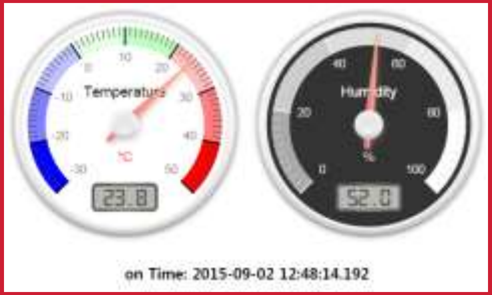
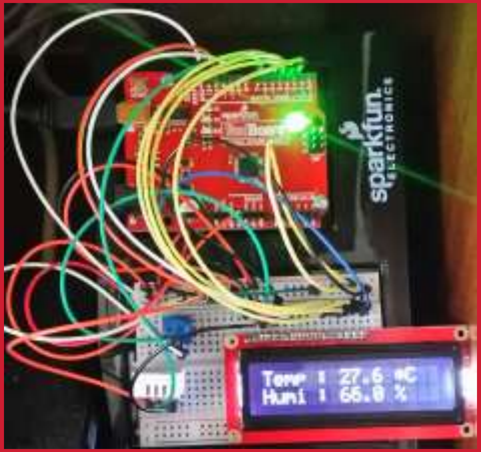
Arduino-IOT
[wk06]
Arduino + nods.js



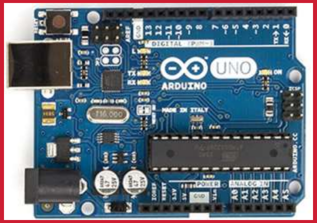


Visualization of Signals using Arduino,
Node.js & Storing Signals in MongoDB
& Mining Data using Python

Comsi, INJE University
2nd semester, 2019
Email : chaos21c@gmail.com

My ID

ID	성명
AA01	김관용
AA02	백동진
AA03	김도훈
AA04	김희찬
AA05	류재현
AA06	문민규
AA07	박진석
AA08	이승협
AA09	표혜성
AA10	김다영
AA11	성소진
AA12	김해인
AA13	신송주
AA14	윤지훈



[Review]

- [wk05]
  - Arduino sensors
  - Complete your project
  - Submit folder : AAnn_Rpt05

wk05 : Practice-04 : AAnn_Rpt05

◆ [Target of this week]

- Complete your works
- Save your outcomes and upload outputs in giyhub

제출폴더명 : **AAnn_Rpt05**

- 제출할 파일들

① **AAnn_TMP36.png**

② **AAnn_LCD_hello.png**

③ **AAnn_LCD_lux.png**

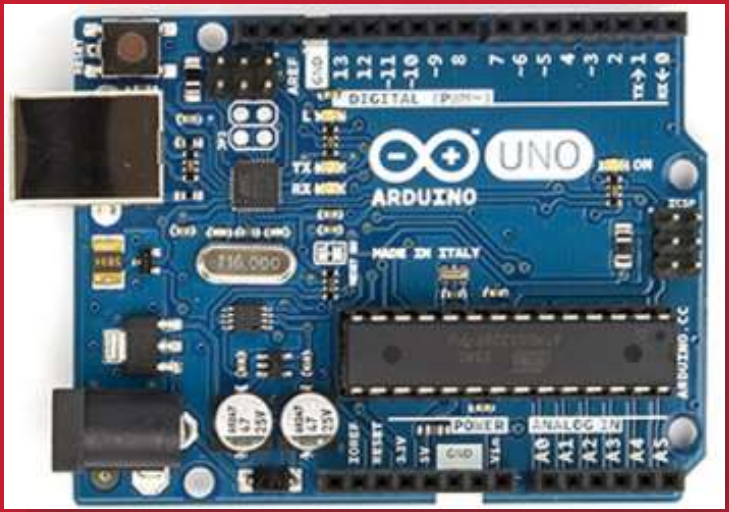
④ **AAnn_tmp36_message.png**

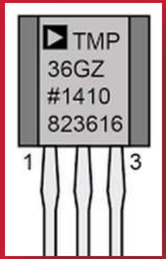
⑤ **AAnn_tmp36_IOT_data.png**

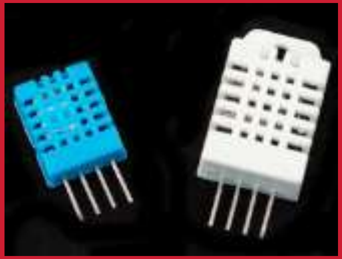
⑥ **All *.ino**

⑦ **All *.js**

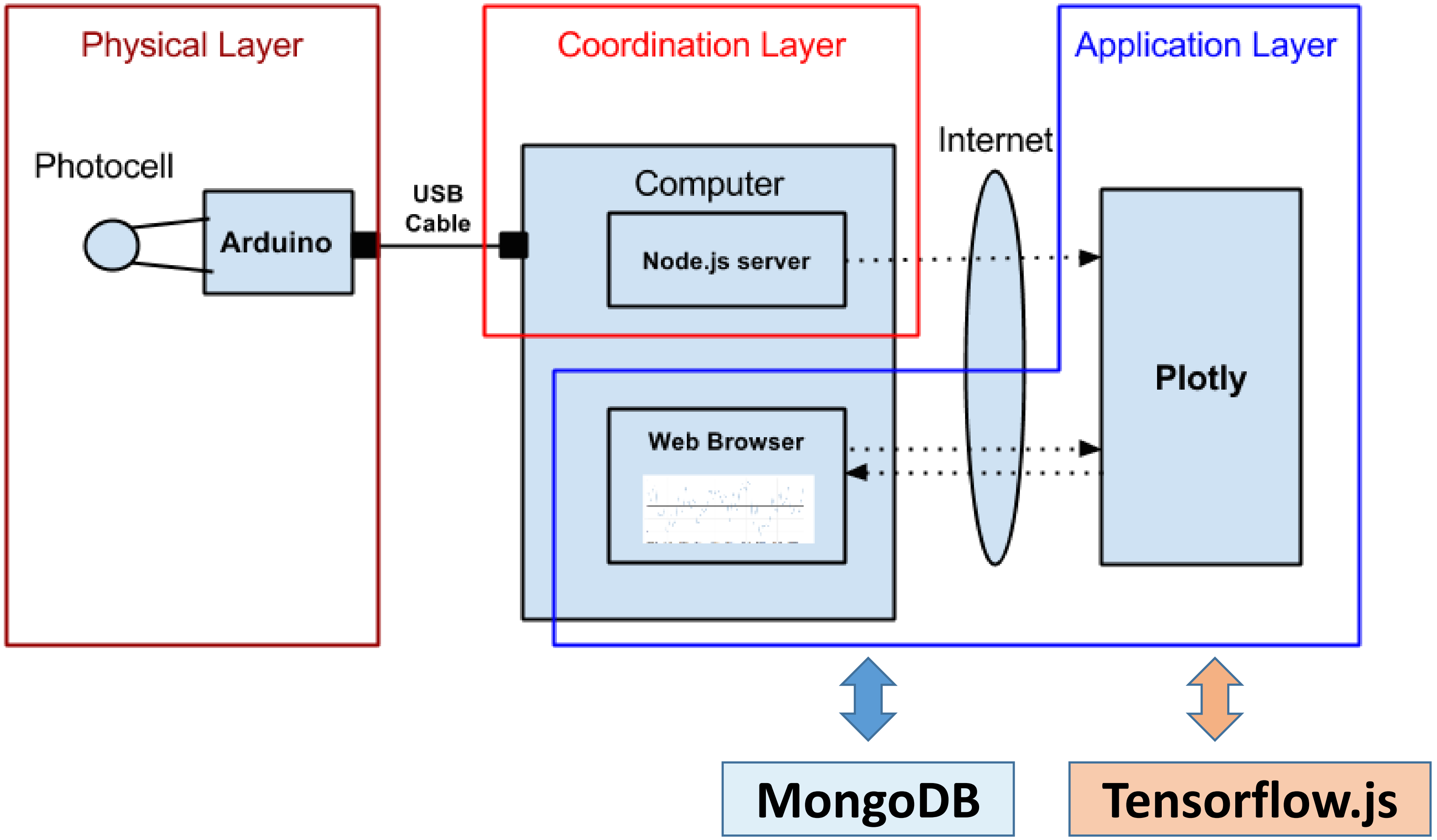
Arduino & Node.js



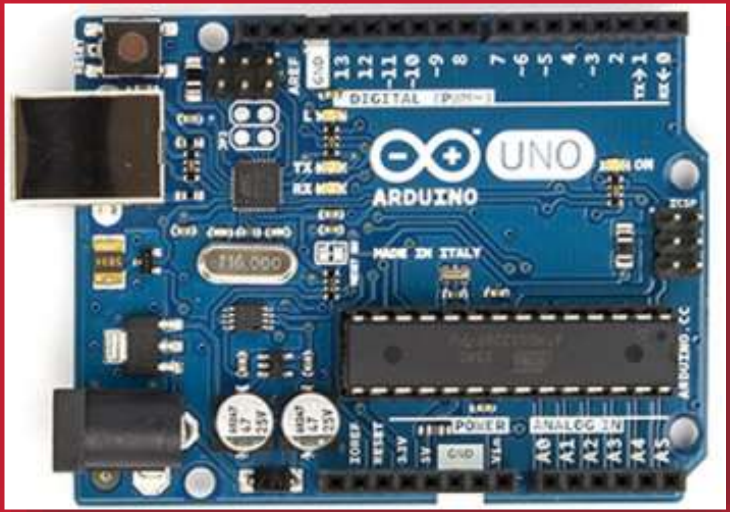


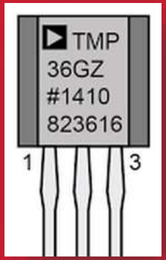


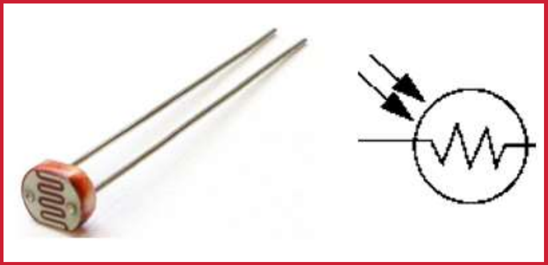
Layout [H S C]

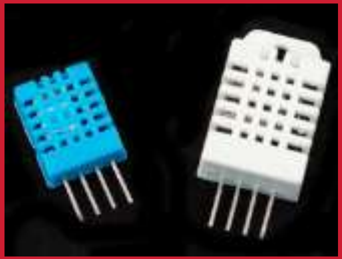

Physical Layer
Coordination Layer
Application Layer
Photocell
Arduino
USB Cable
Computer
Node.js server
Internet
Web Browser
Plotly
MongoDB
Tensorflow.js

Arduino Sensors + Node.js





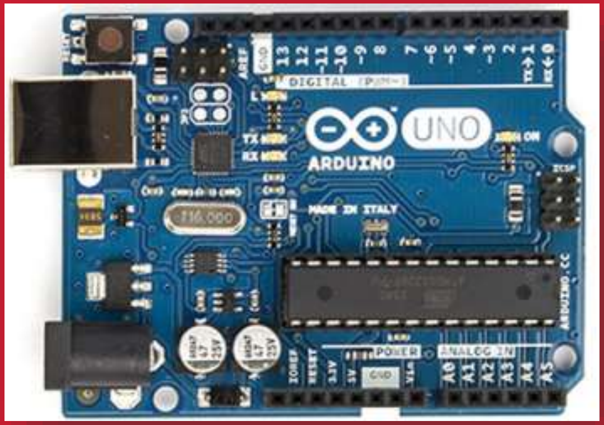


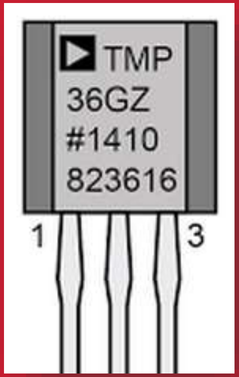


Single sensor: tmp36

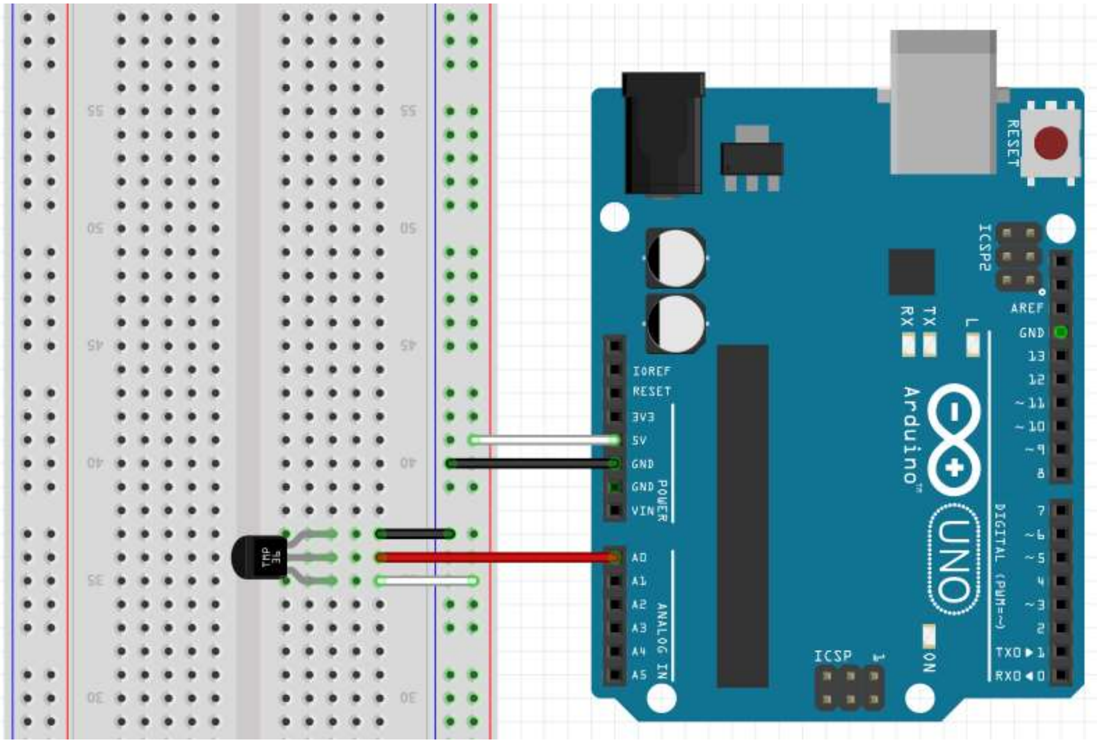
TMP36

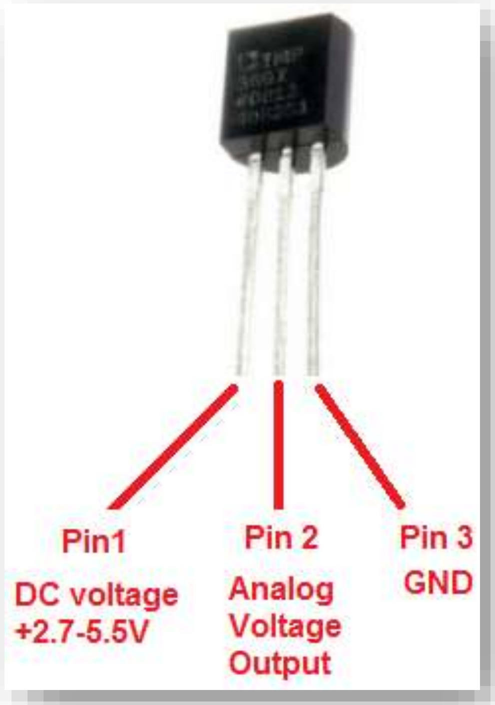
Node project



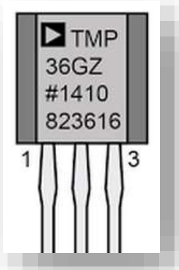


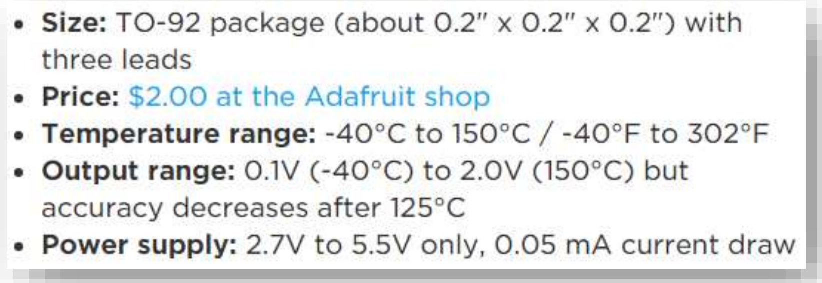
A3.1.1 Temperature sensor [TMP36]





Parts : TMP36



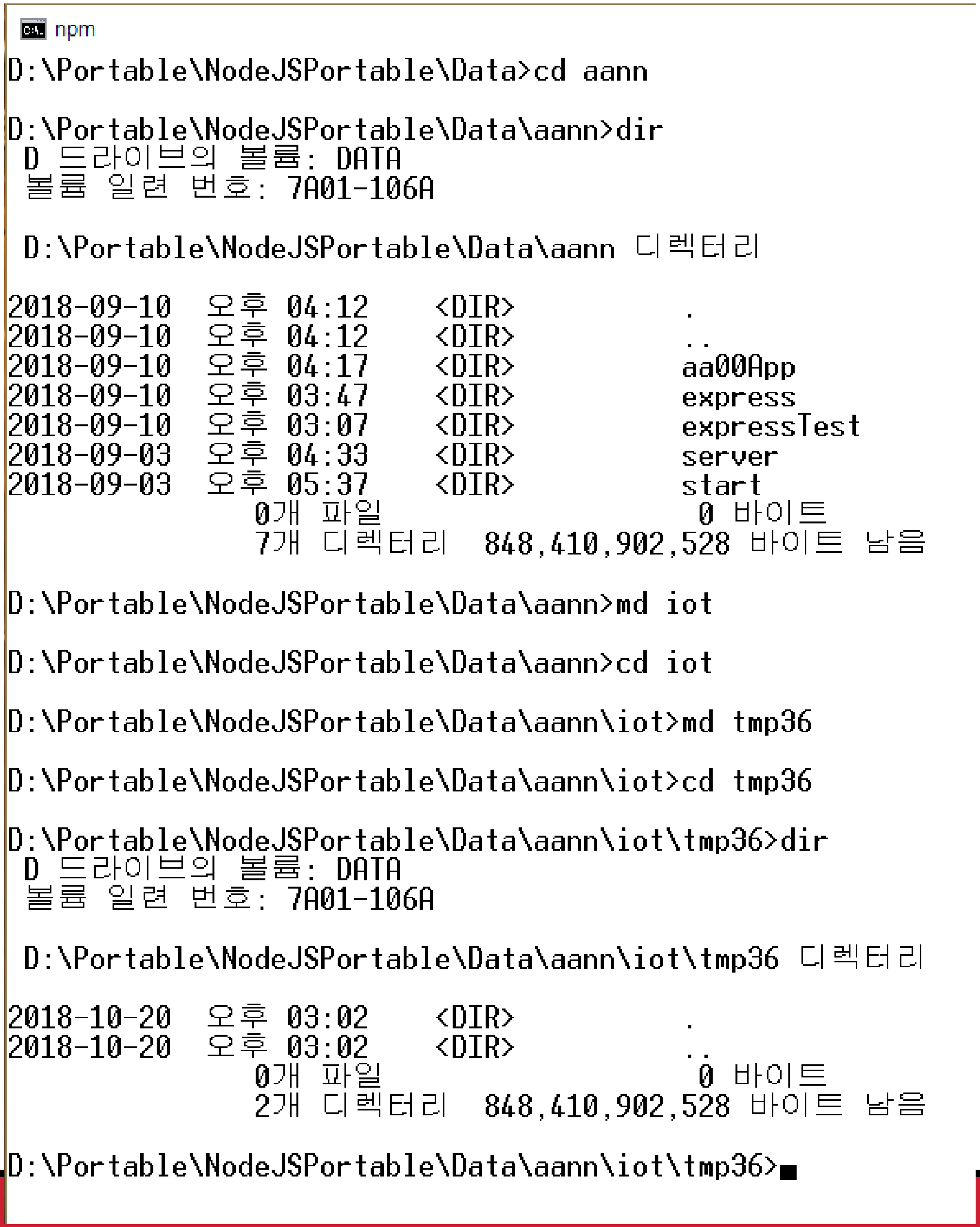


- **Size:** TO-92 package (about 0.2" x 0.2" x 0.2") with three leads
- **Price:** $2.00 at the Adafruit shop
- **Temperature range:** -40°C to 150°C / -40°F to 302°F
- **Output range:** 0.1V (-40°C) to 2.0V (150°C) but accuracy decreases after 125°C
- **Power supply:** 2.7V to 5.5V only, 0.05 mA current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

A4.1.1 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**

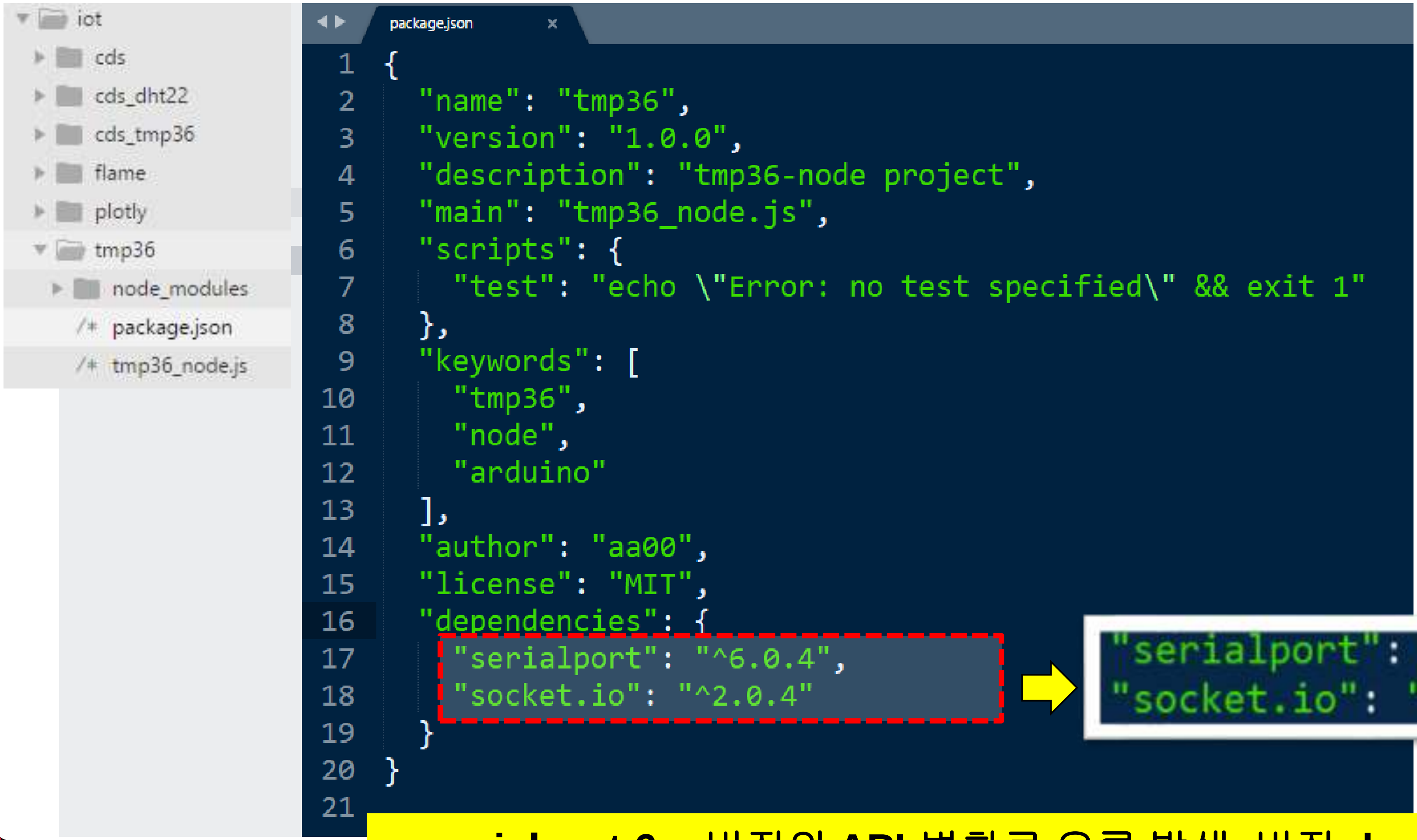


```
npm
D:\Portable\NodeJSPortable\Data>cd aann

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

 D:\Portable\NodeJSPortable\Data\aann 디렉터리

2018-09-10  오후 04:12    <DIR>          .
2018-09-10  오후 04:12    <DIR>          ..
2018-09-10  오후 04:17    <DIR>          aa00App
2018-09-10  오후 03:47    <DIR>          express
2018-09-10  오후 03:07    <DIR>          expressTest
2018-09-03  오후 04:33    <DIR>          server
2018-09-03  오후 05:37    <DIR>          start
               0개 파일                   0 바이트
               7개 디렉터리  848,410,902,528 바이트 남음

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

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

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

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

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

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

2018-10-20  오후 03:02    <DIR>          .
2018-10-20  오후 03:02    <DIR>          ..
               0개 파일                   0 바이트
               2개 디렉터리  848,410,902,528 바이트 남음

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

A4.1.6 tmp36 node project → downgrade

Go to tmp36 subfolder (after deleteing node_modules subfolder)

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

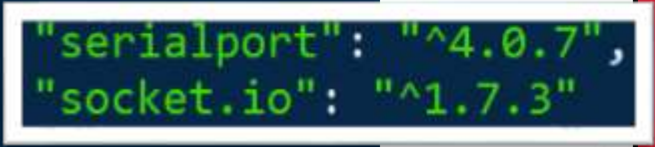


```
iot
  cds
  cds_dht22
  cds_tmp36
  flame
  plotly
  tmp36
    node_modules
    package.json
    tmp36_node.js
```

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",
  "dependencies": {
    "serialport": "^6.0.4",
    "socket.io": "^2.0.4"
  }
}
```

→

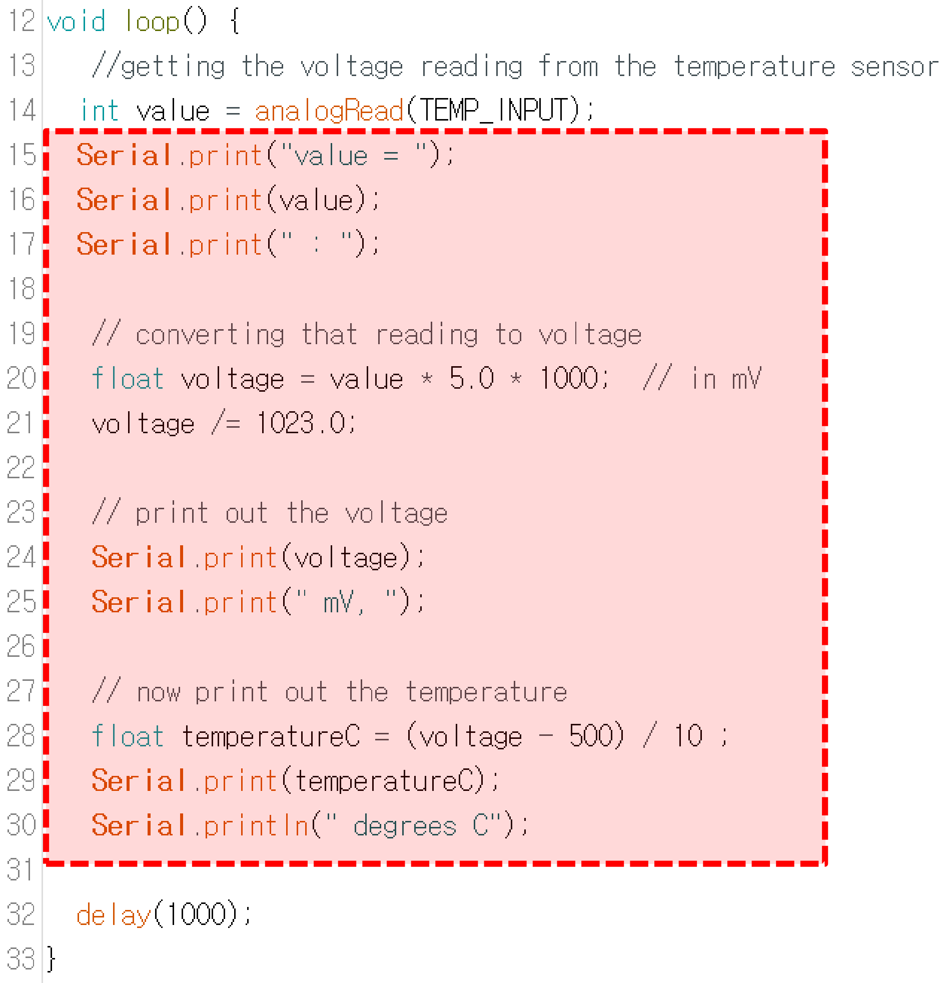


```
"serialport": "^4.0.7",
"socket.io": "^1.7.3"
```

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

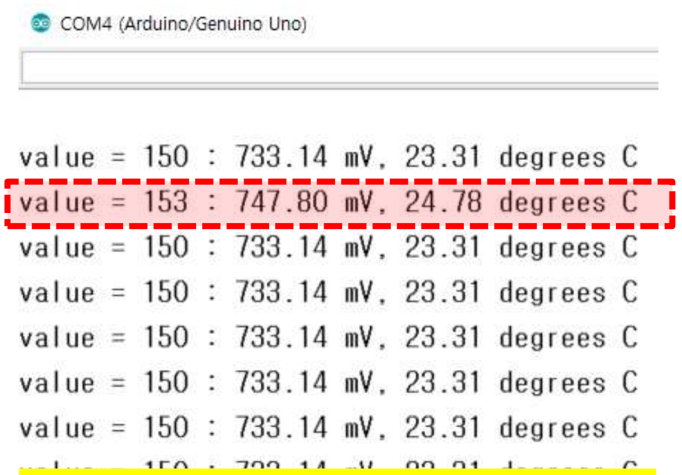
A4.1.9 tmp36 node project (all messages)

AAnn_TMP36_NodeJS.ino



```
void loop() {
  //getting the voltage reading from the temperature sensor
  int value = analogRead(TEMP_INPUT);
  Serial.print("value = ");
  Serial.print(value);
  Serial.print(" : ");

   // converting that reading to voltage
   float voltage = value * 5.0 * 1000;  // in mV
   voltage /= 1023.0;

   // print out the voltage
   Serial.print(voltage);
   Serial.print(" mV, ");

   // now print out the temperature
   float temperatureC = (voltage - 500) / 10 ;
   Serial.print(temperatureC);
   Serial.println(" degrees C");

  delay(1000);
}
```

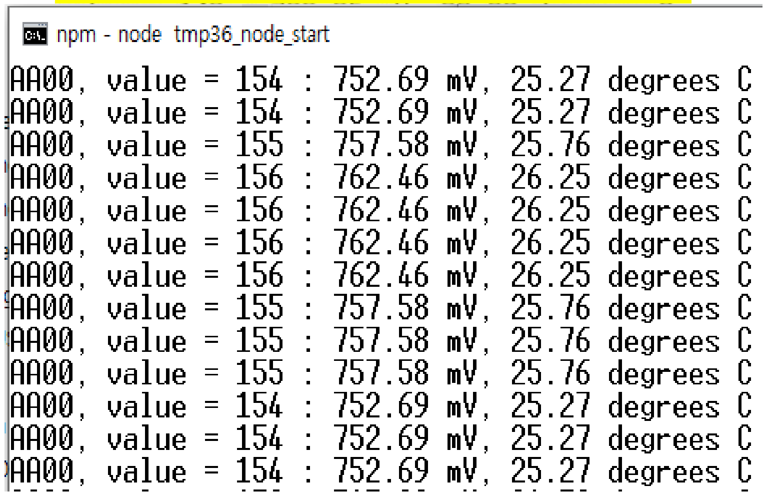
Serial monitor



COM4 (Arduino/Genuino Uno)

```
value = 150 : 733.14 mV, 23.31 degrees C
value = 153 : 747.80 mV, 24.78 degrees C
value = 150 : 733.14 mV, 23.31 degrees C
value = 150 : 733.14 mV, 23.31 degrees C
value = 150 : 733.14 mV, 23.31 degrees C
value = 150 : 733.14 mV, 23.31 degrees C
value = 150 : 733.14 mV, 23.31 degrees C
```

Node cmd

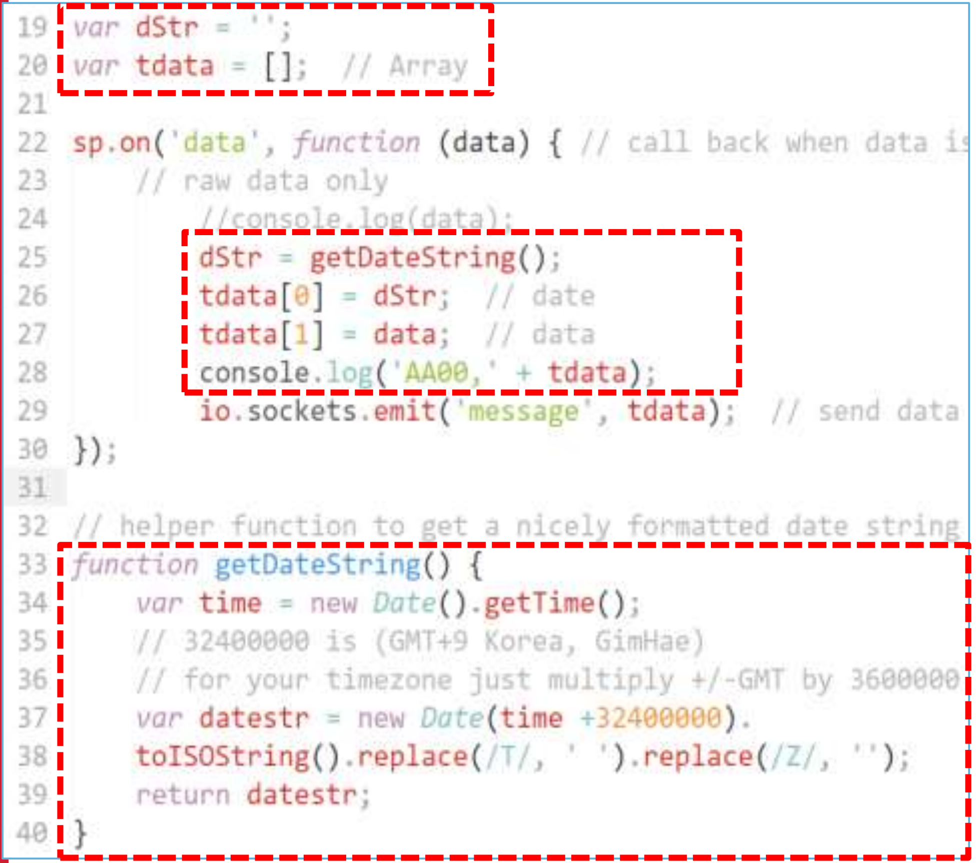


npm - node tmp36_node_start

```
AA00, value = 154 : 752.69 mV, 25.27 degrees C
AA00, value = 154 : 752.69 mV, 25.27 degrees C
AA00, value = 155 : 757.58 mV, 25.76 degrees C
AA00, value = 156 : 762.46 mV, 26.25 degrees C
AA00, value = 156 : 762.46 mV, 26.25 degrees C
AA00, value = 156 : 762.46 mV, 26.25 degrees C
AA00, value = 156 : 762.46 mV, 26.25 degrees C
AA00, value = 155 : 757.58 mV, 25.76 degrees C
AA00, value = 155 : 757.58 mV, 25.76 degrees C
AA00, value = 155 : 757.58 mV, 25.76 degrees C
AA00, value = 154 : 752.69 mV, 25.27 degrees C
AA00, value = 154 : 752.69 mV, 25.27 degrees C
AA00, value = 154 : 752.69 mV, 25.27 degrees C
```

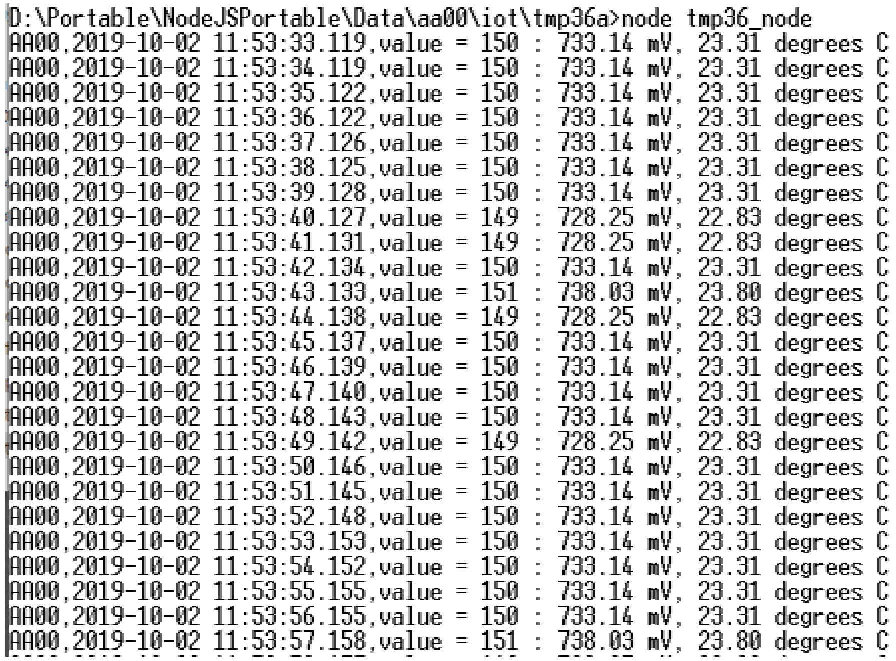
A4.1.9 tmp36 node project (all messages)

tmp36_node.js



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

sp.on('data', function (data) { // call back when data is
    // 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
});

// 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;
}
```

Node cmd에서
node tmp36_node 로
실행

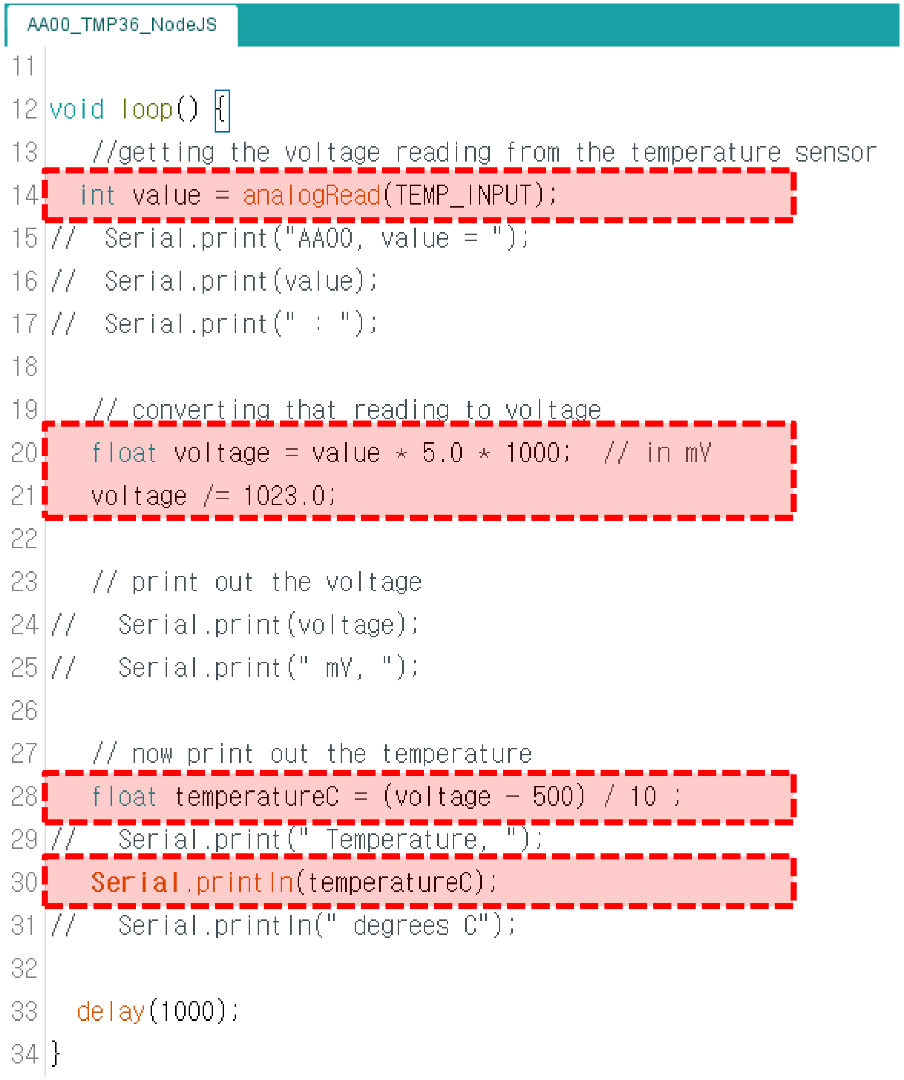


```
D:\Portable\NodeJSPortable\Data\aa00\iot\tmp36a>node tmp36_node
AA00,2019-10-02 11:53:33.119,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:34.119,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:35.122,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:36.122,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:37.126,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:38.125,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:39.128,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:40.127,value = 149 : 728.25 mV, 22.83 degrees C
AA00,2019-10-02 11:53:41.131,value = 149 : 728.25 mV, 22.83 degrees C
AA00,2019-10-02 11:53:42.134,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:43.133,value = 151 : 738.03 mV, 23.80 degrees C
AA00,2019-10-02 11:53:44.138,value = 149 : 728.25 mV, 22.83 degrees C
AA00,2019-10-02 11:53:45.137,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:46.139,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:47.140,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:48.143,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:49.142,value = 149 : 728.25 mV, 22.83 degrees C
AA00,2019-10-02 11:53:50.146,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:51.145,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:52.148,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:53.153,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:54.152,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:55.155,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:56.155,value = 150 : 733.14 mV, 23.31 degrees C
AA00,2019-10-02 11:53:57.158,value = 151 : 738.03 mV, 23.80 degrees C
```

AAnn_tmp36_message.png
로 저장

A4.1.10 tmp36 node project (only data)

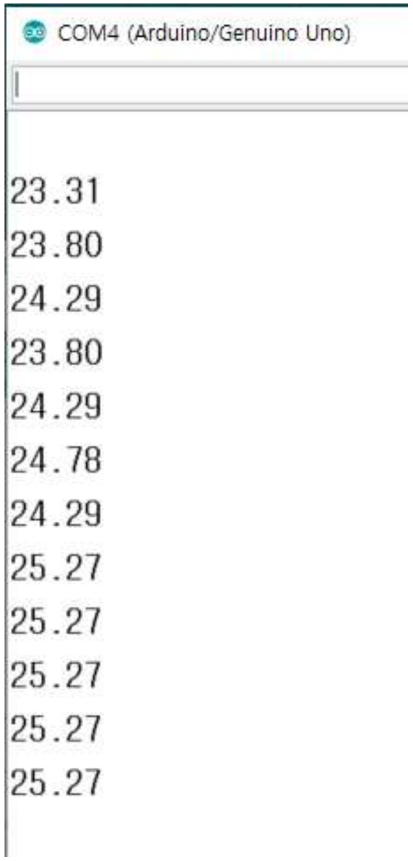
AAnn_TMP36_NodeJS.ino 수정



AA00_TMP36_NodeJS

```

void loop() {
  //getting the voltage reading from the temperature sensor
 int value = analogRead(TEMP_INPUT);
//  Serial.print("AA00, value = ");
//  Serial.print(value);
//  Serial.print(" : ");

  // converting that reading to voltage
  float voltage = value * 5.0 * 1000;  // in mV
  voltage /= 1023.0;

  // print out the voltage
//   Serial.print(voltage);
//   Serial.print(" mV, ");

  // now print out the temperature
  float temperatureC = (voltage - 500) / 10 ;
//   Serial.print(" Temperature, ");
  Serial.println(temperatureC);
//   Serial.println(" degrees C");

 delay(1000);
}
```

실행 결과

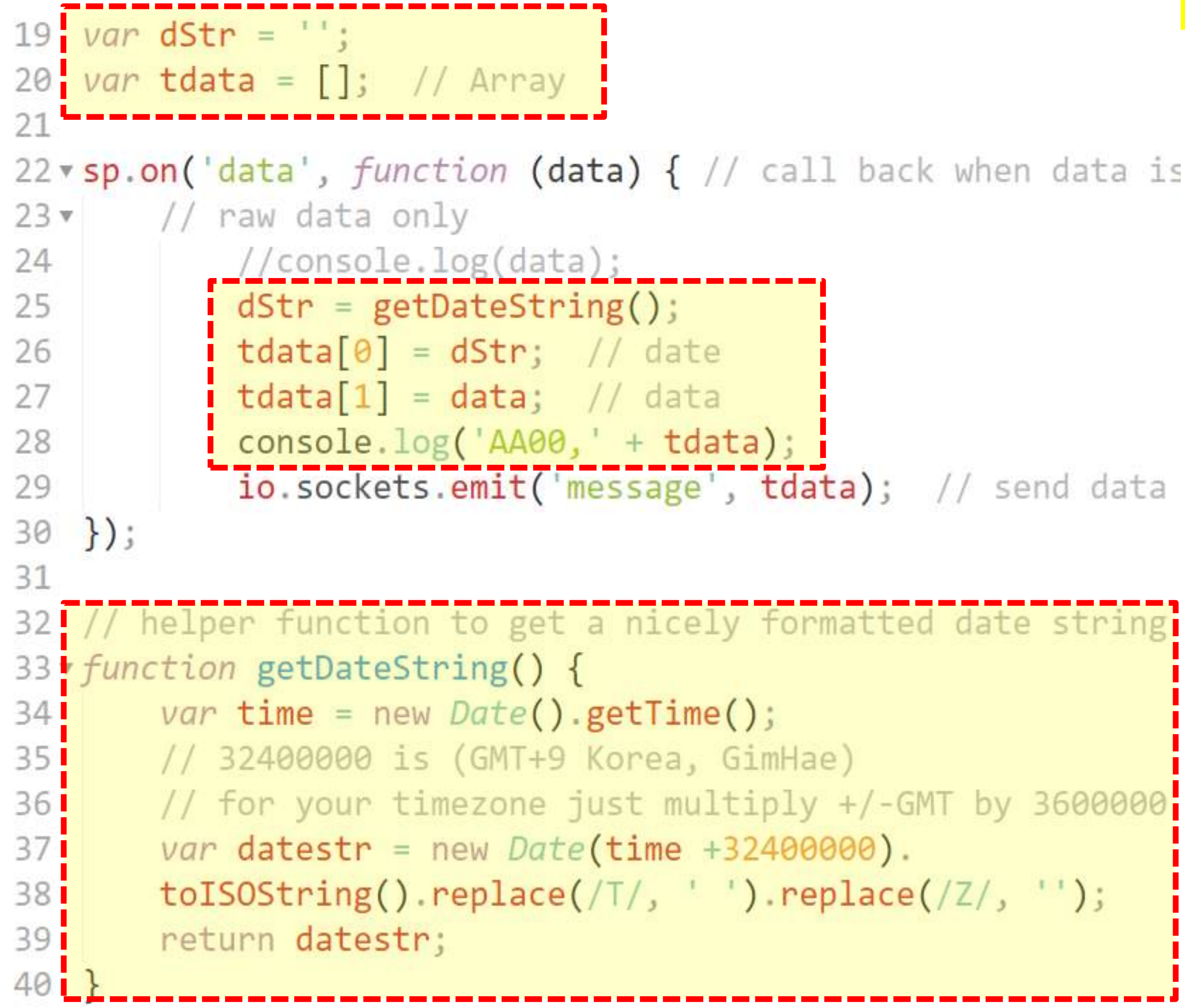


COM4 (Arduino/Genuino Uno)

```
23.31
23.80
24.29
23.80
24.29
24.78
24.29
25.27
25.27
25.27
25.27
25.27
```

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

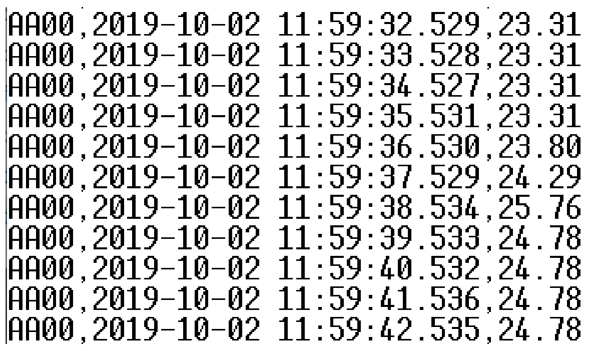
tmp36_node.js



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

sp.on('data', function (data) { // call back when data is
    // 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
});

// 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;
}
```

Node cmd에서
node tmp36_node

IOT data format
시간, data
시간, 온도



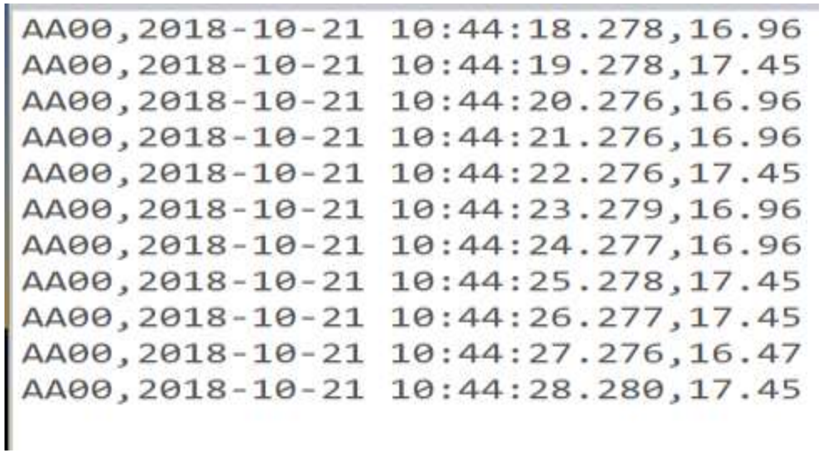
```
AA00,2019-10-02 11:59:32.529,23.31
AA00,2019-10-02 11:59:33.528,23.31
AA00,2019-10-02 11:59:34.527,23.31
AA00,2019-10-02 11:59:35.531,23.31
AA00,2019-10-02 11:59:36.530,23.80
AA00,2019-10-02 11:59:37.529,24.29
AA00,2019-10-02 11:59:38.534,25.76
AA00,2019-10-02 11:59:39.533,24.78
AA00,2019-10-02 11:59:40.532,24.78
AA00,2019-10-02 11:59:41.536,24.78
AA00,2019-10-02 11:59:42.535,24.78
```

시간 , 온도

공백없이 ","로
시간과 온도 구분

A4.1.12 tmp36 node project (실행 결과)

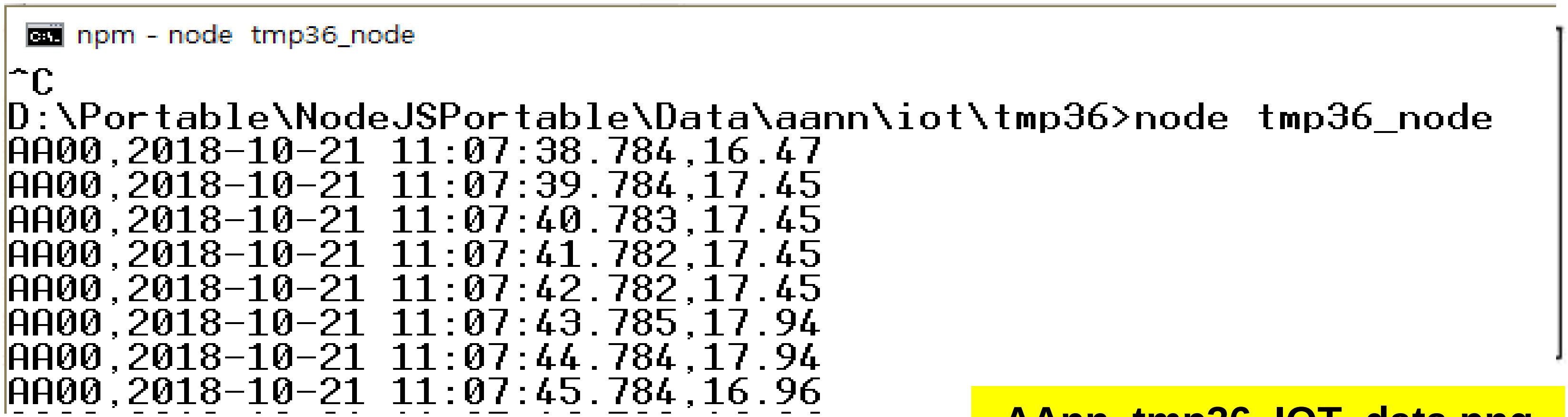
- Sublime Text 3에서 실행



```
AA00,2018-10-21 10:44:18.278,16.96
AA00,2018-10-21 10:44:19.278,17.45
AA00,2018-10-21 10:44:20.276,16.96
AA00,2018-10-21 10:44:21.276,16.96
AA00,2018-10-21 10:44:22.276,17.45
AA00,2018-10-21 10:44:23.279,16.96
AA00,2018-10-21 10:44:24.277,16.96
AA00,2018-10-21 10:44:25.278,17.45
AA00,2018-10-21 10:44:26.277,17.45
AA00,2018-10-21 10:44:27.276,16.47
AA00,2018-10-21 10:44:28.280,17.45
```

- Node cmd에서 실행

```
node tmp36_node
```

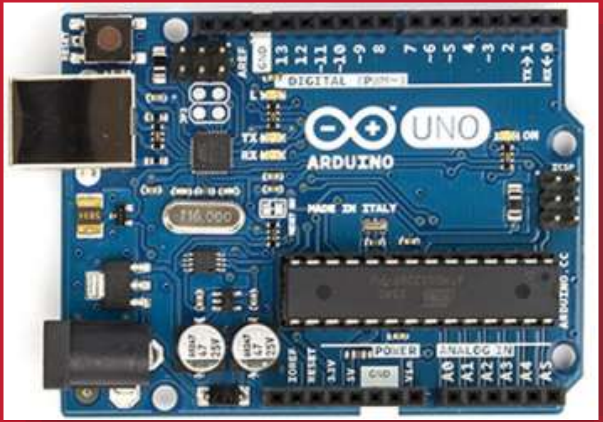


npm - node tmp36_node

```
^C
D:\Portable\NodeJSPortable\Data\aann\iot\tmp36>node tmp36_node
AA00,2018-10-21 11:07:38.784,16.47
AA00,2018-10-21 11:07:39.784,17.45
AA00,2018-10-21 11:07:40.783,17.45
AA00,2018-10-21 11:07:41.782,17.45
AA00,2018-10-21 11:07:42.782,17.45
AA00,2018-10-21 11:07:43.785,17.94
AA00,2018-10-21 11:07:44.784,17.94
AA00,2018-10-21 11:07:45.784,16.96
```

AAnn_tmp36_IOT_data.png
로 저장

Single sensor: CdS

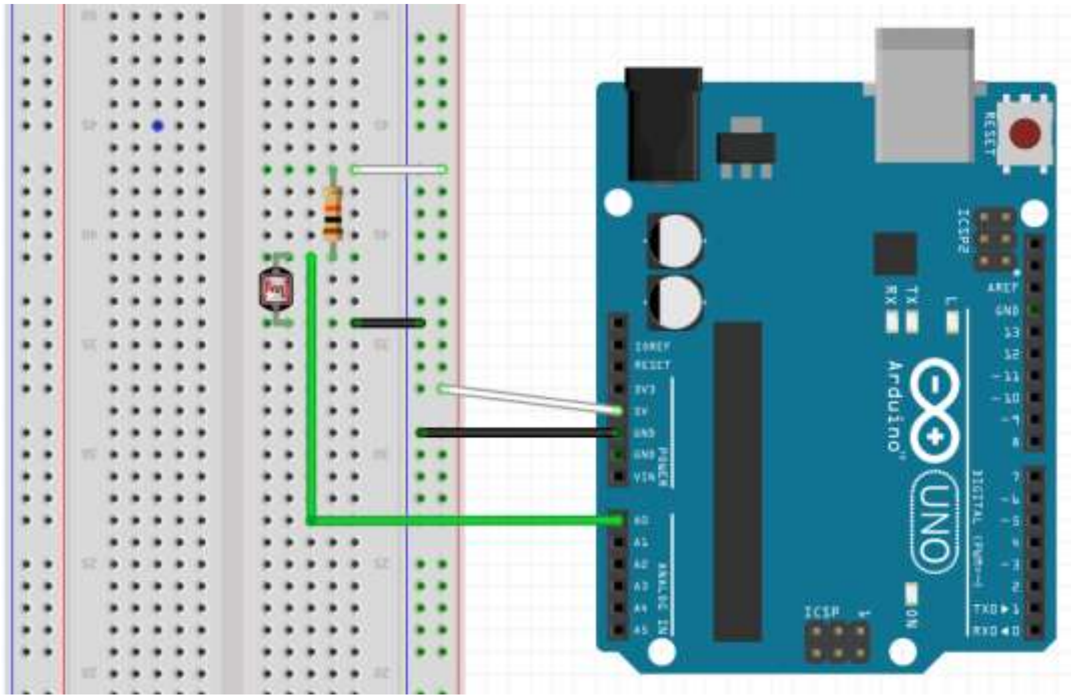


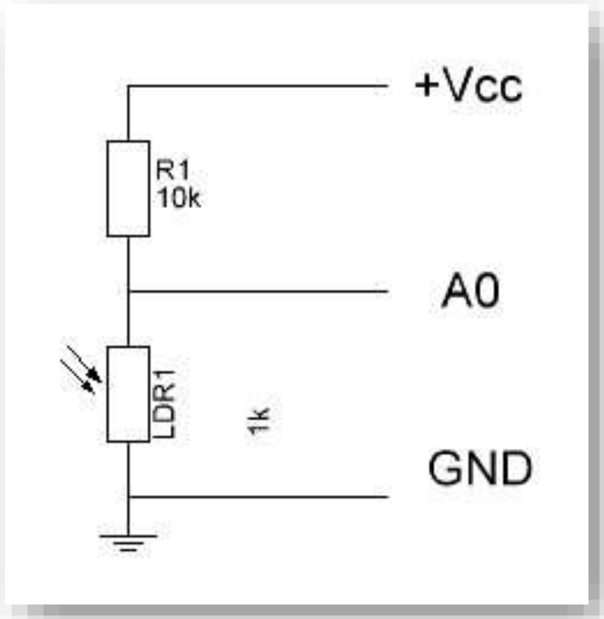
CdS (LDR)

Node project

A3.2.2 Luminosity sensor [Photocell LDR]

CdS 센서 회로



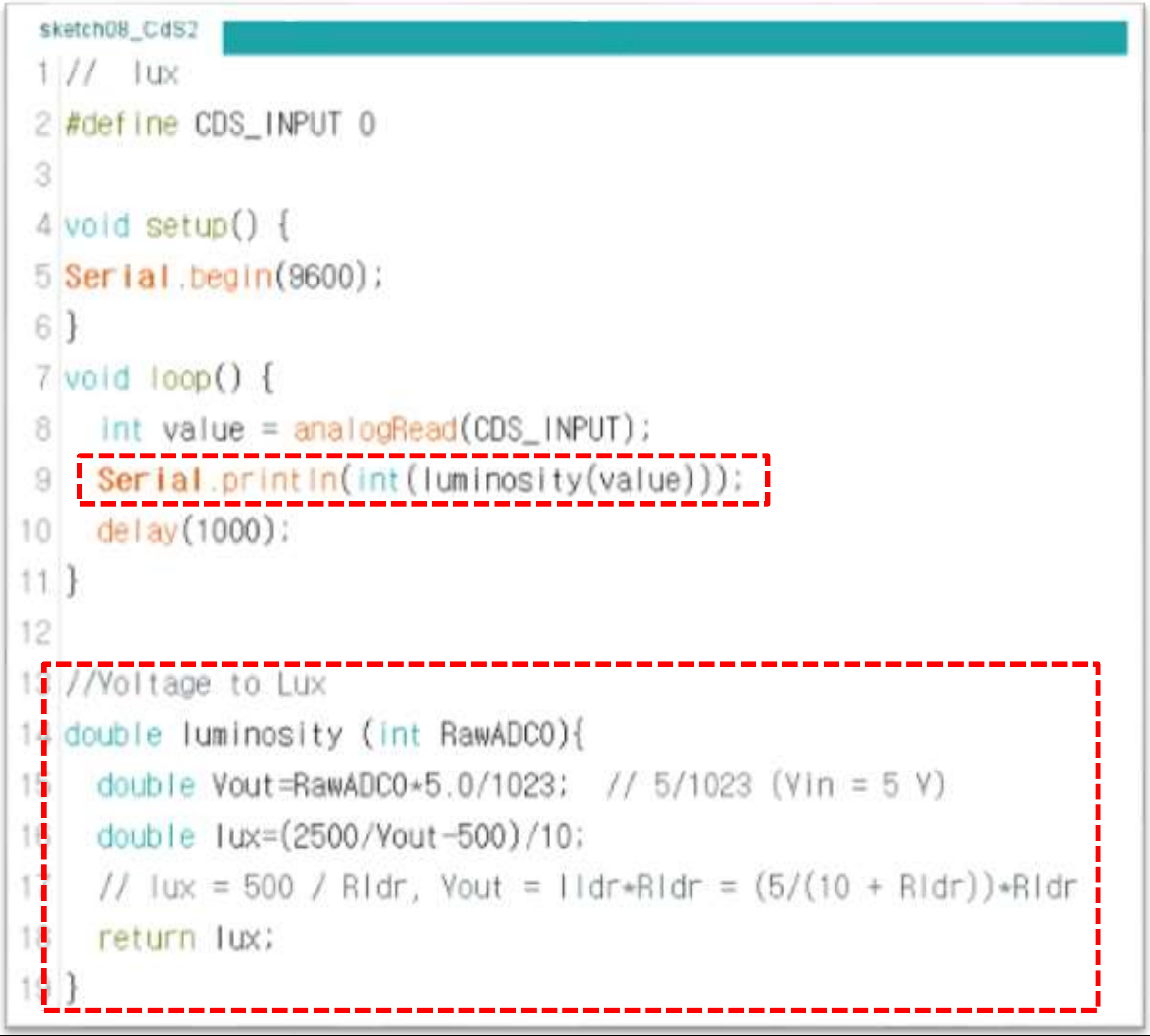


Parts : 20 mm photocell LDR, R (10 kΩ X 1)

광센서에서의 전압 강하 값을 **A0**로 측정

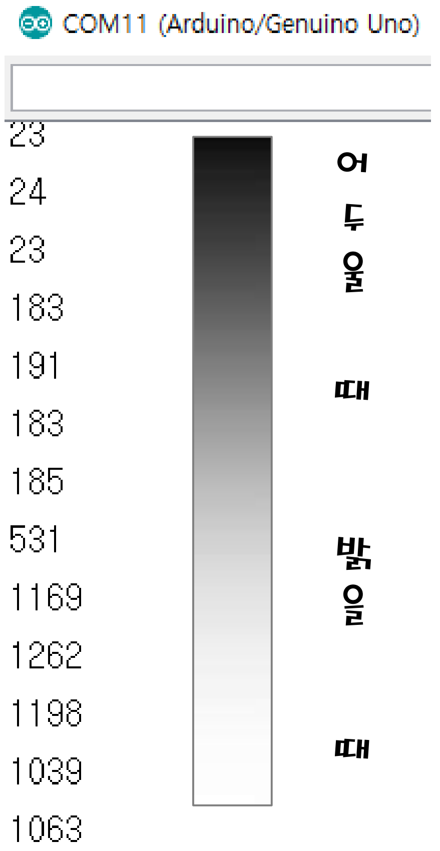
A3.2.6 Luminosity sensor [Photocell LDR]

CdS 센서 회로 - 측정 2.



sketch08_CdS2

```
// lux
#define CDS_INPUT 0

void setup() {
Serial.begin(9600);
}
void loop() {
  int value = analogRead(CDS_INPUT);
  Serial.println(int(luminosity(value)));
  delay(1000);
}

//Voltage to Lux
double luminosity (int RawADC0){
  double Vout=RawADC0*5.0/1023;  // 5/1023 (Vin = 5 V)
  double lux=(2500/Vout-500)/10;
  // lux = 500 / Rldr, Vout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
  return lux;
}
```



COM11 (Arduino/Genuino Uno)

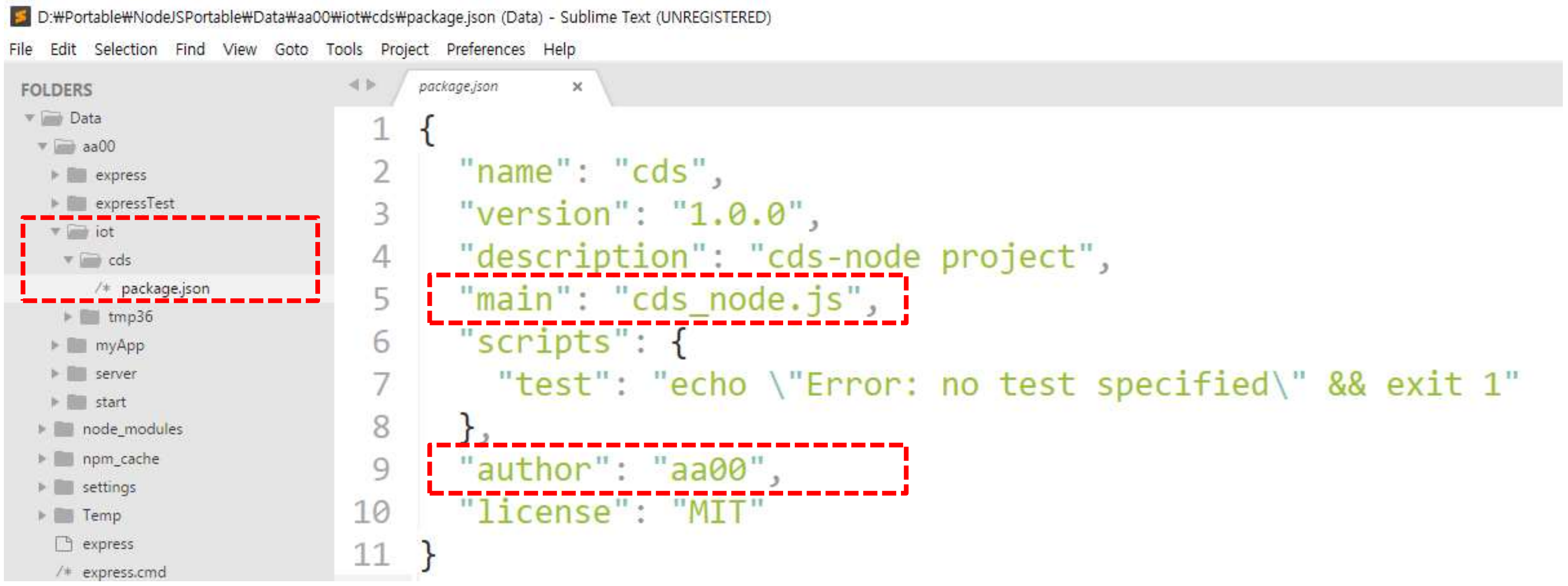
```
23
24
23
183
191
183
185
531
1169
1262
1198
1039
1063
```

밝을수록 측정 값이 커지고
어두울수록 값이 작아진다 !!!

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**

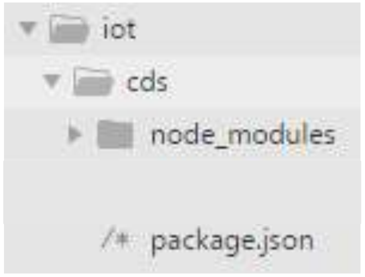
"main": "cds_node.js"
"author": “aann"


D:₩Portable₩NodeJSPortable₩Data₩aa00₩iot₩cds₩package.json (Data) - Sublime Text (UNREGISTERED)

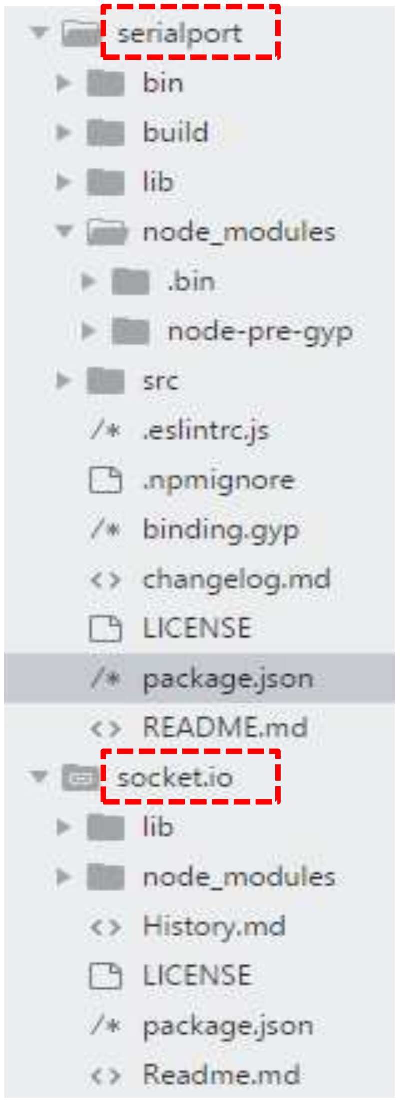
```
{
  "name": "cds",
  "version": "1.0.0",
  "description": "cds-node project",
  "main": "cds_node.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "aa00",
  "license": "MIT"
}
```

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**



```
▼ iot
  ▼ cds
    ▶ node_modules
      package.json
```



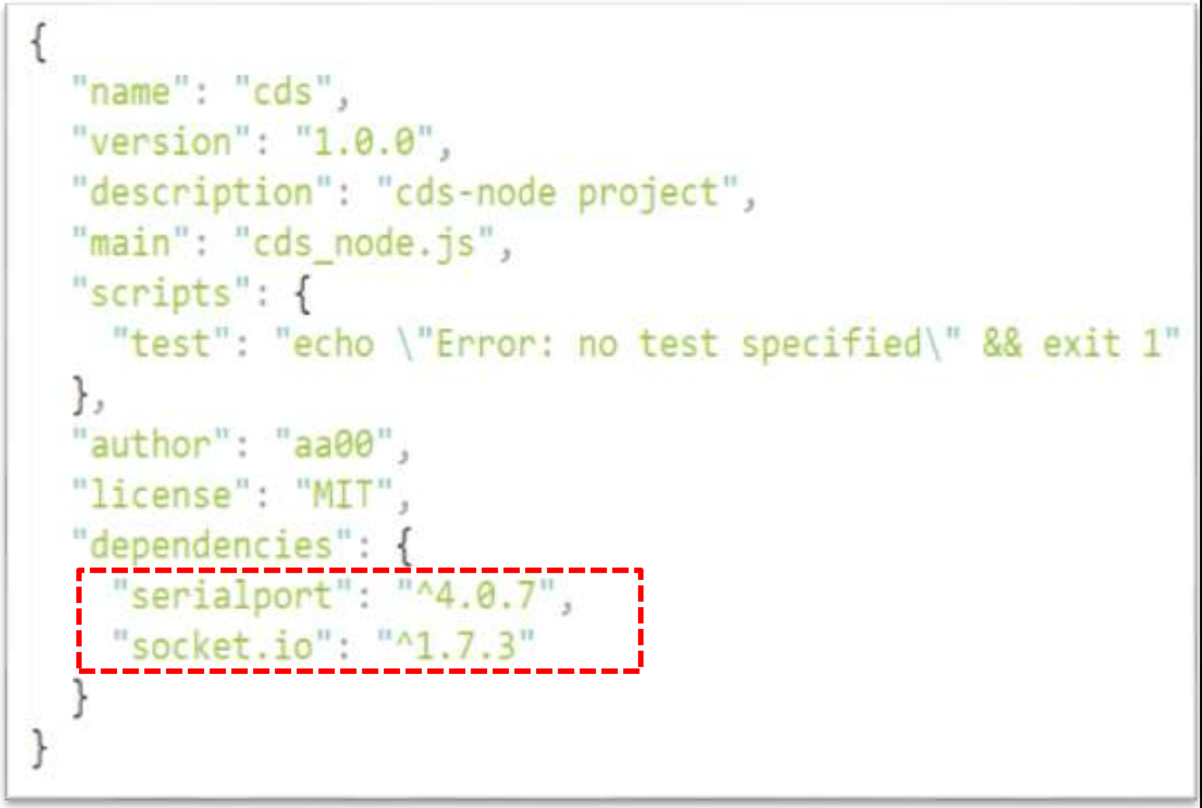
```
▼ serialport
  ▶ bin
  ▶ build
  ▶ lib
  ▼ node_modules
    ▶ .bin
    ▶ node-pre-gyp
  ▶ src
    .eslintrc.js
    .npmignore
    binding.gyp
    changelog.md
    LICENSE
    package.json
    README.md
▼ socket.io
  ▶ lib
  ▶ node_modules
    History.md
    LICENSE
    package.json
    Readme.md
```

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

A4.2.3 Luminosity sensor [Photocell LDR]

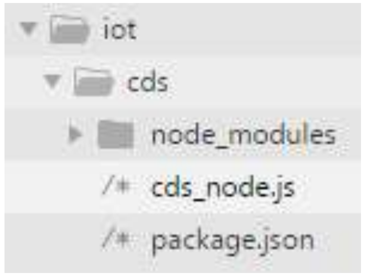
1. **Make cds node project**
   - ➢ **md cds**
   - ➢ **cd cds**
2. **Go to cds subfolder**
   - ➢ **npm init**
   - ➢ **npm install –save serialport@4.0.7**
   - ➢ **npm install –save socket.io@1.7.3**

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]

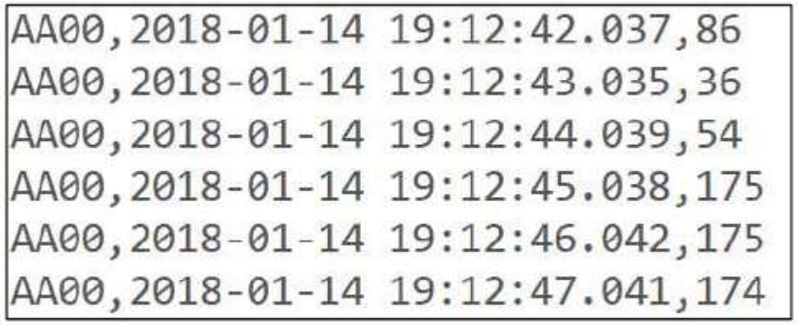


Save tmp36_node.js as cds_node.js

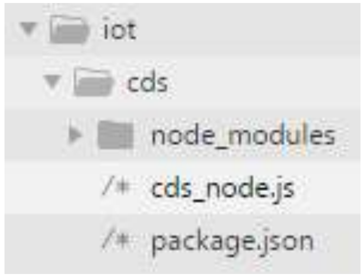
```
var dStr = '';
var tdata = [];

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



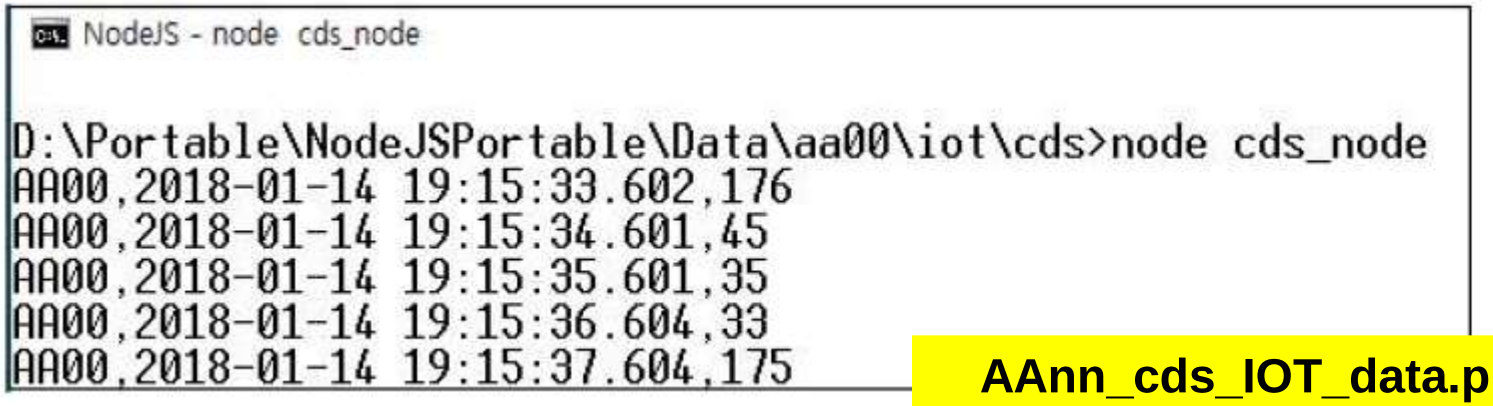
```
AA00,2018-01-14 19:12:42.037,86
AA00,2018-01-14 19:12:43.035,36
AA00,2018-01-14 19:12:44.039,54
AA00,2018-01-14 19:12:45.038,175
AA00,2018-01-14 19:12:46.042,175
AA00,2018-01-14 19:12:47.041,174
```



```
▼ iot
  ▼ cds
    ▶ node_modules
      cds_node.js
      package.json
```

- Node cmd에서 실행

```
node cds_node
```



NodeJS - node cds_node

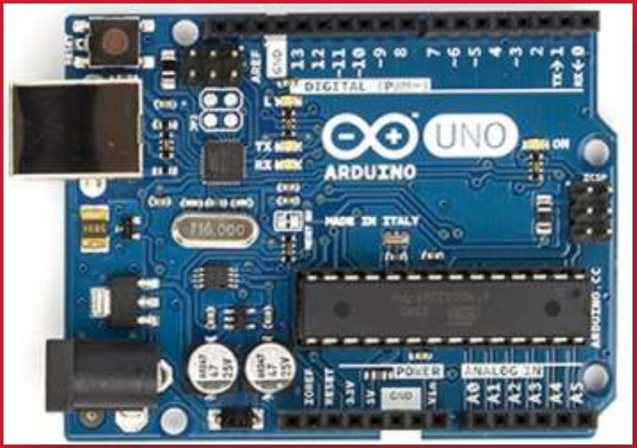
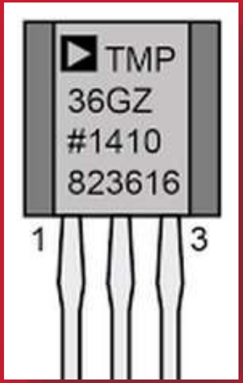
```
D:\Portable\NodeJSPortable\Data\aa00\iot\cds>node cds_node
AA00,2018-01-14 19:15:33.602,176
AA00,2018-01-14 19:15:34.601,45
AA00,2018-01-14 19:15:35.601,35
AA00,2018-01-14 19:15:36.604,33
AA00,2018-01-14 19:15:37.604,175
```

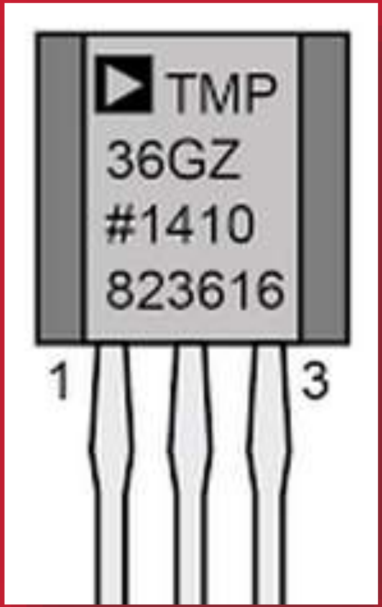
**AAnn_cds_IOT_data.png
로 저장**

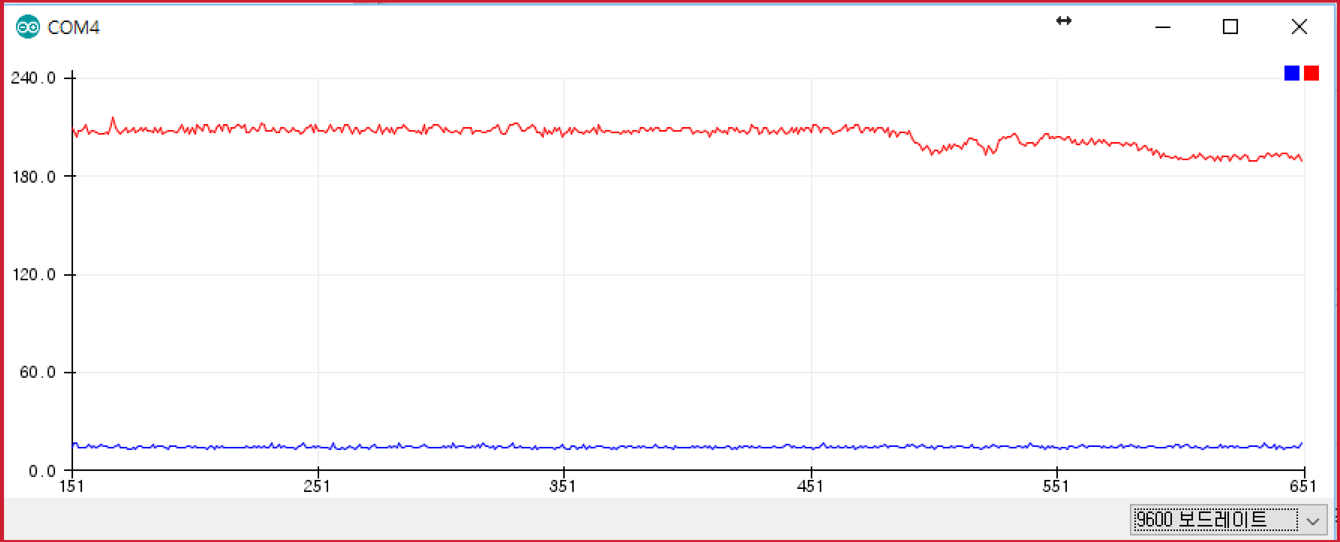
Multiple sensors

Arduino + Node.js

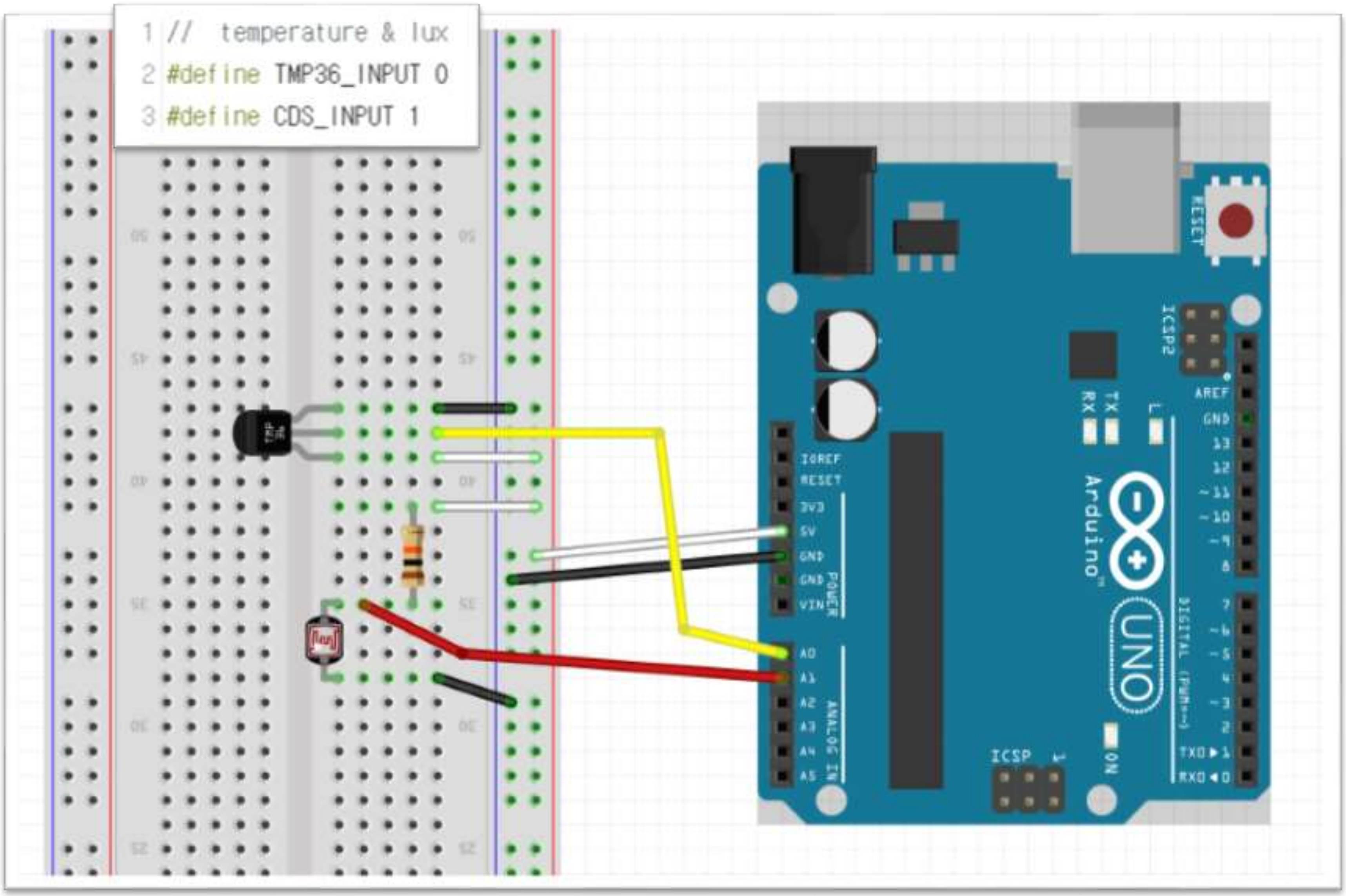
Monitoring via Serial monitor & LCD


TMP
36GZ
#1410
823616
1
3

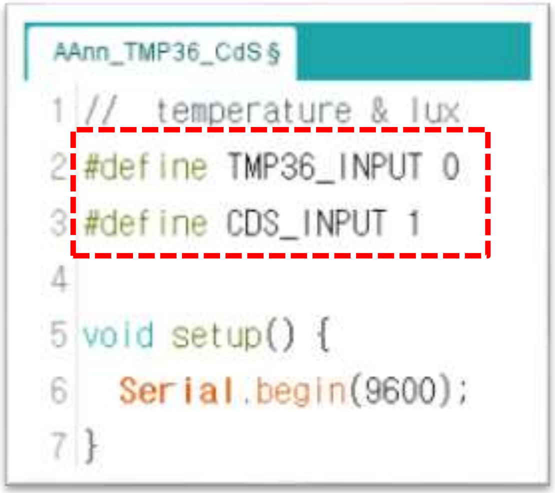

COM4
240.0
180.0
120.0
60.0
0.0
151
251
351
451
551
651
9600 보드레이트

A4.3.1 TMP36 + CdS : circuit



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

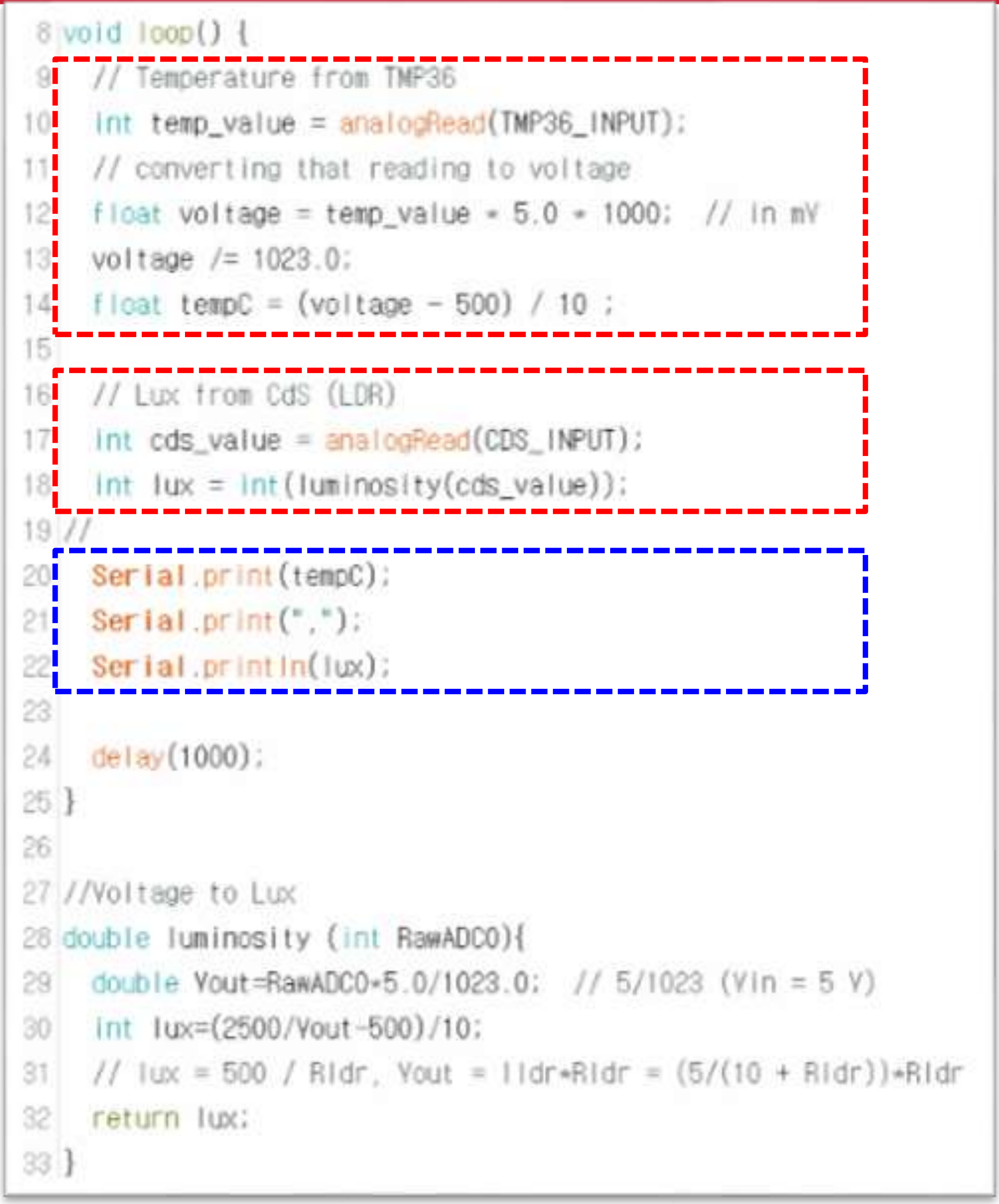
A4.3.2 TMP36 + CdS : code



AAnn_TMP36_CdS §

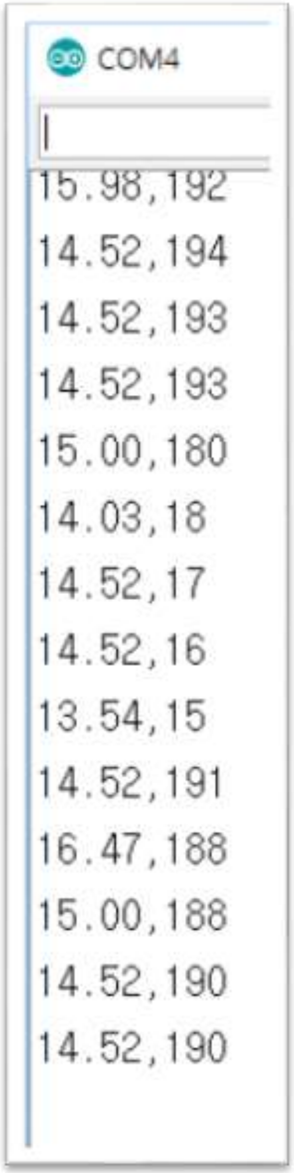
```
//  temperature & lux
#define TMP36_INPUT 0
#define CDS_INPUT 1

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

AAnn_tmp36_cds.ino

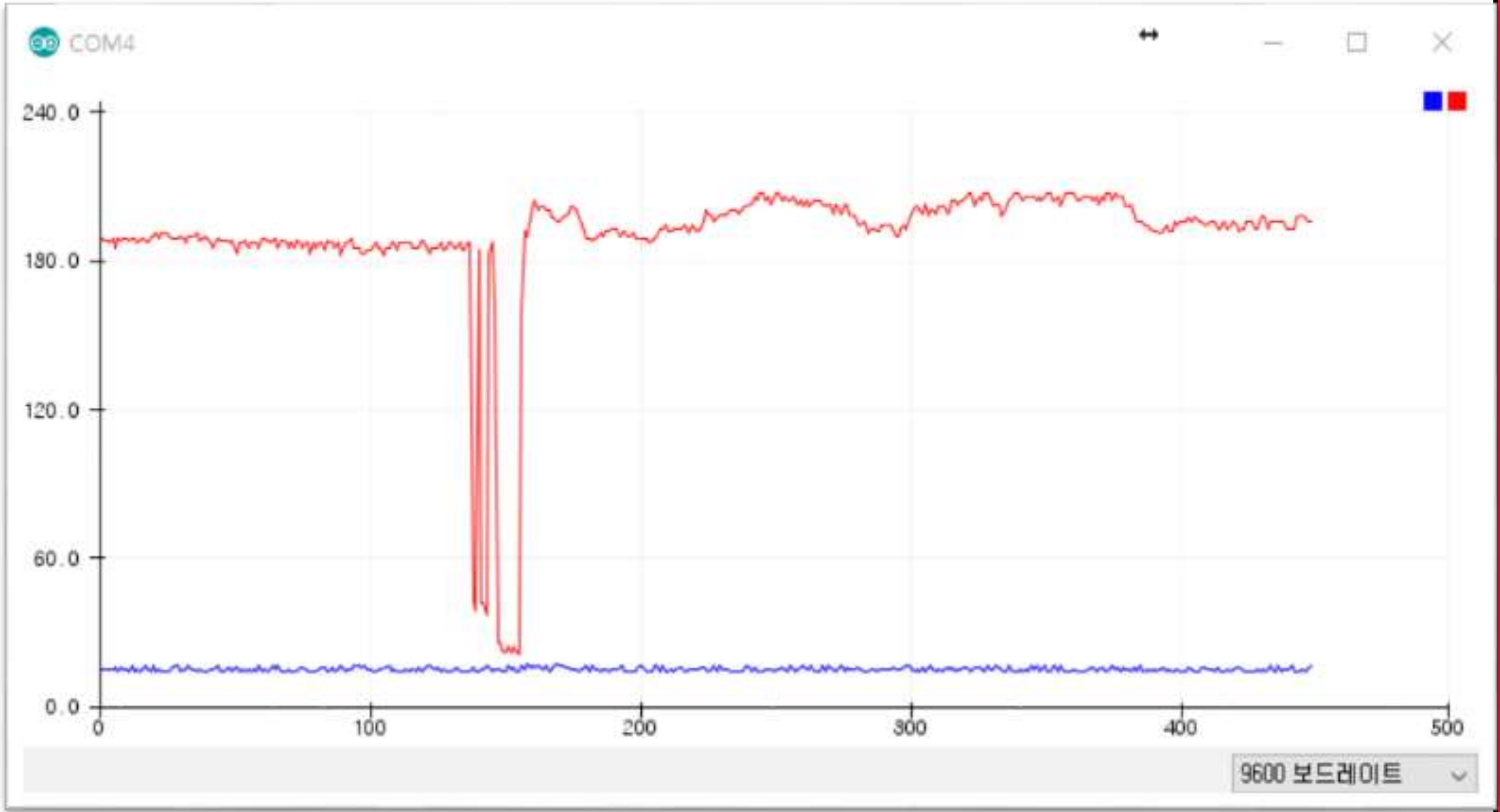


```
void loop() {
  // Temperature from TMP36
  int temp_value = analogRead(TMP36_INPUT);
  // converting that reading to voltage
  float voltage = temp_value * 5.0 * 1000;   // in mV
  voltage /= 1023.0;
  float tempC = (voltage - 500) / 10 ;

  // Lux from CdS (LDR)
  int cds_value = analogRead(CDS_INPUT);
  int lux = int(luminosity(cds_value));
//
  Serial.print(tempC);
  Serial.print(",");
  Serial.println(lux);

  delay(1000);
}

//Voltage to Lux
double luminosity (int RawADC0){
  double Vout=RawADC0*5.0/1023.0;  // 5/1023 (Vin = 5 V)
  int lux=(2500/Vout-500)/10;
  // lux = 500 / Rldr, Vout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
  return lux;
}
```

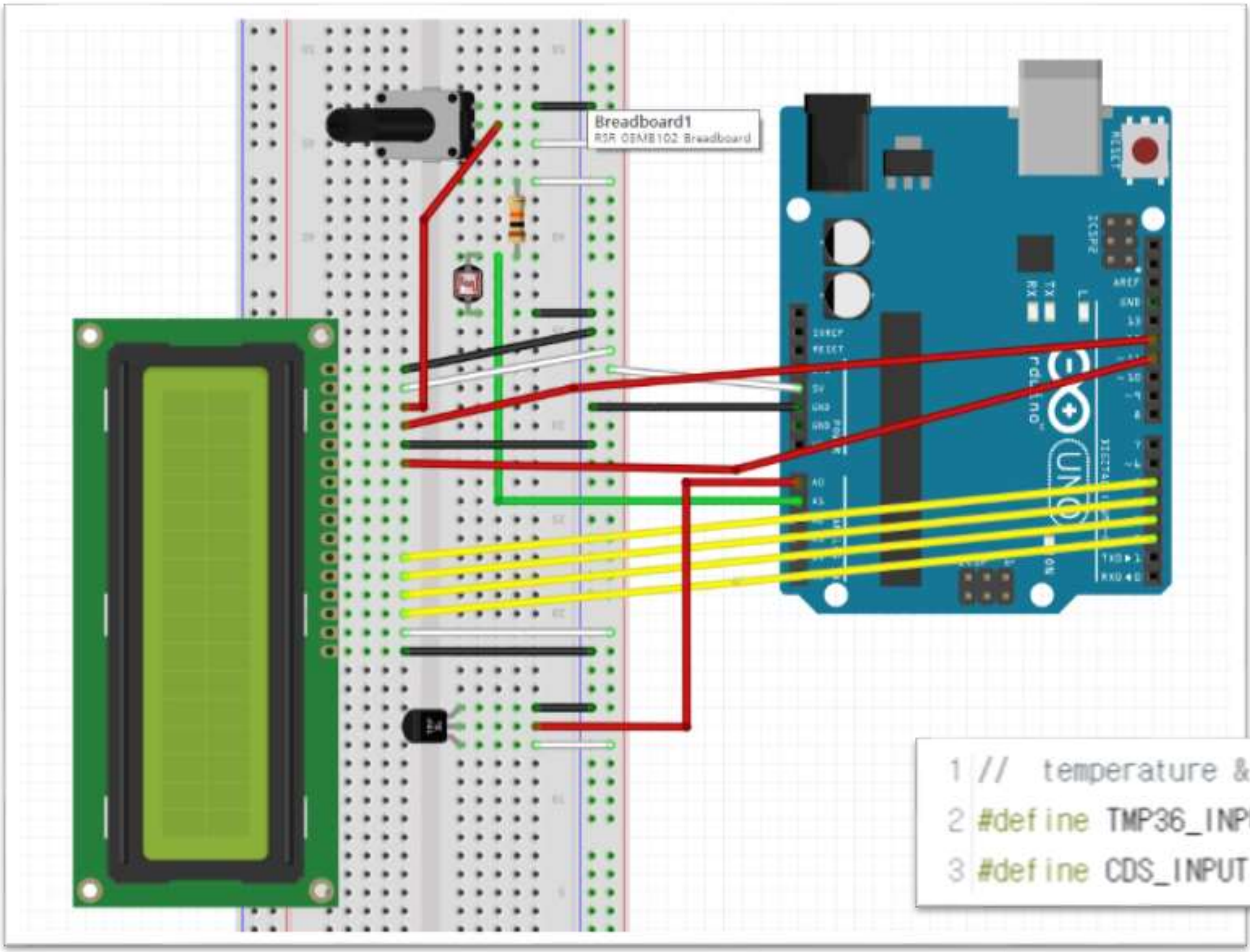
A4.3.3 TMP36 + CdS : Monitoring



```
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
```

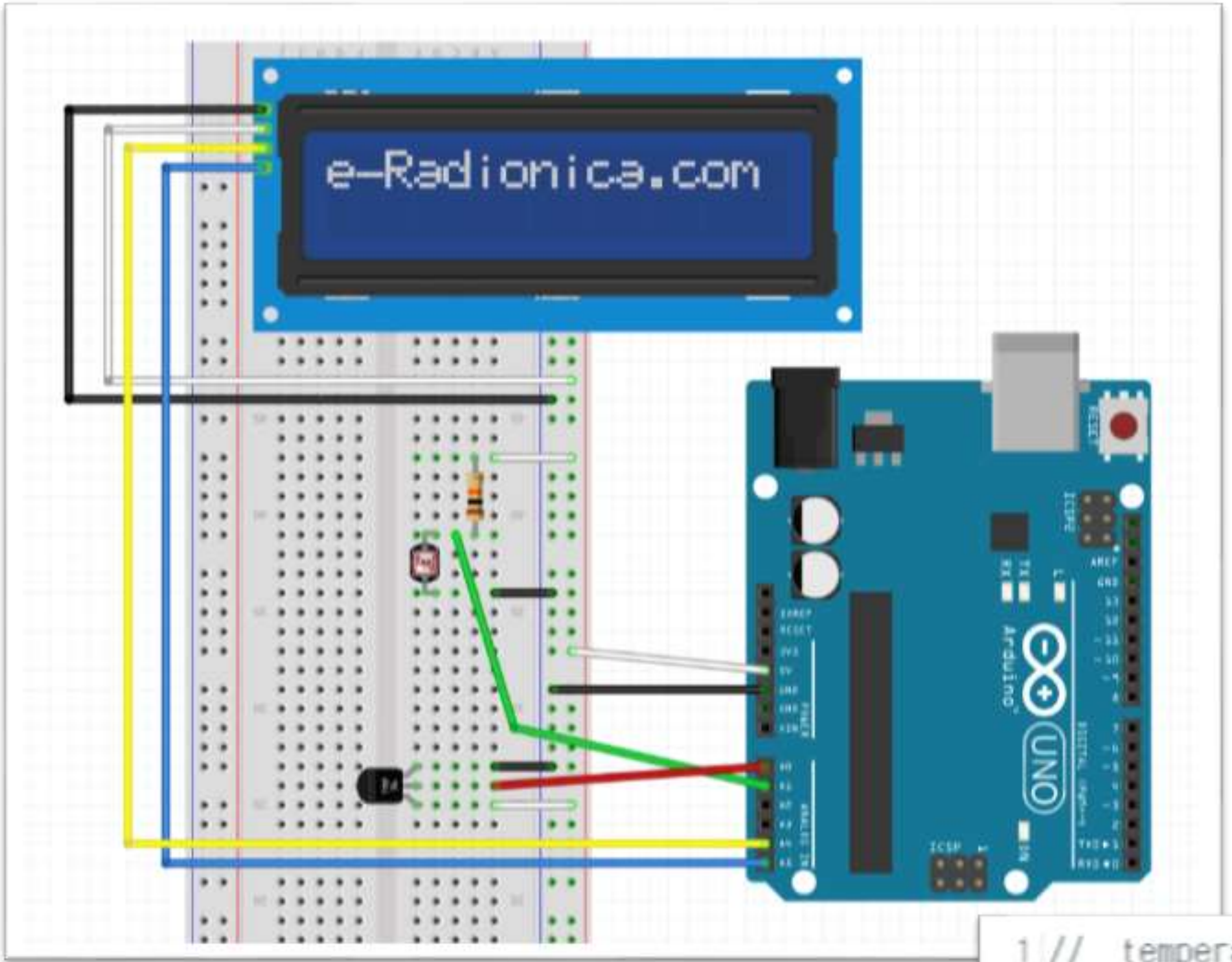


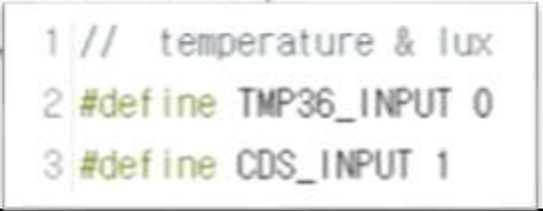
A4.4.1 TMP36 + CdS + LCD : circuit



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

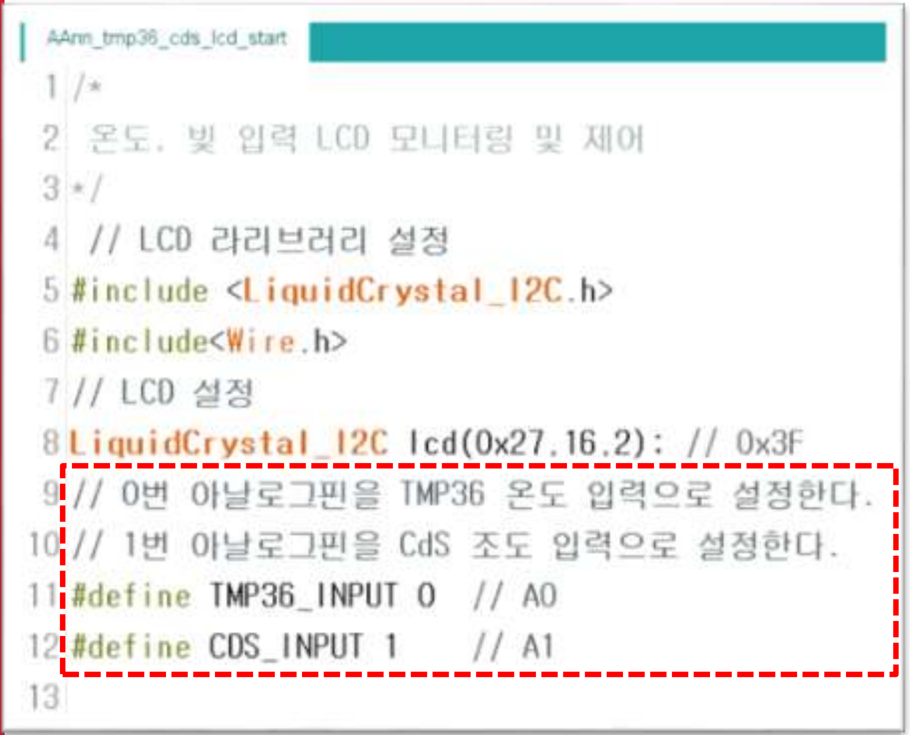
A4.4.1 TMP36 + CdS + LCD : circuit





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

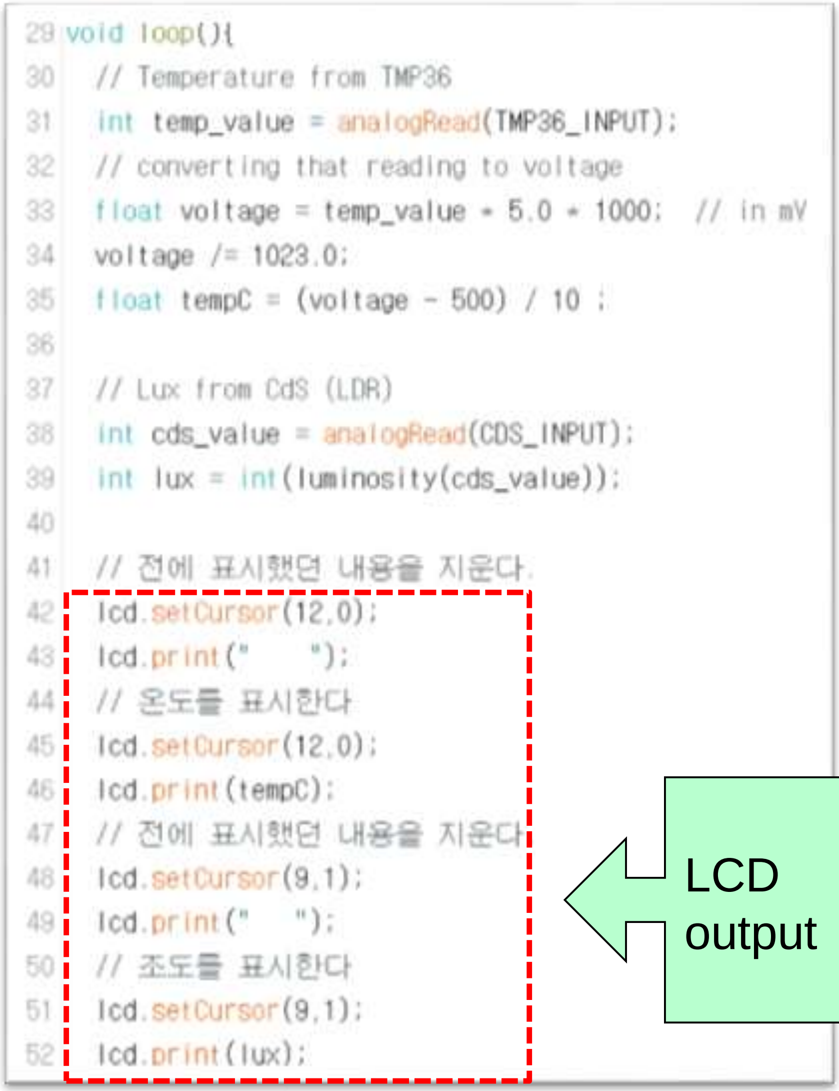
A4.4.2 TMP36 + CdS + LCD : code-1



```
/*
 온도. 빛 입력 LCD 모니터링 및 제어
*/
 // LCD 라리브러리 설정
#include <LiquidCrystal_I2C.h>
#include<Wire.h>
// LCD 설정
LiquidCrystal_I2C lcd(0x27,16,2); // 0x3F
// 0번 아날로그핀을 TMP36 온도 입력으로 설정한다.
// 1번 아날로그핀을 CdS 조도 입력으로 설정한다.
#define TMP36_INPUT 0   // A0
#define CDS_INPUT 1     // A1

```

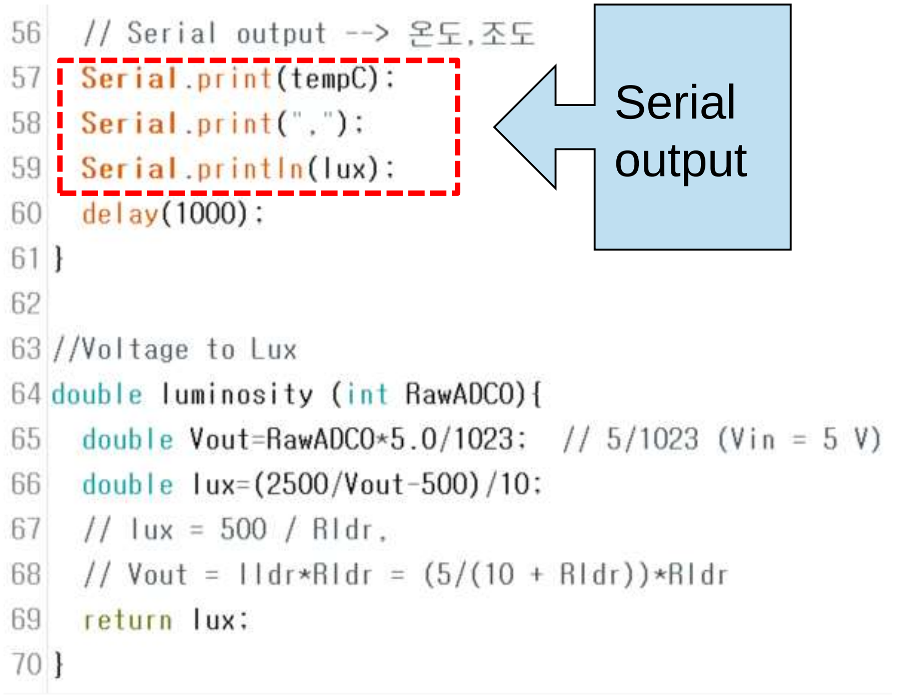
```
void setup() {
  Serial.begin(9600);
// 16X2 LCD 모듈 설정하고 백라이트를 켠다.
  lcd.init();
  lcd.backlight();
// 모든 메세지를 삭제한 뒤
// 숫자를 제외한 부분들을 미리 출력시킨다.
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("AA00,Temp: ");
  lcd.setCursor(0,1);
  lcd.print("Light:  ");
  lcd.setCursor(13,1);
  lcd.print("lux");  //
}
```

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



```
void loop(){
  // Temperature from TMP36
  int temp_value = analogRead(TMP36_INPUT);
  // converting that reading to voltage
  float voltage = temp_value * 5.0 * 1000;  // in mV
  voltage /= 1023.0;
  float tempC = (voltage - 500) / 10 ;

  // Lux from CdS (LDR)
  int cds_value = analogRead(CDS_INPUT);
  int lux = int(luminosity(cds_value));

  // 전에 표시했던 내용을 지운다.
  lcd.setCursor(12,0);
  lcd.print("    ");
  // 온도를 표시한다
  lcd.setCursor(12,0);
  lcd.print(tempC);
  // 전에 표시했던 내용을 지운다.
  lcd.setCursor(9,1);
  lcd.print("   ");
  // 조도를 표시한다
  lcd.setCursor(9,1);
  lcd.print(lux);
```

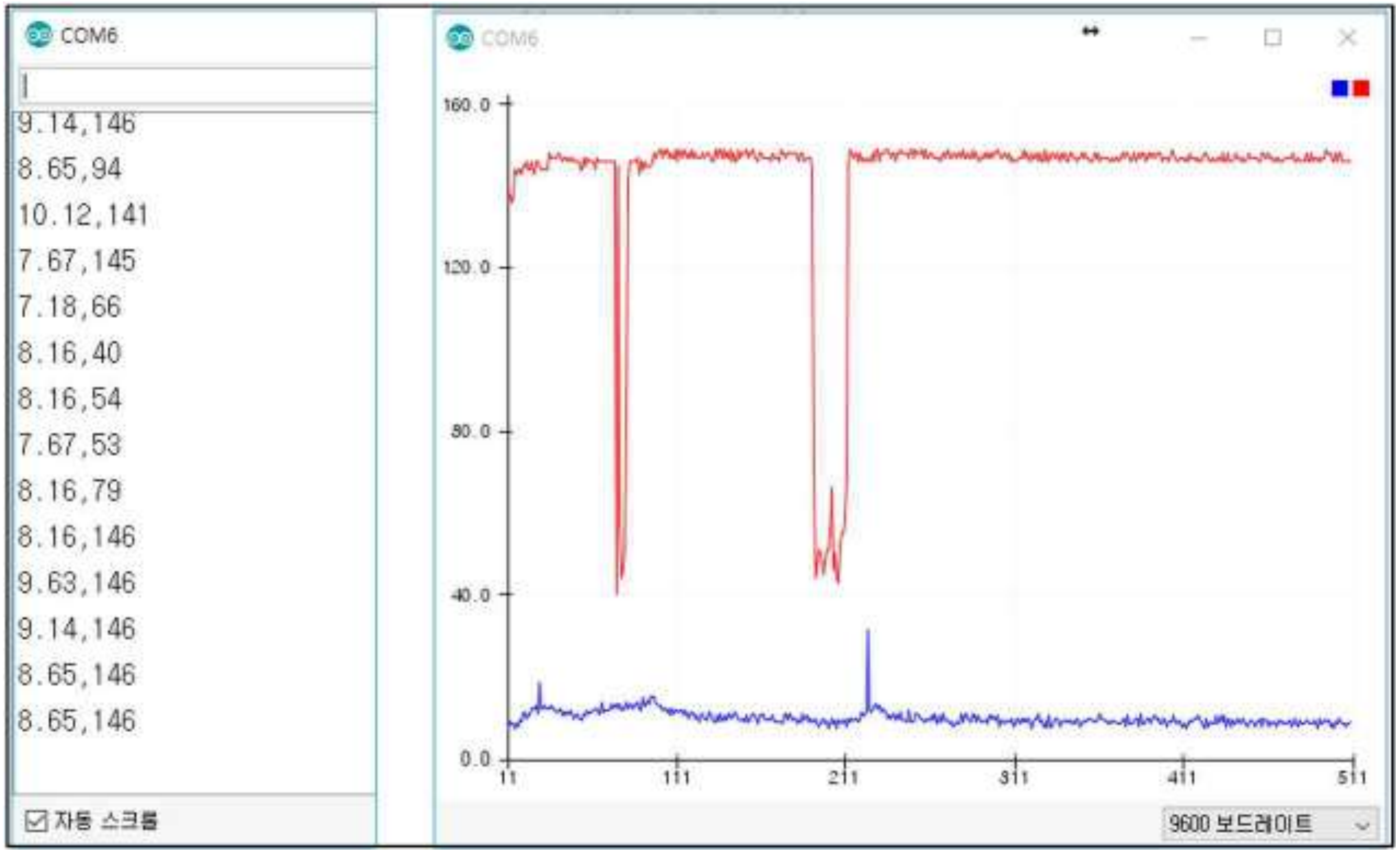
LCD output



```
  // Serial output --> 온도,조도
  Serial.print(tempC);
  Serial.print(",");
  Serial.println(lux);
  delay(1000);
}

//Voltage to Lux
double luminosity (int RawADC0){
  double Vout=RawADC0*5.0/1023;  // 5/1023 (Vin = 5 V)
  double lux=(2500/Vout-500)/10;
  // lux = 500 / Rldr,
  // Vout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
  return lux;
}
```

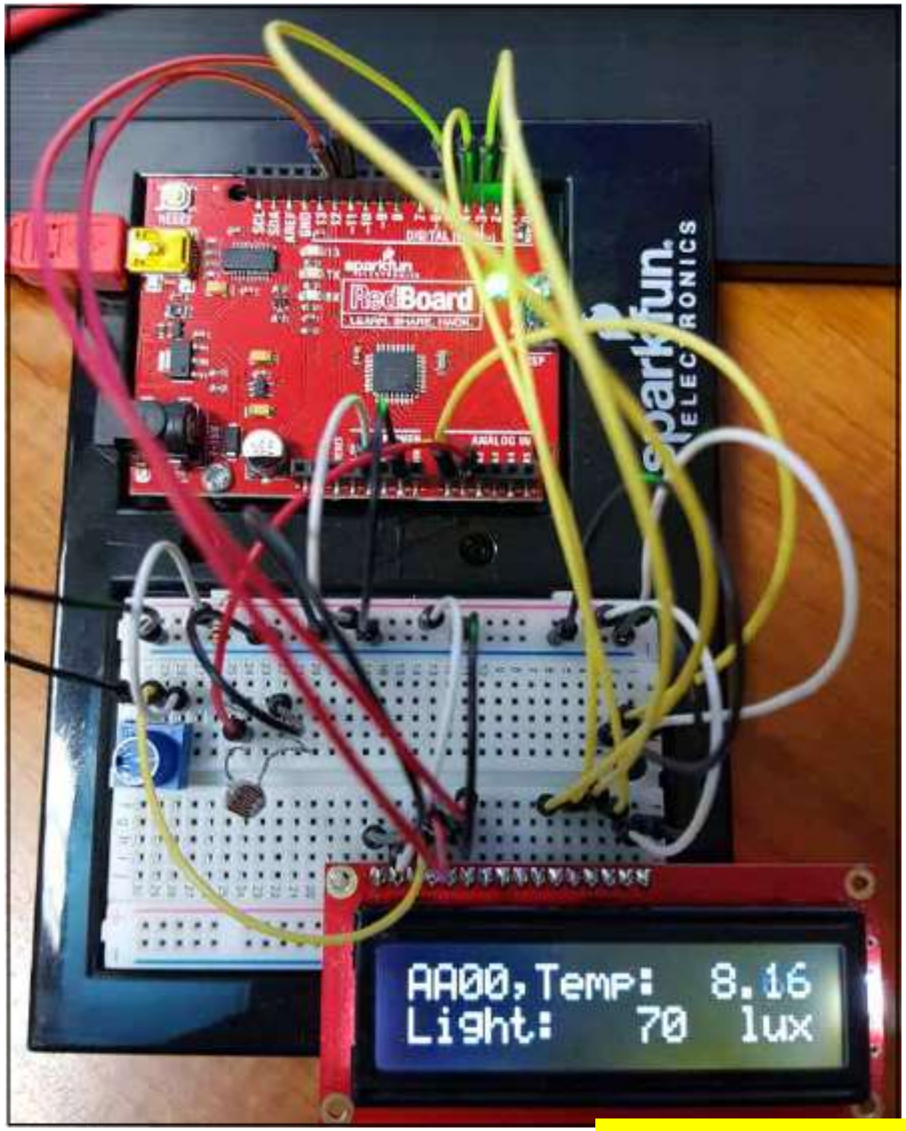
Serial output

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



```
9.14,146
8.65,94
10.12,141
7.67,145
7.18,66
8.16,40
8.16,54
7.67,53
8.16,79
8.16,146
9.63,146
9.14,146
8.65,146
8.65,146
```

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

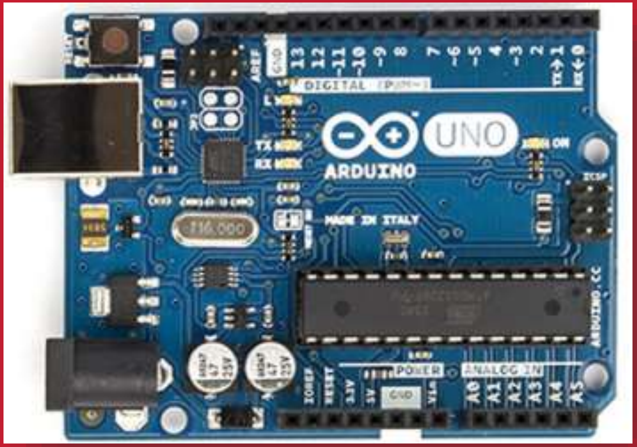


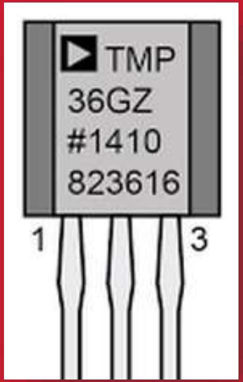
Save as
AAnn_cds_tmp36_lcd.png

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": "aann"

name : cds_tmp36

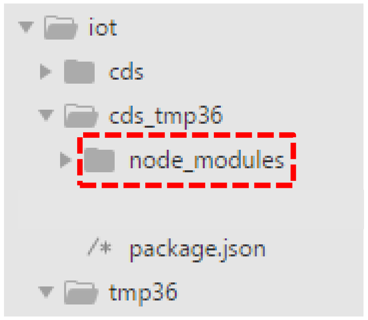
description : cds-tmp36-node project

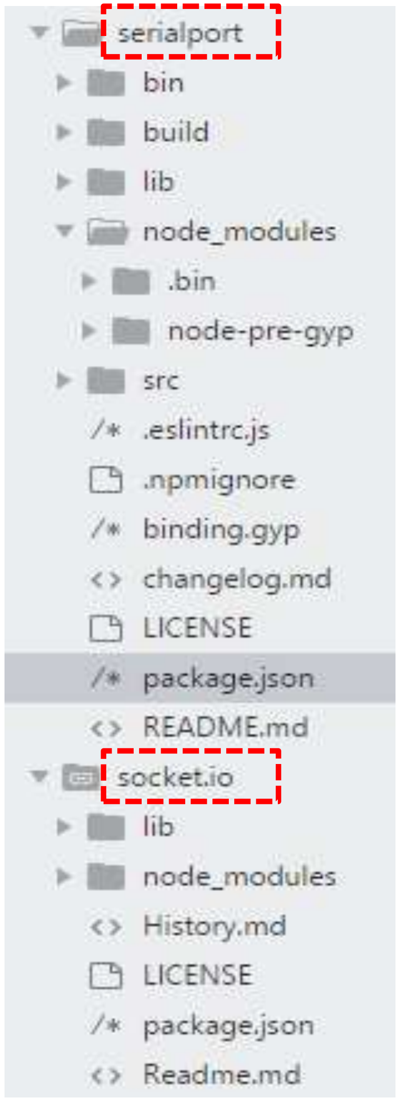
entry point : cds_tmp36_node.js

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

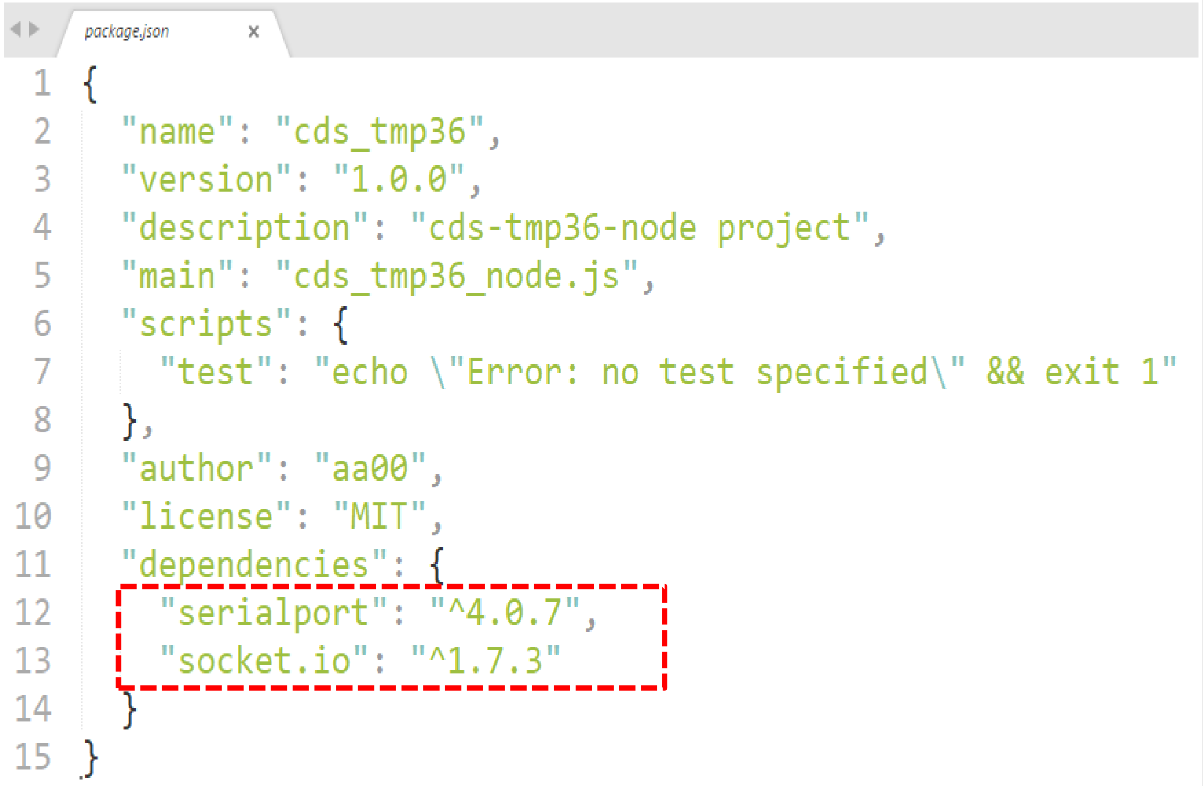
2. Go to cds_tmp36 subfolder

➢ npm init

➢ npm install –save serialport@4.0.7

➢ npm install –save socket.io@1.7.3

package,json

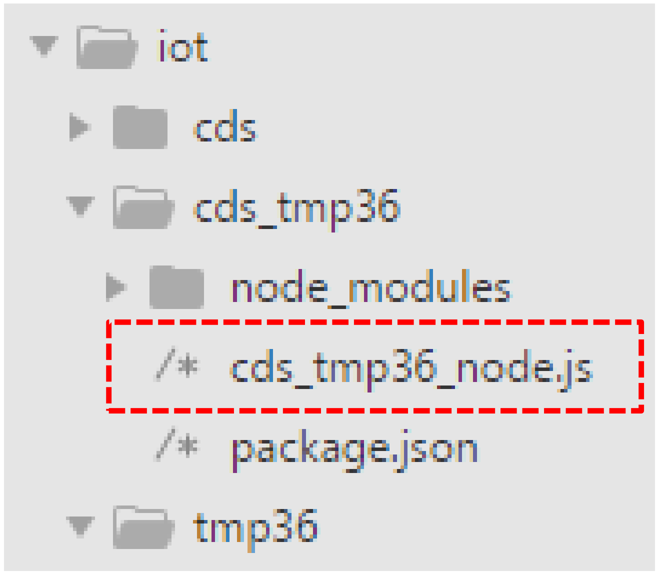

package.json

```
{
  "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"
  },
  "author": "aa00",
  "license": "MIT",
  "dependencies": {
    "serialport": "^4.0.7",
    "socket.io": "^1.7.3"
  }
}
```

A4.5.4 CdS + TMP36 + Node project

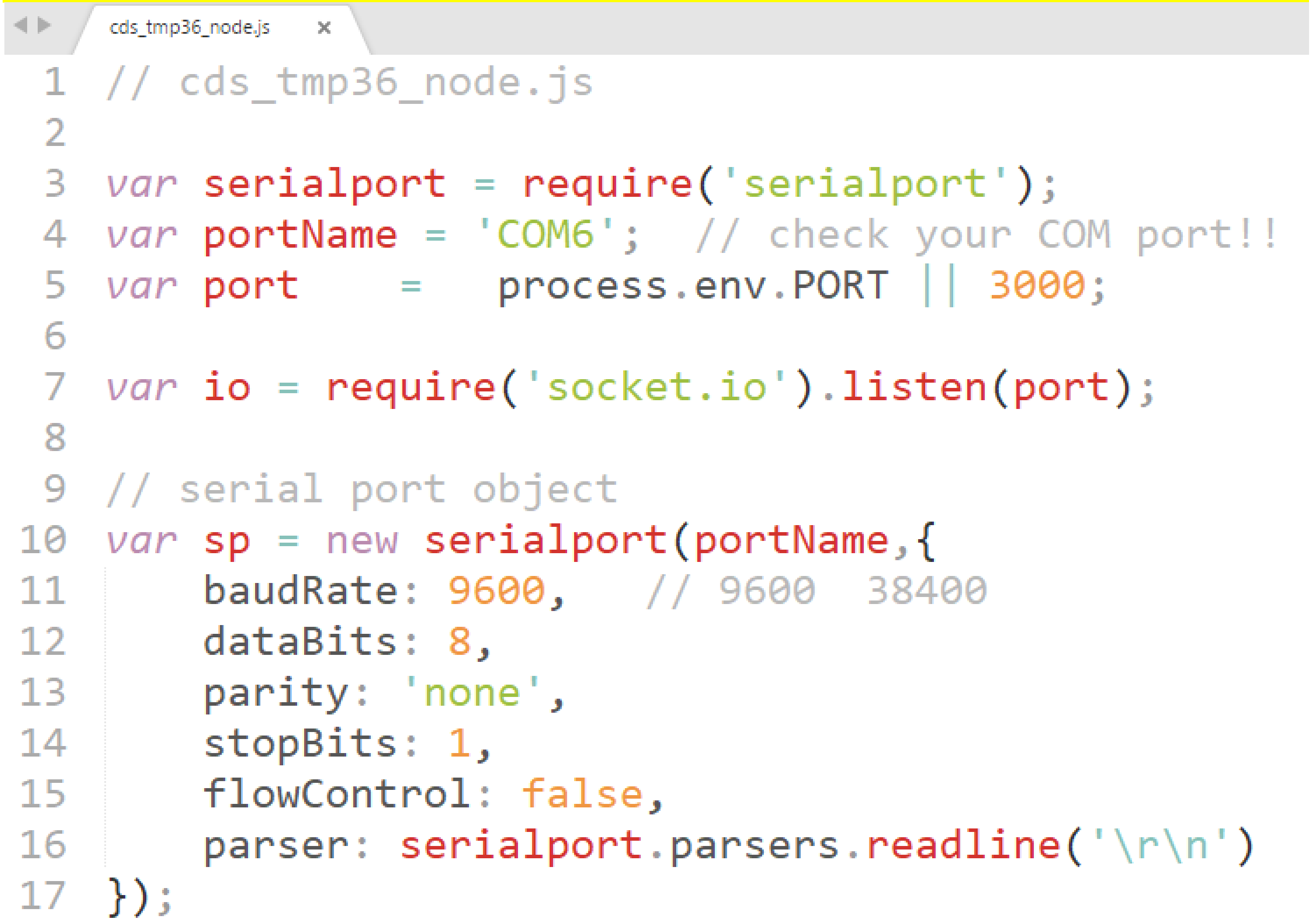
Recycling code:
코드 재활용

Save cds_node.js as
cds_tmp36_node.js



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

cds_tmp36_node.js

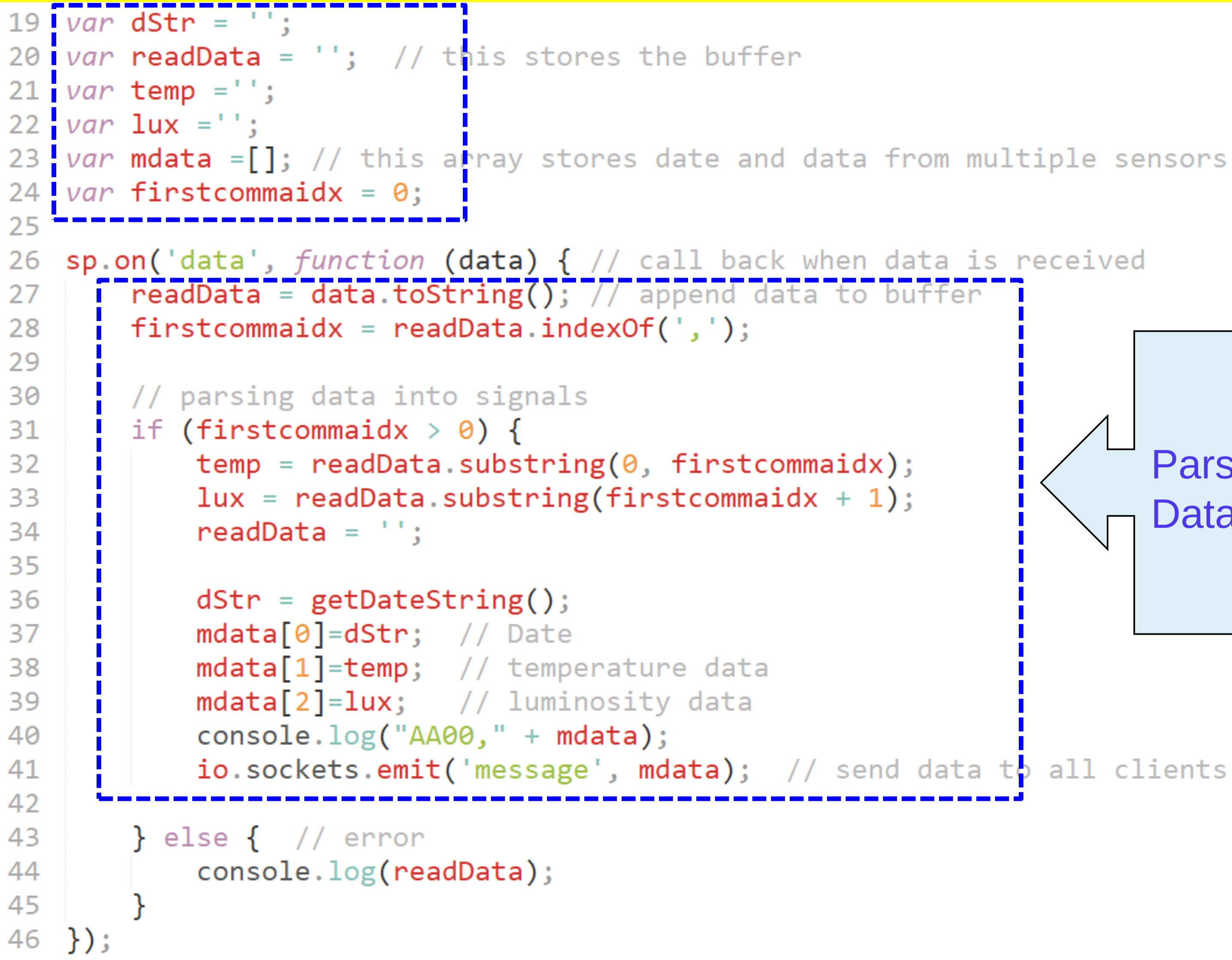


cds_tmp36_node.js

```
// cds_tmp36_node.js

var serialport = require('serialport');
var portName = 'COM6';  // check your COM port!!
var port     =    process.env.PORT || 3000;

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

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

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

cds_tmp36_node.js – parsing data

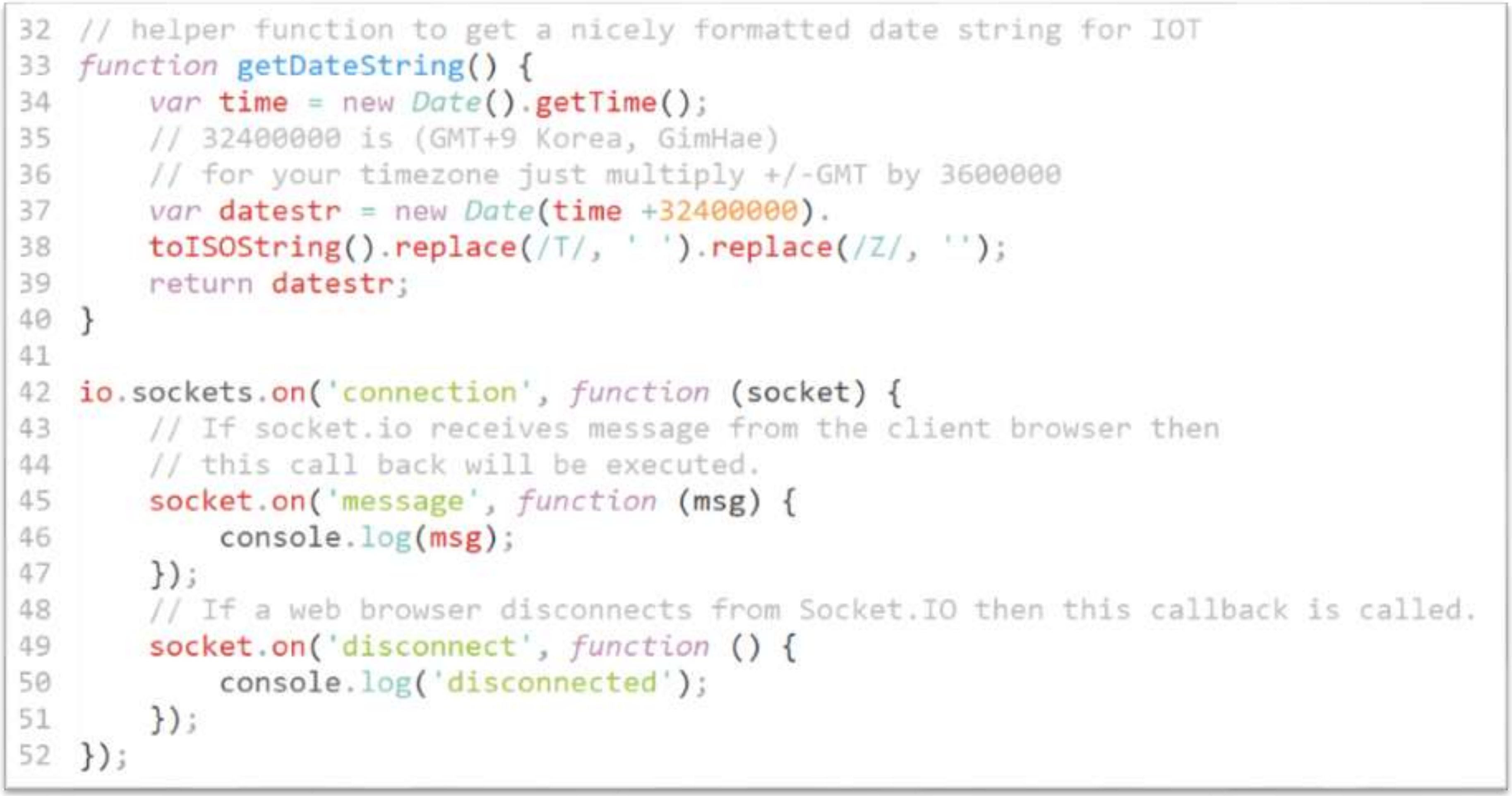


```
var dStr = '';
var readData = '';  // this stores the buffer
var temp ='';
var lux ='';
var mdata =[]; // this array stores date and data from multiple sensors
var firstcommaidx = 0;

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

    // parsing data into signals
    if (firstcommaidx > 0) {
        temp = readData.substring(0, firstcommaidx);
        lux = readData.substring(firstcommaidx + 1);
        readData = '';

        dStr = getDateString();
        mdata[0]=dStr;  // Date
        mdata[1]=temp;  // temperature data
        mdata[2]=lux;   // luminosity data
        console.log("AA00," + mdata);
        io.sockets.emit('message', mdata);  // send data to all clients

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

Parsing Data

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

cds_tmp36_node.js

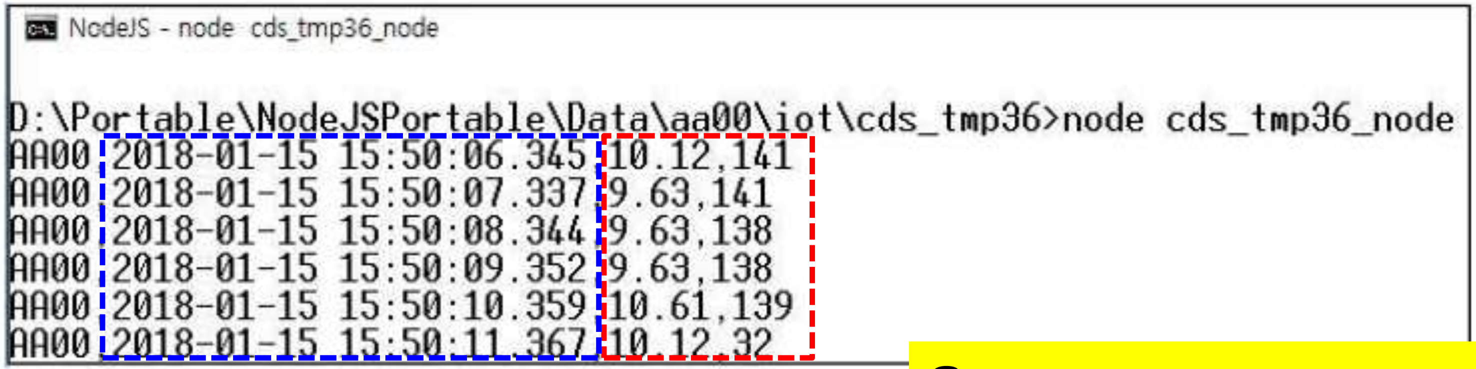


```javascript
// helper function to get a nicely formatted date string for IOT
function getDateString() {
    var time = new Date().getTime();
    // 32400000 is (GMT+9 Korea, GimHae)
    // for your timezone just multiply +/-GMT by 3600000
    var datestr = new Date(time +32400000).
    toISOString().replace(/T/,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

io.sockets.on('connection', function (socket) {
    // If socket.io receives message from the client browser then
    // this call back will be executed.
    socket.on('message', function (msg) {
        console.log(msg);
    });
    // If a web browser disconnects from Socket.IO then this callback is called.
    socket.on('disconnect', function () {
        console.log('disconnected');
    });
});
```

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

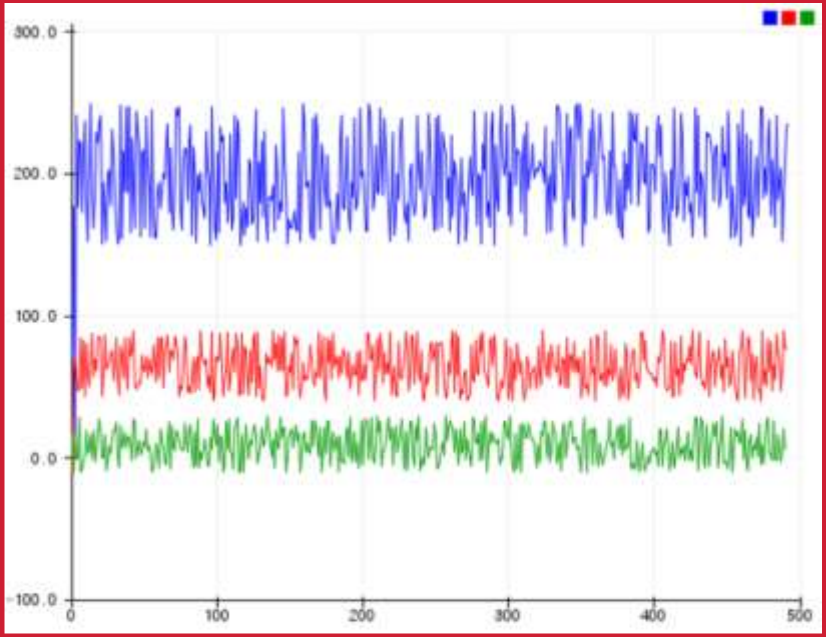
시간, 온도,조도

Save as
AAnn_cds_tmp36_IOT.png

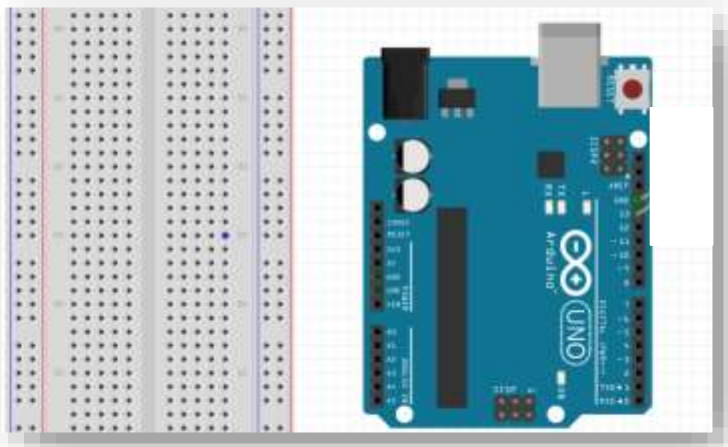
[DIY] Multi-signals

다중신호 시뮬레이션

+ node.js



DIY – 스케치



아두이노에서 LED와 저항을 모두 제거하고 USB만 컴퓨터와 연결한다.

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

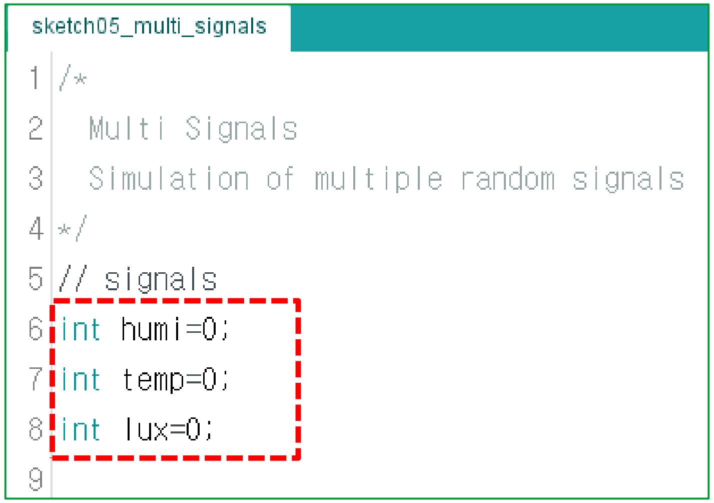
▸ 스케치 구성

1. 3 개의 신호를 담을 변수를 초기화한다.

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

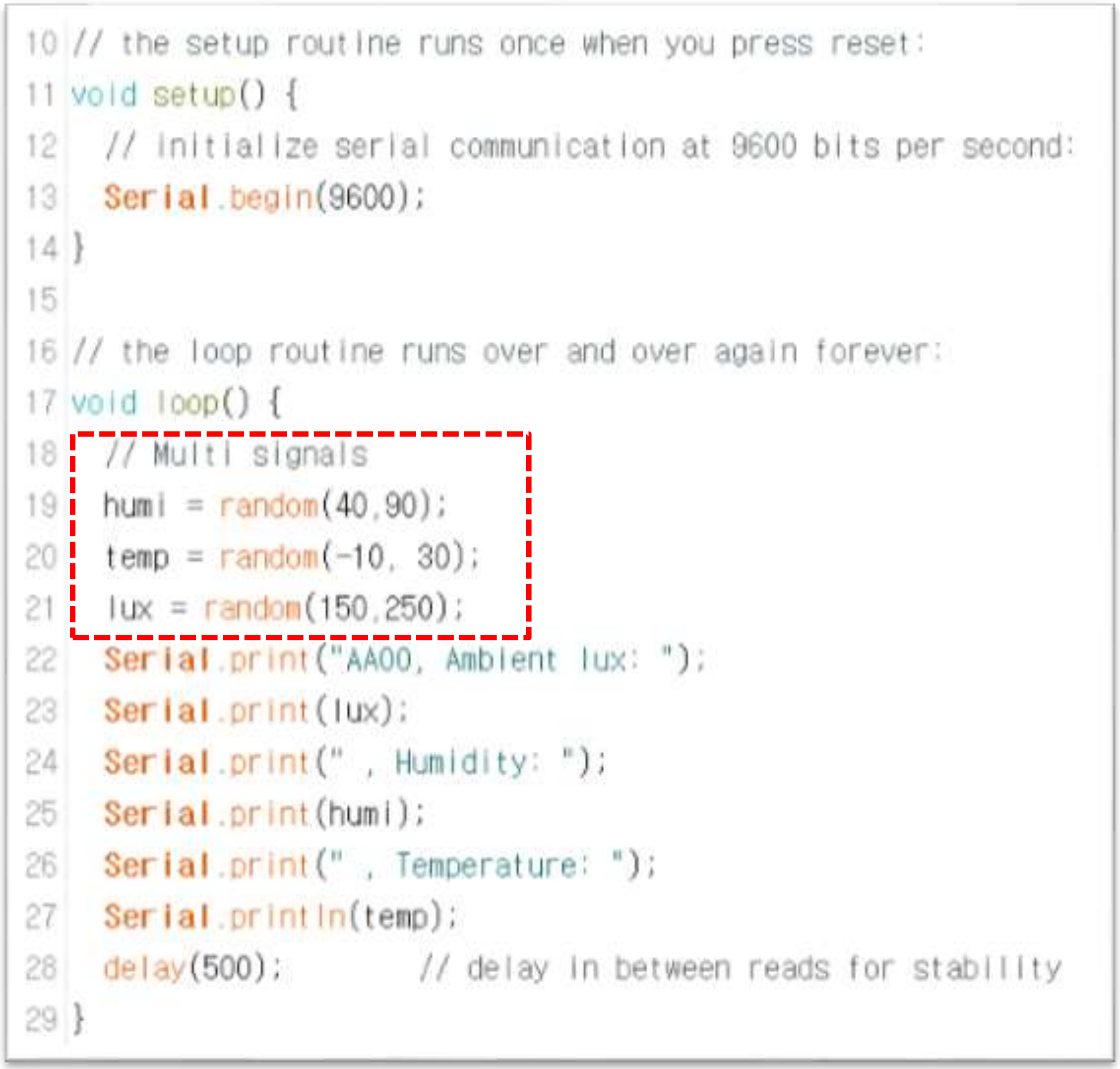
3. loop()에서 마구잡이 수를 세 개 발생시켜서 직렬 통신으로 3 개의 pwm 값을 각각 컴퓨터로 전송한다.

DIY – code



sketch05_multi_signals

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

```

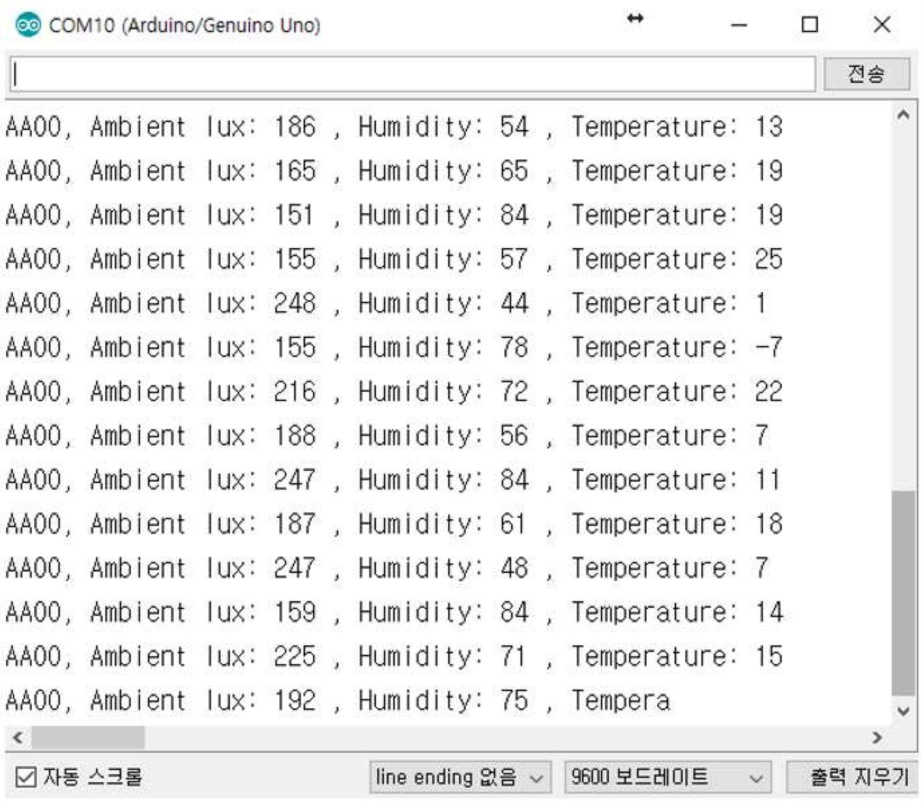


```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// 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("AA00, Ambient lux: ");
  Serial.print(lux);
  Serial.print(" , Humidity: ");
  Serial.print(humi);
  Serial.print(" , Temperature: ");
  Serial.println(temp);
  delay(500);          // delay in between reads for stability
}
```

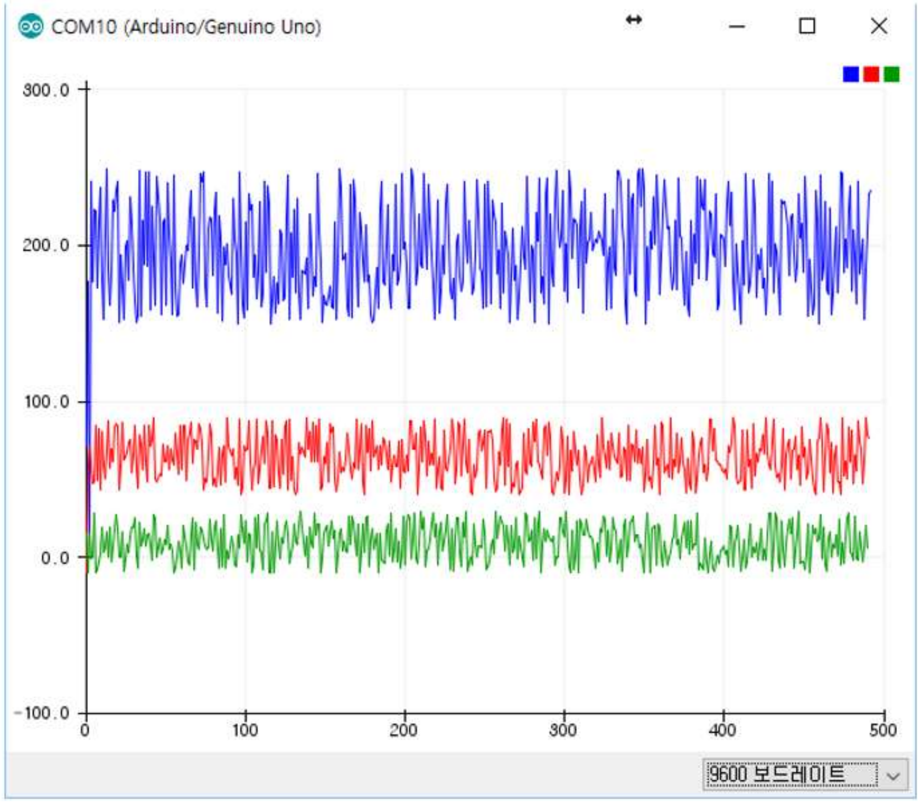
DIY – result

DIY 결과

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

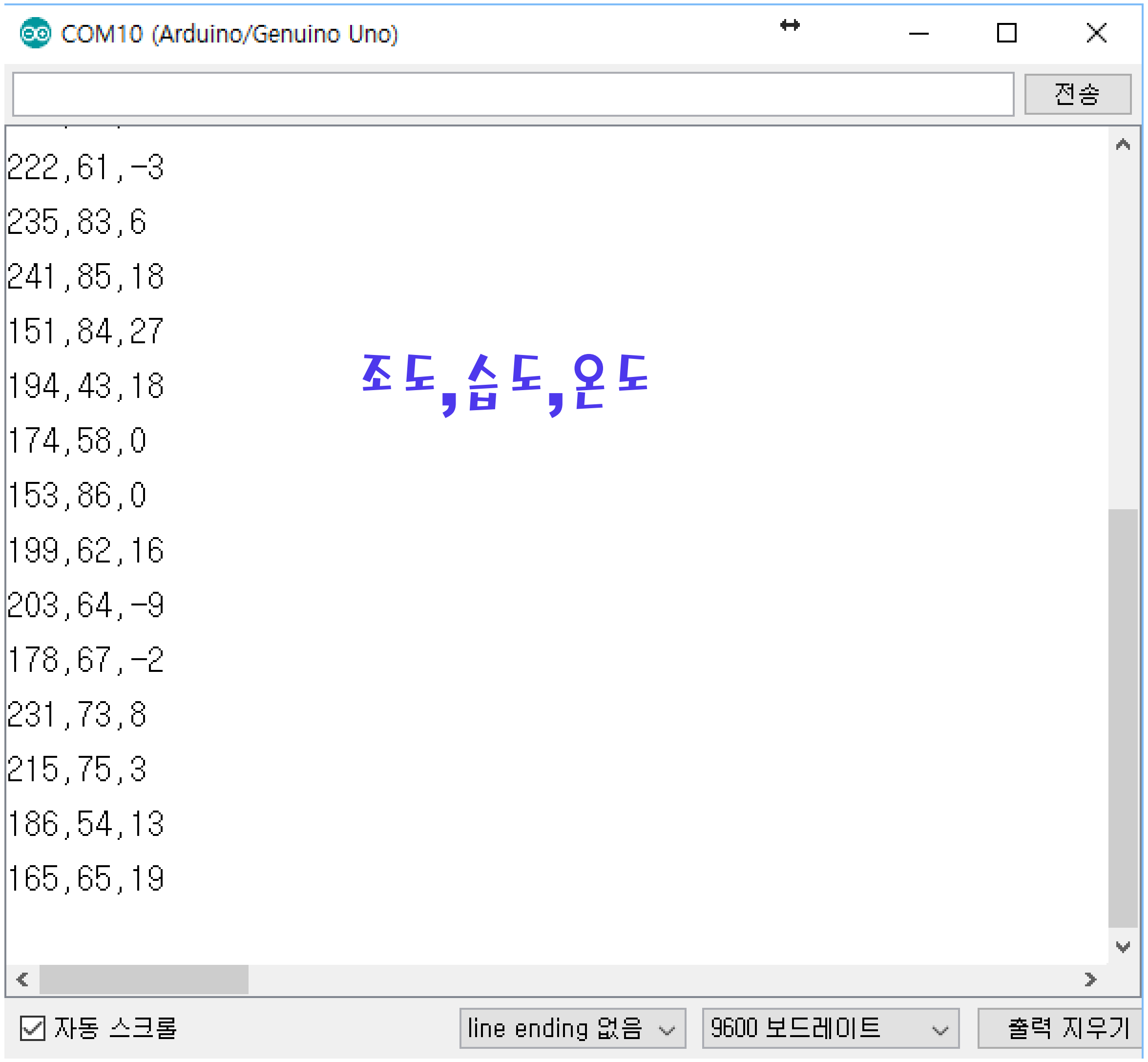


```
AA00, Ambient lux: 186 , Humidity: 54 , Temperature: 13
AA00, Ambient lux: 165 , Humidity: 65 , Temperature: 19
AA00, Ambient lux: 151 , Humidity: 84 , Temperature: 19
AA00, Ambient lux: 155 , Humidity: 57 , Temperature: 25
AA00, Ambient lux: 248 , Humidity: 44 , Temperature: 1
AA00, Ambient lux: 155 , Humidity: 78 , Temperature: -7
AA00, Ambient lux: 216 , Humidity: 72 , Temperature: 22
AA00, Ambient lux: 188 , Humidity: 56 , Temperature: 7
AA00, Ambient lux: 247 , Humidity: 84 , Temperature: 11
AA00, Ambient lux: 187 , Humidity: 61 , Temperature: 18
AA00, Ambient lux: 247 , Humidity: 48 , Temperature: 7
AA00, Ambient lux: 159 , Humidity: 84 , Temperature: 14
AA00, Ambient lux: 225 , Humidity: 71 , Temperature: 15
AA00, Ambient lux: 192 , Humidity: 75 , Tempera
```



DIY – New result 1

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



COM10 (Arduino/Genuino Uno)

```
222,61,-3
235,83,6
241,85,18
151,84,27
194,43,18
174,58,0
153,86,0
199,62,16
203,64,-9
178,67,-2
231,73,8
215,75,3
186,54,13
165,65,19
```

조도,습도,온도

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

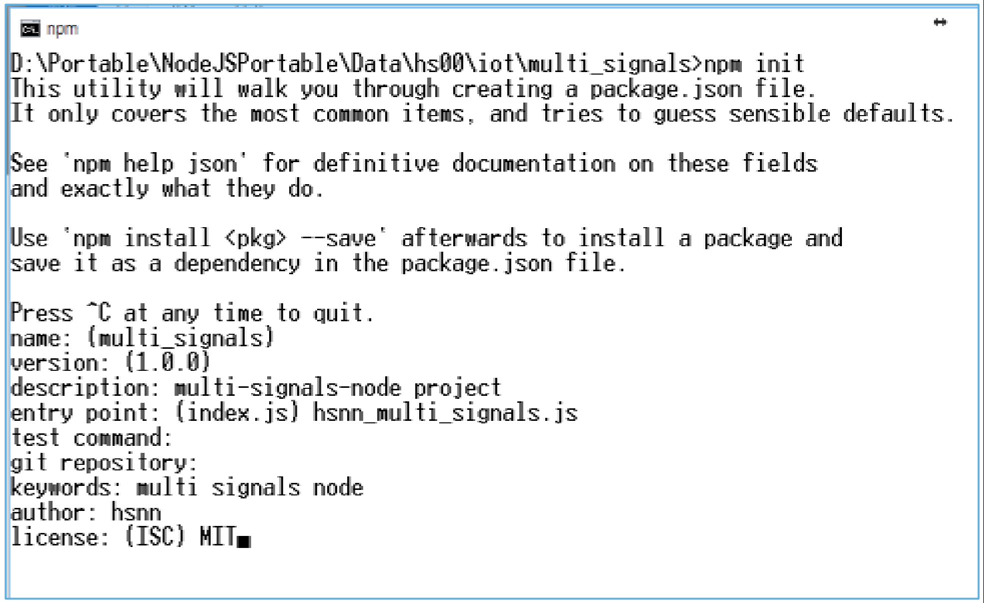
entry point : aann_multi_signals.js

author : aann

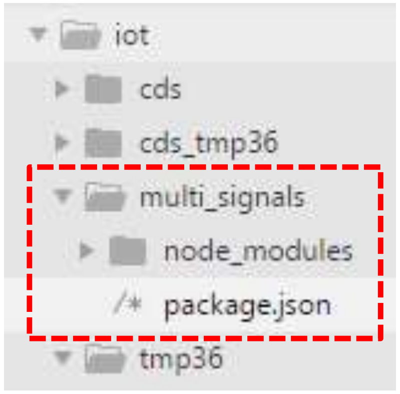
3. Install node modules

➢ npm install –save serialport@4.0.7

➢ npm install –save socket.io@1.7.3



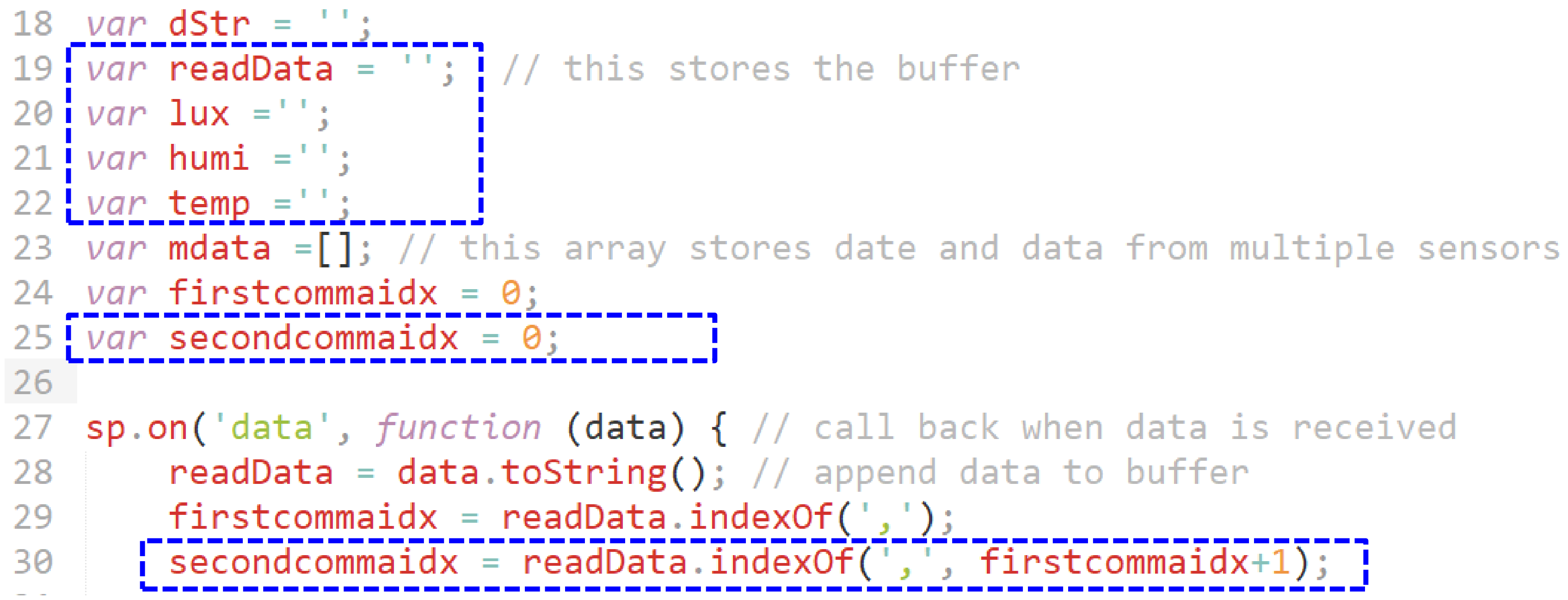
```
D:\Portable\NodeJSPortable\Data\hs00\iot\multi_signals>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

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
AAnn_multi_signals.js in multi_signals subfolder



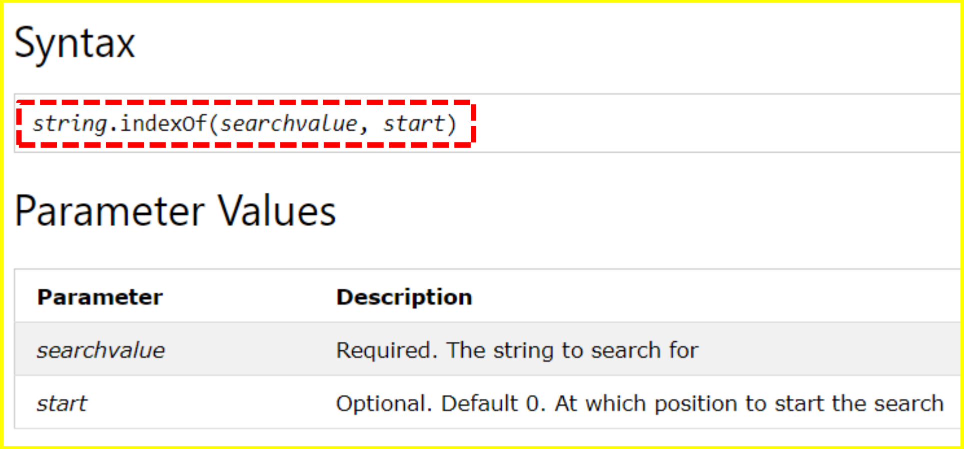
```
var dStr = '';
var readData = ''; // this stores the buffer
var lux ='';
var humi ='';
var temp ='';
var mdata =[]; // this array stores date and data from multiple sensors
var firstcommaidx = 0;
var secondcommaidx = 0;

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);
```

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



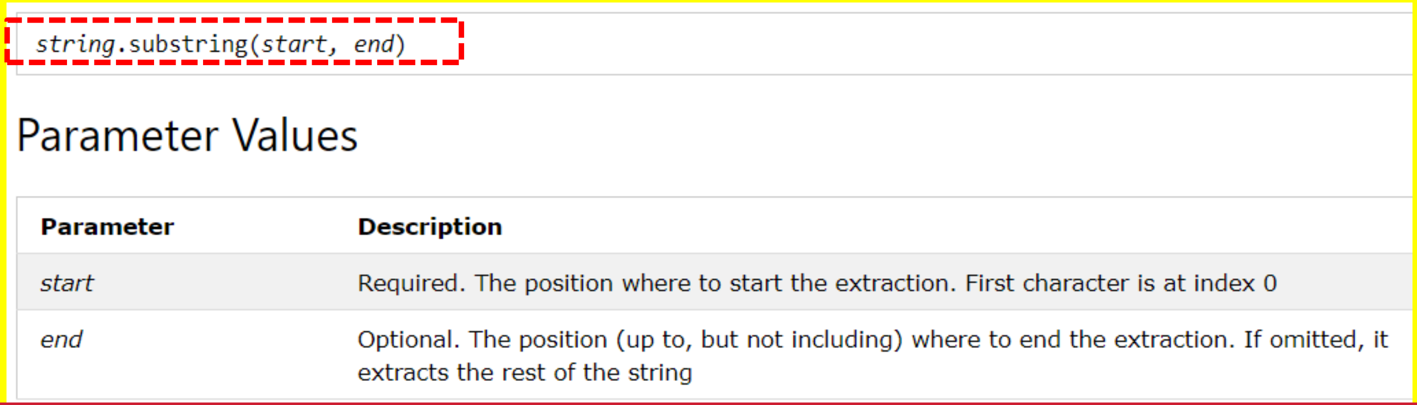
Syntax

```
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
    if (firstcommaidx > 0) {
```

아두이노가 직렬통신으로 전송하는 2 개의 comma (,)로 구분된

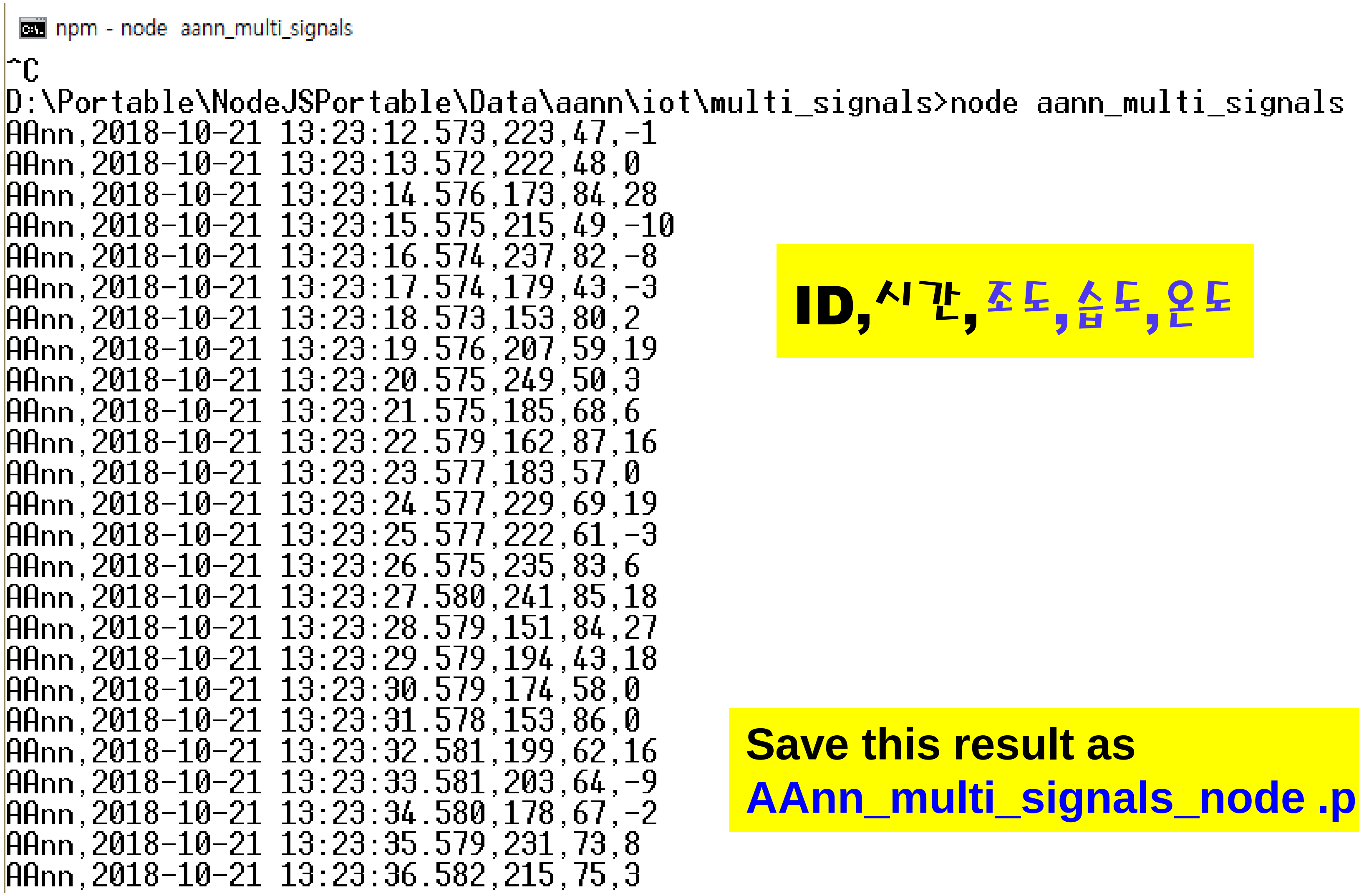
조도,습도,온도 데이터 메시지를 **parsing**하여 **mdata** 배열에 담는 코드를 완성하시오.

substring() 함수에서 firstcommaidx, secondcommaidx를 잘 이용하시오.

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

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

DIY – New result 2-5

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

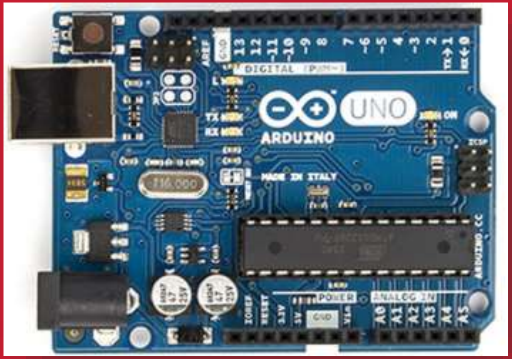


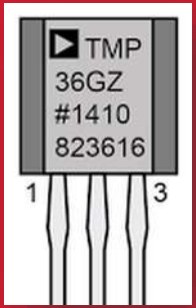
```
npm - node  aann_multi_signals
^C
D:\Portable\NodeJSPortable\Data\aann\iot\multi_signals>node aann_multi_signals
AAnn,2018-10-21 13:23:12.573,223,47,-1
AAnn,2018-10-21 13:23:13.572,222,48,0
AAnn,2018-10-21 13:23:14.576,173,84,28
AAnn,2018-10-21 13:23:15.575,215,49,-10
AAnn,2018-10-21 13:23:16.574,237,82,-8
AAnn,2018-10-21 13:23:17.574,179,43,-3
AAnn,2018-10-21 13:23:18.573,153,80,2
AAnn,2018-10-21 13:23:19.576,207,59,19
AAnn,2018-10-21 13:23:20.575,249,50,3
AAnn,2018-10-21 13:23:21.575,185,68,6
AAnn,2018-10-21 13:23:22.579,162,87,16
AAnn,2018-10-21 13:23:23.577,183,57,0
AAnn,2018-10-21 13:23:24.577,229,69,19
AAnn,2018-10-21 13:23:25.577,222,61,-3
AAnn,2018-10-21 13:23:26.575,235,83,6
AAnn,2018-10-21 13:23:27.580,241,85,18
AAnn,2018-10-21 13:23:28.579,151,84,27
AAnn,2018-10-21 13:23:29.579,194,43,18
AAnn,2018-10-21 13:23:30.579,174,58,0
AAnn,2018-10-21 13:23:31.578,153,86,0
AAnn,2018-10-21 13:23:32.581,199,62,16
AAnn,2018-10-21 13:23:33.581,203,64,-9
AAnn,2018-10-21 13:23:34.580,178,67,-2
AAnn,2018-10-21 13:23:35.579,231,73,8
AAnn,2018-10-21 13:23:36.582,215,75,3
```

ID,시간,조도,습도,온도

Save this result as
AAnn_multi_signals_node .png

Next week

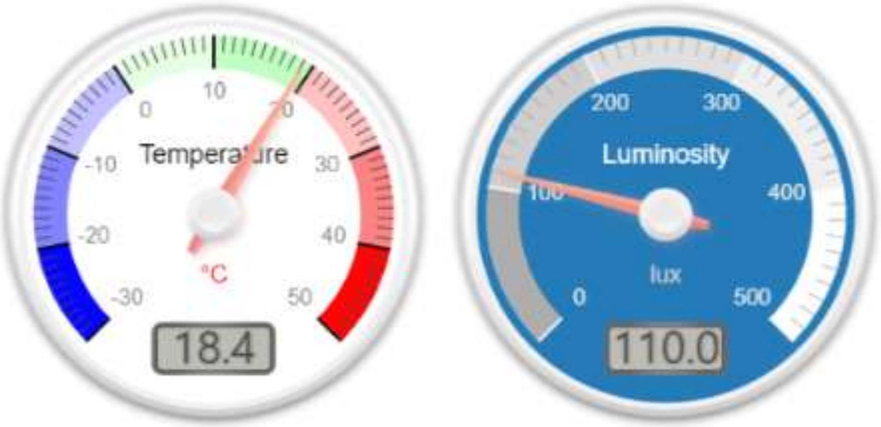




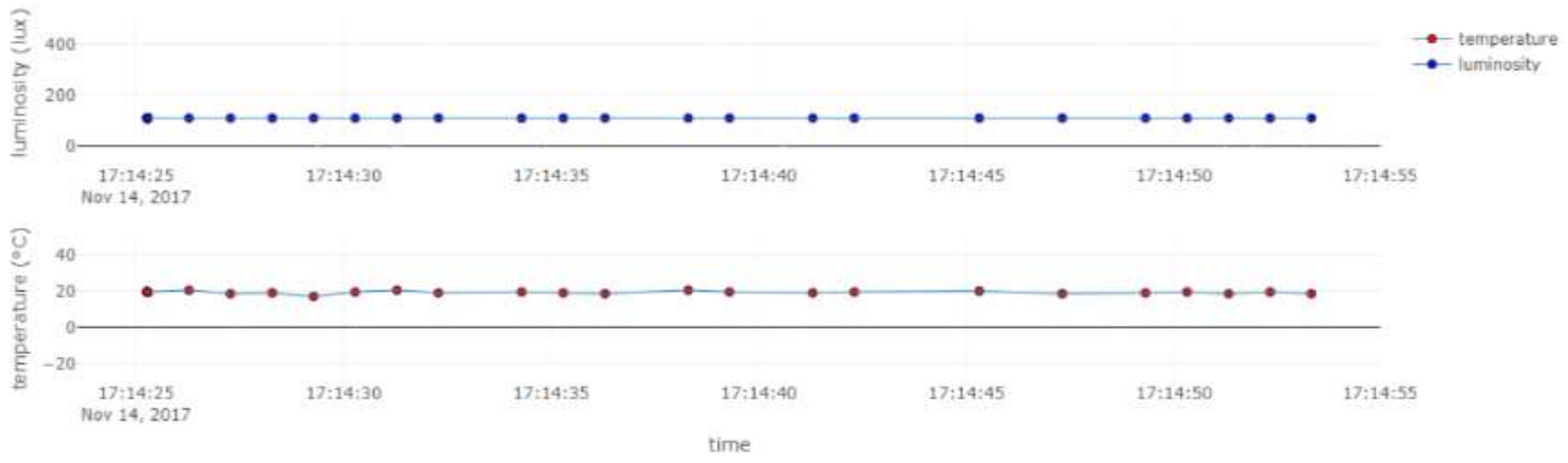
Data visualization using ploy.ly

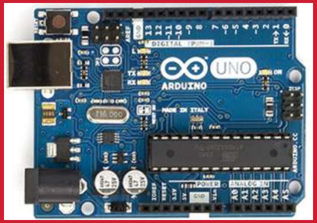
Line Charts
Scatter Plots

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



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





[Practice]

- [wk06]
  - Arduino sensors + Node.js
  - Complete your project
  - Upload folder: Aann_Rpt06

wk06 : Practice : AAnn_Rpt06

◆ **[Target of this week]**

- Complete your works
- **Save your outcomes and upload outputs in github**

제출폴더명 : **AAnn_Rpt06**

- 압축할 파일들

① **AAnn_cds_IOT_data.png**

② **AAnn_cds_tmp36_lcd.png**

③ **AAnn_cds_tmp36_IOT.png**

④ **AAnn_multi_signals_node.png**

⑤ **All *.ino**

⑥ **All *.js**

[Upload to github]

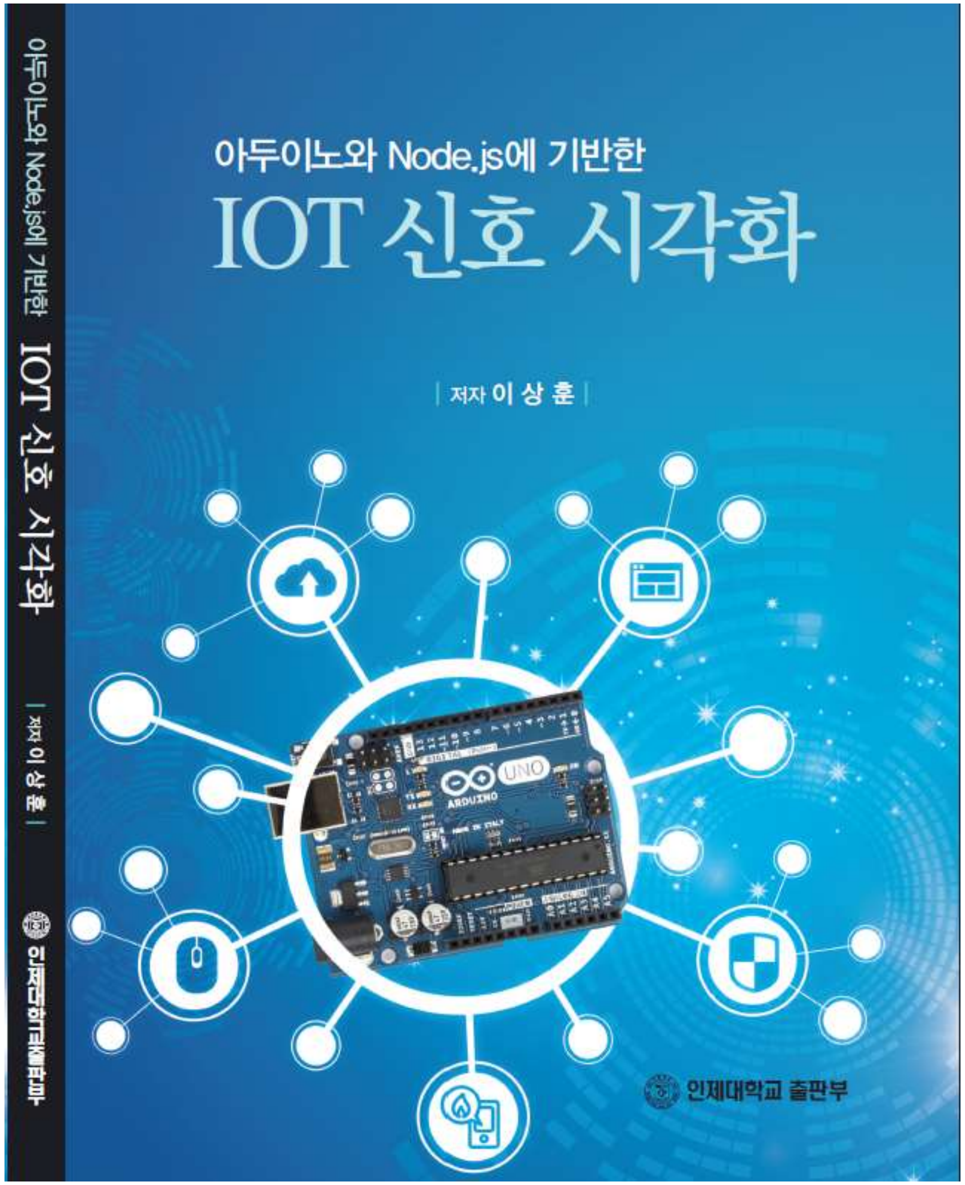
- [wk06]
  - upload all work of this week
  - Use repo "aann" in github
  - upload folder "aann_rpt06" in your github.

Lecture materials

- **References & good sites**

  - ✓ http://www.arduino.cc Arduino Homepage
  - ✓ http://www.nodejs.org/ko Node.js
  - ✓ https://plot.ly/ plotly
  - ✓ https://www.mongodb.com/ MongoDB
  - ✓ http://www.w3schools.com By w3schools
  - ✓ http://www.github.com GitHub

주교재 및 참고도서

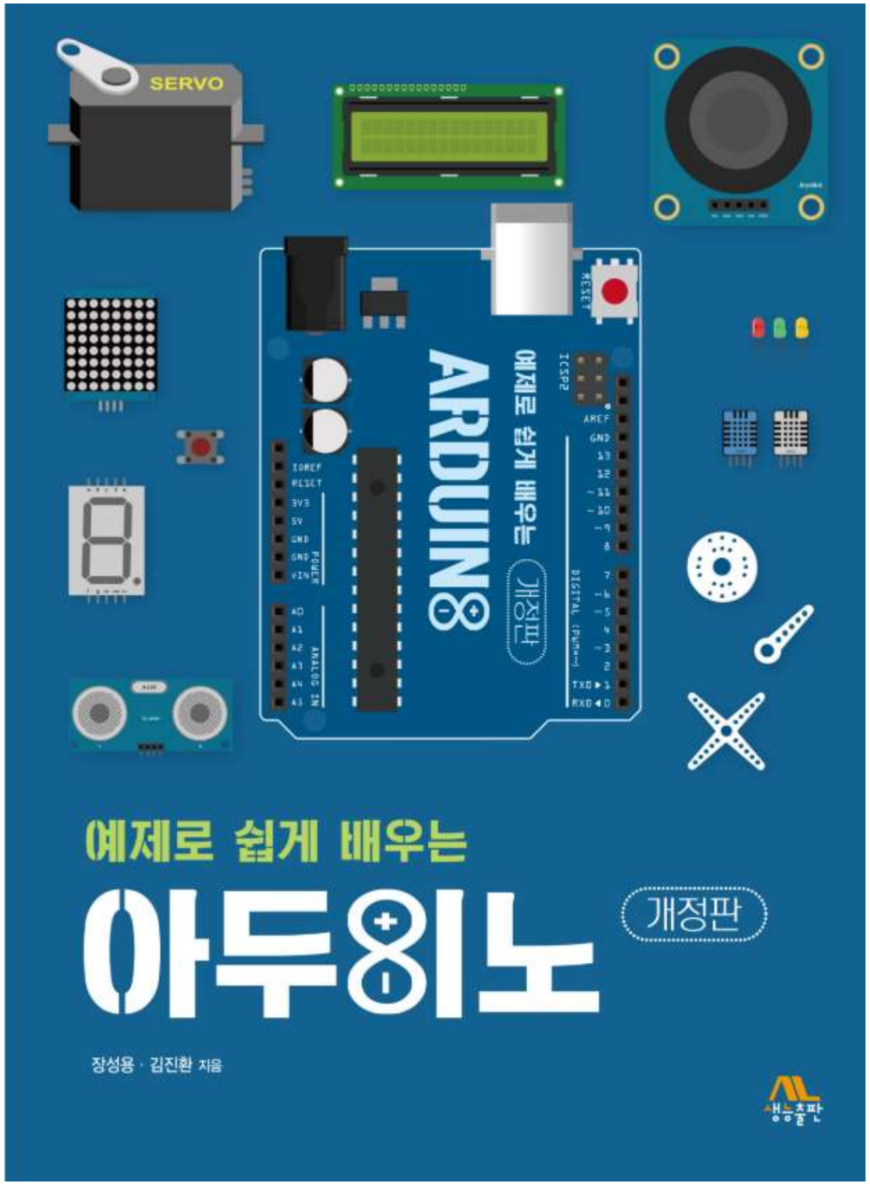


아두이노와 Node.js에 기반한

IOT 신호 시각화

| 저자 이 상 훈 |

인제대학교 출판부



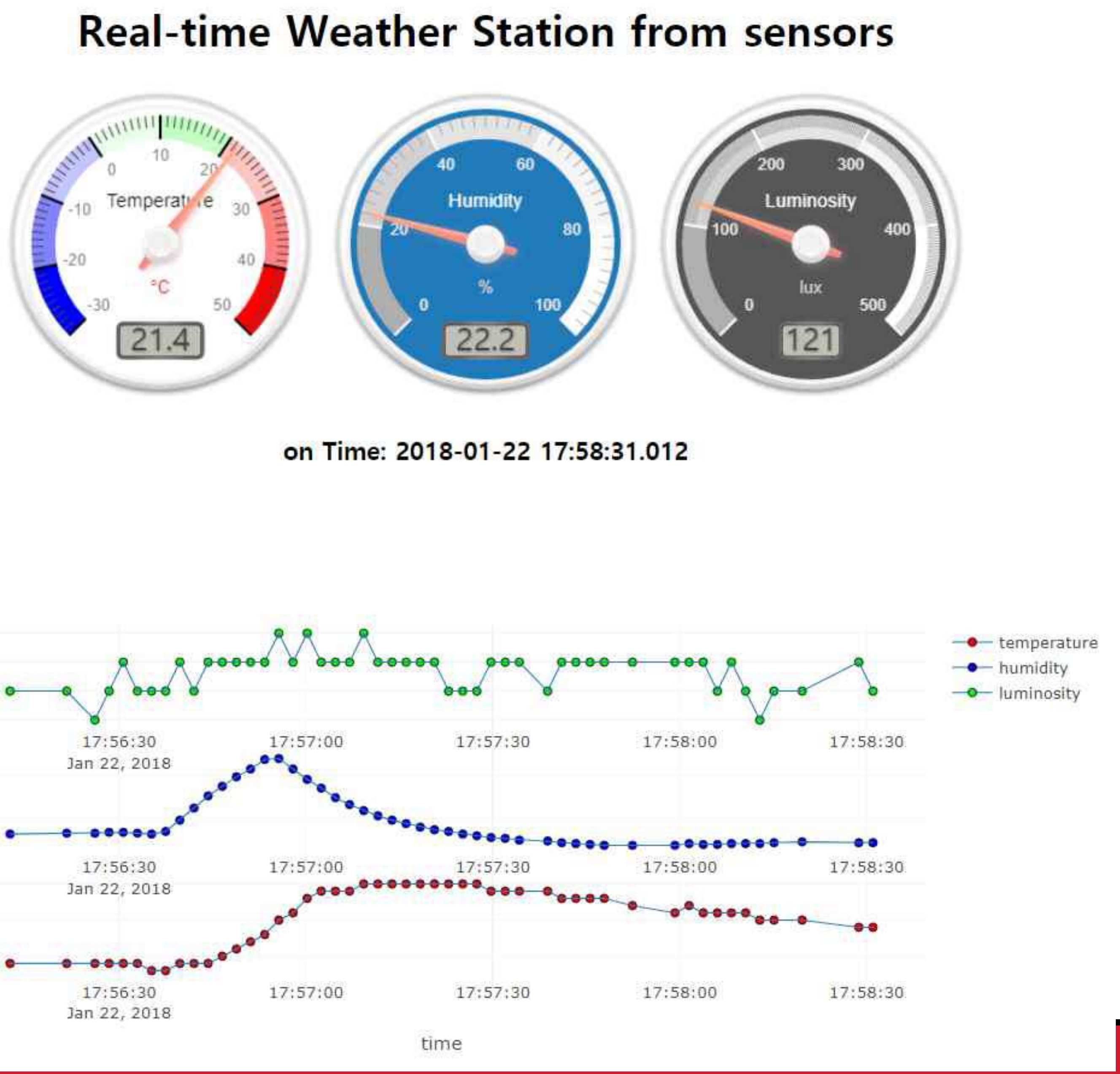
예제로 쉽게 배우는

아두이노 개정판

장성용 · 김진환 지음

생능출판

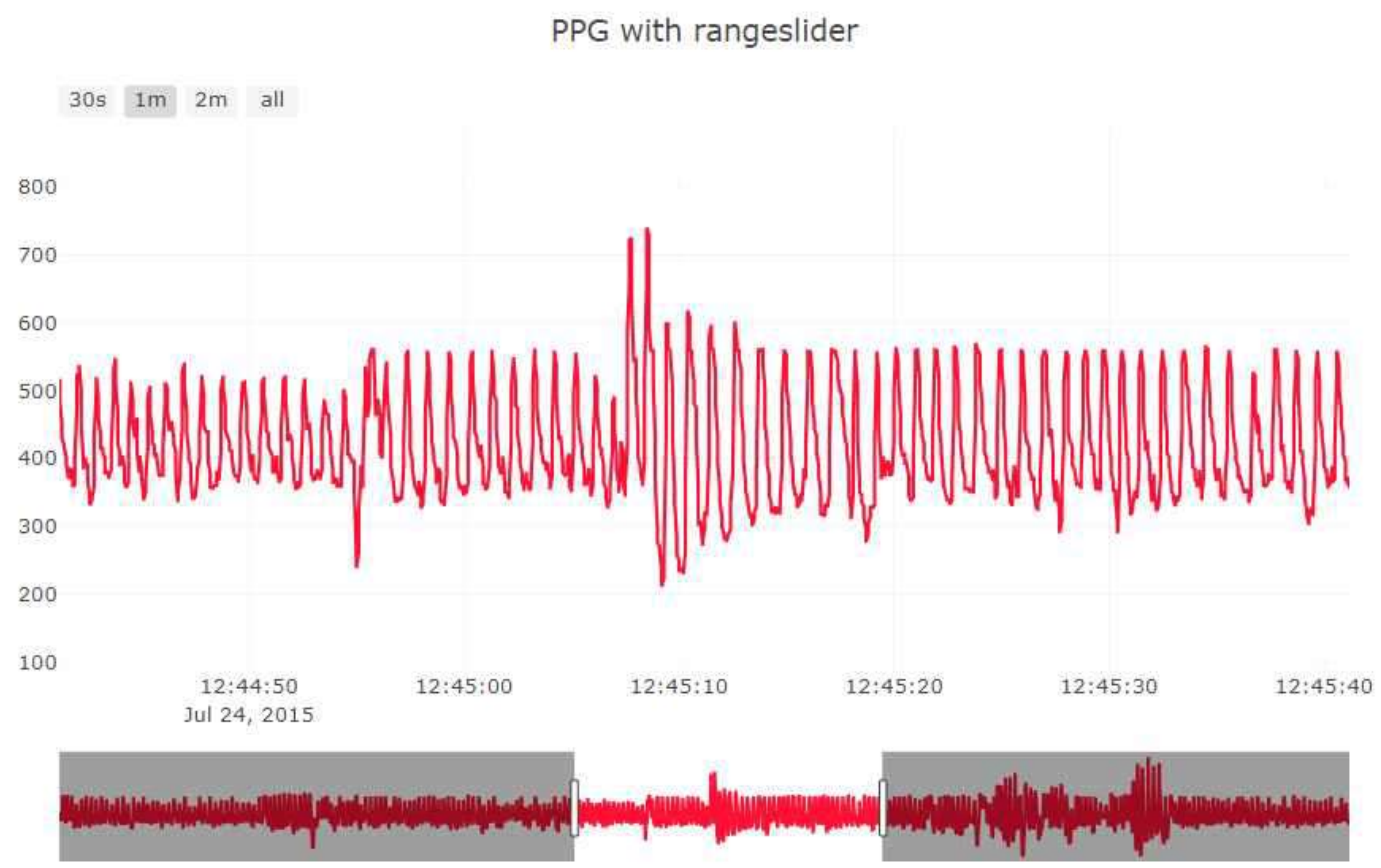
Target of this class



Real-time Weather Station from sensors

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

Another target of this class


PPG with rangeslider
30s
1m
2m
all
12:44:50
Jul 24, 2015
12:45:00
12:45:10
12:45:20
12:45:30
12:45:40