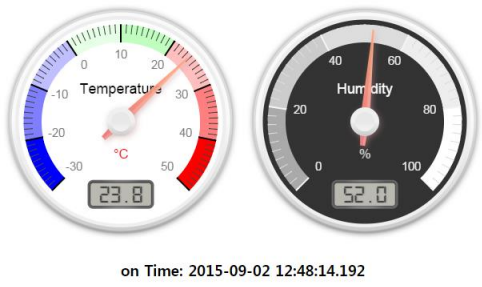
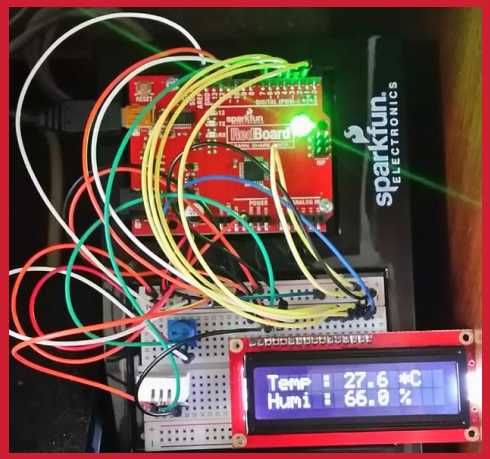
Healthcare-IOT
[wk06]





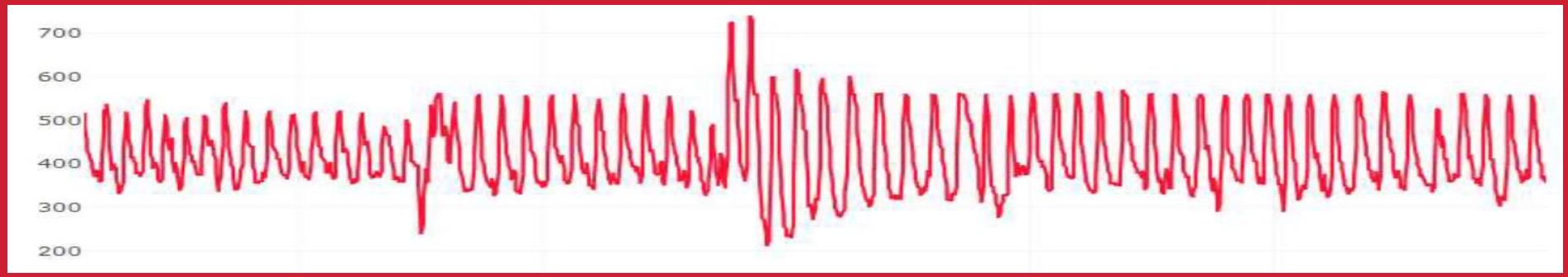
Arduino + Node
I. Single signal

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]

- [wk05]
  - Arduino sensors
  - Complete your project
  - Submit file : HSnn_Rpt04.zip

wk05 : Practice-04 : HSnn_Rpt04.zip

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

제출파일명 : **HSnn_Rpt04.zip**

- 압축할 파일들

① **HSnn_AnalogVoltage.png**

② **HSnn_TMP36.png**

③ **HSnn_Hello_LCD.png**

④ **HSnn_LCD_lux.ino**

⑤ **HSnn_LCD_lux.png**

Email : chaos21c@gmail.com

[제목 : id, 이름 (수정)]

[My working folder]

PC › DATA (D:) › Portable › arduino-1.8.5 › hs00 ›

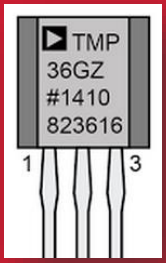
- hcit
- HS00_AnalogRead_fmap
- HSnn_TMP36_NodeJS
- libraries
- sketch01_blink
- sketch01_start
- sketch02_pwm_led
- sketch03_pwm_led_serial
- sketch04_pwm_3_leds
- sketch05_multi_signals
- sketch06_analog_read
- sketch06_analog_read_start
- sketch07_tmp36
- sketch07_tmp36_start
- sketch08_CdS
- sketch08_CdS_start
- sketch08_CdS2
- sketch09_hello_LCD
- sketch09_hello_LCD_start
- sketch10_CdS_LCD_lux
- sketch10_CdS_LCD_lux_start

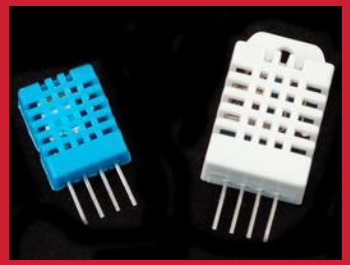
PC › DATA (D:) › Portable › NodeJSPortable › Data › hs00 ›

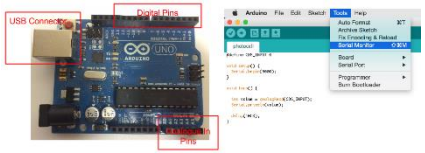
- express
- expressTest
- hs00App
- myApp
- server
- start

Arduino

+ Node.js







IOT: HSC

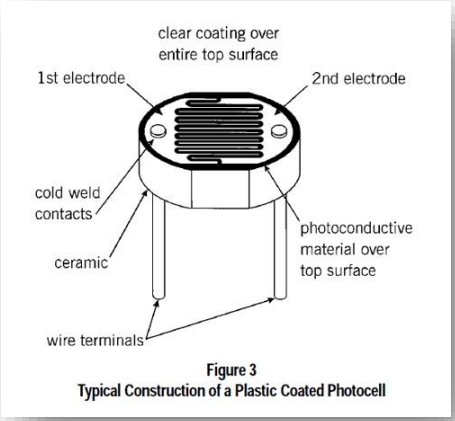
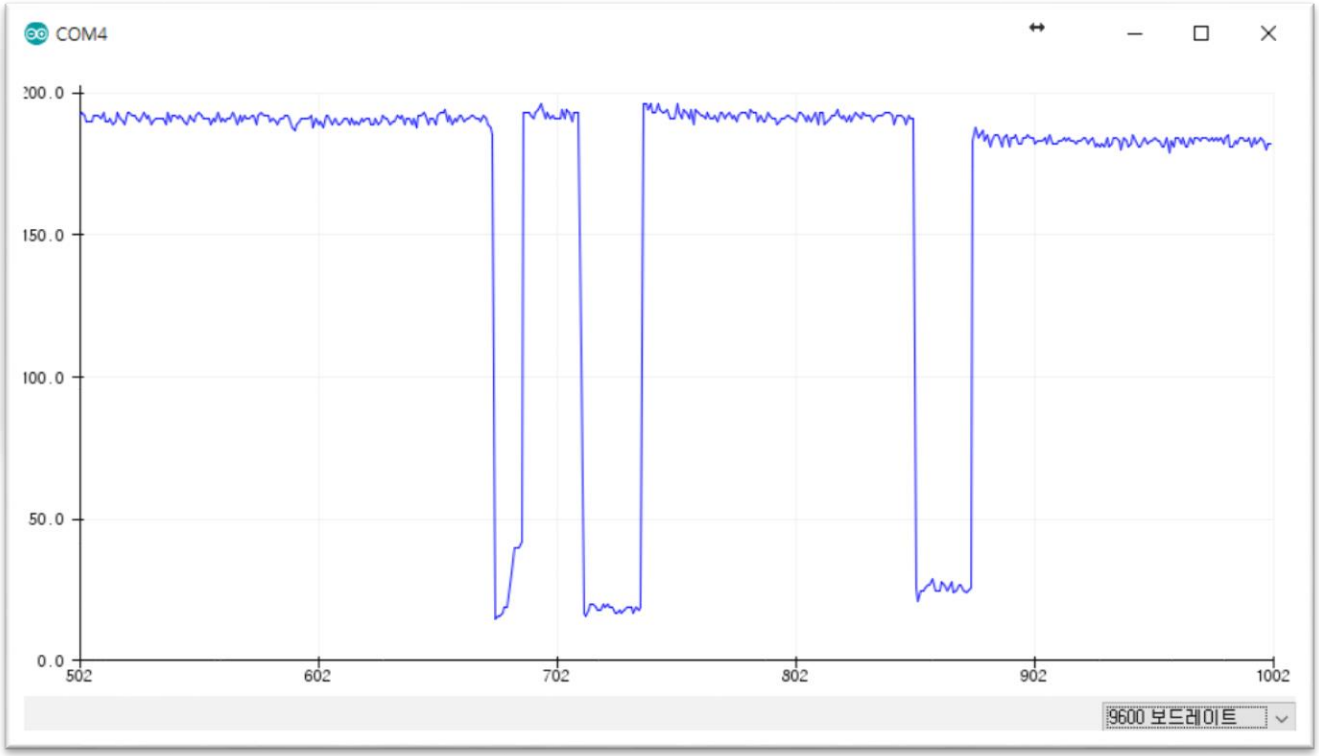
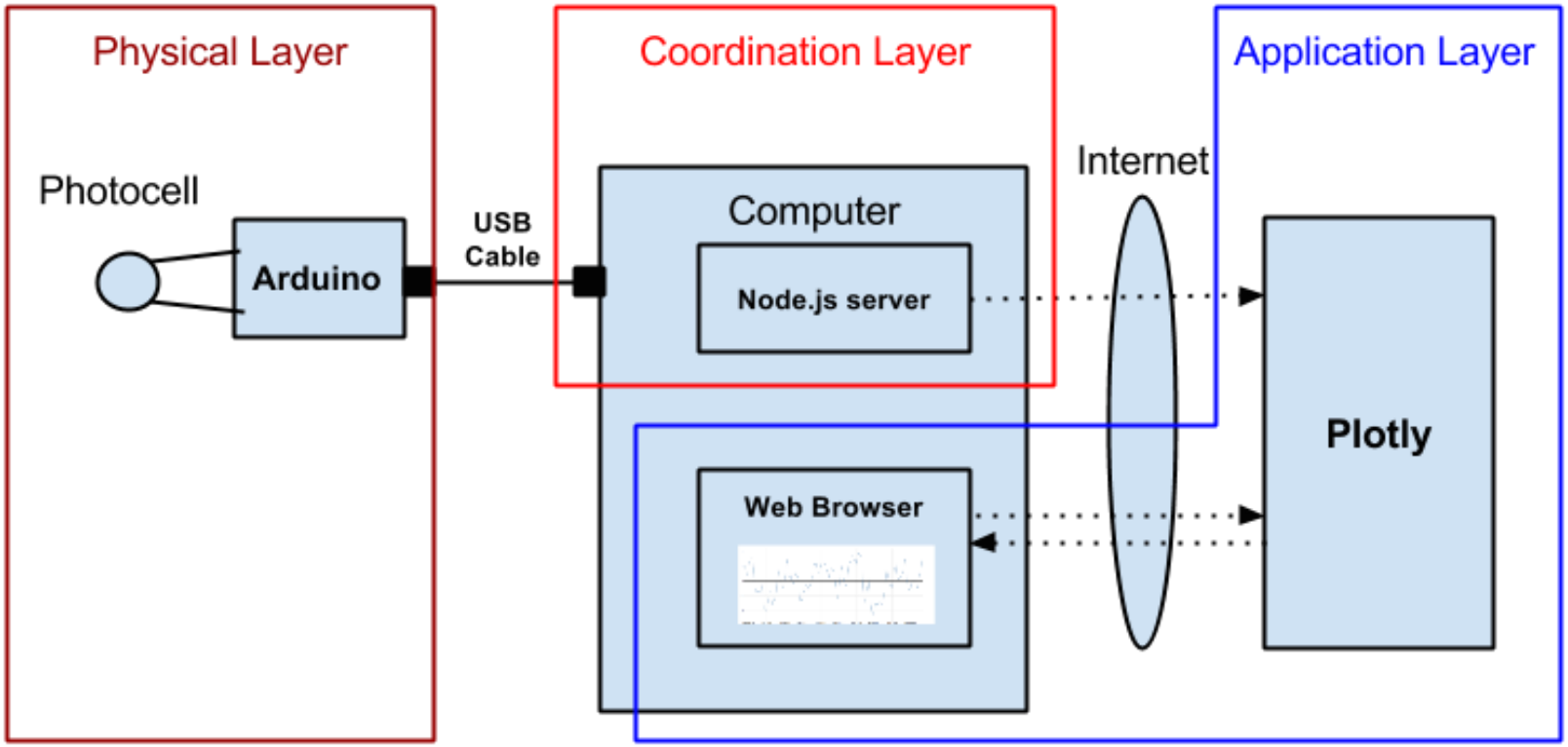


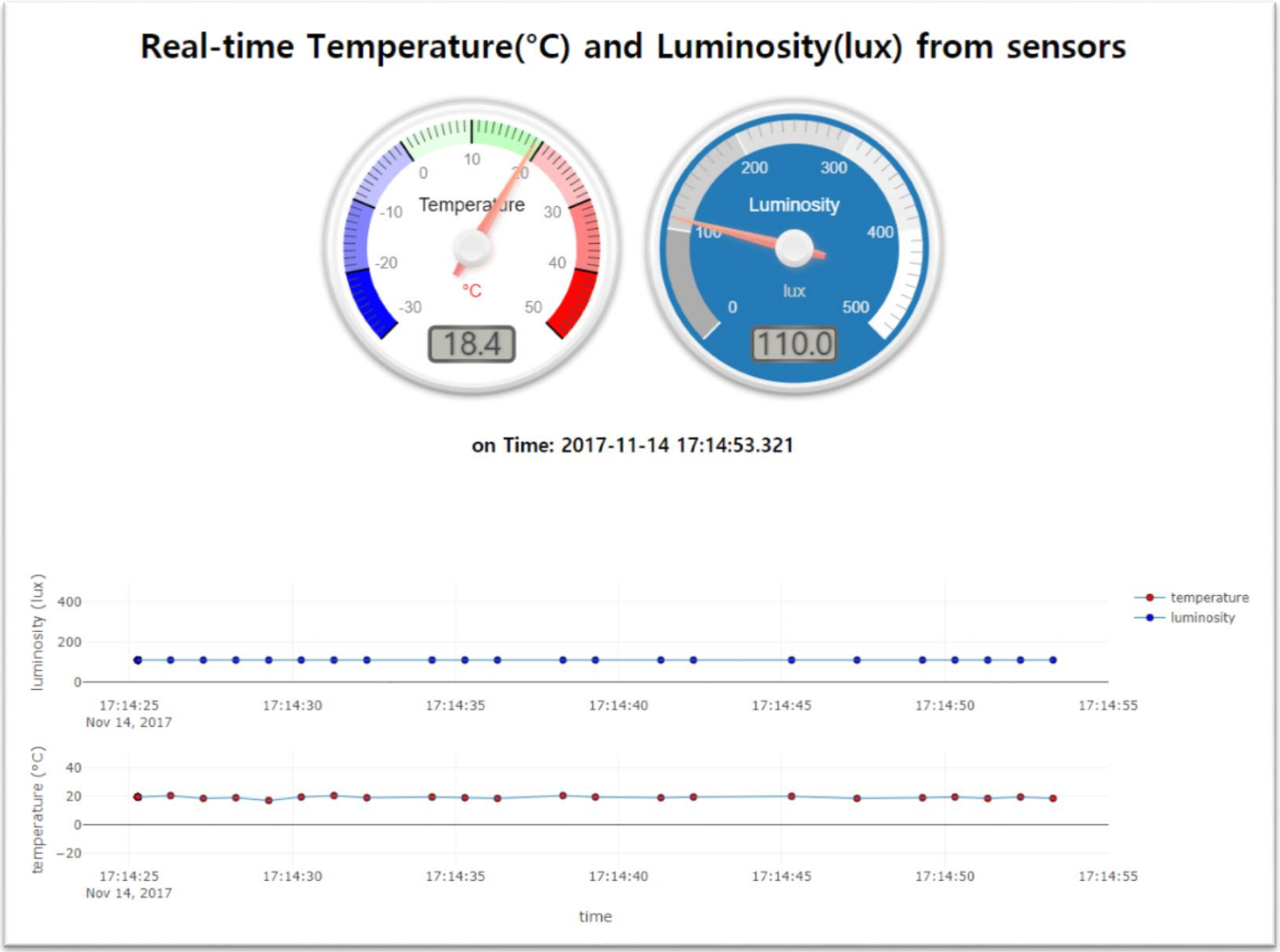
Figure 3
Typical Construction of a Plastic Coated Photocell



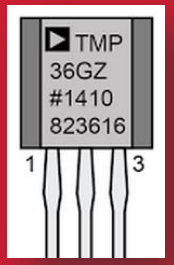
Layout [H S C]

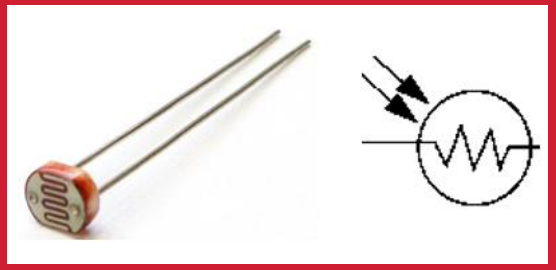

Physical Layer
Photocell
Arduino
USB Cable
Coordination Layer
Computer
Node.js server
Web Browser
Internet
Application Layer
Plotly

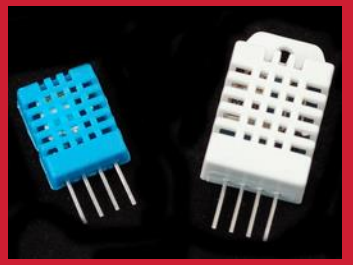
Arduino: node.js + plotly



Arduino Sensors



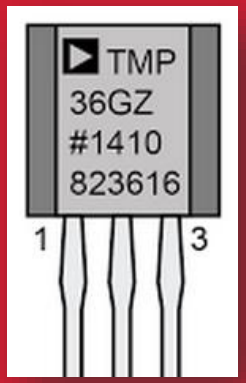




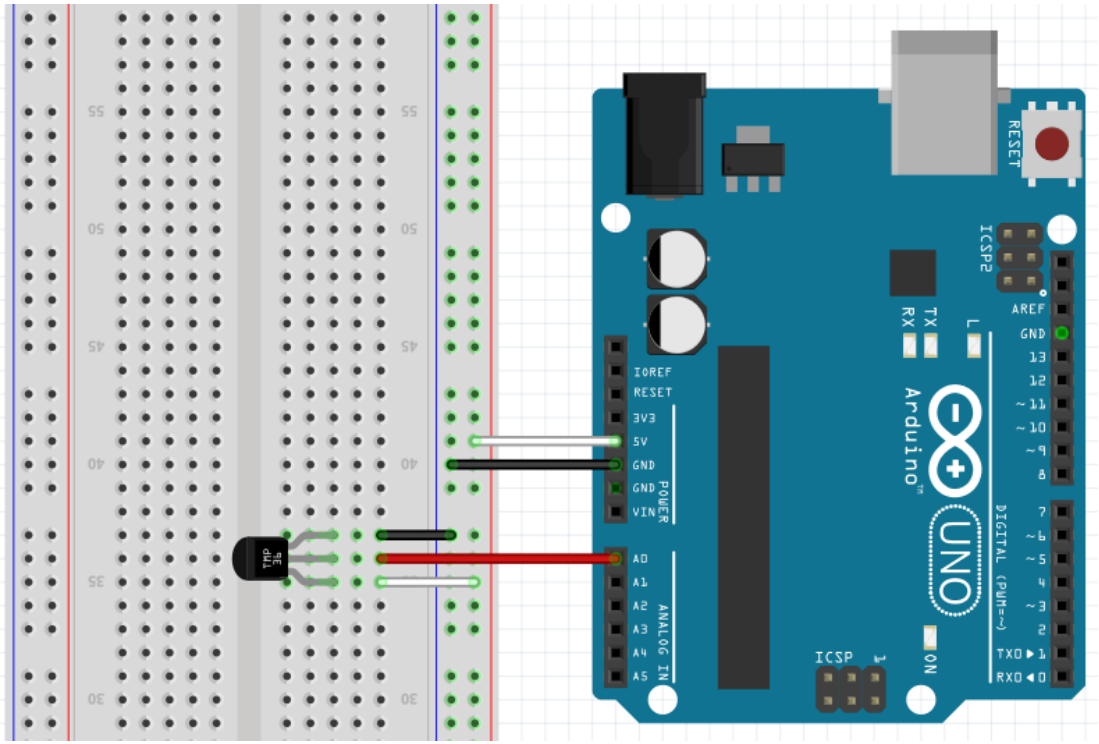
Single sensor: TMP36

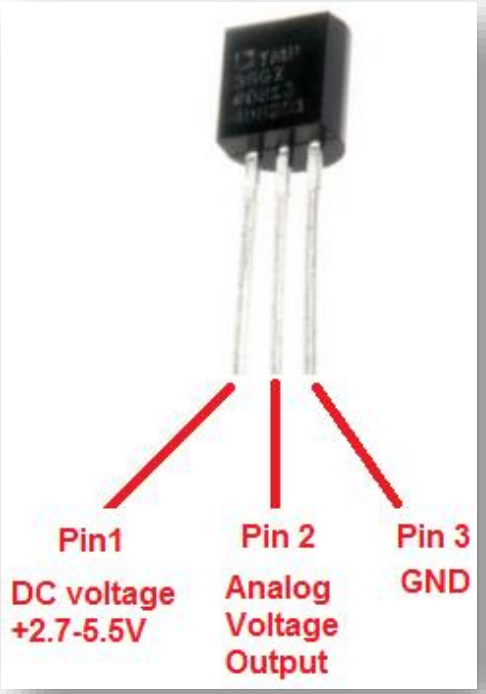
Arduino

+ Node.js

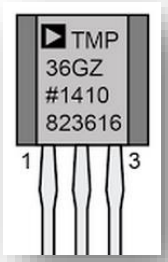


A3.1.1 Temperature sensor [TMP36]





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

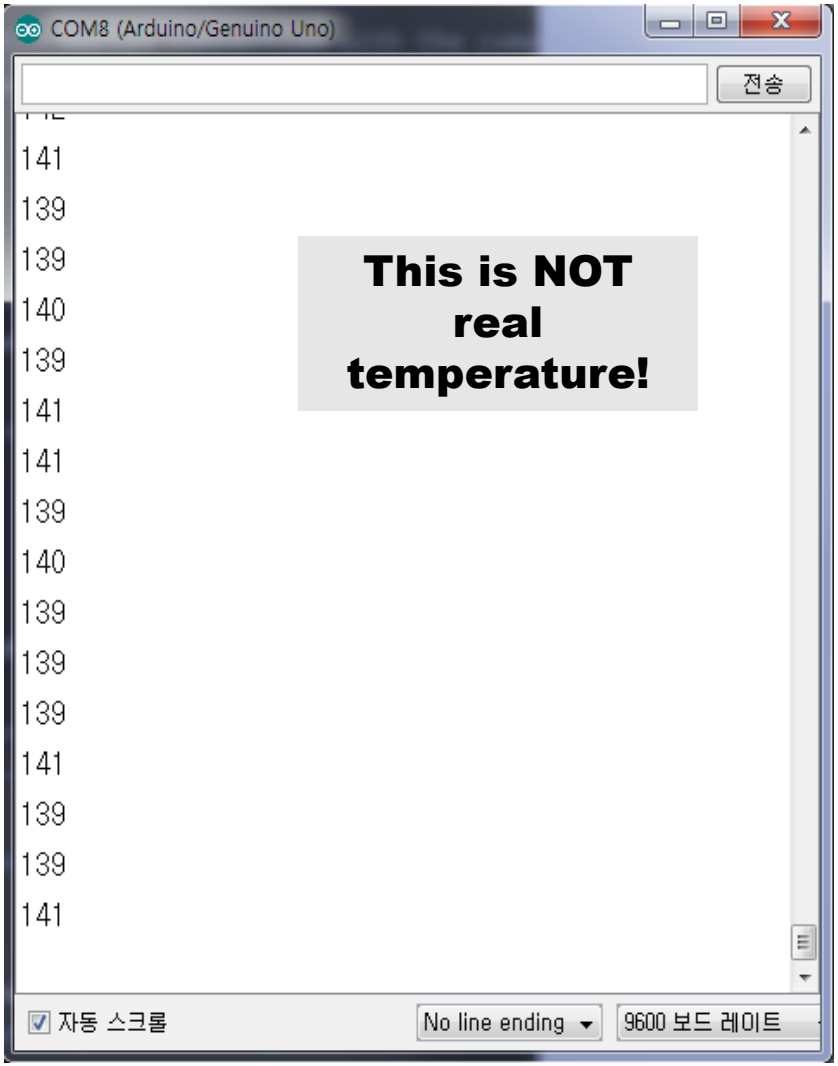
A3.1.2 Temperature sensor [TMP36]

Simple code

TMP36 §

```
//
//    HS00, TMP36 sensor
//

#define TEMP_INPUT 0
// or   int TEMP_INPUT = 0;

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

void loop() {

  int value = analogRead(TEMP_INPUT);
  Serial.println(value);

  delay(1000);
}
```

Serial output (0 ~ 1023)



A3.1.3 Temperature sensor [TMP36]

Sensor property

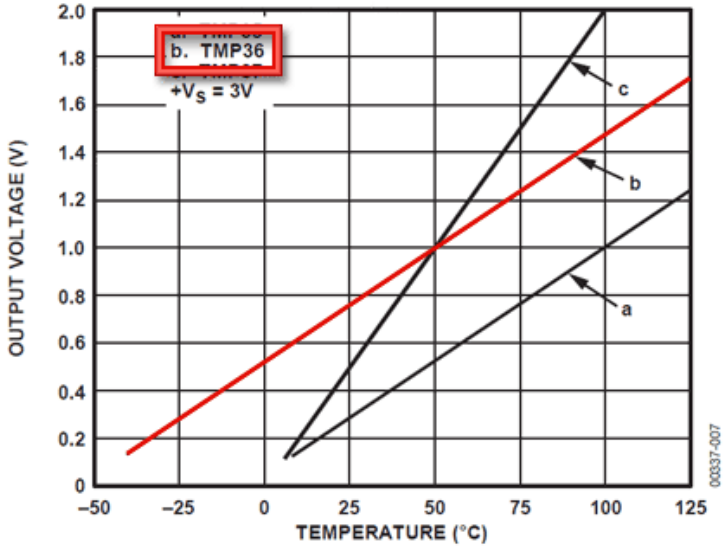


Figure 6. Output Voltage vs. Temperature

Temperature conversion

Temp (° C) = (Vout – 500) / 10

Vout (mV) = value * (5000 / 1023)

(0 <= value <= 1023)

```
// converting that reading to voltage
float voltage = value * 5.0 * 1000;  // in mV
voltage /= 1023.0;
float temperatureC = (voltage - 500) / 10 ;
```

A3.1.4 Temperature sensor [TMP36]

Working code

TMP36

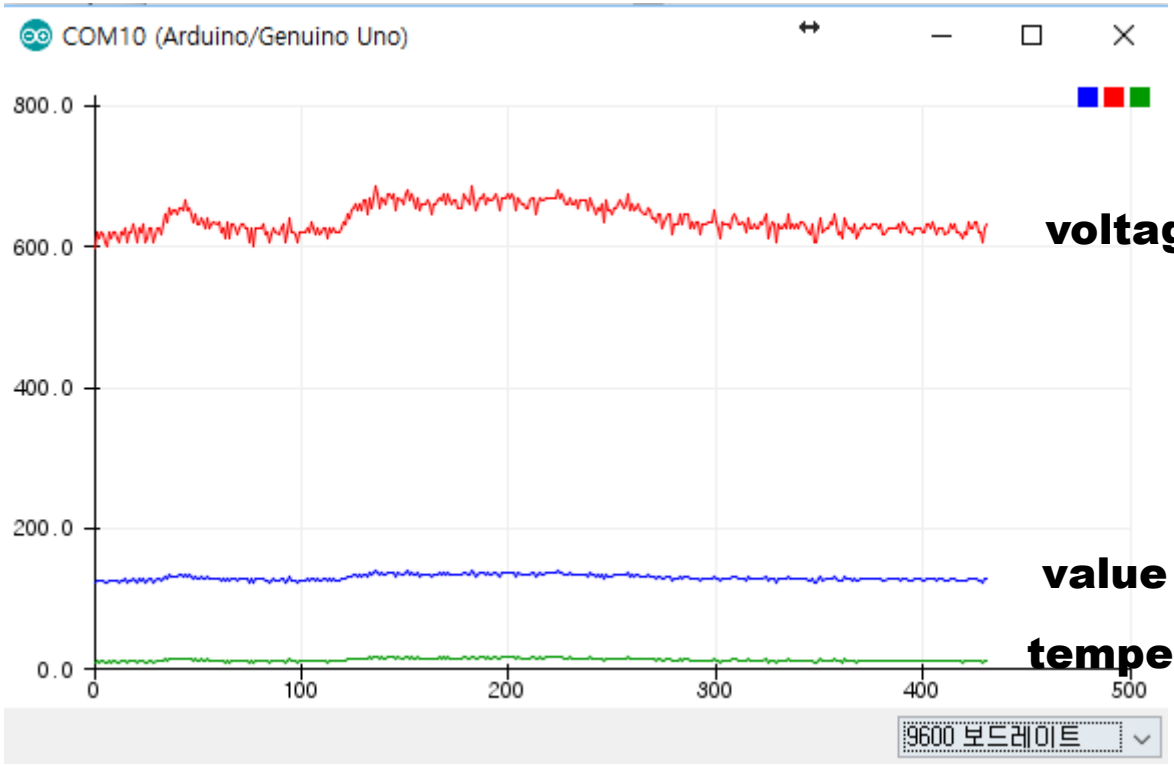
```
10 }
11
12 void loop() {
13    //getting the voltage reading from the temperature sensor
14   int value = analogRead(TEMP_INPUT);
15   Serial.print("AA00, value = ");
16   Serial.print(value);
17   Serial.print(" : ");
18
19    // converting that reading to voltage
20    float voltage = value * 5.0 * 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(temperatureC);
30    Serial.println(" degrees C");
31
32   delay(1000);
33 }
```

Serial output (°C)

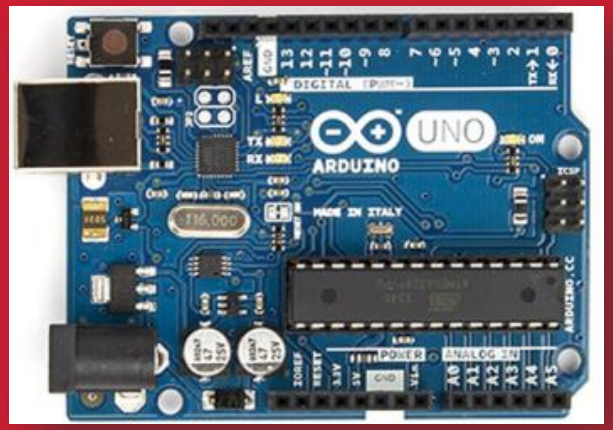
COM4

```
AA00, value = 131 : 640.27 mV, 14.03 degrees C
AA00, value = 130 : 635.39 mV, 13.54 degrees C
AA00, value = 132 : 645.16 mV, 14.52 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 129 : 630.50 mV, 13.05 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 130 : 635.39 mV, 13.54 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 132 : 645.16 mV, 14.52 degrees C
AA00, value = 129 : 630.50 mV, 13.05 degrees C
AA00, value = 132 : 645.16 mV, 14.52 degrees C
AA00, value = 129 : 630.50 mV, 13.05 degrees C
AA00, value = 130 : 635.39 mV, 13.54 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
```

A3.1.5 Temperature sensor [TMP36]

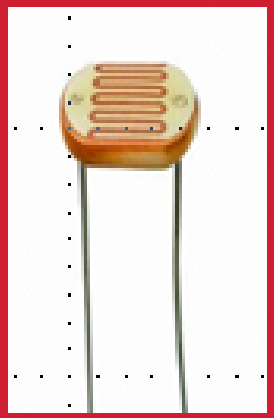


Single sensor: tmp36



TMP36

Node project



A4.1.1 tmp36 node project

Start tmp36-node project

1. **Go to my working folder**
2. **md iot & cd iot**
3. **md tmp36**
4. **cd tmp36**
5. **dir**

NodeJS

```
D:\Portable\NodeJSPortable\Data>cd hs00

D:\Portable\NodeJSPortable\Data\hs00>md iot

D:\Portable\NodeJSPortable\Data\hs00>cd iot

D:\Portable\NodeJSPortable\Data\hs00\iot>md tmp36

D:\Portable\NodeJSPortable\Data\hs00\iot>cd tmp36

D:\Portable\NodeJSPortable\Data\hs00\iot\tmp36>dir
 D 드라이브의 볼륨: DATA
 볼륨 일련 번호: 7A01-106A

 D:\Portable\NodeJSPortable\Data\hs00\iot\tmp36 디렉터리

2018-04-11  오후 03:57    <DIR>          .
2018-04-11  오후 03:57    <DIR>          ..
               0개 파일                   0 바이트
               2개 디렉터리  877,169,827,840 바이트 남음

D:\Portable\NodeJSPortable\Data\hs00\iot\tmp36>
```

A4.1.2 tmp36 node project

Set tmp36-node project

1. npm init
2. description

   tmp36-node project
3. entry point

   tmp36_node.js
4. author

   your id : hsnn

```
NodeJS
D:\Portable\NodeJSPortable\Data\aa00\tmp36>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: (tmp36)
version: (1.0.0)
description: tmp36-node project
entry point: (index.js) tmp36_node.js
test command:
git repository:
keywords: tmp36, node, arduino
author: aa00
license: (ISC) MIT
About to write to D:\Portable\NodeJSPortable\Data\aa00\tmp36\package.json:

{
  "name": "tmp36",
  "version": "1.0.0",
  "description": "tmp36-node project",
  "main": "tmp36_node.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "tmp36",
    "node",
    "arduino"
  ],
  "author": "aa00",
  "license": "MIT"
}


Is this ok? (yes)

D:\Portable\NodeJSPortable\Data\aa00\tmp36>
```

A4.1.3 tmp36 node project

package.json

package.json

```
{
  "name": "tmp36",
  "version": "1.0.0",
  "description": "tmp36-node project",
  "main": "tmp36_node.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "tmp36",
    "node",
    "arduino"
  ],
  "author": "aa00",
  "license": "MIT"
}
```

A4.1.4 tmp36 node project

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.println(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.print(temperatureC);
31 //    Serial.println(" degrees C");
32
33   delay(1000);
34 }
```

Serial output (A0, 0 ~ 1023)

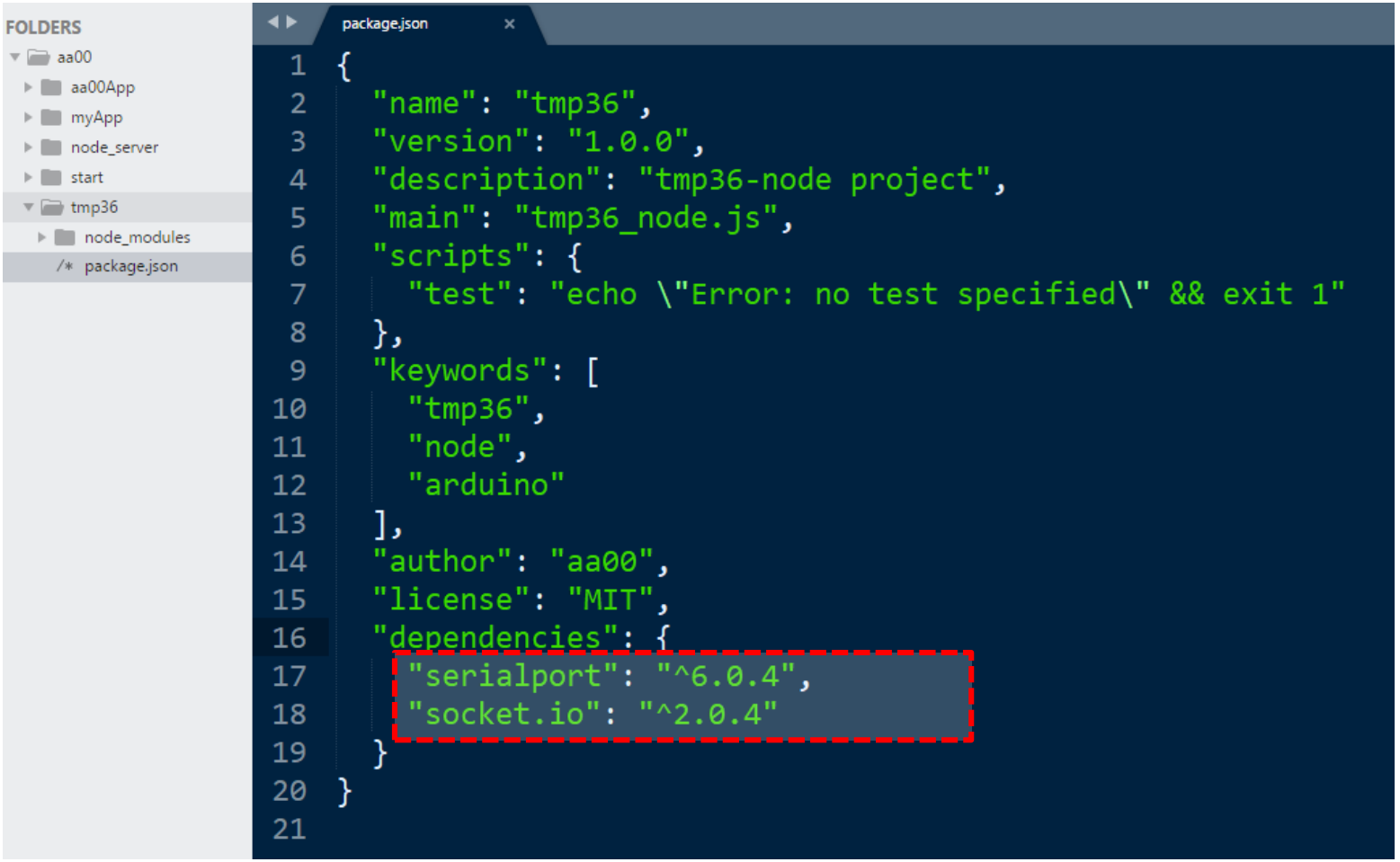
COM4

```
AA00, value = 126
AA00, value = 128
AA00, value = 128
AA00, value = 130
AA00, value = 128
AA00, value = 130
AA00, value = 130
AA00, value = 126
AA00, value = 130
AA00, value = 128
AA00, value = 129
AA00, value = 132
AA00, value = 129
AA00, value = 128
```

A4.1.5 tmp36 node project

Go to tmp36 subfolder

- **npm install –save serialport**
- **npm install –save socket.io**



A4.1.6 tmp36 node project : code-1

tmp36_node.js

```
1  // tmp36_node.js
2
3  var serialport = require('serialport');
4  var portName = 'COM10';  // 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')  // new serialport.parsers.Readline
17 });
18
19 var tdata = 0;  // temperature
20
21 sp.on('data', function (data) { // call back when data is received
22     // raw data only
23         //console.log(data);
24         tdata = data;  // data
25         console.log("HS00," + tdata);
26         io.sockets.emit('message', tdata);  // send data to all clients
27 });
28
```

serialport 6.x 버전의 API 변화로 오류 발생, 버전 downgrade

A4.1.7 tmp36 node project : code-2

tmp36_node.js

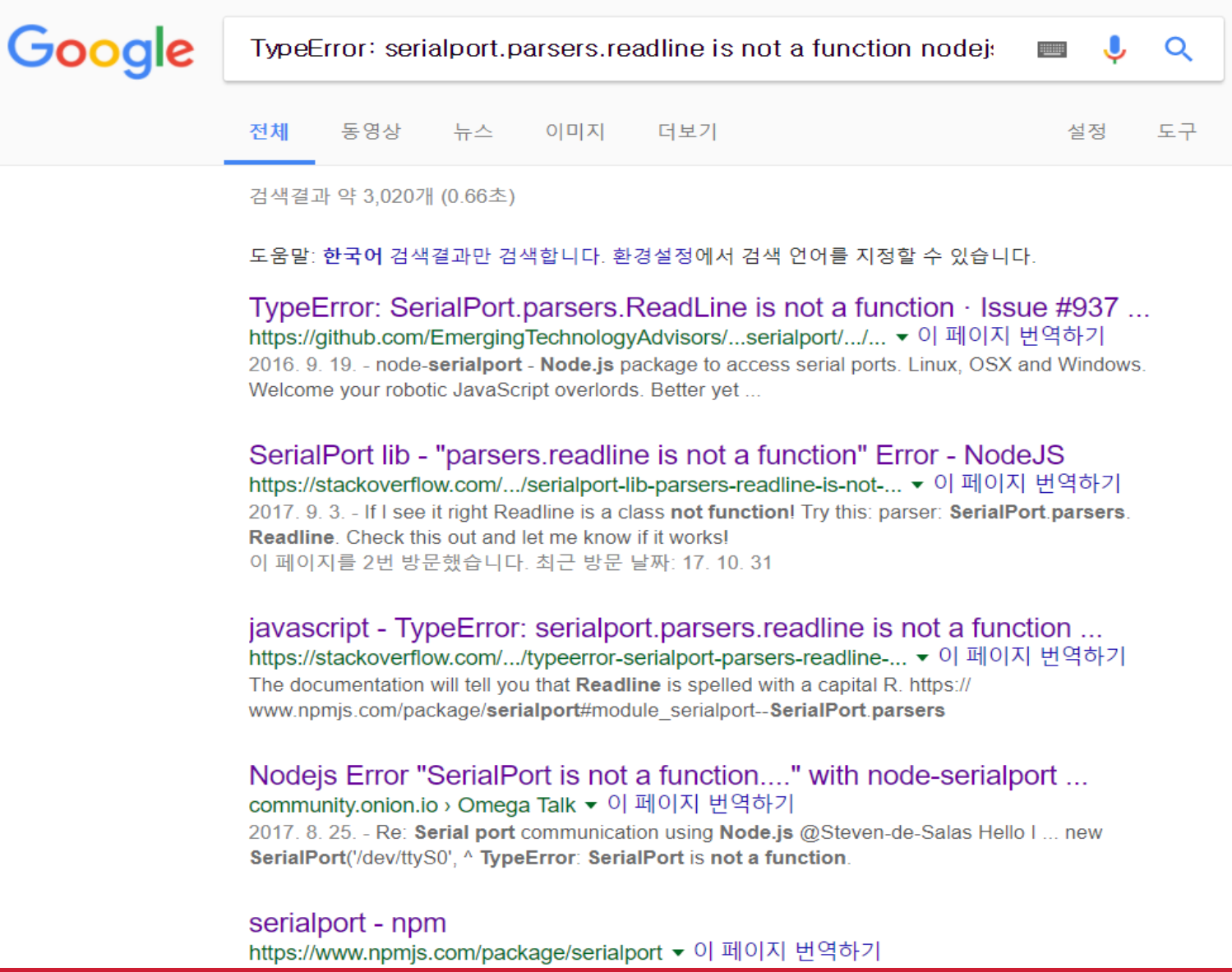
```
33 io.sockets.on('connection', function (socket) {
34     // If socket.io receives message from the client browser then
35     // this call back will be executed.
36     socket.on('message', function (msg) {
37         console.log(msg);
38     });
39     // If a web browser disconnects from Socket.IO then this callback is called.
40     socket.on('disconnect', function () {
41         console.log('disconnected');
42     });
43 });
44
```

serialport 6.x 버전의 API 변화로 오류 발생, 버전 downgrade

TypeError: SerialPort.parsers.ReadLine is not a function · Issue #937 ...
https://github.com/EmergingTechnologyAdvisors/.../... ▾ 이 페이지 번역하기
2016. 9. 19. - node-**serialport** - **Node.js** package to access serial ports. Linux, OSX and Windows. Welcome your robotic JavaScript overlords. Better yet ...

Error & Bug ---

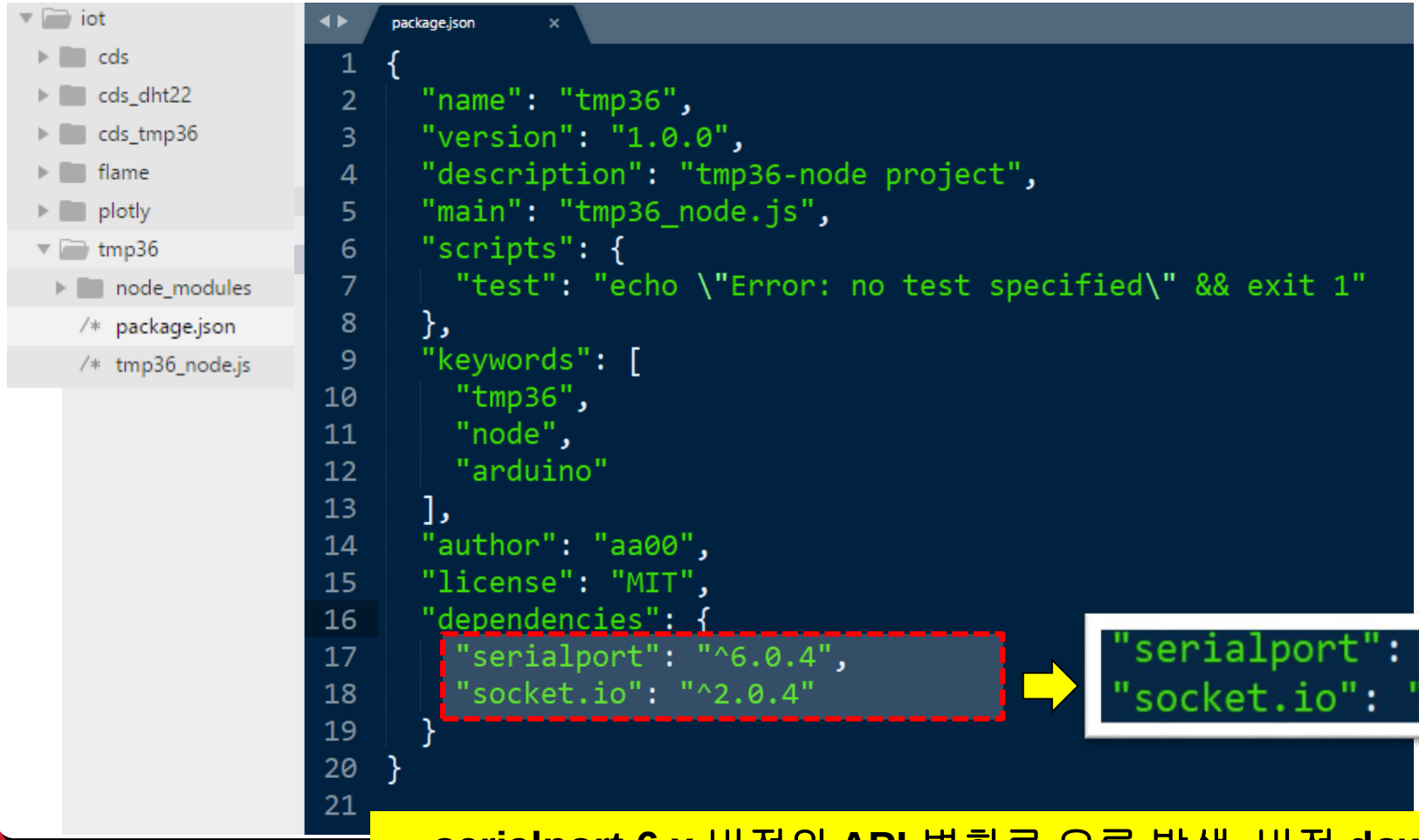
serialport 6.x 버전의 API 변화로 오류 발생, 버전 downgrade



A4.1.6A tmp36 node project → downgrade

Go to tmp36 subfolder (after deleteing node_modules subfolder)

- **"dependencies" 속성의 버전을 아래와 같이 변경**
- **npm install**



serialport 6.x 버전의 API 변화로 오류 발생, 버전 downgrade

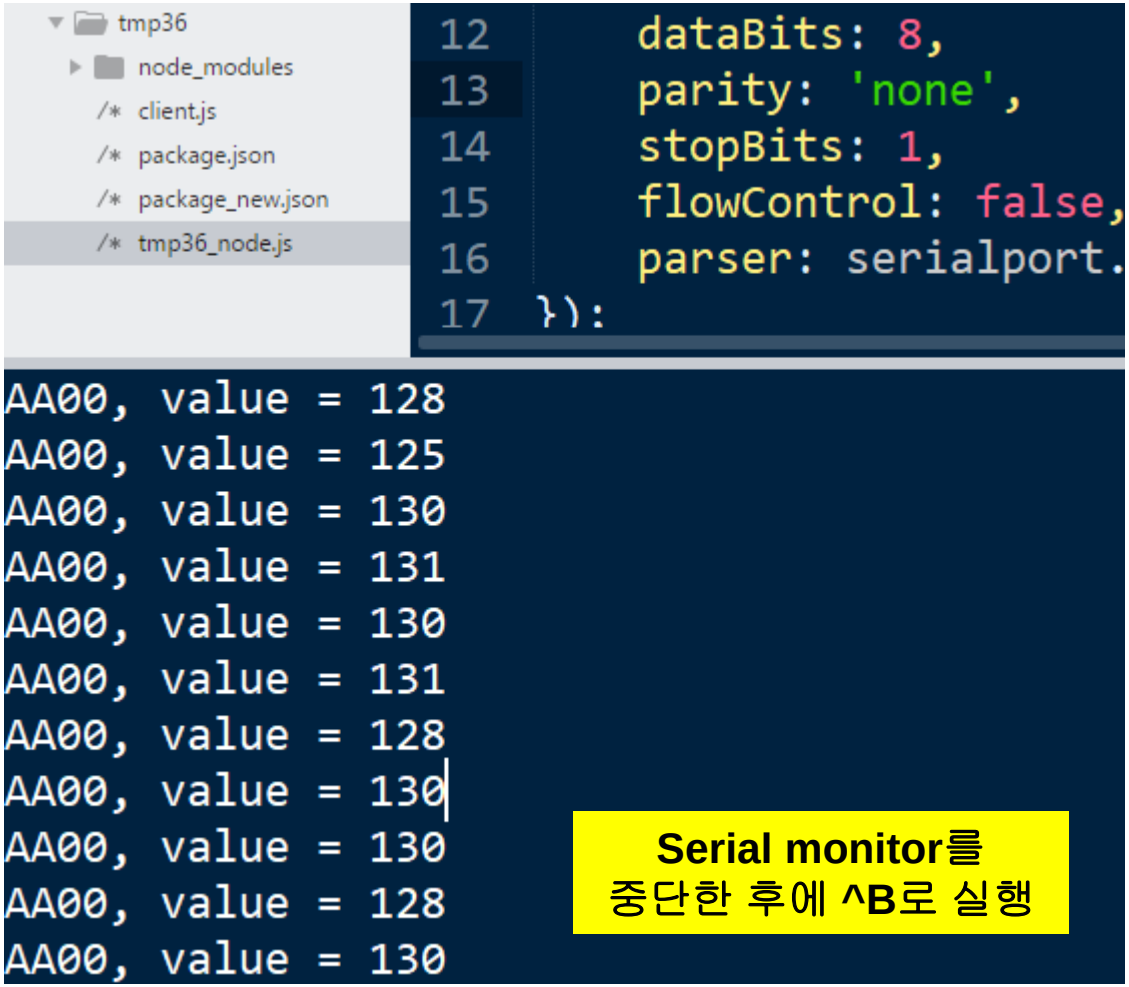
A4.1.8 tmp36 node project (after downgrade)

Serial output (A0 in node)

COM4

```
AA00, value = 126
AA00, value = 131
AA00, value = 132
AA00, value = 129
AA00, value = 130
AA00, value = 132
AA00, value = 128
AA00, value = 128
AA00, value = 128
AA00, value = 130
AA00, value = 126
```

tmp36_node.js (^B로 실행)



A4.1.9 tmp36 node project (all messages)

HSnn_TMP36_NodeJS.ino

HSnn_TMP36_NodeJS

```
11
12 void loop() {
13    //getting the voltage reading from the temperature sensor
14   int value = analogRead(TEMP_INPUT);
15   Serial.print("HS00, 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(temperatureC);
30    Serial.println(" degrees C");
31
32   delay(1000);
33 }
```

SB3에서 tmp36_node.js를 ^B로 실행

```
AA00, value = 128 : 625.61 mV,  Temperature, 12.56 degrees C
AA00, value = 132 : 645.16 mV,  Temperature, 14.52 degrees C
AA00, value = 128 : 625.61 mV,  Temperature, 12.56 degrees C
AA00, value = 128 : 625.61 mV,  Temperature, 12.56 degrees C
AA00, value = 129 : 630.50 mV,  Temperature, 13.05 degrees C
AA00, value = 130 : 635.39 mV,  Temperature, 13.54 degrees C
AA00, value = 131 : 640.27 mV,  Temperature, 14.03 degrees C
AA00, value = 127 : 620.72 mV,  Temperature, 12.07 degrees C
AA00, value = 129 : 630.50 mV,  Temperature, 13.05 degrees C
AA00, value = 128 : 625.61 mV,  Temperature, 12.56 degrees C
AA00, value = 128 : 625.61 mV,  Temperature, 12.56 degrees C
```

HSnn_tmp36_message.png 로 저장

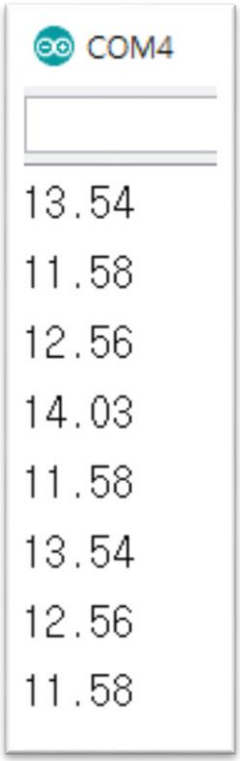
A4.1.10 tmp36 node project (only data)

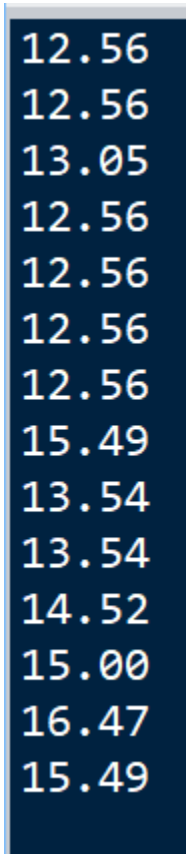
HSnn_TMP36_NodeJS.ino

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
```

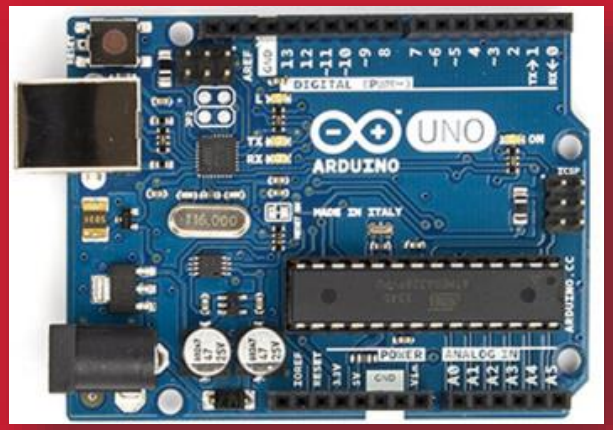
▸ Node cmd에서 실행

```
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
```

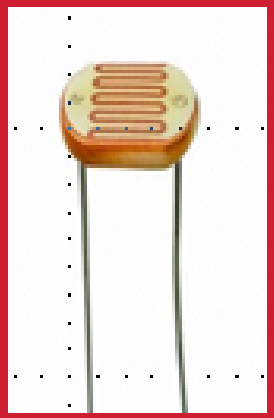
HSnn_tmp36_IOT_data.png 로 저장 (AA00 → HSnn)

Single sensor: CdS



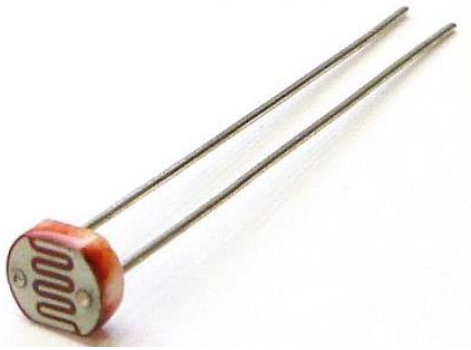
Arduino

+ Node.js

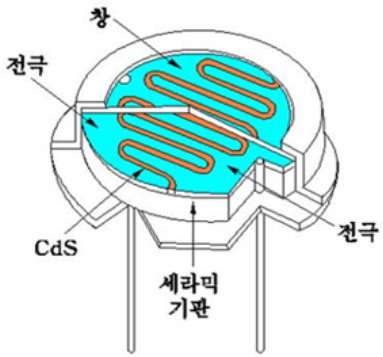


A3.2 Luminosity sensor [Photocell LDR]

CdS 센서- photoresistor



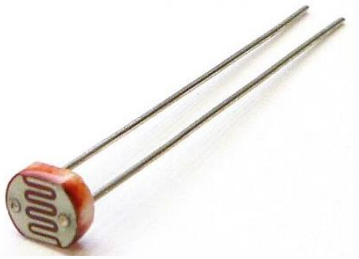
CDS특성



1. 감도
   - -빛의 파장에 따라 감도가 다름
2. 허용손실
   - -비교적 큰 전류를 흘릴 수 있음
3. 암 전류
   - -빛이 없어도 약간의 전류가 흐름
4. 명 전류
   - - 빛을 비추면 흐르는 전류
5. 응답특성
   - - 응답 시간 지연
   - - 빛의 세기에 따라 응답시간 다름
6. 가변저항
   - -빛에 따른 가변저항

https://en.wikipedia.org/wiki/Photoresistor

CdS 센서 - photoresistor



- ✓ CdS 분말을 세라믹 기판 위에 압축하여 제작
- ✓ 빛이 강할 수록 저항 값이 감소
- ✓ ADC를 이용하여 변화된 저항에 전압을 인가하여 전압의 변화를 감지
- ✓ 자동 조명장치, 조도 측정 등에 사용

럭스

다른 뜻에 대해서는 Lux 문서를 참조하십시오.

럭스(lux, 기호 **lx**)는 빛의 조명도를 나타내는 SI 단위이다. 럭스는 루멘에서 유도

$1\ \mathrm{lx} = 1\ \mathrm{lm/m^2} = 1\ \mathrm{cd \cdot sr \cdot m^{-2}}$

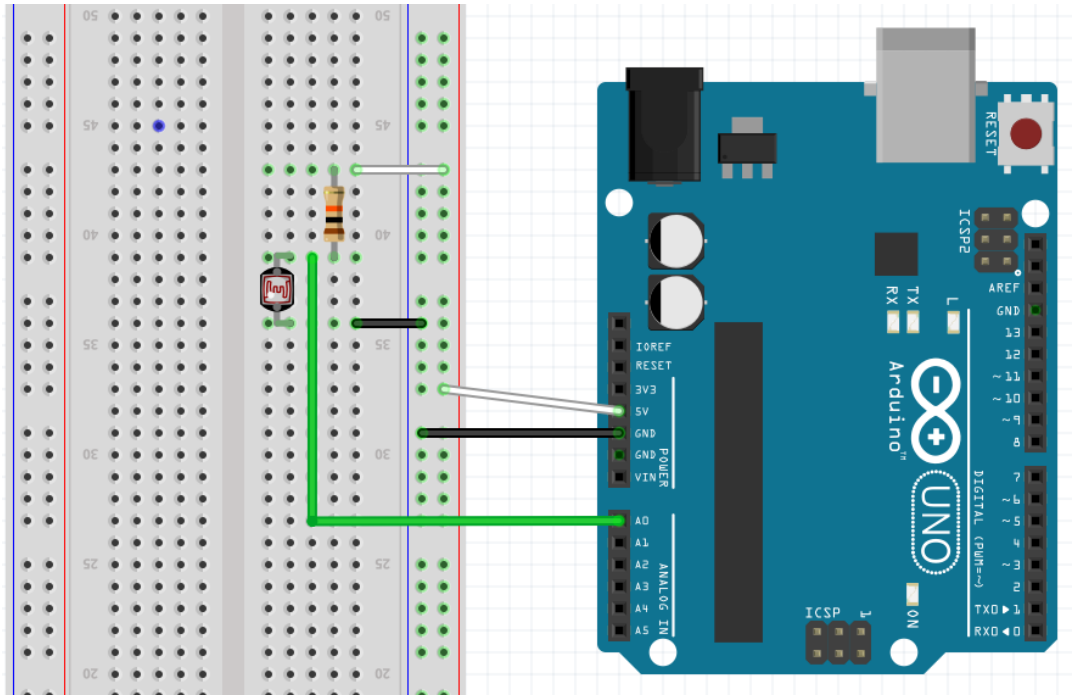
럭스의 예 [편집]

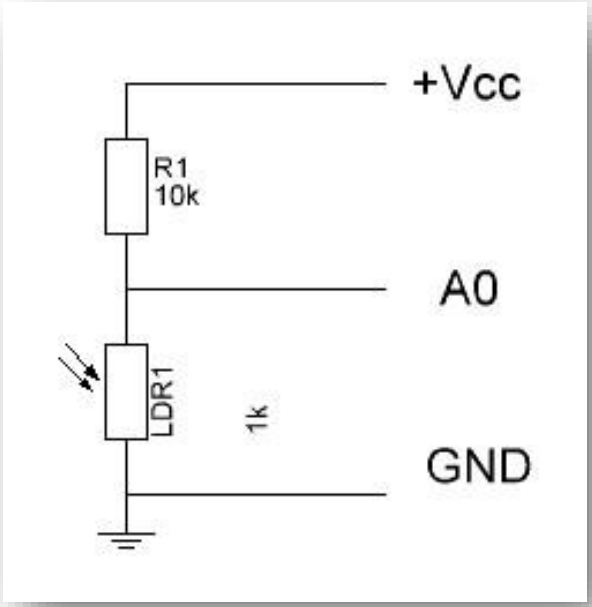
밝기차	예
10^{-5} lux	가장 밝은 별(시리우스)의 빛[1]
10^{-4} lux	하늘을 덮은 완전한 별빛[1]
0.002 lux	대기광이 있는 달 없는 맑은 밤 하늘[1]
0.01 lux	초승달
0.27 lux	맑은 밤의 보름달[1][2]
1 lux	열대 위도를 덮은 보름달[3]
3.4 lux	맑은 하늘 아래의 어두운 황혼[4]
50 lux	거실[5]
80 lux	복도/화장실[6]
100 lux	매우 어두운 낮[1]
320 lux	권장 오피스 조명 (오스트레일리아)[7]
400 lux	맑은 날의 해돋이 또는 해넘이
1000 lux	인공 조명[1]; 일반적인 TV 스튜디오 조명
10,000–25,000 lux	낮 (직사광선이 없을 때)[1]
32,000–130,000 lux	직사광선

https://en.wikipedia.org/wiki/Photoresistor

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

A3.2.3 Luminosity sensor [sketch-1]

▶ 스케치 구성

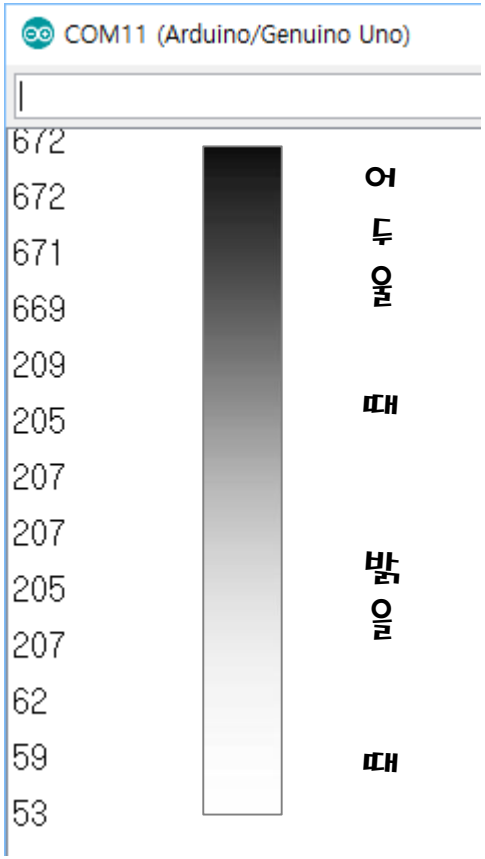
1. A0 핀을 CdS 조도 센서의 입력으로 설정한다.
2. setup()에서 직렬 통신 속도를 9600 bps 로 설정하고 컴퓨터와 연결한다.
3. loop()에서 **analogRead()** 함수로 A0 핀에서 측정되는 값을 읽어 들인다.

A3.2.4 Luminosity sensor [Photocell LDR]

CdS 센서 회로 - 측정 1.

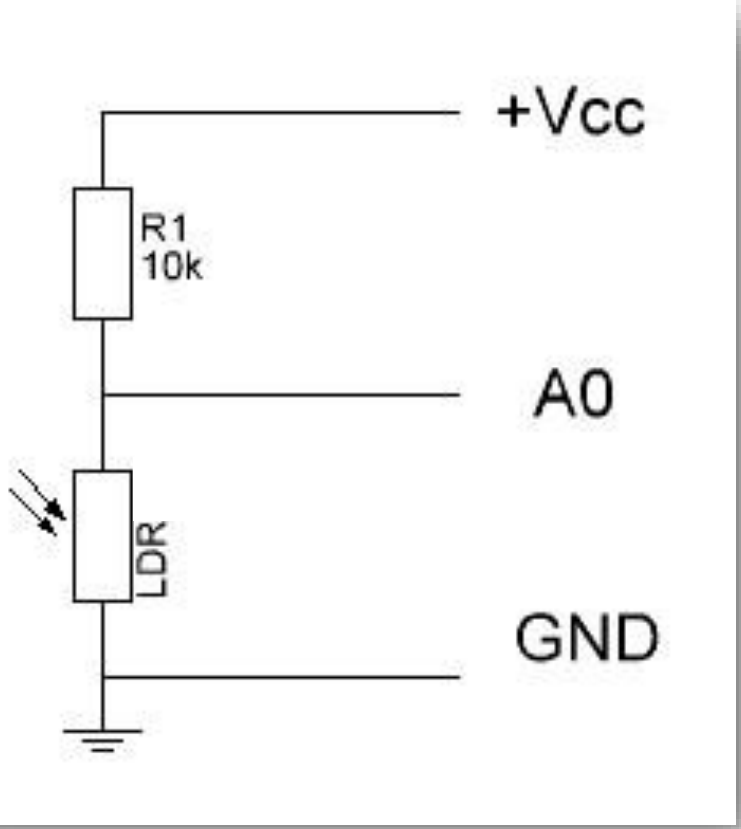
AAnn_CdS

```
1  #define CDS_INPUT 0
2
3  void setup() {
4    Serial.begin(9600);
5  }
6
7  void loop() {
8
9    int value = analogRead(CDS_INPUT);
10   Serial.println(value);
11
12   delay(1000);
13 }
14
```



어두우면 측정 값이 커지고 밝을수록 값이 작아진다 ???

CdS 센서 회로 분석 (1/2)



LDR's (Light dependent resistors) have a low resistance in bright light and a high resistance in the darkness.

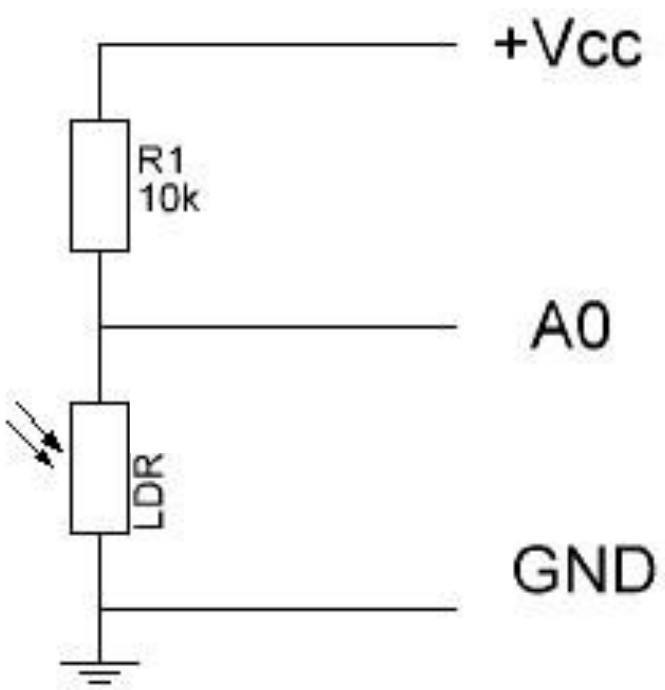
If you would us the LDR as the lower part of a voltage divider, then in darkness there would be a high voltage over the LDR, while in bright light, there would be a low voltage over that resistor.

어두우면 측정 값이 작아지고 밝을수록 값이 커져야 된다.
그리고 측정 값은 **lux**로 표현된다.

$$V_{out} = \frac{R_{ldr}}{R_1 + R_{ldr}} * V_{cc}$$

A0에서 측정되는 **LDR** 양단의 전압 = **Vout**

CdS 센서 회로 분석 (2/2)



$$(a)\ V_{out} = \frac{R_{ldr}}{(R_1 + R_{ldr})} * V_{CC}\ ,$$

$$(b)\ R_{ldr} = \frac{10 * V_{out}}{(5 - V_{out})}\ (k\Omega)\ ,$$

$$(c)\ V_{out} = value * V_{CC}/1023\ ,$$

$$(d)\ Lux = \frac{500}{R_{ldr}}\ ,$$

$$(e)\ Lux = (\frac{2500}{V_{out}} - 500)/10\ (lux).$$

$$V_{out} = \frac{R_{ldr}}{R_1 + R_{ldr}} * V_{cc}$$

A0에서 측정되는 LDR 양단의 전압 = Vout

A3.2.5 Luminosity sensor [sketch-2]

▸ 스케치 구성

1. A0 핀을 CdS 조도 센서의 입력으로 설정한다.
2. setup()에서 직렬 통신 속도를 9600 bps 로 설정하고 컴퓨터와 연결한다.
3. loop()에서 **analogRead**() 함수로 A0 핀에서 측정되는 값을 읽어 들인다.
4. A0 측정값 (0 ~ 1023)을 전압 (0 ~ 5 V)으로 환산한다.
5. 전압 (V)을 온도 (°C)로 환산한 후, A0 측정값, 환산 전압, 환산 조도를 한 줄로 1 초마다 컴퓨터로 전송한다.

A3.2.6 Luminosity sensor [Photocell LDR]

CdS 센서 회로 - 측정 2.

sketch08_CdS2

```
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 Lux
14 double luminosity (int RawADC0){
15   double Vout=RawADC0*5.0/1023;  // 5/1023 (Vin = 5 V)
16   double lux=(2500/Vout-500)/10;
17   // lux = 500 / Rldr, Vout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
18   return lux;
19 }
```

COM11 (Arduino/Genuino Uno)

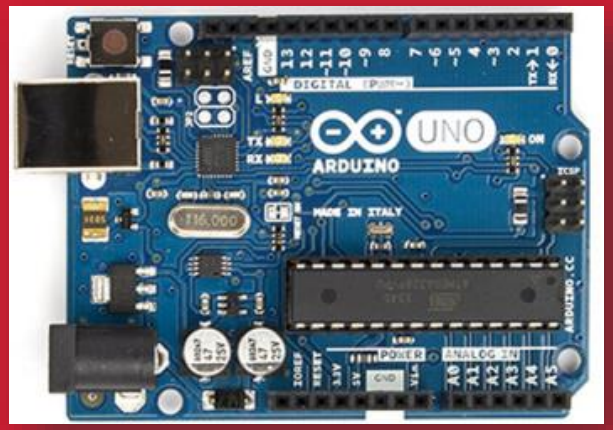
```
23
24
23
183
191
183
185
531
1169
1262
1198
1039
1063
```

어두울 때

밝을 때

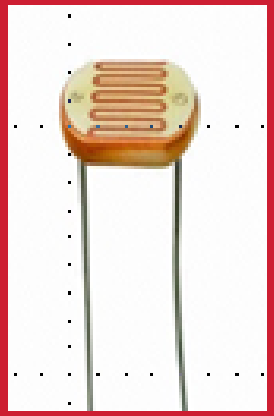
밝을수록 측정 값이 커지고
어두을수록 값이 작아진다 !!!

Single sensor: CdS



CdS (LDR)

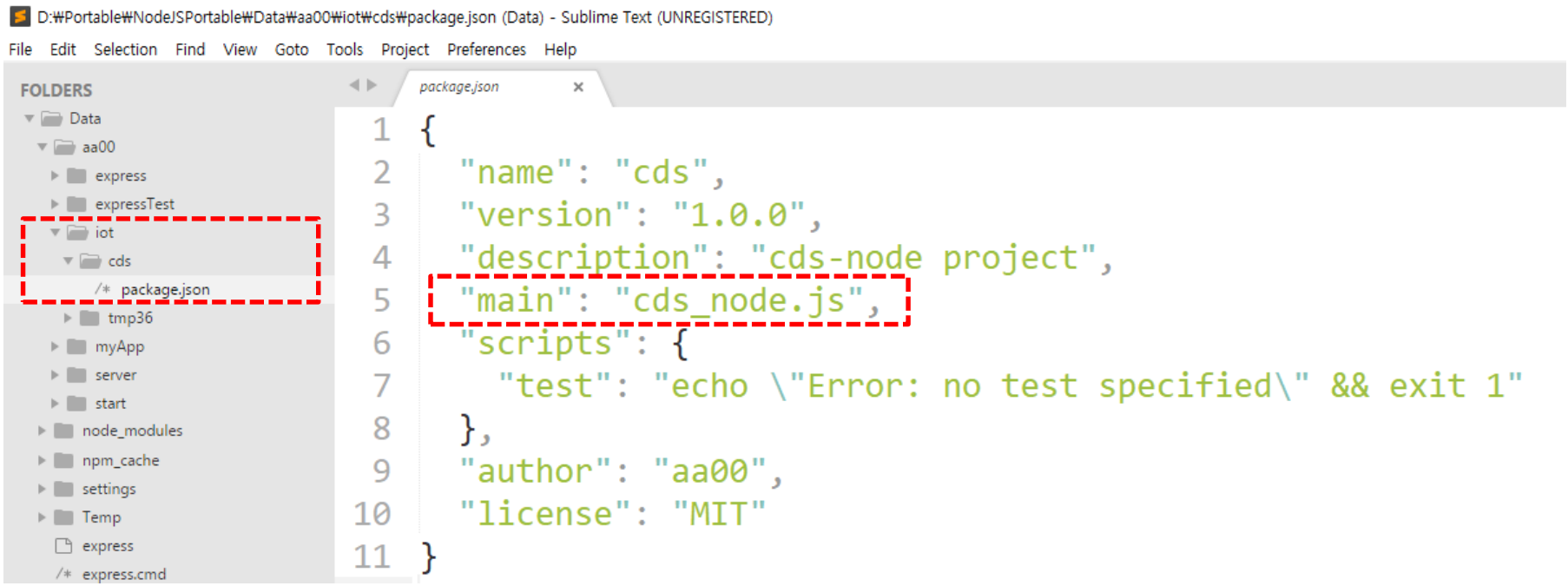
Node project



A4.2.1 Luminosity sensor [Photocell LDR]

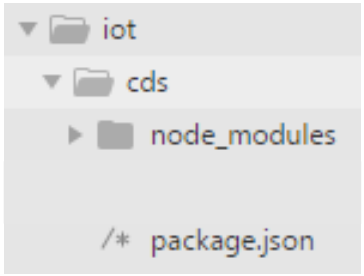
1. Make cds node project
   - ➢ md cds in iot folder
   - ➢ cd cds
2. Go to cds subfolder
   - ➢ npm init

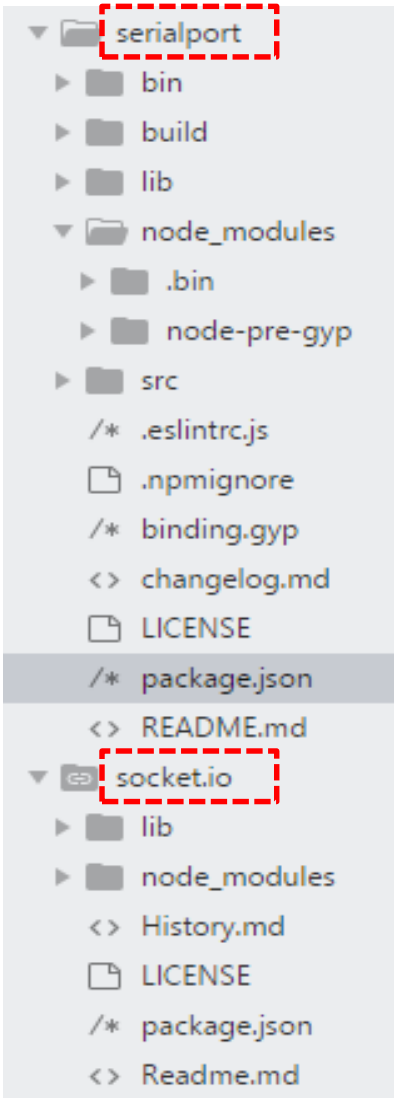
"main": "cds_node.js"
"author": "hsnn"



A4.2.2 Luminosity sensor [Photocell LDR]

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

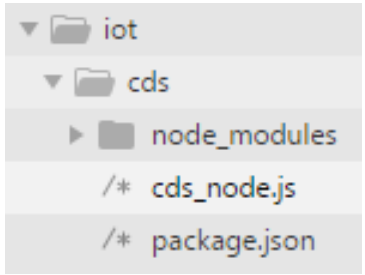
A4.2.3 Luminosity sensor [Photocell LDR]

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

package,json

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

A4.2.4 Luminosity sensor [Photocell LDR]



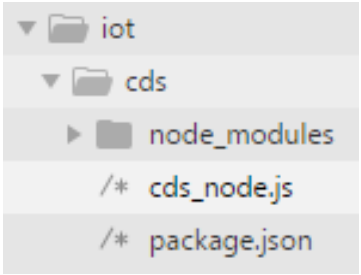
Save tmp36_node.js as cds_node.js

```
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

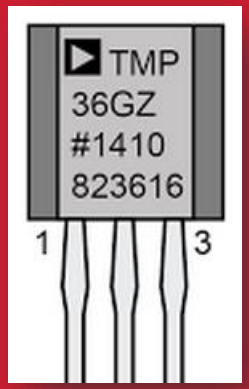
```
NodeJS - node  cds_node

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
```

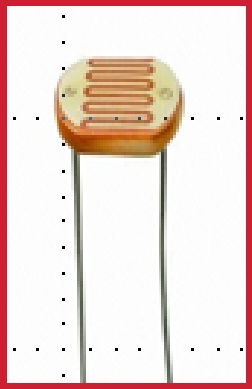
HSnn_cds_IOT_data.png
로 저장 (AA00 → HSnn)

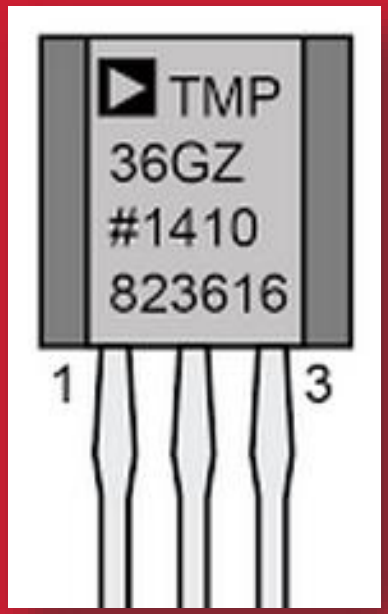
Multiple sensors

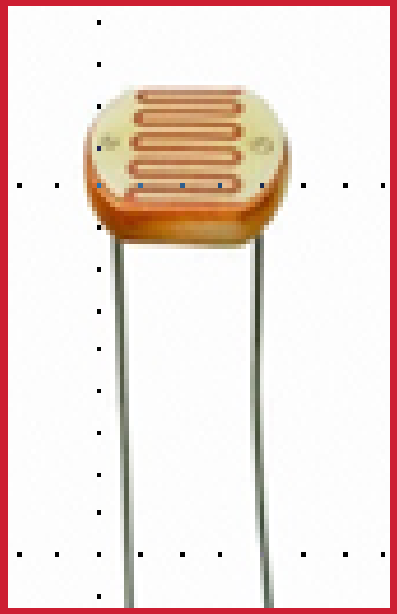
Arduino

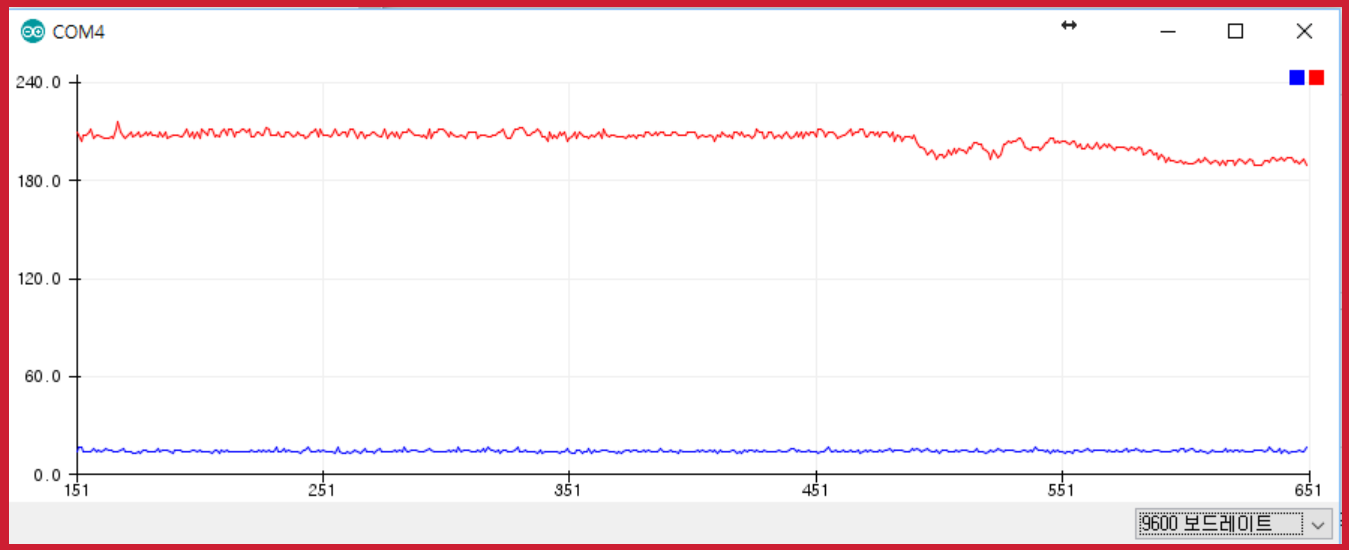
+ Node.js





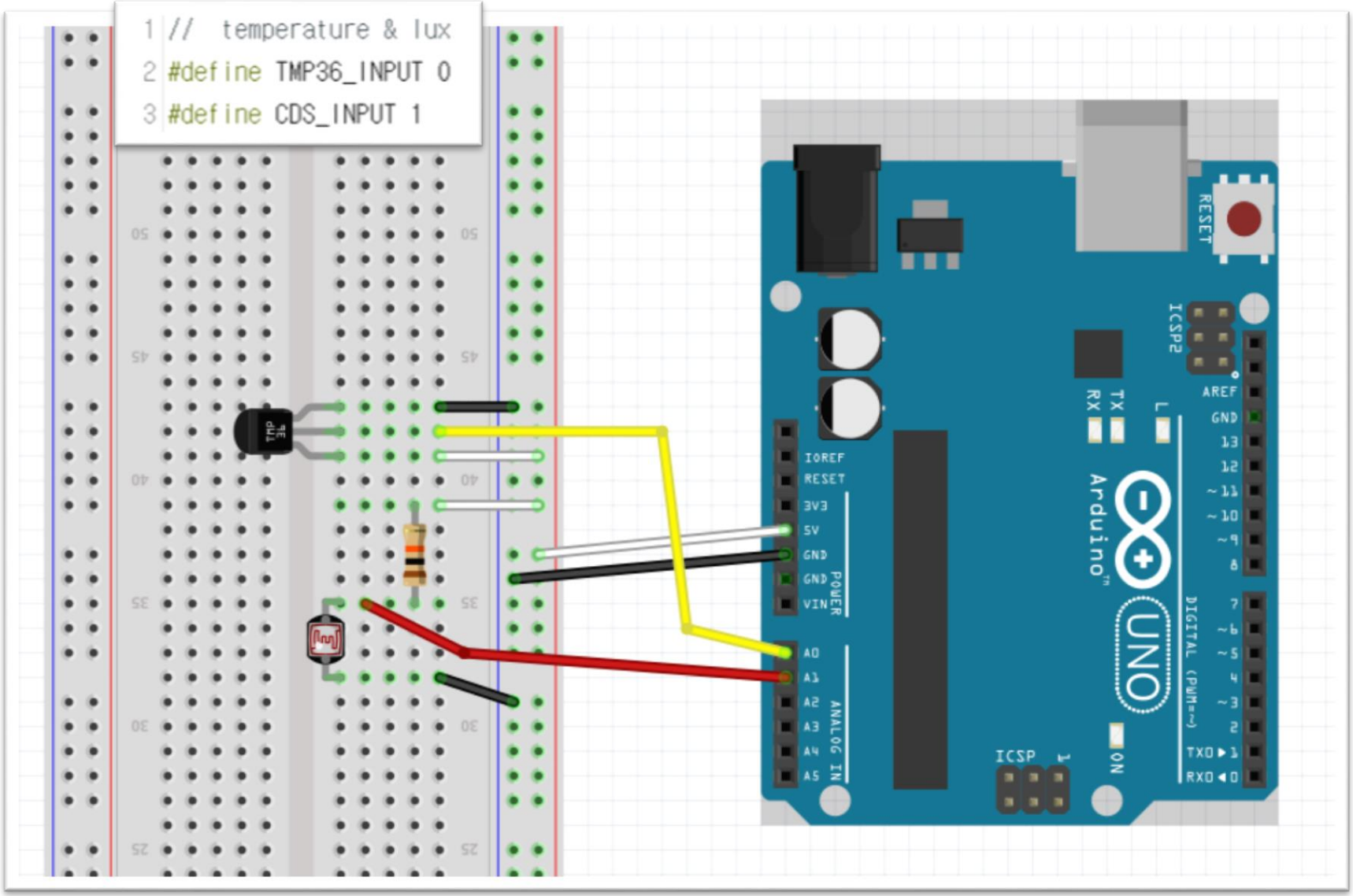

TMP
36GZ
#1410
823616
1
3




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  }
```

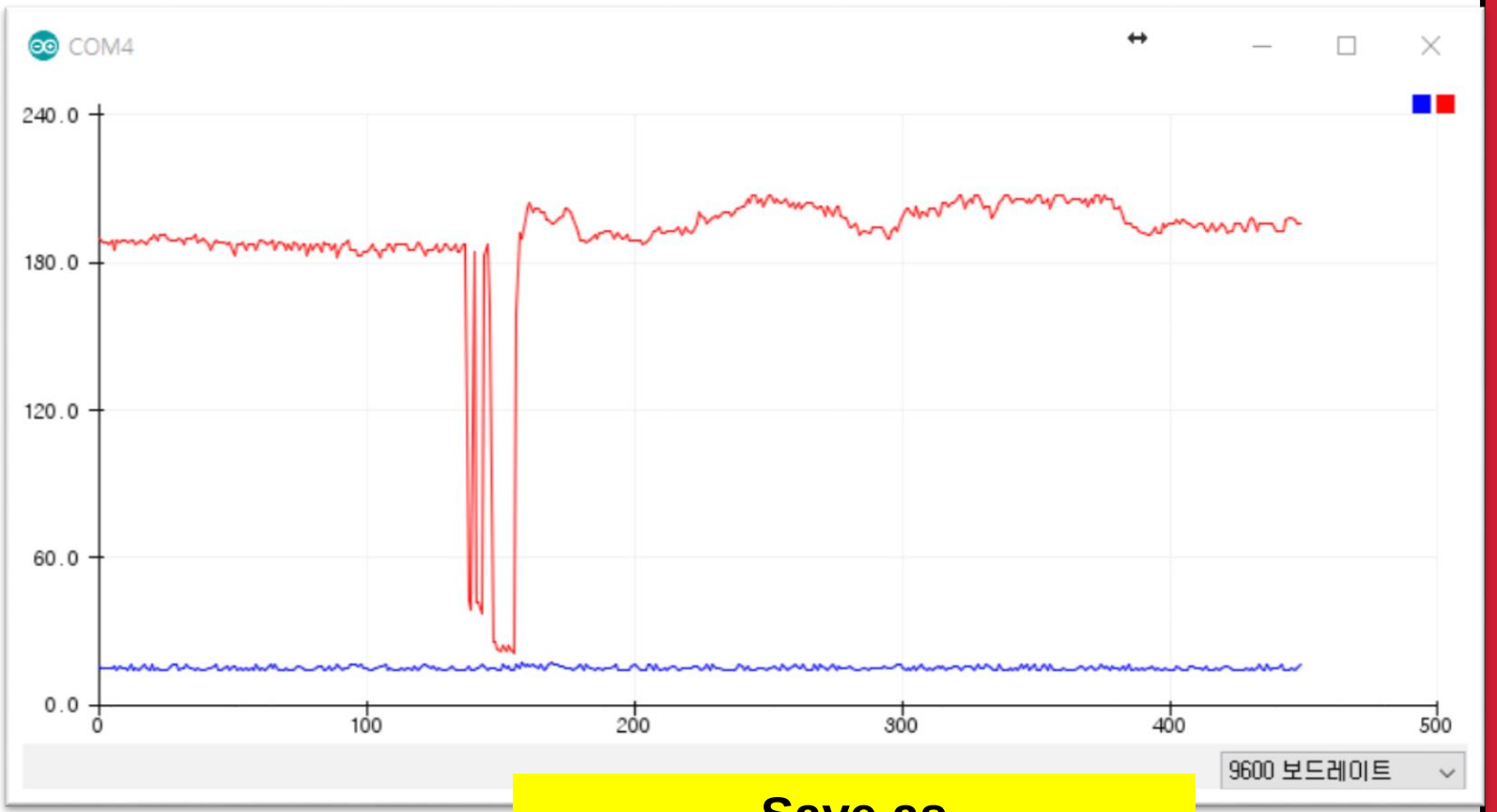
HSnn_tmp36_cds.ino

```
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
20   Serial.print(tempC);
21   Serial.print(",");
22   Serial.println(lux);
23
24   delay(1000);
25 }
26
27 //Voltage to LuxLux
28 double luminosity (int RawADC0){
29   double Vout=RawADC0*0.0048828125;  // 5/1024 (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

[Practice]

- [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)]

Lecture materials

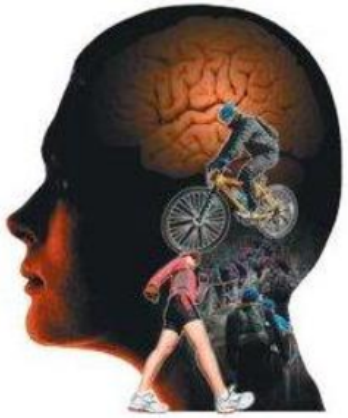
References & good sites

- ✓ 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

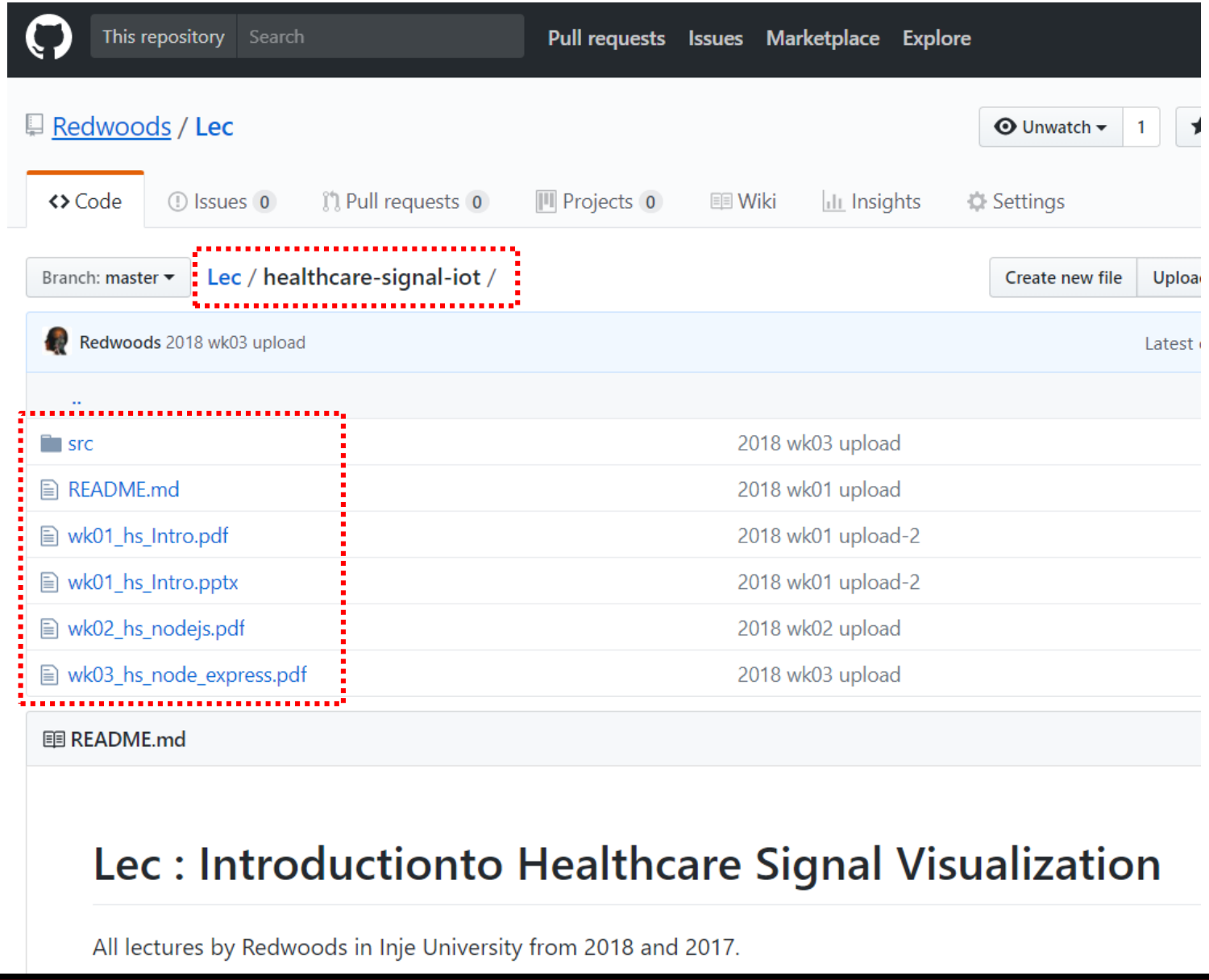
81 commits | 1 branch | 0 releases

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

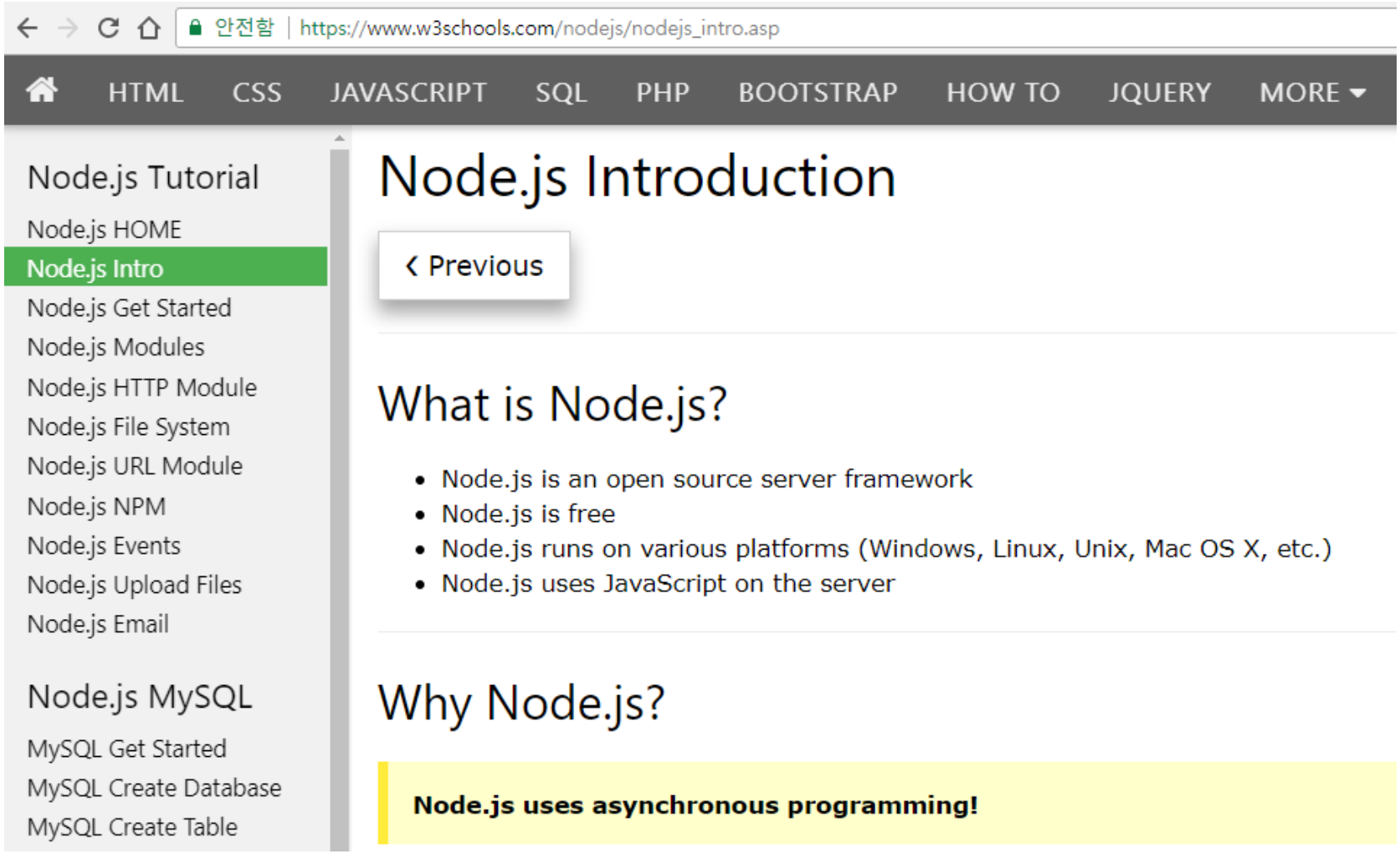
Redwoods 2018 wk01 upload

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

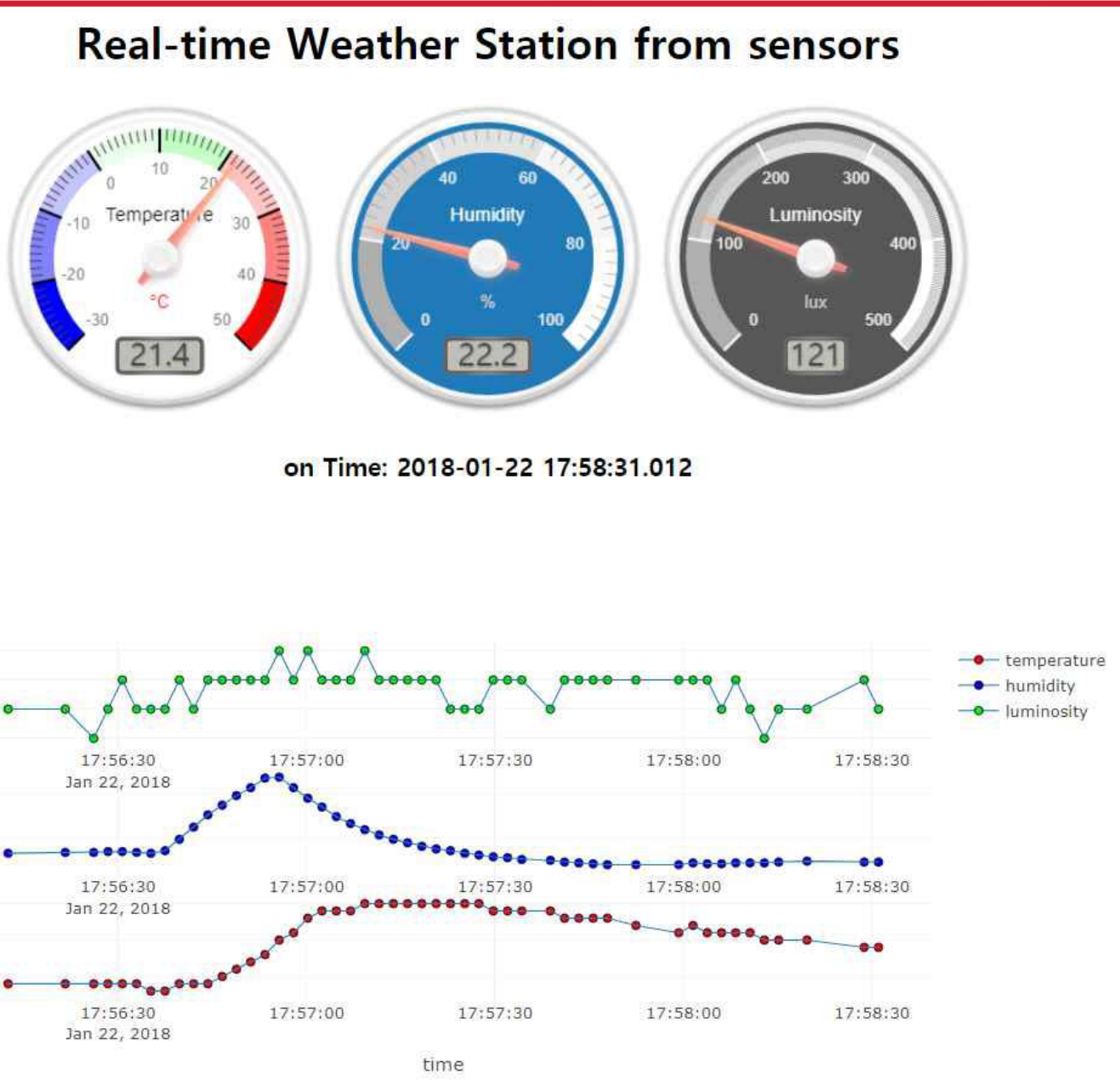


1.0 What is node.js?

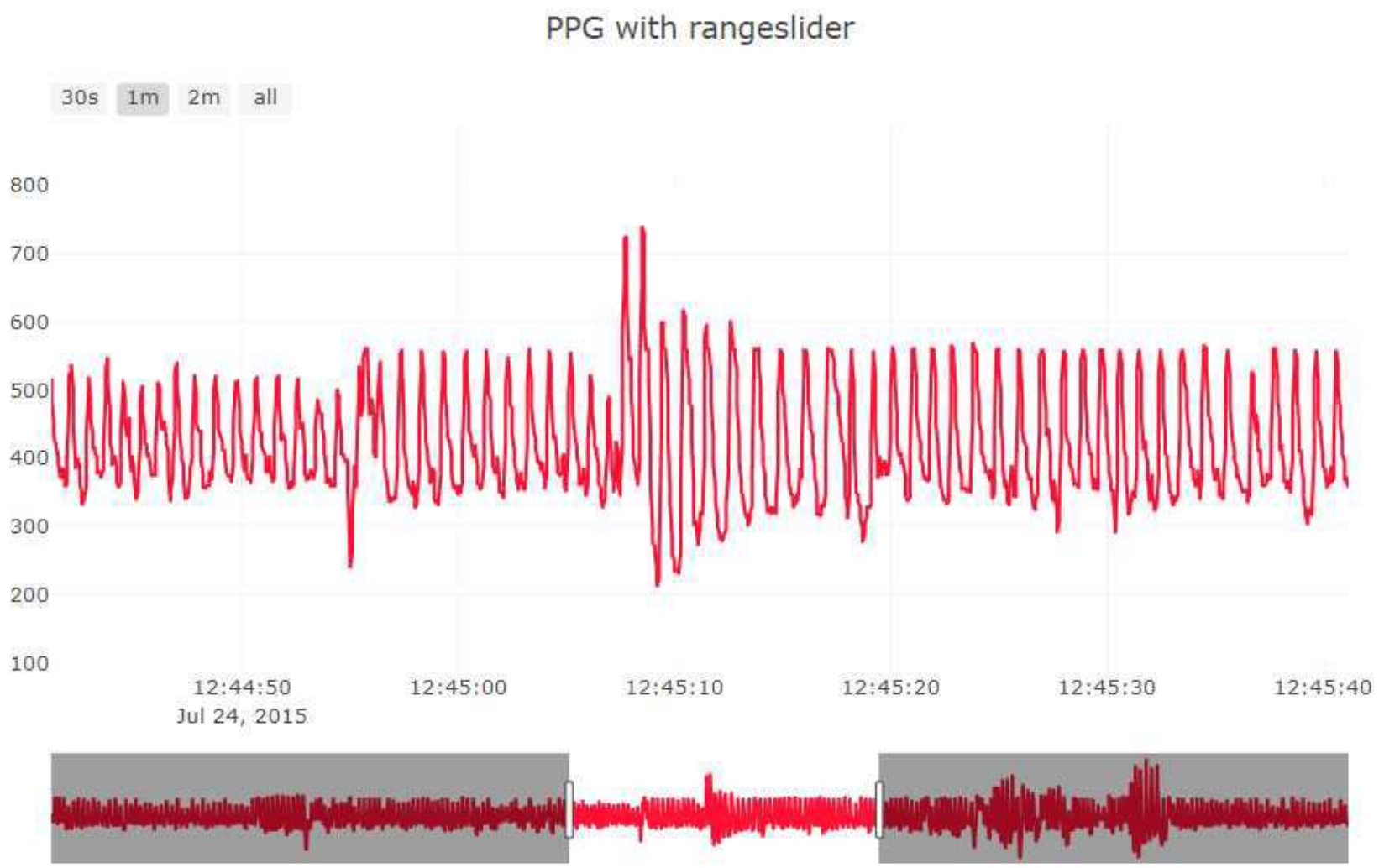


https://www.w3schools.com/nodejs/nodejs_intro.asp

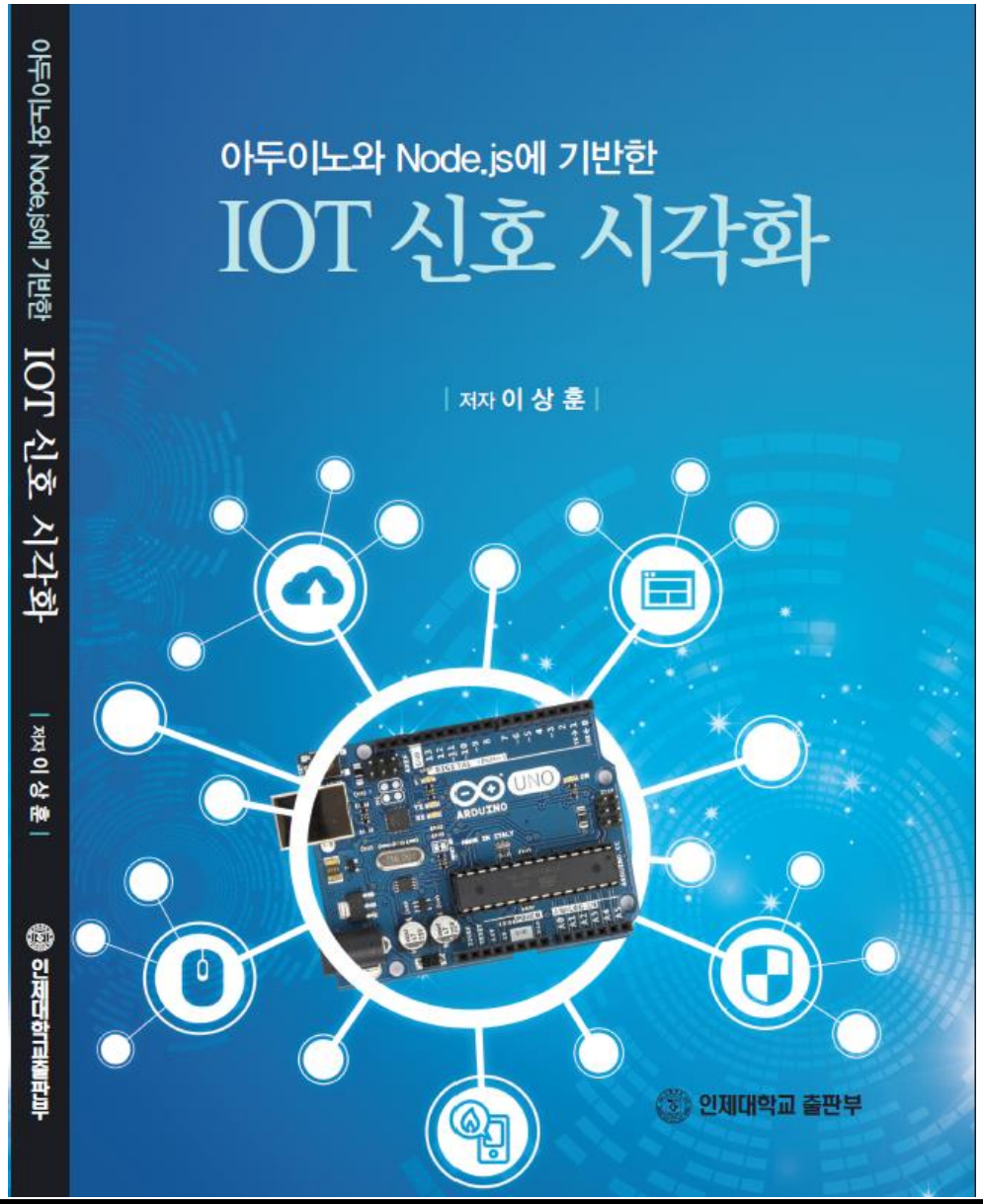
Target of this class



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 |