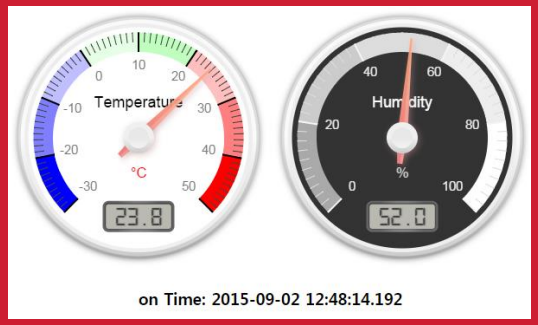
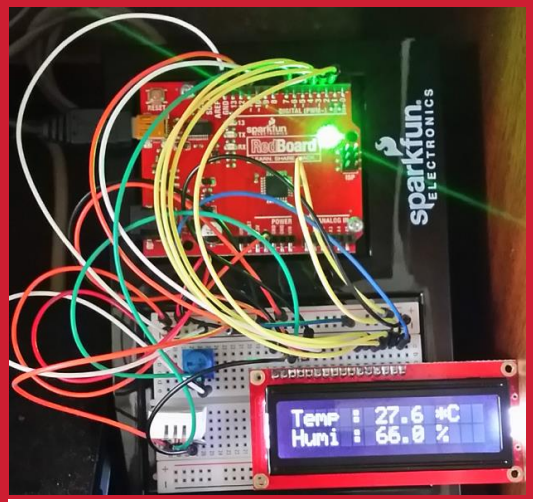
HW-SW-Connectivity

[wk15]

Arduino & NodeJS Project



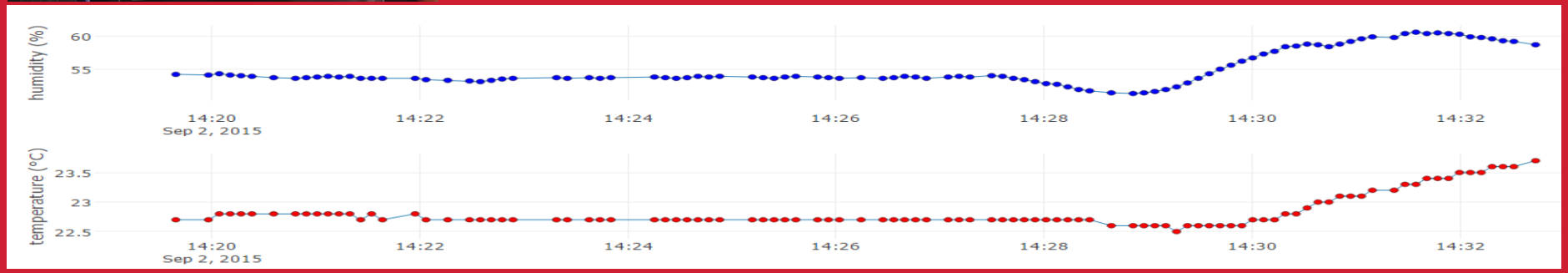


Basic HW and SW Integration using Arduino & Javascript

COMSI, INJE University

2nd semester, 2017

Email : yish@inje.ac.kr



[DIY]

Real-time

TMP36 & CdS monitoring

using plot.ly

→ Data streaming

[5.1] WEB client: client_tmp36_ldr_update.html
(using plotly streaming without nextPt())

Comment function nextPt()

```
// function nextPt() {

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

//      xTrack.shift();
//      xTrack.push(dtda[1]);  // sensor 1:temp
//      yTrack.shift();
//      yTrack.push(dtda[2]);  // sensor 2:lux

//      Plotly.redraw(streamPlot);
// }
```

[5.2] WEB client: client_tmp36_ldr_update.html
(using plotly streaming without nextPt())

```
//nextPt();
xArray = xArray.concat(dtda[0])
xArray.splice(0, 1)  // remove the oldest data
xTrack = xTrack.concat(dtda[1])
xTrack.splice(0, 1)  // remove the oldest data

var update = {
    x:  [xArray],
    y:  [xTrack]
}

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

Complete this part of the code.

```
// function nextPt() {

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

//      xTrack.shift();
//      xTrack.push(dtda[1]);  // sensor 1:temp
//      yTrack.shift();
//      yTrack.push(dtda[2]);  // sensor 2:lux

//      Plotly.redraw(streamPlot);
// }
```

Save the complete code as AAnn_update.html

[5.2] WEB client: client_tmp36_ldr_update.html
(using plotly streaming without nextPt())

Stupid

```
if (dtda[1] != preX || dtda[2] != preY) {
    preX = dtda[1];
    preY = dtda[2];

    gauge_temp.setValue(dtda[1]);
    gauge_lux.setValue(dtda[2]);
    ctime.innerHTML = dtda[0];
    // nextPt();
    xArray = xArray.concat(dtda[0])
    xArray.splice(0, 1)
    xTrack = xTrack.concat(dtda[1])
    xTrack.splice(0, 1)
    yTrack = yTrack.concat(dtda[2])
    yTrack.splice(1, 2)

    var update = {
        x: [xArray],
        y: [xTrack],
        z: [yTrack]

    }

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

Bug?

```
if (dtda[1] != preX || dtda[2] != preY) {
    preX = dtda[1];
    preY = dtda[2];

    gauge_temp.setValue(dtda[1]);
    gauge_lux.setValue(dtda[2]);
    ctime.innerHTML = dtda[0];
    //nextPt();

    xArray = xArray.concat(dtda[0])
    xArray.splice(0,1)
    xTrack = xTrack.concat(dtda[1])
    xTrack.splice(0,1)
    yTrack = yTrack.concat(dtda[2])
    yTrack.splice(0,1)

    var update = {
        x: [xArray],
        y: [xTrack,yTrack]
    }

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

Great

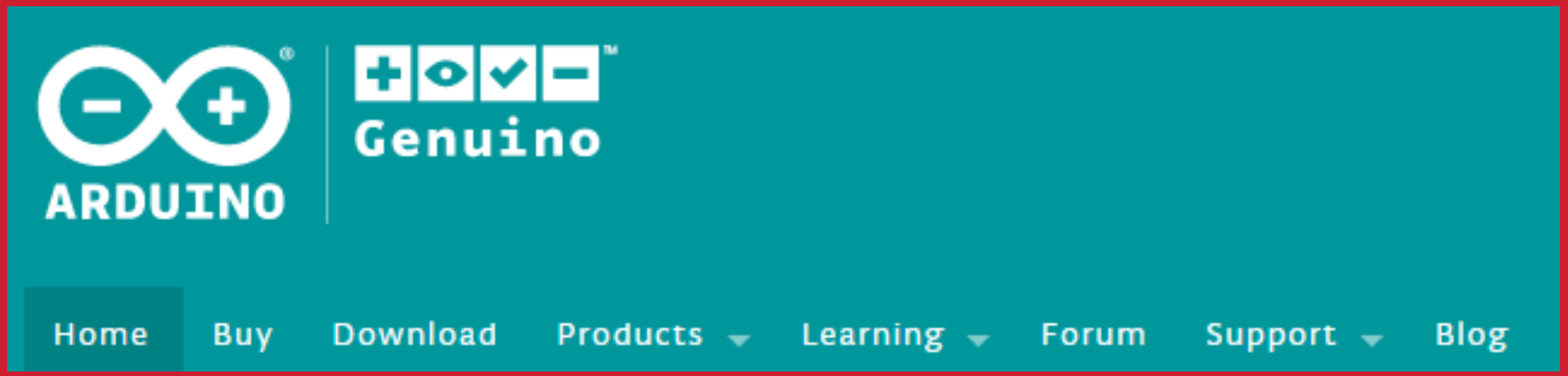
```
if (dtda[1] != preX || dtda[2] != preY) {
    preX = dtda[1];
    preY = dtda[2];

    // when new data is coming, keep on st
    ctime.innerHTML = dtda[0];
    gauge_temp.setValue(dtda[1])  // temp
    gauge_lux.setValue(dtda[2]);  // lux g
    //nextPt();
    xArray = xArray.concat(dtda[0])
    xArray.splice(0, 1)  // remove the old
    xTrack = xTrack.concat(dtda[1])
    xTrack.splice(0, 1)  // remove the old
    yTrack = yTrack.concat(dtda[2])
    yTrack.splice(0, 1)

    var update = {
        x:  [xArray, xArray],
        y:  [xTrack, yTrack]
    }

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

Arduino

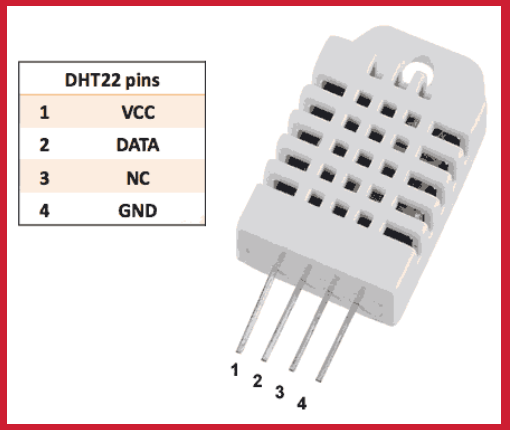


https://www.arduino.cc/

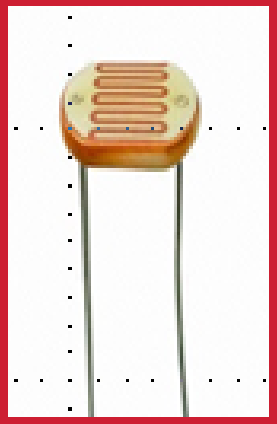
Arduino & Node.js

Multi-sensors

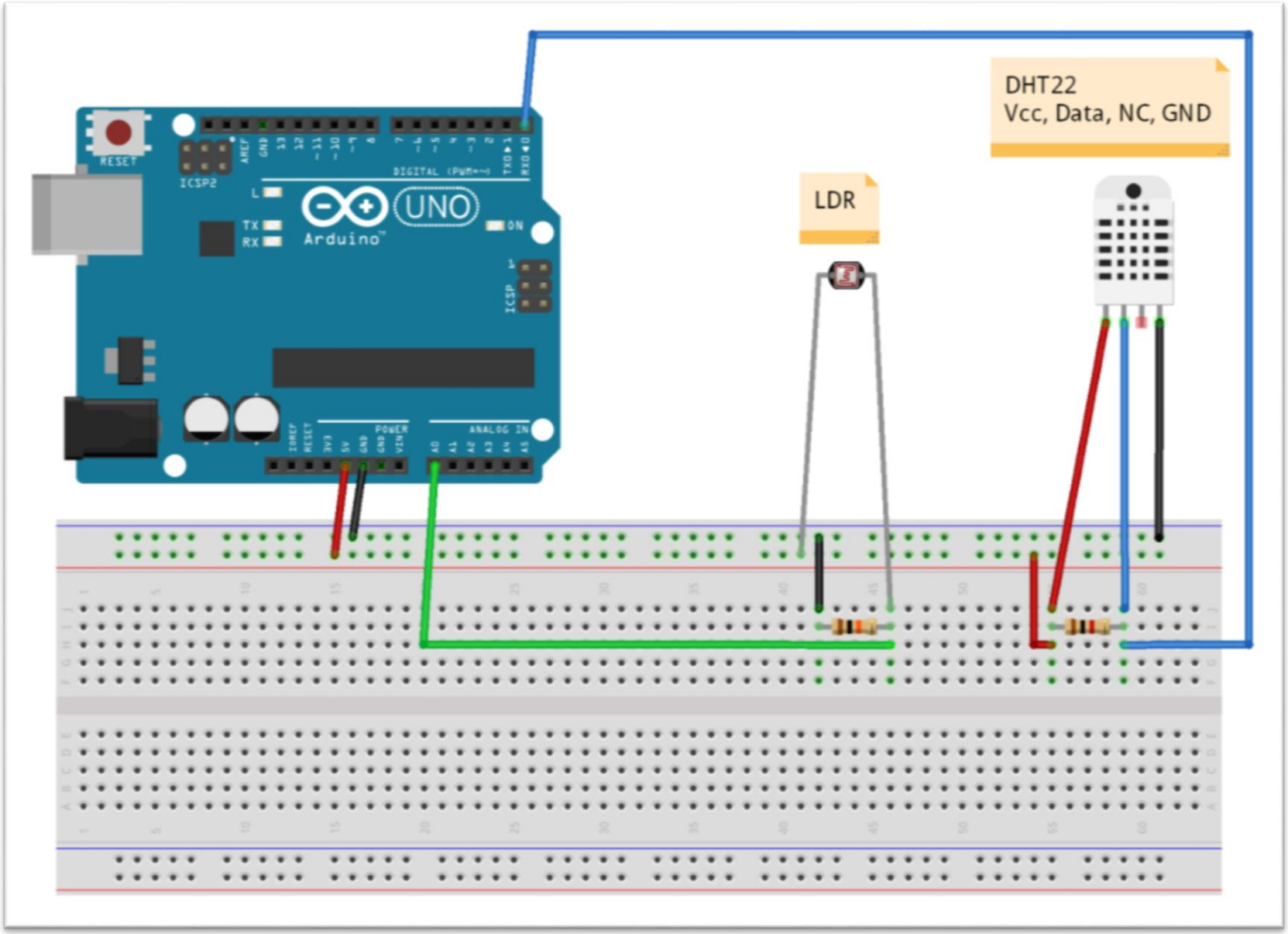
DHT22 + CdS



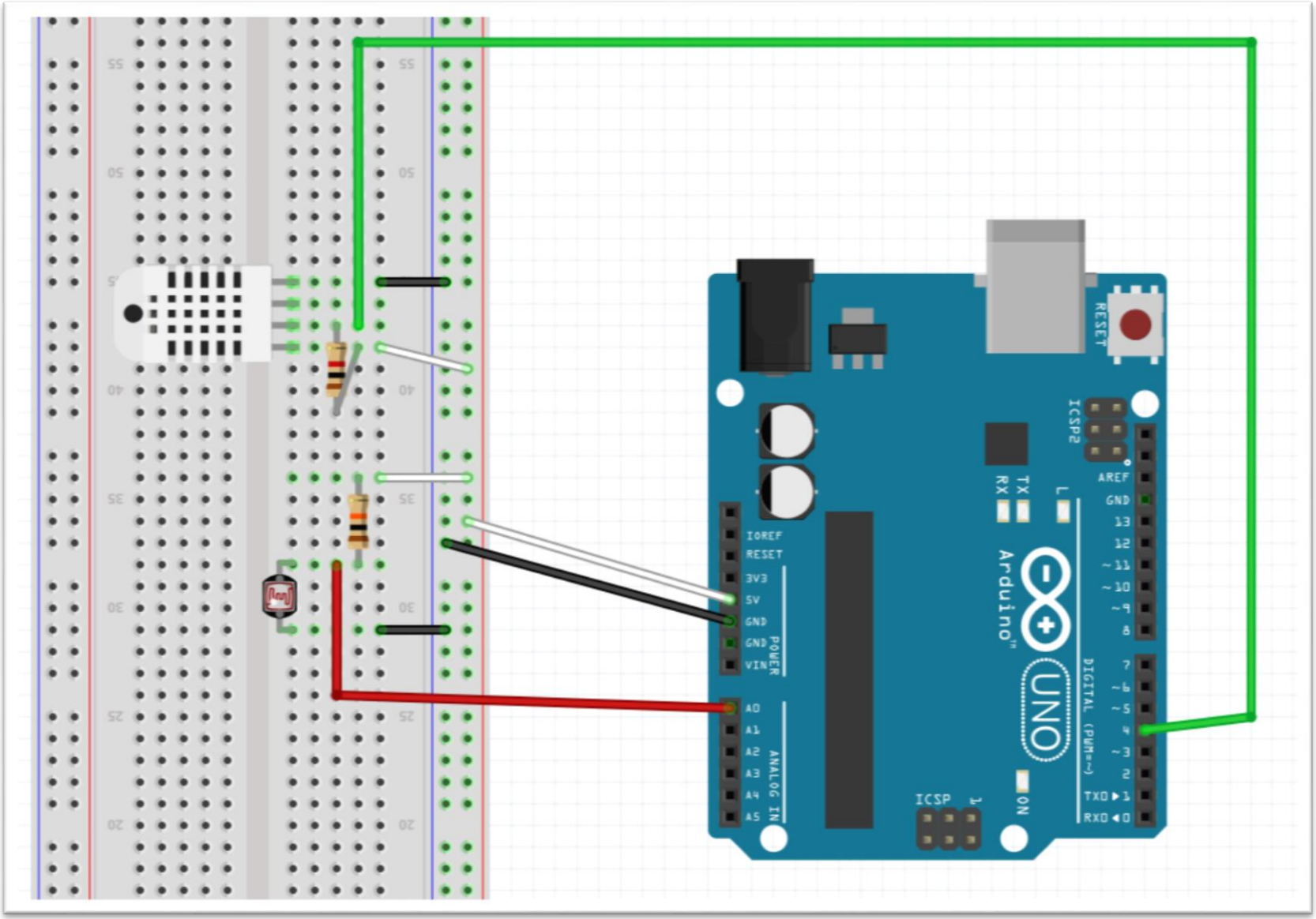
DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND



A7.1 DHT22 + CdS : circuit (Google)



A7.1 DHT22 + CdS : circuit



A7.1 DHT22 + CdS : DHT library

Features Business Explore Marketplace Pricing This repository

adafruit / DHT-sensor-library

Code Issues 21 Pull requests 15 Projects 0 Wiki Insights

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

54 commits 1 branch 8 releases

Branch: master New pull request

microbuilder Merged unified and raw libraries

.github	Add GitHub issue template
examples	Merged unified and raw libraries
DHT.cpp	Fix #44 by conditionally excluding unused port and bitmask state on n...
DHT.h	Fix #44 by conditionally excluding unused port and bitmask state on n...
DHT_U.cpp	Merged unified and raw libraries
DHT_U.h	Merged unified and raw libraries

A7.1.1 DHT22 + CdS : circuit

[1] Arduino code: AAnn_DHT22_CdS.ino

AAnn_DHT22_CdS

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

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

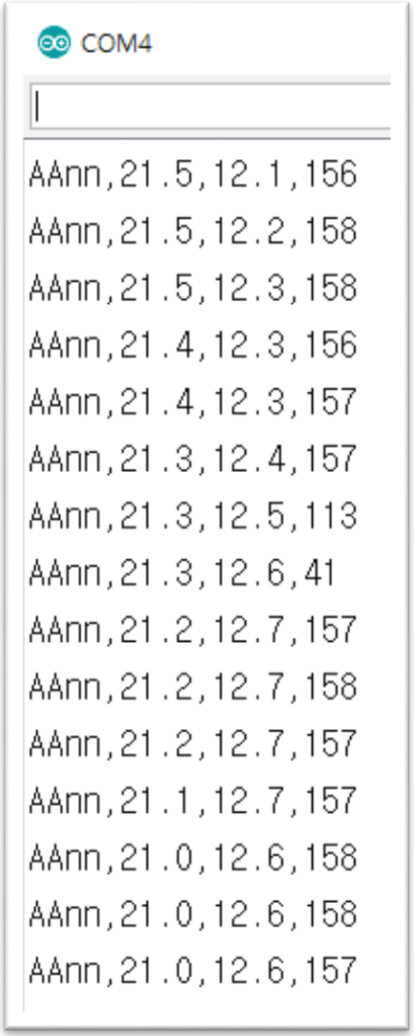
```
//Voltage to LuxLux
double luminosity (int RawADC0){
  double Vout=RawADC0*0.0048828125;  // 5/1
  int lux=(2500/Vout-500)/10;
  // lux = 500 / Rldr, Vout = Ildr*Rldr = (
  return lux;
}
```

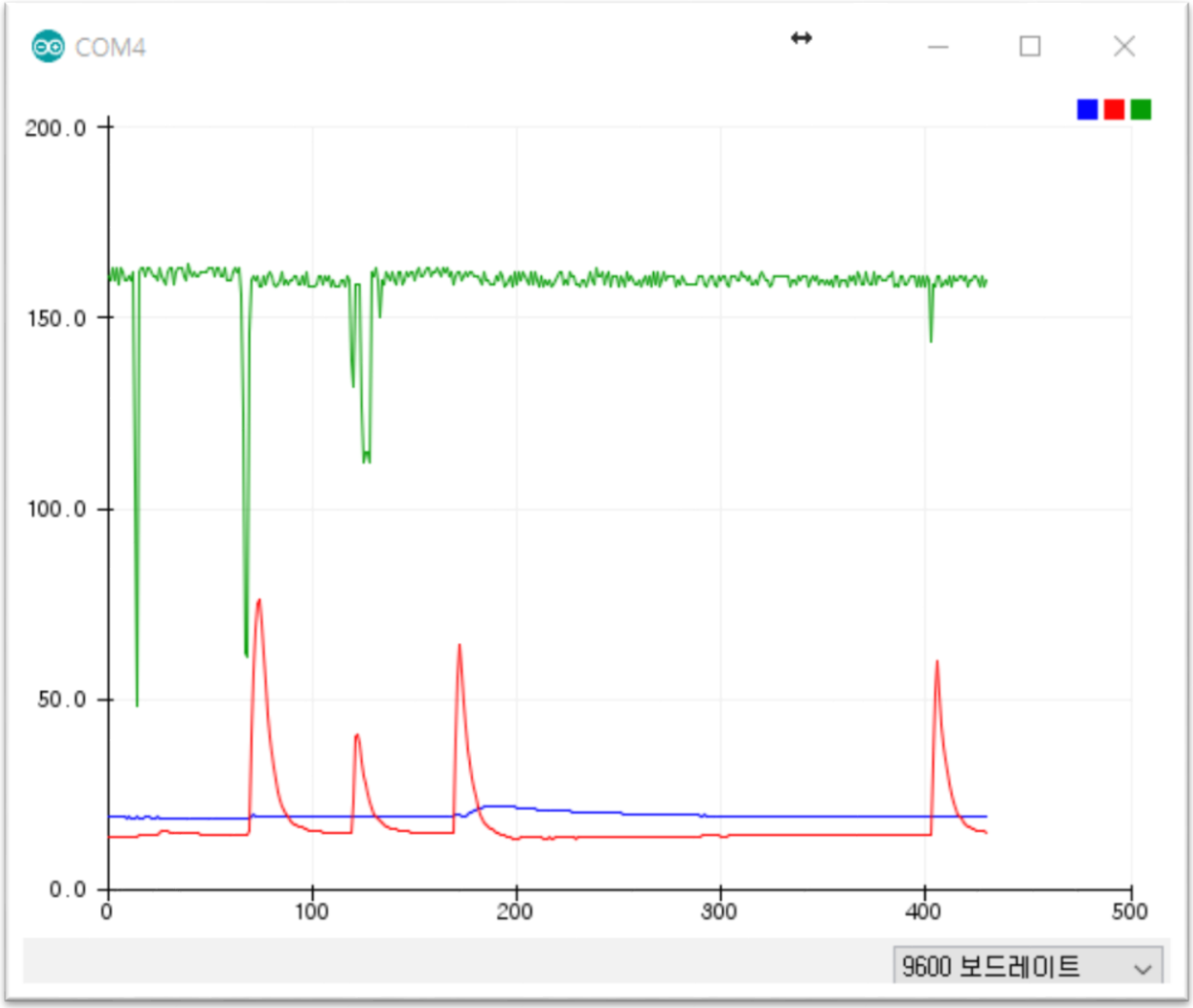
```
void loop() {
  int cds_value, lux;
  float temp, humi;
  // Lux from CdS (LDR)
  cds_value = analogRead(CDS_INPUT);
  lux = int(luminosity(cds_value));
  // Reading temperature or humidity takes a given interval!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  humi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

  // Check if any reads failed and exit early (to try again).
  if (isnan(humi) || isnan(temp) || isnan(lux)) {
    Serial.println("Failed to read from DHT sensor or CdS!");
    return;
  }
  else {
    Serial.print("AAnn,");
    Serial.print(temp,1);  // temperature
    Serial.print(",");
    Serial.print(humi,1);  // humidity
    Serial.print(",");
    Serial.println(lux);   // luminosity
  }
  delay(2000);  // 2000 msec, a data per 6 min = 6 * 60 * 1000 = 360000
}
```

A7.1.2 DHT22 + CdS : Serial monitor

[1] Arduino code: AAnn_DHT22_CdS.ino





A7.2.1 DHT22 + CdS + Node.js

[2.1] NodeJS code: dht22_ldr_node.js (← tmp36_ldr_node.js)

```
// tmp36_ldr_node.js

var serialport = require('serialport');
var portName = 'COM4';  // check your COM port!!
var port     =   process.env.PORT || 3000;

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

// serial port object
var sp = new serialport(portName,{
    baudRate: 9600,   // 9600  38400
    dataBits: 8,
    parity: 'none',
    stopBits: 1,
    flowControl: false,
    parser: serialport.parsers.readline('\r\n')
});
```

A7.2.2 DHT22 + CdS + Node.js

[2.2] NodeJS code: dht22_ldr_node.js (Complete your parser code)

```
var readData = '';  // this stores the buffer
var temp ='';
var humi ='';
var lux ='';
var mdata =[]; // this array stores date and data from multiple sensors
var firstcommaidx = 0;

sp.on('data', function (data) { // call back when data is received
    readData = data.toString(); // append data to buffer
    firstcommaidx = readData.indexOf(','); // string.indexOf(searchvalue,start)
    //console.log(data);

    // parsing data into signals

        //console.log(firstcolonidx + "," + readData.indexOf(':', firstcolonidx+1))
        readData = '';

        dStr = getDateString();
        mdata[0]=dStr;  // Date
        mdata[1]=temp;  // temperature data
        mdata[2]=humi;  // humidity data
        mdata[3]=lux;   // luminosity data
        console.log(mdata);
        io.sockets.emit('message', mdata);  // send data to all clients

    } else {  // error
        console.log(readData);
    }

});
```

Complete your parser code!!

A7.3 DHT22 + CdS + Node.js

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

COM4

```
AAnn,21.5,12.1,156
AAnn,21.5,12.2,158
AAnn,21.5,12.3,158
AAnn,21.4,12.3,156
AAnn,21.4,12.3,157
AAnn,21.3,12.4,157
AAnn,21.3,12.5,113
AAnn,21.3,12.6,41
AAnn,21.2,12.7,157
AAnn,21.2,12.7,158
AAnn,21.2,12.7,157
AAnn,21.1,12.7,157
AAnn,21.0,12.6,158
AAnn,21.0,12.6,158
AAnn,21.0,12.6,157
```

NodeJS - node dht22_ldr_node

```
D:\Portable\NodeJSPortable\Data\aa00\ldr>node dht22_ldr_node
[ '2017-12-05 17:55:08.320', '18.8', '14.1', '160' ]
[ '2017-12-05 17:55:10.593', '18.8', '14.1', '158' ]
[ '2017-12-05 17:55:12.851', '18.8', '14.1', '160' ]
[ '2017-12-05 17:55:15.125', '18.8', '14.1', '155' ]
[ '2017-12-05 17:55:17.397', '18.8', '14.1', '76' ]
[ '2017-12-05 17:55:19.670', '18.8', '14.1', '158' ]
[ '2017-12-05 17:55:21.943', '19.3', '33.9', '140' ]
[ '2017-12-05 17:55:24.216', '19.5', '50.6', '158' ]
[ '2017-12-05 17:55:26.474', '19.2', '52.5', '160' ]
[ '2017-12-05 17:55:28.747', '19.1', '46.7', '159' ]
[ '2017-12-05 17:55:31.021', '19.1', '40.9', '160' ]
[ '2017-12-05 17:55:33.293', '19.1', '36.3', '159' ]
[ '2017-12-05 17:55:35.566', '19.1', '32.0', '161' ]
[ '2017-12-05 17:55:37.839', '19.1', '28.6', '159' ]
[ '2017-12-05 17:55:40.097', '19.1', '25.6', '158' ]
[ '2017-12-05 17:55:42.370', '19.1', '23.4', '158' ]
[ '2017-12-05 17:55:44.644', '19.1', '21.6', '160' ]
[ '2017-12-05 17:55:46.916', '19.0', '20.0', '158' ]
[ '2017-12-05 17:55:49.188', '19.1', '19.2', '135' ]
[ '2017-12-05 17:55:51.462', '19.0', '18.5', '146' ]
[ '2017-12-05 17:55:53.720', '19.5', '43.0', '146' ]
[ '2017-12-05 17:55:55.993', '19.5', '54.1', '160' ]
[ '2017-12-05 17:55:58.265', '19.3', '46.4', '158' ]
[ '2017-12-05 17:56:00.538', '19.2', '39.5', '160' ]
[ '2017-12-05 17:56:02.812', '19.2', '34.6', '160' ]
[ '2017-12-05 17:56:05.068', '19.2', '31.4', '158' ]
[ '2017-12-05 17:56:07.343', '19.2', '28.1', '159' ]
[ '2017-12-05 17:56:09.616', '19.3', '25.5', '158' ]
[ '2017-12-05 17:56:11.890', '19.5', '23.4', '161' ]
[ '2017-12-05 17:56:14.162', '19.7', '21.7', '159' ]
[ '2017-12-05 17:56:16.436', '20.0', '20.3', '158' ]
[ '2017-12-05 17:56:18.693', '20.2', '19.5', '160' ]
[ '2017-12-05 17:56:20.965', '20.5', '18.7', '161' ]
[ '2017-12-05 17:56:23.238', '20.8', '18.1', '159' ]
```

Save as AAnn_dht22_ldr_data.png

A7.4.1 DHT22 + CdS + Node.js

[4.1] WEB client: client_dht22_ldr.html

```html
<!DOCTYPE html>
<head>
  <meta charset="utf-8">
  <title>plotly.js Project: Real time signals from multiple sensors</title>
  <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
  <script type="text/javascript"
  src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/1.3.6/socket.io.js"></script>

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

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

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

A7.4.2 DHT22 + CdS + Node.js

[4.2] WEB client: client_dht22_ldr.html

```
29      <script>
30        /* JAVASCRIPT CODE GOES HERE */
31        var streamPlot = document.getElementById('myDiv');
32        var ctime = document.getElementById('time');
33      /*var streamToggle = document.getElementById('streamtoggle');
34      */
35      var xArray = [], // time of data arrival
36          y1Track = [], // value of sensor 1 : temperature
37          y2Track = [], // value of sensor 2 : humidity
38          y3Track = [], // value of sensor 3 : Luminosity
39          numPts = 100, // number of data points in x-axis
40          dtda = [],  // 1 x 4 array : [date, data1, data2, data3]
41          preX = -1,
42          preY = -1,
43          preZ = -1,
44          initFlag = true;
```

Check

xTrack → y1Track, yTrack → y2Track

& add y3Track

A7.4.3 DHT22 + CdS + Node.js

[4.3] WEB client: client_dht22_ldr.html

```
var socket = io.connect('http://localhost: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]);  // tempe
            dtda[2]=parseFloat(msg[2]);  // Lumin
            init();
            initFlag=false;
        }

        // Convert value to integer
        dtda[0]=msg[0];
        dtda[1] = parseFloat(msg[1]);
        dtda[2] = parseFloat(msg[2]);
```

```
// Only when any of temperature or Luminosity is di
// the screen is redrawed
if (dtda[1] != preX || dtda[2] != preY) {  // any ch
    preX = dtda[1];
    preY = dtda[2];

    // when new data is coming, keep on streaming
    ctime.innerHTML = dtda[0];
    gauge_temp.setValue(dtda[1])  // temp gauge
    gauge_lux.setValue(dtda[2]);  // lux gauge
    //nextPt();
    xArray = xArray.concat(dtda[0])
    xArray.splice(0, 1)  // remove the oldest data
    y1Track = y1Track.concat(dtda[1])
    y1Track.splice(0, 1)  // remove the oldest data
    y2Track = y2Track.concat(dtda[2])
    y2Track.splice(0, 1)

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

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

Update

to include three signals:

temp, humi, lux

A7.4.4 DHT22 + CdS + Node.js

[4.4] WEB client: client_dht22_ldr.html

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

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

Update

to include three signals:

temp, humi, lux

A7.4.5 DHT22 + CdS + Node.js

[4.5] WEB client: client_dht22_ldr.html - data

```
// data
var data = [{
    x : xArray,
    y : y1Track,
    name : 'temperature',
    mode: "markers+lines",  // "
    line: {
        color: "#1f77b4",
        width: 1
    },
    marker: {
        color: "rgb(255, 0, 0)",
        size: 6,
        line: {
          color: "black",
          width: 0.5
        }
    }
},
```

```
{
x : xArray,
y : y2Track,
name : 'humidity',
xaxis: 'x2',
yaxis : 'y2',
    mode: "markers+lines",  // "
    line: {
        color: "#1f77b4",
        width: 1
    },
    marker: {
        color: "rgb(0, 0, 255)",
        size: 6,
        line: {
          color: "black",
          width: 0.5
        }
    }
},
```

```
{
x : xArray,
y : y3Track,
name : 'luminosity',
xaxis: 'x3',
yaxis : 'y3',
    mode: "markers+lines",  // "
    line: {
        color: "#1f77b4",
        width: 1
    },
    marker: {
        color: "rgb(0, 255, 0)",
        size: 6,
        line: {
          color: "black",
          width: 0.5
        }
    }
}];
```

Update data

to include three signals:

temp, humi, lux

A7.4.6 DHT22 + CdS + Node.js

[4.6] WEB client: client_dht22_ldr.html - layout

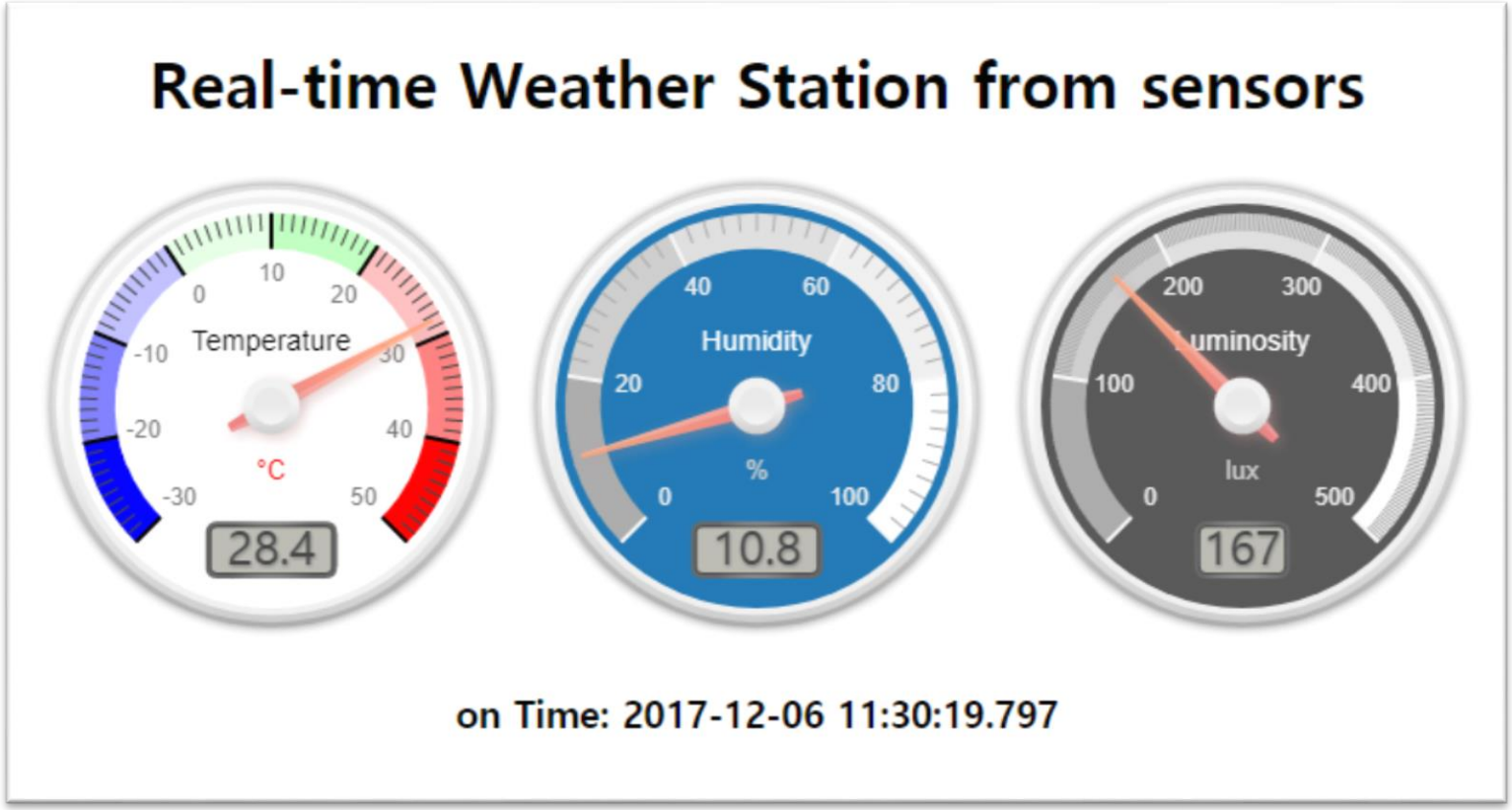
```
var layout = {
    xaxis : {
        title : 'time',
        domain : [0, 1]
    },
    yaxis : {
        title : 'temp (°C)',
        domain : [0, 0.3],
        range : [-30, 50]
    },
    xaxis2 : {
        title : '',
        domain : [0, 1],
        position : 0.35
    },
    yaxis2 : {
        title : 'humi (%)',
        domain : [0.35, 0.65],
        range : [0, 100]
    },
    xaxis3 : {
        title : '',
        domain : [0, 1],
        position : 0.7
    },
    yaxis3 : {
        title : 'lumi (lux)',
        domain : [0.7, 1],
        range : [0, 500]
    }
```

1. Update layout to include three signals: temp, humi, lux.
2. Check the domain & position.

Save the complete code as AAnn_dht22_ldr.html

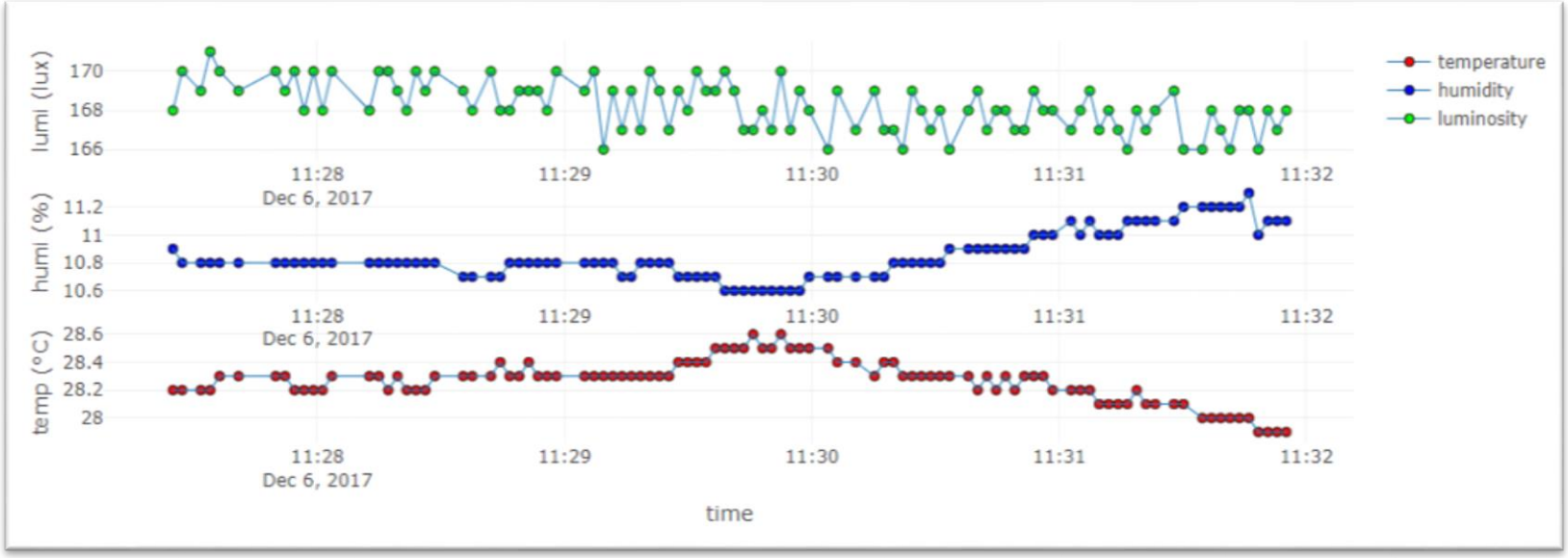
A7.5.1 DHT22 + CdS + Node.js

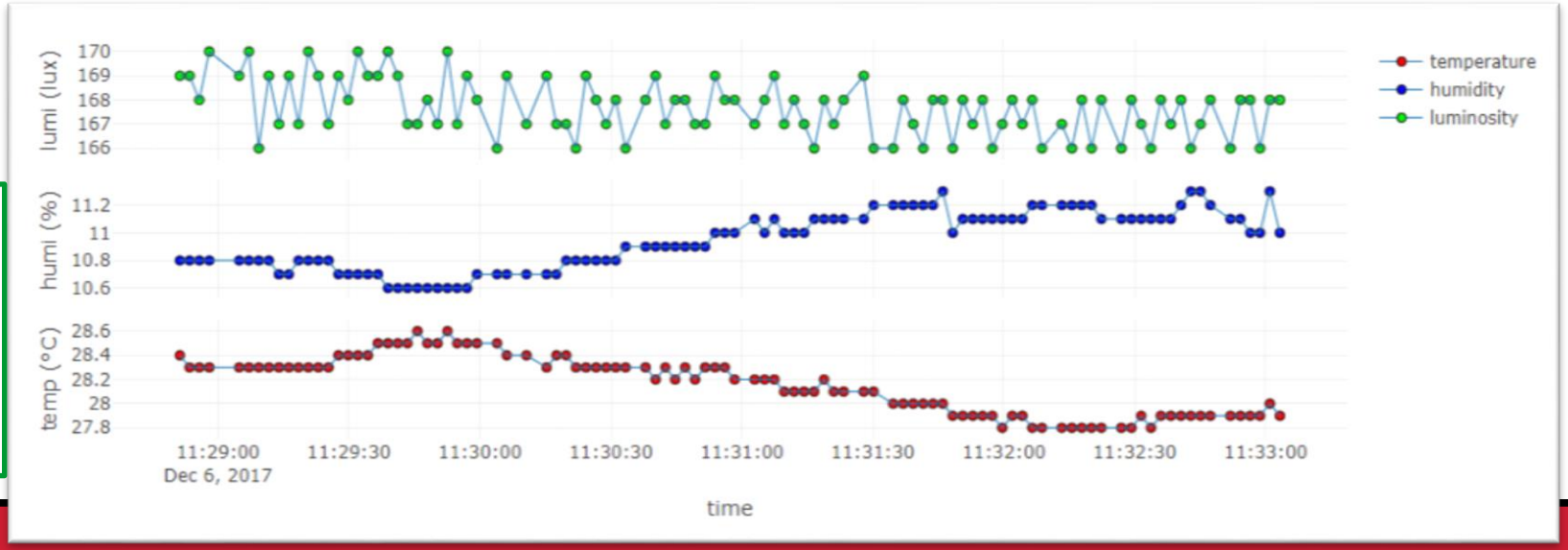
[5.1] WEB client: client_dht22_ldr.html – Design your gauges



A7.5.2 DHT22 + CdS + Node.js

[5.2] WEB client: client_dht22_ldr.html – Design layout



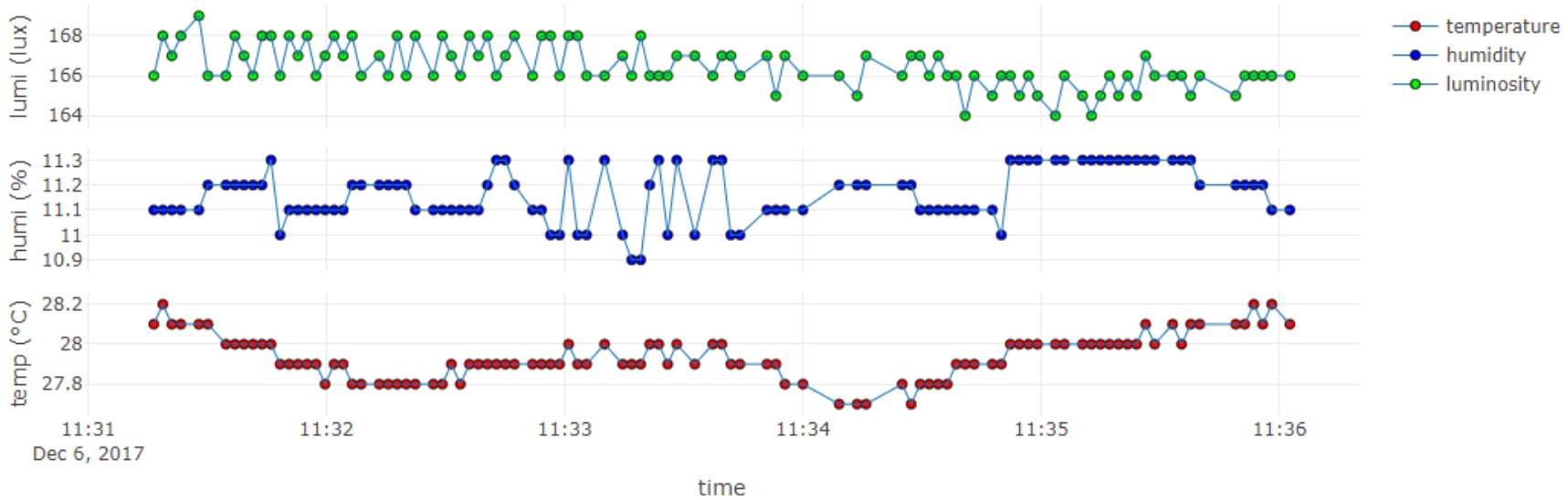


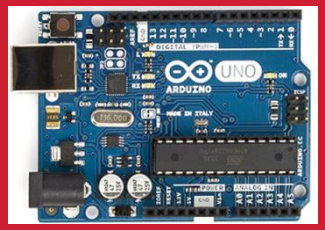
[Hint]

[Plot.ly]

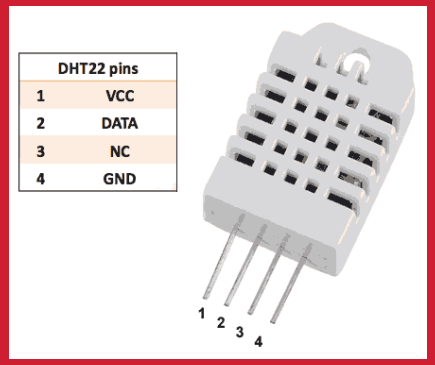
Real-time Weather Station from sensors

on Time: 2017-12-06 11:36:02.639





[Practice]



- [wk15 - 10점 기말 실기]
  - RT Data Visualization with node.js
  - DHT22 + CdS sensors
  - Complete your real-time charts
  - AAnn_Rpt12.zip

wk15 : Practice-12 : AAnn_Rpt12.zip

- [Target of this week]
  - Complete your plots of real-time streaming of DHT22 & CdS
  - Design your own gauge and layout.
  - Save your outcomes and compress them.

제출파일명 : **AAnn_Rpt12.zip**

- 압축할 파일들

① **AAnn_dht22_ldr_data.png**

② **AAnn_dht22_ldr.html**

Email : chaos21c@gmail.com

wk15 : Final Examination

필기시험

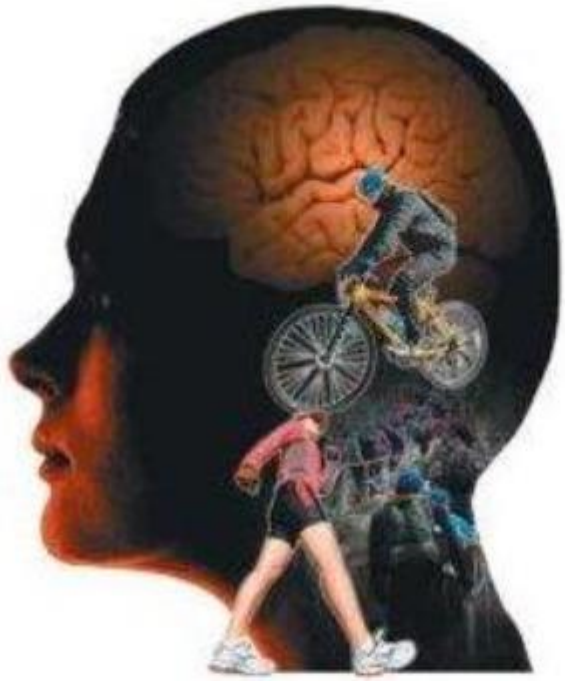
1. 일시: **12월 13일(수)** 오후 **2**시 **~ 3**시

2. 장소: **E323**

3. **20**문제 **(**객관식 및 단답형**)**

4. 범위 **: node.js + arduino (**중간고사 이후 **code** 출제**)**

Further study to store data to MongoDB



Overview | Repositories 5 | Stars 2 | Followers

Redwoods Yi

Redwoods

Block or report user

Pinned repositories

dht22-iot-project

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

HTML

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.

[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.

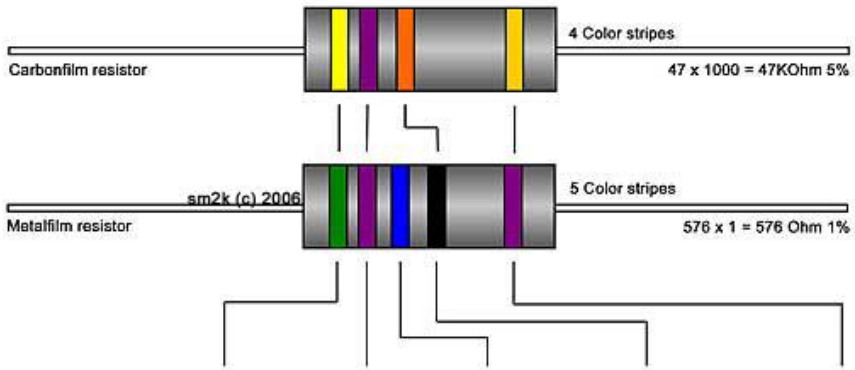
```
import sublime, sublime_plugin
import webbrowser

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

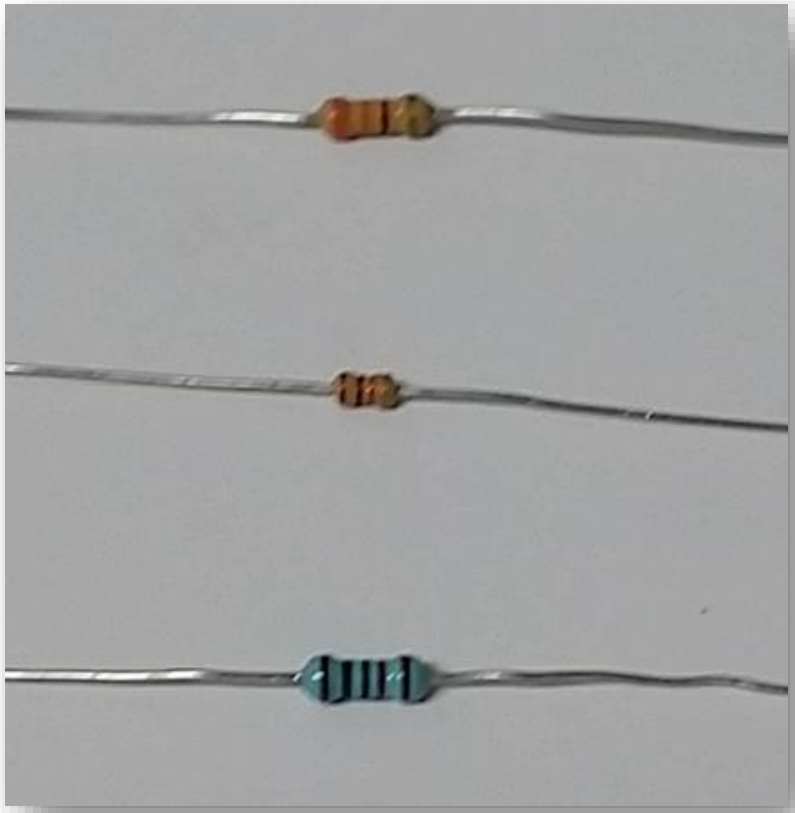
2. Preferences -> Key Bindings - User로 이동한 후 단축키를 할당한다.

```
{ "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

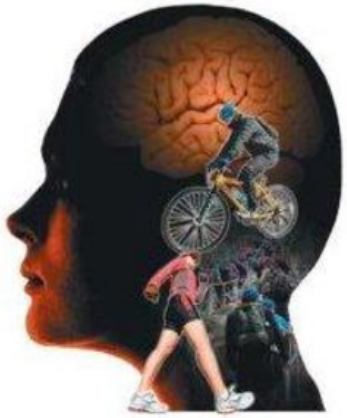
References & good sites

- ✓ 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.

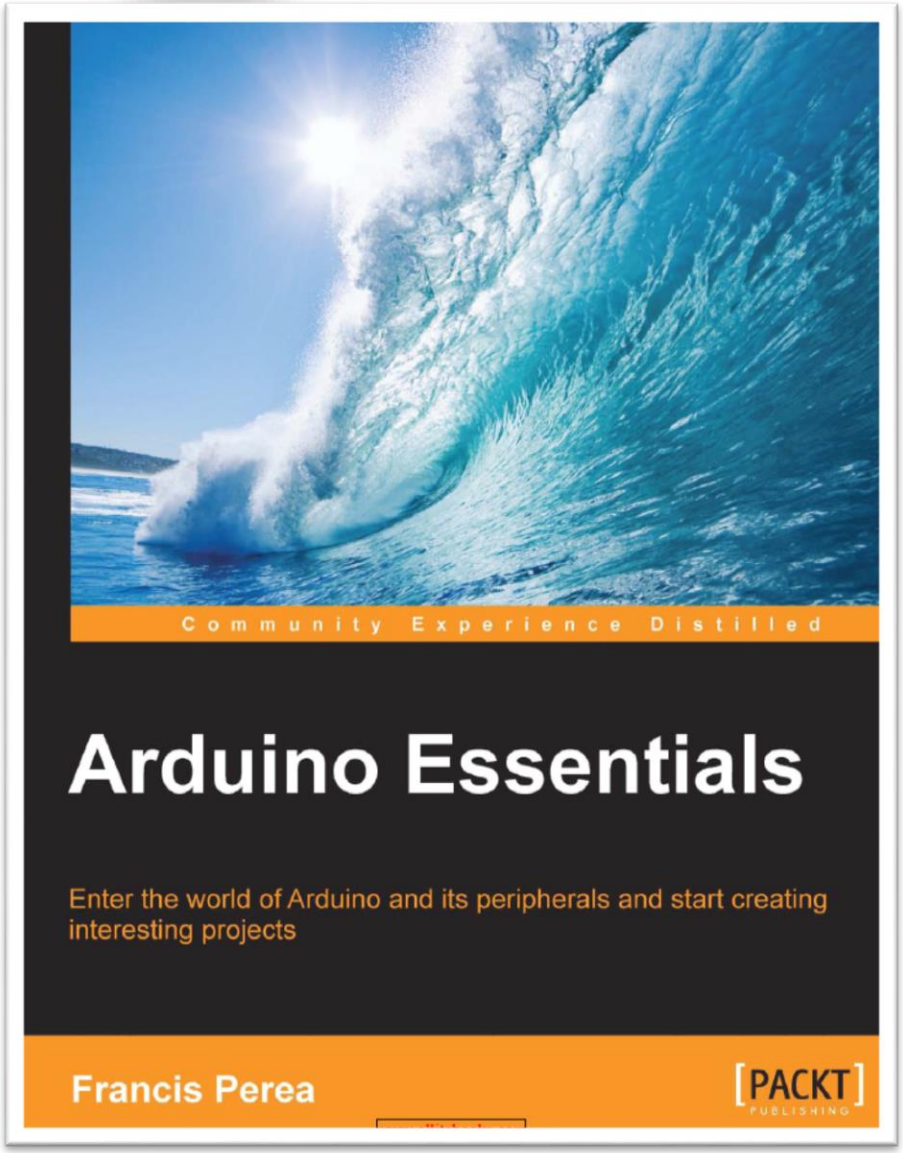
HTML

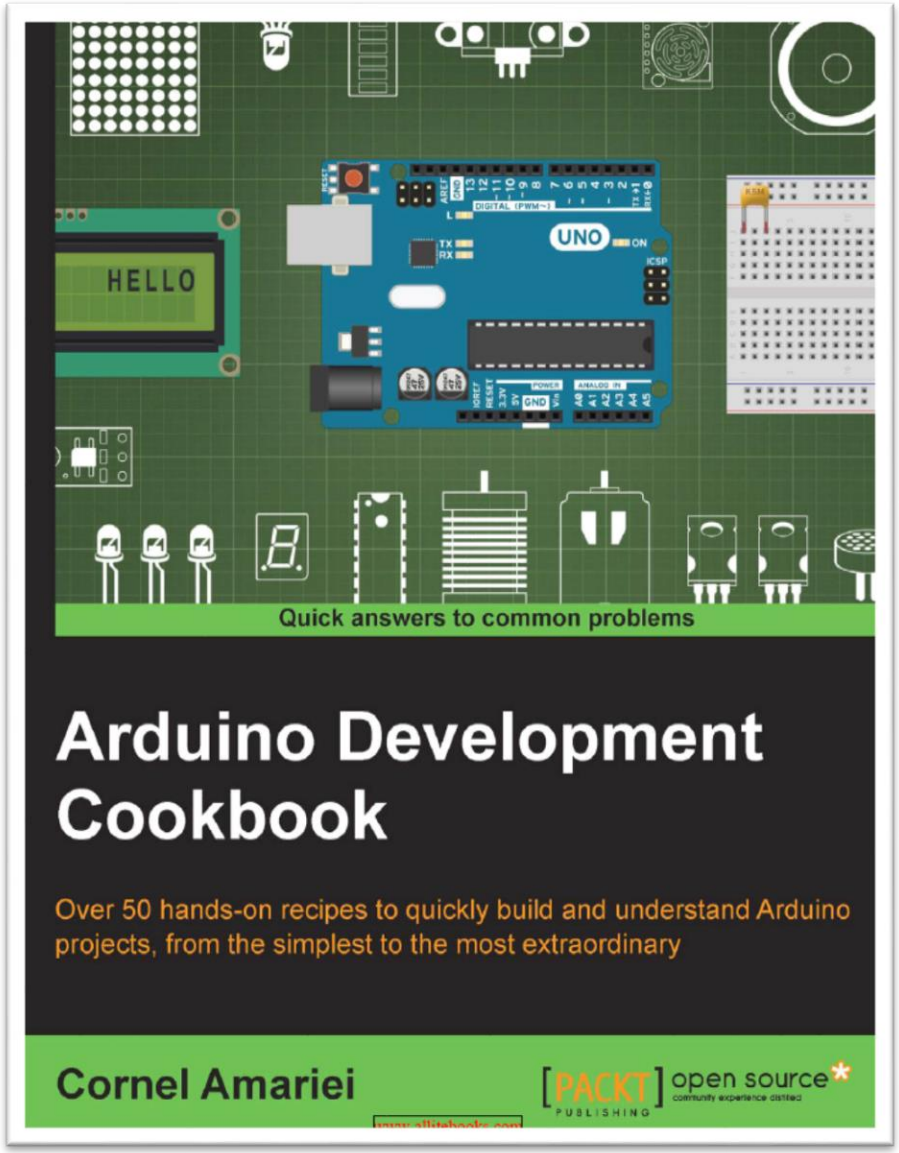
hw-coding

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