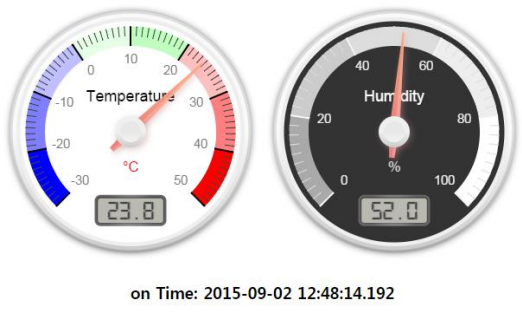
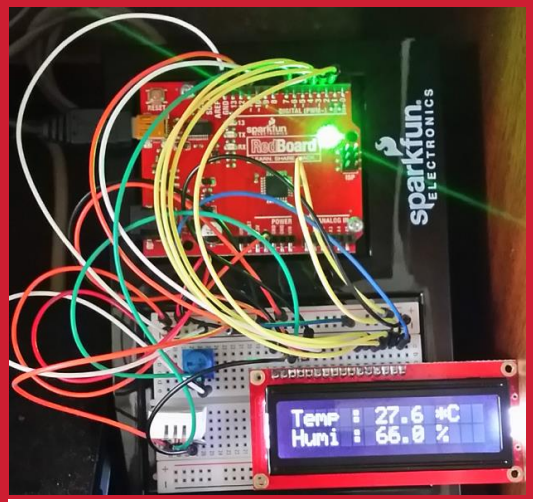
HW-SW-Connectivity

[wk13]

Arduino & NodeJS IV



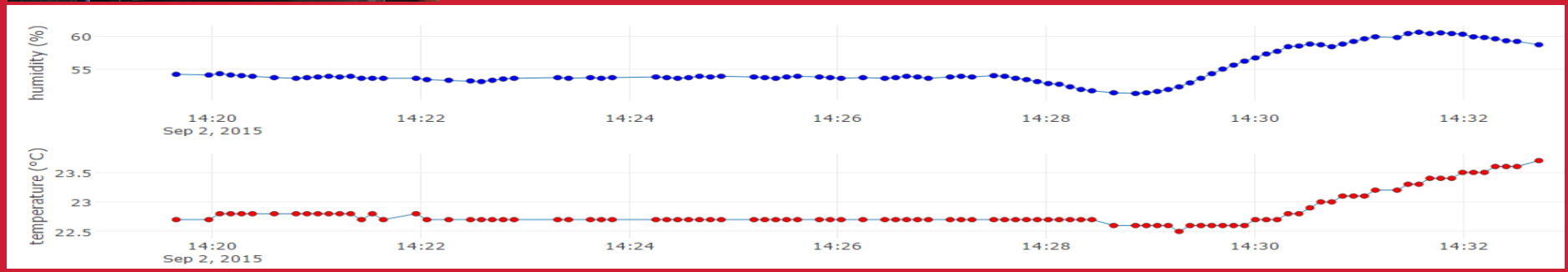


Basic HW and SW Integration using Arduino & Javascript

COMSI, INJE University

2nd semester, 2017

Email : yish@inje.ac.kr



[Practice]

- [wk12]
  - Charts by plotly
  - Complete your plotly chart project
  - Upload file name : AAnn_Rpt09.zip

wk12 : Practice-09 : AAnn_Rpt09.zip

- [Target of this week]
  - Complete your charts
  - Save your outcomes and compress 5 figures & one html.

제출파일명 : **AAnn_Rpt09.zip**

- 압축할 파일들

① AAnn_Chart_Layout.png

② AAnn_Plot_Style.png

③ AAnn_Axis_Title.png

④ AAnn_Line_Dash.png

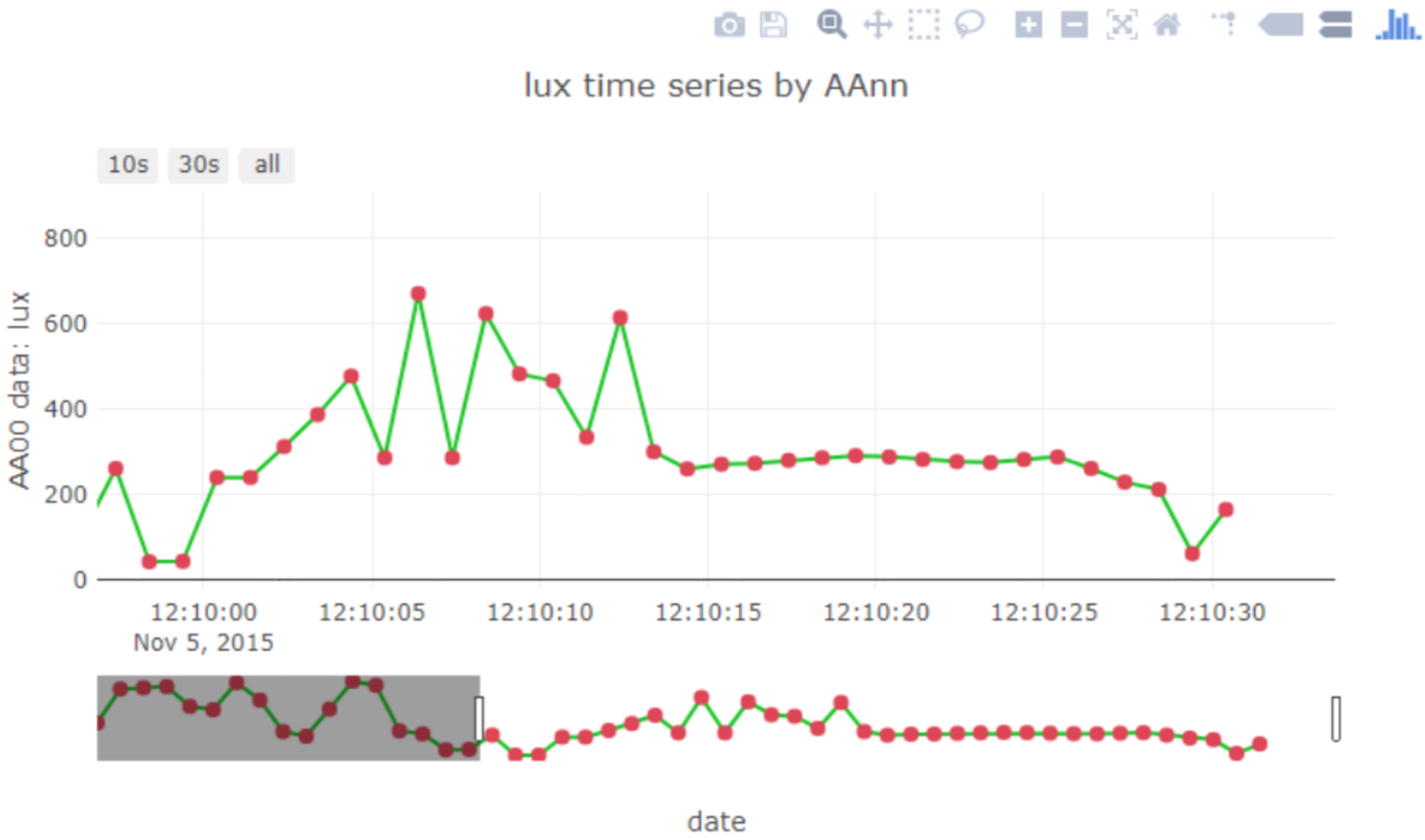
⑤ AAnn_lux_Time_Series.png

⑥ AAnn_lux_Rangeslider.html

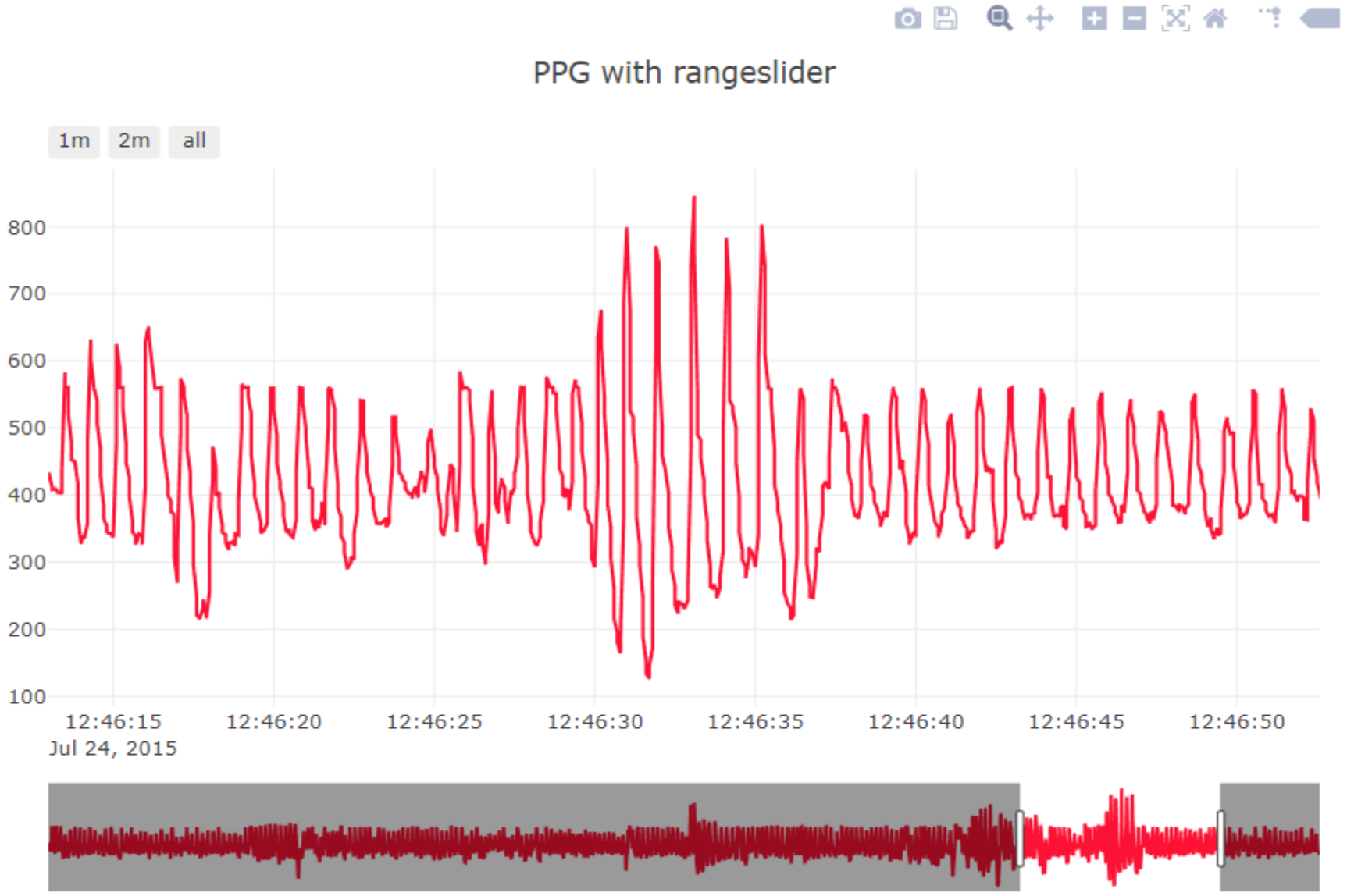
Email : chaos21c@gmail.com

Data visualization : plotly.js

Time series by AAnn



Time series : PPG



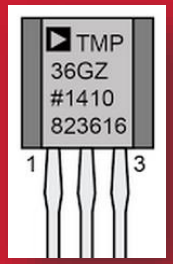
Arduino

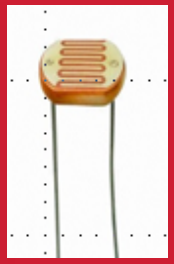
ARDUINO Genuino

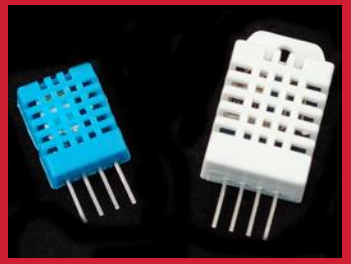
Home | Buy | Download | Products | Learning | Forum | Support | Blog

https://www.arduino.cc/

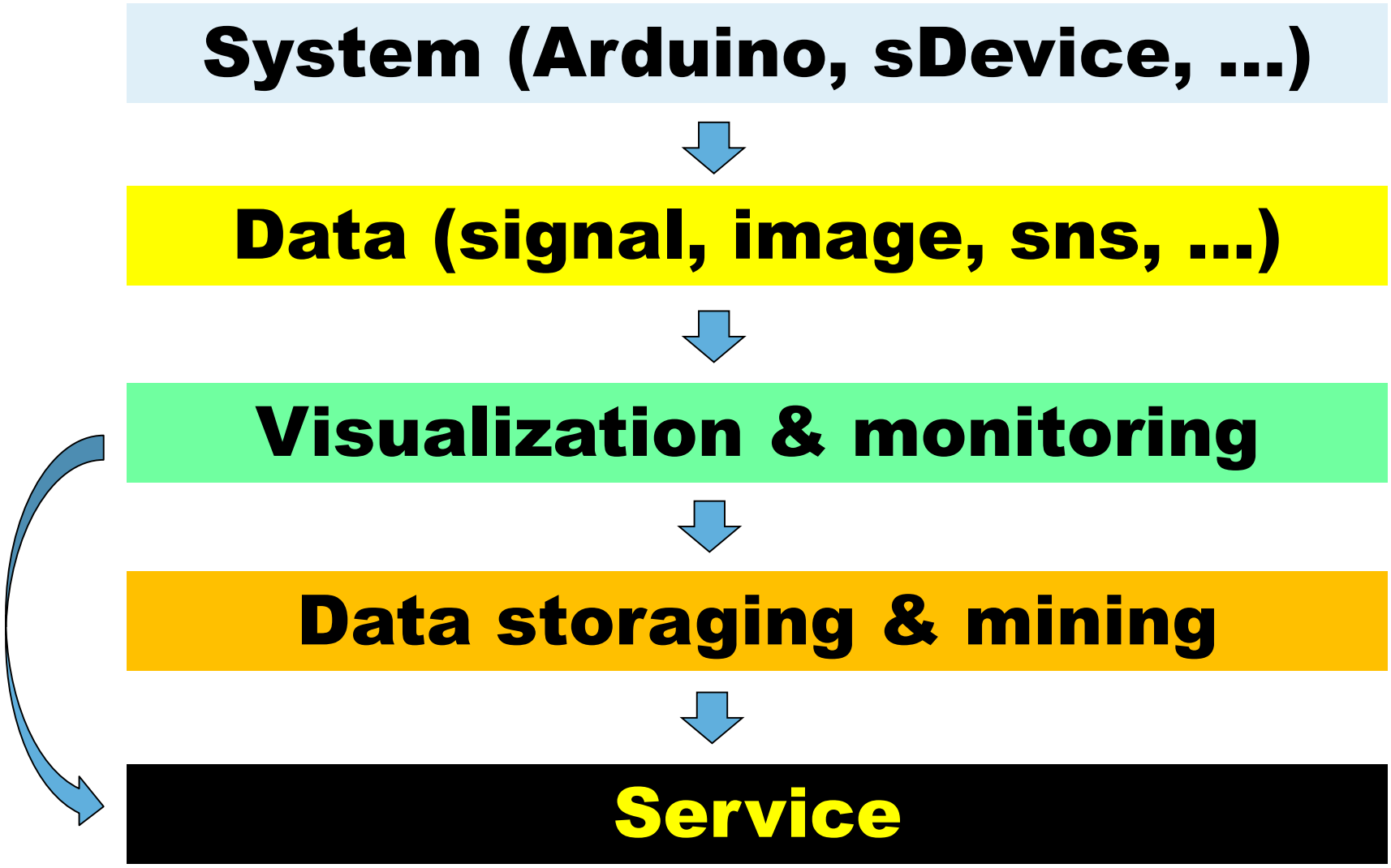
Arduino & Node.js



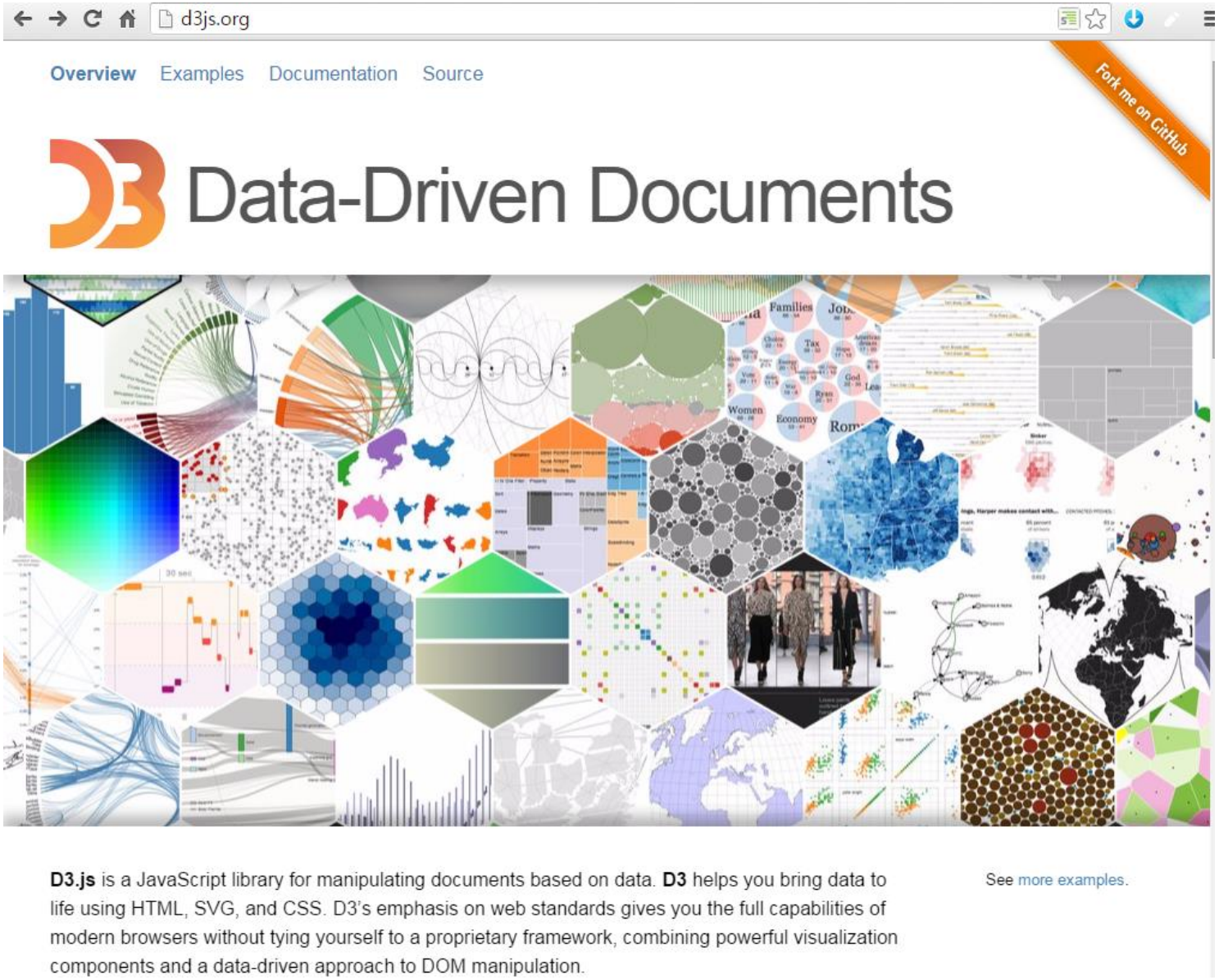




A6.1. Introduction to visualization



A6.1.1 Introduction to visualization



http://d3js.org/

A6.1.1 Introduction to visualization

https://plot.ly/

A6.1.8 Introduction to plot.ly

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

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

A6.1.10 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.

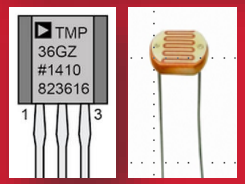
```
<head>
    <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
</head>
```

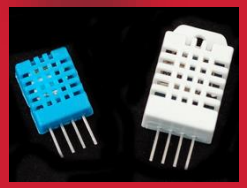
Else, if you want to get a specific version of plotly.js, say 1.2.0:

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

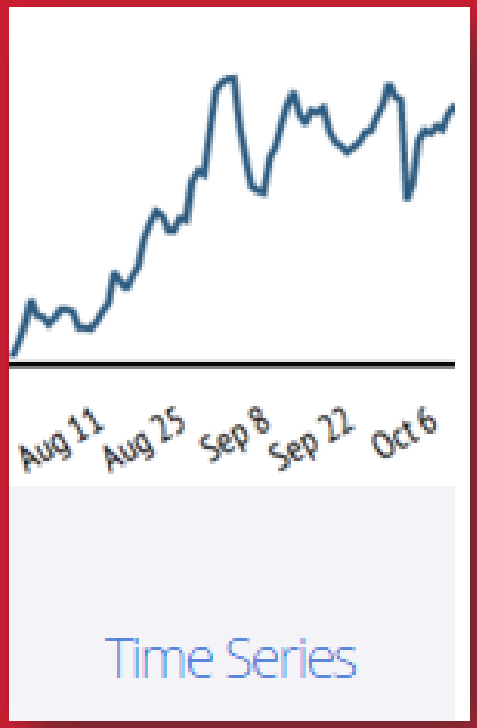
<script src="https://cdn.plot.ly/plotly-latest.min.js"></script>

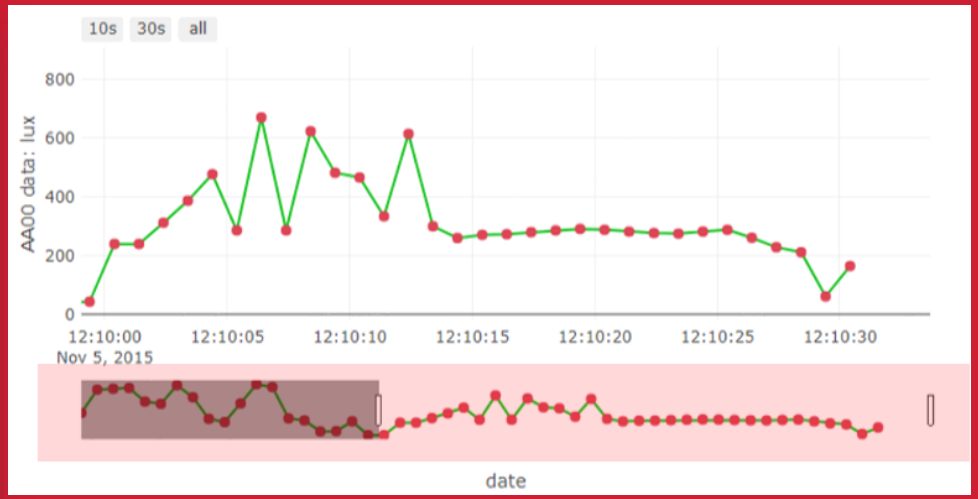
https://plot.ly/javascript/getting-started/





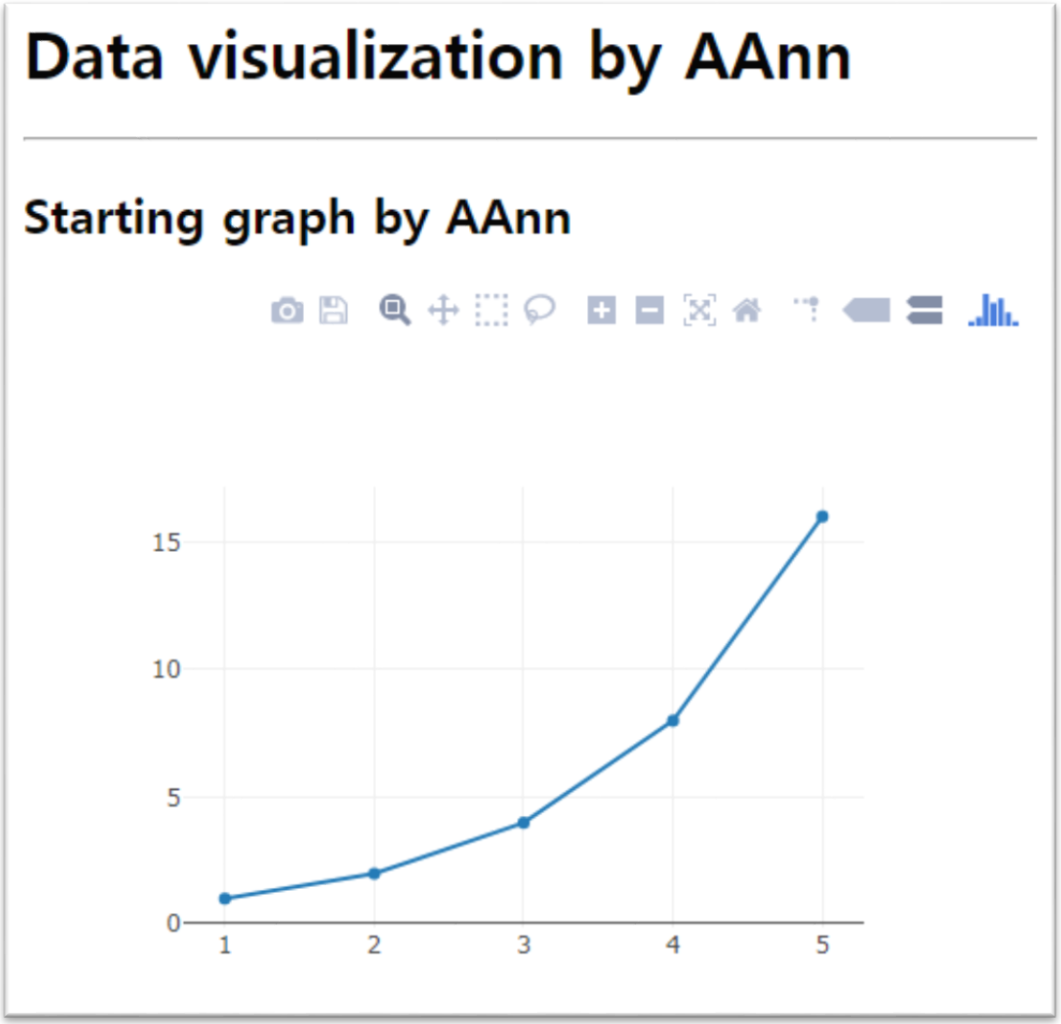
Data visualization using ploy.ly





A6.2.2 Getting started: plotly.js

Graph : Hello plotly data!

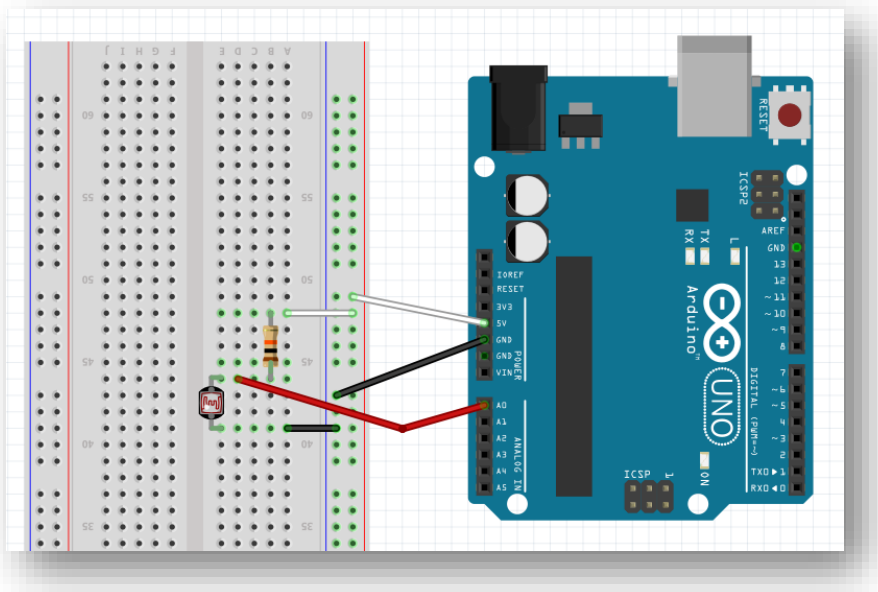


A6.3.9 plotly.js: Time series

[3] Time series : my lux data

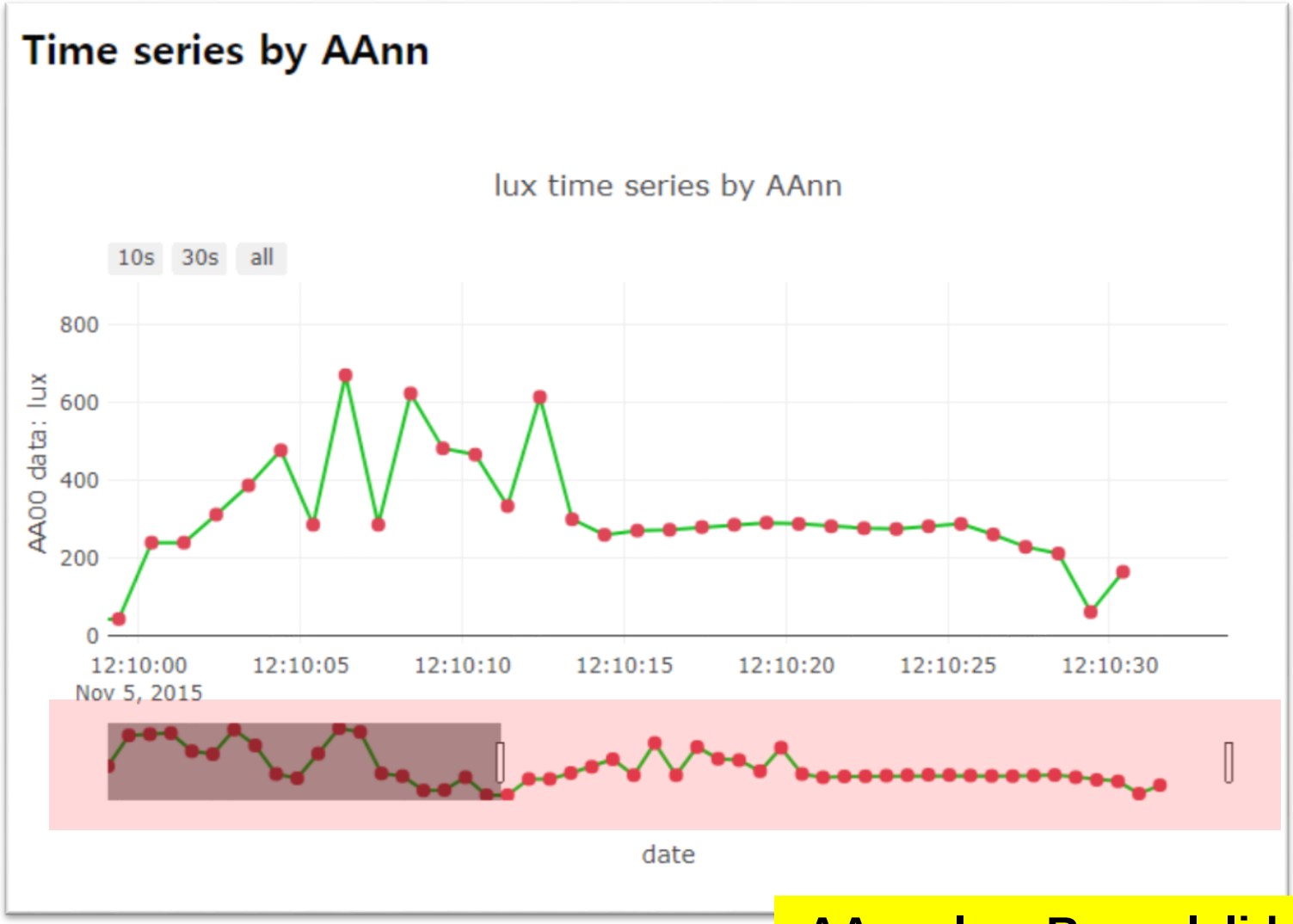
```
'2015-11-05 12:09:41.382',
'2015-11-05 12:09:42.380',
'2015-11-05 12:09:43.378',
'2015-11-05 12:09:44.377',
'2015-11-05 12:09:45.375',
'2015-11-05 12:09:46.389',
'2015-11-05 12:09:47.388',
'2015-11-05 12:09:48.386',
'2015-11-05 12:09:49.384',
'2015-11-05 12:09:50.383',
'2015-11-05 12:09:51.381',
'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



Project: Time series with Rangeslider

[Project-DIY] AAnn_lux_Rangelslider.html

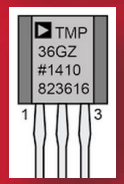


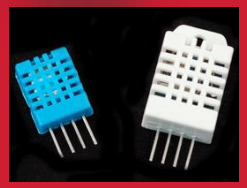
AAnn_lux_Rangelslider.html

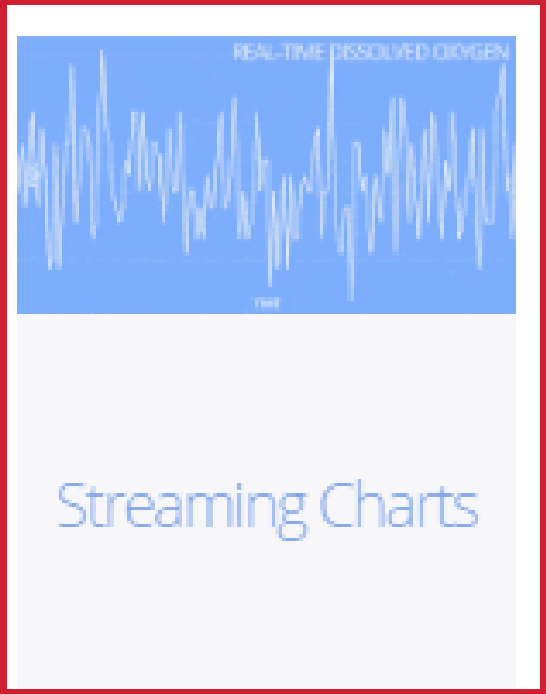
Project: Time series with Rangeslider

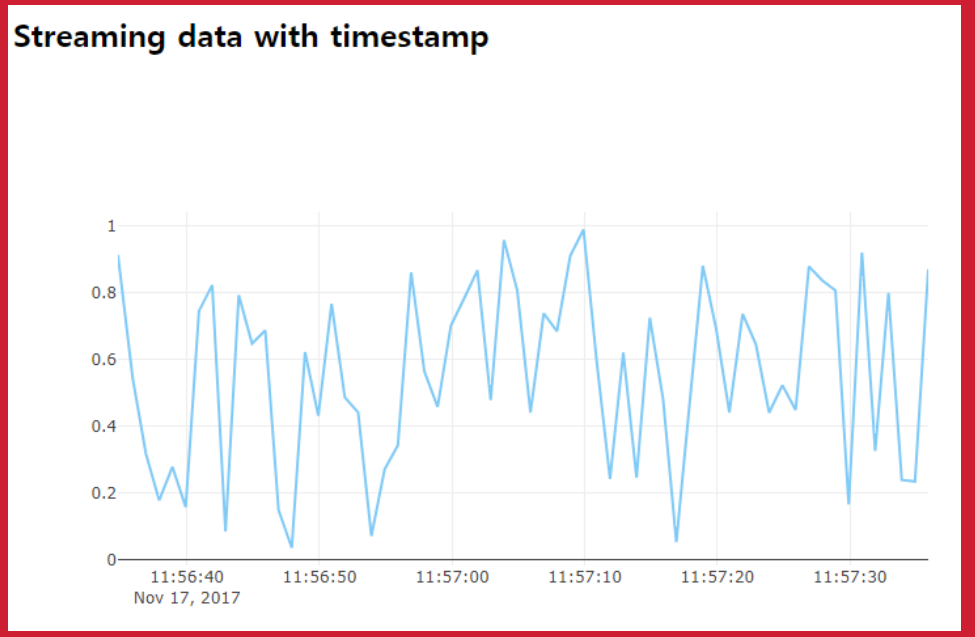
```
var layout = {
  title:'lux time series by AAnn',
  width: 750, height: 500,
  margin: {
    l: 50,
    r: 50,
    b: 100,
    t: 100,
    pad: 4
  },
  xaxis: {
    title: 'date',
    autorange: true,
  range: ['2015-11-05 12:09:40.383', '2015-11-05 12:10:30.413'],
  rangeselector: {buttons: [
      {
        count: 10,
        label: '10s',
        step: 'second',
        stepmode: 'backward'
      },
      {
        count: 30,
        label: '30s',
        step: 'second',
        stepmode: 'backward'
      },
      {step: 'all'}
    ]},
  rangeslider: {range: ['2015-11-05 12:09:40.383', '2015-11-05 12:10:30.413']},
  type: 'date'
  },
  yaxis: {
    title: 'AA00 data: lux'
  }
};
```

Data Streaming using ploy.ly



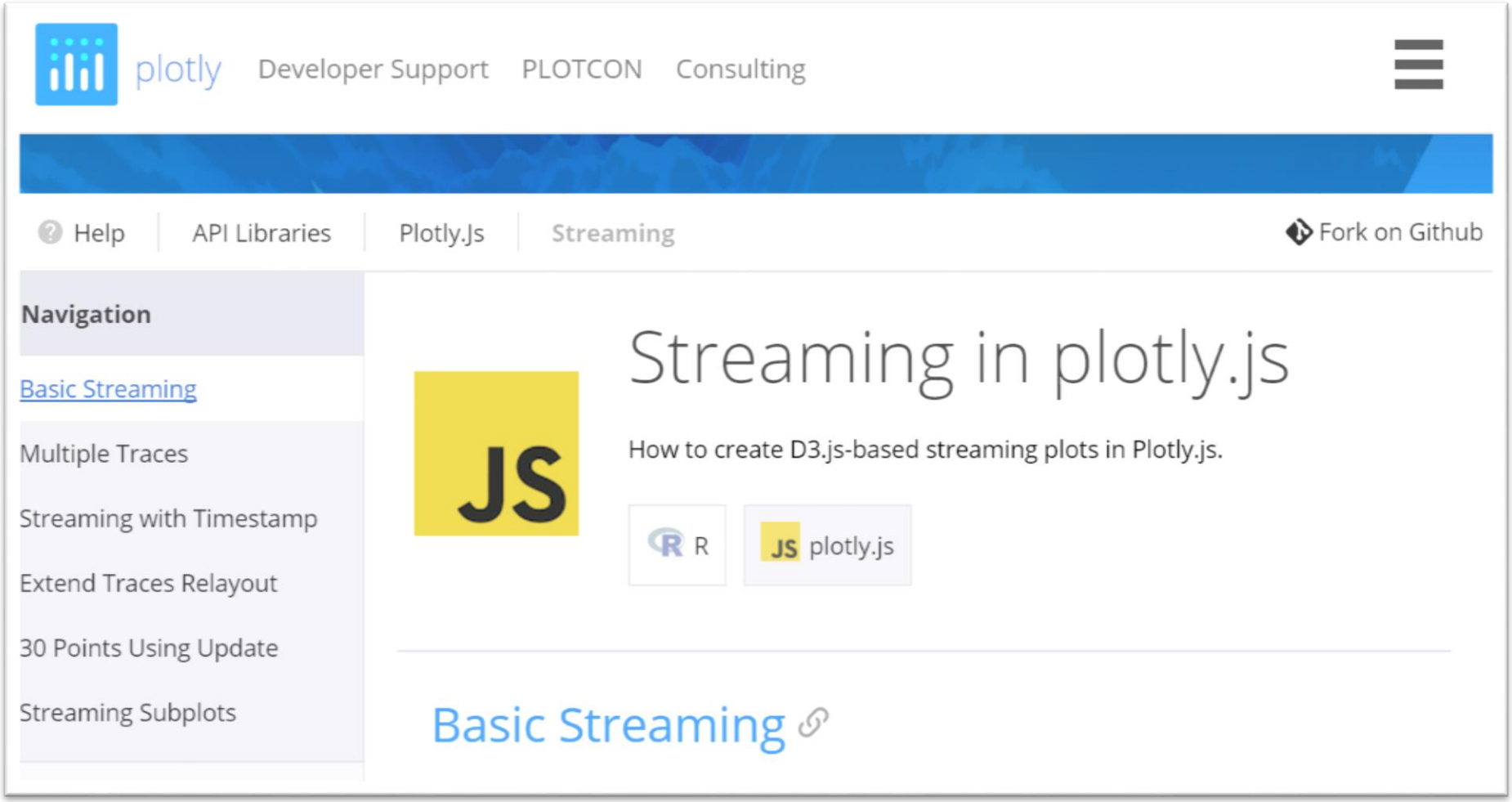






A6.4 plotly.js: Streaming data

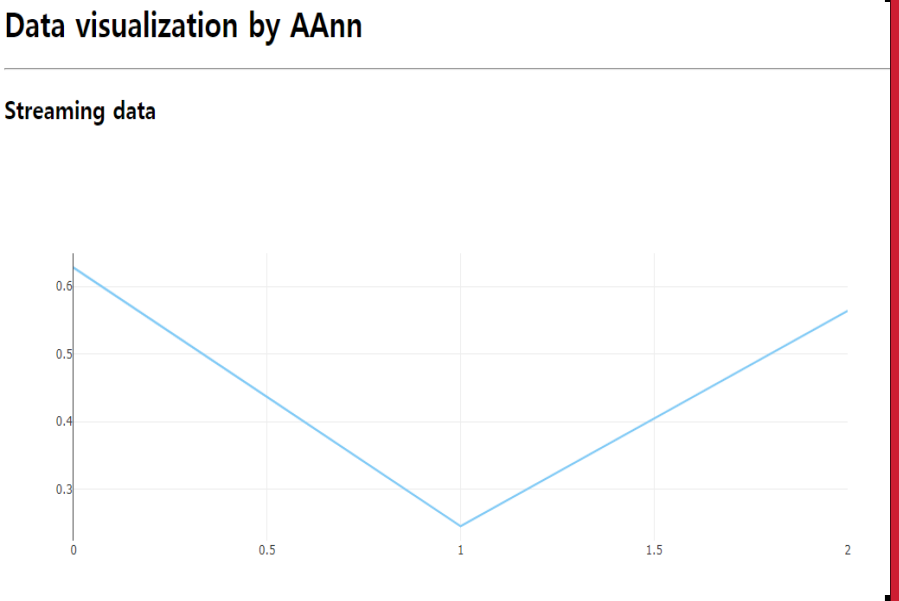
Plot.ly > Streaming



A6.4.1 plotly.js: Streaming data

[1] Basic streaming

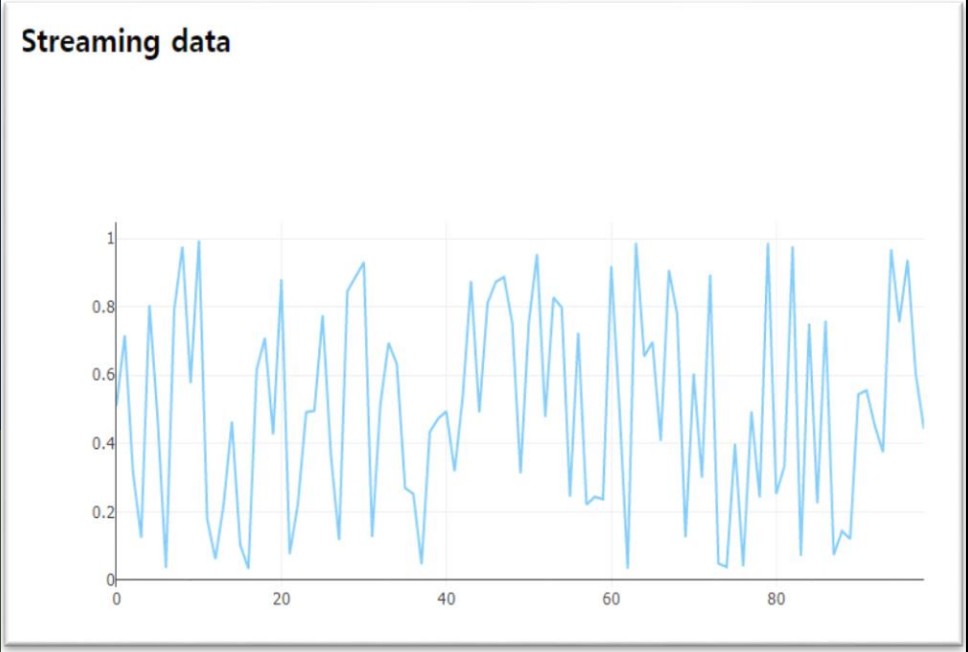
```
<h2>Streaming data</h2>
<div id="graph"></div>

<script>
    function rand() {
        return Math.random(); // 0.0 ~ 1.0
    }

    trace = {
        y: [1,2,3].map(rand),
        mode: 'lines',
        line: {color: '#80CAF6'}
    };

    data = [trace];

    Plotly.plot('graph', data);

    // var cnt = 0;

    // var interval = setInterval(function() {

    //   cnt++;
    //   Plotly.extendTraces('graph', {
    //       y: [[rand()]]
    //   }, [0]);

    //   if(cnt == 50) clearInterval(interval);
    // }, 1000);
</script>
```



A6.4.2 plotly.js: Streaming data

[1] Basic streaming

```
<div id="graph"></div>

<script>
    function rand() {
        return Math.random(); // 0.0 ~ 1.0
    }

    trace = {
        y: [1,2,3].map(rand),
        mode: 'lines',
        line: {color: '#80CAF6'}
    };

    data = [trace];

    Plotly.plot('graph', data);

    var cnt = 0;

    var interval = setInterval(function() {

        cnt++;
        Plotly.extendTraces('graph', {
            y: [[rand()]]
        }, [0]);

        if(cnt == 50) clearInterval(interval);
    }, 1000);
</script>
```

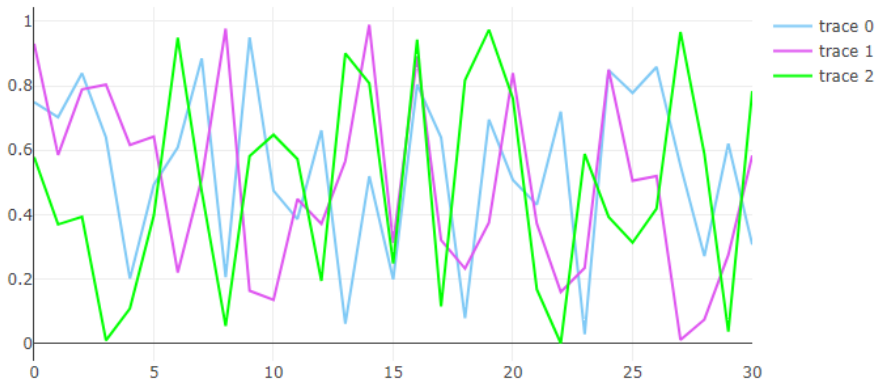


A6.4.3 plotly.js: Streaming data

[2] Streaming multiple traces

```
function rand() {
    return Math.random();
}

// initial plot
trace1 = {
    y: [1,2,3].map(rand),
    mode: 'lines',
    line: {color: '#80CAF6'}
};
trace2 = {
    y: [1,2,3].map(rand),
    mode: 'lines',
    line: {color: '#DF56F1'}
};
trace3 = {
    y: [1,2,3].map(rand),
    mode: 'lines',
    line: {color: '#00FF00'}
};

data = [trace1, trace2, trace3];

Plotly.plot('graph', data);
```

```
// continous plot
var cnt = 0;
var interval = setInterval(function() {

    Plotly.extendTraces('graph', {
        y: [[rand()], [rand()], [rand()]]
    }, [0, 1, 2])

    cnt++;
    if(cnt === 100) clearInterval(interval);
}, 300);
```

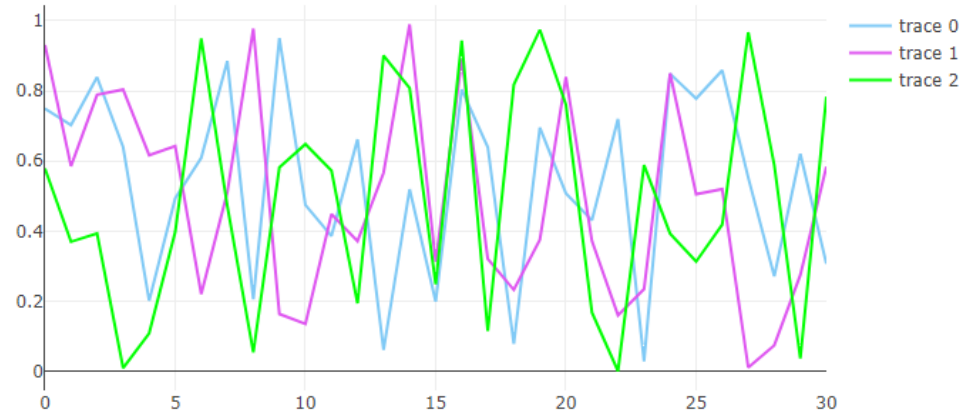


A6.4.4 plotly.js: Streaming data

[2] Streaming multiple traces

```
function rand() {
    return Math.random();
}

// initial plot
Plotly.plot('graph', [{
    y: [1,2,3].map(rand),
    mode: 'lines',
    line: {color: '#80CAF6'}
}, {
    y: [1,2,3].map(rand),
    mode: 'lines',
    line: {color: '#DF56F1'}
}, {
    y: [1,2,3].map(rand),
    mode: 'lines',
    line: {color: '#00FF00'}
}]);
```

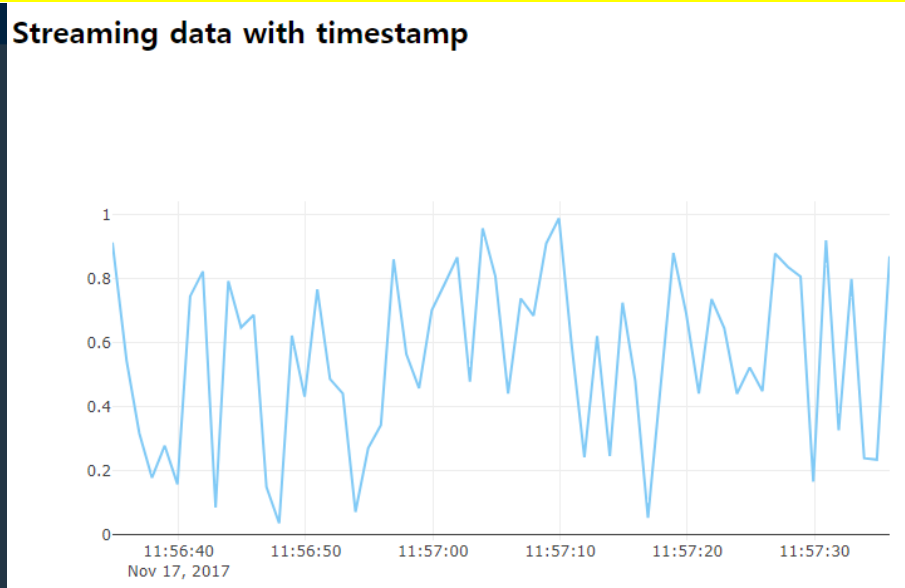
```
// continous plot
var cnt = 0;
var interval = setInterval(function() {

    Plotly.extendTraces('graph', {
        y: [[rand()], [rand()], [rand()]]
    }, [0, 1, 2])

    cnt++;
    if(cnt === 100) clearInterval(interval);
}, 300);
```



A6.4.5 plotly.js: Streaming data

[3] Streaming data with timestamp

```
<script>
    function rand() {
        return Math.random();
    }

    var time = new Date();
    var data = [{
        x: [time],
        y: [rand()],
        mode: 'lines',
        line: {color: '#80CAF6'}
    }]

    Plotly.plot('graph', data);

    var cnt = 0;
    var interval = setInterval(function() {

        var time = new Date();
        var update = {
            x:  [[time]],
            y: [[rand()]]
        }

        Plotly.extendTraces('graph', update, [0])
        // cnt++;
        if(cnt === 100) clearInterval(interval);
    }, 1000);

</script>
```

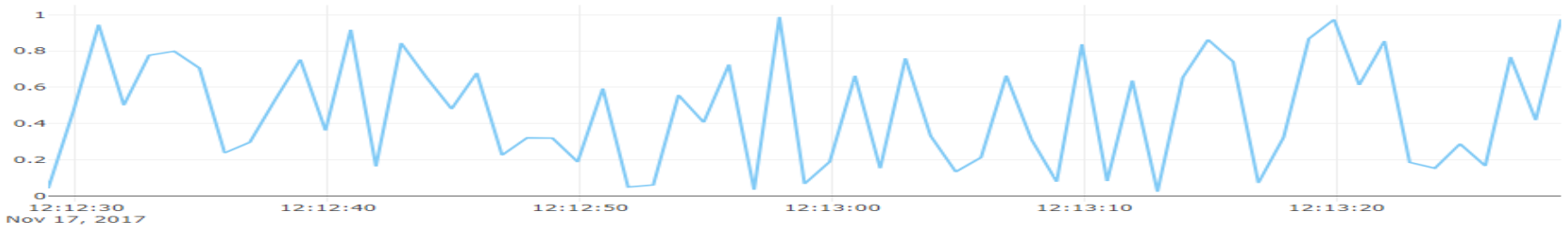


A6.4.6 plotly.js: Streaming data

[4] Streaming data by relayout

```
function rand() {
    return Math.random();
}

var time = new Date();

var data = [{
    x: [time],
    y: [rand],
    mode: 'lines',
    line: {color: '#80CAF6'}
}]

Plotly.plot('graph', data);
```

```
var cnt = 0;
var interval = setInterval(function() {

    var time = new Date();
    var update = {
        x:  [[time]],
        y: [[rand()]]
    }

    var olderTime = time.setMinutes(time.getMinutes() - 1);
    var futureTime = time.setMinutes(time.getMinutes() + 1);

    var minuteView = {
        xaxis: {
            type: 'date',
            range: [olderTime,futureTime]
        }
    };

    Plotly.relayout('graph', minuteView);
    Plotly.extendTraces('graph', update, [0])
    // cnt++;
    if(cnt === 200) clearInterval(interval);
}, 1000);
```



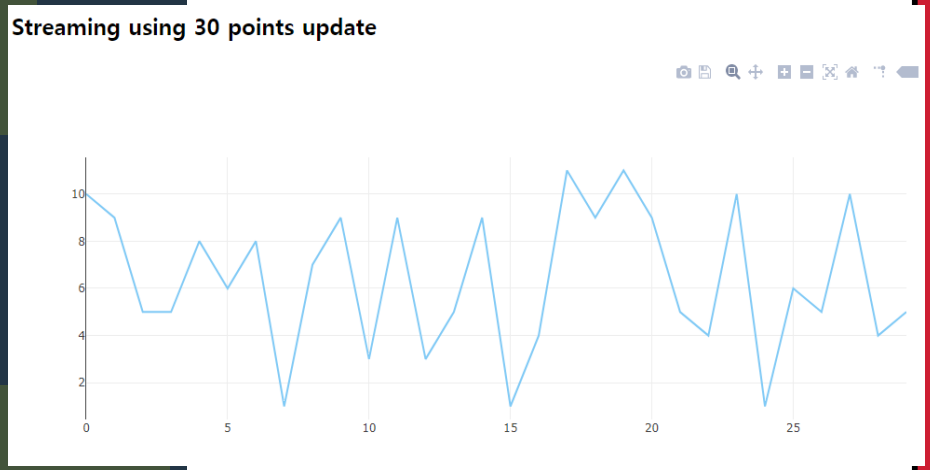
A6.4.7 plotly.js: Streaming data

[5] Streaming data using 30 points update

```
var arrayLength = 30
var newArray = []

for(var i = 0; i < arrayLength; i++) {
    var y = Math.round(Math.random()*10) + 1
    newArray[i] = y
}
```

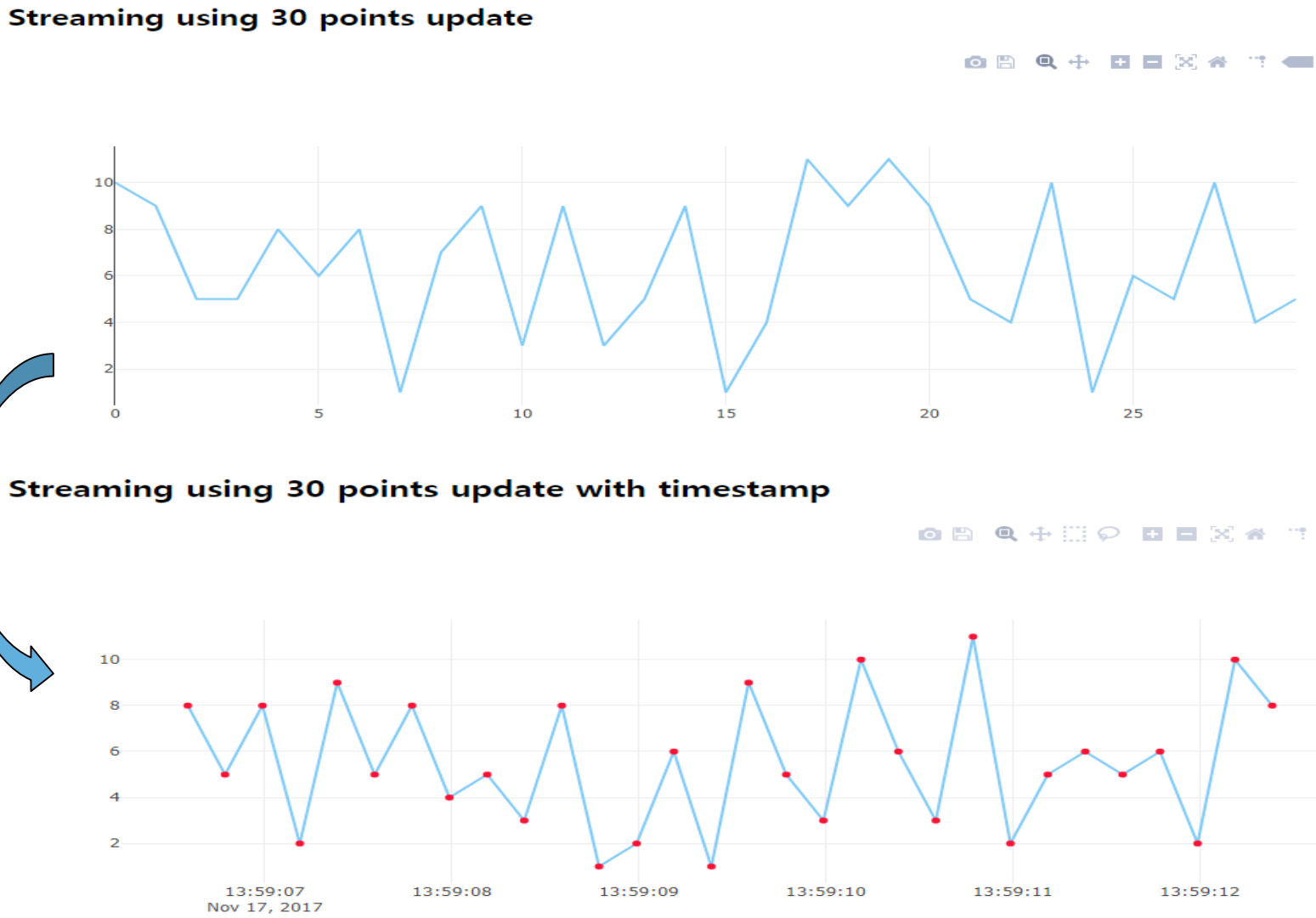
```
Plotly.plot('graph', [{
    y: newArray,
    mode: 'lines',
    line: {color: '#80CAF6'}
}]);
```

```
var cnt = 0;
var interval = setInterval(function() {

    var y = Math.round(Math.random()*10) + 1
    newArray = newArray.concat(y)  // add new data
    newArray.splice(0, 1)  // remove the oldest data

    var data_update = {
        y: [newArray]
    };

    Plotly.update('graph', data_update)
    //cnt++;
    if(cnt === 100) clearInterval(interval);
}, 1000);
```



A6.4.8 plotly.js: Streaming data

[DIY] Streaming time series using 30 points update



AAnn_DS_30timestamps.png 로 캡처 저장.

A6.4.9 plotly.js: Streaming data

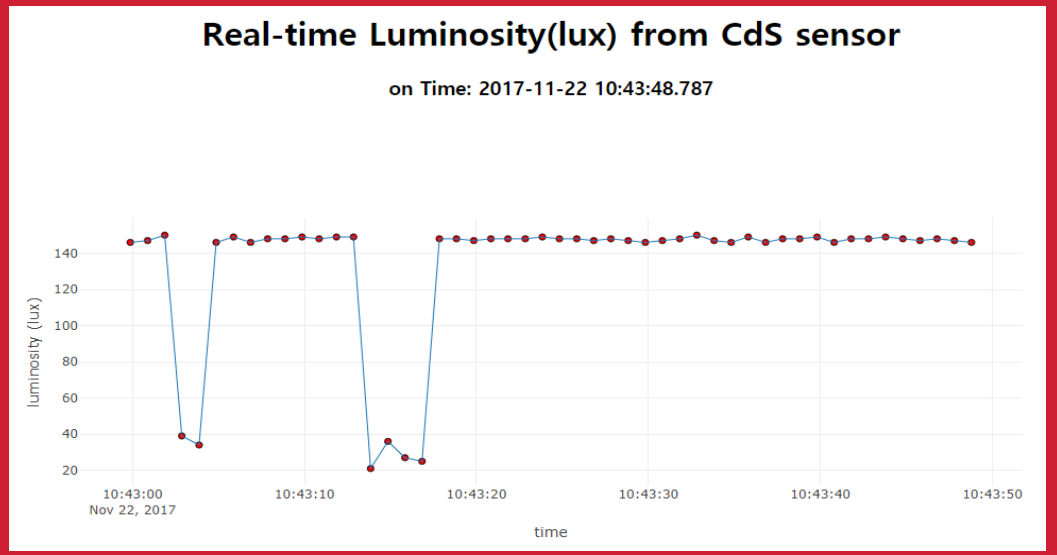
[DIY-hint] Streaming time series using 30 points update

```
<script>
    var arrayLength = 30
    var newArray = []
    var timeArray = []

    // initial 30 data
    for(var i = 0; i < arrayLength; i++) {
        var y = Math.round(Math.random()*10) + 1
        var time = new Date();
        newArray[i] = y
        timeArray[i] = time
    }

    var data = [{
        x: timeArray,
        y: newArray,
        mode: 'lines+markers',
        line: {color: '#80CAF6'},
        marker: {color: '#FC1234'}
    }]


    Plotly.plot('graph', data);
```

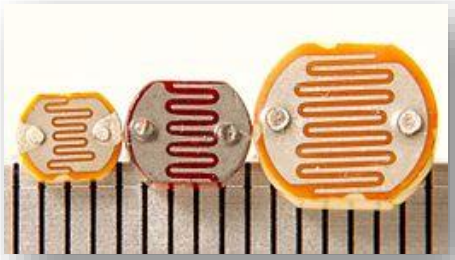
Arduino sensor data

RT visualization using ploy.ly

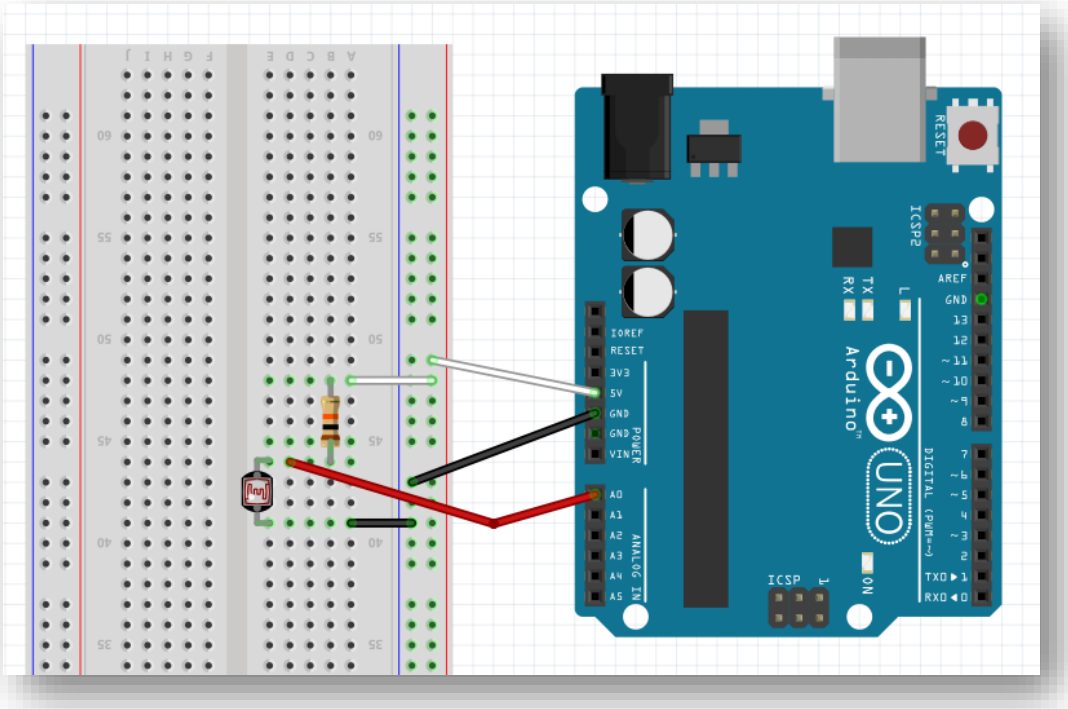
```
AA00,2017-11-22 10:43:11.859,149
AA00,2017-11-22 10:43:12.851,149
AA00,2017-11-22 10:43:13.845,21
AA00,2017-11-22 10:43:14.854,36
AA00,2017-11-22 10:43:15.844,27
AA00,2017-11-22 10:43:16.837,25
AA00,2017-11-22 10:43:17.846,148
AA00,2017-11-22 10:43:18.839,148
AA00,2017-11-22 10:43:19.847,147
```



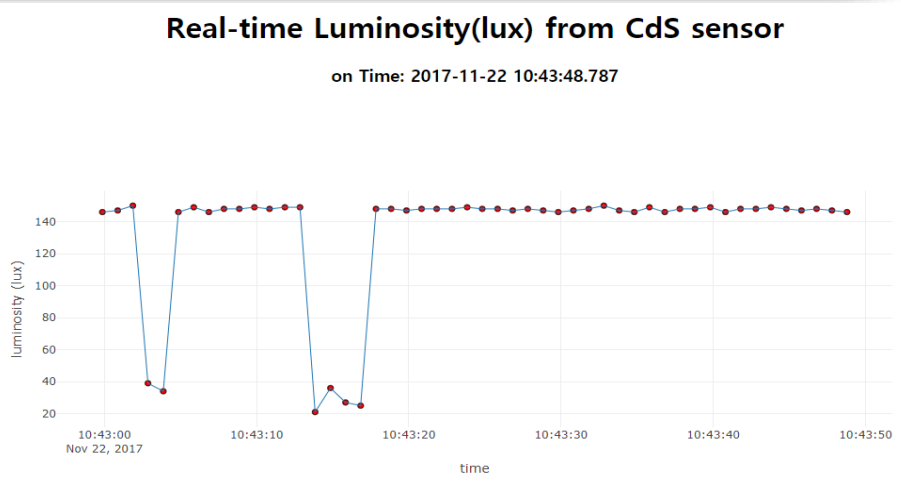
Luminosity sensor [Photocell LDR]



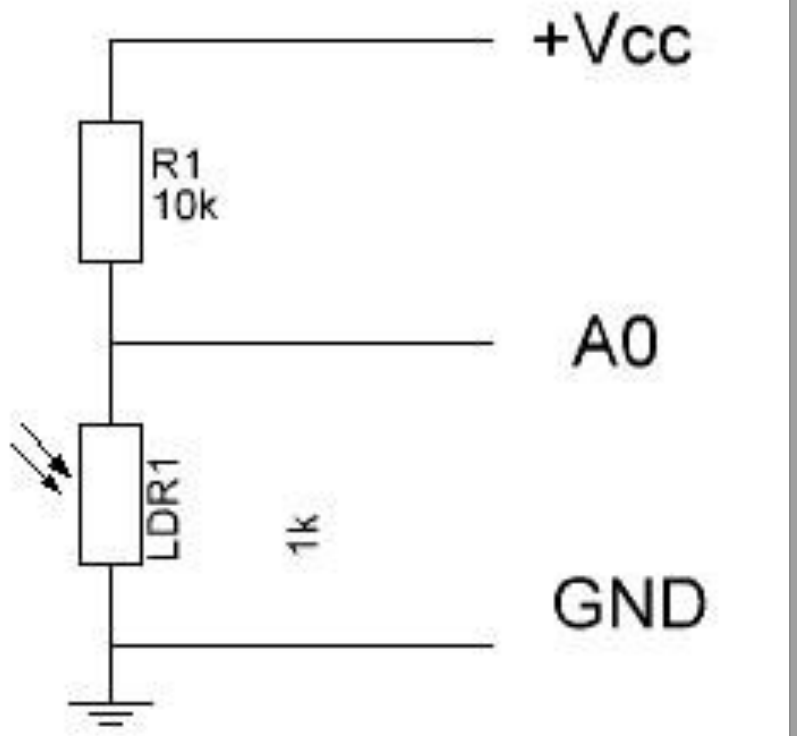
CdS




```
AA00,2017-11-22 10:43:11.859,149
AA00,2017-11-22 10:43:12.851,149
AA00,2017-11-22 10:43:13.845,21
AA00,2017-11-22 10:43:14.854,36
AA00,2017-11-22 10:43:15.844,27
AA00,2017-11-22 10:43:16.837,25
AA00,2017-11-22 10:43:17.846,148
AA00,2017-11-22 10:43:18.839,148
AA00,2017-11-22 10:43:19.847,147
```



CdS 센서 회로 분석 (2/3)



Ao → Vo → lux

$lux = 500 / R_{ldr}$

$V_o = I_{ldr} * R_{ldr}$

$= (5/(10 + R_{ldr}))* R_{ldr}$

$R_{ldr} = 10*V_o / (5 - V_o)$

$lux = 250/V_o - 50$

$V_o = 5.0 * Ao / 1023.0$

A3.2.5 Luminosity sensor [Photocell LDR]

CdS 센서 회로 - 측정 2.

AA00_CdS

```
1  //  lux
2  #define CDS_INPUT 0
3
4  void setup() {
5  Serial.begin(9600);
6  }
7  void loop() {
8    int value = analogRead(CDS_INPUT);
9    Serial.println(int(luminosity(value)));
10   delay(1000);
11 }
12
13 //Voltage to LuxLux
14 double luminosity (int RawADC0){
15   double Vout=RawADC0*0.0048828125;  // 5/1024 (Vin = 5 V)
16   int lux=(2500/Vout-500)/10;  // lux = 500 / Rldr, Vout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
17   return lux;
18 }
```

COM4

```
1
1
2
1
1
0
72
166
167
168
167
167
166
166
```

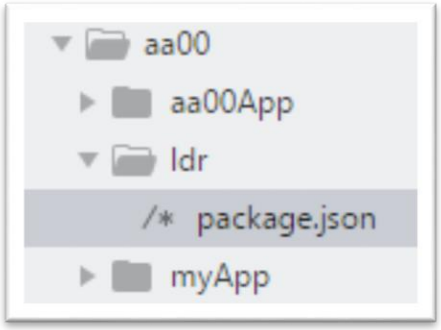
어두울 때

밝을 때

밝을수록 측정 값이 커지고 어두을수록 값이 작아진다 !!!

A5.2.1 Luminosity sensor [Photocell LDR]

1. Make ldr node project
   - md ldr
   - cd ldr
2. Go to ldr subfolder
   - npm init

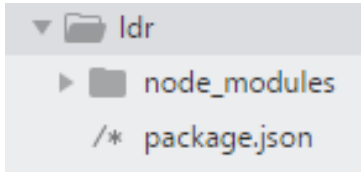


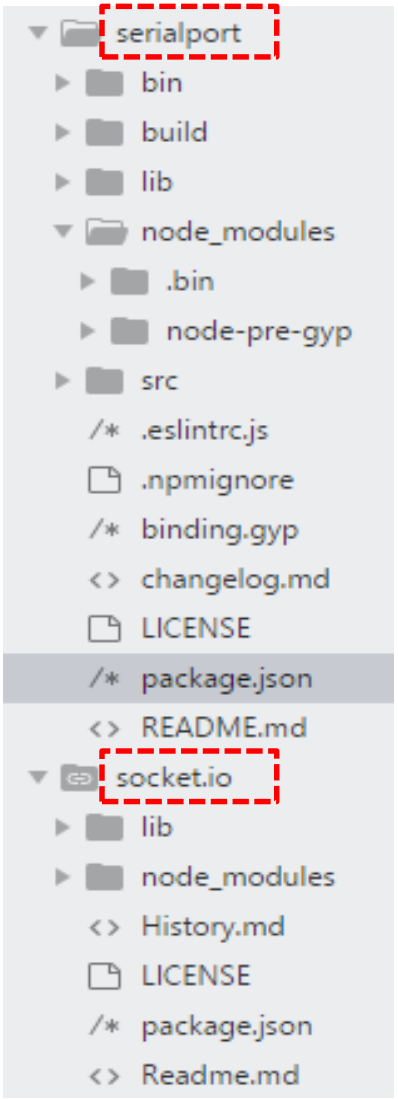

```
{
  "name": "ldr",
  "version": "1.0.0",
  "description": "",
  "main": "ldr_node.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "cds",
    "ldr",
    "node",
    "arduino"
  ],
  "author": "aa00",
  "license": "MIT"
}
```

"main": "ldr_node.js“
"author": "aann"

A5.2.2 Luminosity sensor [Photocell LDR]

1. Make ldr node project
   - md ldr
   - cd ldr
2. Go to ldr subfolder
   - npm init
   - **npm install –save serialport@4.0.7**
   - **npm install –save socket.io@1.7.3**





You can check version of each module by browing package.json in each module subfolder.

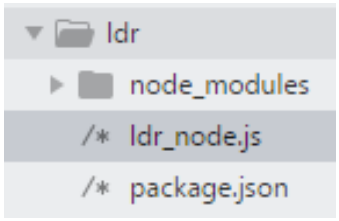
A5.2.3 Luminosity sensor [Photocell LDR]

1. **Make ldr node project**
   - **md ldr**
   - **cd ldr**
2. **Go to ldr subfolder**
   - **npm init**
   - **npm install –save serialport@4.0.7**
   - **npm install –save socket.io@1.7.3**

package,json

```
{
  "name": "ldr",
  "version": "1.0.0",
  "description": "",
  "main": "ldr_node.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "cds",
    "ldr",
    "node",
    "arduino"
  ],
  "author": "aa00",
  "license": "MIT",
  "dependencies": {
    "serialport": "^4.0.7",
    "socket.io": "^1.7.3"
  }
}
```

A5.2.4 Luminosity sensor [Photocell LDR]

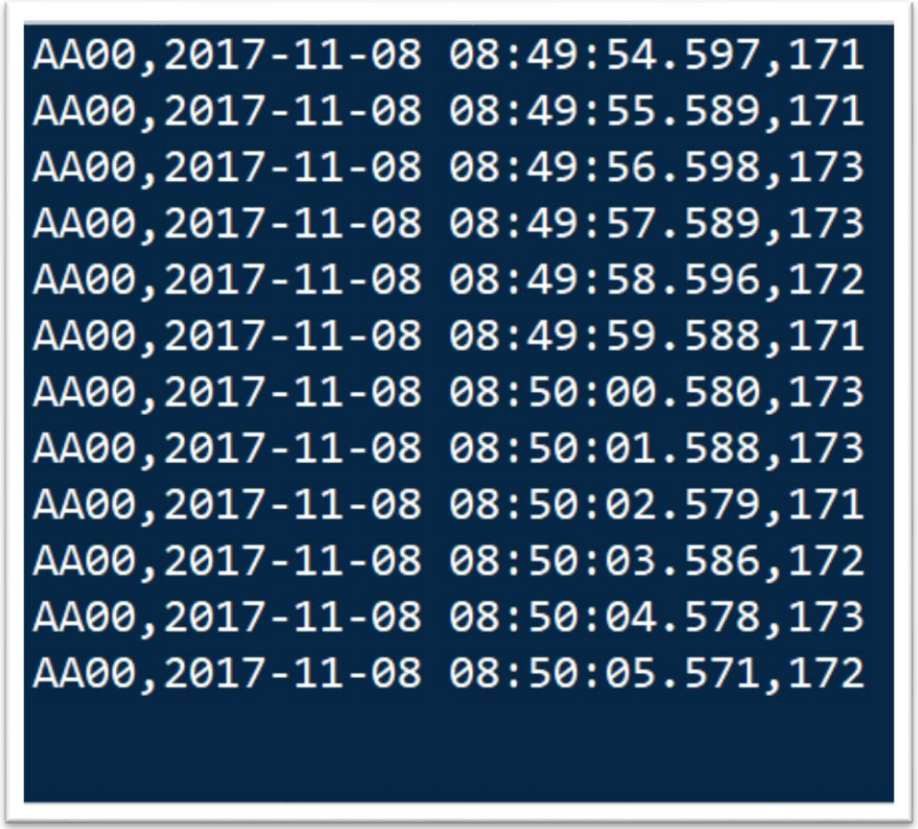


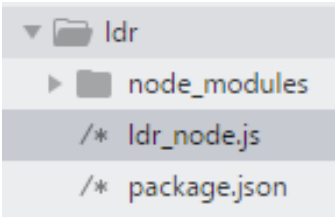
Save tmp36_node.js as ldr_node.js

```
var dStr = '';
var tdata = [];

sp.on('data', function (data) { // call back when data is received
    // raw data only
        //console.log(data);
        dStr = getDateString();
        tdata[0] = dStr;  // date
        tdata[1] = data;  // data
        console.log("AA00," + tdata);
        io.sockets.emit('message', tdata);  // send data to all clients
});

// helper function to get a nicely formatted date string
function getDateString() {
    var time = new Date().getTime();
    // 32400000 is (GMT+9 Korea, GimHae)
    // for your timezone just multiply +/-GMT by 3600000
    var datestr = new Date(time +32400000).
    toISOString().replace(/T/, ' ').replace(/Z/, '');
    return datestr;
}
```

A5.2.5 Luminosity sensor [Photocell LDR]

Run ldr_node.js (^B)

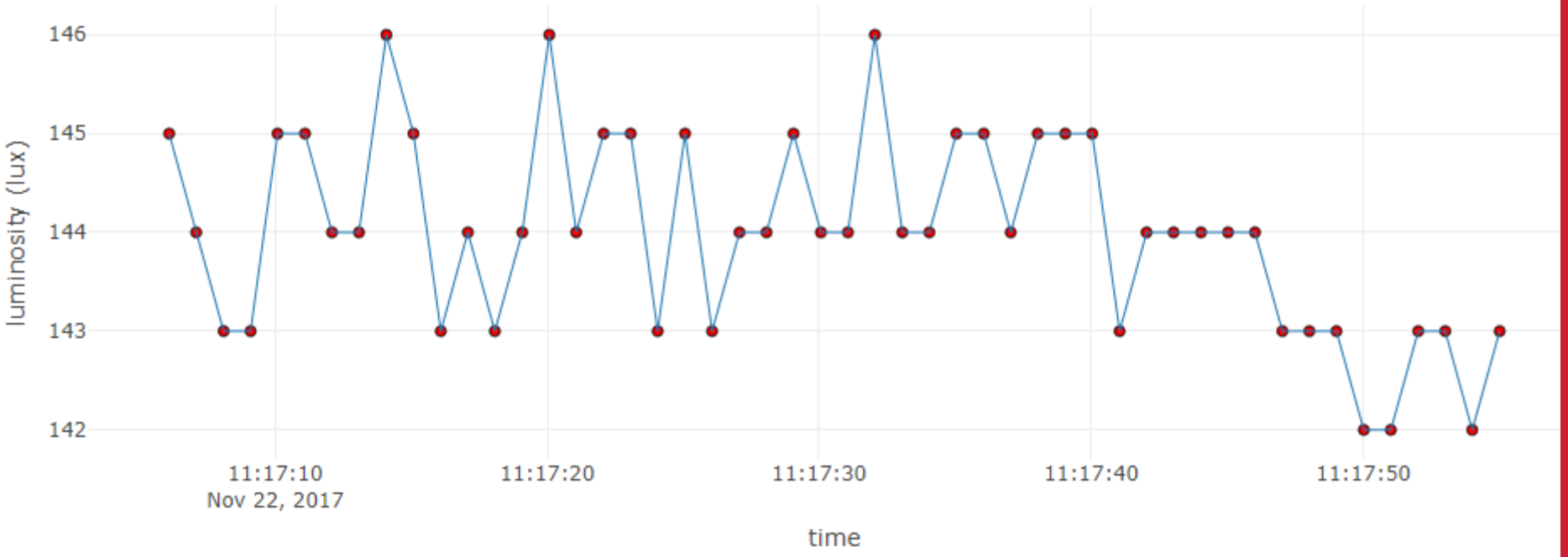





```
io.sockets.emit('message', tdata);  // send data to all clients
```

Real-time Luminosity(lux) from CdS sensor

on Time: 2017-11-22 11:17:55.020



A6.5.1 RT sensor-data streaming in Arduino

[1] Client html : client_ldr.html (using socket.io.js)

```
<!DOCTYPE html>
<head>
  <meta charset="utf-8">
  <title>plotly.js Example: Real time signals from 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/1.3.6/socket.io.js"></script>
  <style>body{padding:0;margin:30;background:#fff}</style>
</head>
```

A6.5.2 RT sensor-data streaming in Arduino

[2] Client html : client_ldr.html (global variables)

```
<body>  <!-- style="width:100%;height:100%"> -->
<!-- Plotly chart will be drawn inside this DIV -->
<h1 align="center"> Real-time Luminosity(lux) from CdS sensor </h1>

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

<div id="myDiv"></div> <!-- graph here! -->

<hr>

  <script>
  /* JAVASCRIPT CODE GOES HERE */
    var streamPlot = document.getElementById('myDiv');
    var ctime = document.getElementById('time');

    var xArray = [], // time of data arrival
        xTrack = [], // value of CdS sensor 1 : lux
        numPts = 50, // number of data points in x-axis
        dtda = [],  // 1 x 2 array : [date, lux] from CdS
        preX = -1,  // check change in data
        initFlag = true;
```

A6.5.3 RT sensor-data streaming in Arduino

[3] Client html : client_ldr.html (socket connection & handling message)

```
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]);  // Lux
            init();  // start streaming
            initFlag=false;
        }
        console.log(msg[0]);
        console.log(parseFloat(msg[1]));
        // Convert value to integer
        dtda[0]=msg[0];
        dtda[1] = parseFloat(msg[1]); // integer

        // when new data is coming, keep on streaming data
        ctime.innerHTML = dtda[0];
        nextPt();
    });
});
```

[4] Client html : client_ldr.html (init() & nextPt())

```
function init() {  // initial screen ()
    // starting point : first data (lux)
    for ( i = 0; i < numPts; i++) {
        xArray.push(dtda[0]);  // date
        xTrack.push(dtda[1]);  // CdS sensor (lux)
    }

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

function nextPt() {

    xArray.shift();
    xArray.push(dtda[0]);

    xTrack.shift();
    xTrack.push(dtda[1]);  // CdS sensor: lux

    Plotly.redraw(streamPlot);
}
```

[5] Client html : client_ldr.html (data & layout)

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

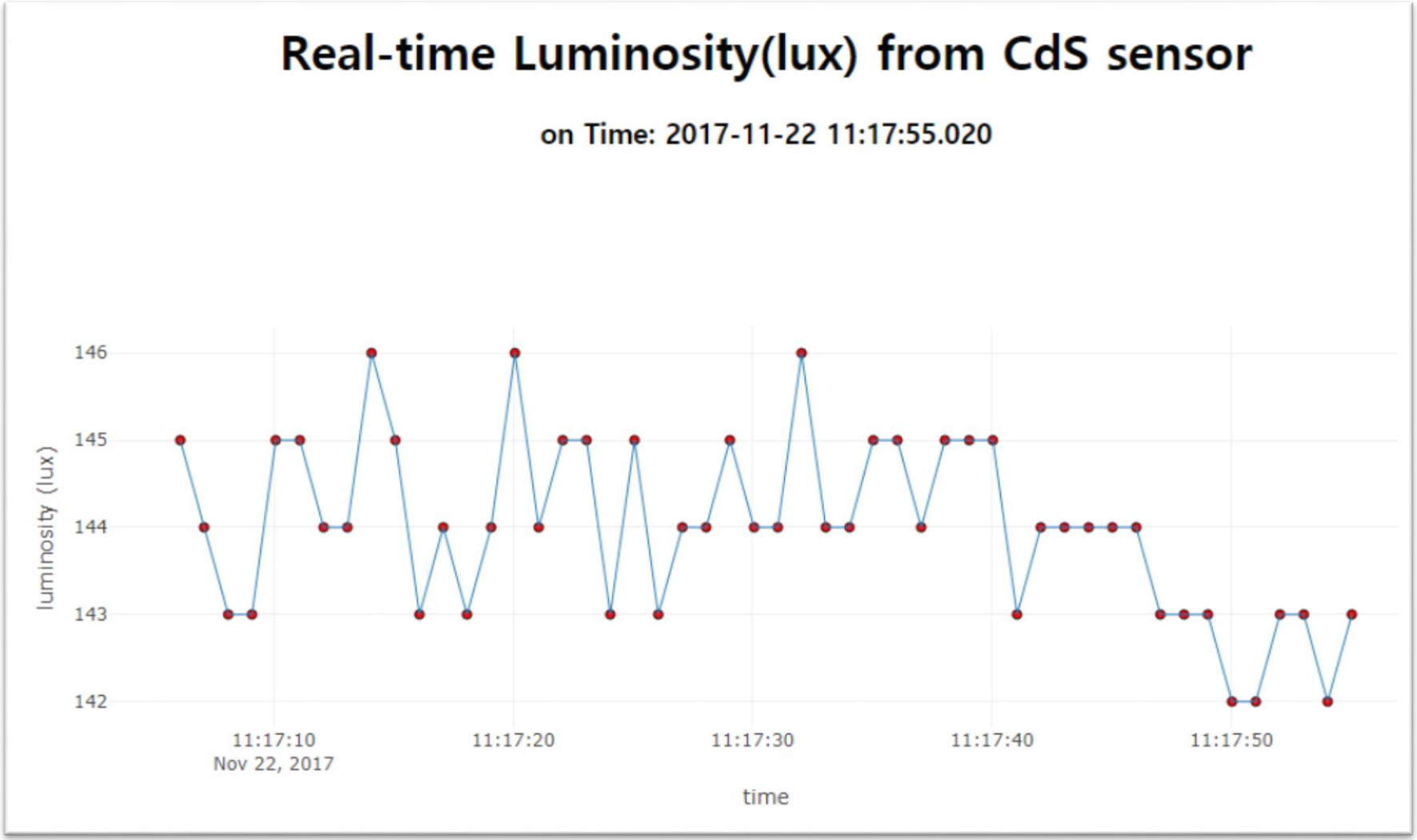
```
// Layout
var layout = {
    xaxis : {
        title : 'time',
        domain : [0, 1]
    },
    yaxis : {
        title : 'luminosity (lux)',
        domain : [0, 1],
        range : [0, 500]
    }
};
```

domain: [0,1] → x 또는 y 축을 100% 사용

range: [0,500] → y 축의 범위를 0~500 설정

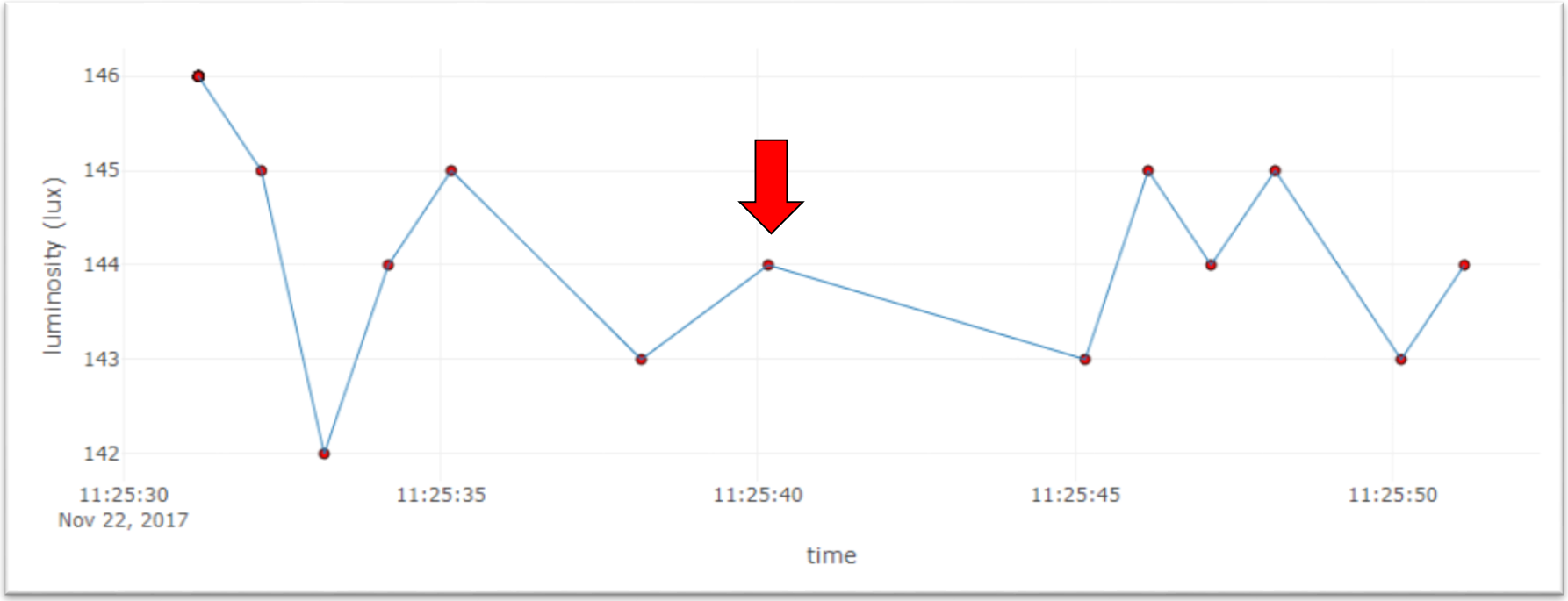
A6.5.6 RT sensor-data streaming in Arduino

[6] Client html : client_ldr.html (real time monitoring of the luminosity)



A6.5.7 RT sensor-data streaming in Arduino

[DIY 1.] Client html : client_ldr_change.html (detecting change)



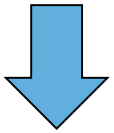
입력되는 lux 값이 변하는 경우에만 그래프를 그림.

실시간 모니터링에서 이상 감지 기능이 필요함.

밝기 값 변화의 문턱값을 설정해서 이상 감지 기능 구현

[DIY 1. hint] Client html : client_ldr_change.html (detecting change)

```
// when new data is coming, keep on streaming data
ctime.innerHTML = dtda[0];
nextPt();
```

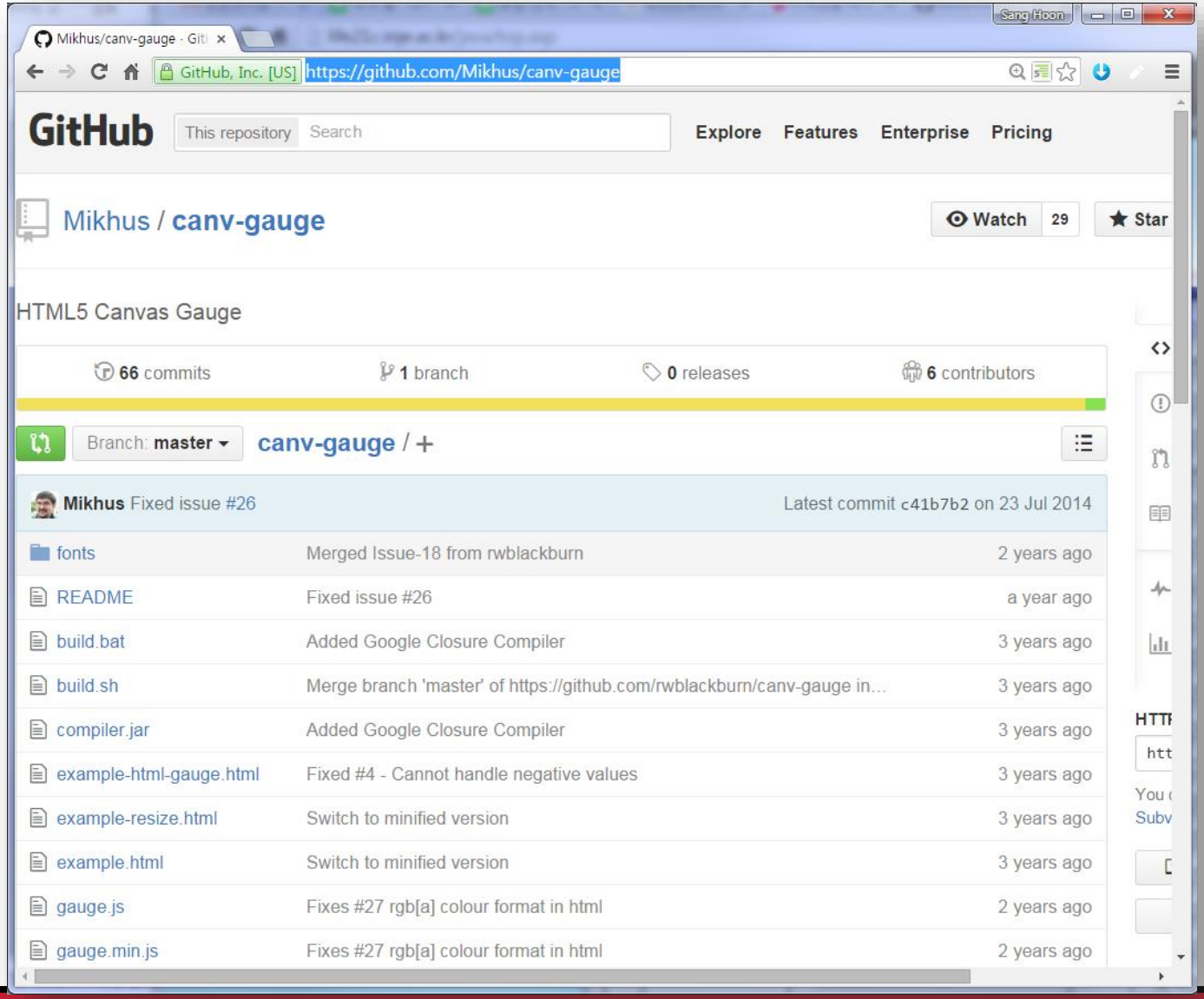


```
// when new data is coming, keep on streaming data
// Only when the value of lux is different from the previous one,
// the screen is redrawed.
if (dtda[1] != preX) {  // any change?
    preX = dtda[1];

    ctime.innerHTML = dtda[0];
    nextPt();
}
```

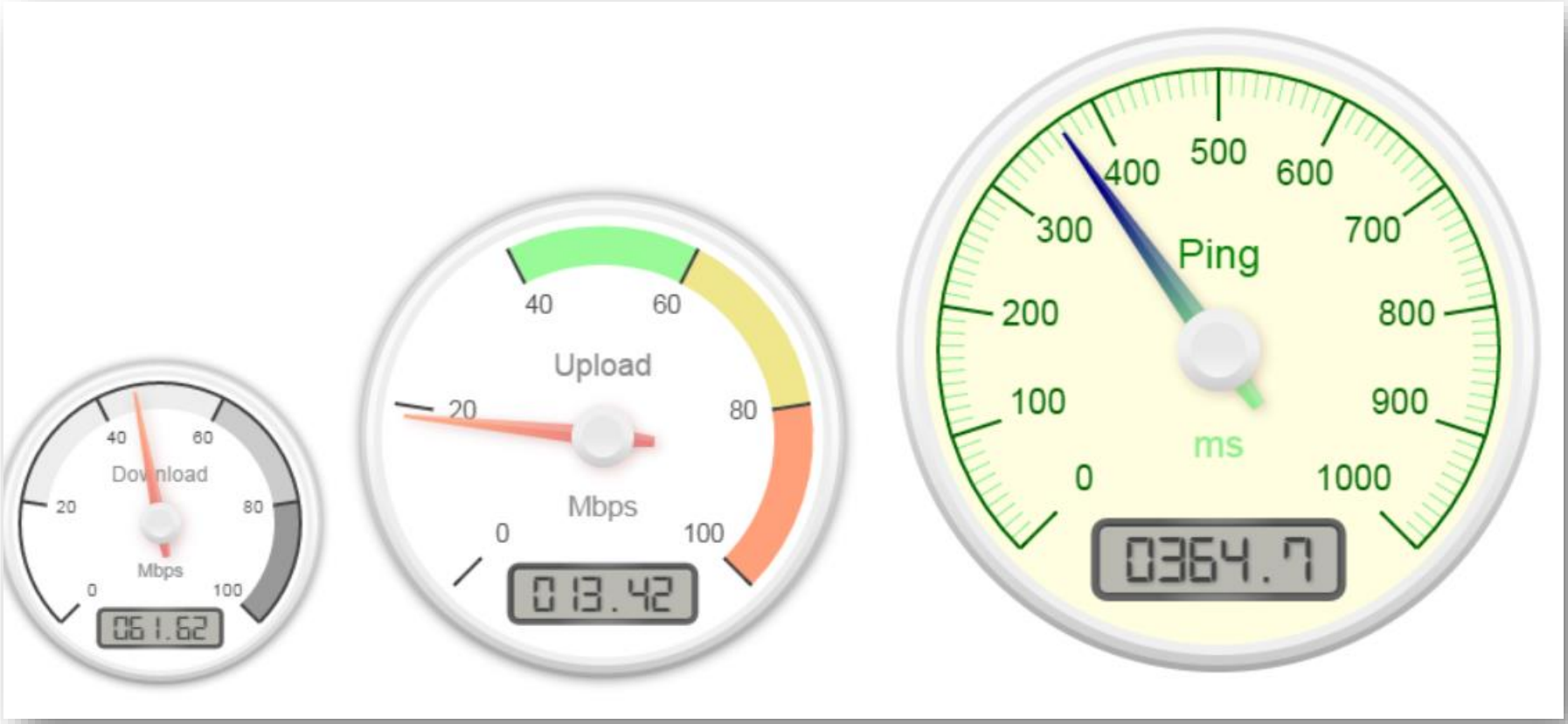
측정되는 주변광의 밝기가 일정 시간 유지되다가 변하는
그래프를 캡처하여 AAnn_ldr_change.png 로 저장

Canvas Gauge

[1] Canvas gauge javascript library : gauge.js



[2] Canvas gauge javascript library : example



http://ru.smart-ip.net/gauge.html

[DIY 2] Client html : client_ldr_gauge.html (add Gauge)

```
<script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
<script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/
socket.io/1.3.6/socket.io.js"></script>

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

```
<body>  <!-- style="width:100%;height:100%"> -->
<!-- Plotly chart will be drawn inside this DIV -->
<h1 align="center"> Real-time Luminosity(lux) from CdS sensor by AAnn</h1>
<!-- Lux gauge -->
<div align="center">
    <canvas id="gauge"> </canvas>
</div>

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

[DIY 2] Client html : client_ldr_gauge.html (add Gauge)

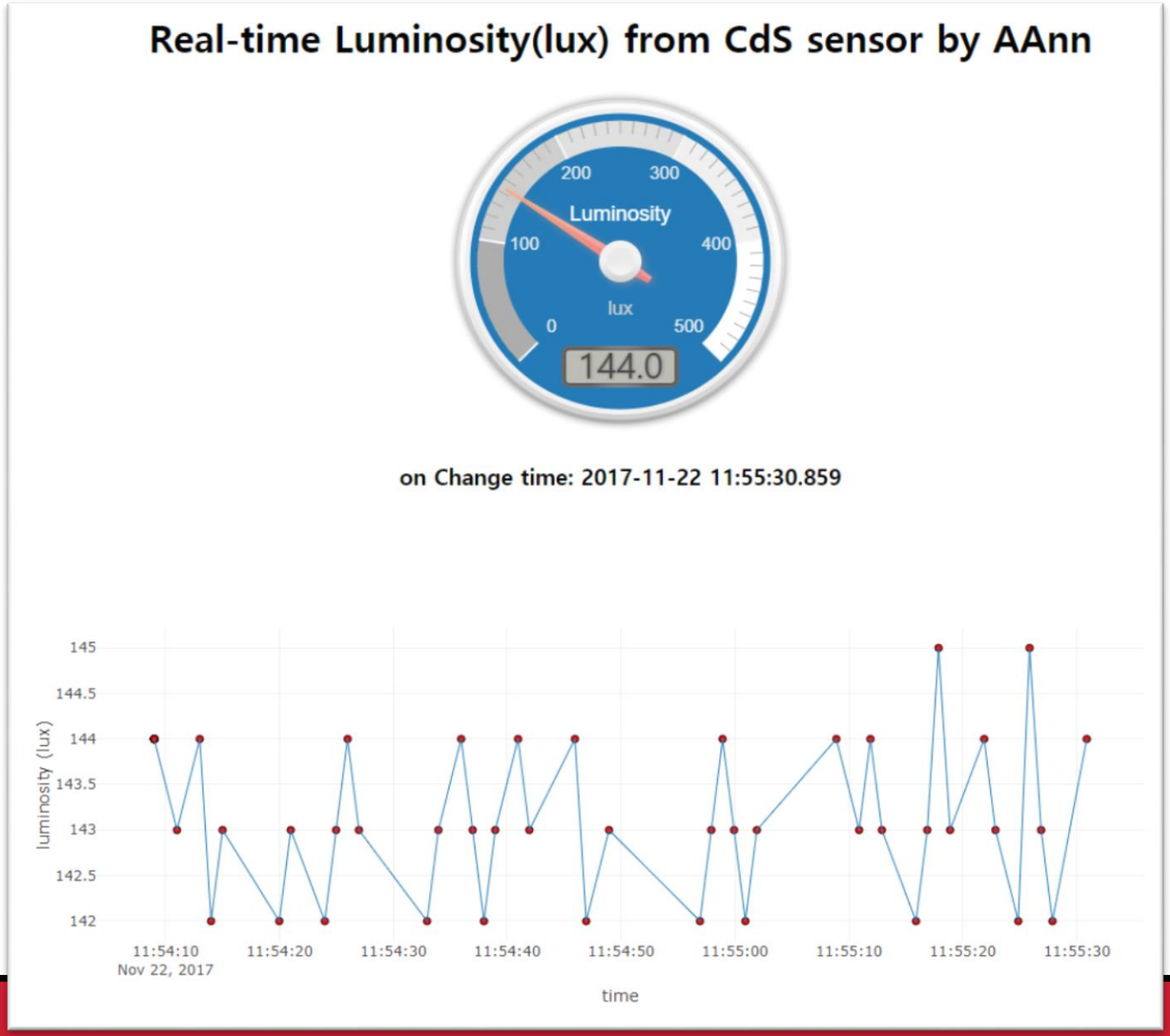
```
// when new data is coming, keep on streaming data
// Only when the value of lux is different from the previous one,
// the screen is redrawed.
if (dtda[1] != preX) {  // any change?
    preX = dtda[1];

    gauge_lux.setValue(dtda[1]);  // lux gauge

    ctime.innerHTML = dtda[0];
    nextPt();
}
```

[DIY 2] Client html : client_ldr_gauge.html (design of Gauge)

```
var gauge_lux = new Gauge({
    renderTo     : 'gauge',
    width        : 300,
    height       : 300,
    glow         : true,
    units        : 'lux',
    valueFormat  : { int : 3, dec : 1 },
    title        : "Luminosity",
    minValue     : 0,
    maxValue     : 500,  // new
    majorTicks   : ['0','100','200','300','400','500'],
    minorTicks   : 10,
    strokeTicks  : false,
    highlights   : [
        { from : 0,   to : 100, color : '#aaa' },
        { from : 100, to : 200, color : '#ccc' },
        { from : 200, to : 300, color : '#ddd' },
        { from : 300, to : 400, color : '#eee' },
        { from : 400, to : 500, color : '#fff' }
    ],
    colors       : {
        plate      : '#1f77b4',
        majorTicks : '#f5f5f5',
        minorTicks : '#aaa',
        title      : '#fff',
        units      : '#ccc',
        numbers    : '#eee',
        needle     : { start : 'rgba(240, 128, 128, 1)', end : 'rgba(255, 160,
            122, .9)' }
    }
});

gauge_lux.draw();
```

[DIY 2] Client html : client_ldr_gauge.html (change design of Gauge)

변경된 디자인으로 된 그래프를 캡처하여 AAnn_ldr_gauge.png 로 저장

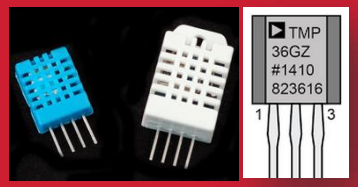


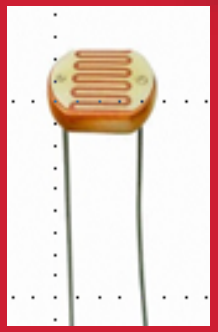
[Tip] Client html : client_ldr_gauge.html (change size of Gauge)

창의 크기 변화에 대응해서 gauge 크기 조정?

```
// draw gauge on canvas
gauge.draw();

window.onresize= function() {
    gauge.updateConfig({
        width  : document.body.offsetWidth,
        height : document.body.offsetHeight
    });
};
```

[Practice]





- [wk13]
  - RT Data Visualization with node.js
  - Usage of gauge.js
  - Complete your real-time LUX charts
  - AAnn_Rpt10.zip

wk13 : Practice-10 : AAnn_Rpt10.zip

- [Target of this week]
  - Complete your plots of real-time streaming of luminosity (lux).
  - Design your own gauge and chart.
  - Save your outcomes and compress 3 figures

제출파일명 : **AAnn_Rpt10.zip**

- 압축할 파일들

① AAnn_DS_30timestamps.png

② AAnn_ldr_change.png

③ AAnn_ldr_gauge.png

Email : chaos21c@gmail.com

[Tip] Using WEB browser in SB text3

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

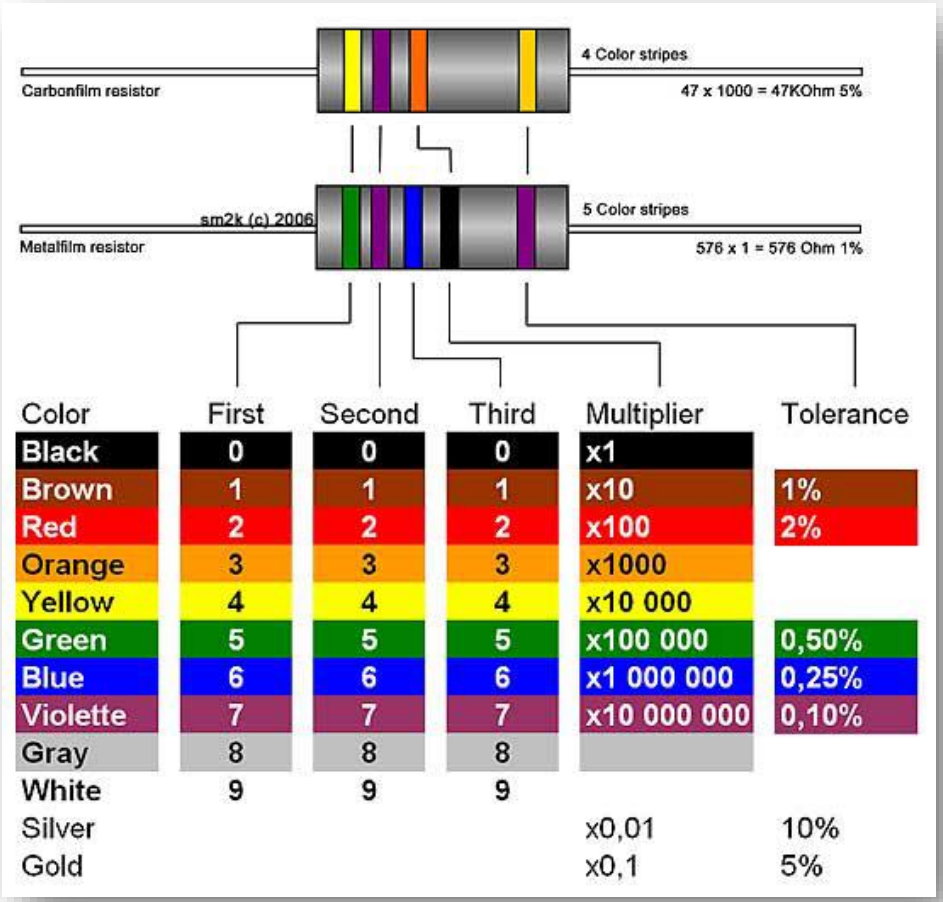
1. Tool -> New Plugin을 실행 한 후 아래 내용으로 덮어 씌운 후 **'open_browser'**으로 저장한다**.**

```
import sublime, sublime_plugin
import webbrowser

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

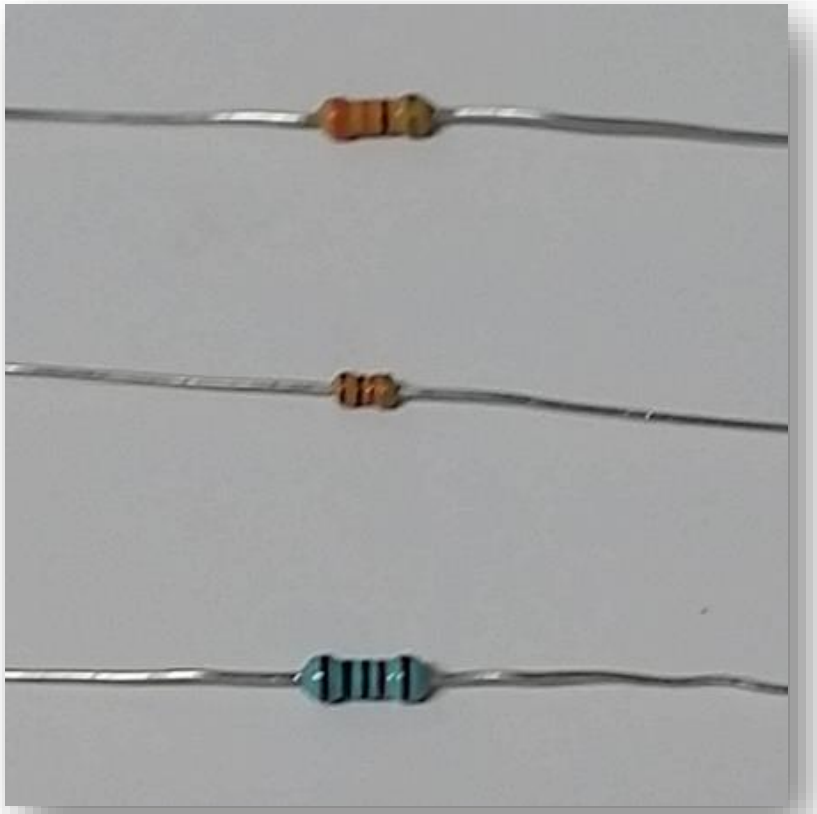
2. Preferences -> Key Bindings - User로 이동한 후 단축키를 할당한다**.**

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

[참고 : 저항 값 읽기]



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



Lecture materials

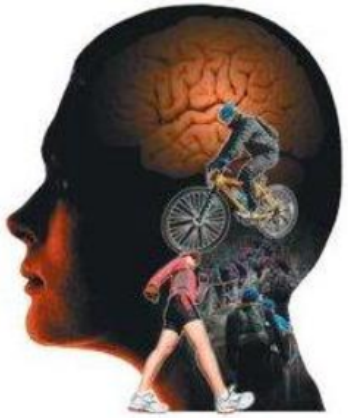
References & good sites

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

Github.com/Redwoods

GitHub, Inc. [US] | https://github.com Redwoods (Redwoods Yi) · GitHub

Features Business Explore Marketplace Pricing Search GitHub Sign in or Sign up



Redwoods Yi

Redwoods

Block or report user

GimHae, Republic of Korea

Overview Repositories 5 Stars 2 Followers 0 Following 0

Pinned repositories

dht22-iot-project

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

HTML

Lec

All lectures by Redwoods in Inje University

arduino-nodejs-plotly-streaming

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

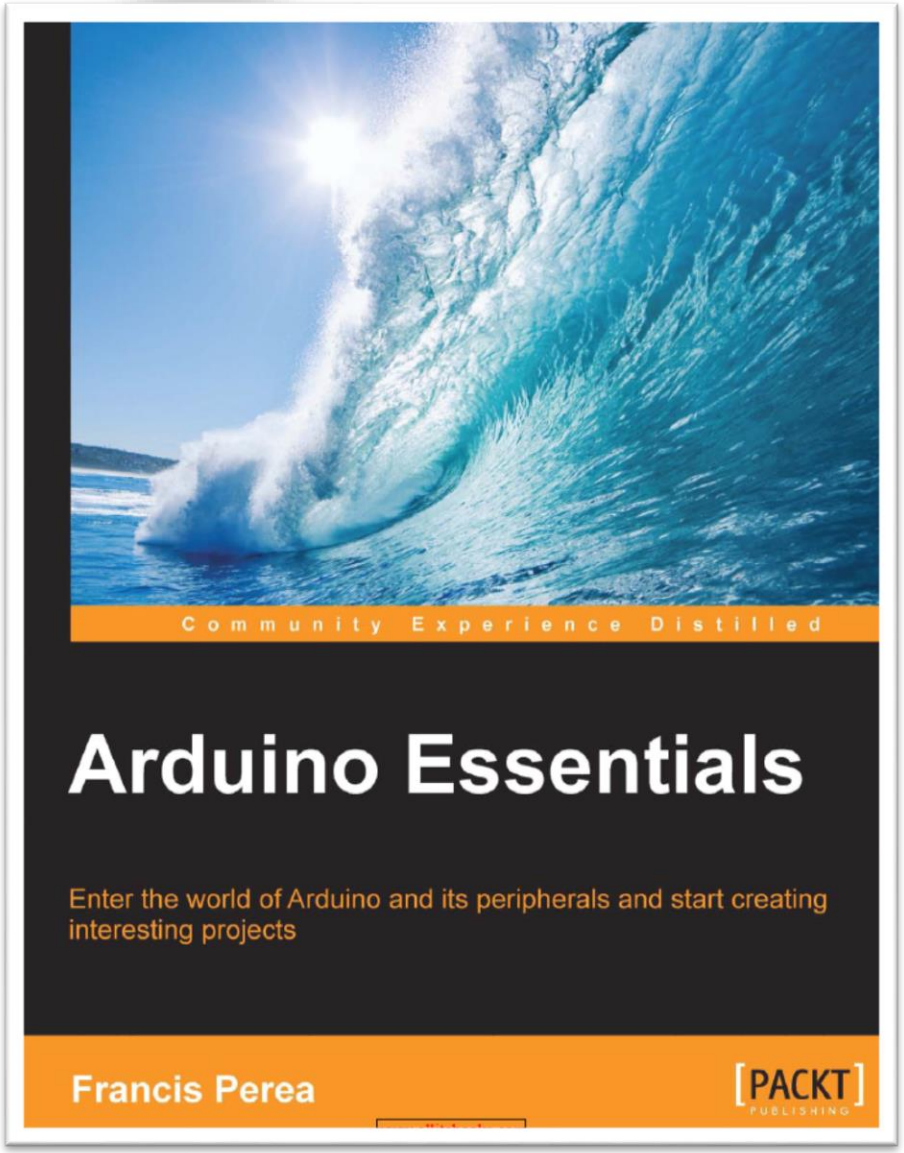
HTML

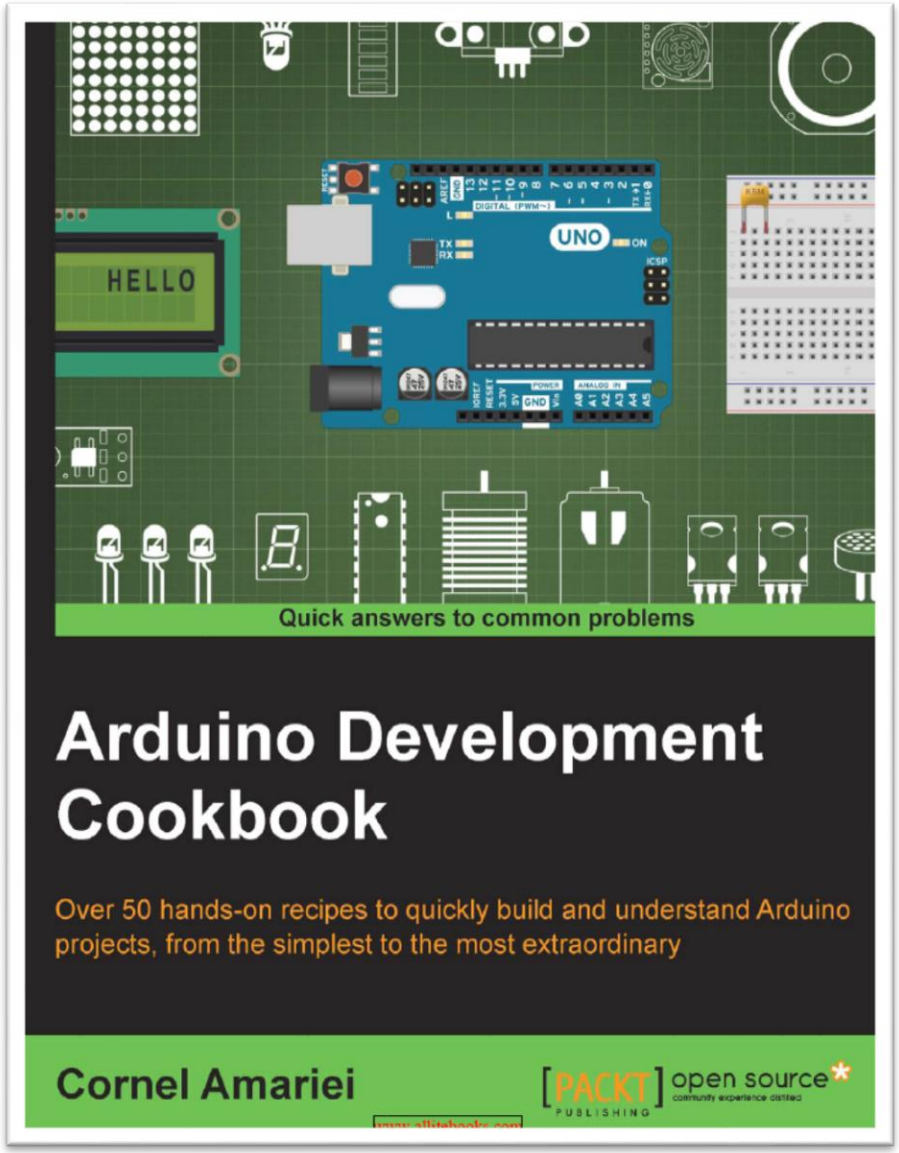
hw-coding

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

Arduino

References


Community Experience Distilled
Arduino Essentials
Enter the world of Arduino and its peripherals and start creating interesting projects
Francis Perea
[PACKT]
PUBLISHING


HELLO
UNO
Quick answers to common problems
Arduino Development Cookbook
Over 50 hands-on recipes to quickly build and understand Arduino projects, from the simplest to the most extraordinary
Cornel Amariei
[PACKT] open source
PUBLISHING