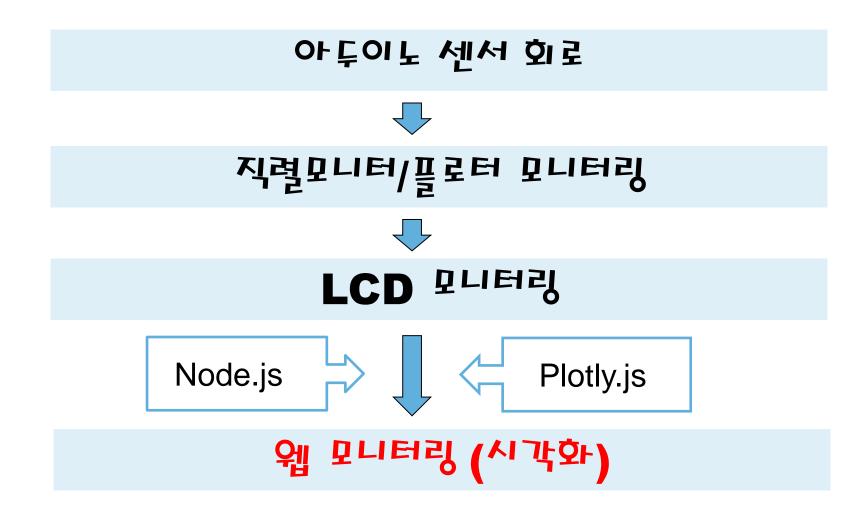


Service





A5.1 Introduction to data visualization



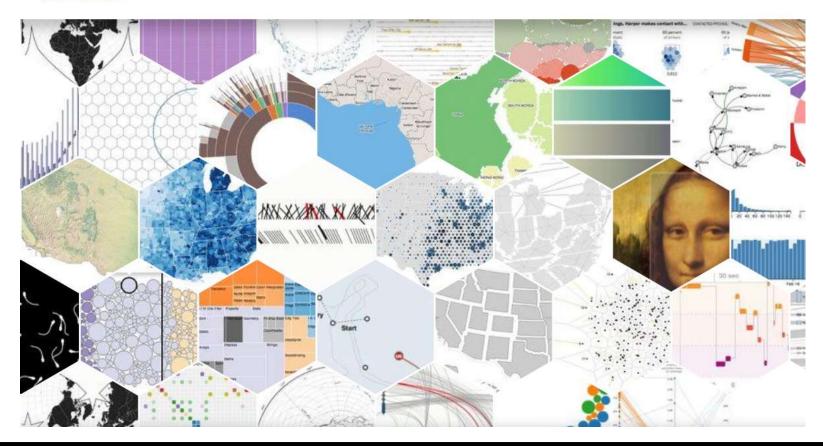




A5.1.1 D3.js

Overview Examples Documentation API Source

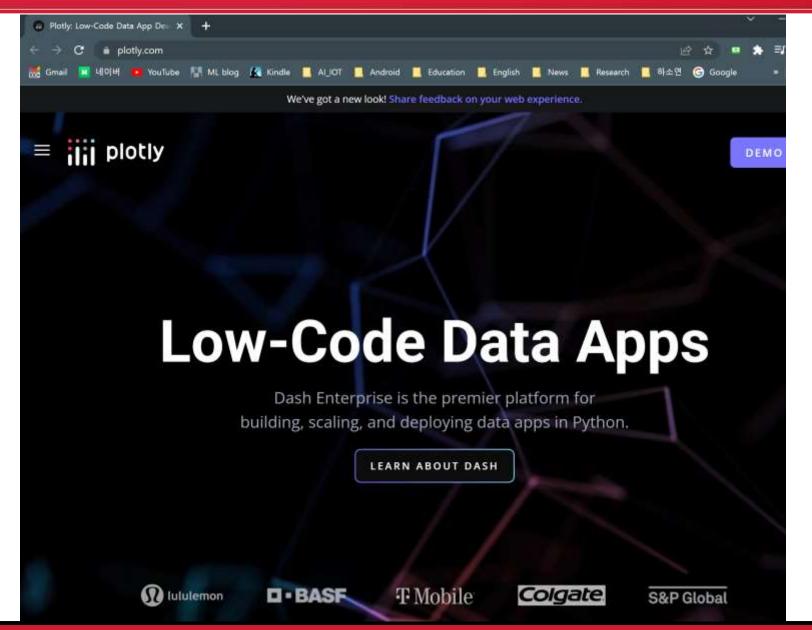








A5.1.2 plotly







A5.1.3 plotly.js



Built on top of <u>d3.js</u> and <u>stack.gl</u>,

Plotly.js is a high-level, declarative

charting library.

plotly.js ships with over 40 chart types,

including 3D charts, statistical graphs, and

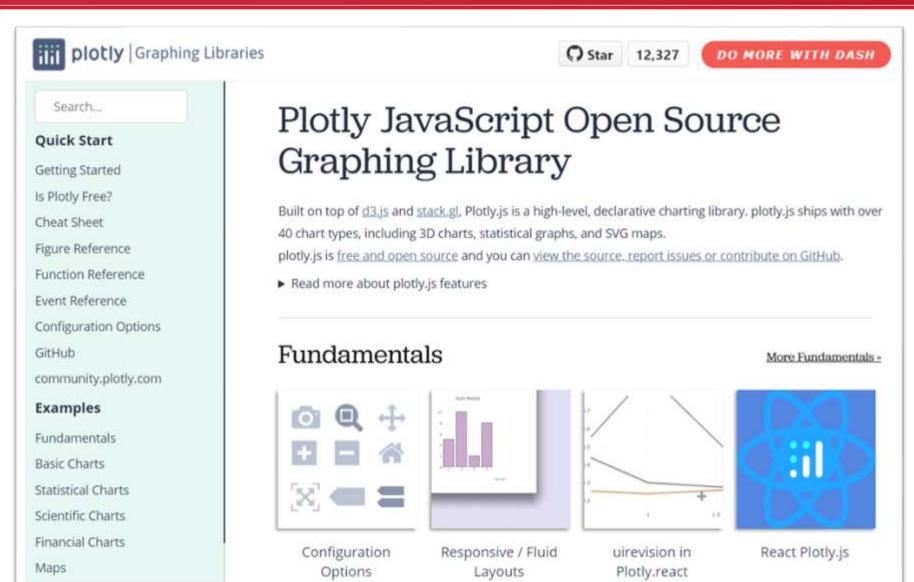
SVG maps.

https://plot.ly/javascript/





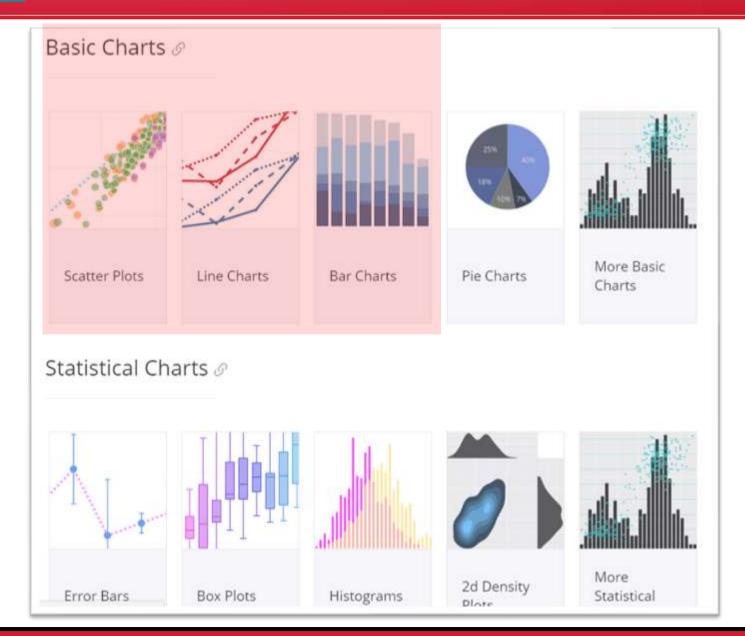
A5.1.4 Introduction to plotly.js







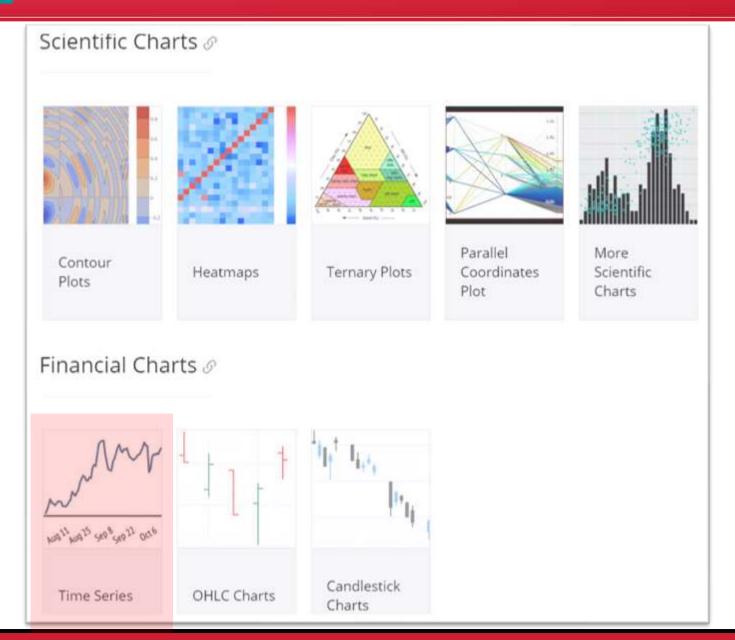
A5.1.5 Introduction to plotly.js charts







A5.1.6 Introduction to plotly.js charts







A5.1.7 Introduction to plotly.js charts

Maps Ø



Choropleth Maps



Scatter Plots on Maps



Bubble Maps



Lines on Maps



Scatter Plots on Mapbox

3D Charts @



3D Scatter



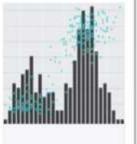
Ribbon Plots



3D Surface Plots



3D Mesh Plots



More 3D Charts





A5.1.8 plotly.js: time series & streaming





https://plot.ly/javascript/time-series/

https://plot.ly/javascript/streaming/





A5.1.9 Getting started: plotly.js



https://plot.ly/javascript/getting-started/



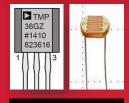
A5.1.10 Getting started: plotly.js

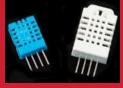


<script src="https://cdn.plot.ly/plotly-latest.min.js"></script>

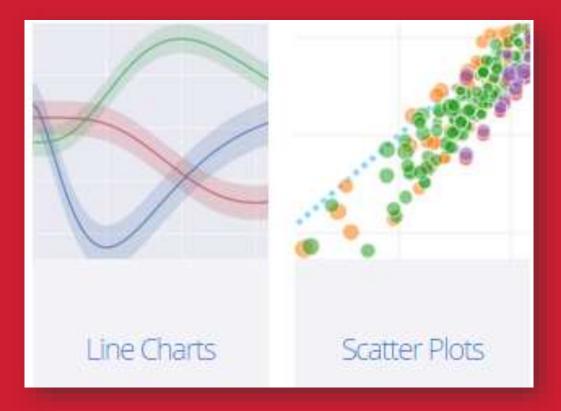








Data charts using plotly.js







A5.2 Data charts

Navigation Basic Line Plot Line and Scatter Plot Adding Names to Line and Scatter Plot Line and Scatter Styling Styling Line Plot Colored and Styled Scatter Plot Line Shape Options for Interpolation Graph and Axes Titles Line Dash Connect Gaps Between Data Labelling Lines with Annotations Back To Plotly.Js



Line Charts in plotly.js

How to make D3.js-based line charts in JavaScript.







Basic Line Plot &

```
var trace1 = {
 x: [1, 2, 3, 4],
 y: [10, 15, 13, 17],
 type: 'scatter'
var trace2 = {
 x: [1, 2, 3, 4],
  y: [16, 5, 11, 9].
  type: 'scatter'
```





A5.2.1 Working folders



- > 📙 data
- > adata_charts
 - client_signal_start.html
 - DV_chart00_start.html
 - DV_chart01_hello_data.html
 - DV_ts01_hello_time_series.html
 - DV_ts02_aapl_chart.html
 - DV_ts03_sensor_chart.html
 - DV_ts04_remote_chart_rangeslider.html
 - Hint_code.txt





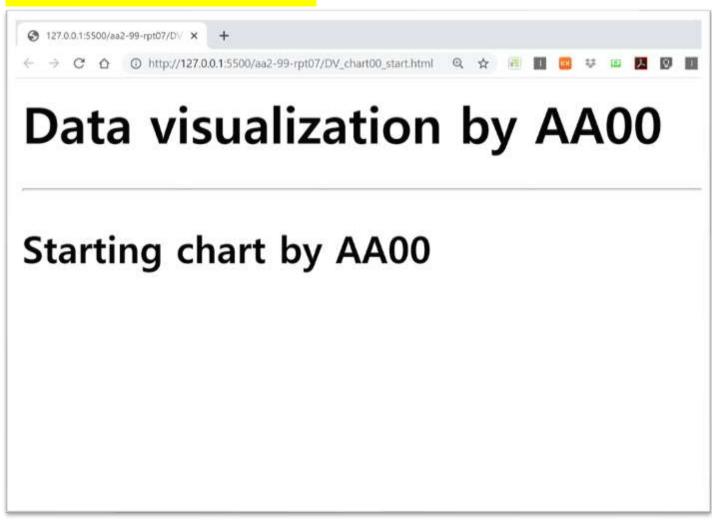
A5.2.2.1 Starting plotly basic chart

```
DV_chart00_start.html
                                               Starting chart!
   <html>
   <head>
       <meta charset="utf-8">
    <!-- Plotly.js -->
     <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
   </head>
   <body>
       <h1>Data visualization by AA00</h1>
 9
       (hr)
10
       <h2>Starting graph by AA00</h2>
11
12
       <!-- Plotly chart will be drawn inside this DIV -->
       <div id="myDiv" style="width: 500px; height: 300px"></div>
13
14
15
       <script>
           <!-- JAVASCRIPT CODE GOES HERE -->
16
17
18
19
      </script>
   </body>
20
   </html>
21
22
```



A5.2.2.2 Starting plotly basic chart

VSCode, live server





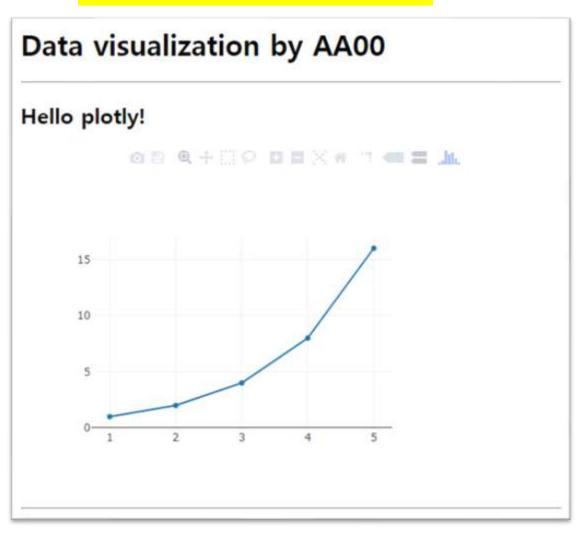
A5.2.3.1 Hello plotly basic chart

```
<html>
                                               Hello plotly data chart!
 2 <head>
                                            DV_chart01_hello_data.html
       <meta charset="utf-8">
       <!-- Plotly.js -->
       <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
  </head>
   <body>
       <h1>Data visualization by AA00</h1>
8
9
       (hr)
       <h2>Hello plotly!</h2>
10
       <!-- Plotly chart will be drawn inside this DIV -->
11
12
     <div id="myDiv" style="width: 500px;height: 400px"></div>
13
       <hr>
14
       <script>
15
           <!-- JAVASCRIPT CODE GOES HERE -->
16
           var data = [
17
18
               x: [1, 2, 3, 4, 5],
                                                      data는 무엇?
               y: [1, 2, 4, 8, 16],
19
                                                  그래프 객체들의 구조,
               type: 'scatter'
20
21
           }];
                                                       데이터 배열
22
23
           Plotly newPlot('myDiv', data);
24
25
       </script>
   </body>
   </html>
```



A5.2.3.2 Hello plotly basic chart

Graph: Hello plotly chart!



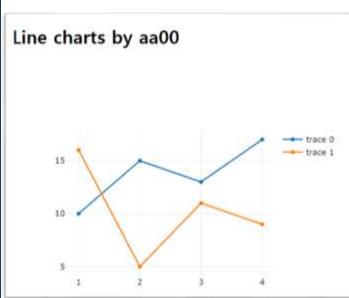




A5.2.4 plotly.js: Line Charts

[1] Basic multi-line charts → DV_chart02_two_traces.html

```
<script>
    <!-- JAVASCRIPT CODE GOES HERE -->
    var trace1 = {
        x: [1, 2, 3, 4],
        y: [10, 15, 13, 17],
        type: 'scatter'
    };
    var trace2 = {
        x: [1, 2, 3, 4],
        y: [16, 5, 11, 9],
        type: 'scatter'
    };
    var data = [trace1, trace2];
    Plotly.newPlot('myDiv', data);
</script>
```







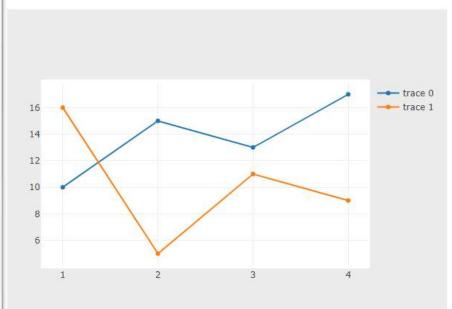
A5.2.5 plotly.js: Line Charts

[2] Basic line charts with layout -> DV_chart03_layout.html

```
var layout = {
    autosize: false,
    width: 600,
    height: 450,
    margin: {
        1: 50, // left
        r: 50, // right
        b: 100, // bottom
        t: 100, // top
        pad: 4 // padding
    },
    paper bgcolor: '#ececec',
    plot bgcolor: '#ffffff' //'#rrggbb'
};
Plotly.newPlot('myDiv', data, layout);
```

Test: pad \rightarrow 40

Line charts with layout by AA00



Save AAnn_Chart_Layout.png





A5.2.6.1 plotly.js: Line & Scatter plot

[3] Line & scatter plot : setting mode -> DV_chart04_modes.html

```
var trace1 = {
   x: [1, 2, 3, 4],
    y: [10, 15, 13, 17],
   mode: 'markers'
};
var trace2 = {
   x: [2, 3, 4, 5],
    y: [16, 5, 11, 9],
   mode: 'lines'
};
var trace3 = {
    x: [1, 2, 3, 4],
    y: [12, 9, 15, 12],
   mode: 'lines+markers'
};
```

```
var data = [ trace1, trace2, trace3 ];
var layout = {
    title: 'Line and Scatter charts by AA00',
    width: 600,
    height: 450,
    margin: {
        1: 50,
        r: 50,
       b: 100,
       t: 100,
        pad: 4
    },
};
Plotly.newPlot('myDiv', data, layout);
```

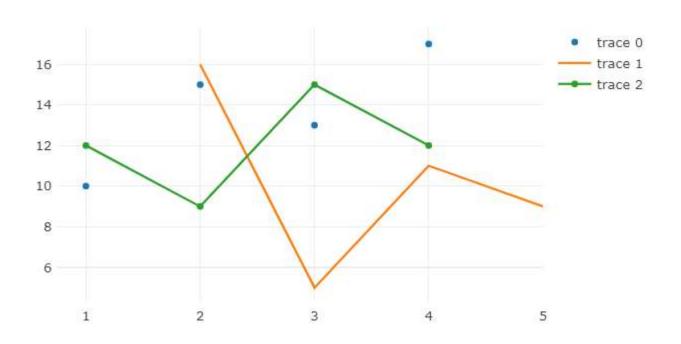




A5.2.6.2 plotly.js: Line & Scatter plot

[3.1] Line & scatter plot with mode & title

Line and Scatter charts by AA00





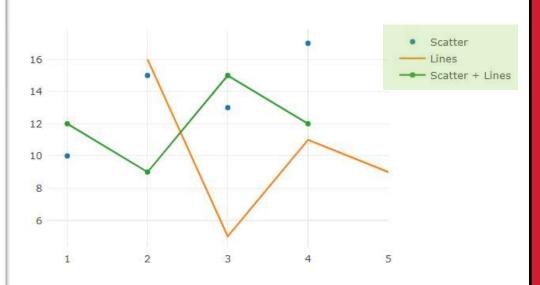


A5.2.6.3 plotly.js: Line & Scatter plot

[3.2] Line & scatter plot with axis name > DV_chart05_axis_name.html

```
var trace1 = {
    x: [1, 2, 3, 4],
    y: [10, 15, 13, 17],
    mode: 'markers',
    name: 'Scatter'
};
var trace2 = {
    x: [2, 3, 4, 5],
    y: [16, 5, 11, 9],
    mode: 'lines',
    name: 'Lines'
};
var trace3 = {
    x: [1, 2, 3, 4],
    y: [12, 9, 15, 12],
    mode: 'lines+markers',
    name: 'Scatter + Lines'
};
```

Line and Scatter charts by AA00





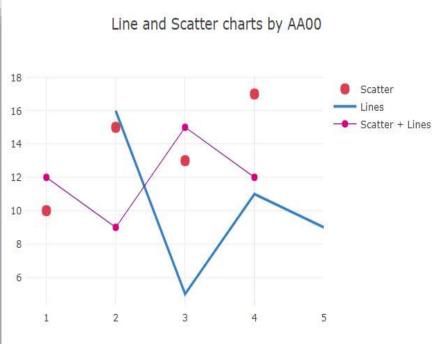


A5.2.6.4 plotly.js: Line & Scatter plot

[3.3] Line & scatter plot with style > DV_chart06_style.html

```
var trace1 = {
 x: [1, 2, 3, 4],
 y: [10, 15, 13, 17],
 mode: 'markers',
 name: 'Scatter',
 marker: {
   color: 'rgb(219, 64, 82)',
   size: 12
var trace2 = {
 x: [2, 3, 4, 5],
 y: [16, 5, 11, 9],
 mode: 'lines',
 name: 'Lines',
 line: {
   color: 'rgb(55, 128, 191)',
   width: 3
```

```
var trace3 = {
 x: [1, 2, 3, 4],
 y: [12, 9, 15, 12],
 mode: 'lines+markers',
 name: 'Scatter + Lines',
 marker: {
   color: 'rgb(128, 0, 128)',
   size: 8
 line: {
   color: 'rgb(128, 0, 128)',
  width: 1
```







A5.2.6.5 plotly.js: Line & Scatter plot

[3.4] Line & scatter plot with axis titles -> DV_chart07_axis_title.html

```
var layout = {
 title:'Line and Scatter Plot',
 width: 600, height: 450,
 margin: {
   l: 50,
   r: 50,
   b: 100,
   t: 100,
   pad: 4
 xaxis: {
   title: 'x-axis title'
 yaxis: {
   title: 'y-axis title'
```





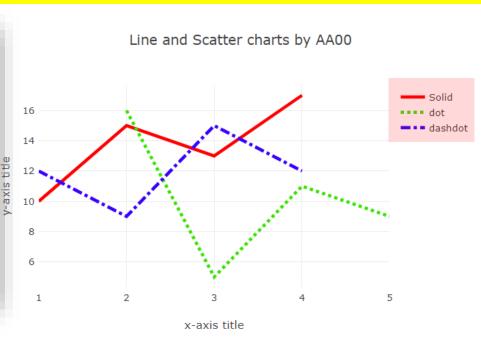


A5.2.6.6 plotly.js: Line & Scatter plot

[3.5] Line & scatter plot with dash and dot - DV_chart08_line_style.html

```
var trace1 = {
 x: [1, 2, 3, 4],
 y: [10, 15, 13, 17],
 mode: 'lines',
 name: 'Solid',
 line: {
   color: 'rgb(255, 0, 0)',
   dash: 'solid',
   width: 4
var trace2 = {
 x: [2, 3, 4, 5],
 y: [16, 5, 11, 9],
 mode: 'lines',
 name: 'dot',
 line: {
   color: 'rgb(55, 228, 0)'
   dash: 'dot',
   width: 4
```

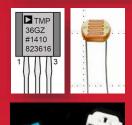
```
var trace3 = {
 x: [1, 2, 3, 4],
 y: [12, 9, 15, 12],
 mode: 'lines',
 name: 'dashdot',
 line: {
   color: 'rgb(55, 0, 255',
   dash: 'dashdot',
   width: 4
};
```



Save AAnn_Line_Dash_Dot.png







Data visualization using plotly.js









A5.3. Time series







A5.3.1 plotly.js: Time series

[1] Time series : date strings

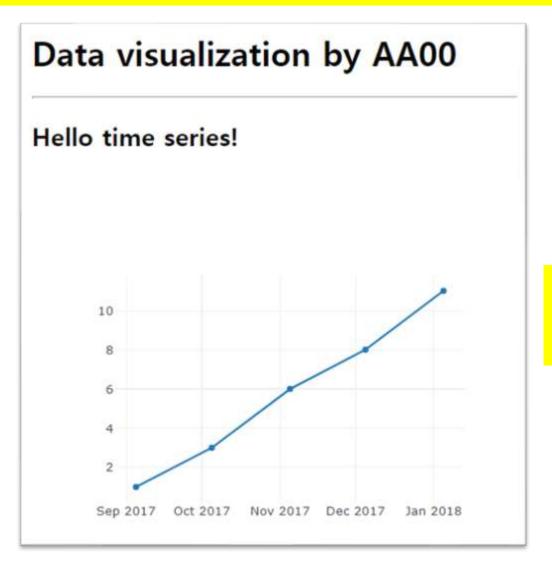
```
<!-- Plotly chart will be drawn inside this DIV -->
<div id="myDiv" style="width: 500px;height: 400px"></div>
<script>
    <!-- JAVASCRIPT CODE GOES HERE -->
    var data = [
        x: ['2017-9-04 22:23:00',
        '2017-10-04 22:23:00',
        '2017-11-04 22:23:00',
        '2017-12-04 22:23:00',
        '2018-01-04 22:23:00'],
        y: [1, 3, 6, 8, 11],
        type: 'scatter'
    ];
    Plotly.newPlot('myDiv', data);
```





A5.3.2 plotly.js: Time series

Time series : date strings → DV_ts01_hello_time_series.html



[DIY]

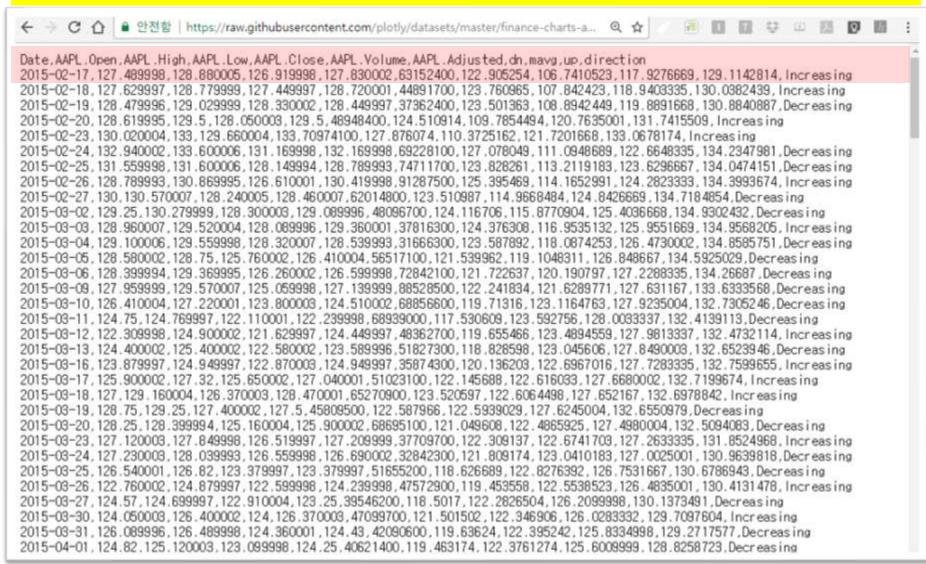
오늘 날자와 데이터를 추가





A5.3.3.1 plotly.js: Time series

[2] Time series: financial data strings – AAPL stock price







A5.3.3.2 plotly.js: Time series

[2] Time series: financial data strings – AAPL stock price

```
Plotly.d3.csv("https://raw.githubusercontent.com/plotly/datasets/master/
    finance-charts-apple.csv", function(err, rows){
   function unpack(rows, key) {
        return rows.map(function(row) { return row[key]; });
   var trace1 = {
       type: "scatter",
        mode: "lines",
        name: 'AAPL High',
        x: unpack(rows, 'Date'),
       y: unpack(rows, 'AAPL.High'),
       line: {color: '#17BECF'}
   var trace2 = {
       type: "scatter",
        mode: "lines",
        name: 'AAPL Low',
        x: unpack(rows, 'Date'),
        y: unpack(rows, 'AAPL.Low'),
        line: {color: '#7F7F7F'}
   var data = [trace1,trace2];
```

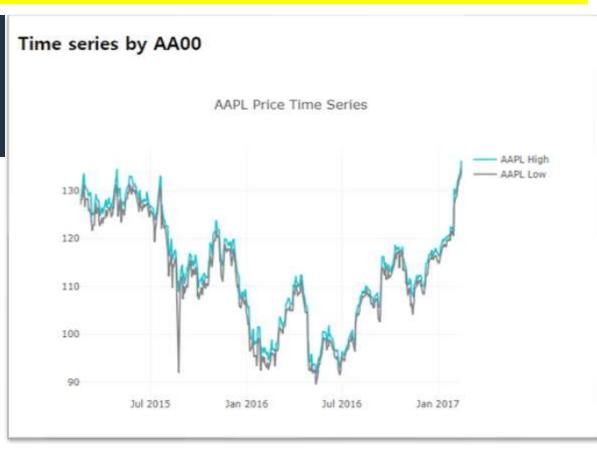




A5.3.3.3 plotly.js: Time series

[2] Time series: financial data strings – AAPL stock price

```
var data = [trace1,trace2];
var layout = {
    title: 'AAPL Price Time Series',
};
Plotly.newPlot('myDiv', data, layout);
```



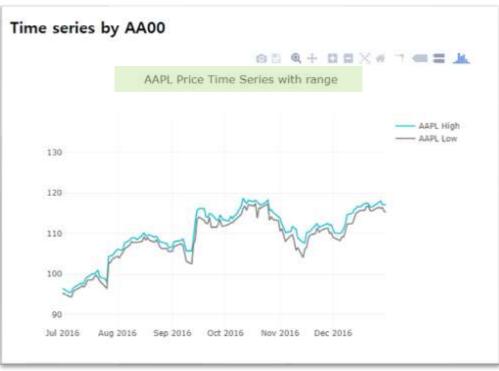




A5.3.3.4 plotly.js: Time series

[2] Time series: financial data strings – set range

```
var data = [trace1,trace2];
var layout = {
   title: 'AAPL Price Time Series with range',
   xaxis: {
        range: ['2016-07-01', '2016-12-31'],
       type: 'date'
   yaxis: {
        autorange: true,
        range: [86.8700008333, 138.870004167],
        type: 'linear'
Plotly newPlot('myDiv', data, layout);
```



날짜와 주가의 범위를 지정

→ 2016년 하반기





A5.3.3.5 plotly.js: Time series

[2] Time series: financial data strings – Range slider

DV_ts03_aapl_chart_range_slider.html

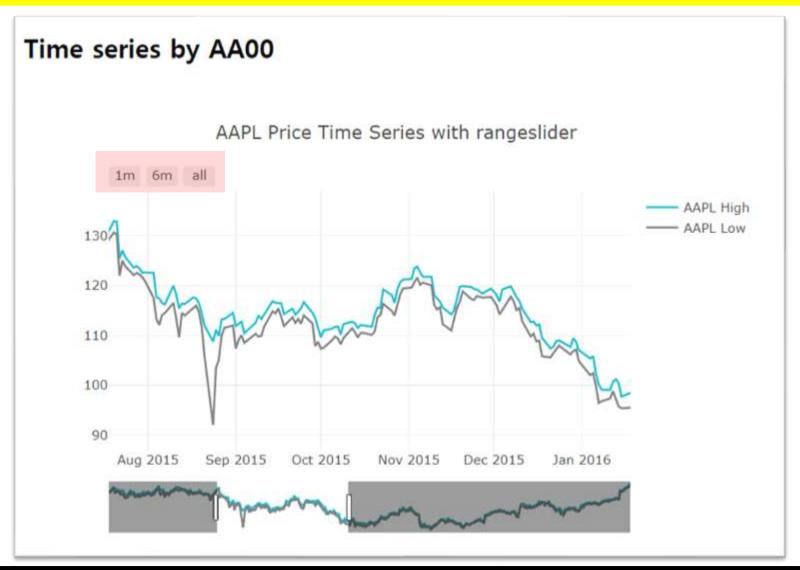
```
var layout = {
    title: 'AAPL Price Time Series with rangeslider',
    xaxis: {
        autorange: true,
        range: ['2015-02-17', '2017-02-16'],
        rangeselector: {buttons: [
                count: 1,
                label: '1m',
                step: 'month',
                stepmode: 'backward'
            },
{
                count: 6,
                label: '6m',
                step: 'month',
                stepmode: 'backward'
            {step: 'all'}
            1}.
            rangeslider: {range: ['2015-02-17', '2017-02-16']},
            type: 'date'
        },
        yaxis: {
            autorange: true,
            range: [86.8700008333, 138.870004167],
            type: 'linear'
```





A5.3.3.6 plotly.js: Time series

[2] Time series: financial data strings – Range slider





A5.3.4.1 plotly.js: Sensor time series

831,

641,

249,

544.

847,

802, 307,

101,

106,

261,

43,

44,

240,

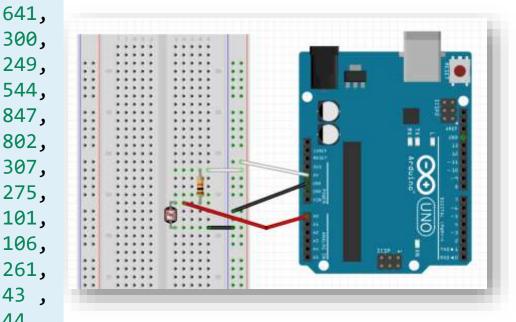
240,

312,

[3] Time series: my lux data

```
2015-11-05 12:09:41.382',
'2015-11-05 12:09:42.380',
'2015-11-05 12:09:43.378',
'2015-11-05 12:09:44.377',
'2015-11-05 12:09:45.375',
'2015-11-05 12:09:46.389',
'2015-11-05 12:09:47.388',
'2015-11-05 12:09:48.386',
'2015-11-05 12:09:49.384',
'2015-11-05 12:09:50.383',
'2015-11-05 12:09:51.381',
'2015-11-05 12:09:52.380',
'2015-11-05 12:09:53.394',
'2015-11-05 12:09:54.392',
'2015-11-05 12:09:55.391',
'2015-11-05 12:09:56.389',
'2015-11-05 12:09:57.387',
'2015-11-05 12:09:58.386',
'2015-11-05 12:09:59.384',
'2015-11-05 12:10:00.398',
'2015-11-05 12:10:01.397',
```

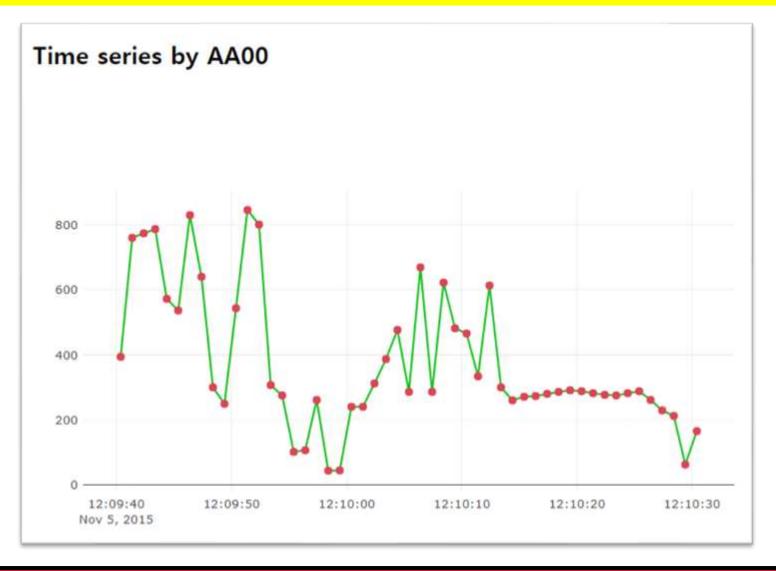
394, Data: 761, 775, 788, date, value 573, 537,





A5.3.4.2 plotly.js: Time series

[3] Time series : my lux data -> DV_ts03_sensor_chart.html

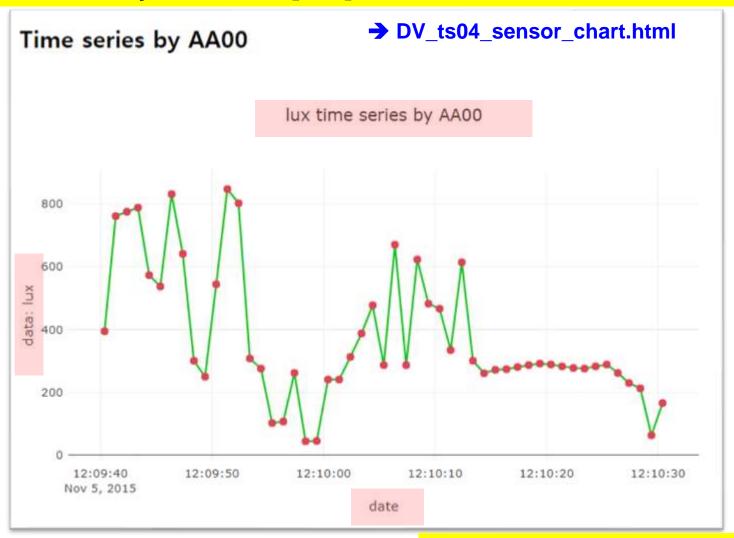






A5.3.4.3 plotly.js: Time series

[3] Time series: my lux data – [DIY] → Set title and axis title



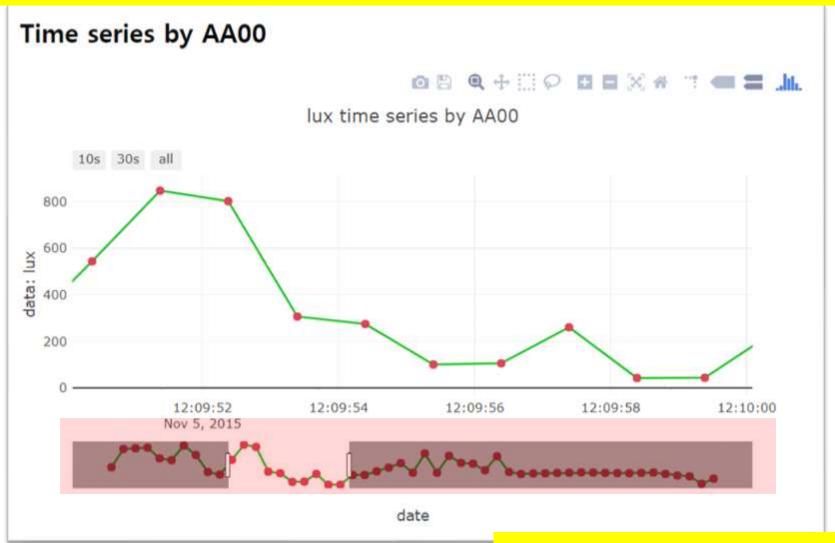
AAnn_lux_Time_Series.png





Project: Time series with Rangeslider

[Project-DIY] → DV_ts05_sensor_Rangeslider.html



AAnn_lux_Rangelslider.png





[Practice]

- ♦ [wk08]
- > charts by plotly
- Complete your project
- Upload folder: aann-rpt08
- Use repo "aann" in github

wk08: Practice: aann-rpt08



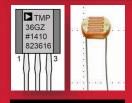
- [Target of this week]
 - Complete your works
 - Save your outcomes and upload outputs in github

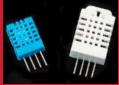
제출폴더명: aann-rpt08

- 압축할 파일들
 - ① AAnn_Chart_Layout.png
 - ② AAnn_Axis_Title.png
 - 3 AAnn_Line_Dash_Dot.png
 - 4 AAnn_lux_Time_Series.png
 - **5** AAnn_lux_Rangeslider.png
 - 6 All *.html in data_charts folder









Data visualization using plotly.js





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.com
- √ http://www.github.com GitHub

Target of this class





Real-time Weather Station from nano 33 BLE sensors



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

