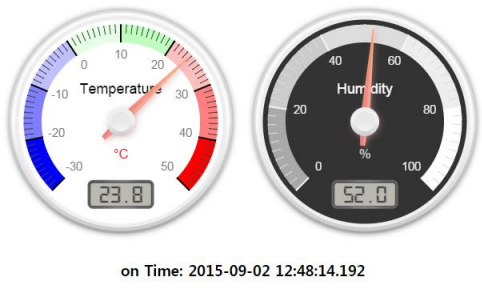
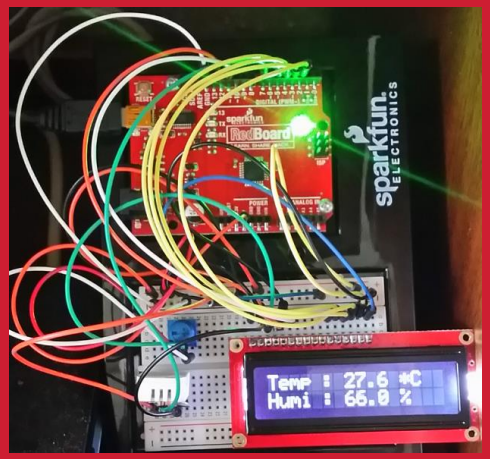
Healthcare-IOT
[wk07]





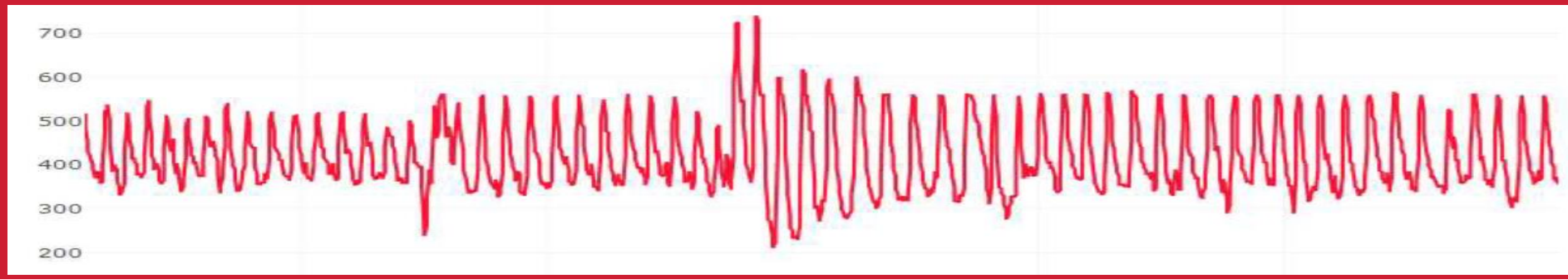
Arduino + Node
II. Multiple Signals

Visualization of Healthcare Signals using Arduino & Node.js

HCit, INJE University

1st semester, 2018

Email : chaos21c@gmail.com



My ID

오전

성명	ID
김민선	HS01
김영걸	HS02
김주란	HS03
김주현	HS04
김태민	HS05
여준하	HS06
이수민	HS07
정민지	HS08
정유현	HS09
정재은	HS10
주하영	HS11
한준영	HS12

오후

성명	ID
신영주	HS21
오가영	HS22
윤민수	HS23
윤진아	HS24
이진영	HS25
임상은	HS26
임재형	HS27
최민영	HS28
황유빈	HS29

주간계획서

주간계획서			
주차	수업방법	수업내용	과제물
1	강의/실습	수업 및 실습 안내 -포터블 소프트웨어 설치	
2	강의/실습	Node.js I - Node.js 코드의 기본 구조 - 기초 Node 서버 및 클라이언트	실습확인
3	강의/실습	Node.js II - Node.js Express 서버	실습확인
4	강의/실습/발표	Arduino I - 아날로그 신호 회로 - LCD를 이용한 센서 신호 모니터링	실습확인
5	강의/실습	Arduino II - 단일 센서 회로와 Node.js 연결 - 다중 센서 회로와 Node.js 연결	실습확인
6	강의/실습	프로젝트1 - 생체 센서 회로와 Node.js 연결 - 생체 신호 소개	프로젝트1
7	강의/실습/발표	IOT 데이터 시각화 I (Plotly.js) - 데이터 및 시계열 차트 - 데이터 스트리밍	실습확인
8	시험	중간고사	
9	강의/실습	IOT 데이터 시각화 II (Plotly.js) - 다중 센서 데이터 시각화 - 다중 센서 데이터 스트리밍	실습확인
10	강의/실습/발표	프로젝트II - 생체 센서 데이터 시각화 - 생체 센서 데이터 스트리밍	프로젝트II
11	강의/실습	IOT 데이터 저장과 처리 - MongoDB 설치 및 Mongo shell - MongoDB와 Node.js 연결 및 데이터 저장	실습확인
12	강의/실습	프로젝트III - MongoDB에 IOT 데이터 저장 및 모니터링 - 생체 센서 데이터 저장 및 시각화	프로젝트III
13	강의/실습	IOT 데이터 마이닝 - 아두이노에서 발생된 데이터 관리 - 데이터마이닝 소개	실습확인
14	강의/실습/발표	프로젝트IV - 생체 센서 데이터 관리 - 생체 센서 데이터 마이닝	프로젝트IV
15	시험	기말고사	

Purpose of HS

주요 수업 목표는 다음과 같다.

1. Node.js를 이용한 아두이노 센서 신호 처리
2. Plotly.js를 이용한 아두이노 센서 신호 시각화
3. MongoDB에 아두이노 센서 데이터 저장 및 처리
4. 생체 센서 발생 신호 처리, 시각화 및 저장
5. 생체 센서 발생 신호 저장 및 분석
6. 생체 신호 장비 활용 능력

[Review]

- [wk06]
  - Arduino + Node.js I. single sensor
  - Complete your project
  - Submit file : HSnn_Rpt05.zip

wk06 : Practice-06 : HSnn_Rpt06.zip

- [Target of this week]
  - Complete your works
  - Save your outcomes and compress 4 outputs

제출파일명 : **HSnn_Rpt05.zip**

- 압축할 파일들

① **HSnn_TMP36_message.png**

② **HSnn_TMP36_IOT_data.png**

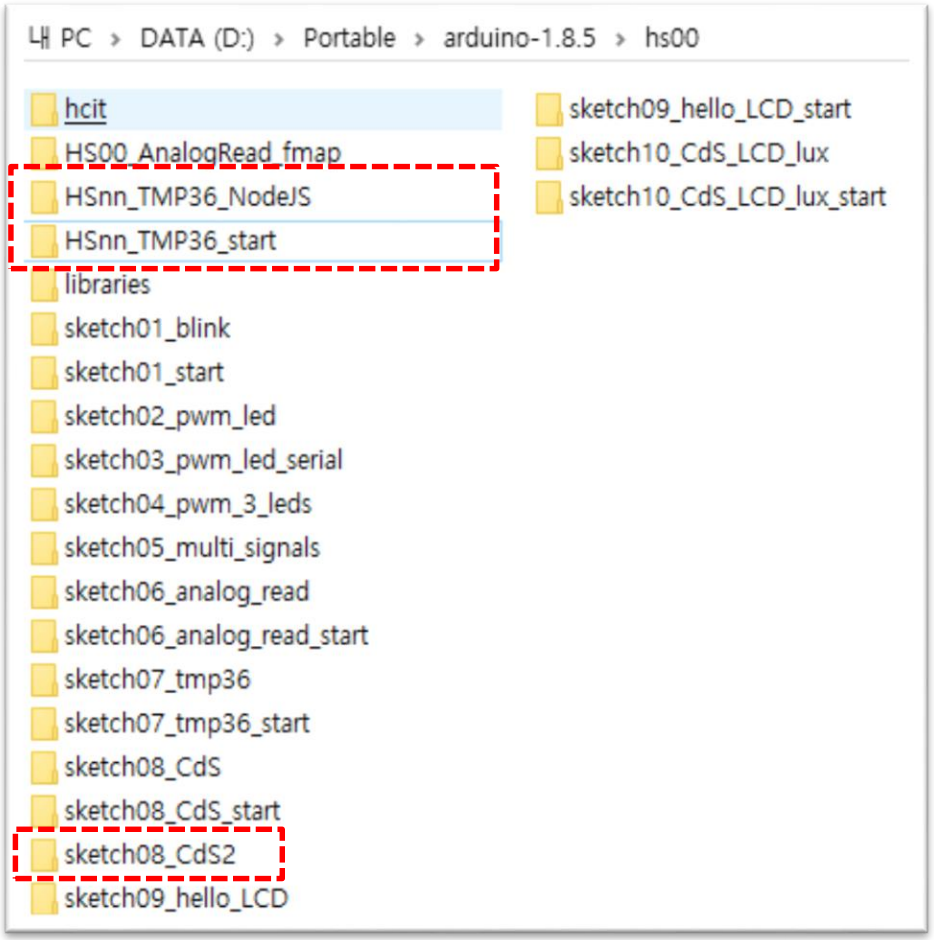
③ **HSnn_cds_IOT_data.png**

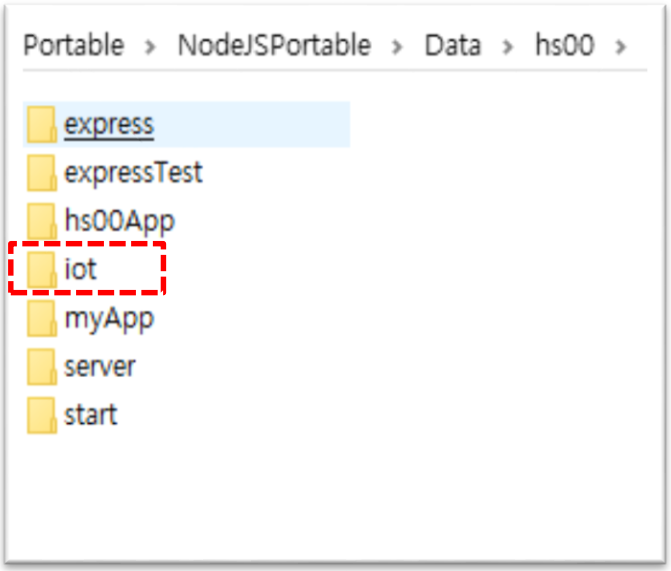
④ **HSnn_multiple_data.png**

Email : chaos21c@gmail.com

[제목 : id, 이름 (수정)]

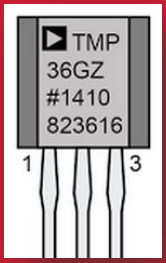
[My working folder – wk06]

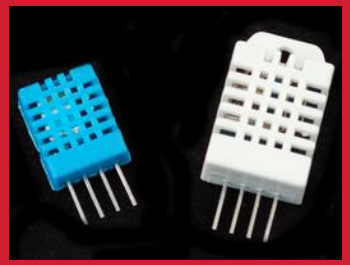


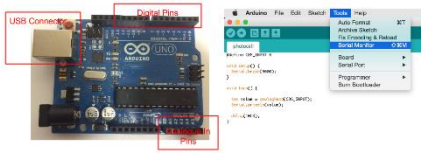


Arduino

+ Node.js







IOT: HSC

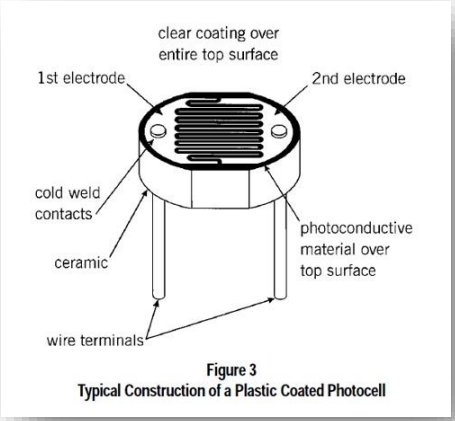
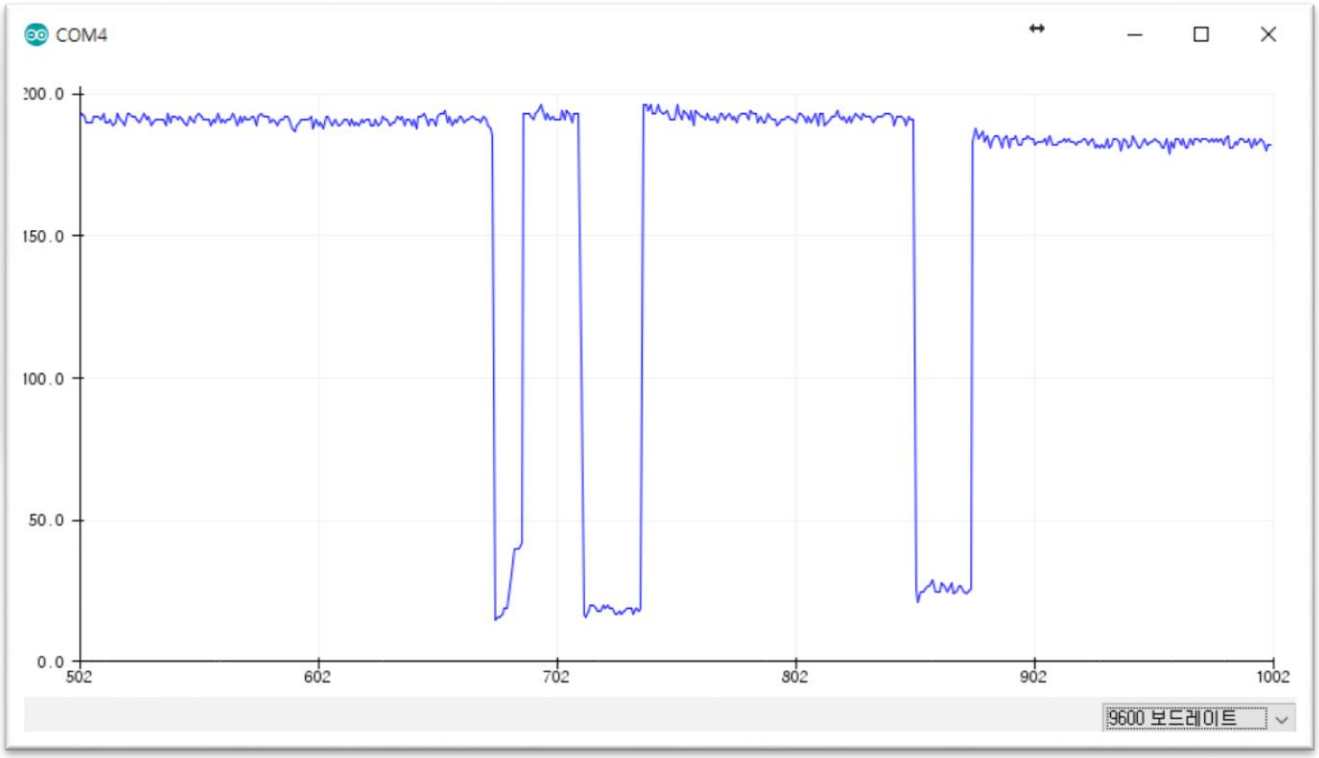
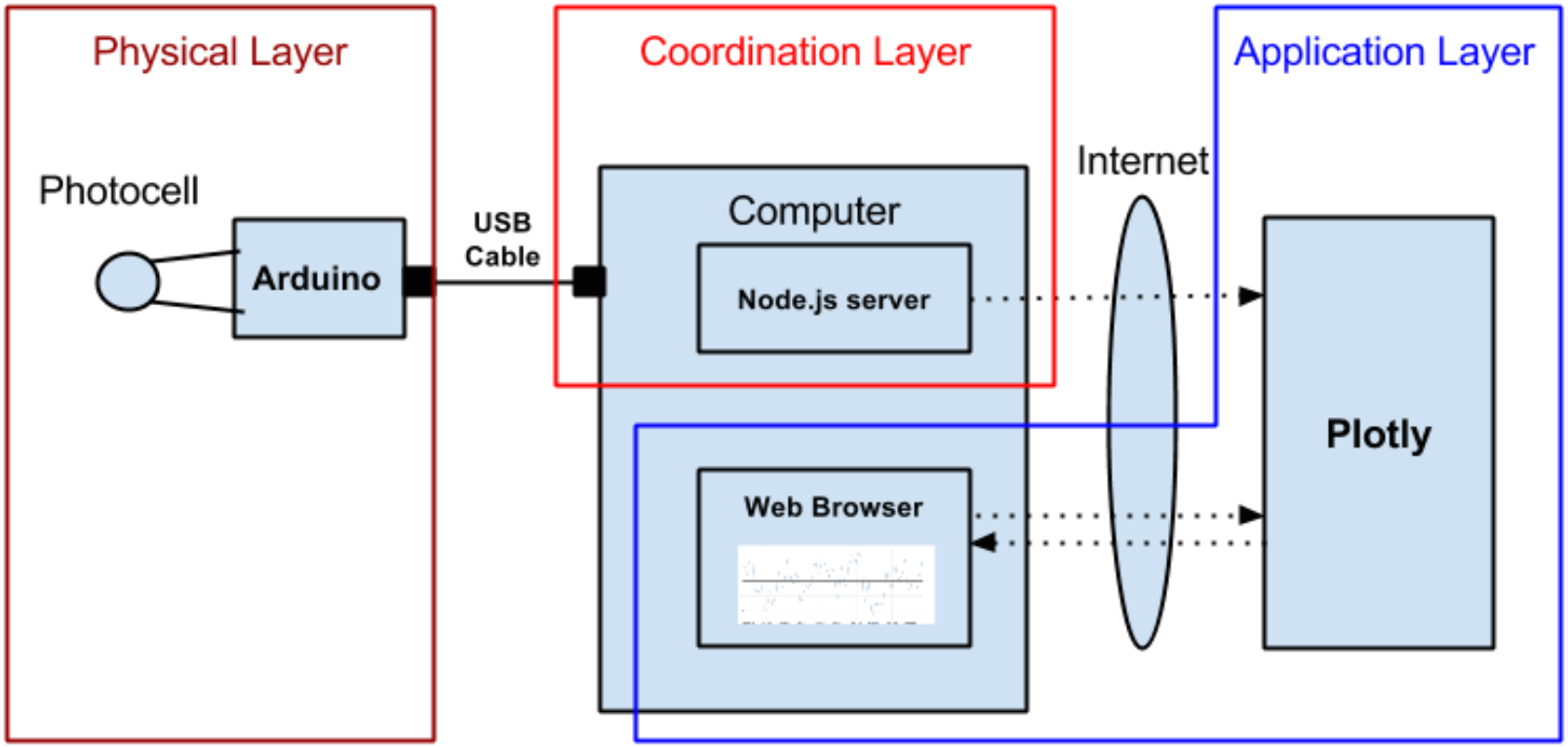


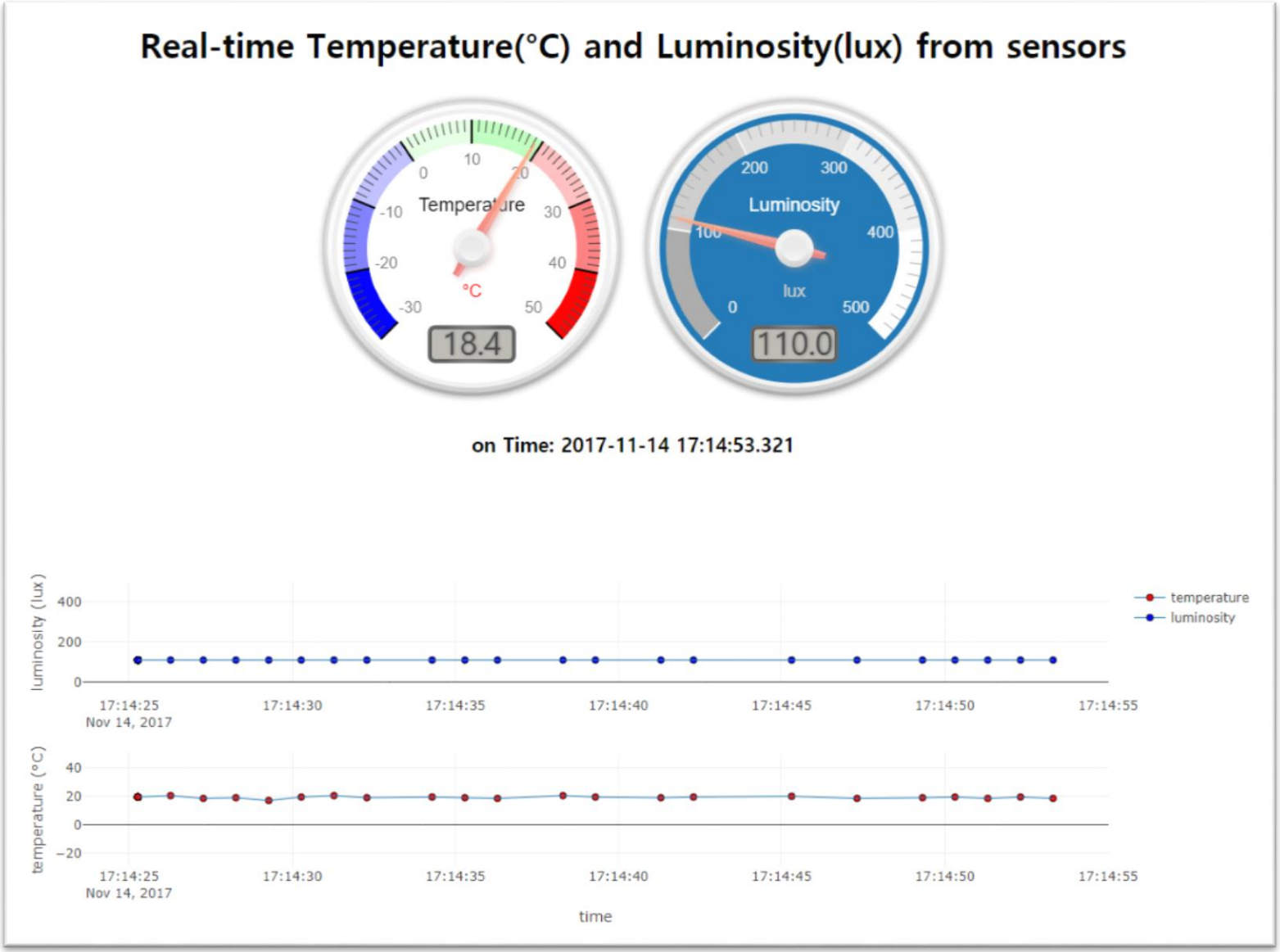
Figure 3
Typical Construction of a Plastic Coated Photocell



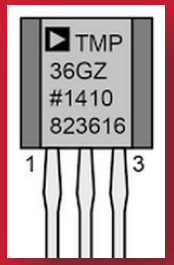
Layout [H S C]

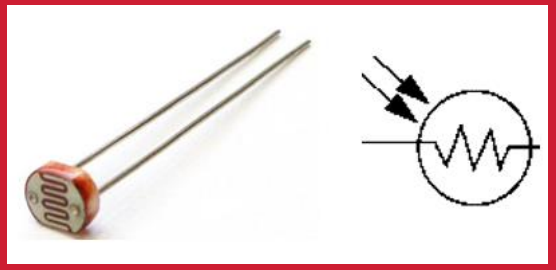

Physical Layer
Photocell
Arduino
USB Cable
Coordination Layer
Computer
Node.js server
Web Browser
Internet
Application Layer
Plotly

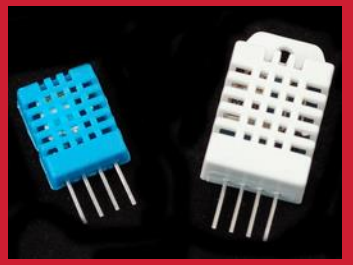
Arduino: node.js + plotly



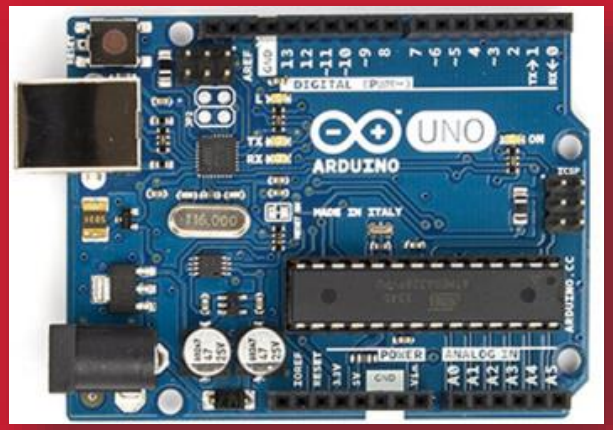
Arduino Sensors





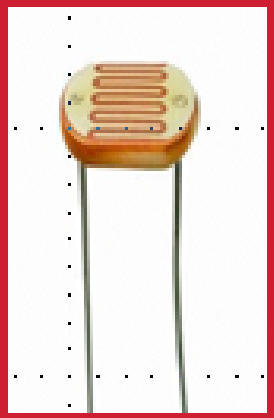


Single sensor: tmp36

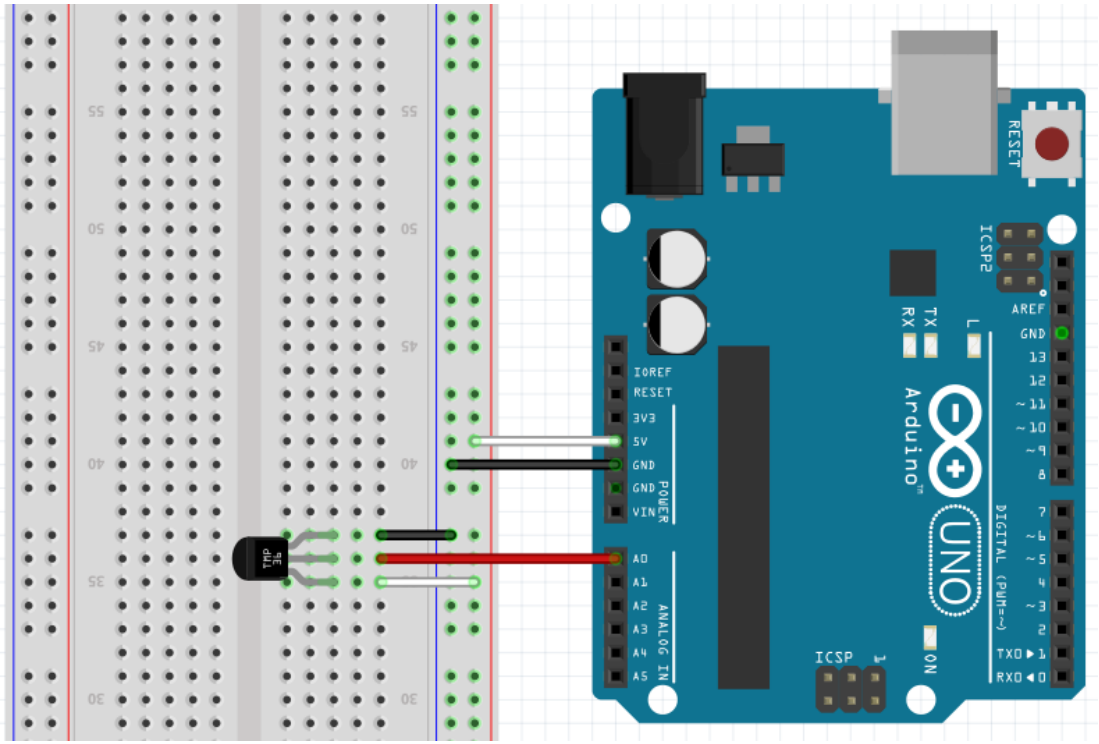


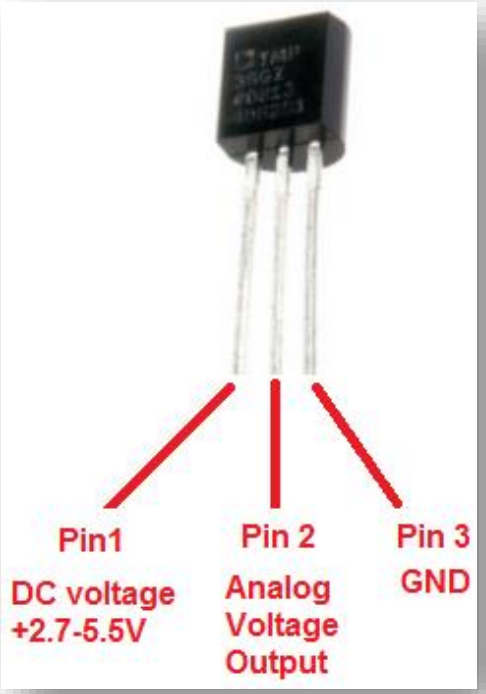
TMP36

Node project

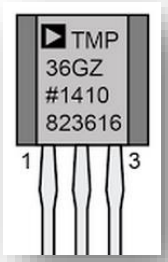


A3.1.1 Temperature sensor [TMP36]





Parts : TMP36



- **Size:** TO-92 package (about 0.2" x 0.2" x 0.2") with three leads
- **Price:** $2.00 at the Adafruit shop
- **Temperature range:** -40°C to 150°C / -40°F to 302°F
- **Output range:** 0.1V (-40°C) to 2.0V (150°C) but accuracy decreases after 125°C
- **Power supply:** 2.7V to 5.5V only, 0.05 mA current draw

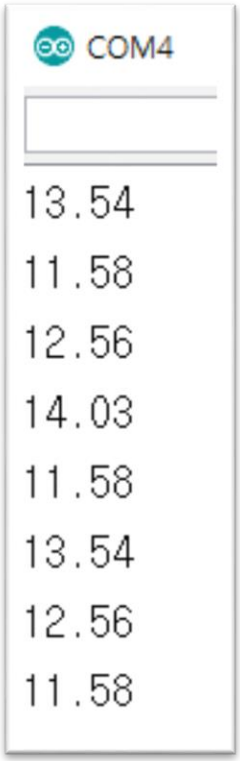
A4.1.10 tmp36 node project (only data)

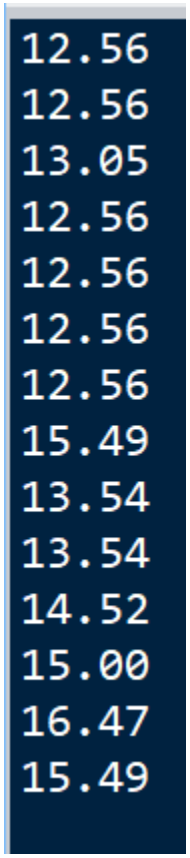
HSnn_TMP36_NodeJS.ino

AA00_TMP36_NodeJS

```
11
12 void loop() {
13   //getting the voltage reading from the temperature sensor
14   int value = analogRead(TEMP_INPUT);
15 //  Serial.print("AA00, value = ");
16 //  Serial.print(value);
17 //  Serial.print(" : ");
18
19   // converting that reading to voltage
20   float voltage = value * 5.0 * 1000;  // in mV
21   voltage /= 1023.0;
22
23   // print out the voltage
24 //   Serial.print(voltage);
25 //   Serial.print(" mV, ");
26
27   // now print out the temperature
28   float temperatureC = (voltage - 500) / 10 ;
29 //   Serial.print(" Temperature, ");
30   Serial.println(temperatureC);
31 //   Serial.println(" degrees C");
32
33   delay(1000);
34 }
```

실행 결과





SB3에서 tmp36_node.js를 ^B로 실행하여 동일한 결과를 얻으시오.

A4.1.11 tmp36 node project (date & data → IOT)

tmp36_node.js

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

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("HS00," + tdata);
        io.sockets.emit('message', tdata);  // send data to all clients
});

// helper function to get a nicely formatted date string for IOT
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;
}
```

IOT data format

시간, 온도

```
[ '2017-11-01 12:46:20.033', '15.49'
[ '2017-11-01 12:46:21.042', '15.49'
[ '2017-11-01 12:46:22.034', '13.54'
[ '2017-11-01 12:46:23.026', '14.03'
[ '2017-11-01 12:46:24.035', '15.00'
[ '2017-11-01 12:46:25.027', '14.52'
[ '2017-11-01 12:46:26.035', '16.47'
[ '2017-11-01 12:46:27.028', '15.98'
[ '2017-11-01 12:46:28.020', '15.98'
[ '2017-11-01 12:46:29.028', '15.49'
[ '2017-11-01 12:46:30.021', '13.05'
[ '2017-11-01 12:46:31.013', '15.49'
[ '2017-11-01 12:46:32.021', '15.00'
```

시간 , 온도

A4.1.12 tmp36 node project (실행 결과)

▸ Sublime Text 3에서 실행

```
AA00,2018-01-14 17:50:22.748,9.14
AA00,2018-01-14 17:50:23.753,10.61
AA00,2018-01-14 17:50:24.753,10.12
AA00,2018-01-14 17:50:25.751,10.61
AA00,2018-01-14 17:50:26.750,10.61
AA00,2018-01-14 17:50:27.750,8.65
```

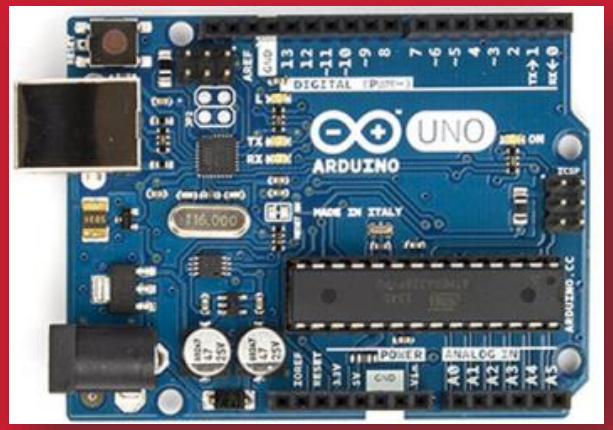
▸ Node cmd에서 실행

```
node tmp36_node
```

```
NodeJS - node  tmp36_node
D:\Portable\NodeJSPortable\Data\aa00\iot\tmp36>node tmp36_node
AA00,2018-01-14 18:07:42.248,5.72
AA00,2018-01-14 18:07:43.252,7.18
AA00,2018-01-14 18:07:44.250,7.18
AA00,2018-01-14 18:07:45.251,7.67
AA00,2018-01-14 18:07:46.248,8.16
```

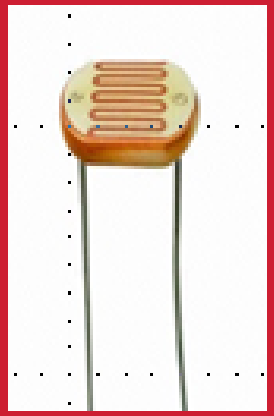
HSnn_tmp36_IOT_data.png 로 저장 (AA00 → HSnn)

Single sensor: CdS



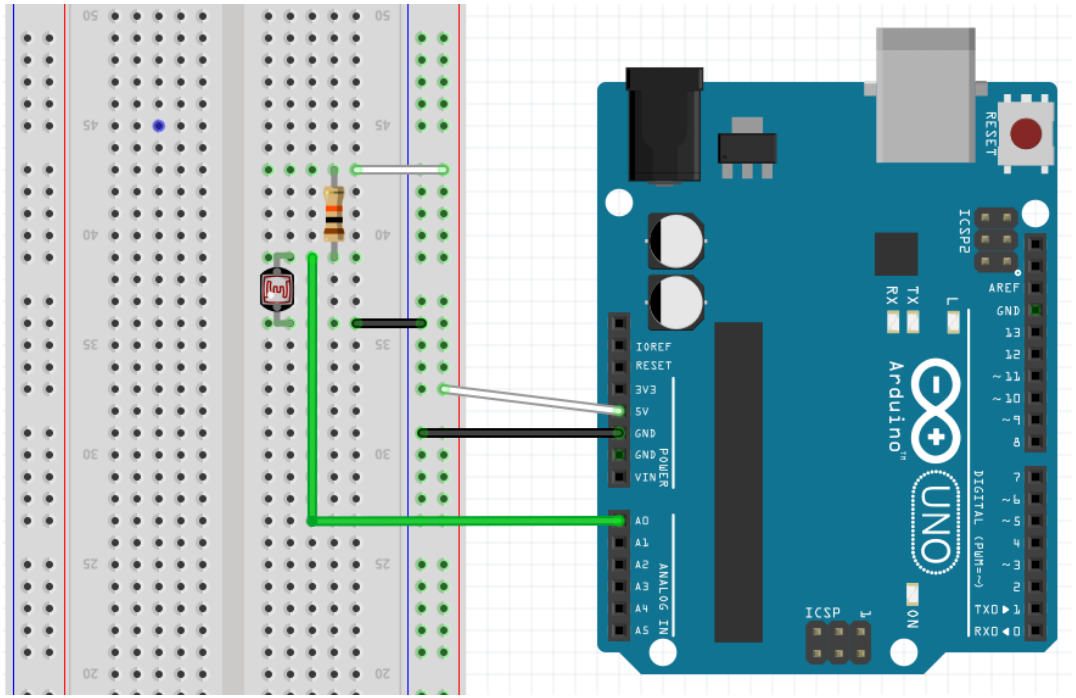
CdS (LDR)

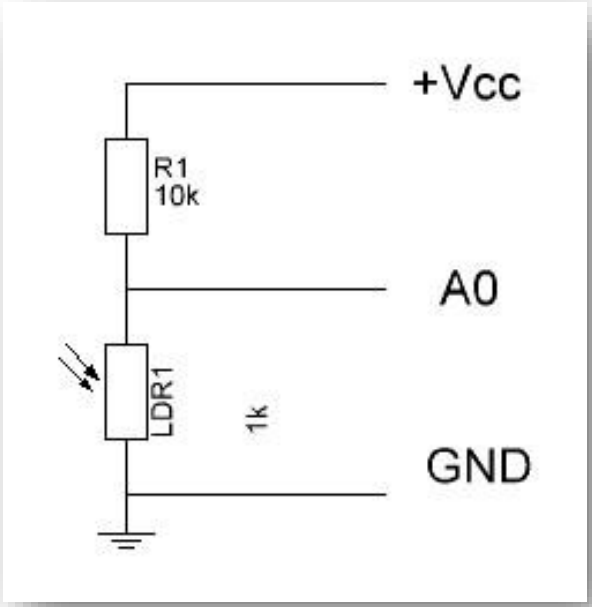
Node project



A3.2.2 Luminosity sensor [Photocell LDR]

CdS 센서 회로

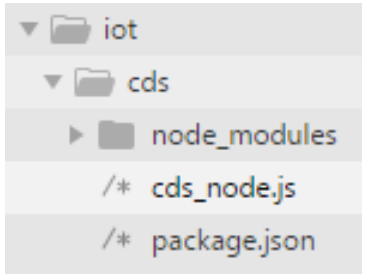




Parts : 20 mm photocell LDR, R (10 kΩ X 1)

광센서에서의 전압 강하 값을 **A0**로 측정

A4.2.4 Luminosity sensor [Photocell LDR]



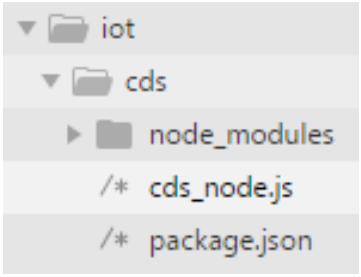
Save tmp36_node.js as cds_node.js

```
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;
}
```

A4.2.5 cds_ node project (실행 결과)

▸ Sublime Text 3에서 실행

```
AA00,2018-01-14 19:12:42.037,86
AA00,2018-01-14 19:12:43.035,36
AA00,2018-01-14 19:12:44.039,54
AA00,2018-01-14 19:12:45.038,175
AA00,2018-01-14 19:12:46.042,175
AA00,2018-01-14 19:12:47.041,174
```



▸ Node cmd에서 실행

node cds_node

NodeJS - node cds_node

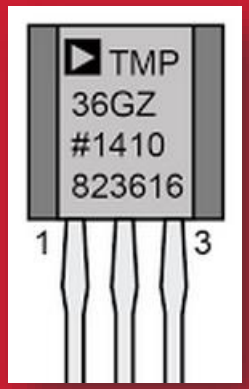
```
D:\Portable\NodeJSPortable\Data\aa00\iot\cds>node cds_node
AA00,2018-01-14 19:15:33.602,176
AA00,2018-01-14 19:15:34.601,45
AA00,2018-01-14 19:15:35.601,35
AA00,2018-01-14 19:15:36.604,33
AA00,2018-01-14 19:15:37.604,175
```

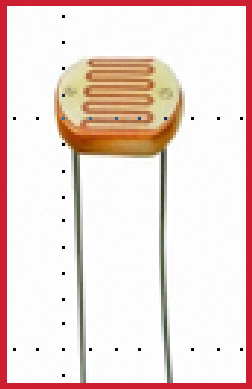
HSnn_cds_IOT_data.png
로 저장

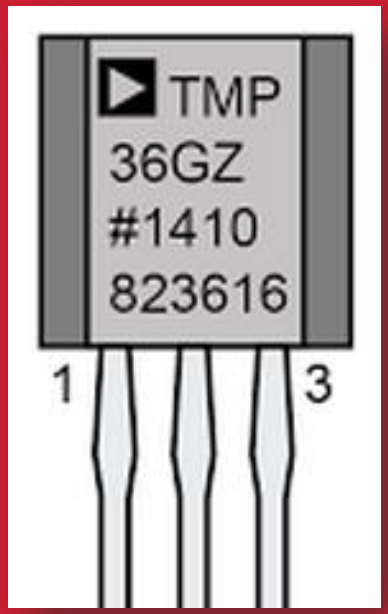
Multiple sensors

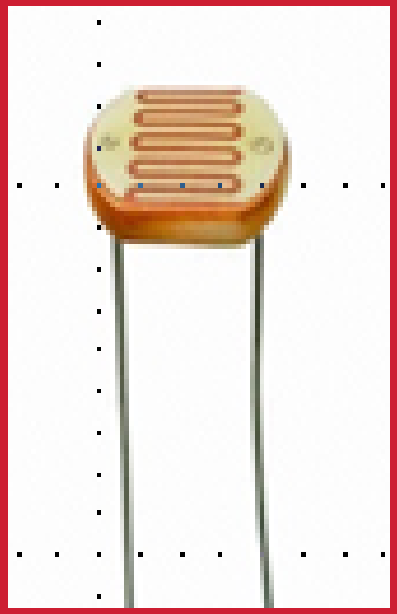
Arduino

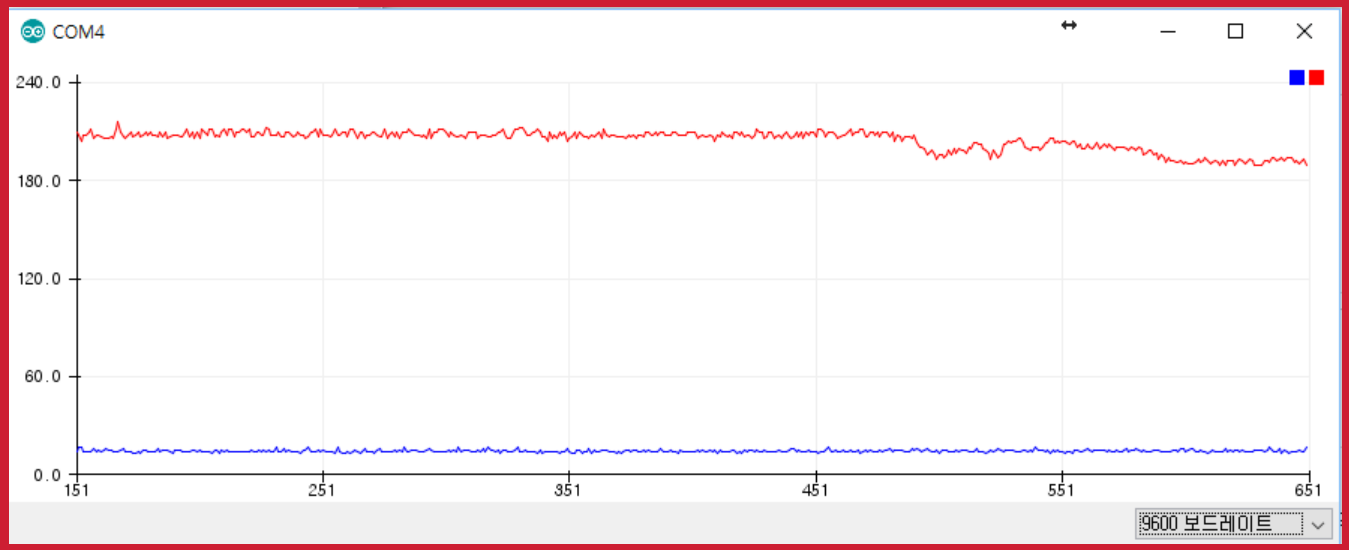
+ Node.js





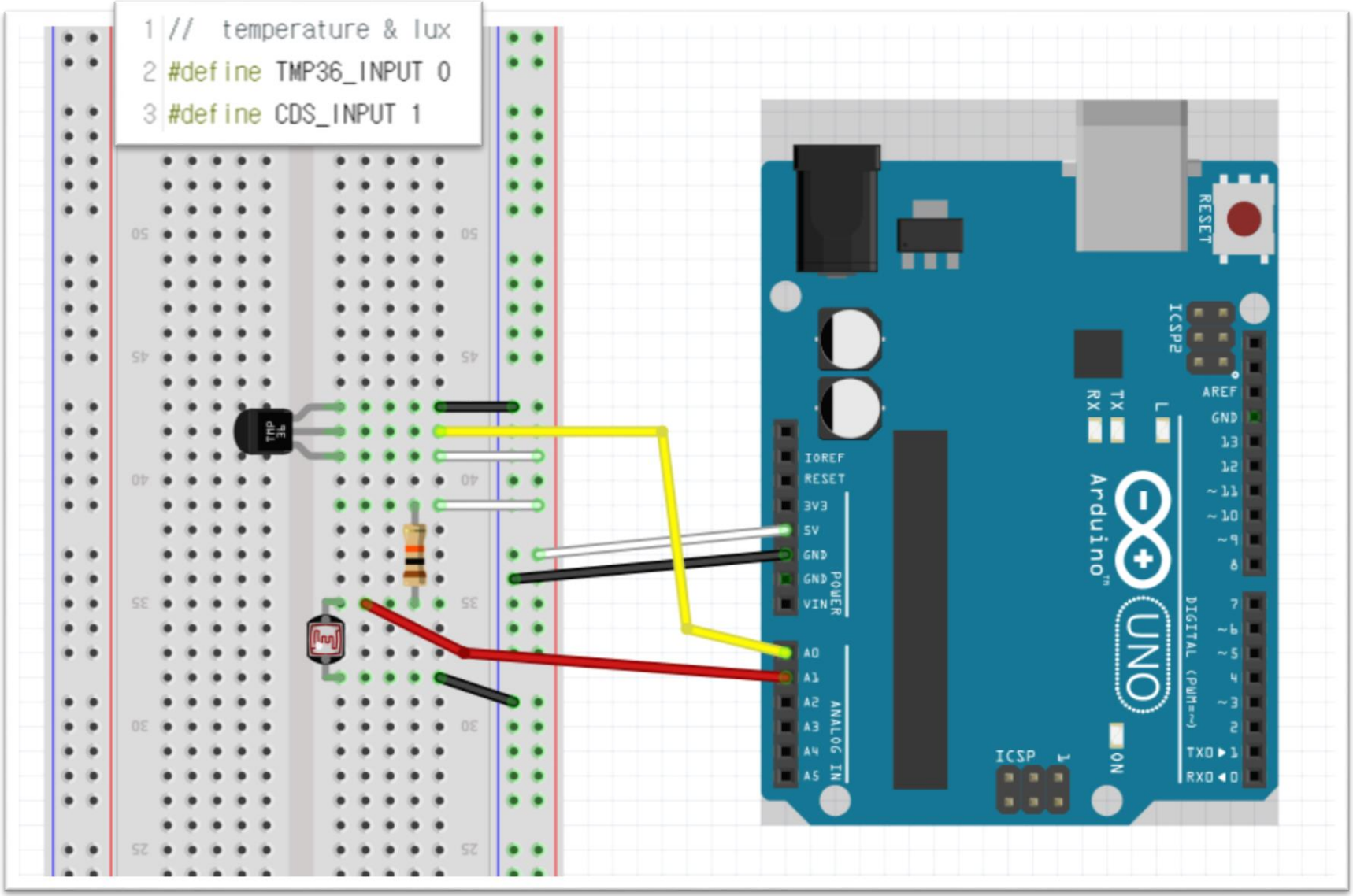

TMP
36GZ
#1410
823616
1
3




COM4
240.0
180.0
120.0
60.0
0.0
151
251
351
451
551
651
9600 보드레이트

A4.3.1 TMP36 + CdS : circuit

```
1 //  temperature & lux
2 #define TMP36_INPUT 0
3 #define CDS_INPUT 1
```



A4.3.2 TMP36 + CdS : code

AAnn_TMP36_CdS §

```
1  //  temperature & lux
2  #define TMP36_INPUT 0
3  #define CDS_INPUT 1
4
5  void setup() {
6    Serial.begin(9600);
7  }
```

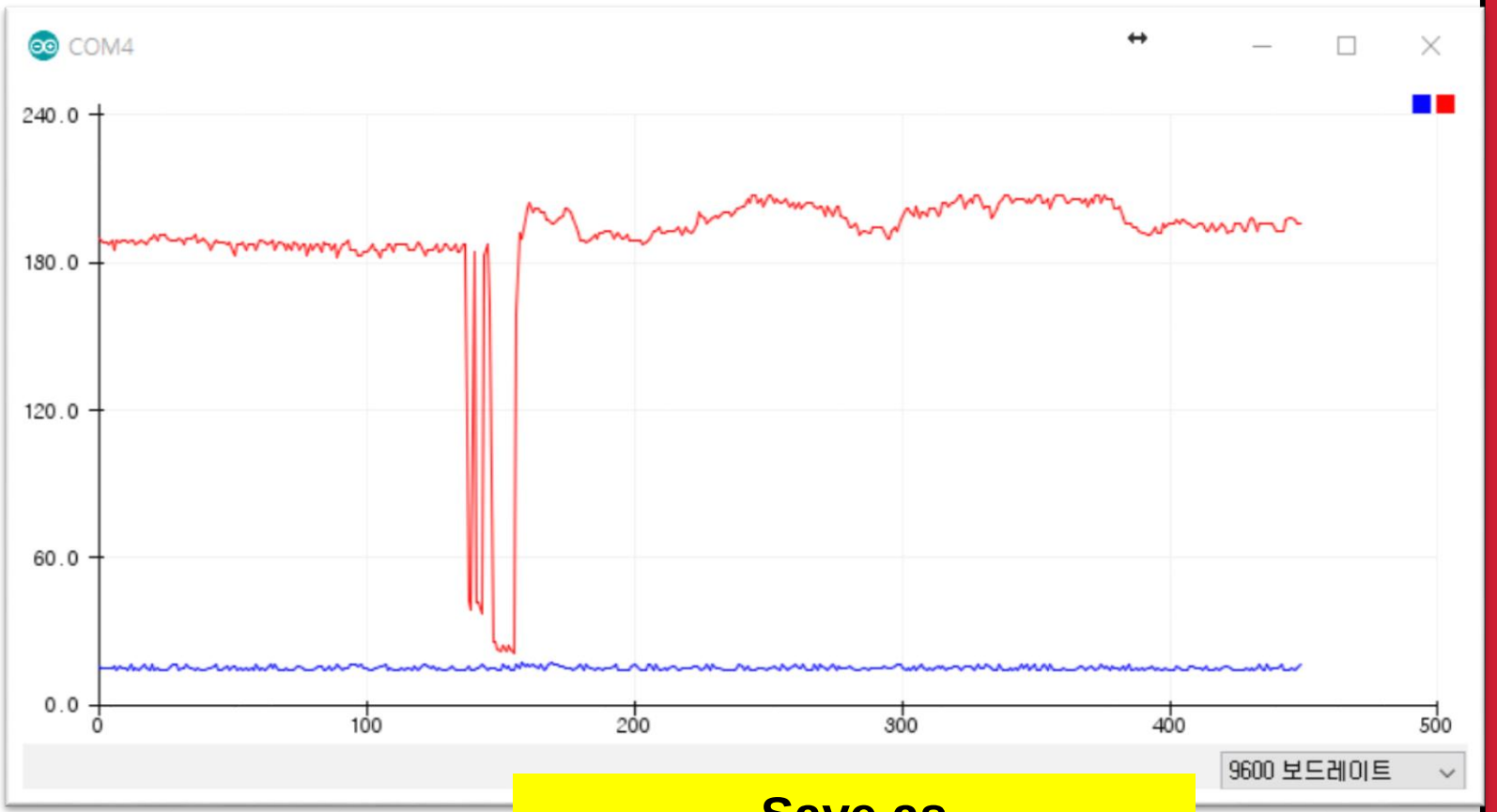
HSnn_tmp36_cds.ino

```
8  void loop() {
9    // Temperature from TMP36
10   int temp_value = analogRead(TMP36_INPUT);
11   // converting that reading to voltage
12   float voltage = temp_value * 5.0 * 1000;  // in mV
13   voltage /= 1023.0;
14   float tempC = (voltage - 500) / 10 ;
15
16   // Lux from CdS (LDR)
17   int cds_value = analogRead(CDS_INPUT);
18   int lux = int(luminosity(cds_value));
19 //  Serial.print("HSnn,");
20   Serial.print(tempC);
21   Serial.print(",");
22   Serial.println(lux);
23
24   delay(1000);
25 }
26
27 //Voltage to Lux
28 double luminosity (int RawADC0){
29   double Vout=RawADC0*5.0/1023.0;  // 5/1023 (Vin = 5 V)
30   int lux=(2500/Vout-500)/10;
31   // lux = 500 / Rldr, Vout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
32   return lux;
33 }
```

A4.3.3 TMP36 + CdS : result

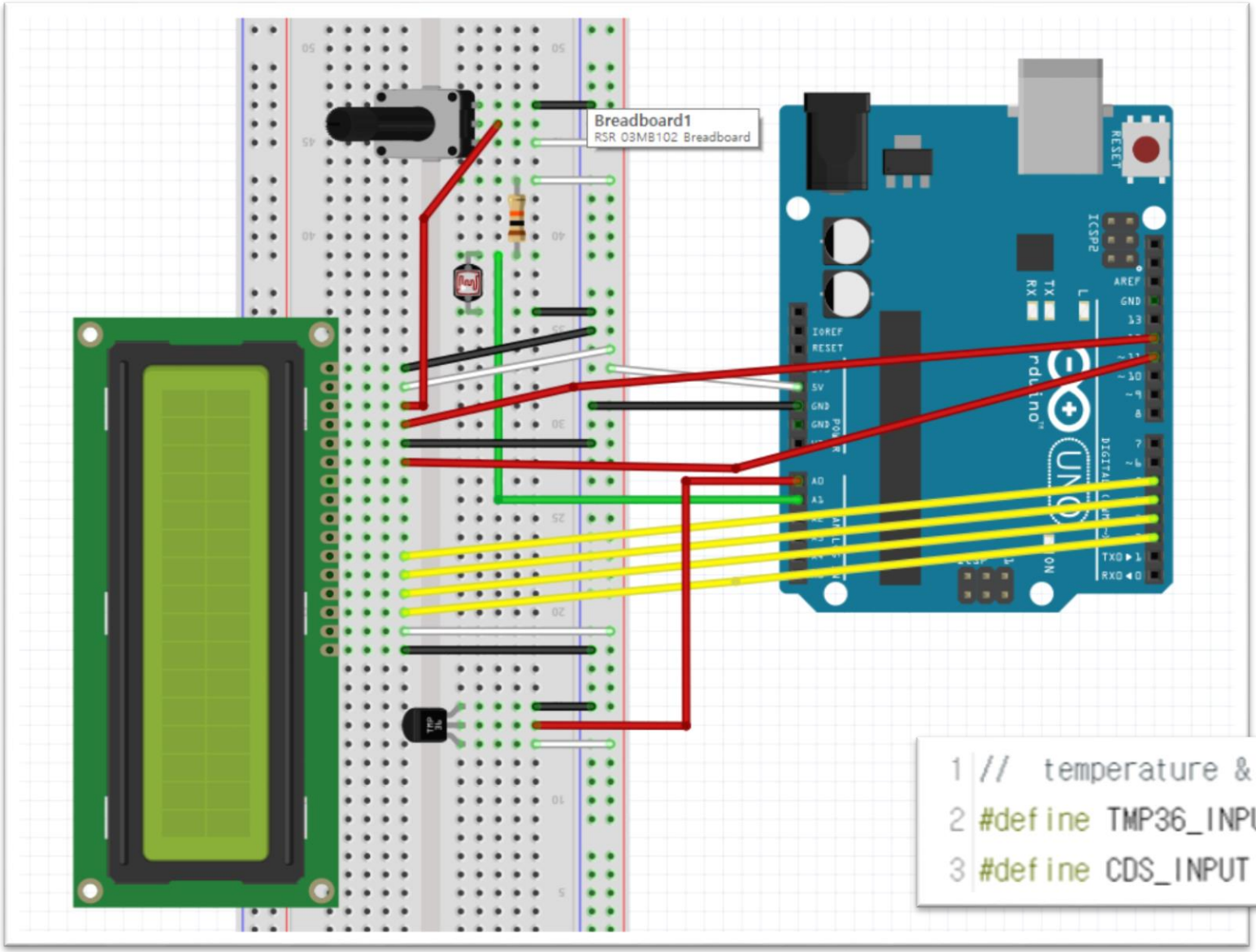
COM4

15.98,192
14.52,194
14.52,193
14.52,193
15.00,180
14.03,18
14.52,17
14.52,16
13.54,15
14.52,191
16.47,188
15.00,188
14.52,190
14.52,190



Save as
HSnn_multiple_data.png

A4.4.1 TMP36 + CdS + LCD : circuit




```
1 //  temperature & lux
2 #define TMP36_INPUT 0
3 #define CDS_INPUT 1
```

A4.4.2 TMP36 + CdS + LCD : code-1

sketch12_CdS_TMP36_LCD

```
/*
 온도, 빛 입력 및 LCD 모니터링
*/

// LCD 라이브러리 설정
#include <LiquidCrystal.h>
// LCD 설정
LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //  rs,en,d4,d5,d6,d7
// 0번 아날로그핀을 TMP36 온도 입력으로 설정한다.
// 1번 아날로그핀을 CdS 조도 입력으로 설정한다.
#define TMP36_INPUT 0
#define CDS_INPUT 1

void setup() {
  Serial.begin(9600);
// 16X2 LCD 모듈 설정하고 백라이트를 켠다.
  lcd.begin(16,2);
// 모든 메세지를 삭제한 뒤
// 숫자를 제외한 부분들을 미리 출력시킨다.
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("HS00,Temp: ");
  lcd.setCursor(0,1);
  lcd.print("Light:  ");
  lcd.setCursor(13,1);
  lcd.print("lux");  //
}

```

A4.4.3 TMP36 + CdS + LCD : code-2

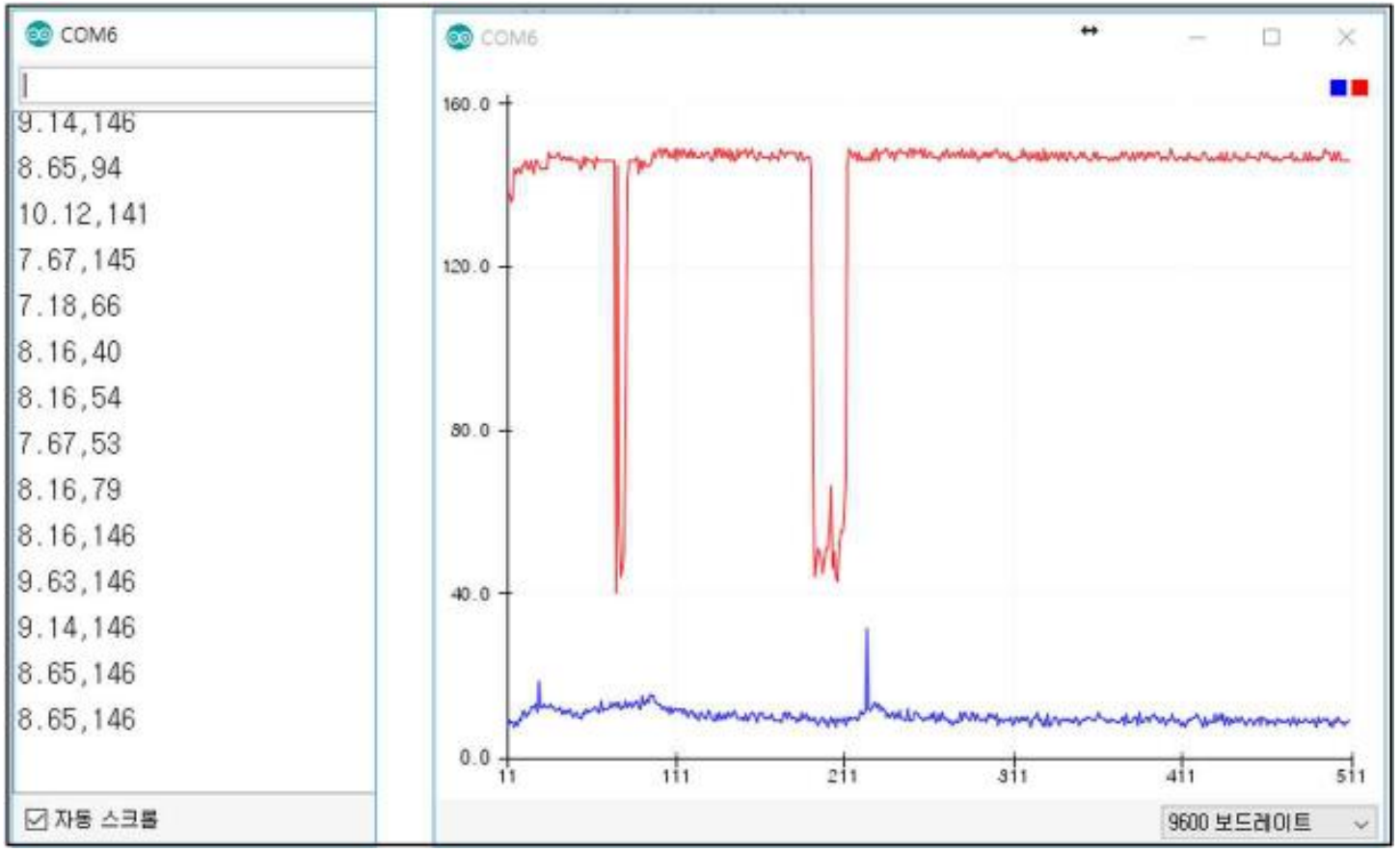
```
29 void loop(){
30   // Temperature from TMP36
31   int temp_value = analogRead(TMP36_INPUT);
32   // converting that reading to voltage
33   float voltage = temp_value * 5.0 * 1000;  // in mV
34   voltage /= 1023.0;
35   float tempC = (voltage - 500) / 10 ;
36
37   // Lux from CdS (LDR)
38   int cds_value = analogRead(CDS_INPUT);
39   int lux = int(luminosity(cds_value));
40
41   // 전에 표시했던 내용을 지운다.
42   lcd.setCursor(12,0);
43   lcd.print("    ");
44   // 온도를 표시한다
45   lcd.setCursor(12,0);
46   lcd.print(tempC);
47   // 전에 표시했던 내용을 지운다.
48   lcd.setCursor(9,1);
49   lcd.print("   ");
50   // 조도를 표시한다
51   lcd.setCursor(9,1);
52   lcd.print(lux);
```

LCD output

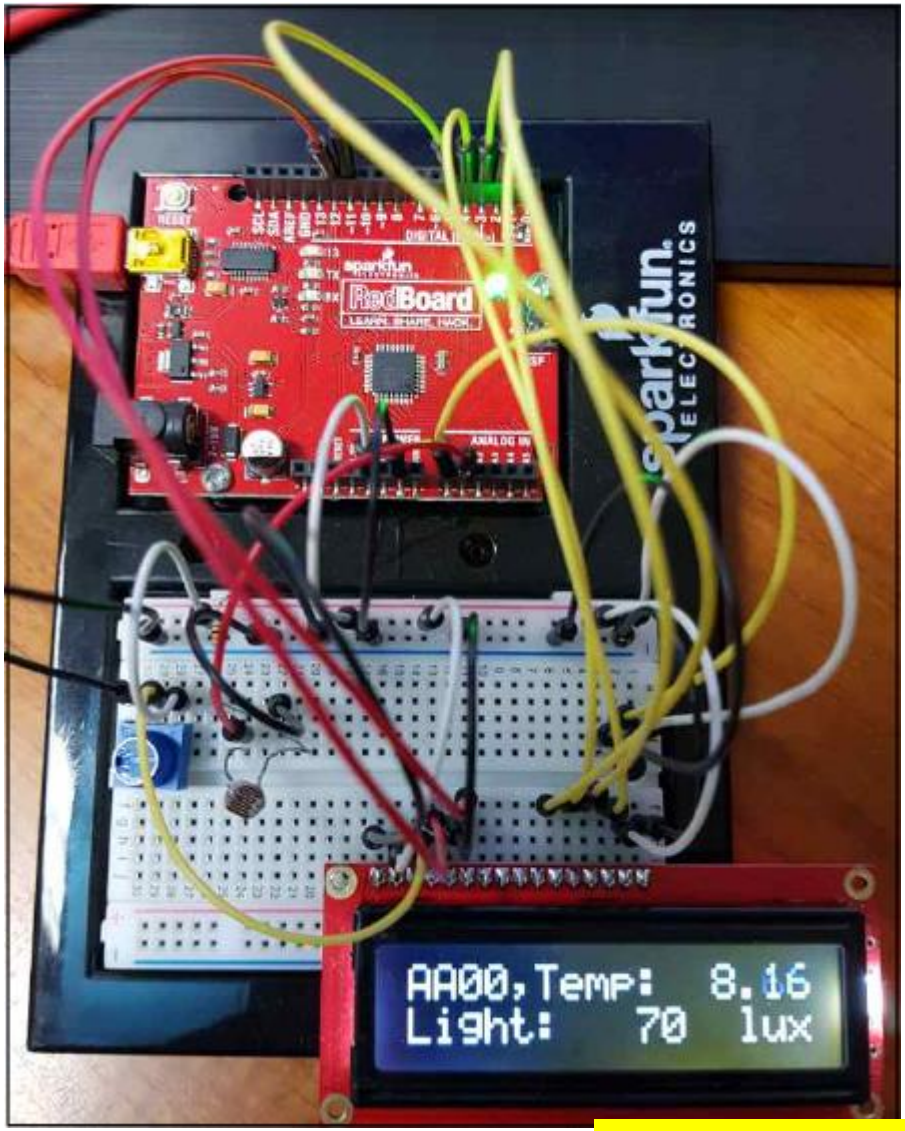
```
54   // Serial output --> 온도,조도
55   Serial.print(tempC);
56   Serial.print(",");
57   Serial.println(lux);
58   delay(1000);
59 }
60
61 //Voltage to Lux
62 double luminosity (int RawADC0){
63   double Vout=RawADC0*5.0/1023.0;  // 5/1023 (Vin = 5 V)
64   double lux=(2500/Vout-500)/10.0;
65   // lux = 500 / Rldr, Vout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
66   return lux;
67 }
```

Serial output

A4.4.4 TMP36 + CdS + LCD : result-1



A4.4.5 TMP36 + CdS + LCD : result-2

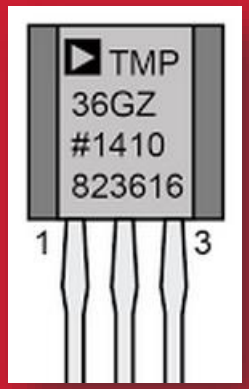


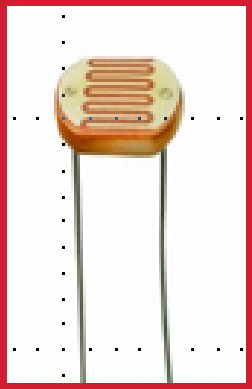
Save as
HSnn_cds_tmp36_lcd.png

Multiple sensors

CdS + TMP36

Node project





A4.5.1 CdS + TMP36 + Node project

1. Make cds_tmp36 node project
   - md cds_tmp36 in iot folder
   - cd cds_tmp36
2. Go to cds_tmp36 subfolder
   - npm init

```
"main":
"cds_tmp36_node.js"
"author": "hsnn"
```

name : cds_tmp36

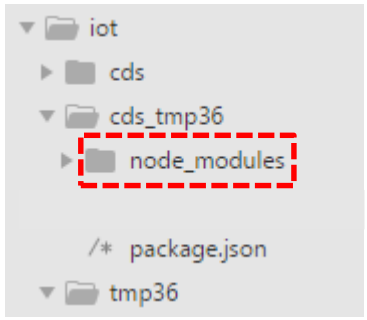
description : cds-tmp36-node project

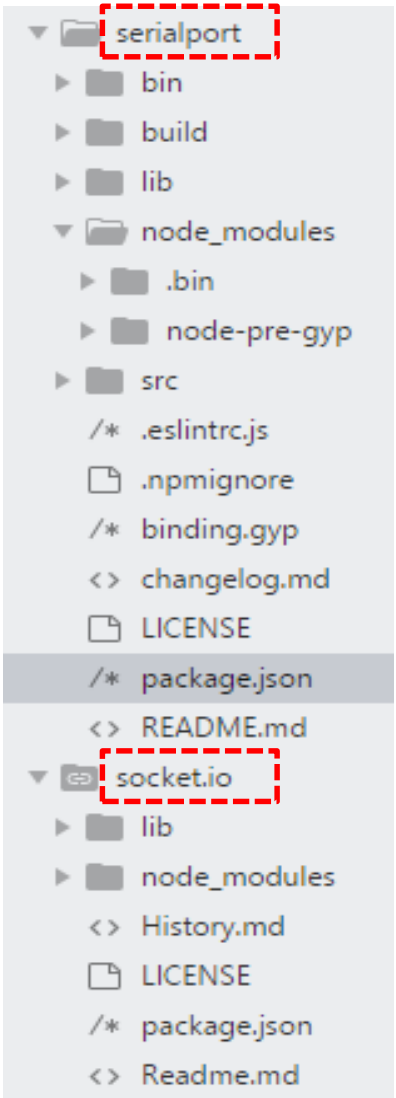
entry point : cds_tmp36_node.js

author : hsnn

A4.5.2 CdS + TMP36 + Node project

1. Make cds_tmp36 node project
   - md cds_tmp36 in iot folder
   - cd cds_tmp36
2. Go to cds_tmp36 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.

A4.5.3 CdS + TMP36 + Node project

1. **Make cds_tmp36 node project**
   - **md cds_tmp36**
   - **cd cds_tmp36**
2. **Go to cds_tmp36 subfolder**
   - **npm init**
   - **npm install –save serialport@4.0.7**
   - **npm install –save socket.io@1.7.3**

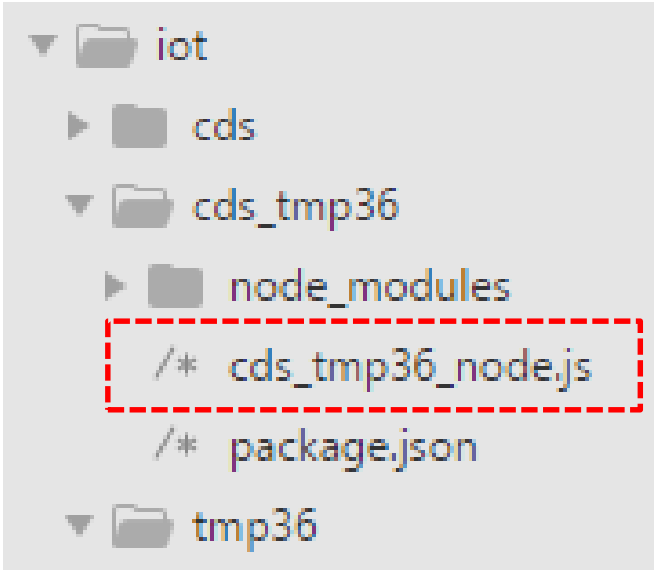
package,json

```
{
  "name": "cds_tmp36",
  "version": "1.0.0",
  "description": "cds-tmp36-node project",
  "main": "cds_tmp36_node.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "cds",
    "tmp36",
    "node"
  ],
  "author": "hs00",
  "license": "MIT",
  "dependencies": {
    "serialport": "^4.0.7",
    "socket.io": "^1.7.3"
  }
}
```

A4.5.4 CdS + TMP36 + Node project

Recycling code:

Save cds_node.js as
cds_tmp36_node.js



A4.5.5.1 CdS + TMP36 + Node project : code-1

cds_tmp36_node.js

cds_tmp36_node.js

```
1  // cds_tmp36_node.js
2
3  var serialport = require('serialport');
4  var portName = 'COM6';  // check your COM port!!
5  var port     =   process.env.PORT || 3000;
6
7  var io = require('socket.io').listen(port);
8
9  // serial port object
10 var sp = new serialport(portName,{
11     baudRate: 9600,   // 9600  38400
12     dataBits: 8,
13     parity: 'none',
14     stopBits: 1,
15     flowControl: false,
16     parser: serialport.parsers.readline('\r\n')
17 });
```

A4.5.5.2 CdS + TMP36 + Node project : code-2

cds_tmp36_node.js – parsing data

```
18 var dStr = '';
19 var readData = '';  // this stores the buffer
20 var temp ='';
21 var lux ='';
22 var mdata =[]; // this array stores date and data from multiple sensors
23 var firstcommaidx = 0;
24
25 sp.on('data', function (data) { // call back when data is received
26     readData = data.toString(); // append data to buffer
27     firstcommaidx = readData.indexOf(',');
28
29     // parsing data into signals
30     if (firstcommaidx > 0) {
31         temp = readData.substring(0, firstcommaidx);
32         lux = readData.substring(firstcommaidx + 1);
33         readData = '';
34
35         dStr = getDateString();
36         mdata[0]=dStr;  // Date
37         mdata[1]=temp;  // temperature data
38         mdata[2]=lux;   // luminosity data
39         console.log("HSnn," + mdata);
40         io.sockets.emit('message', mdata);  // send data to all clients
41
42     } else {  // error
43         console.log(readData);
44     }
45 });
```

Parsing Data

A4.5.5.3 CdS + TMP36 + Node project : code-3

cds_tmp36_node.js

```
32 // helper function to get a nicely formatted date string for IOT
33 function getDateString() {
34     var time = new Date().getTime();
35     // 32400000 is (GMT+9 Korea, GimHae)
36     // for your timezone just multiply +/-GMT by 3600000
37     var datestr = new Date(time +32400000).
38     toISOString().replace(/T/, ' ').replace(/Z/, '');
39     return datestr;
40 }
41
42 io.sockets.on('connection', function (socket) {
43     // If socket.io receives message from the client browser then
44     // this call back will be executed.
45     socket.on('message', function (msg) {
46         console.log(msg);
47     });
48     // If a web browser disconnects from Socket.IO then this callback is called.
49     socket.on('disconnect', function () {
50         console.log('disconnected');
51     });
52 });
```

A4.5.6 CdS + TMP36 + Node project : result

Node cmd 에서 실행

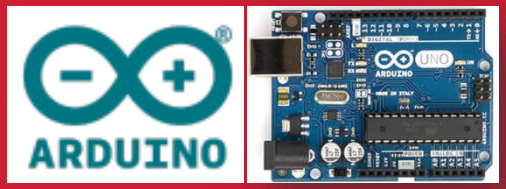
```
node cds_tmp36_node
```

```
NodeJS - node  cds_tmp36_node

D:\Portable\NodeJSPortable\Data\aa00\iot\cds_tmp36>node cds_tmp36_node
AA00 2018-01-15 15:50:06.345 10.12,141
AA00 2018-01-15 15:50:07.337 9.63,141
AA00 2018-01-15 15:50:08.344 9.63,138
AA00 2018-01-15 15:50:09.352 9.63,138
AA00 2018-01-15 15:50:10.359 10.61,139
AA00 2018-01-15 15:50:11.367 10.12,32
```

IOT data format

시간, 온도,조도

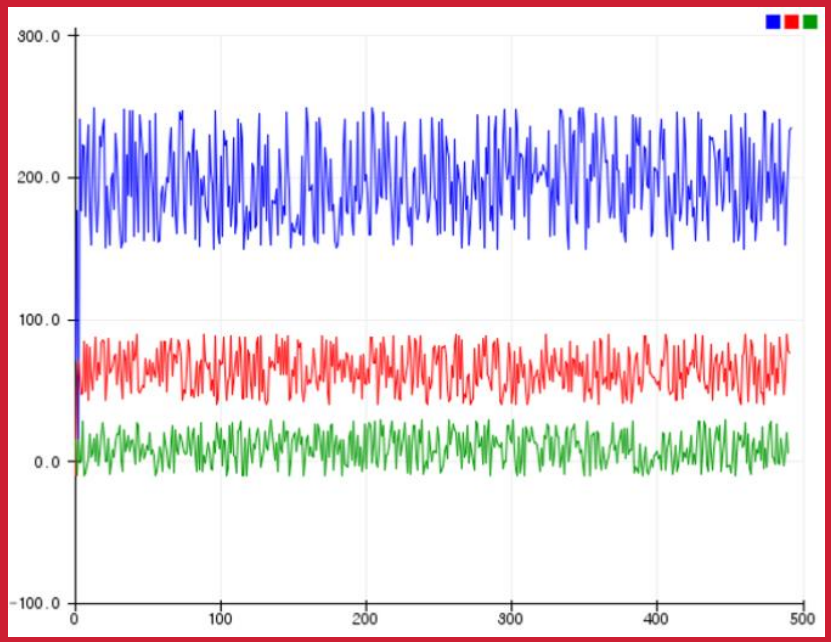
Save as
HSnn_cds_tmp36_IOT.png



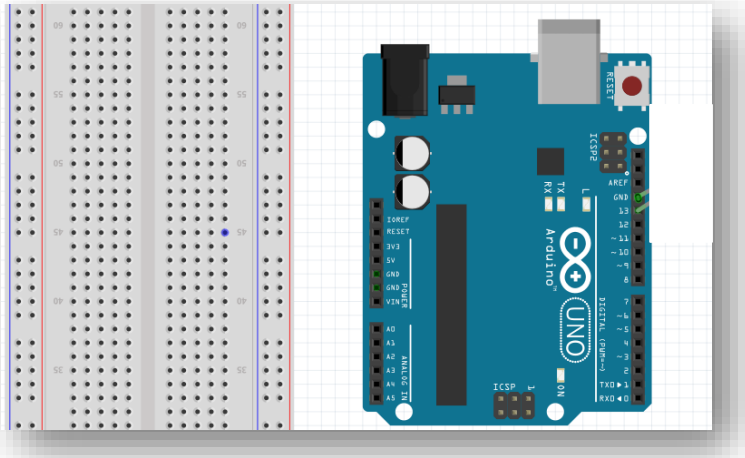
[DIY] Multi-signals

다중신호 시뮬레이션

+ node.js



DIY – 스케치



아두이노에서 **LED**와 저항을 모두 제거하고 **USB**만 컴퓨터와 연결한다.

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

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

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

▶ 스케치 구성

1. 3 개의 신호를 담을 변수를 초기화한다.

2. setup()에서 직렬 통신 속도를 9600 bps 로 설정하고 컴퓨터와 연결한다.

3. loop()에서 마구잡이 수를 세 개 발생시켜서 직렬 통신으로 3 개의 pwm 값을 각각 컴퓨터로 전송한다.

DIY - code

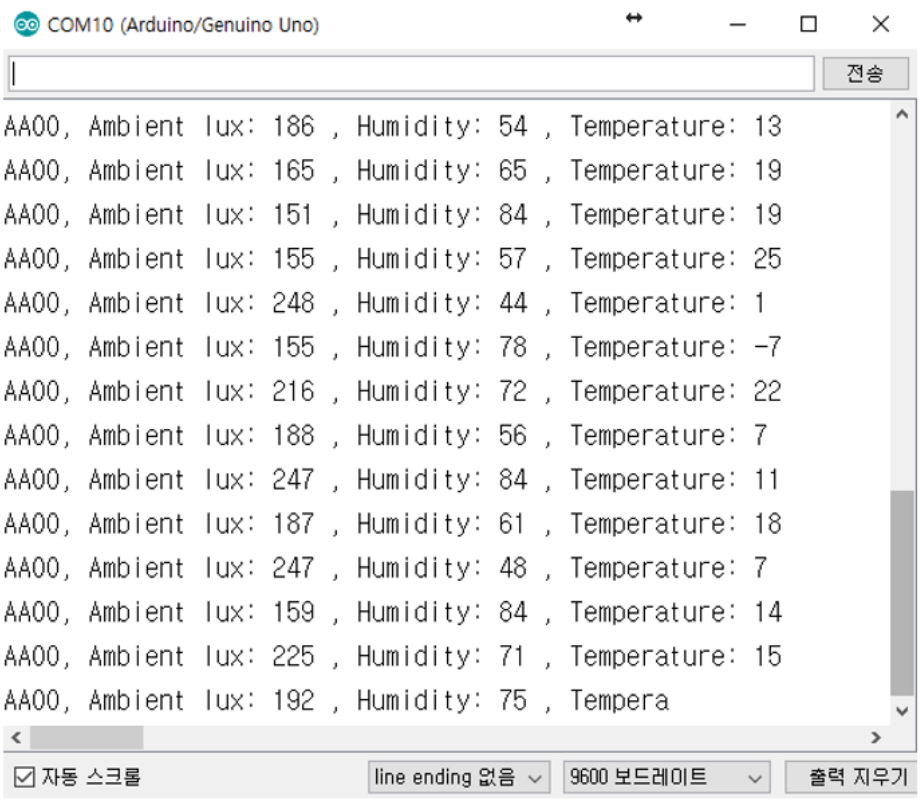
sketch05_multi_signals

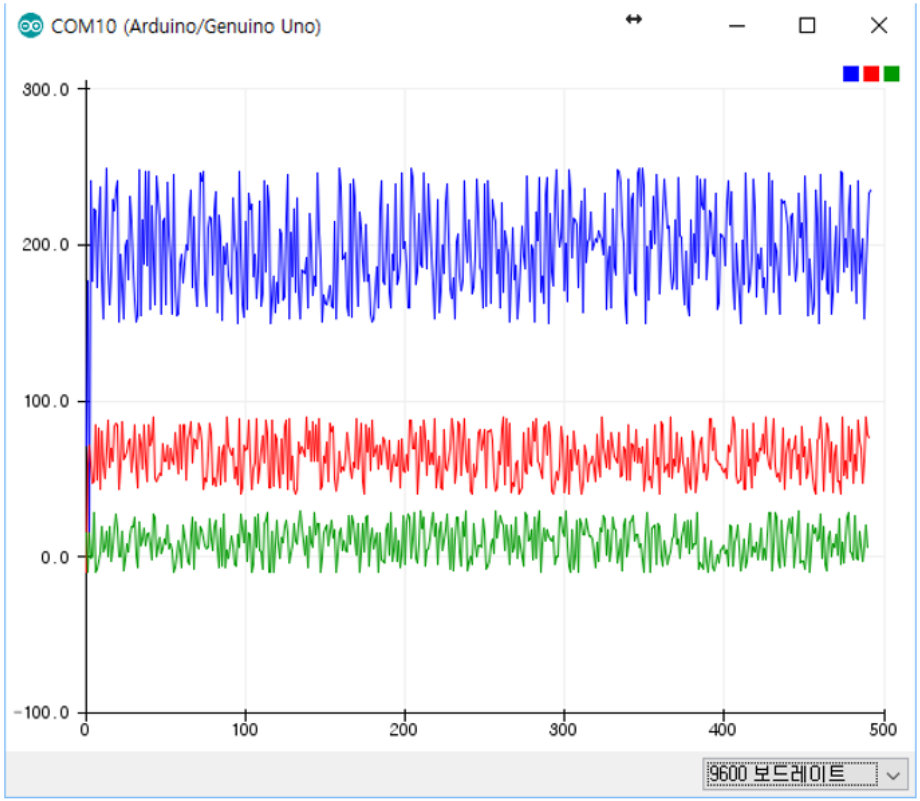
```
/*
  Multi Signals
  Simulation of multiple random signals
*/
// signals
int humi=0;
int temp=0;
int lux=0;

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // Multi signals
  humi = random(40,90);
  temp = random(-10, 30);
  lux = random(150,250);
  Serial.print("HS00, Ambient lux: ");
  Serial.print(lux);
  Serial.print(" , Humidity: ");
  Serial.print(humi);
  Serial.print(" , Temperature: ");
  Serial.println(temp);
  delay(100);        // delay in between reads for stability
}
```

DIY - result

DIY 결과

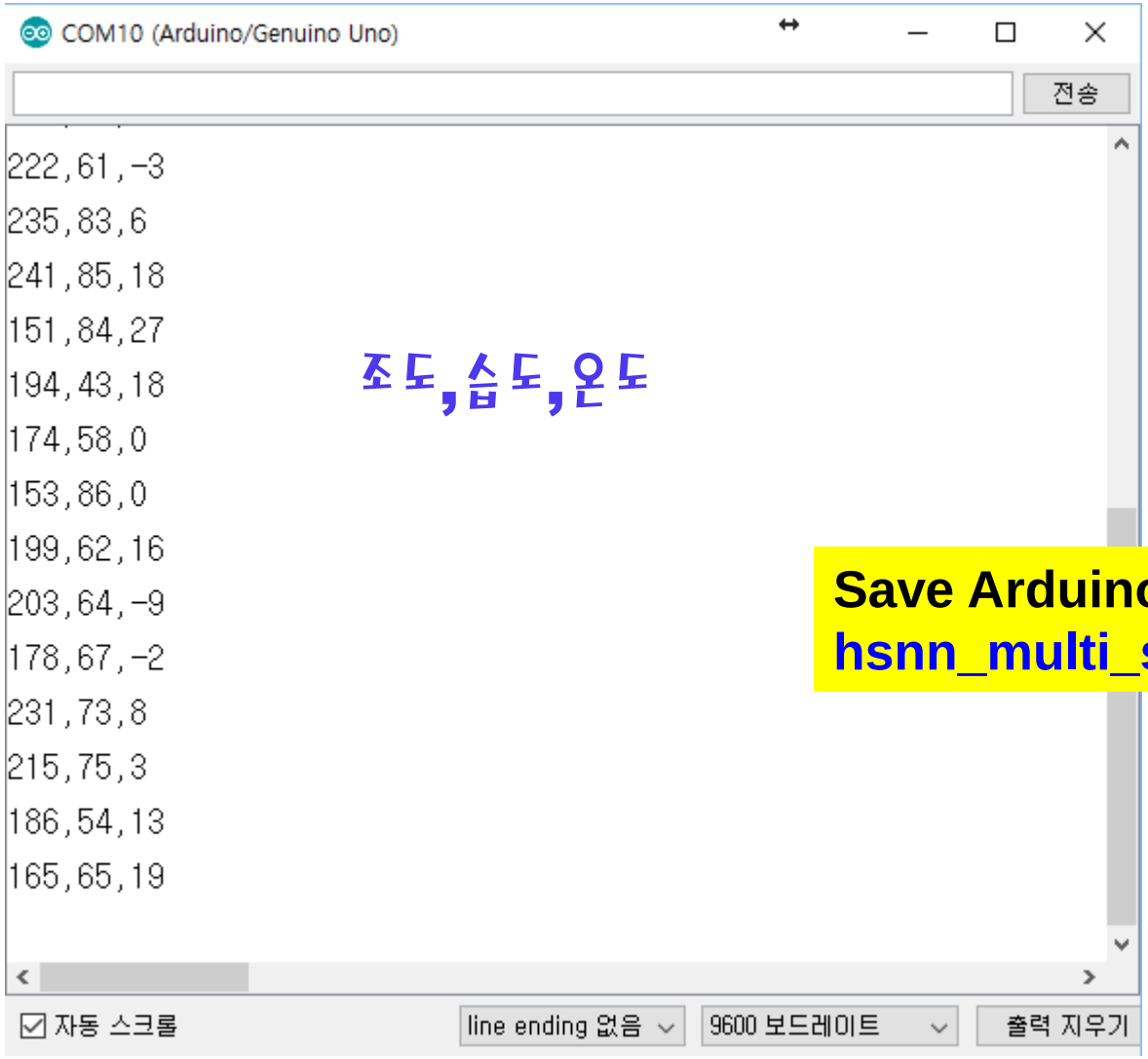
가상적인 세 개의 센서신호 시뮬레이션:조도(위), 습도(중간), 온도(아래).





DIY – New result 1

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



Save Arduino code as hsnn_multi_signals.ino

DIY – New result 2-1

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

[1 단계] Node cmd

1. Make multi_signals node project

- md multi_signals
- cd multi_signals

2. Go to multi_signals subfolder

- npm init

  name : multi_signals

  description : multi-signals-node project

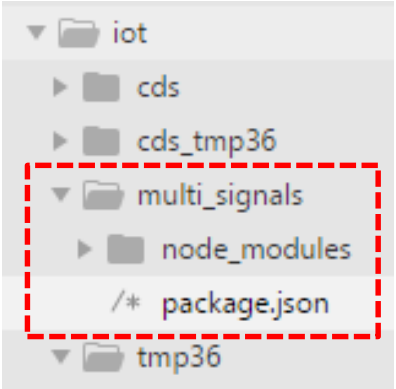
  entry point : hsnn_multi_signals.js

  author : hsnn

3. Install node modules

- npm install –save serialport@4.0.7
- npm install –save socket.io@1.7.3

```
npm
D:\Portable\NodeJSPortable\Data\hs00\iot\multi_signals>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See 'npm help json' for definitive documentation on these fields
and exactly what they do.

Use 'npm install <pkg> --save' afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
name: (multi_signals)
version: (1.0.0)
description: multi-signals-node project
entry point: (index.js) hsnn_multi_signals.js
test command:
git repository:
keywords: multi signals node
author: hsnn
license: (ISC) MIT
```



DIY – New result 2-2

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

Recycling code:

Save cds_tmp36_node.js as
hsnn_multi_signals.js

```
18 var dStr = '';
19 var readData = ''; // this stores the buffer
20 var lux ='';
21 var humi ='';
22 var temp ='';
23 var mdata =[]; // this array stores date and data from multiple sensors
24 var firstcommaidx = 0;
25 var secondcommaidx = 0;
26
27 sp.on('data', function (data) { // call back when data is received
28     readData = data.toString(); // append data to buffer
29     firstcommaidx = readData.indexOf(',');
30     secondcommaidx = readData.indexOf(',', firstcommaidx+1);
```

DIY – New result 2-3

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

Hint:

javascript function : indexOf()

https://www.w3schools.com/jsref/jsref_indexof.asp

Syntax

```
string.indexOf(searchvalue, start)
```

Parameter Values

Parameter	Description
searchvalue	Required. The string to search for
start	Optional. Default 0. At which position to start the search

javascript function : substring()

```
string.substring(start, end)
```

Parameter Values

Parameter	Description
start	Required. The position where to start the extraction. First character is at index 0
end	Optional. The position (up to, but not including) where to end the extraction. If omitted, it extracts the rest of the string

DIY – New result 2-4

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

```
sp.on('data', function (data) { // call back when data is received
    readData = data.toString(); // append data to buffer
    firstcommaidx = readData.indexOf(',');
    secondcommaidx = readData.indexOf(',', firstcommaidx+1);

    // parsing data into signals
        console.log("HSnn," + mdata);
        io.sockets.emit('message', mdata);  // send data to all clients

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

아두이노가 직렬통신으로 전송하는 2 개의 comma (,)로 구분된

조도,습도,온도 데이터 메시지를 **parsing**하여 **mdata** 배열에 담는 코드를 완성하시오.

substring() 함수에서 firstcommaidx, secondcommaidx를 잘 이용하시오.

Save this code as hsnn_multi_signals_code.png

DIY – New result 2-5

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

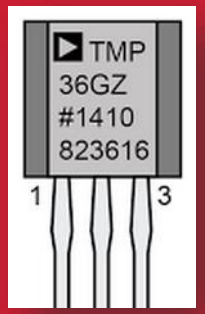
NodeJS - node hsnn_multi_signals

```
D:\Portable\NodeJSPortable\Data\hs00\iot\multi_signals>node hsnn_multi_signals
HSnn,2018-04-18 15:15:57.409,223,47,-1
HSnn,2018-04-18 15:15:57.907,222,48,0
HSnn,2018-04-18 15:15:58.410,173,84,28
HSnn,2018-04-18 15:15:58.912,215,49,-10
HSnn,2018-04-18 15:15:59.410,237,82,-8
HSnn,2018-04-18 15:15:59.909,179,43,-3
HSnn,2018-04-18 15:16:00.410,153,80,2
HSnn,2018-04-18 15:16:00.913,207,59,19
HSnn,2018-04-18 15:16:01.413,249,50,3
HSnn,2018-04-18 15:16:01.913,185,68,6
HSnn,2018-04-18 15:16:02.413,162,87,16
HSnn,2018-04-18 15:16:02.913,183,57,0
HSnn,2018-04-18 15:16:03.416,229,69,19
HSnn,2018-04-18 15:16:03.916,222,61,-3
HSnn,2018-04-18 15:16:04.415,235,83,6
HSnn,2018-04-18 15:16:04.914,241,85,18
HSnn,2018-04-18 15:16:05.415,151,84,27
HSnn,2018-04-18 15:16:05.918,194,43,18
HSnn,2018-04-18 15:16:06.417,174,58,0
HSnn,2018-04-18 15:16:06.918,153,86,0
HSnn,2018-04-18 15:16:07.417,199,62,16
```

ID,시간,조도,습도,온도

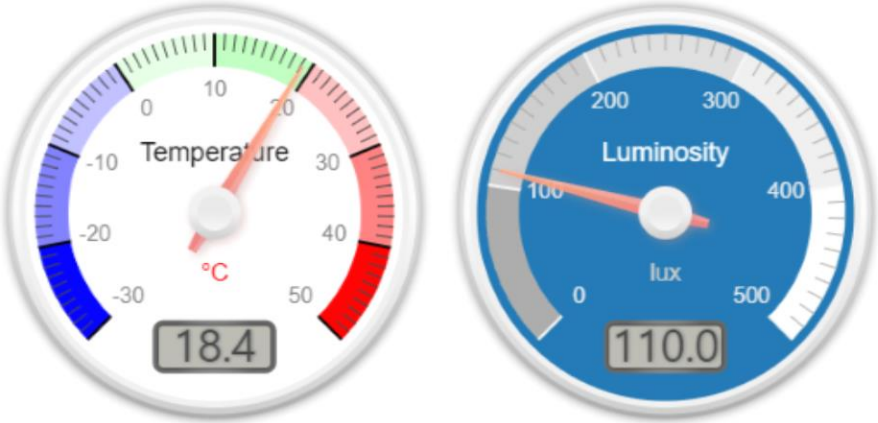
Save this result as hsnn_multi_signals_node .png

Next week

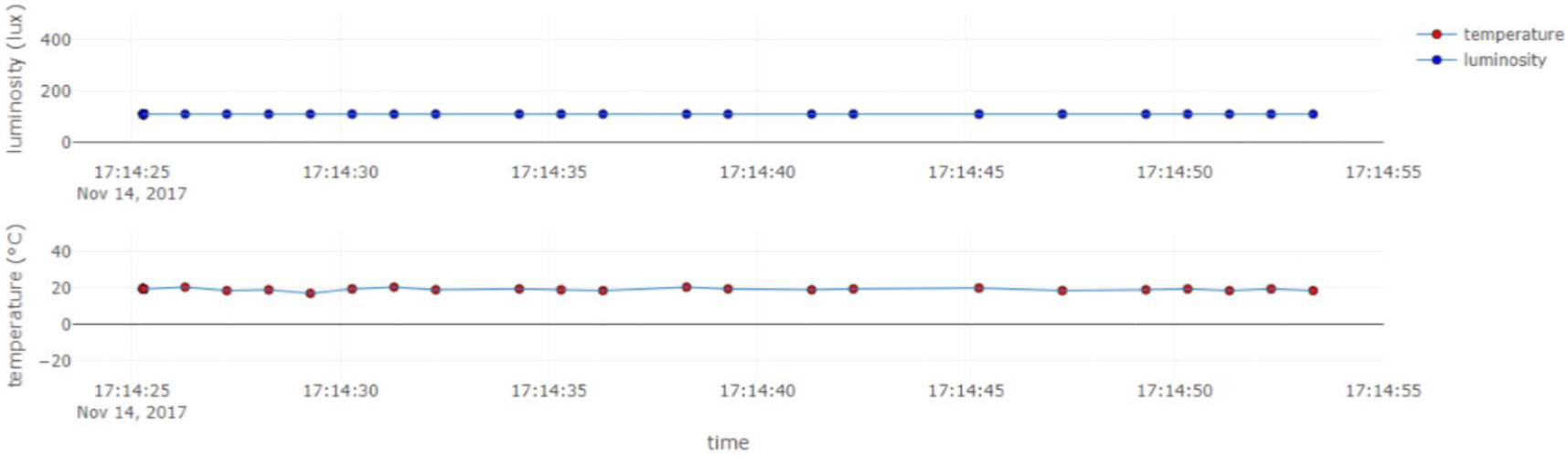


Data visualization using ploy.ly

Real-time Temperature(°C) and Luminosity(lux) from sensors



on Time: 2017-11-14 17:14:53.321



[Practice]

- [wk06]
  - Arduino + Node.js I. single sensor
  - Complete your project
  - Submit file : HSnn_Rpt05.zip

wk07 : Practice-mid : HSnn_Rpt06.zip

- [Target of this week – 10 points]
  - Complete your works
  - Save your outcomes and compress 5 outputs

제출파일명 : **HSnn_Rpt06.zip**

- 압축할 파일들

① **HSnn_cds_tmp36_lcd.png**

② **HSnn_cds_tmp36_IOT.png**

③ **HSnn_multi_signals.ino**

④ **hsnn_multi_signals_code.png**

⑤ **HSnn_multi_signals_node.png**

Email : chaos21c@gmail.com

[제목 : id, 이름 (수정)]

중간고사
Arduino & Node

[1] 과목명 : 헬스케어신호처리개론 (헬스케어IT 학과 2학년)
[2] 4월 26일(목), 5교시: 오후 1시~2시 (장소 : F1002)
[3] 필기 시험 (20점)
- Arduino & Node Coding에서 20문제 (선다형/단답형/서술형)
- 수업 시간에 배운 Code의 이해와 활용 능력 평가

**** 잘 준비해서 웃기를 ... ^_^

Lecture materials

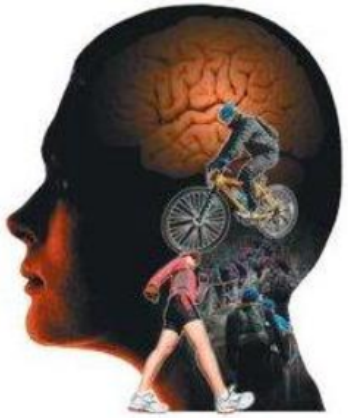
References & good sites

- ✓ 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

Github.com/Redwoods/healthcare-signal-iot

Redwoods / Lec

Unwatch 1

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

All lectures by Redwoods in Inje University

Add topics

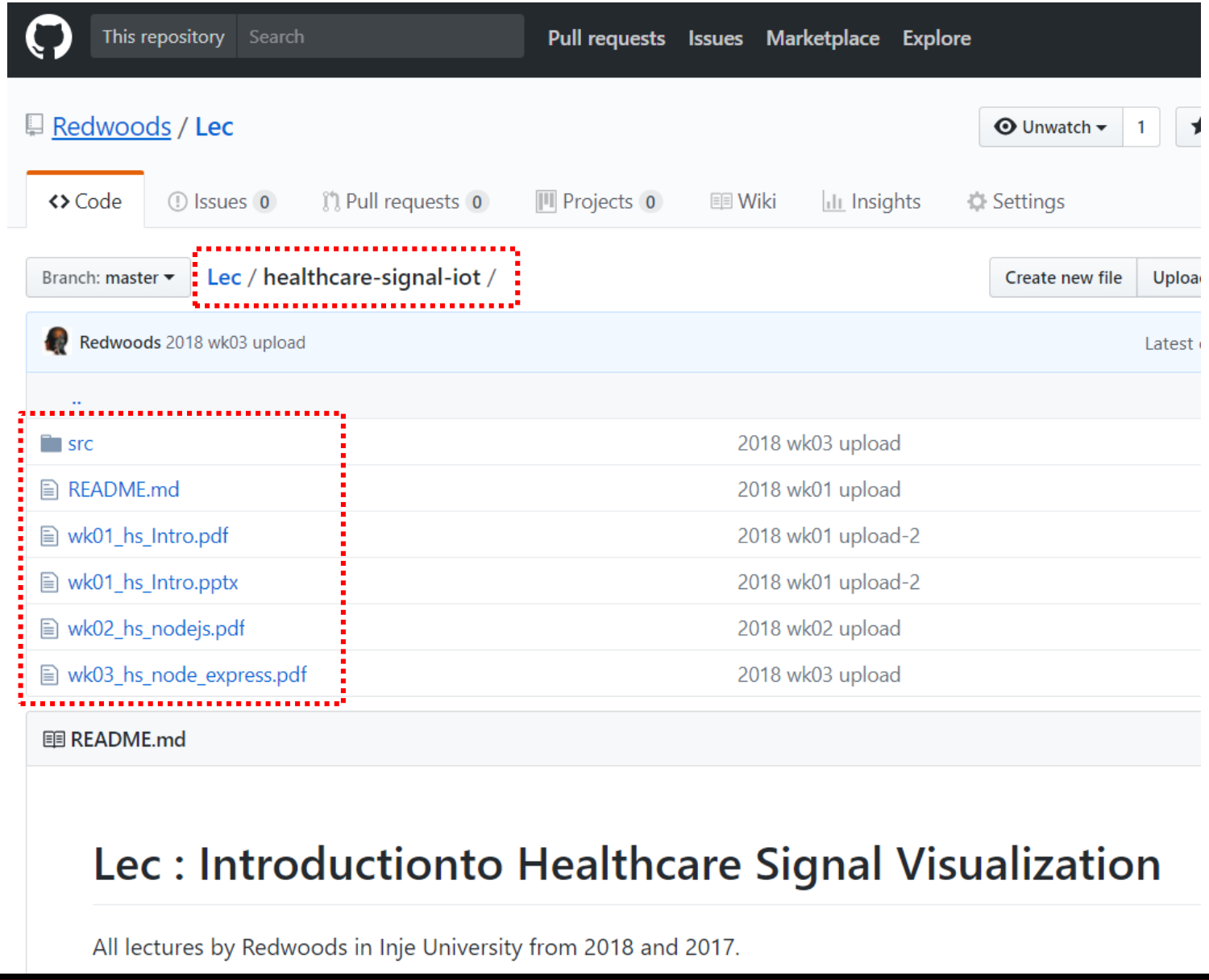
81 commits | 1 branch | 0 releases

Branch: master | New pull request | Create new file | Upload files

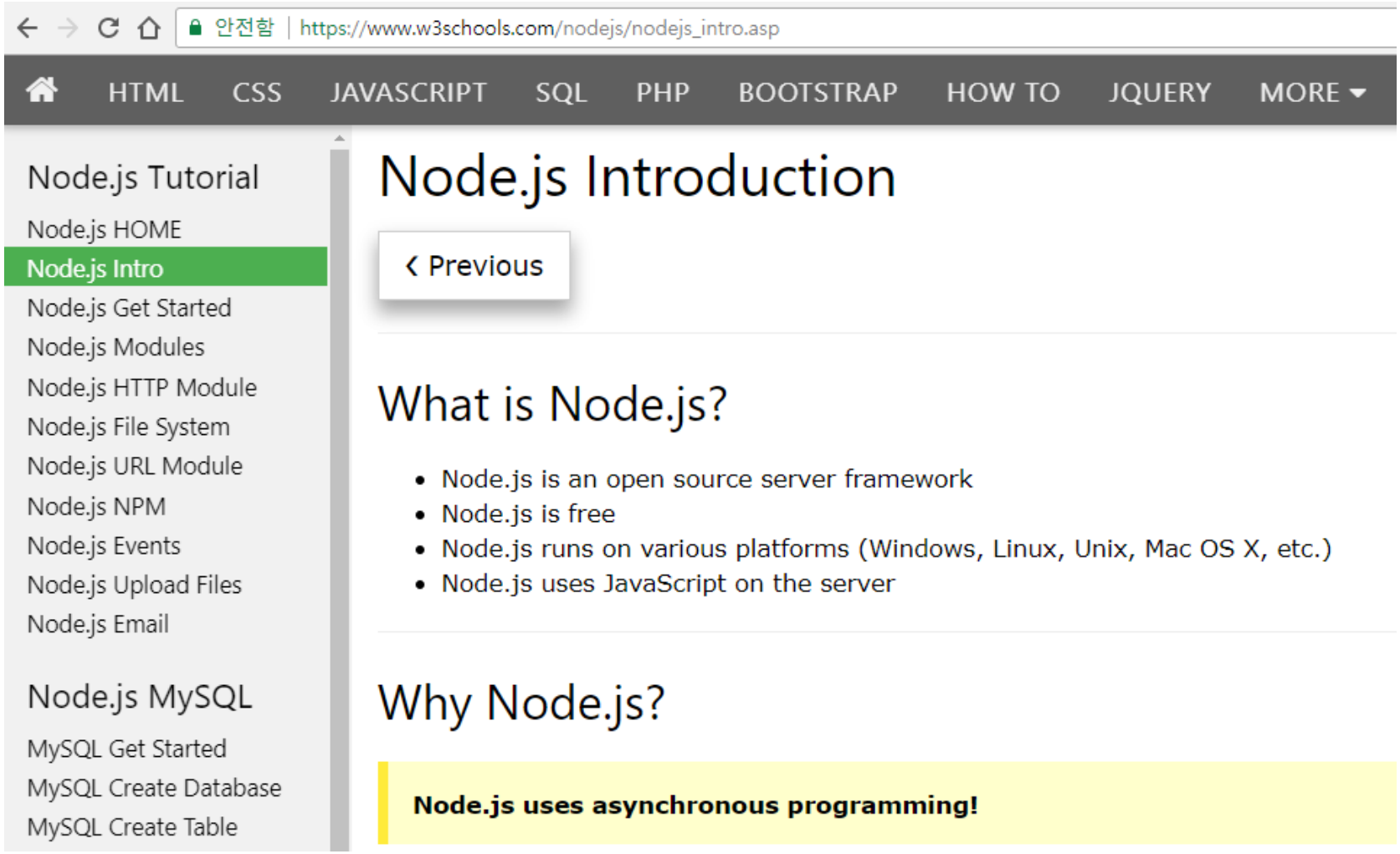
Redwoods 2018 wk01 upload

advanced-Arduino-iot	wk16 exam upload
ev3	wk16 final exam. answers
healthcare-signal-iot	2018 wk01 upload
html5-basic	2018 wk01 upload
html5-mobile-simulation	wk15 lec upload
Lec.Rproj	2018 wk01 upload
README.md	wk03 upload and fix links

Github.com/Redwoods/healthcare-signal-iot

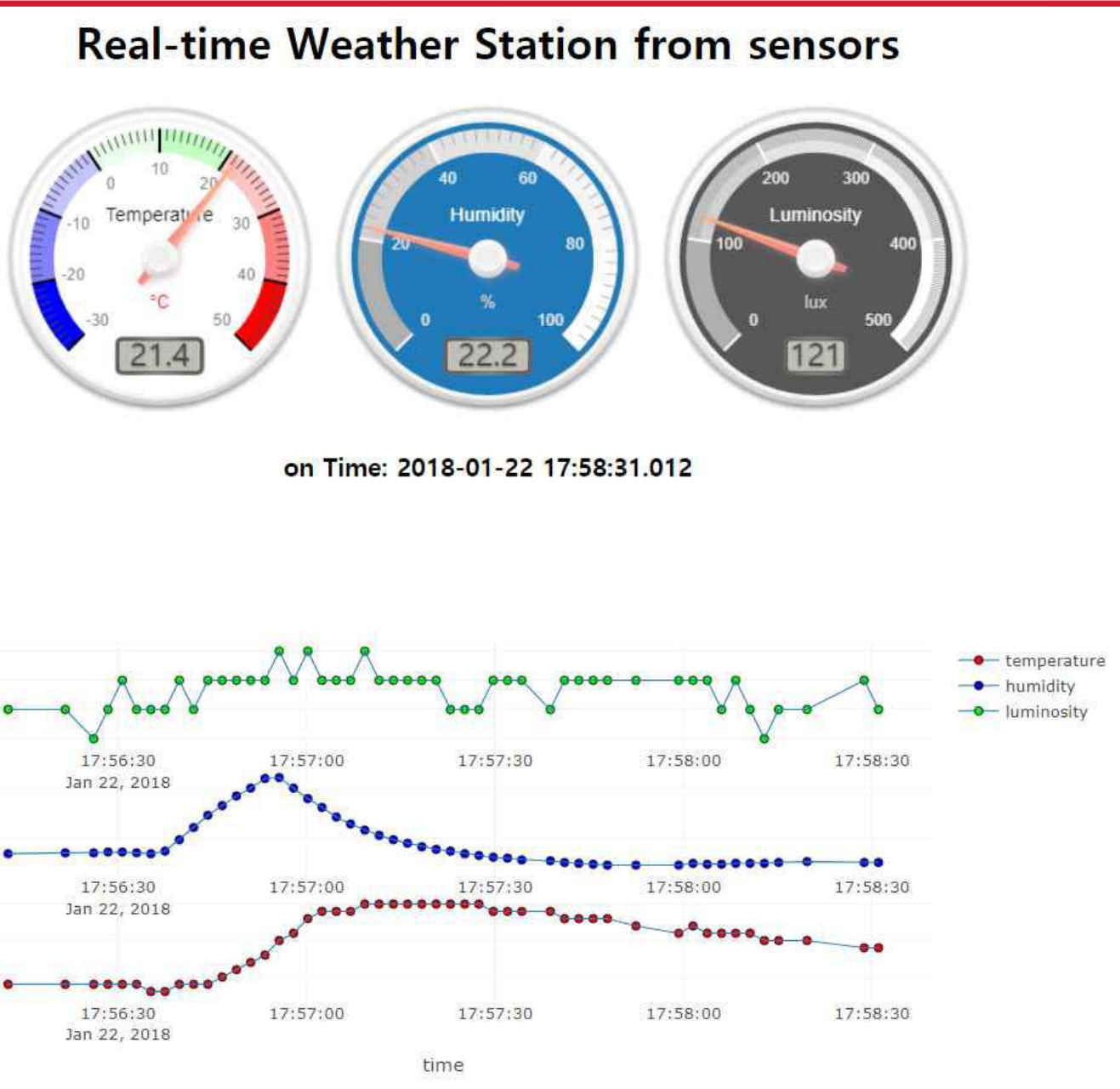


1.0 What is node.js?

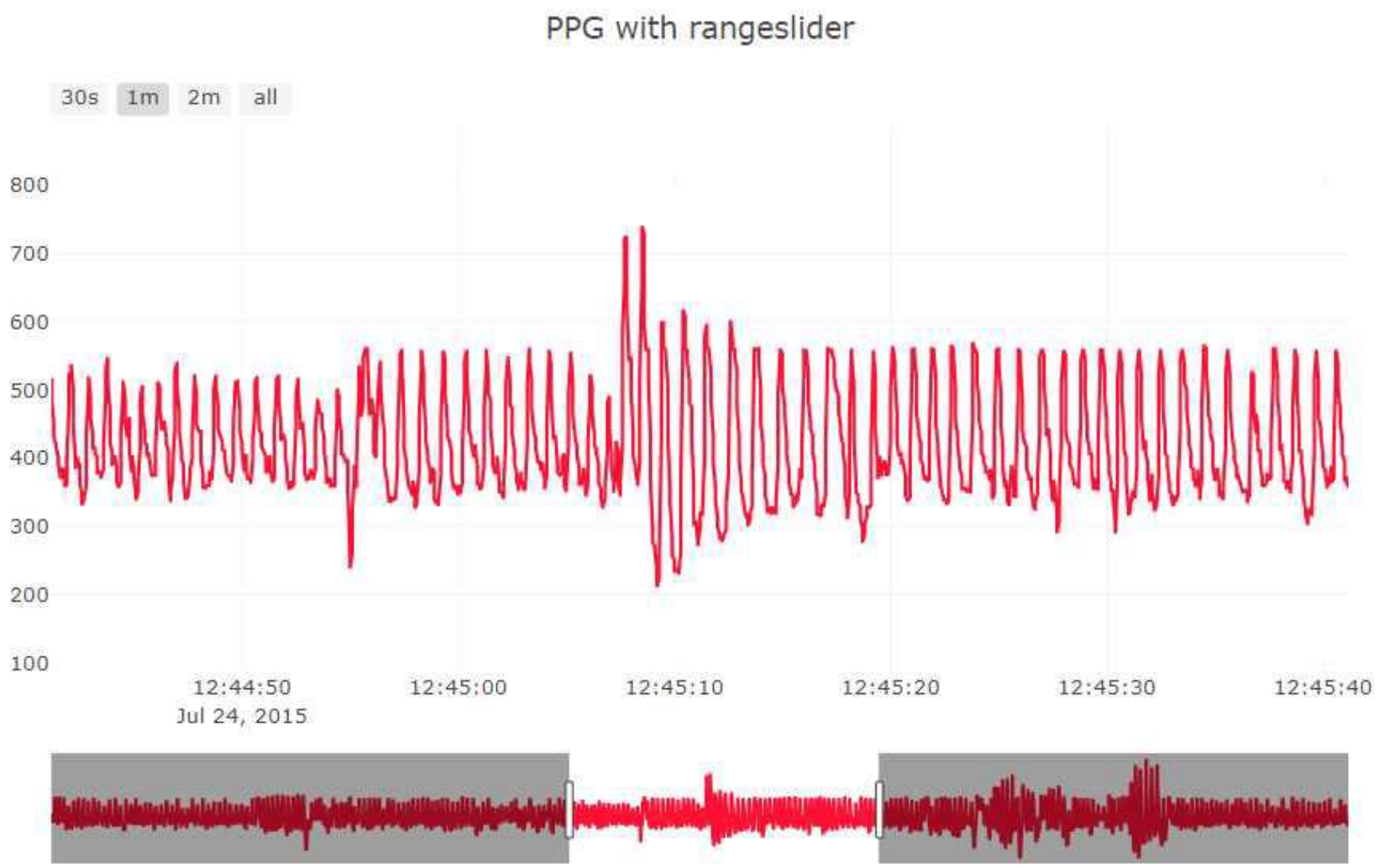


https://www.w3schools.com/nodejs/nodejs_intro.asp

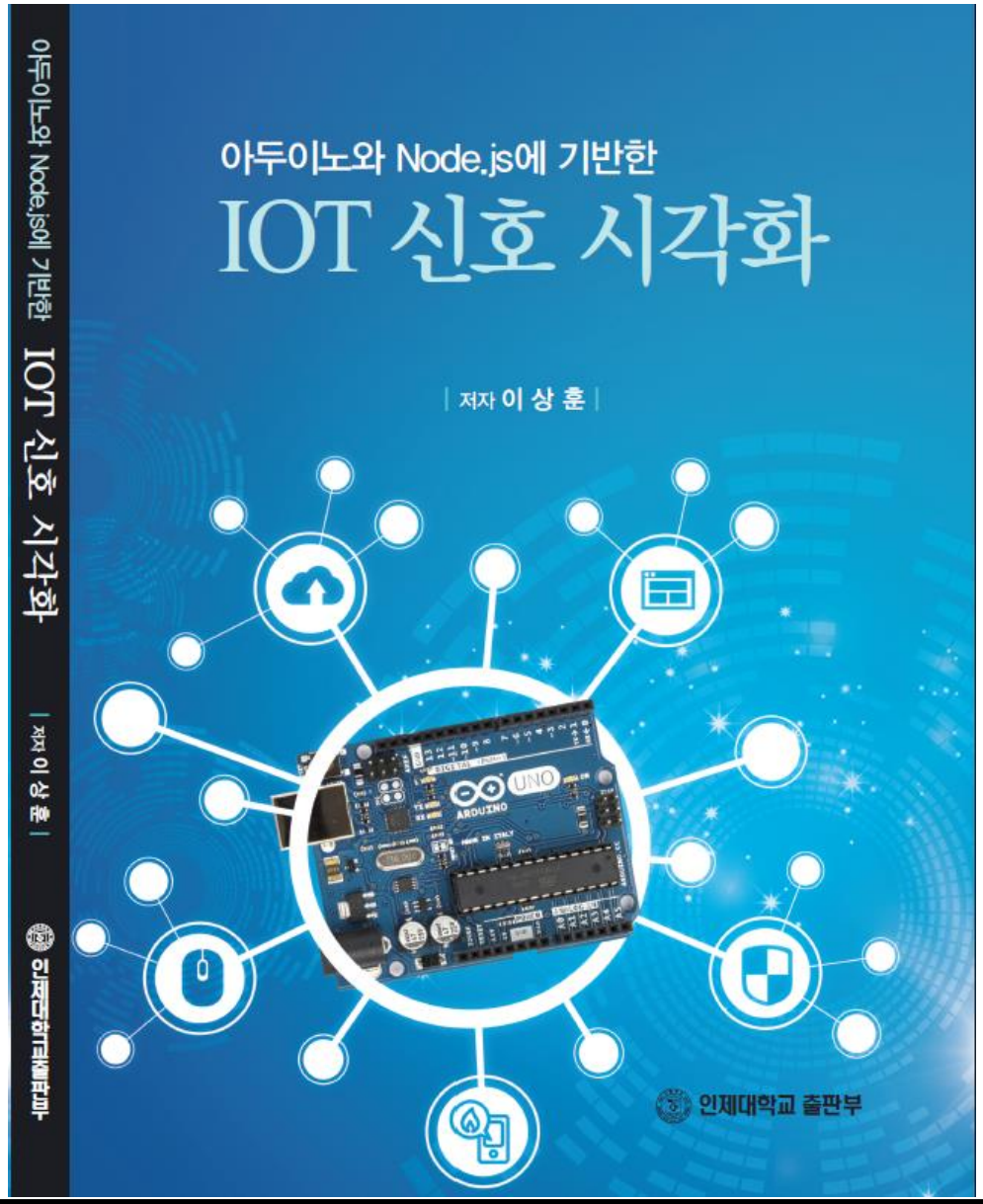
Target of this class



Project of this class



주교재


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