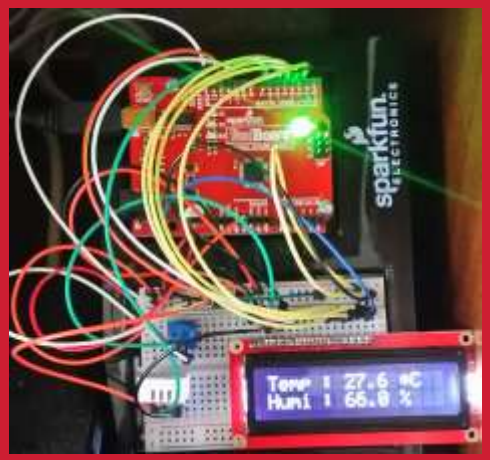
Arduino-IOT

[wk08]

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

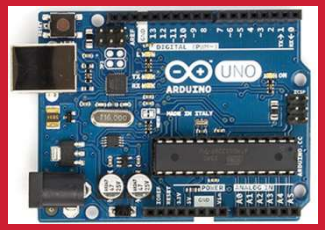
My ID

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]

- [wk06]
  - Arduino sensors + Node.js
  - Complete your project
  - Upload folder: aax-nn-rpt06
  - Use repo “aax-nn” in github

wk06 : Practice : AAnn_Rpt06

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

제출폴더명 : **aax-nn-rpt06**

- 압축할 파일들

① **AAnn_cds_IOT_data.png**

② **AAnn_cds_tmp36_serial.png**

③ **AAnn_cds_tmp36_lcd.png**

④ **AAnn_cds_tmp36_IOT.png**

⑤ **AAnn_multi_signals_node.png**

⑥ **All *.ino**

⑦ **All *.js**

⑧ **NO node_modules folder**

IOT: HSC

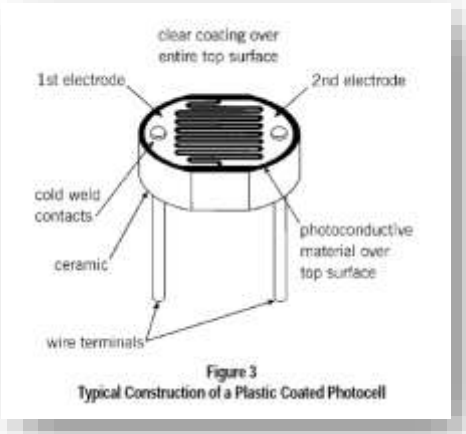
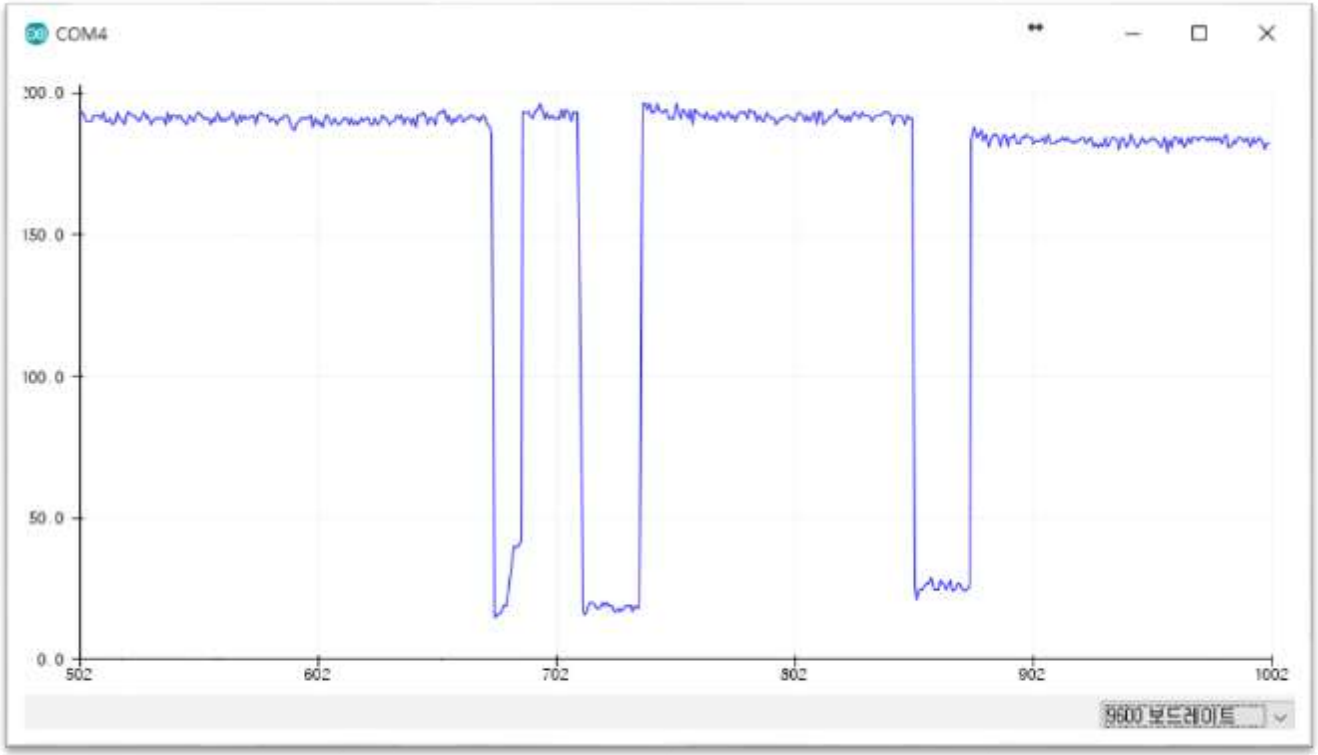
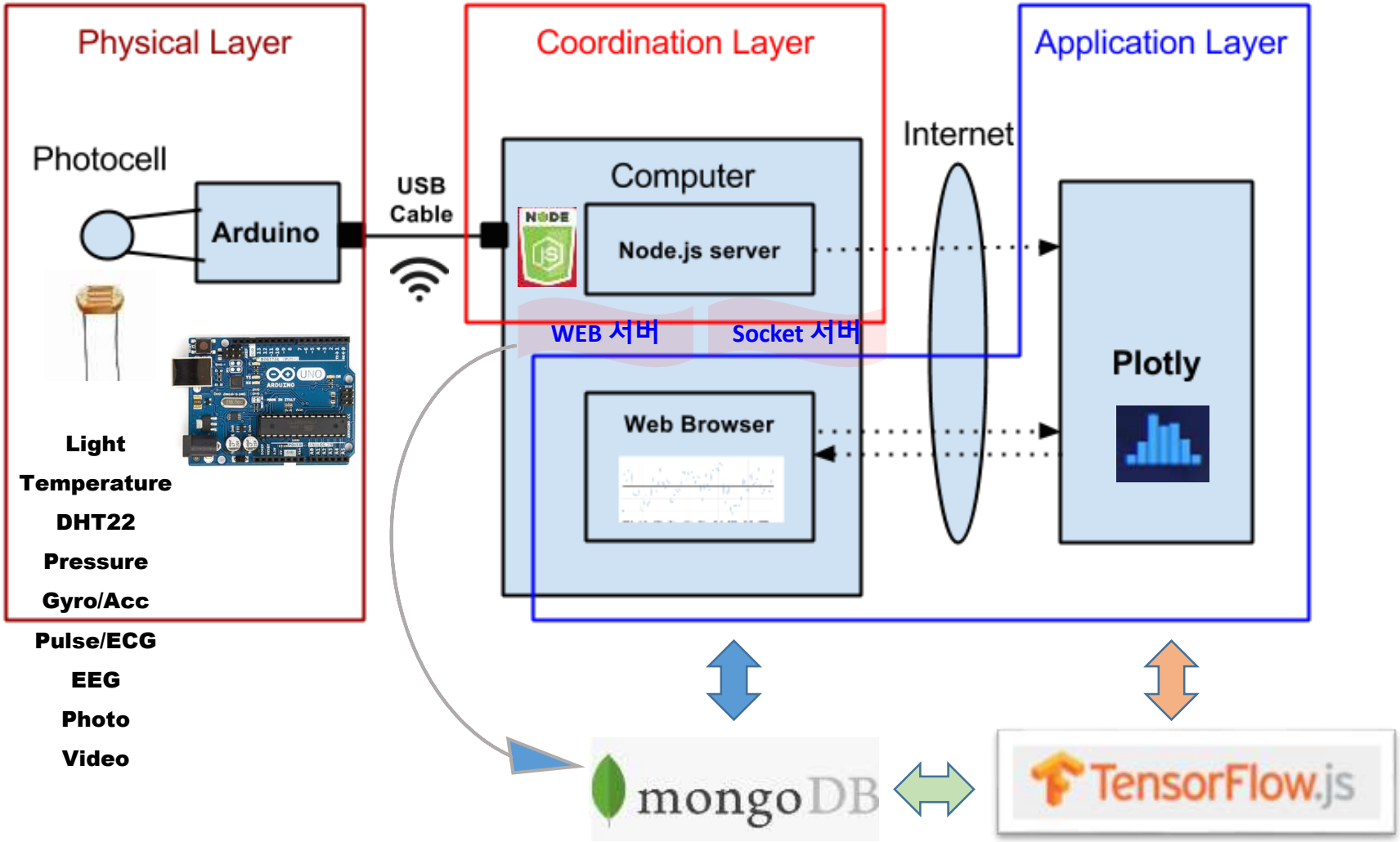


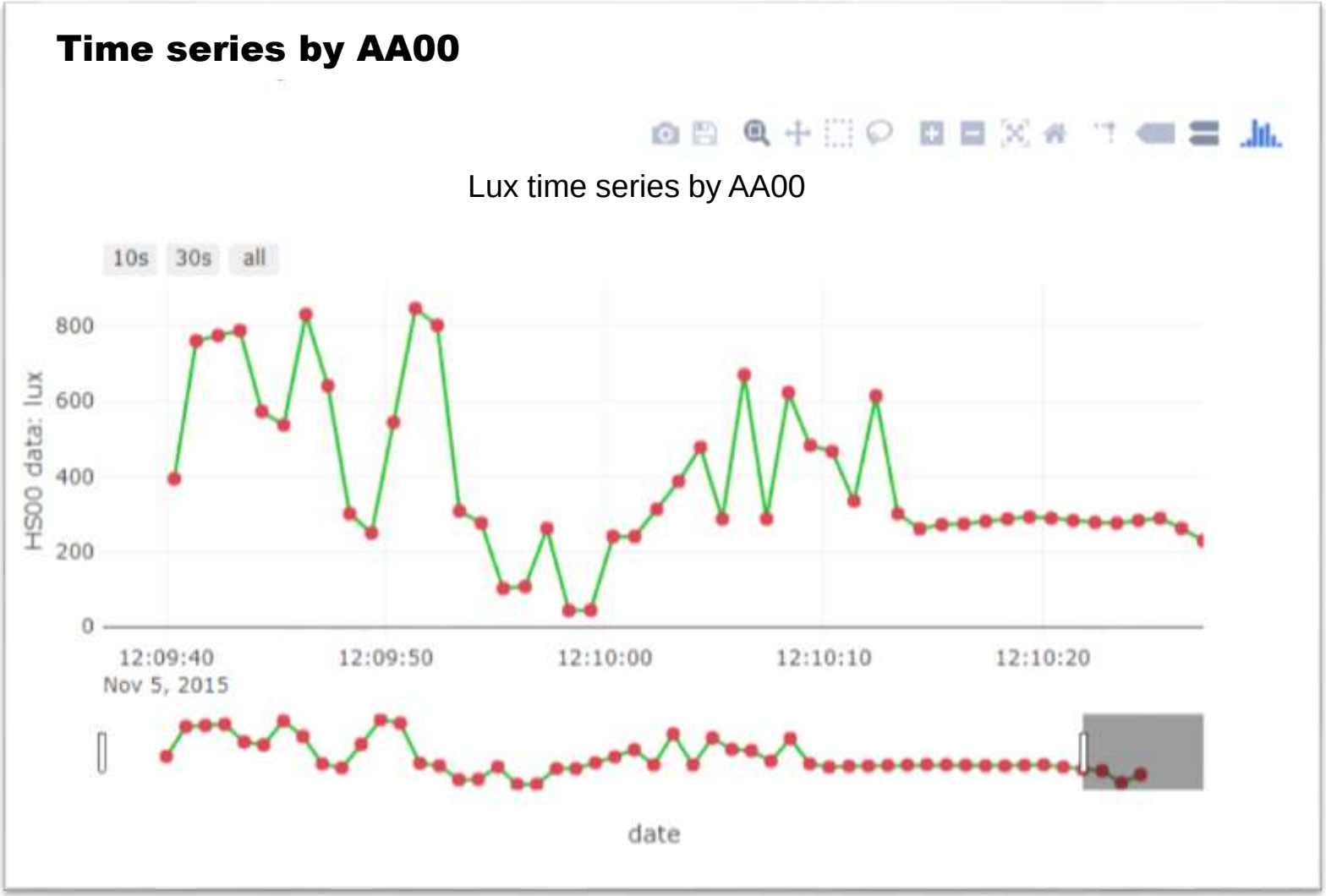
Figure 3
Typical Construction of a Plastic Coated Photocell

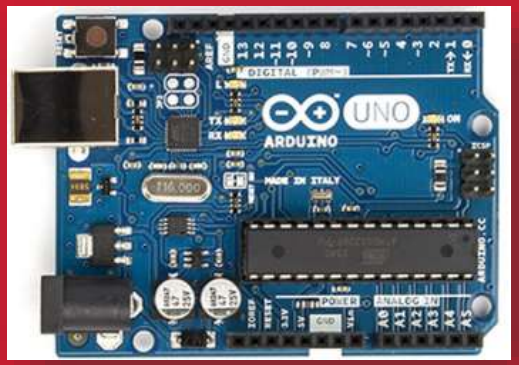


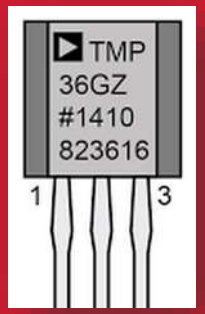
Layout [H S C]


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



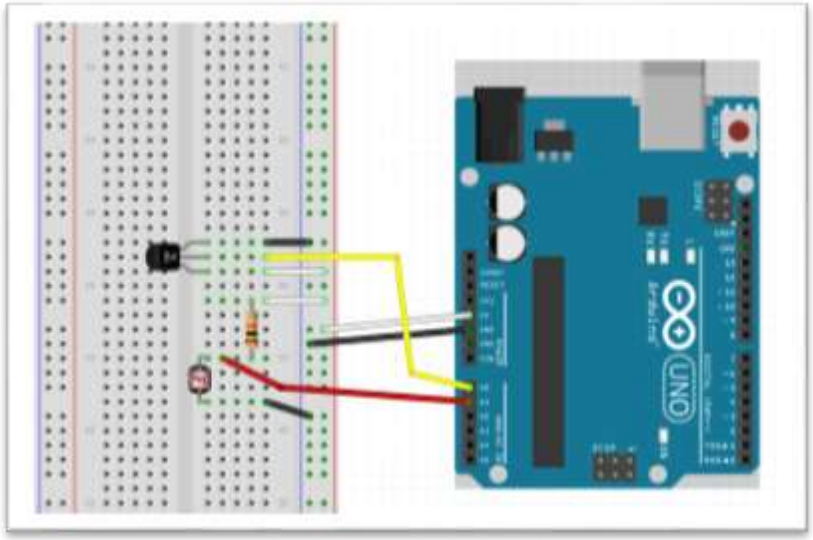




Data visualization using plot.ly

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
```

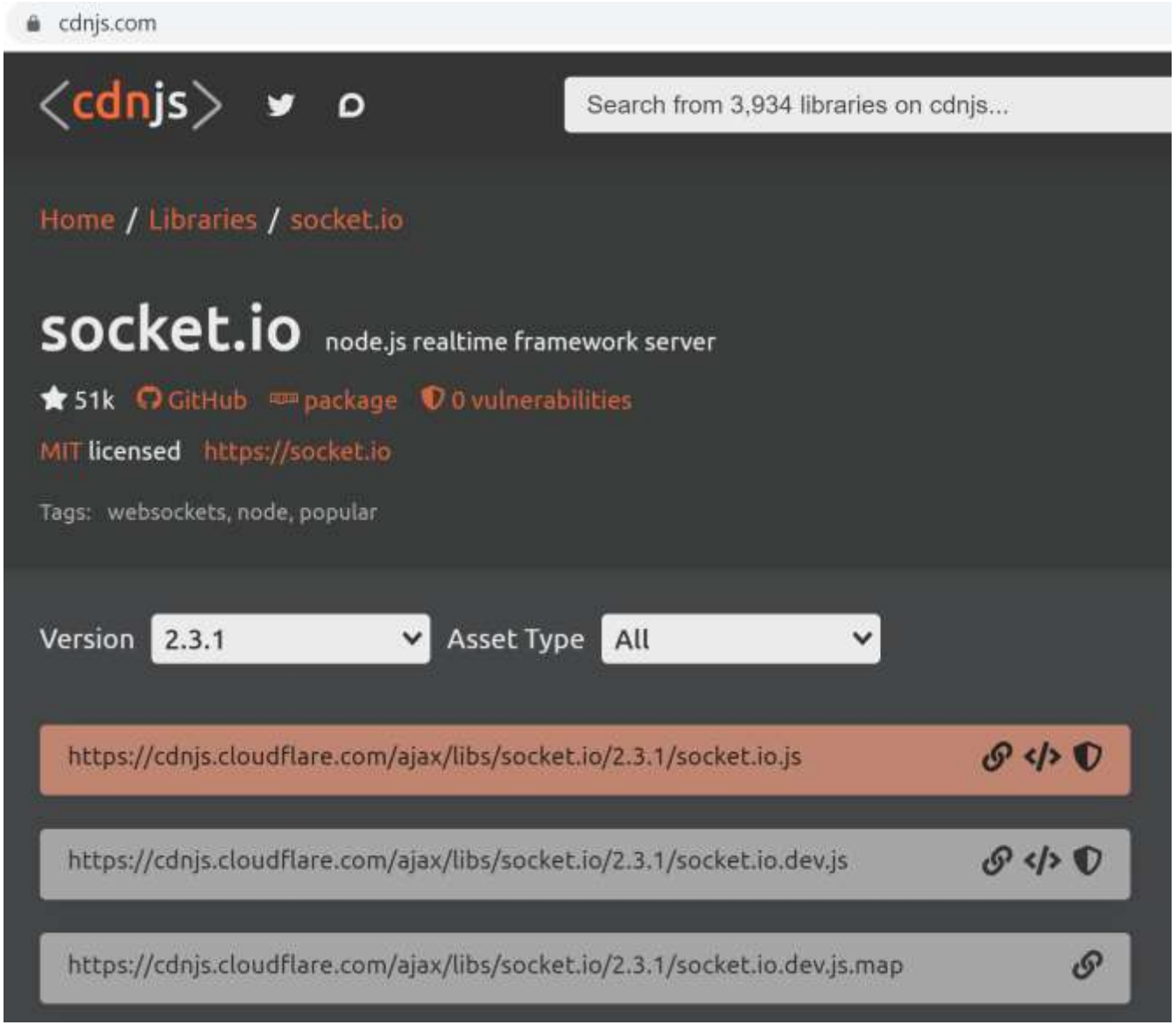
```
var readData = "";
var temp = "";
var lux = "";
var mdata = [];
var firstcommaidx = 0;

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
    mdata[2] = lux;
    console.log("AA00," + mdata);
    io.sockets.emit("message", mdata); // send data
  } else {
    console.log(readData);
  }
});
```

시간,온도,조도

Arduino data on network socket

Google search

socket.io.js cdn



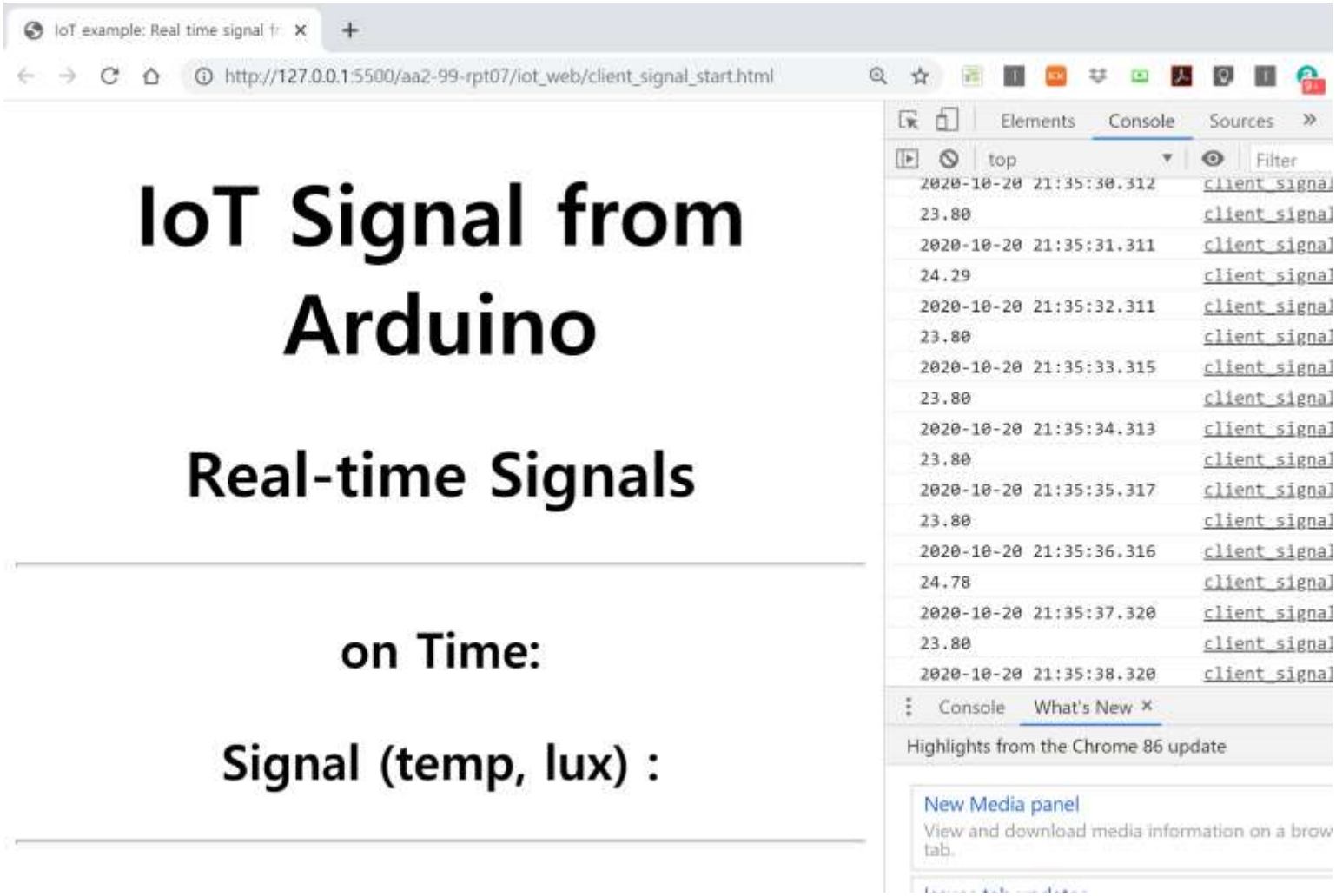
Arduino data on network socket

client_signal_start.html

```html
<!DOCTYPE html>
<head>
  <meta charset="utf-8">
  <title>IoT example: Real time signal from Arduino</title>

  <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/2.3.1/socket.io.js"></script>
  <!-- <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>
</head>

<body>  <!-- style="width:100%;height:100%"> -->

<h1 align="center"> IoT Signal from Arduino </h1>

<h2 align="center"> Real-time Signals </h2>

<hr>

<h3 align="center"> on Time: <span id="time"> </span> </h3>

<h3 align="center"> Signal (temp, lux) : <span id="data"> </span> </h3>
```

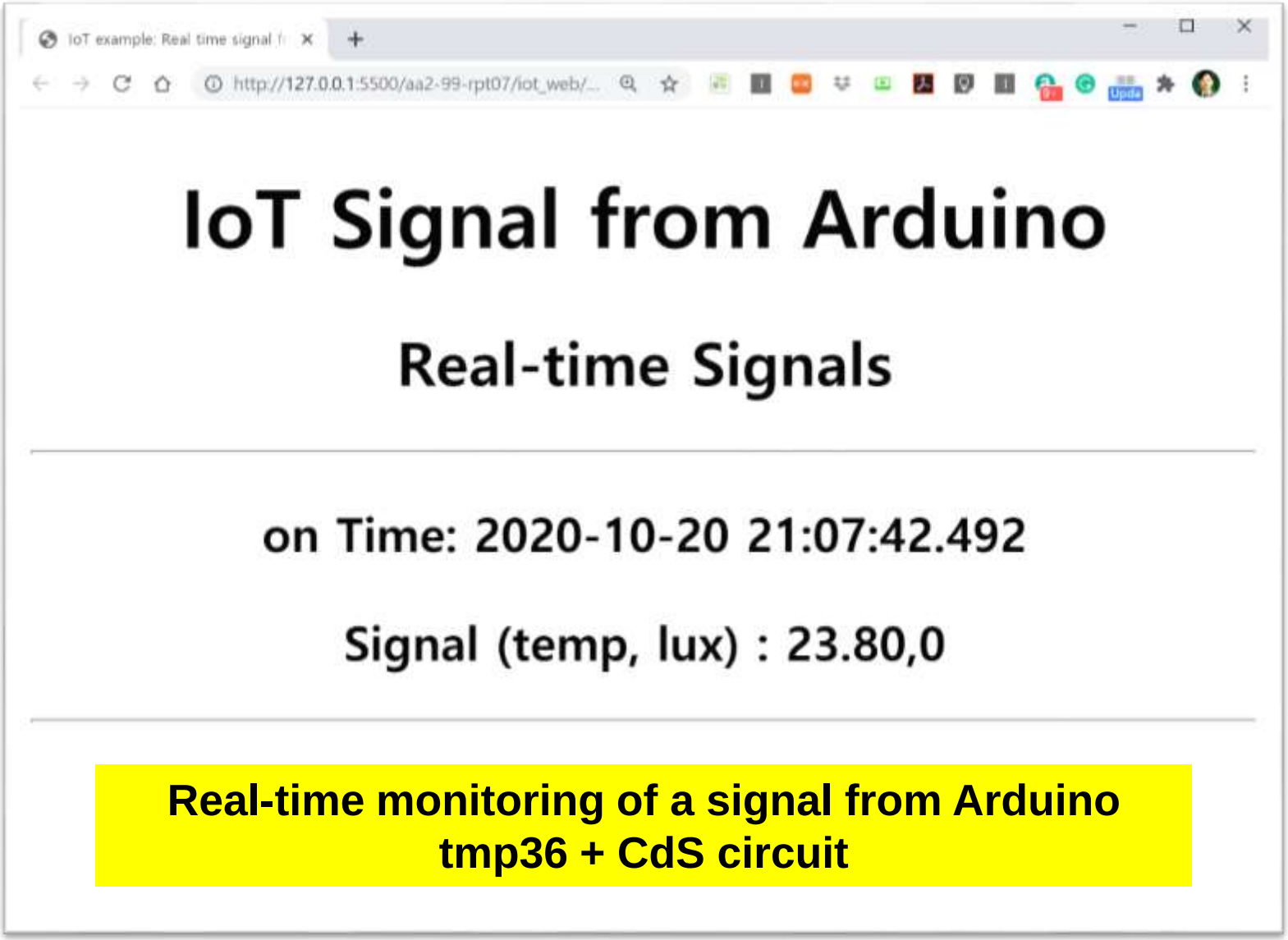
Google search : socket.io.js cdn

Arduino data on network socket



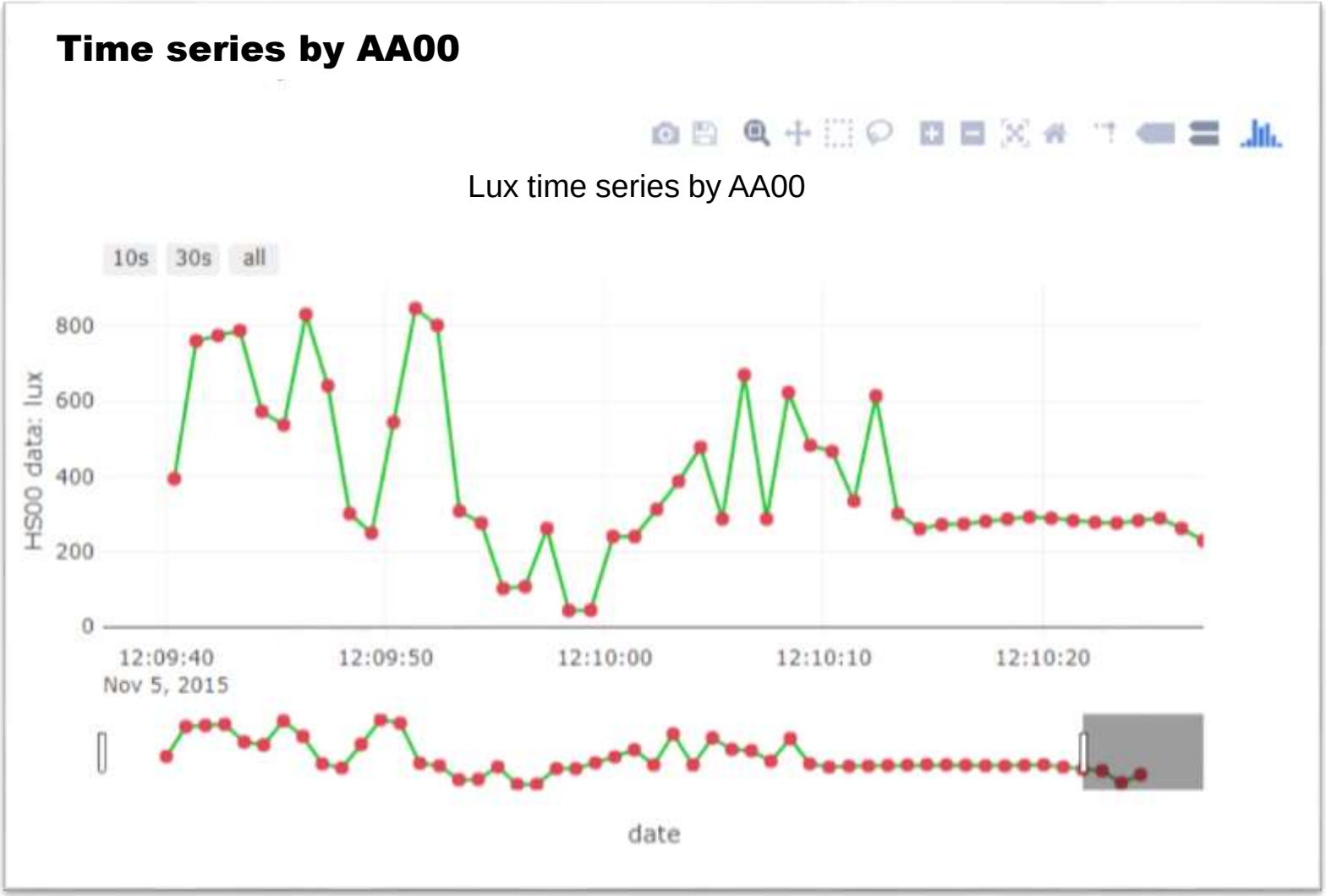
Real-time console showing a signal from Arduino in Chrome browser

Arduino data on network socket

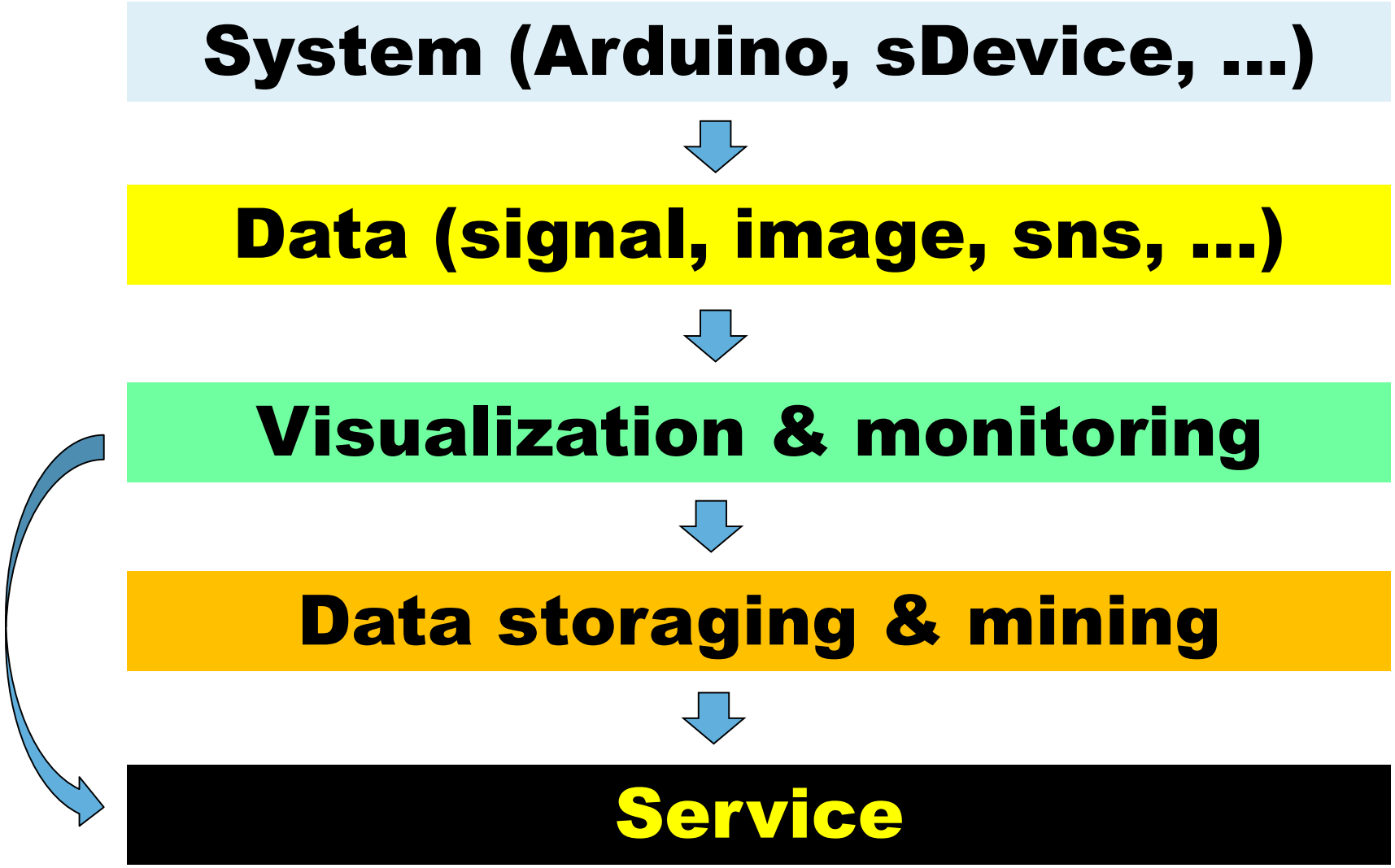


Real-time monitoring of a signal from Arduino
tmp36 + CdS circuit

Arduino data + plotly



A5. Introduction to visualization



A5.1 Introduction to data visualization

아두이노 센서 회로

↓

직렬모니터/플로터 모니터링

↓

LCD 모니터링

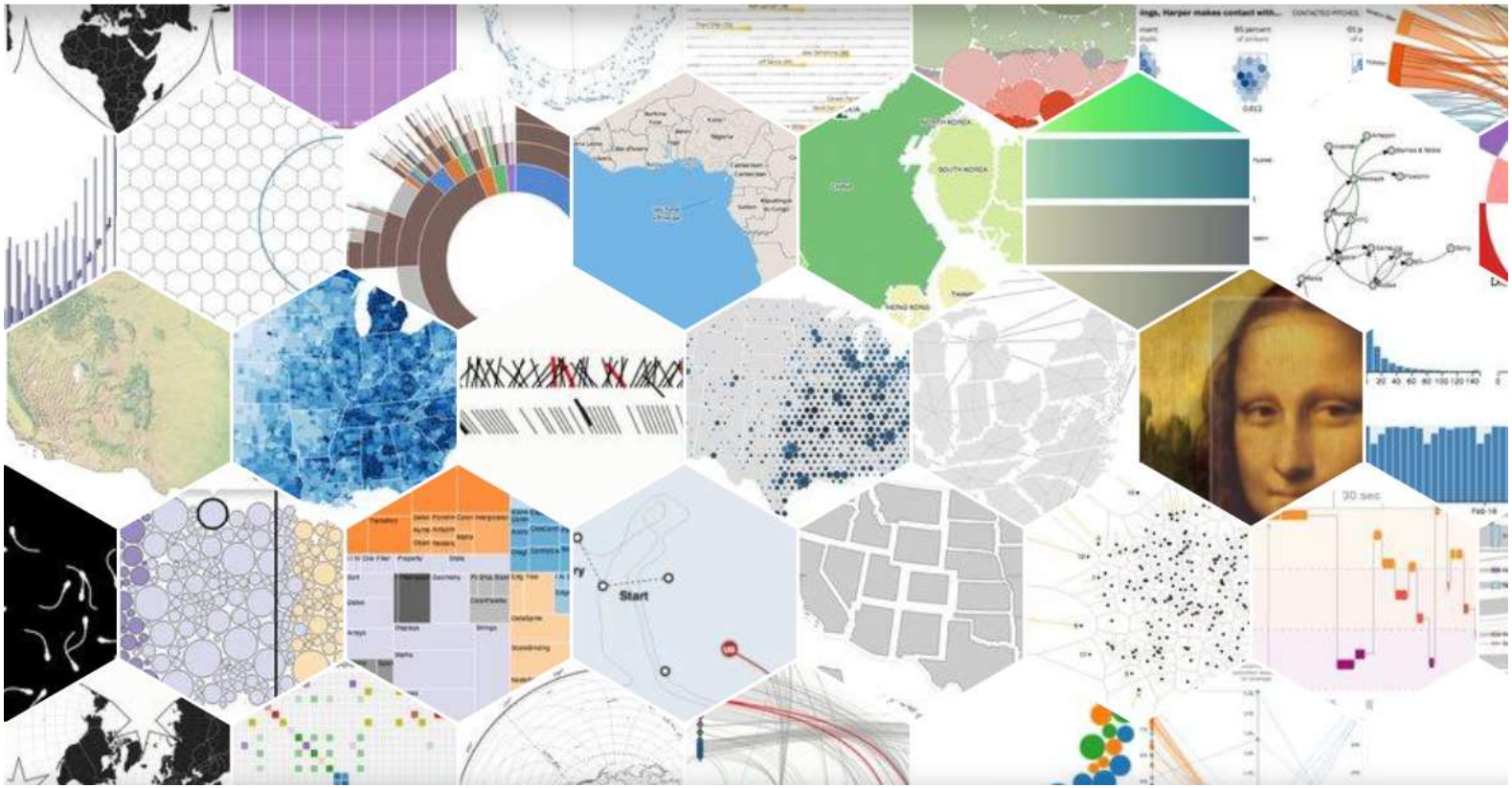
Node.js → ↓ ← Plotly.js

웹 모니터링 (시각화)

A5.1.1 D3.js

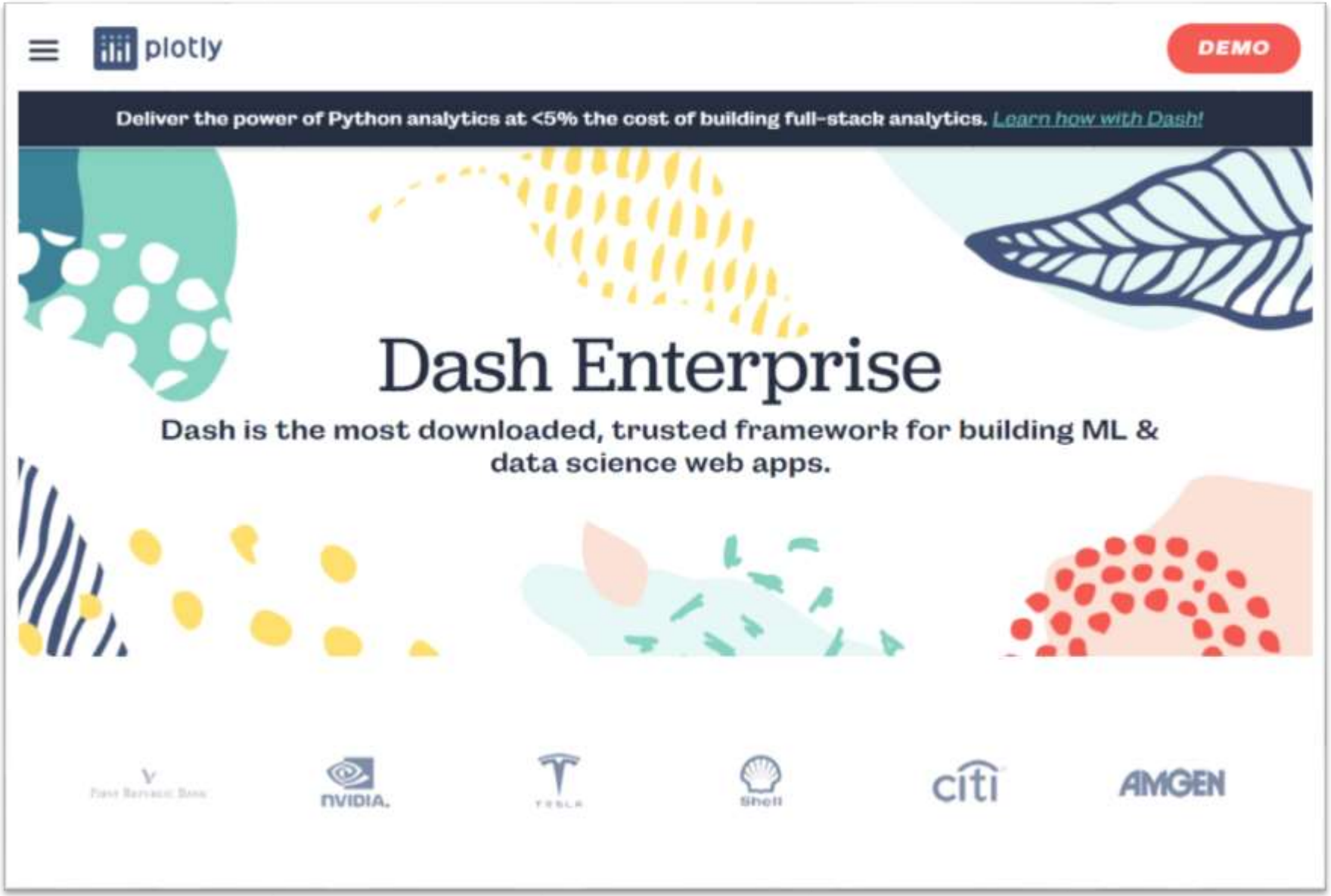
Overview Examples Documentation API Source

Data-Driven Documents



http://d3js.org/

A5.1.2 plot.ly



https://plotly.com/

A5.1.3 plotly.js

Built on top of d3.js and stack.gl,

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

plotly | Graphing Libraries

Star 12,327 DO MORE WITH DASH

Search...

Quick Start

- Getting Started
- Is Plotly Free?
- Cheat Sheet
- Figure Reference
- Function Reference
- Event Reference
- Configuration Options
- GitHub
- community.plotly.com

Examples

- Fundamentals
- Basic Charts
- Statistical Charts
- Scientific Charts
- Financial Charts
- Maps

Plotly JavaScript Open Source Graphing Library

Built on top of d3.js and stack.gl, 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.

plotly.js is free and open source and you can view the source, report issues or contribute on GitHub.

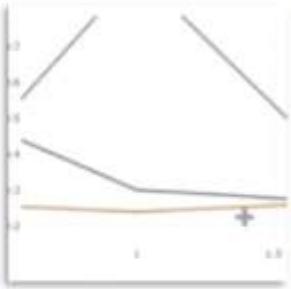
▸ Read more about plotly.js features

Fundamentals

More Fundamentals »

Configuration Options

Responsive / Fluid Layouts

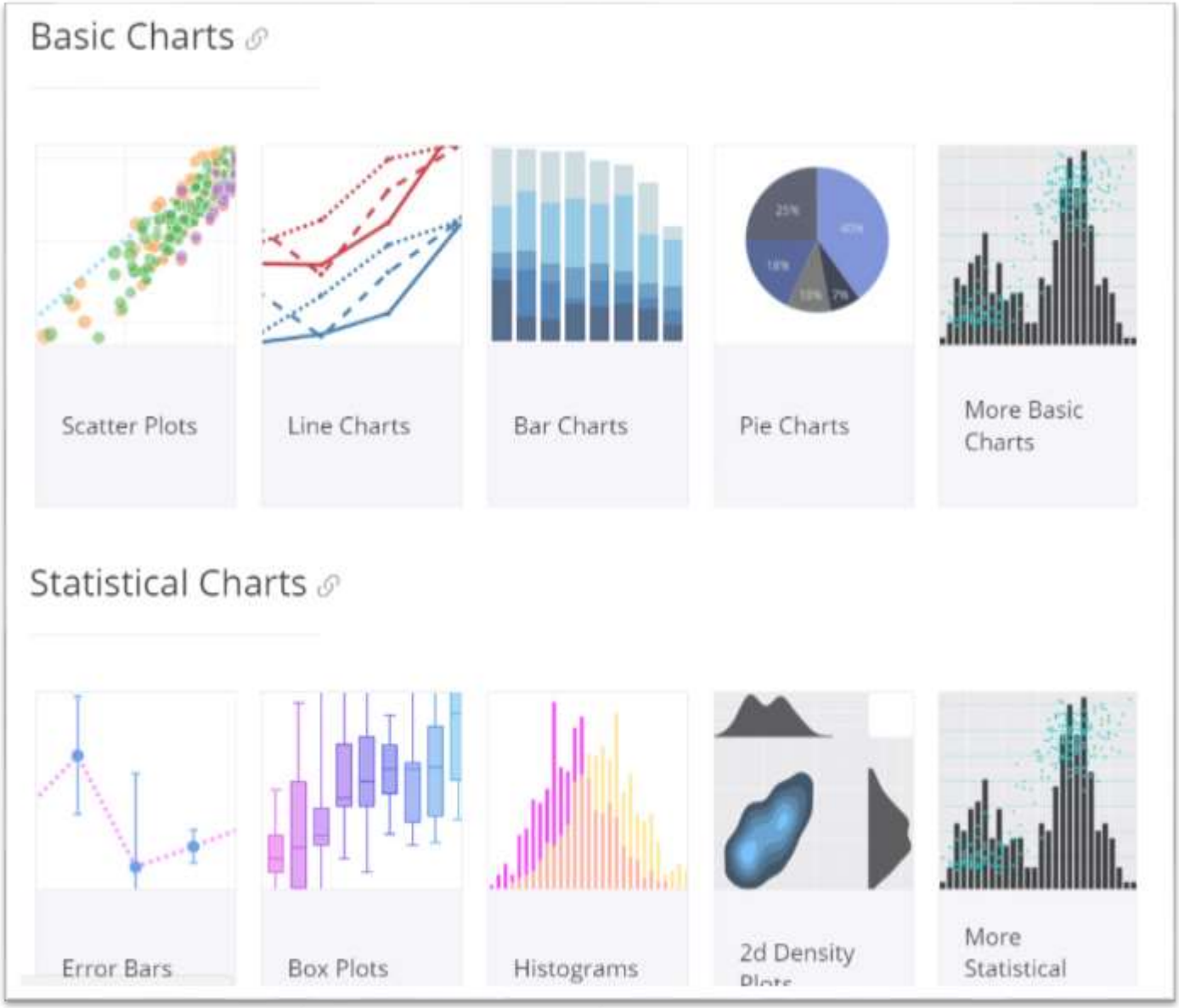


uirevision in Plotly.react

React Plotly.js

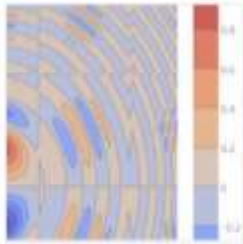
https://plotly.com/javascript/

A5.1.5 Introduction to plotly.js charts

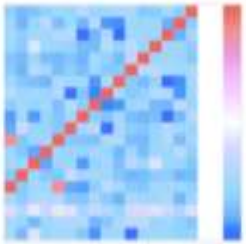


A5.1.6 Introduction to plotly.js charts

Scientific Charts

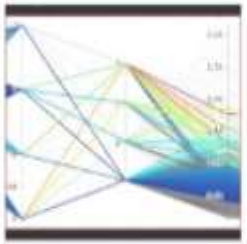


Contour Plots

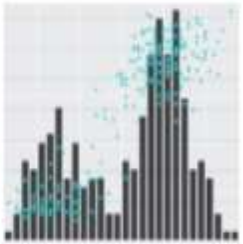


Heatmaps

Ternary Plots

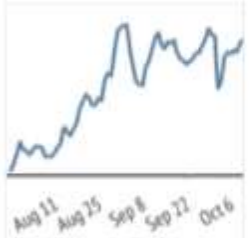


Parallel Coordinates Plot

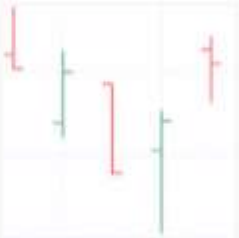


More Scientific Charts

Financial Charts



Time Series



OHLC Charts

Candlestick Charts

A5.1.7 Introduction to plotly.js charts

Maps

Choropleth Maps

Scatter Plots on Maps

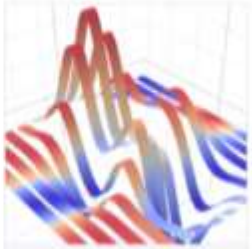
Bubble Maps

Lines on Maps

Scatter Plots on Mapbox

3D Charts

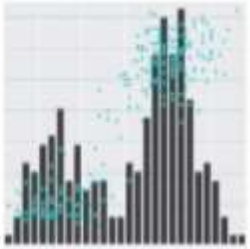
3D Scatter Plots



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

plotly.js CDN

You can also use the ultrafast plotly.js CDN link. This CDN is graciously provided by the incredible team at Fastly.

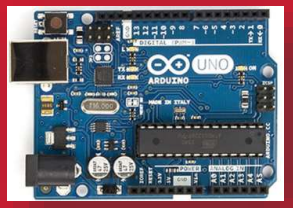
```
<head>
    <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
</head>
```

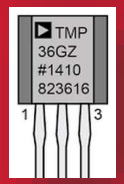
Else, if you want to get a specific version of plotly.js, say 1.2.0:

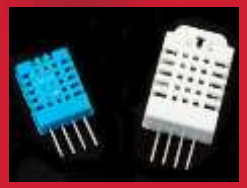
```
<head>
    <script src="https://cdn.plot.ly/plotly-1.2.0.min.js"></script>
</head>
```

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

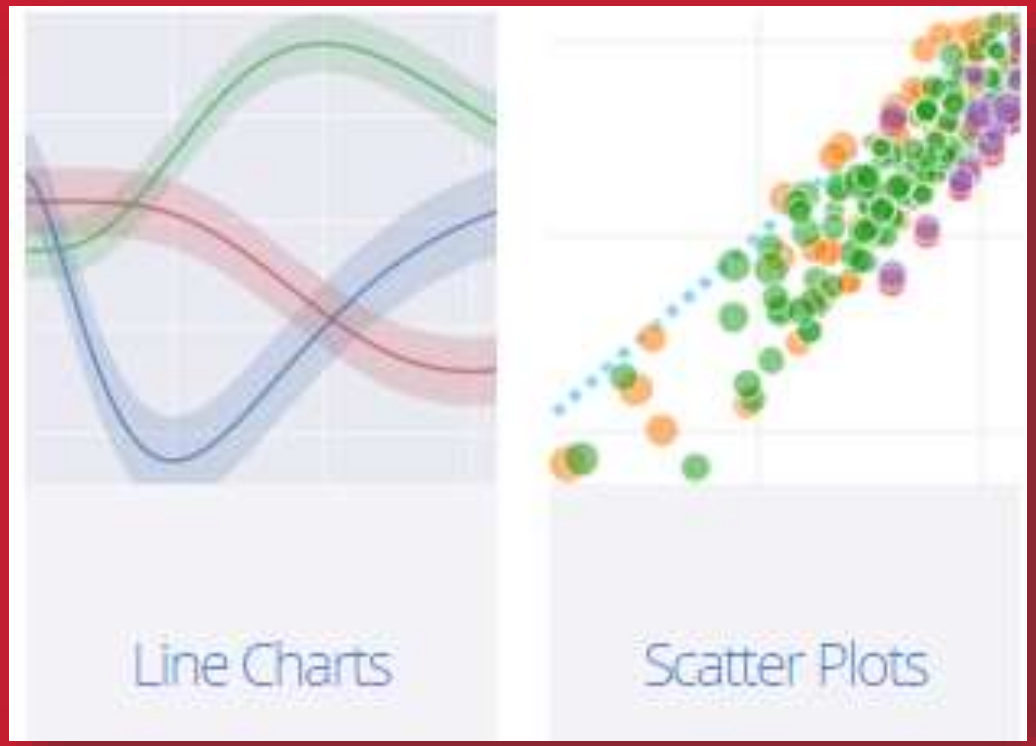
https://plot.ly/javascript/getting-started/







Data charts using plotly.js



A5.2 Data charts

Navigation

Back To Plotly.Js

Line Charts in plotly.js

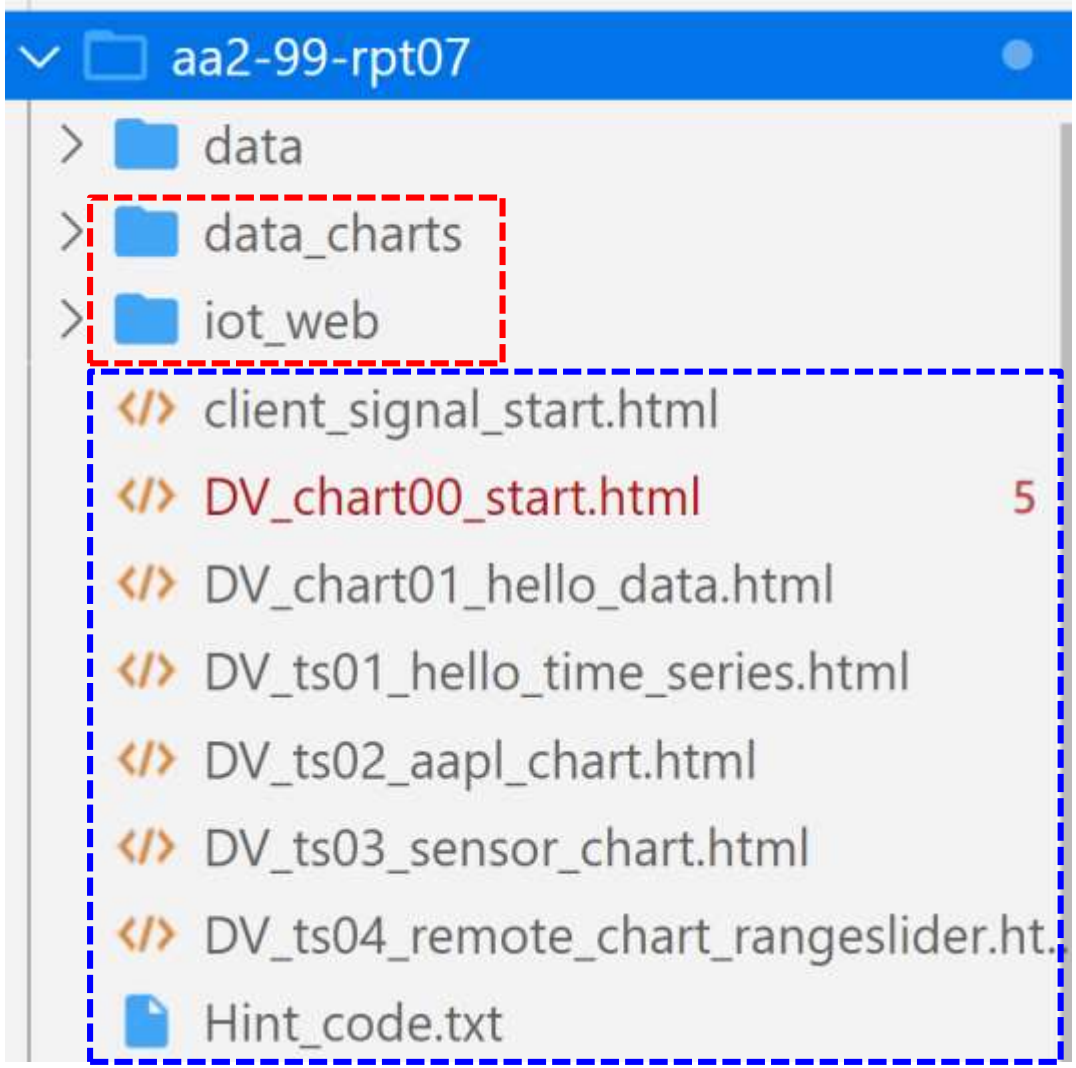
How to make D3.js-based line charts in JavaScript.

R | plotly.js | Python | Pandas

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



A5.2.2.1 Starting plotly basic chart

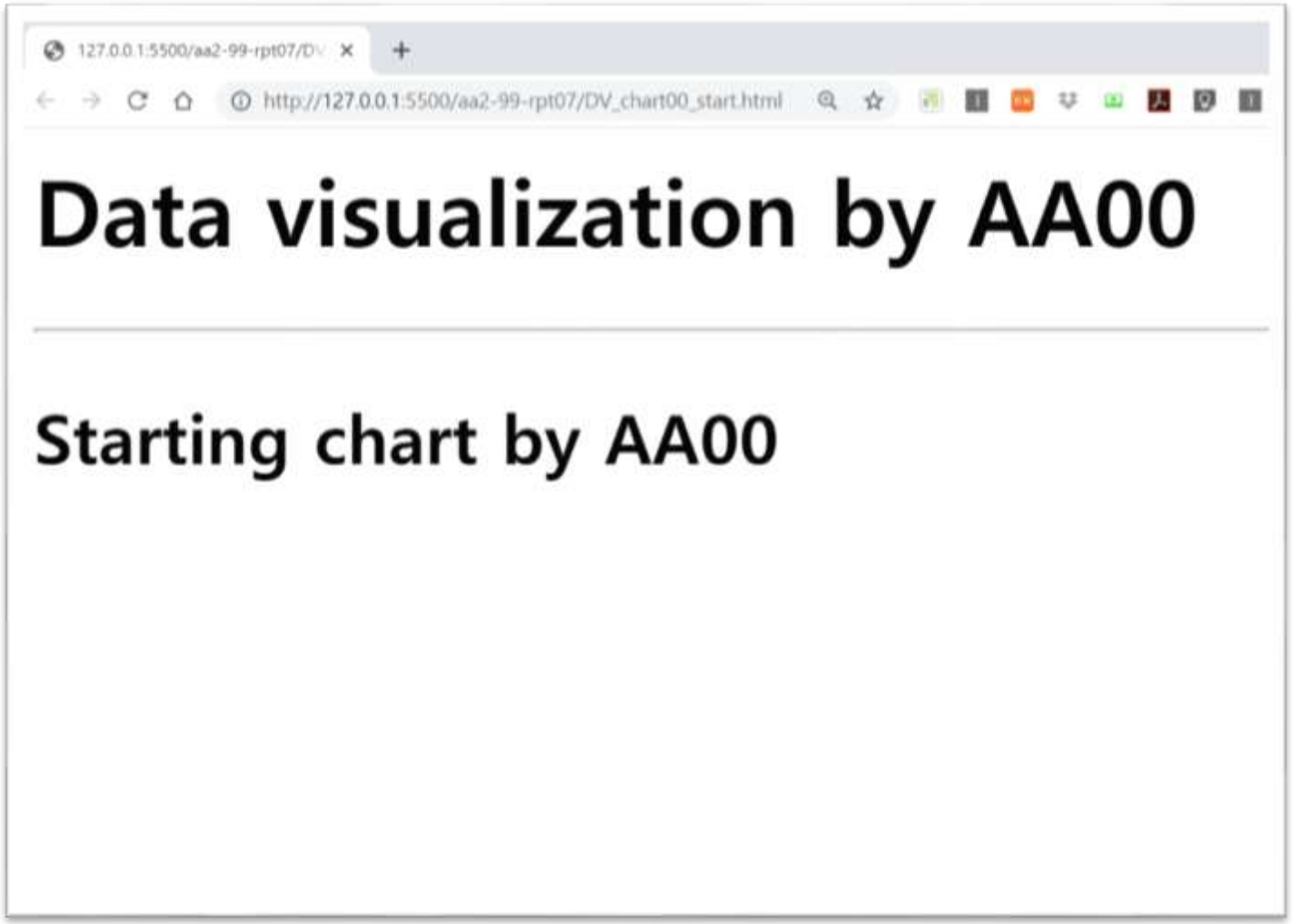
DV_chart00_start.html

Starting chart!

```html
<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>
    <hr>
    <h2>Starting graph by AA00</h2>

    <!-- Plotly chart will be drawn inside this DIV -->
    <div id="myDiv" style="width: 500px;height: 300px"></div>

    <script>
        <!-- JAVASCRIPT CODE GOES HERE -->


    </script>
</body>
</html>
```

A5.2.2.2 Starting plotly basic chart

VSCode, live server



A5.2.3.1 Hello plotly basic chart

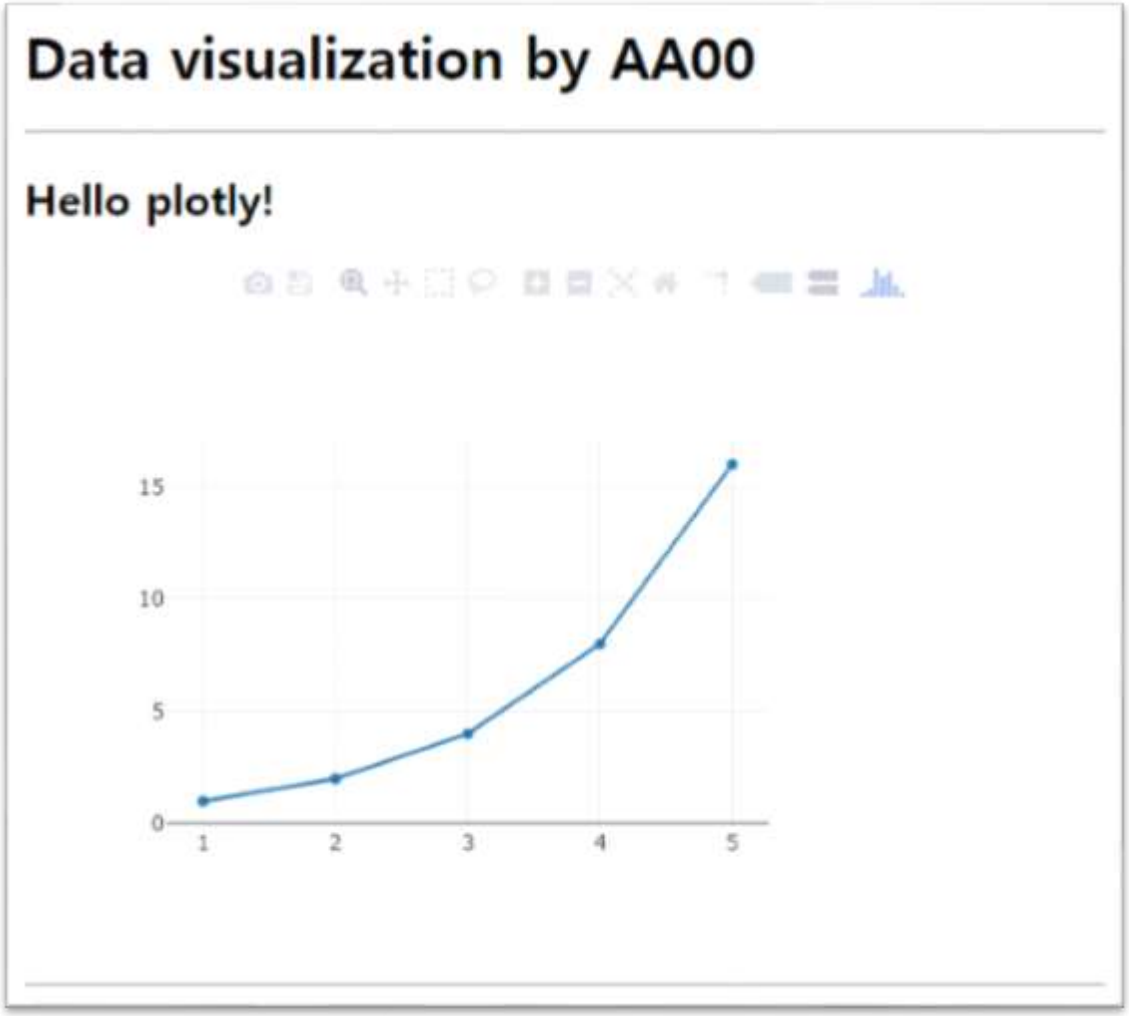
Hello plotly data chart!

```html
<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>
    <hr>
    <h2>Hello plotly!</h2>
    <!-- Plotly chart will be drawn inside this DIV -->
    <div id="myDiv" style="width: 500px;height: 400px"></div>
    <hr>
    <script>
        <!-- JAVASCRIPT CODE GOES HERE -->
        var data = [
        {
            x: [1, 2, 3, 4, 5],
            y: [1, 2, 4, 8, 16],
            type: 'scatter'
        }];

        Plotly.newPlot('myDiv', data);

    </script>
</body>
</html>
```

data는 무엇?
그래프 객체들의 구조,
데이터 배열

A5.2.3.2 Hello plotly basic chart

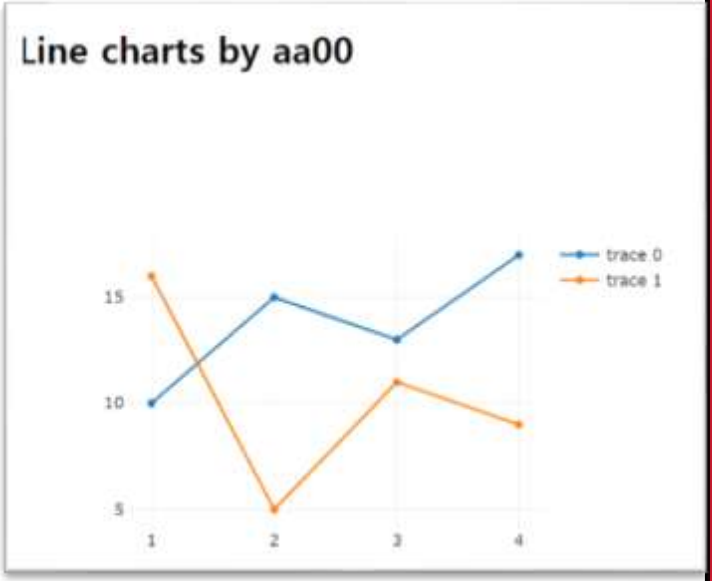
Graph : Hello plotly chart!



A5.2.4 plotly.js: Line Charts

[1] Basic multi-line charts

```
<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>
```

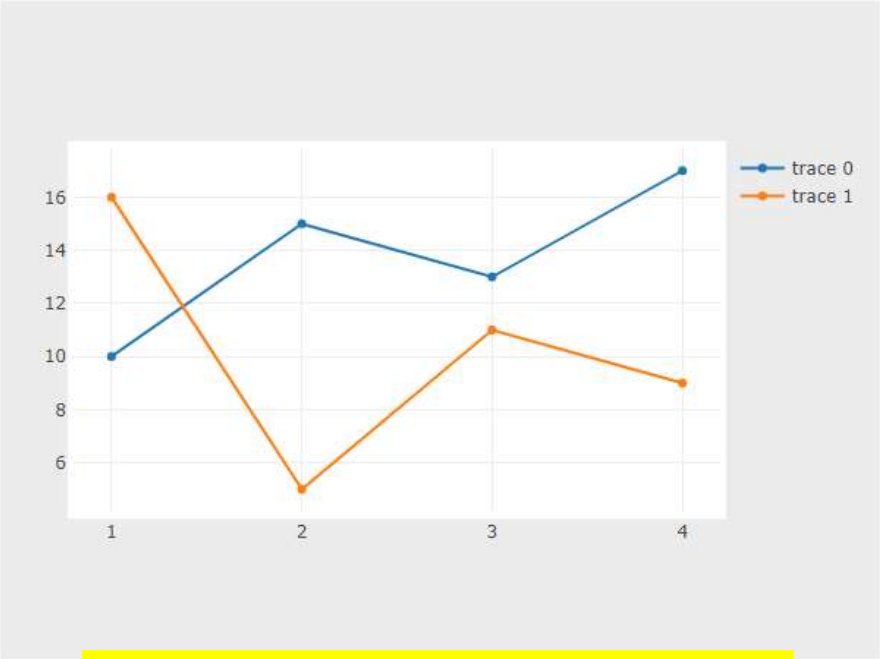


A5.2.5 plotly.js: Line Charts

[2] Basic line charts with layout

```
var layout = {
    autosize: false,
    width: 600,
    height: 450,
    margin: {
        l: 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);
```

Line charts with layout by AA00



AAnn_Chart_Layout.png

Test: pad → 40

A5.2.6.1 plotly.js: Line & Scatter plot

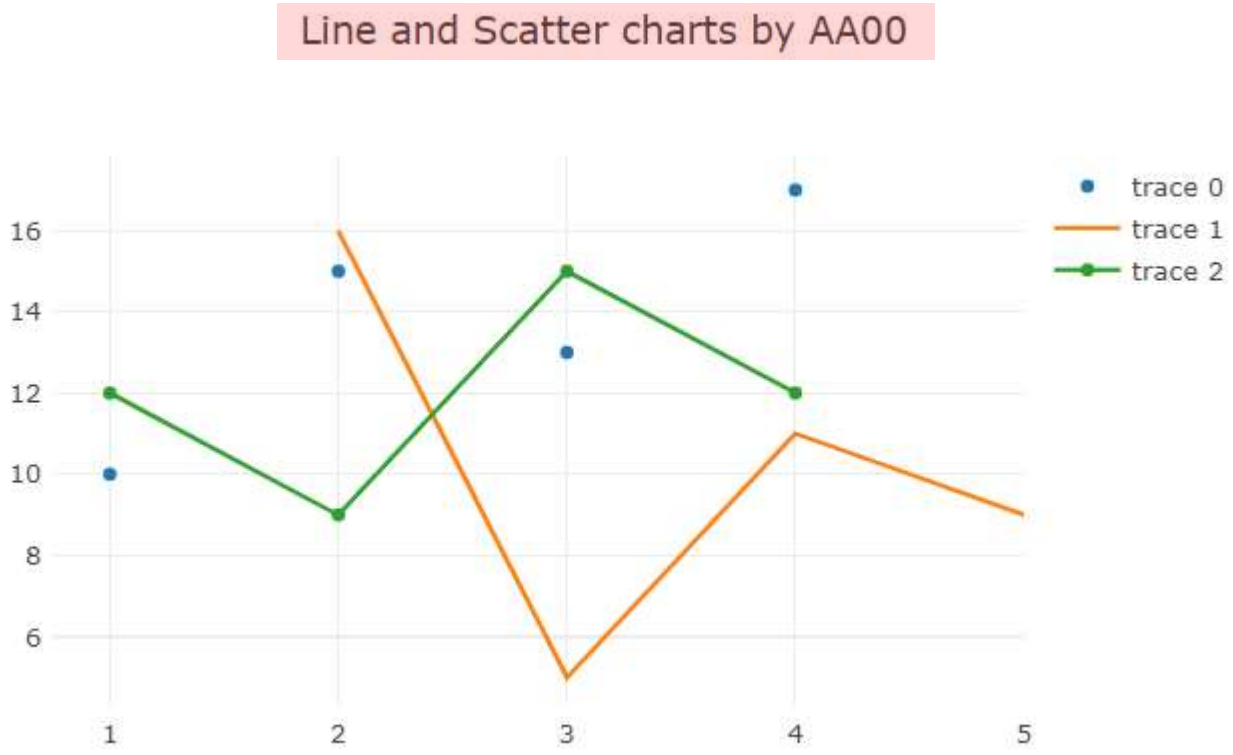
[3] Line & scatter plot : setting mode

```
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: {
        l: 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 title



A5.2.6.3 plotly.js: Line & Scatter plot

[3.2] Line & scatter plot with axis name

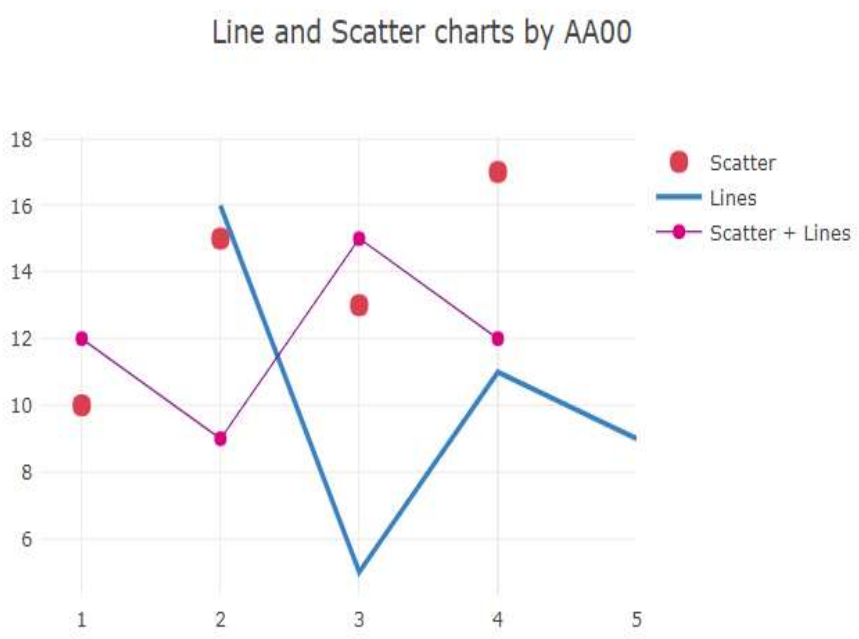
```
var trace1 = {
    x: [1, 2, 3, 4],
    y: [10, 15, 13, 17],
    mode: 'markers',
    name: 'Scatter'
};

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'
};
```

A5.2.6.4 plotly.js: Line & Scatter plot

[3.3] Line & scatter plot with style

```
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
  }
};
```

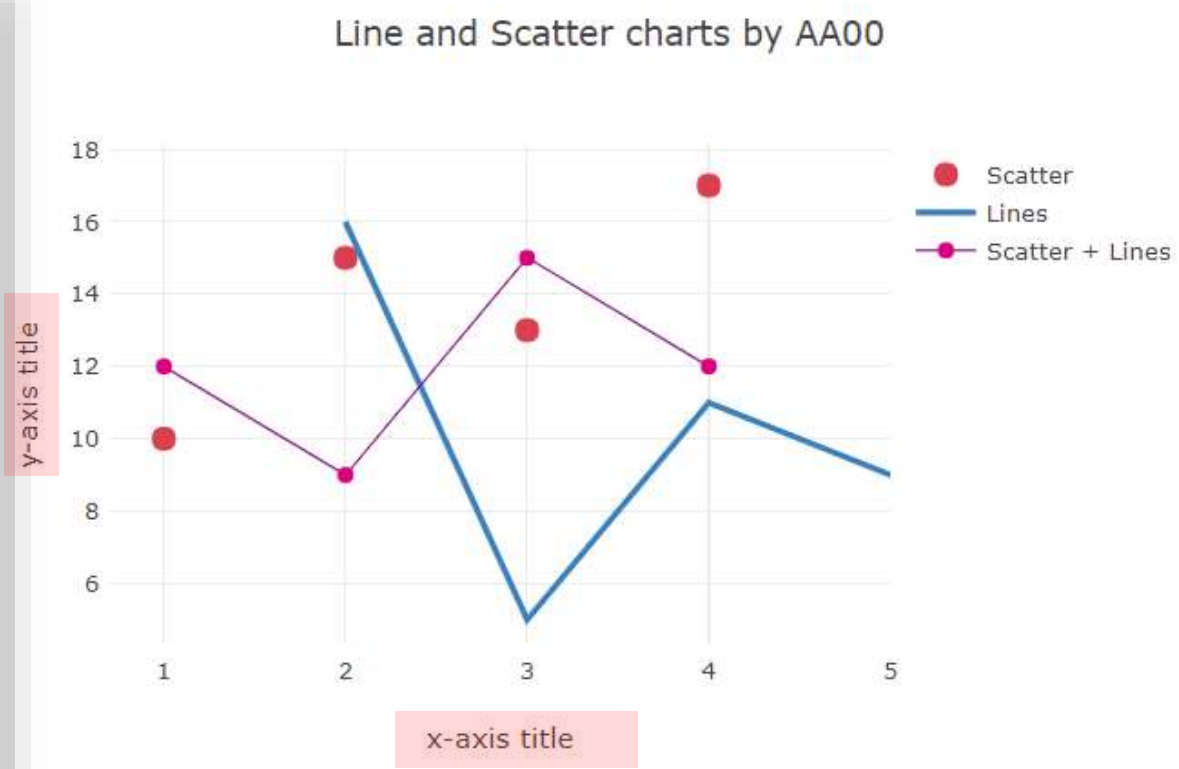
```
var trace3 = {
  x: [1, 2, 3, 4],
  y: [12, 9, 15, 12],
  mode: 'lines+markers',
  name: 'Scatter + Lines',
  marker: {
    color: 'rgb(128, 0, 128)',
    size: 8
  },
  line: {
    color: 'rgb(128, 0, 128)',
    width: 1
  }
};
```



A5.2.6.5 plotly.js: Line & Scatter plot

[3.4] Line & scatter plot with axis titles

```
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'
  }
};
```



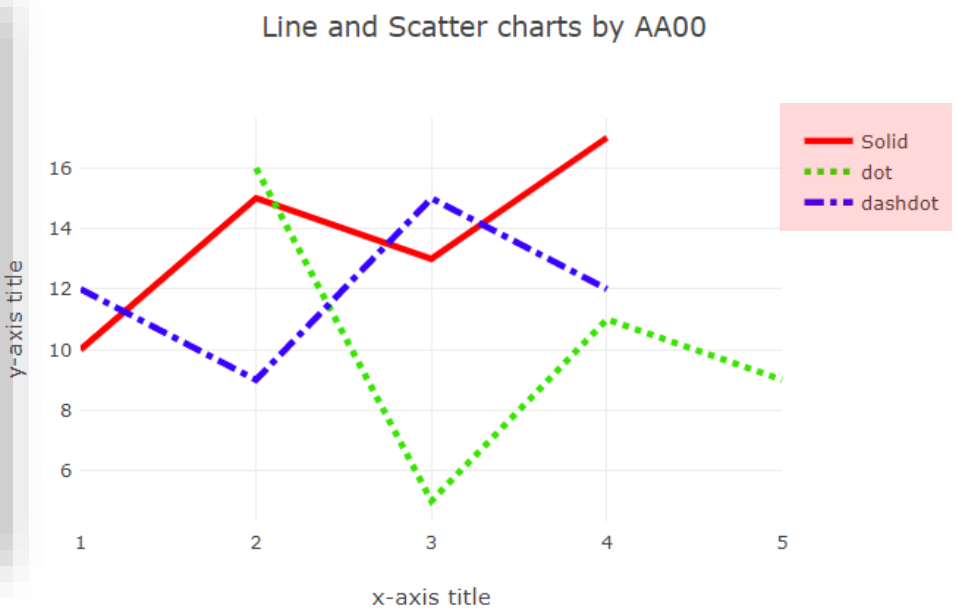
AAnn_Axis_Title.png

A5.2.6.6 plotly.js: Line & Scatter plot

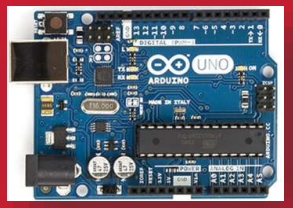
[3.5] Line & scatter plot with dash and dot

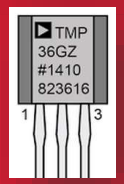
```
var trace1 = {
  x: [1, 2, 3, 4],
  y: [10, 15, 13, 17],
  mode: 'lines',
  name: 'Solid',
  line: {
    color: 'rgb(255, 0, 0)',
    dash: 'solid',
    width: 4
  }
};

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
  }
};
```

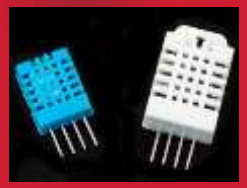
```
var trace3 = {
  x: [1, 2, 3, 4],
  y: [12, 9, 15, 12],
  mode: 'lines',
  name: 'dashdot',
  line: {
    color: 'rgb(55, 0, 255',
    dash: 'dashdot',
    width: 4
  }
};
```



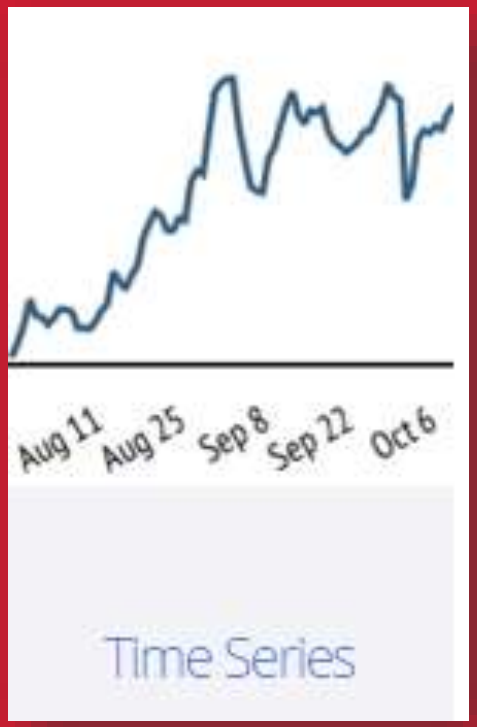
AAnn_Line_Dash_Dot.png







Data visualization using plotly.js



A5.3. Time series

Navigation

Date Strings

Basic Time Series

Manually Set Range

Time Series with Rangeslider

Back To Plotly.Js

Time Series in plotly.js

How to plot D3.js-based date and time in Plotly.js. An example of a time-series plot.

R | Python | matplotlib | plotly.js | Pandas | node.js | MATLAB

Date Strings

```
var data = [
  {
    x: ['2013-10-04 22:23:00', '2013-11-04 22:23:00', '2013-12-04 22:23:00'],
    y: [1, 3, 6],
    type: 'scatter'
  }
];

Plotly.newPlot('myDiv', data);
```

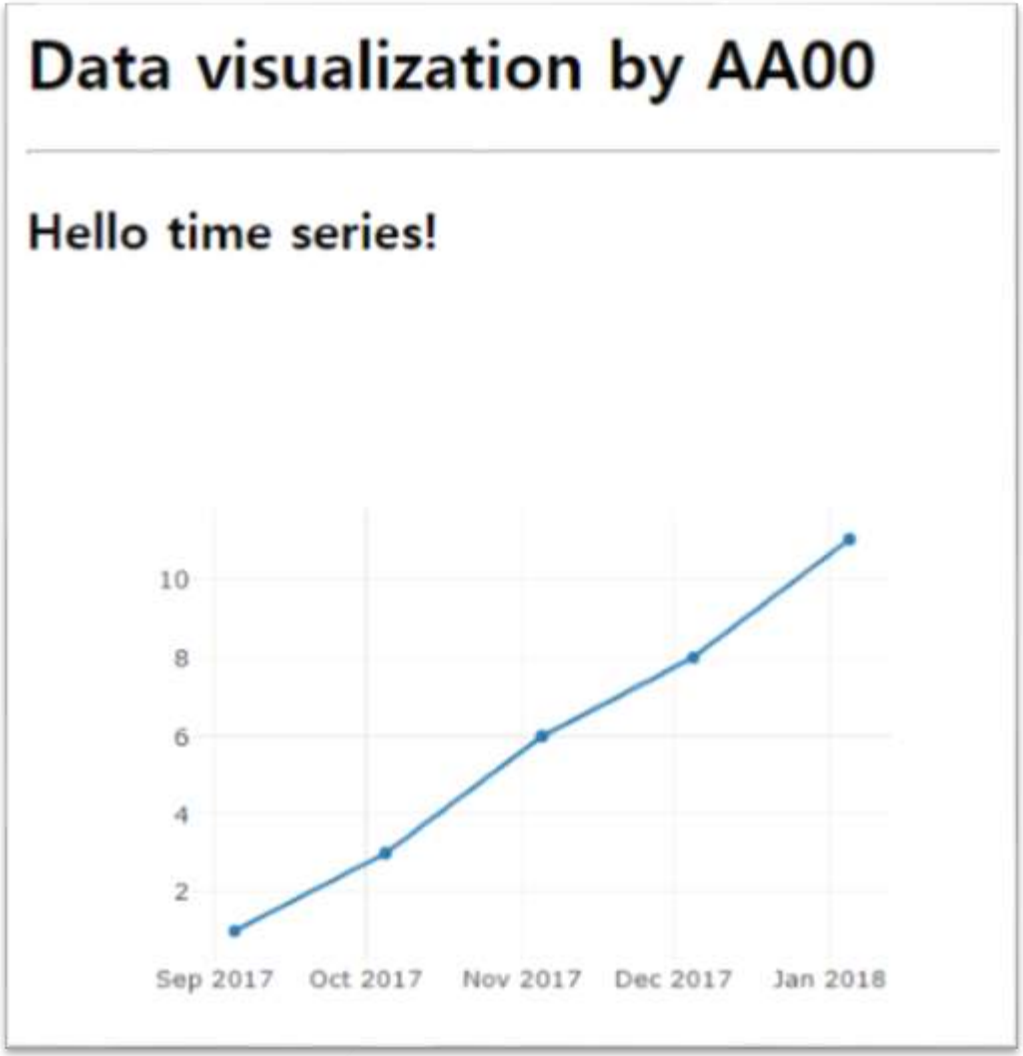
A5.3.1 plotly.js: Time series

[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'],
        y: [1, 3, 6, 8],
        type: 'scatter'
    }
    ];

    Plotly.newPlot('myDiv', data);

</script>
```

A5.3.2 plotly.js: Time series

Time series : date strings – result



오늘 날자와
데이터를 추가

A5.3.3.1 plotly.js: Time series

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

안전함 | https://raw.githubusercontent.com/plotly/datasets/master/finance-charts-a...

```
Date,AAPL.Open,AAPL.High,AAPL.Low,AAPL.Close,AAPL.Volume,AAPL.Adjusted,dn,mavg,up,direction
2015-02-17,127.489998,128.880005,126.919998,127.830002,63152400,122.905254,106.7410523,117.9276669,129.1142814,Increasing
2015-02-18,127.629997,128.779999,127.449997,128.720001,44891700,123.760965,107.842423,118.9403335,130.0382439,Increasing
2015-02-19,128.479996,129.029999,128.330002,128.449997,37362400,123.501363,108.8942449,119.8891668,130.8840887,Decreasing
2015-02-20,128.619995,129.5,128.050003,129.5,48948400,124.510914,109.7854494,120.7635001,131.7415509,Increasing
2015-02-23,130.020004,133,129.660004,133,70974100,127.876074,110.3725162,121.7201668,133.0678174,Increasing
2015-02-24,132.940002,133.600006,131.169998,132.169998,69228100,127.078049,111.0948689,122.6648335,134.2347981,Decreasing
2015-02-25,131.559998,131.600006,128.149994,128.789993,74711700,123.828261,113.2119183,123.6296667,134.0474151,Decreasing
2015-02-26,128.789993,130.869995,126.610001,130.419998,91287500,125.395469,114.1652991,124.2823333,134.3993674,Increasing
2015-02-27,130,130.570007,128.240005,128.460007,62014800,123.510987,114.9668484,124.8426669,134.7184854,Decreasing
2015-03-02,129.25,130.279999,128.300003,129.089996,48096700,124.116706,115.8770904,125.4036668,134.9302432,Decreasing
2015-03-03,128.960007,129.520004,128.089996,129.360001,37816300,124.376308,116.9535132,125.9551669,134.9568205,Increasing
2015-03-04,129.100006,129.559998,128.320007,128.539993,31666300,123.587892,118.0874253,126.4730002,134.8585751,Decreasing
2015-03-05,128.580002,128.75,125.760002,126.410004,56517100,121.539962,119.1048311,126.848667,134.5925029,Decreasing
2015-03-06,128.399994,129.369995,126.260002,126.599998,72842100,121.722637,120.190797,127.2288335,134.26687,Decreasing
2015-03-09,127.959999,129.570007,125.059998,127.139999,88528500,122.241834,121.6289771,127.631167,133.6333568,Decreasing
2015-03-10,126.410004,127.220001,123.800003,124.510002,68856600,119.71316,123.1164763,127.9235004,132.7305246,Decreasing
2015-03-11,124.75,124.769997,122.110001,122.239998,68939000,117.530609,123.592756,128.0033337,132.4139113,Decreasing
2015-03-12,122.309998,124.900002,121.629997,124.449997,48362700,119.655466,123.4894559,127.9813337,132.4732114,Increasing
2015-03-13,124.400002,125.400002,122.580002,123.589996,51827300,118.828598,123.045606,127.8490003,132.6523946,Decreasing
2015-03-16,123.879997,124.949997,122.870003,124.949997,35874300,120.136203,122.6967016,127.7283335,132.7599655,Increasing
2015-03-17,125.900002,127.32,125.650002,127.040001,51023100,122.145688,122.616033,127.6680002,132.7199674,Increasing
2015-03-18,127,129.160004,126.370003,128.470001,65270800,123.520597,122.6064498,127.652167,132.6978842,Increasing
2015-03-19,128.75,129.25,127.400002,127.5,45809500,122.587966,122.5939029,127.6245004,132.6550979,Decreasing
2015-03-20,128.25,128.399994,125.160004,125.900002,68695100,121.049608,122.4865925,127.4980004,132.5094083,Decreasing
2015-03-23,127.120003,127.849998,126.519997,127.209999,37709700,122.309137,122.6741703,127.2633335,131.8524968,Increasing
2015-03-24,127.230003,128.039993,126.559998,126.690002,32842300,121.809174,123.0410183,127.0025001,130.9639818,Decreasing
2015-03-25,126.540001,126.82,123.379997,123.379997,51655200,118.626689,122.8276392,126.7531667,130.6786943,Decreasing
2015-03-26,122.760002,124.879997,122.599998,124.239998,47572900,119.453558,122.5538523,126.4835001,130.4131478,Increasing
2015-03-27,124.57,124.699997,122.910004,123.25,39546200,118.5017,122.2826504,126.2099998,130.1373491,Decreasing
2015-03-30,124.050003,126.400002,124,126.370003,47099700,121.501502,122.346906,126.0283332,129.7097604,Increasing
2015-03-31,126.089996,126.489998,124.360001,124.43,42090600,119.63624,122.395242,125.8334998,129.2717577,Decreasing
2015-04-01,124.82,125.120003,123.099998,124.25,40621400,119.463174,122.3761274,125.6009999,128.8258723,Decreasing
```

A5.3.3.2 plotly.js: Time series

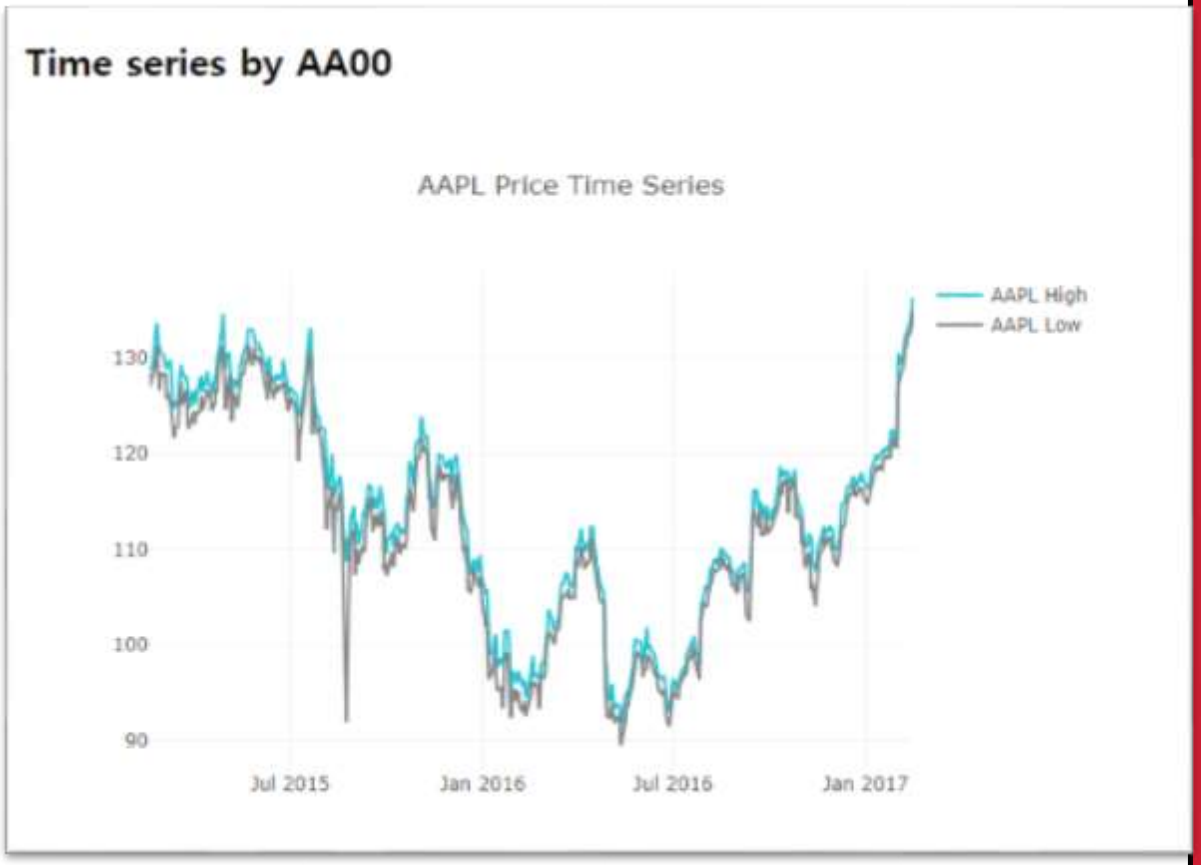
[2] Time series : financial data strings – AAPL stock price

```
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

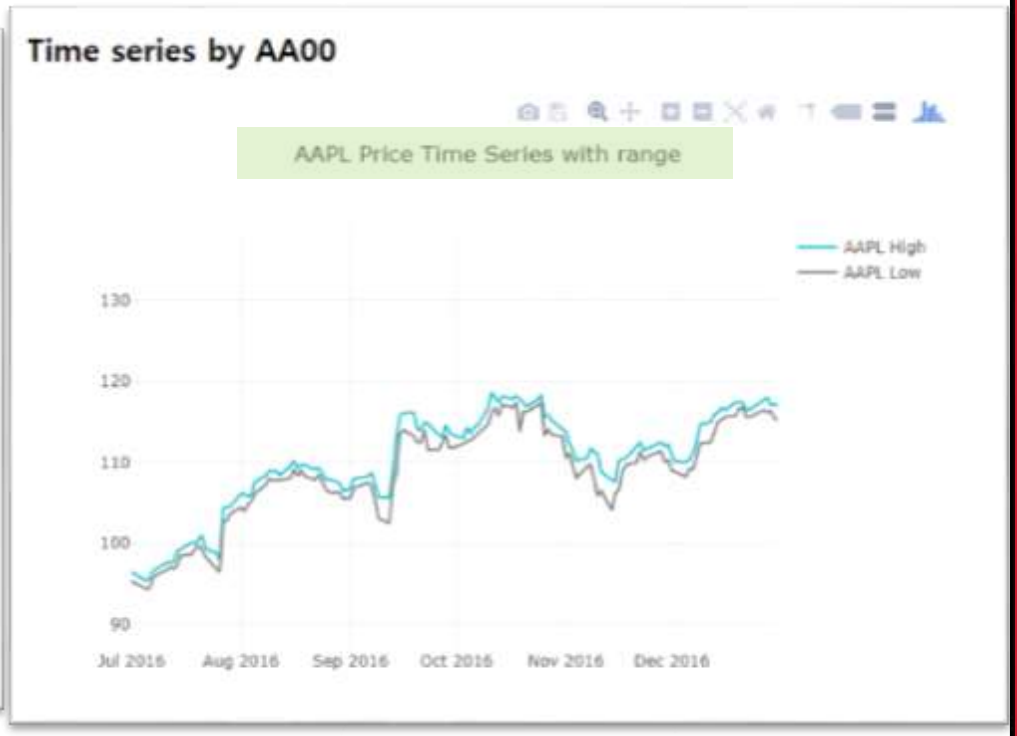
```
var data = [trace1,trace2];

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

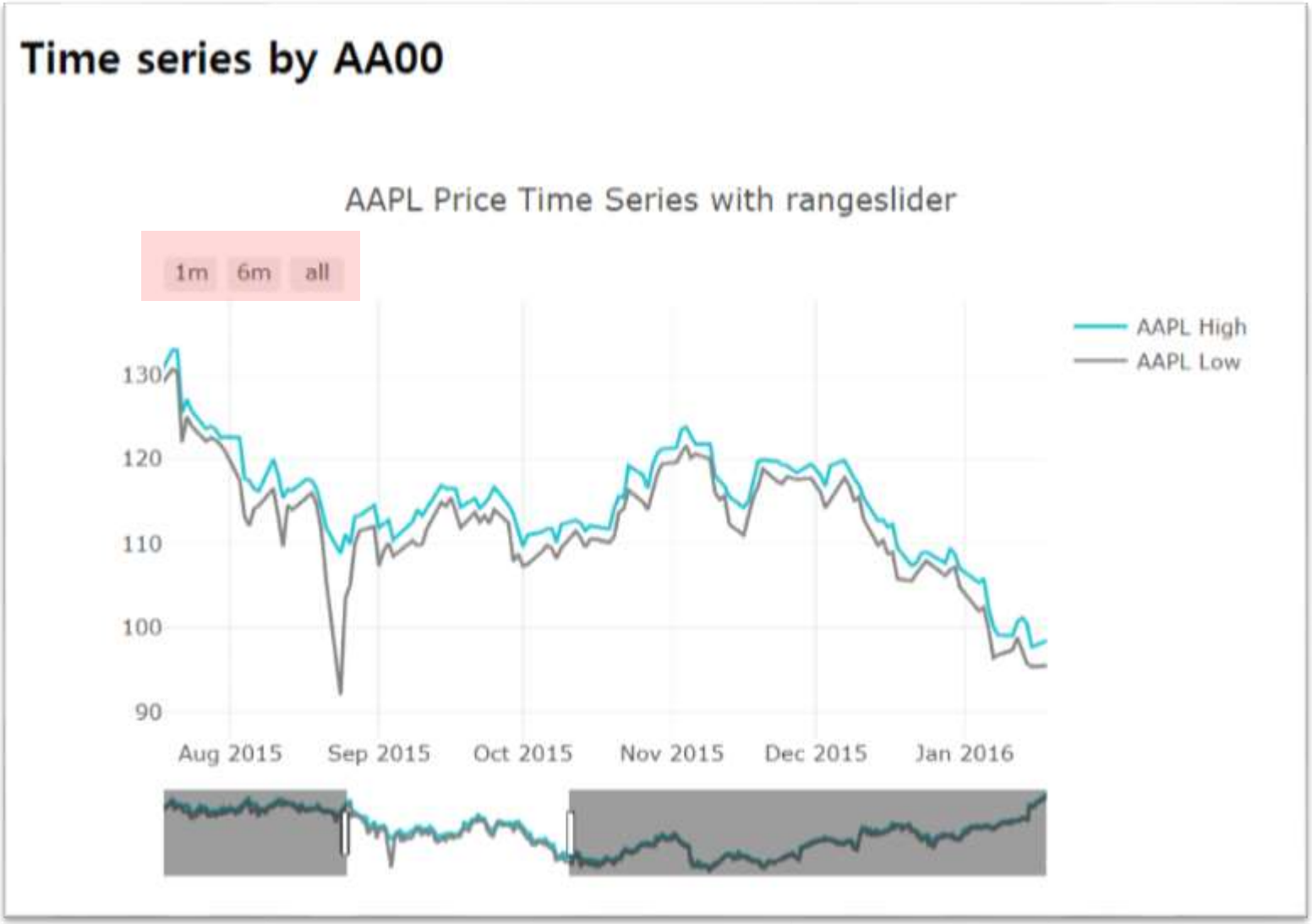
A5.3.3.5 plotly.js: Time series

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

```
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'}
            ]},
            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



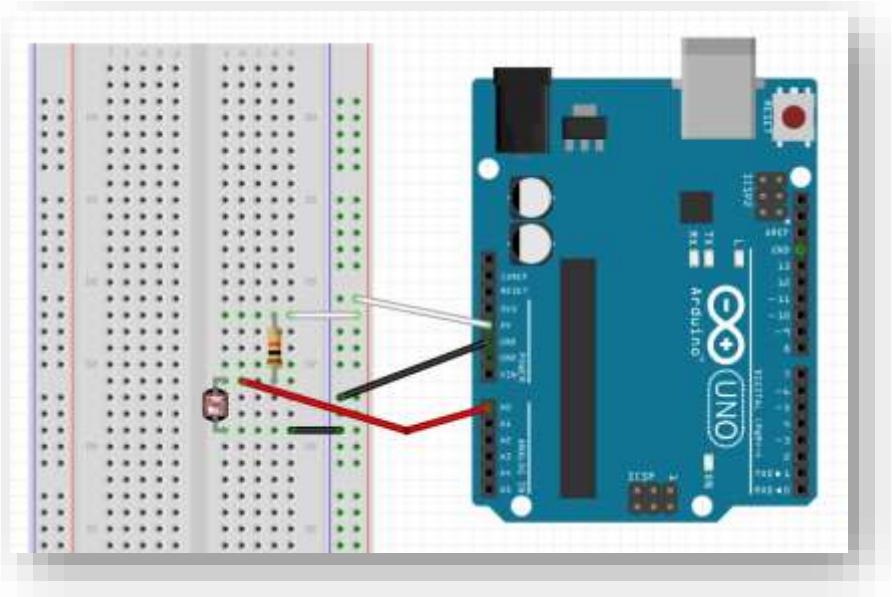
A5.3.4.1 plotly.js: Sensor time series

[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',
```

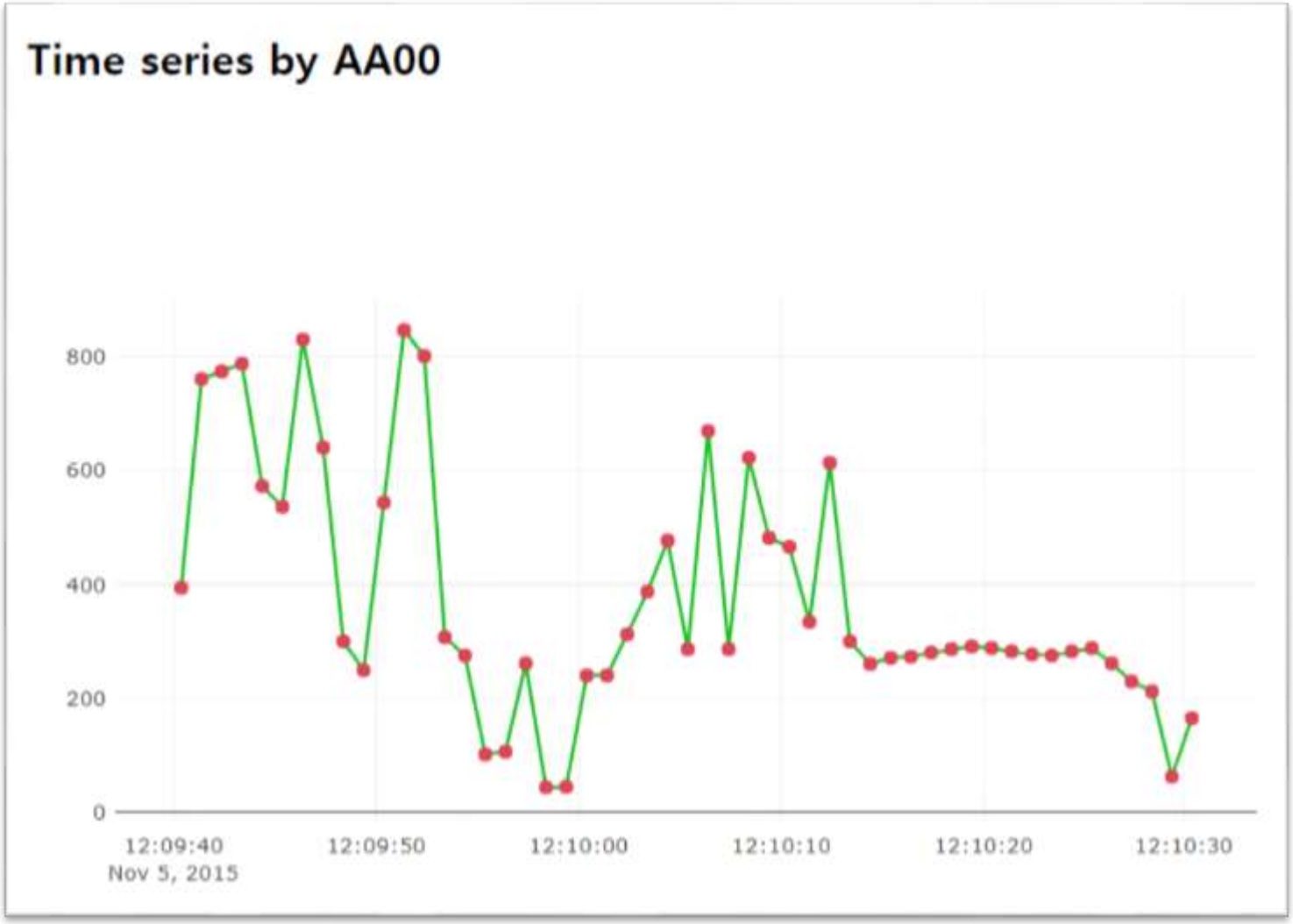
Data :

date,value



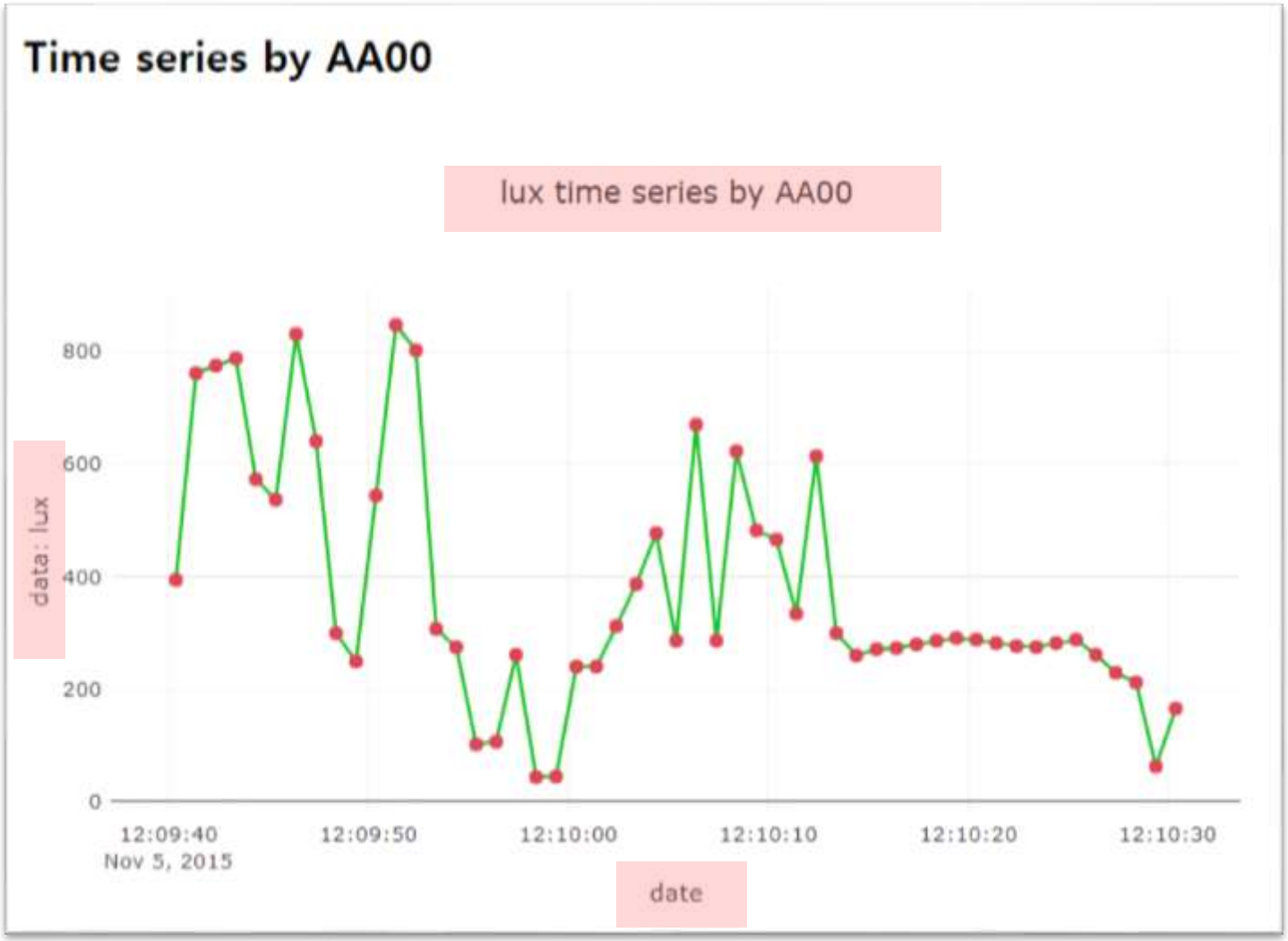
A5.3.4.2 plotly.js: Time series

[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

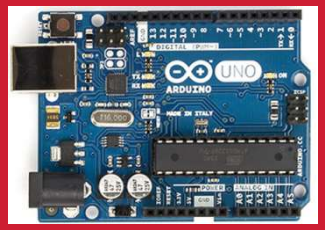


AAnn_lux_Time_Series.png

Project: Time series with Rangeslider

[Project-DIY] AAnn_lux_Rangelslider.html

AAnn_lux_Rangelslider.png



[Practice]

- [wk07]
  - charts by plotly
  - Complete your project
  - Upload folder: aax-nn-rpt07
  - Use repo “aax-nn” in github

wk07 : Practice : AAnn_Rpt07

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

제출폴더명 : **aax-nn-rpt07**

- 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

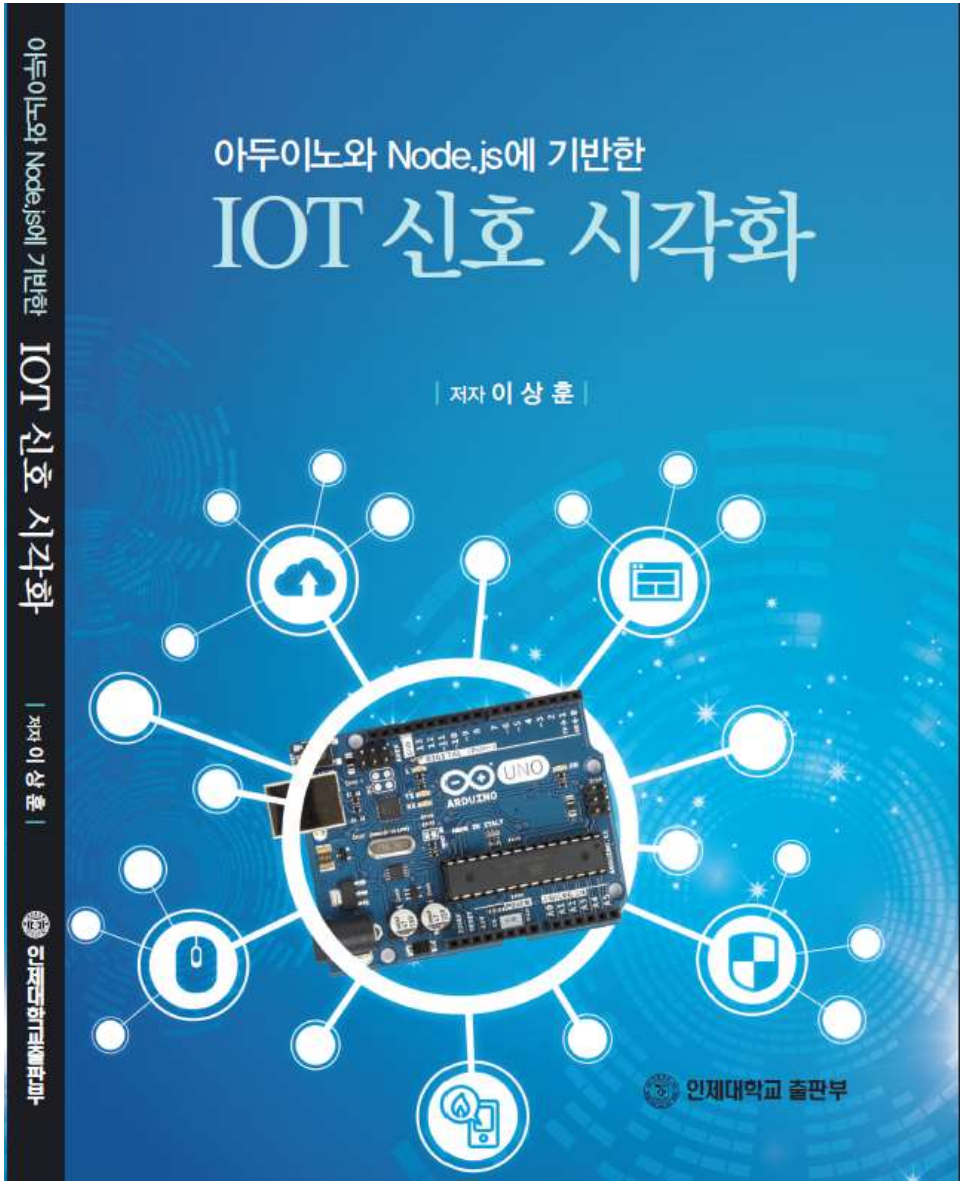
⑥ All *.html

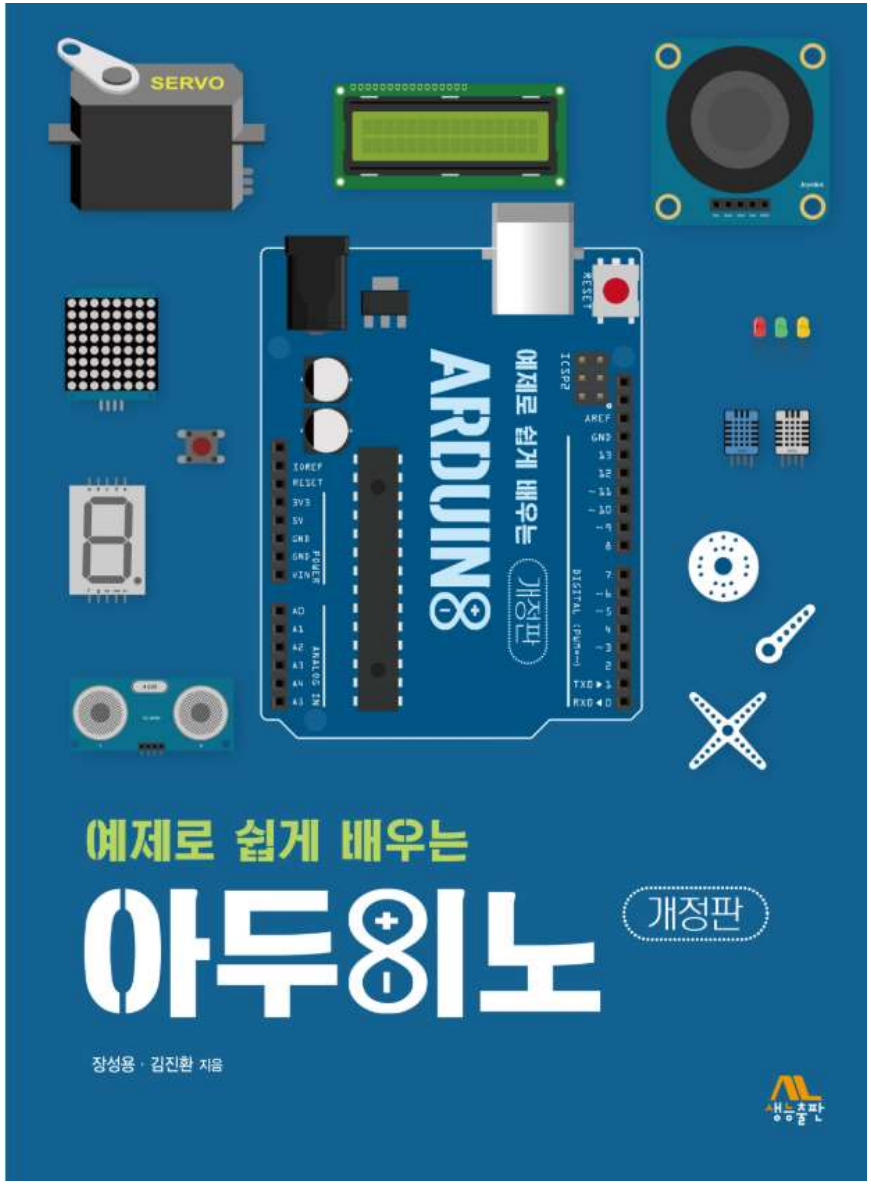
Lecture materials

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 |
인제대학교 출판부


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

Target of this class

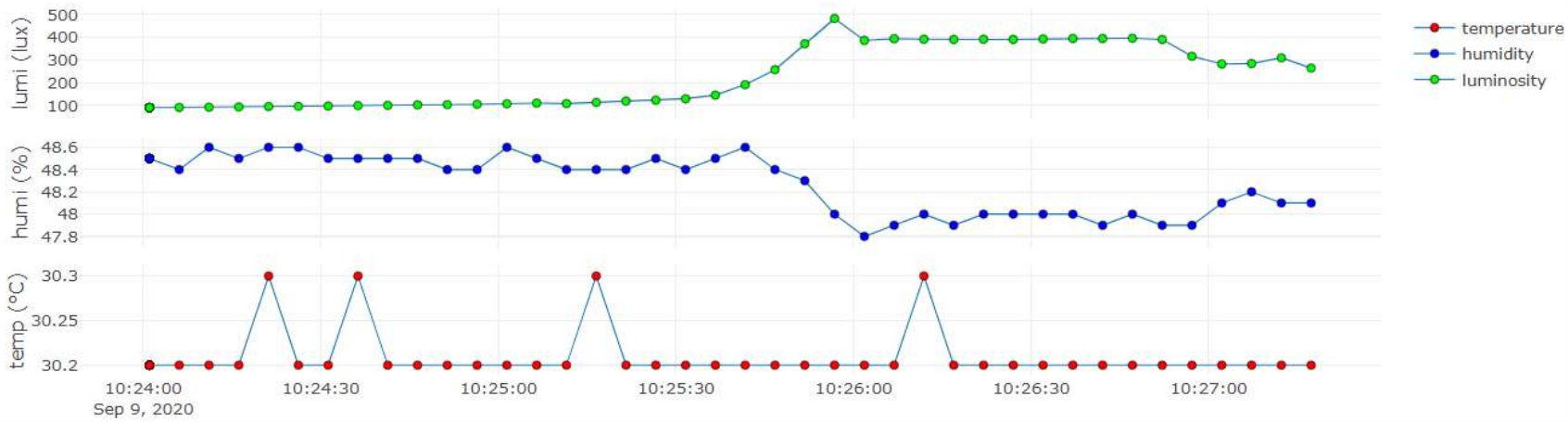
Real-time Weather Station from sensors

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