



# Healthcare-IOT

## [wk06]



# Arduino + Node

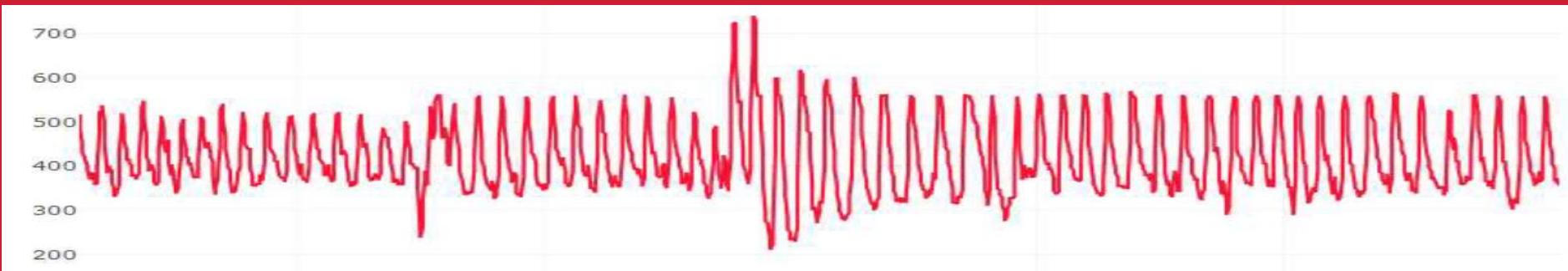
## I. Single signal

Visualization of Healthcare Signals using  
Arduino & Node.js

HCit, INJE University

1<sup>st</sup> semester, 2018

Email : [chaos21c@gmail.com](mailto: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](mailto:chaos21c@gmail.com)

[ 제목 : **id**, 이름 (수정) ]



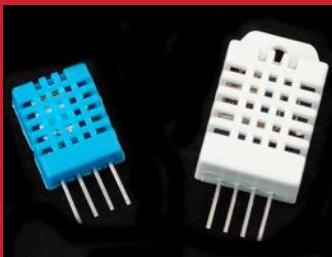
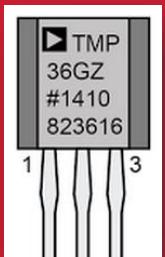
# [My working folder]

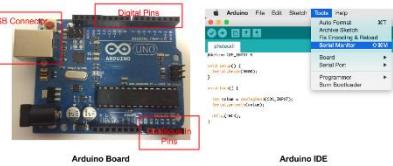
PC > DATA (D:) > Portable > arduino-1.8.5 > hs00 >	
hcit	sketch10_CdS_LCD_lux
HS00_AnalogRead_fmap	sketch10_CdS_LCD_lux_start
HSnn_TMP36_NodeJS	
libraries	
sketch01_blink	
sketch01_start	
sketch02_pwm_led	
sketch03_pwm_led_serial	
sketch04_pwm_3_leds	
sketch05_multi_signals	
sketch06_analog_read	
sketch06_analog_read_start	
sketch07_tmp36	
sketch07_tmp36_start	
sketch08_CdS	
sketch08_CdS_start	
sketch08_CdS2	
sketch09_hello_LCD	
sketch09_hello_LCD_start	

PC > DATA (D:) > Portable > NodeJSPortable > Data > hs00 >	
express	
expressTest	
hs00App	
myApp	
server	
start	

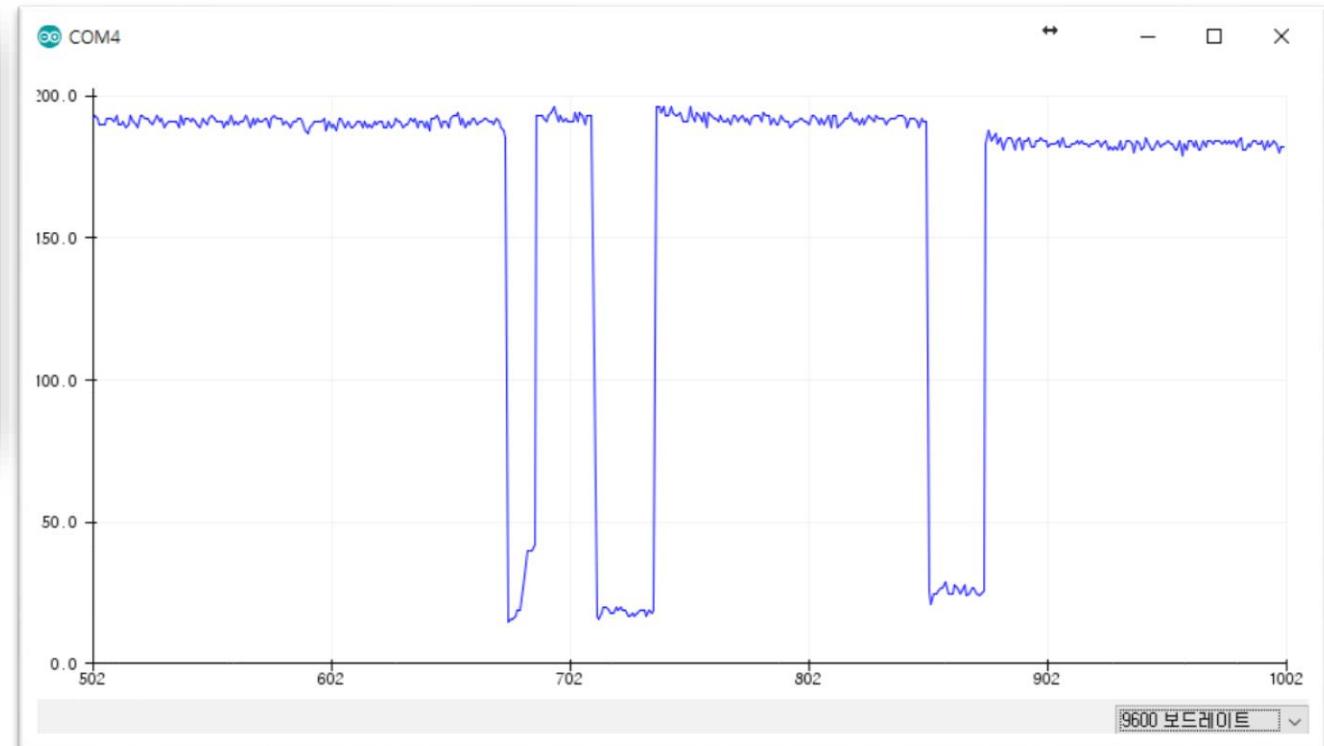
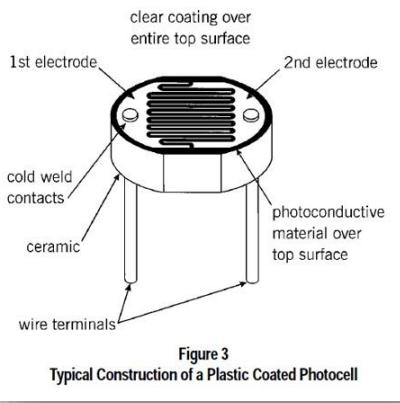


# Arduino + Node.js

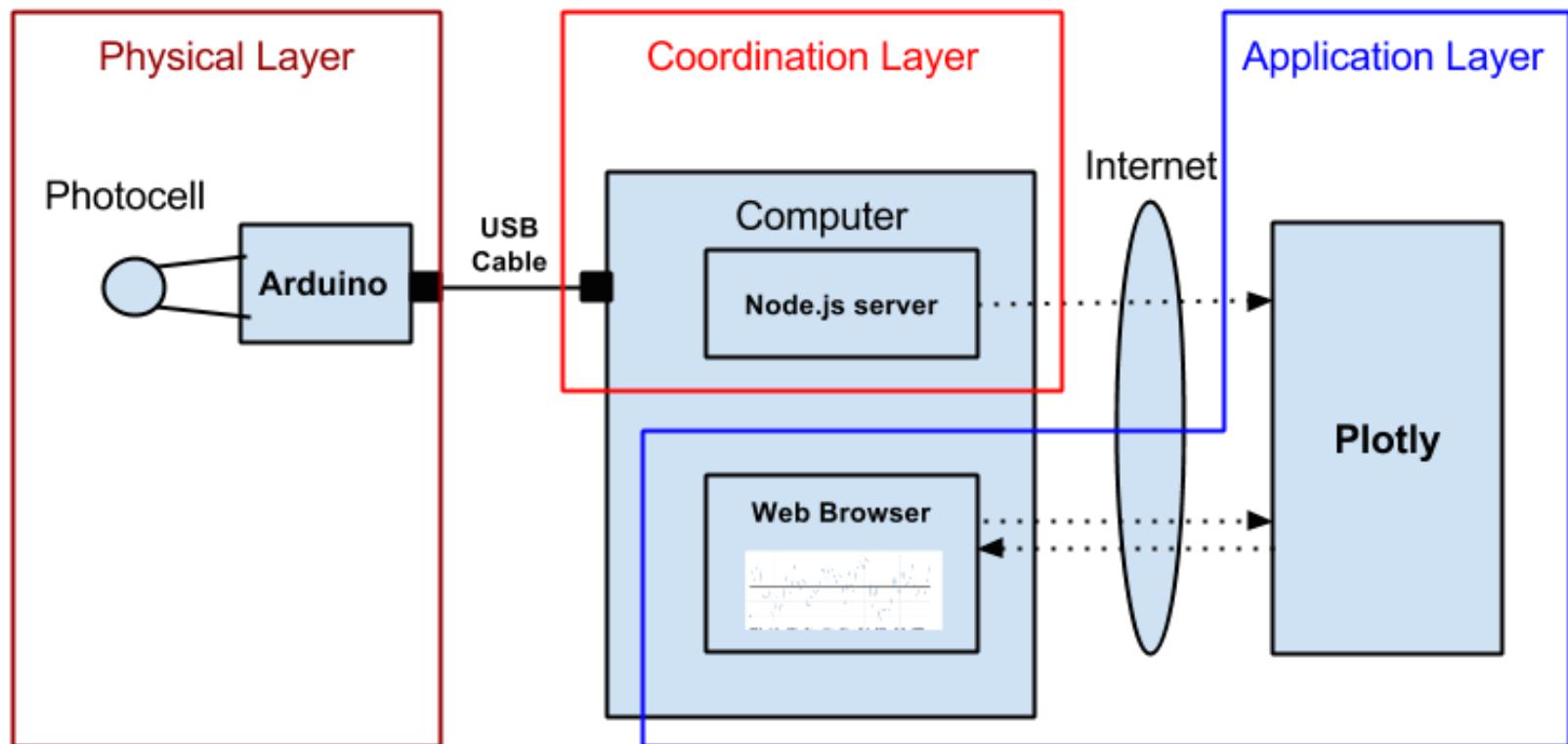




# IOT: HSC



# Layout [H S C]

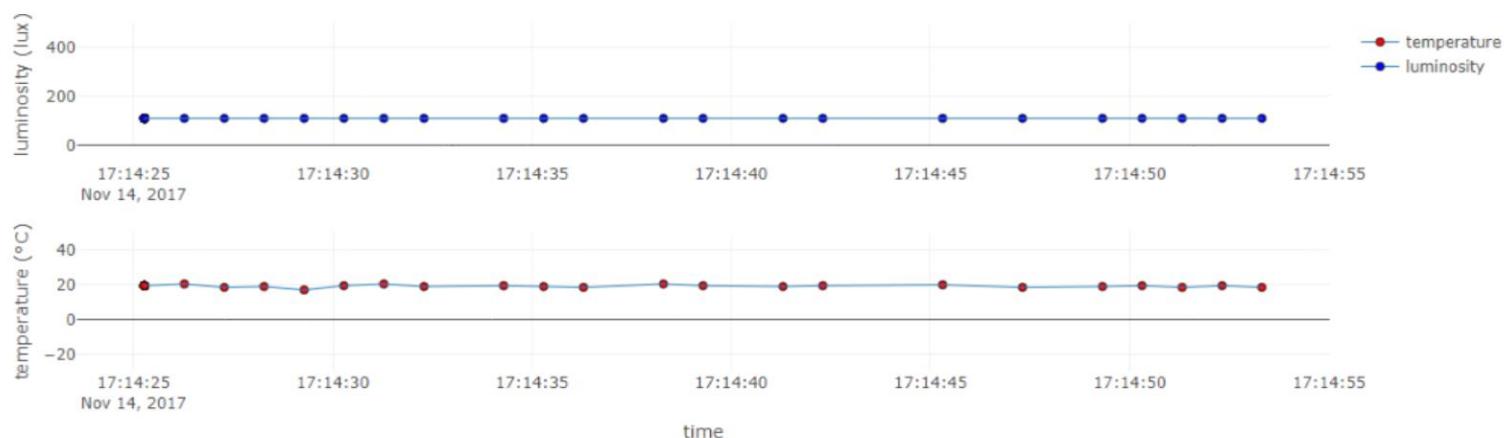


# Arduino: node.js + plotly

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

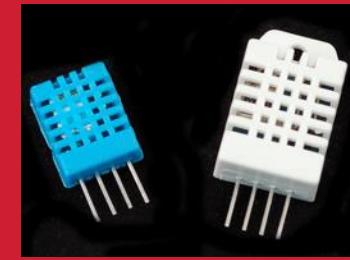
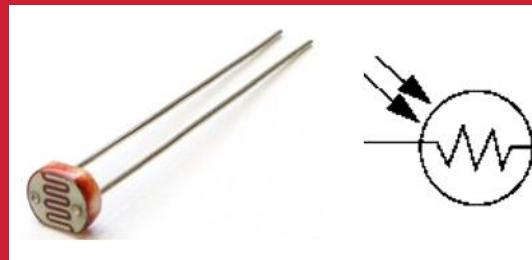
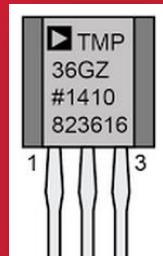


on Time: 2017-11-14 17:14:53.321





# Arduino Sensors

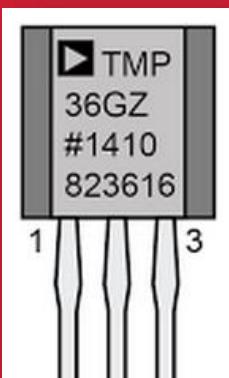




# Single sensor: TMP36

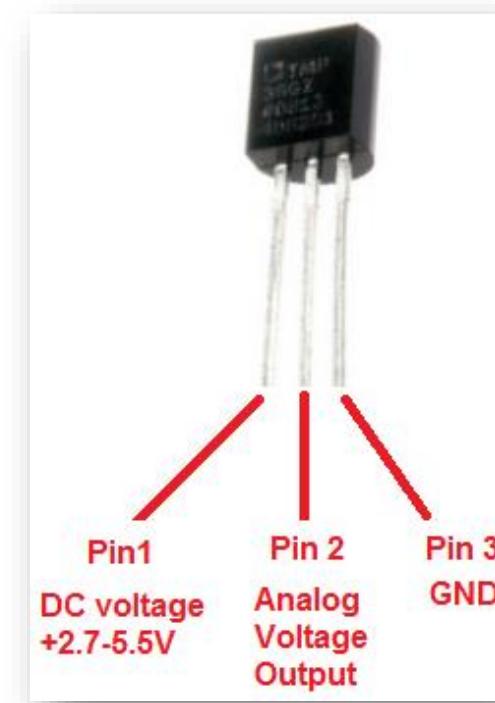
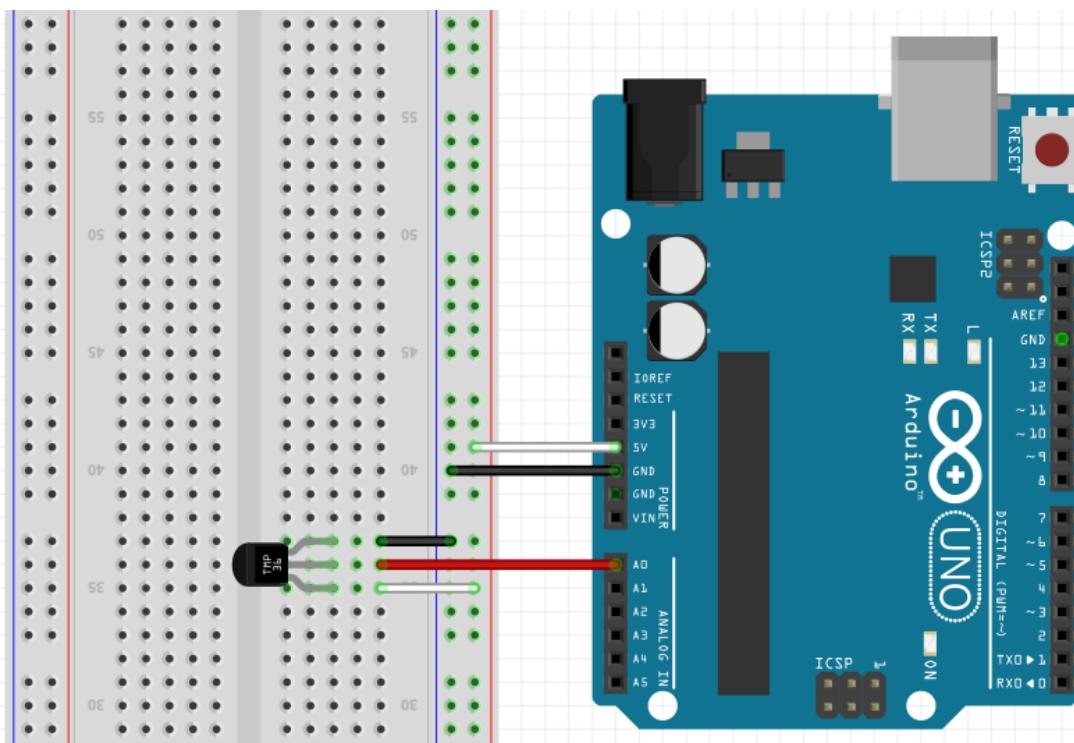


Arduino  
+ Node.js

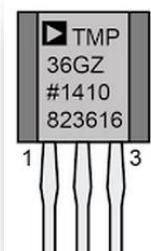




# A3.1.1 Temperature sensor [ TMP36]



## 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$  
1 //  
2 // HS00, TMP36 sensor  
3 //  
4  
5 #define TEMP_INPUT 0  
6 // or int TEMP_INPUT = 0;  
7  
8 void setup() {  
9   Serial.begin(9600);  
10 }  
11  
12 void loop() {  
13  
14   int value = analogRead(TEMP_INPUT);  
15   Serial.println(value);  
16  
17   delay(1000);  
18 }
```

## Serial output (0 ~ 1023)

The screenshot shows the Arduino Serial Monitor window titled "COM8 (Arduino/Genuino Uno)". The window displays a series of integer values: 141, 139, 139, 140, 139, 141, 141, 139, 140, 139, 139, 139, 141, 139, 139, 141. A gray callout box with a black border and white text is overlaid on the monitor, containing the text "This is NOT real temperature!".

```
141  
139  
139  
140  
139  
141  
141  
139  
140  
139  
139  
139  
141  
139  
139  
141
```

**This is NOT  
real  
temperature!**

자동 스크롤  No line ending ▾ 9600 보드 레이트

# A3.1.3 Temperature sensor [ TMP36]

## Sensor property

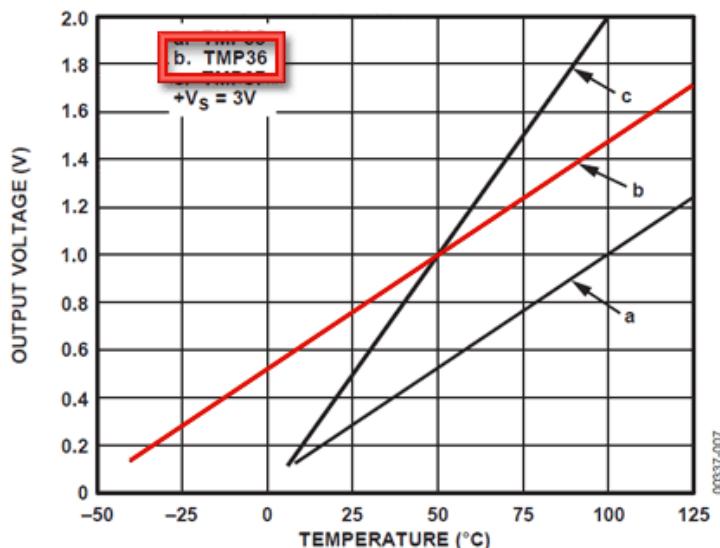


Figure 6. Output Voltage vs. Temperature

## Temperature conversion

$$\text{Temp } (\text{ }^{\circ} \text{ C}) = (\text{Vout} - 500) / 10$$

$$\text{Vout (mV)} = \text{value} * (5000 / 1023)$$
$$(0 \leq \text{value} \leq 1023)$$



```
// 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

```
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 }
```

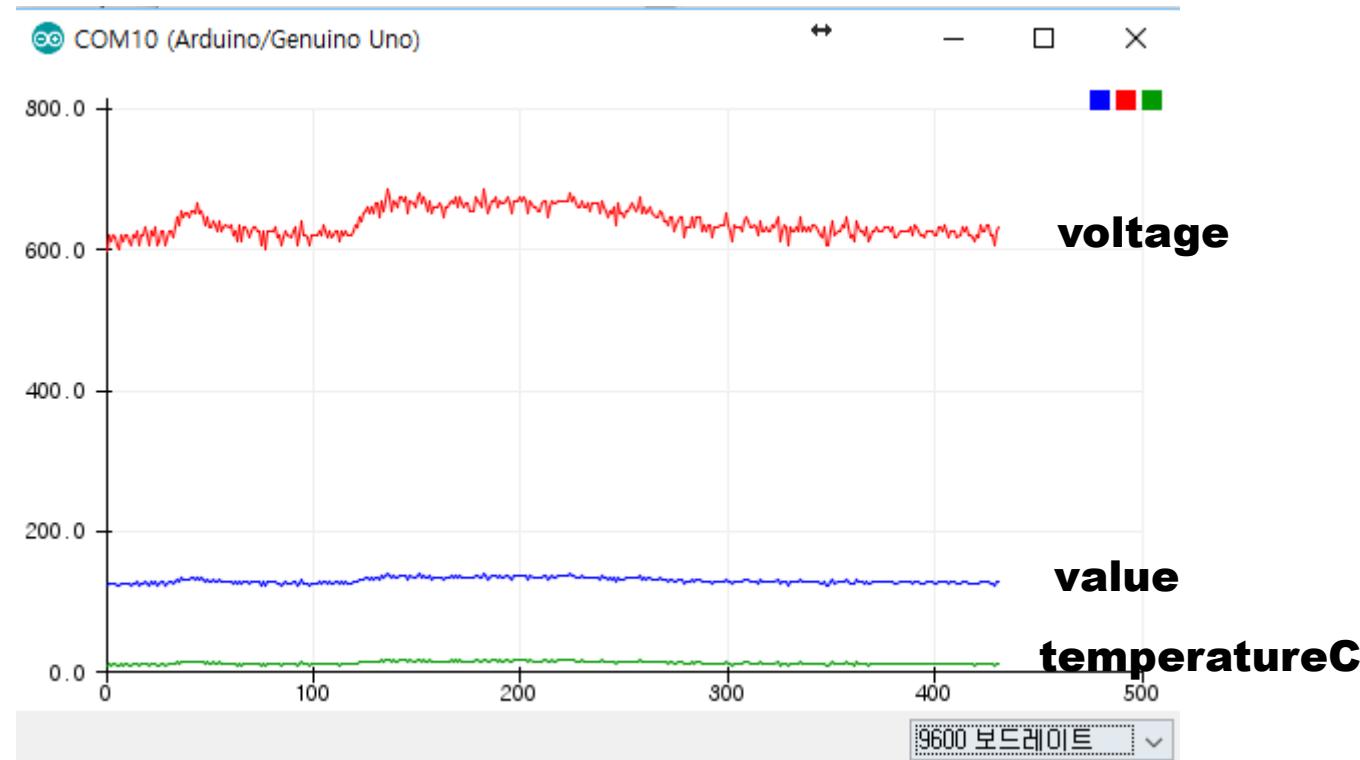
## Serial output ( °C)

The screenshot shows the Arduino Serial Monitor window titled "COM4". It displays a series of temperature readings printed in a repeating loop. Each reading consists of the string "AA00, value = ", followed by a numerical value, and then "mV, " and "degrees C". The values fluctuate between 12.56 and 14.52 degrees Celsius.

```
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
```

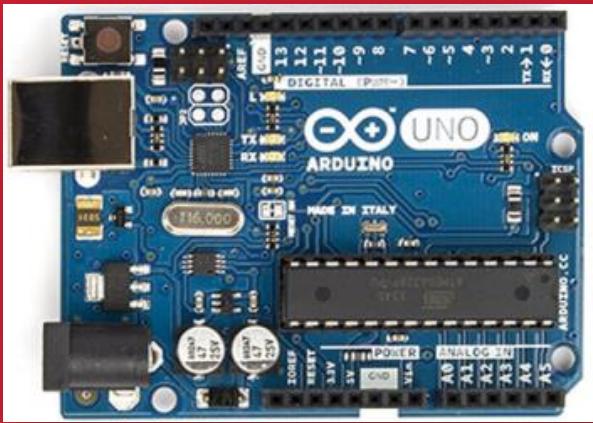


## A3.1.5 Temperature sensor [ TMP36]



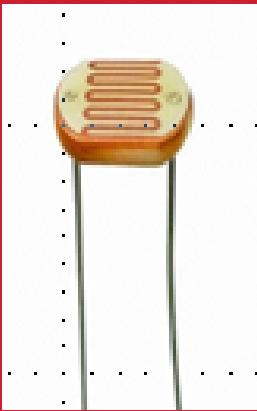


# Single sensor: tmp36



# 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**

cd 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

```
cmd 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 ×
1 {
2   "name": "tmp36",
3   "version": "1.0.0",
4   "description": "tmp36-node project",
5   "main": "tmp36_node.js",
6   "scripts": {
7     "test": "echo \\\"Error: no test specified\\\" && exit 1"
8   },
9   "keywords": [
10     "tmp36",
11     "node",
12     "arduino"
13   ],
14   "author": "aa00",
15   "license": "MIT"
16 }
```



# 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

```
1 {  
2   "name": "tmp36",  
3   "version": "1.0.0",  
4   "description": "tmp36-node project",  
5   "main": "tmp36_node.js",  
6   "scripts": {  
7     "test": "echo \\\"Error: no test specified\\\" && exit 1"  
8   },  
9   "keywords": [  
10     "tmp36",  
11     "node",  
12     "arduino"  
13   ],  
14   "author": "aa00",  
15   "license": "MIT",  
16   "dependencies": {  
17     "serialport": "^6.0.4",  
18     "socket.io": "^2.0.4"  
19   }  
20 }  
21
```



# 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/...serialport/.../...> ▾ 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

Google search results for "TypeError: serialport.parsers.readline is not a function". The results show various links related to the error, including issues on GitHub and Stack Overflow, and documentation from the node-serialport package.

TypeError: serialport.parsers.readline is not a function nodejs

전체 동영상 뉴스 이미지 더보기 설정 도구

검색결과 약 3,020개 (0.66초)

도움말: 한국어 검색결과만 검색합니다. 환경설정에서 검색 언어를 지정할 수 있습니다.

[TypeError: SerialPort.parsers.ReadLine is not a function · Issue #937 ...](#)

<https://github.com/EmergingTechnologyAdvisors/...serialport/.../> ▾ 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 ...

[SerialPort lib - "parsers.readline is not a function" Error - NodeJS](#)

<https://stackoverflow.com/.../serialport-lib-parsers-readline-is-not-...> ▾ 이 페이지 번역하기

2017. 9. 3. - If I see it right Readline is a class **not function!** Try this: parser: **SerialPort.parsers**.

**Readline**. Check this out and let me know if it works!

이 페이지를 2번 방문했습니다. 최근 방문 날짜: 17. 10. 31

[javascript - TypeError: serialport.parsers.readline is not a function ...](#)

<https://stackoverflow.com/.../typeerror-serialport-parsers-readline-...> ▾ 이 페이지 번역하기

The documentation will tell you that **Readline** is spelled with a capital R. [https://www.npmjs.com/package/serialport#module\\_serialport--SerialPort.parsers](https://www.npmjs.com/package/serialport#module_serialport--SerialPort.parsers)

[Nodejs Error "SerialPort is not a function...." with node-serialport ...](#)

[community.onion.io › Omega Talk](https://community.onion.io/.../Omega Talk) ▾ 이 페이지 번역하기

2017. 8. 25. - Re: Serial port communication using Node.js @Steven-de-Salas Hello I ... new

[SerialPort\('/dev/ttyS0', ^](https://www.npmjs.com/package/serialport) **TypeError: SerialPort is not a function.**

[serialport - npm](#)

<https://www.npmjs.com/package/serialport> ▾ 이 페이지 번역하기



## A4.1.6A tmp36 node project → downgrade

Go to tmp36 subfolder (after deleting node\_modules subfolder)

- “dependencies” 속성의 버전을 아래와 같이 변경
- npm install

The screenshot shows a code editor with a file tree on the left and a code editor window on the right. The file tree shows a folder structure under 'iot' with subfolders 'cds', 'cds\_dht22', 'cds\_tmp36', 'flame', 'plotly', and a 'tmp36' folder containing 'node\_modules', 'package.json', and 'tmp36\_node.js'. The 'package.json' file is open in the code editor.

```
1 {  
2   "name": "tmp36",  
3   "version": "1.0.0",  
4   "description": "tmp36-node project",  
5   "main": "tmp36_node.js",  
6   "scripts": {  
7     "test": "echo \\\"Error: no test specified\\\" && exit 1"  
8   },  
9   "keywords": [  
10     "tmp36",  
11     "node",  
12     "arduino"  
13   ],  
14   "author": "aa00",  
15   "license": "MIT",  
16   "dependencies": {  
17     "serialport": "^6.0.4",  
18     "socket.io": "^2.0.4"  
19   }  
20 }  
21
```

A red dashed box highlights the 'dependencies' section of the JSON object. An arrow points from this section to a callout box containing the updated dependency versions:

```
"serialport": "^4.0.7",  
"socket.io": "^1.7.3"
```

At the bottom of the slide, a yellow bar contains the text: "serialport 6.x 버전의 API 변화로 오류 발생, 버전 downgrade".

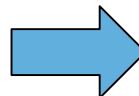
serialport 6.x 버전의 API 변화로 오류 발생, 버전 downgrade



## 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)

```
tmp36
  node_modules
    /* client.js
    /* package.json
    /* package_new.json
    /* tmp36_node.js
12      dataBits: 8,
13      parity: 'none',
14      stopBits: 1,
15      flowControl: false,
16      parser: serialport.
17  }):
AA00, value = 128
AA00, value = 125
AA00, value = 130
AA00, value = 131
AA00, value = 130
AA00, value = 131
AA00, value = 128
AA00, value = 130
AA00, value = 130
AA00, value = 130
AA00, value = 128
AA00, value = 130
```

Serial monitor를  
중단한 후에 ^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

COM4

13.54
11.58
12.56
12.56
12.56
12.56
14.03
11.58
13.54
13.54
12.56
14.52
15.00
16.47
15.49

12.56
12.56
13.05
12.56
12.56
12.56
12.56
15.49
13.54
13.54
12.56
14.52
15.00
16.47
15.49

SB3에서 tmp36\_node.js를  
^B로 실행하여 동일한 결과를  
얻으시오.



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

## tmp36\_node.js

```
19 var dStr = '';
20 var tdata = []; // Array
21
22 sp.on('data', function (data) { // call back when data is received
23     // raw data only
24     //console.log(data);
25     dStr = getDateString();
26     tdata[0] = dStr; // date
27     tdata[1] = data; // data
28     console.log("H500," + tdata);
29     io.sockets.emit('message', tdata); // send data to all clients
30 });
31
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
```

## 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
```

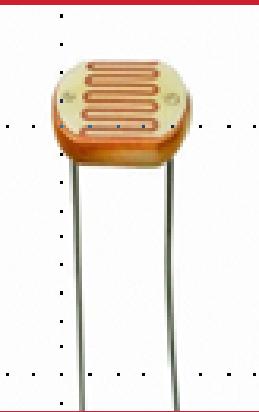
HSnn\_tmp36\_IOT\_data.png  
로 저장 (AA00 → HSnn)



# Single sensor: Cds

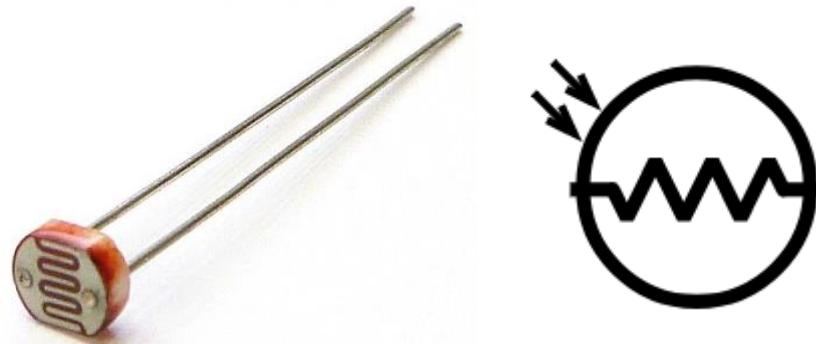


# Arduino + Node.js

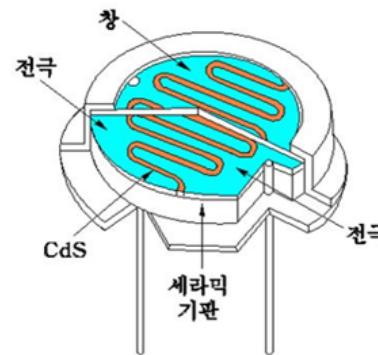


# A3.2 Luminosity sensor [ Photocell LDR]

## CdS 센서- photoresistor



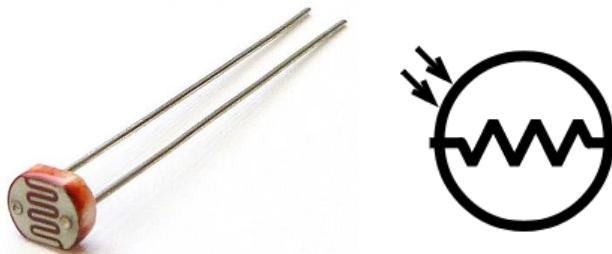
### CDS특성



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

# A3.2.1 Luminosity sensor [ Photocell LDR]

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

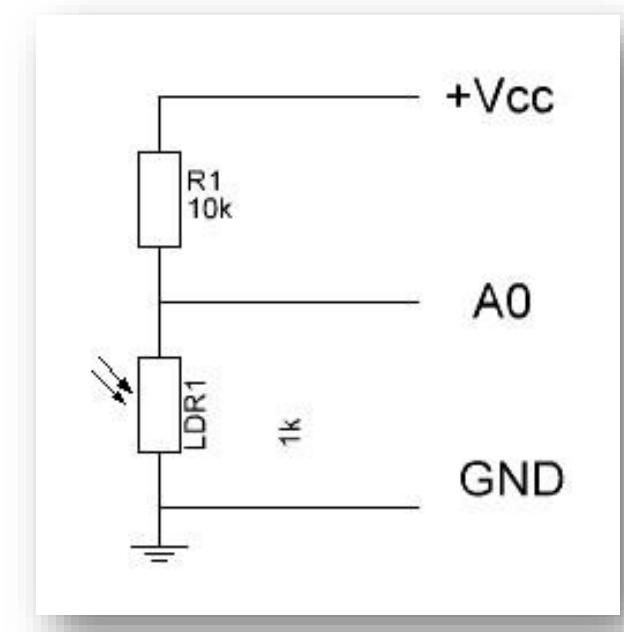
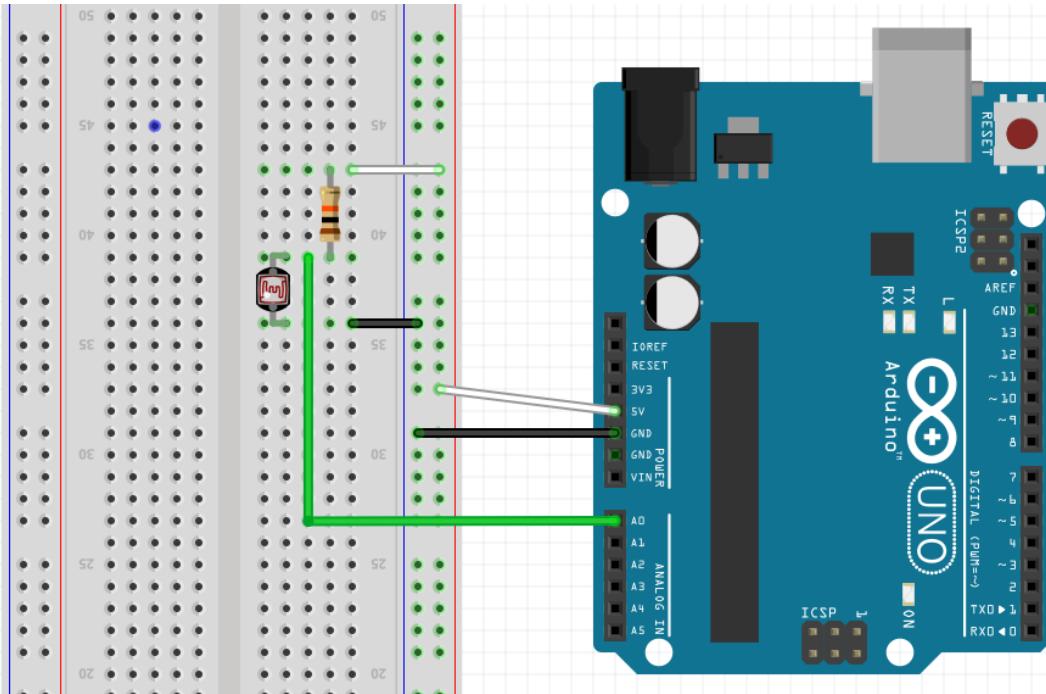
#### 럭스의 예 [\[편집\]](#)

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



## A3.2.2 Luminosity sensor [ Photocell LDR ]

CdS 센서 회로



Parts : 20 mm photocell LDR, R ( $10 \text{ k}\Omega \times 1$ )

광센서에서의 전압 강하 값을 A0로 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

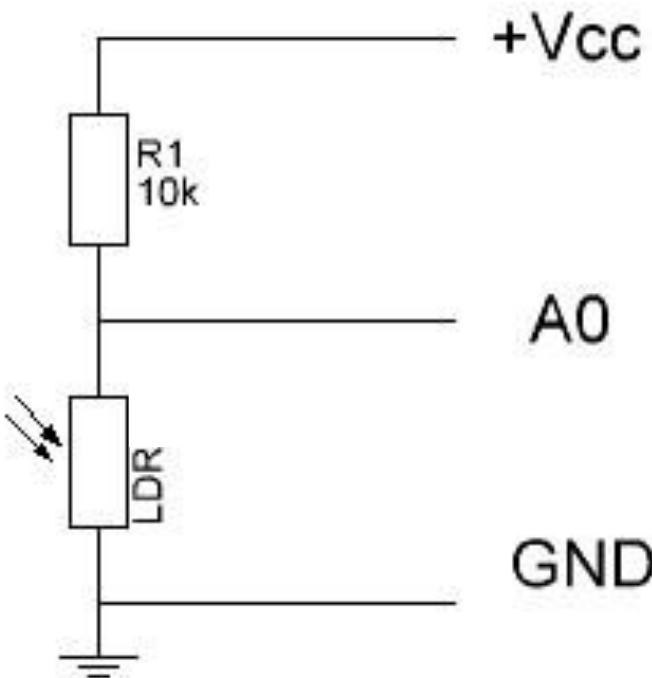
```
1 #define CDS_INPUT 0
2
3 void setup() {
4     Serial.begin(9600);
5 }
6
7 void loop() {
8
9     int value = analogRead(CDS_INPUT);
10    Serial.println(value);
11
12    delay(1000);
13 }
14
```

COM11 (Arduino/Genuino Uno)

672	어
672	두
671	울
669	
209	때
205	
207	때
207	
205	밝
207	을
62	
59	때
53	

어두으면 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.**

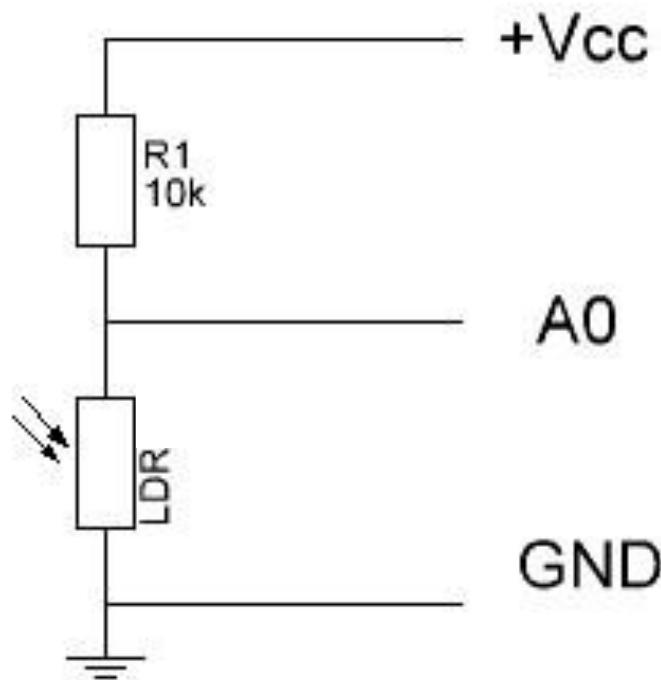
**If you would us the LDR as the lower part of a voltage divider, then in darkness there would be a high voltage over the LDR, while in bright light, there would be a low voltage over that resistor.**

어두우면 측정 값이 작아지고 밝을수록 값이 커져야 된다.  
그리고 측정 값은 lux로 표현된다.

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

A0에서 측정되는 LDR  
양단의 전압 =  $V_{out}$

## 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 =  $V_{out}$**



## 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 ( $^{\circ}\text{C}$ )로 환산한 후, A0 측정값, 환산 전압, 환산 조도를 한 줄로 1 초마다 컴퓨터로 전송한다.

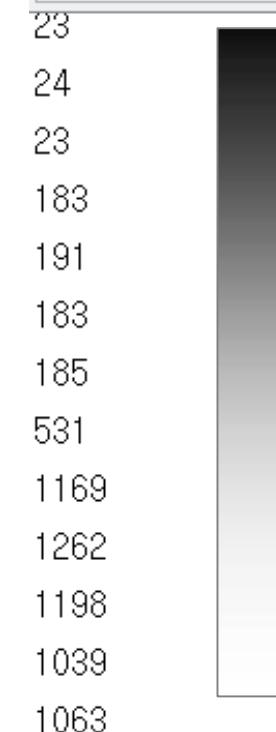


## A3.2.6 Luminosity sensor [ Photocell LDR]

### 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)



밝을수록 측정 값이 커지고  
어두울수록 값이 작아진다 !!!

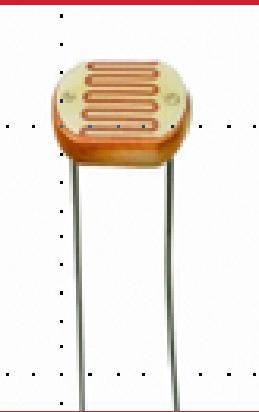


# Single sensor: Cds



# 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"**



D:\#Portable#\NodeJS\Portable#\Data\aa00#\iot\cds\package.json (Data) - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

FOLDERS

```
1 {  
2   "name": "cds",  
3   "version": "1.0.0",  
4   "description": "cds-node project",  
5   "main": "cds_node.js",  
6   "scripts": {  
7     "test": "echo \\"Error: no test specified\\" && exit 1"  
8   },  
9   "author": "aa00",  
10  "license": "MIT"  
11 }
```

The screenshot shows a Sublime Text window with the file structure on the left and the package.json content on the right. A red dashed box highlights the 'iot' folder and its 'cds' subfolder, which contains the package.json file. Another red dashed box highlights the 'main' field in the JSON code.



## A4.2.2 Luminosity sensor [ Photocell LDR]

### 1. Make cds node project

➤ md cds in iot folder

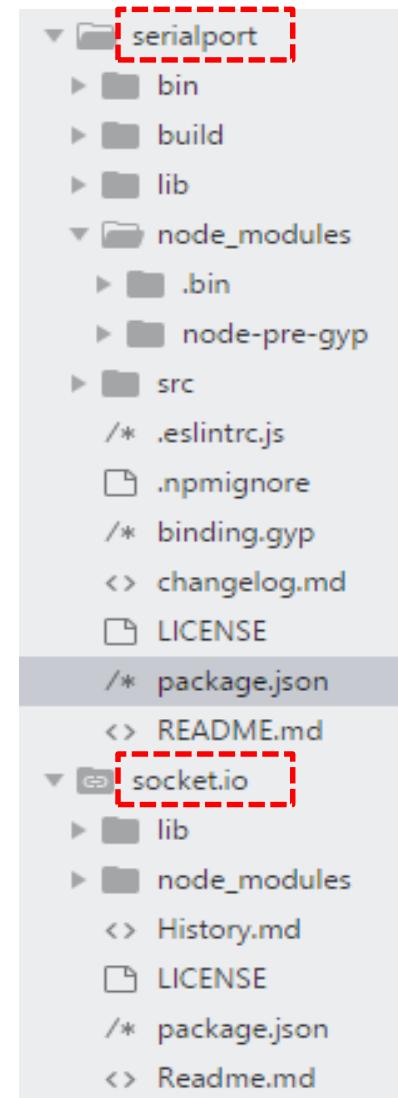
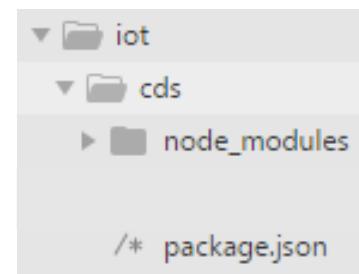
➤ cd cds

### 2. Go to cds subfolder

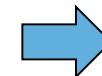
➤ npm init

➤ npm install –save serialport@4.0.7

➤ npm install –save socket.io@1.7.3



You can check version of each module by browsing 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

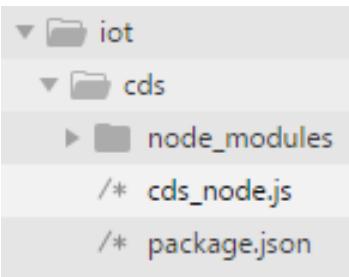
- **npm init**
- **npm install --save serialport@4.0.7**
- **npm install --save socket.io@1.7.3**

### 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
});

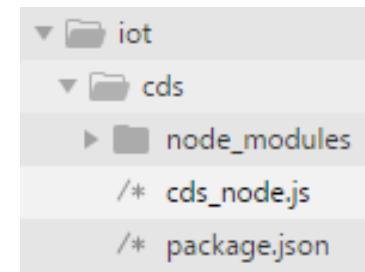
// helper function to get a nicely formatted date string
function getDateString() {
    var time = new Date().getTime();
    // 32400000 is (GMT+9 Korea, GimHae)
    // for your timezone just multiply +/-GMT by 3600000
    var datestr = new Date(time +32400000).
        toISOString().replace(/\T/, '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

```
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
```

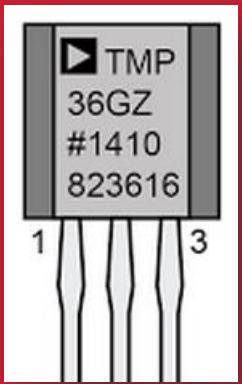
HSnn\_cds\_IOT\_data.png  
로 저장 (AA00 → HSnn)

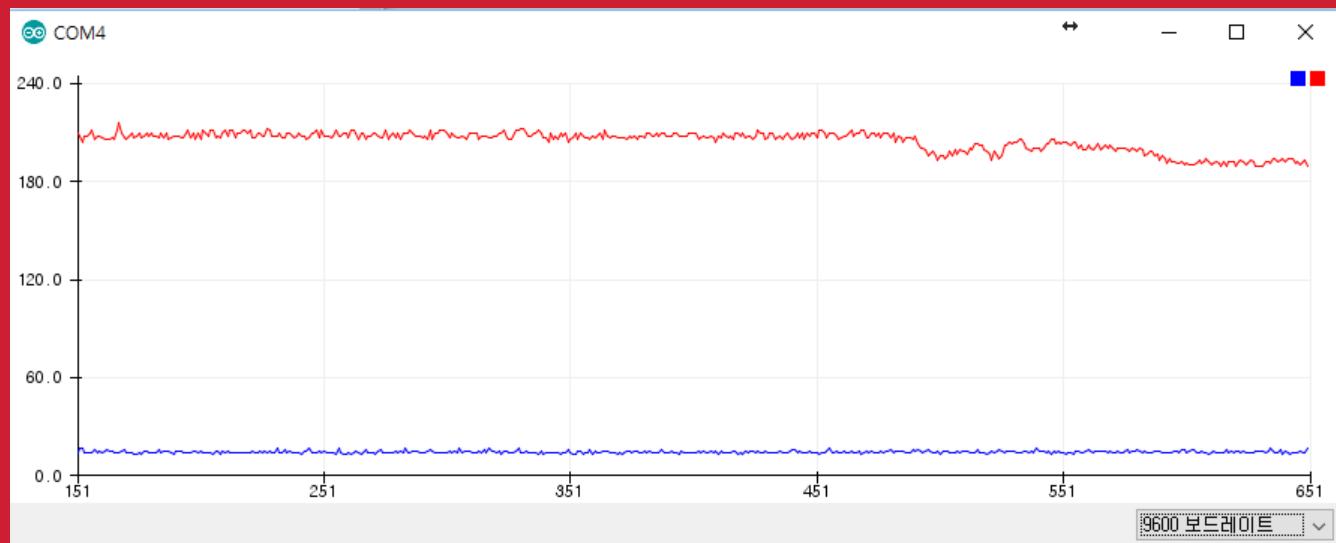
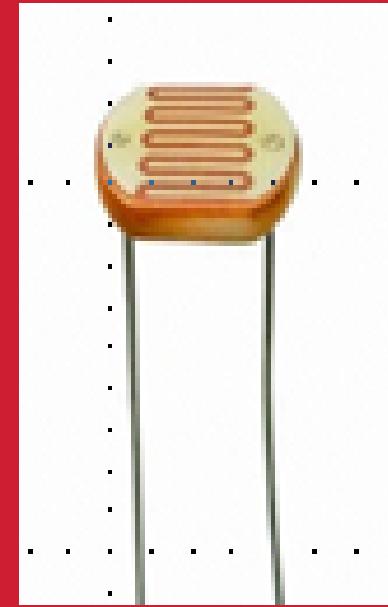
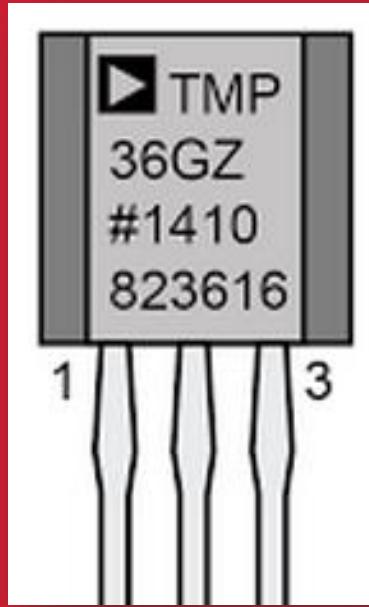


# Multiple sensors



# Arduino + Node.js

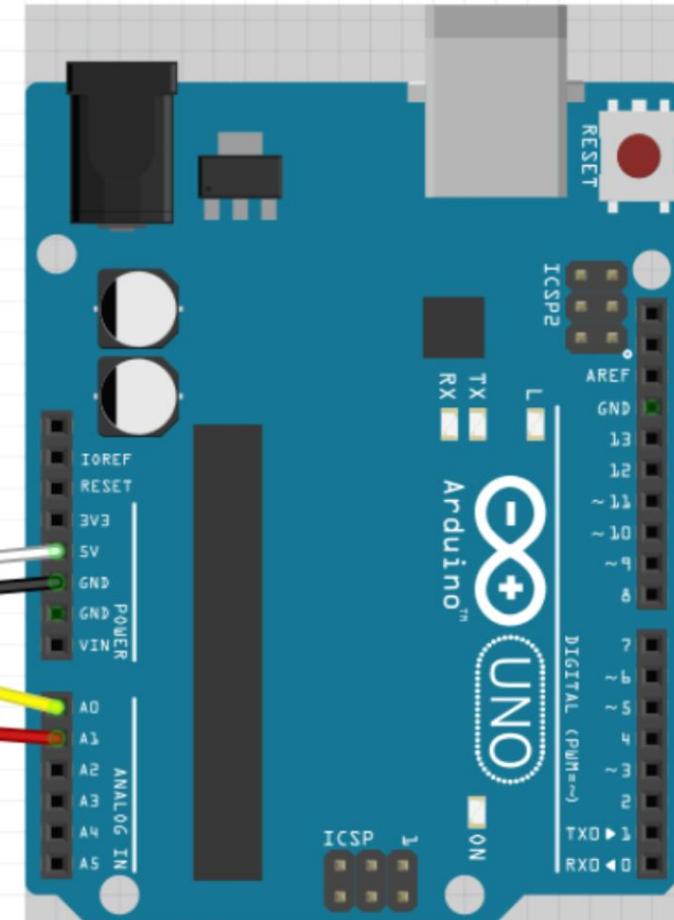
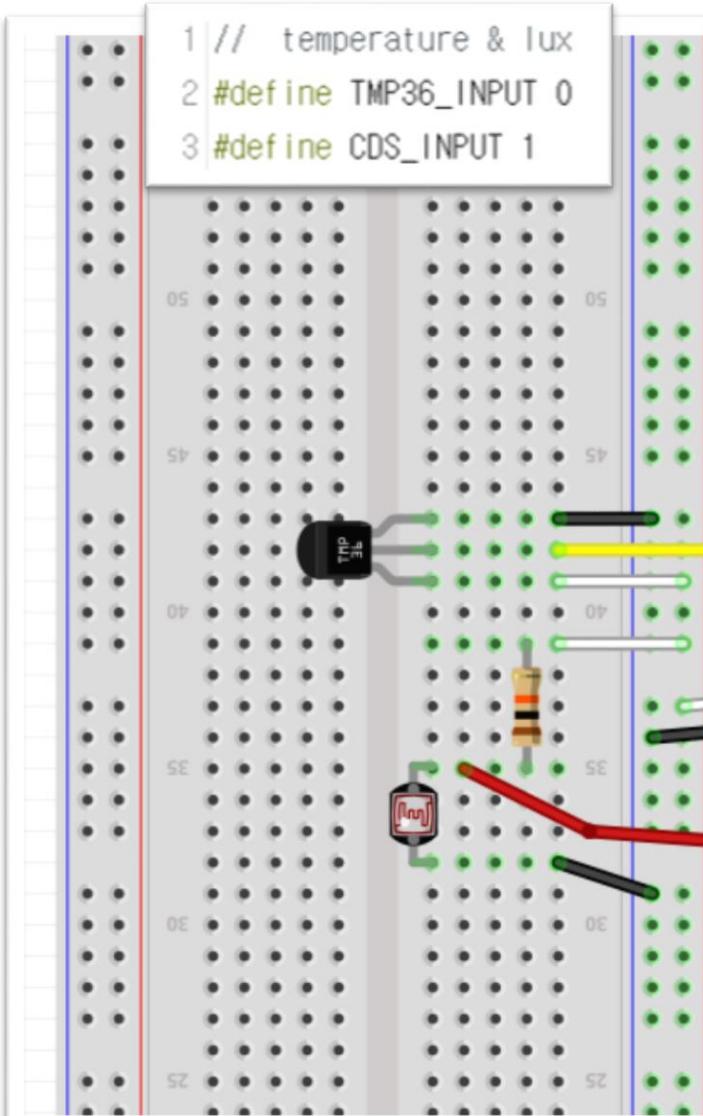






# A4.3.1 TMP36 + CdS : circuit

```
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

COM4

240.0

180.0

120.0

60.0

0.0

0

100

200

300

400

500

9600 보드레이트

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](mailto:chaos21c@gmail.com)

[ 제목 : **id**, 이름 (수정) ]

# Lecture materials



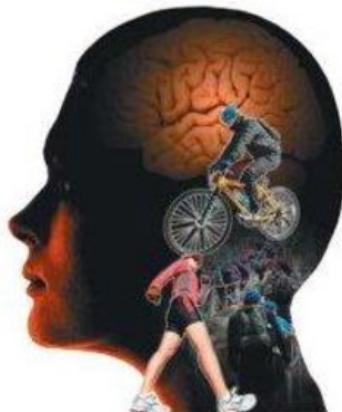
## ● References & good sites

- ✓ <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

< > | N GitHub, Inc. [US] | https://github.com Redwoods (Redwoods Yi) - GitHub

Search GitHub Sign in or Sign up

Features Business Explore Marketplace Pricing



## 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

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

Last

 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

This repository Search Pull requests Issues Marketplace Explore

Unwatch 1

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Branch: master Lec / healthcare-signal-iot / Create new file Upload

Redwoods 2018 wk03 upload Latest

..

src	2018 wk03 upload
README.md	2018 wk01 upload
wk01_hs_Intro.pdf	2018 wk01 upload-2
wk01_hs_Intro.pptx	2018 wk01 upload-2
wk02_hs_nodejs.pdf	2018 wk02 upload
wk03_hs_node_express.pdf	2018 wk03 upload

README.md

## Lec : Introduction to Healthcare Signal Visualization

All lectures by Redwoods in Inje University from 2018 and 2017.



# 1.0 What is node.js?

← → ⌂ ⌂ 안전함 | [https://www.w3schools.com/nodejs/nodejs\\_intro.asp](https://www.w3schools.com/nodejs/nodejs_intro.asp)

HOME CSS JAVASCRIPT SQL PHP BOOTSTRAP HOW TO JQUERY MORE ▾

Node.js Tutorial  
Node.js HOME  
**Node.js Intro**  
Node.js Get Started  
Node.js Modules  
Node.js HTTP Module  
Node.js File System  
Node.js URL Module  
Node.js NPM  
Node.js Events  
Node.js Upload Files  
Node.js Email

Node.js MySQL  
MySQL Get Started  
MySQL Create Database  
MySQL Create Table

## Node.js Introduction

< Previous

### What is Node.js?

- Node.js is an open source server framework
- Node.js is free
- Node.js runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- Node.js uses JavaScript on the server

### Why Node.js?

**Node.js uses asynchronous programming!**

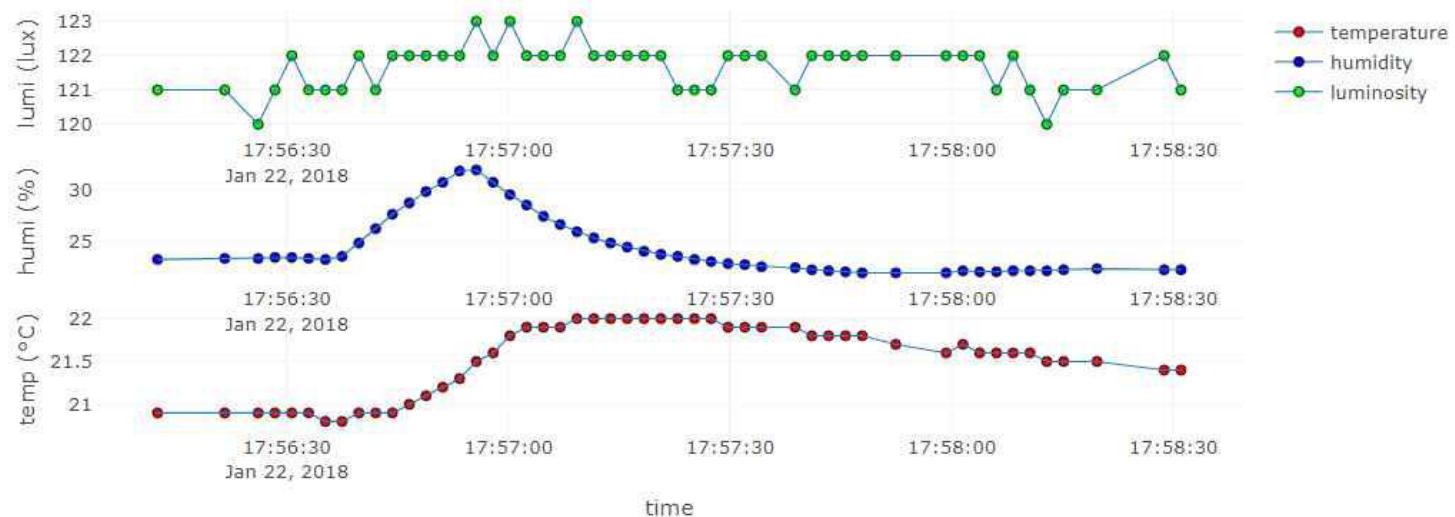
[https://www.w3schools.com/nodejs/nodejs\\_intro.asp](https://www.w3schools.com/nodejs/nodejs_intro.asp)

# Target of this class

## Real-time Weather Station from sensors

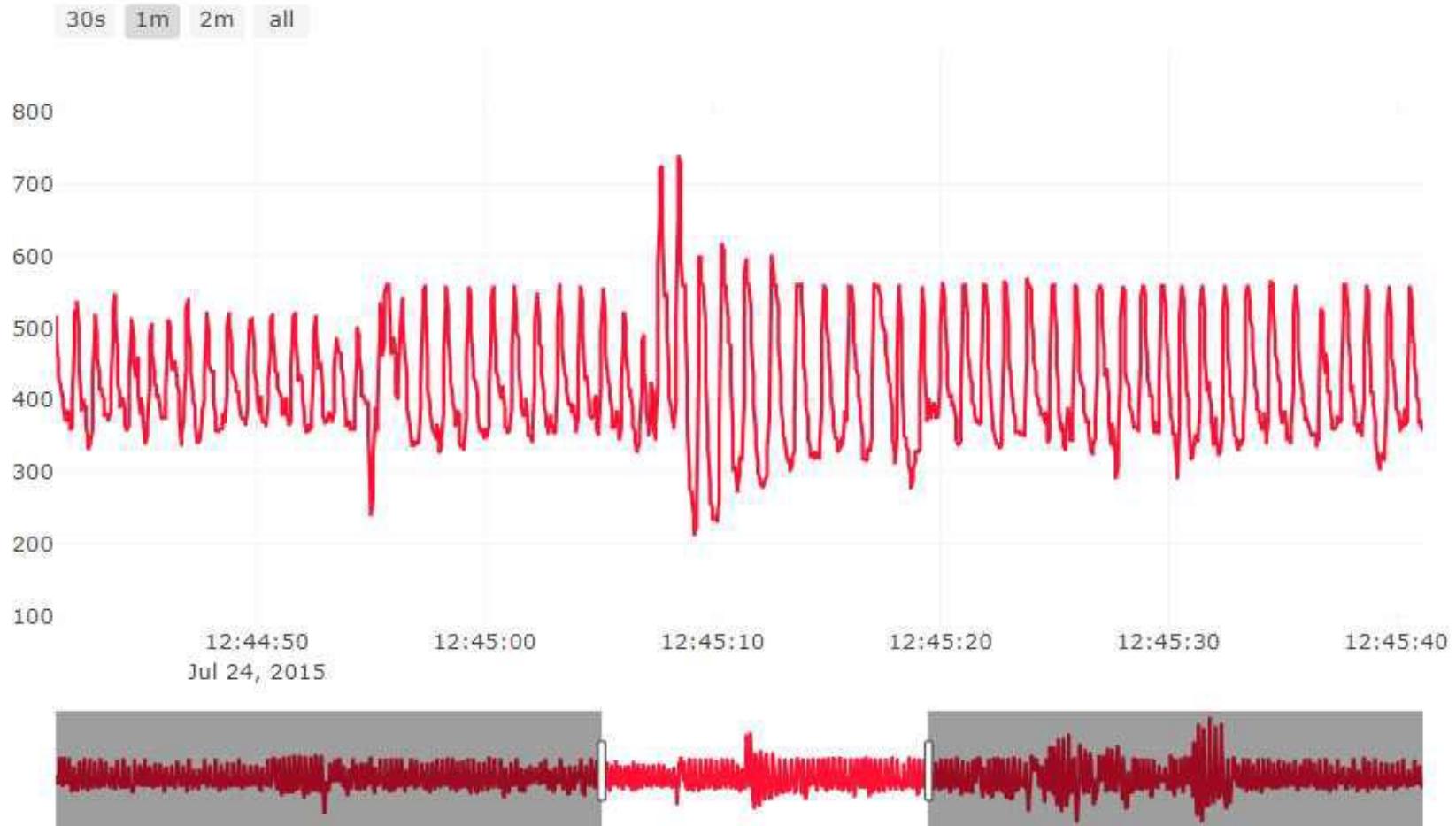


on Time: 2018-01-22 17:58:31.012



# Project of this class

PPG with rangeslider





# 주교재

아두이노와 Node.js에 기반한

IOT 신호 시각화

| 저자 이상훈 |

인제대학교 출판부

## 아두이노와 Node.js에 기반한 IOT 신호 시각화

| 저자 이상훈 |



인제대학교 출판부