



Arduino-IoT

[wk04]

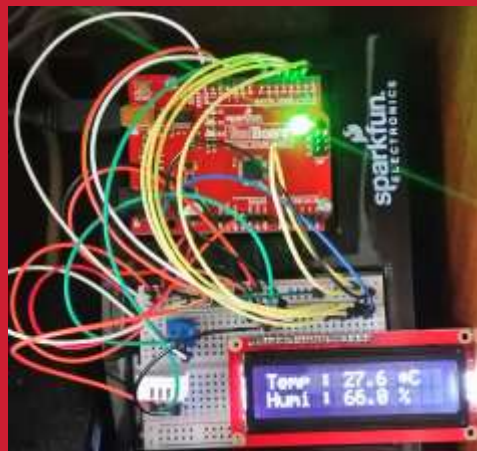
Arduino Sensors

Visualization of Signals using Arduino, Node.js & storing signals in MongoDB & mining data using Python

Drone-IoT-Comsi, INJE University

2nd semester, 2023

Email : chaos21c@gmail.com





My ID

ID를 확인하고 github에 repo 만들기

ID	성명
AA01	강동하
AA02	고서진
AA03	김민재
AA04	김예원
AA05	김주호
AA06	김창욱
AA07	김현서
AA08	박종혁
AA09	서명진
AA10	유동기
AA11	
AA12	이근보
AA13	정호기

위의 id를 이용해서 github에 repo를 만드시오.

Option: 아두이노응용 실습 과제 - AAnn

Public, README.md **check**



[Review]

◆ [wk03]

➤ aann-rpt03

➤ aann-rpt04

◆ [Target of this week]

- Complete your works
- Save your outcomes and 1 figure

Upload folder : aann-rpt03

■ 제출할 파일들

- ① **AAnn_Express.png**
- ② **AAnn_Express_2server.png**
- ③ **app.js**
- ④ **app2.js**

◆ [Target of this week]

- Complete your works
- Save your outcomes and upload 3 figures in github

Upload folder : aann-rpt04

- 제출할 파일들

- ① **AAnn_multi_Monitoring.png**
- ② **AAnn_multi_Signals.png**
- ③ **All *.ino**

Purpose of AA

주요 수업 목표는 다음과 같다.

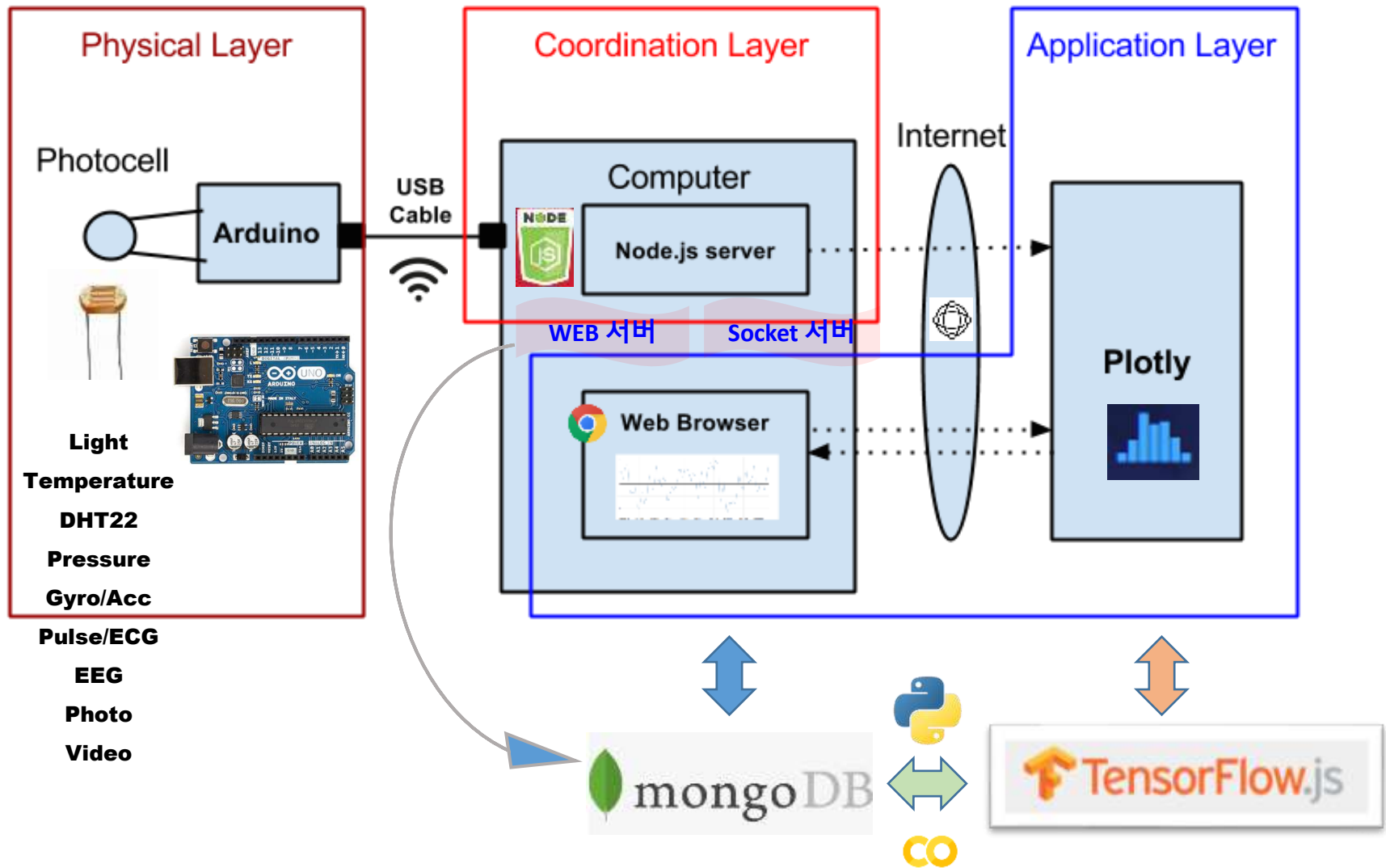
1. Node.js를 이용한 아두이노 센서 신호 처리
2. Plotly.js를 이용한 아두이노 센서 신호 시각화
3. MongoDB에 아두이노 센서 데이터 저장 및 처리



4. 저장된 IoT 데이터의 마이닝 (파이썬 코딩)



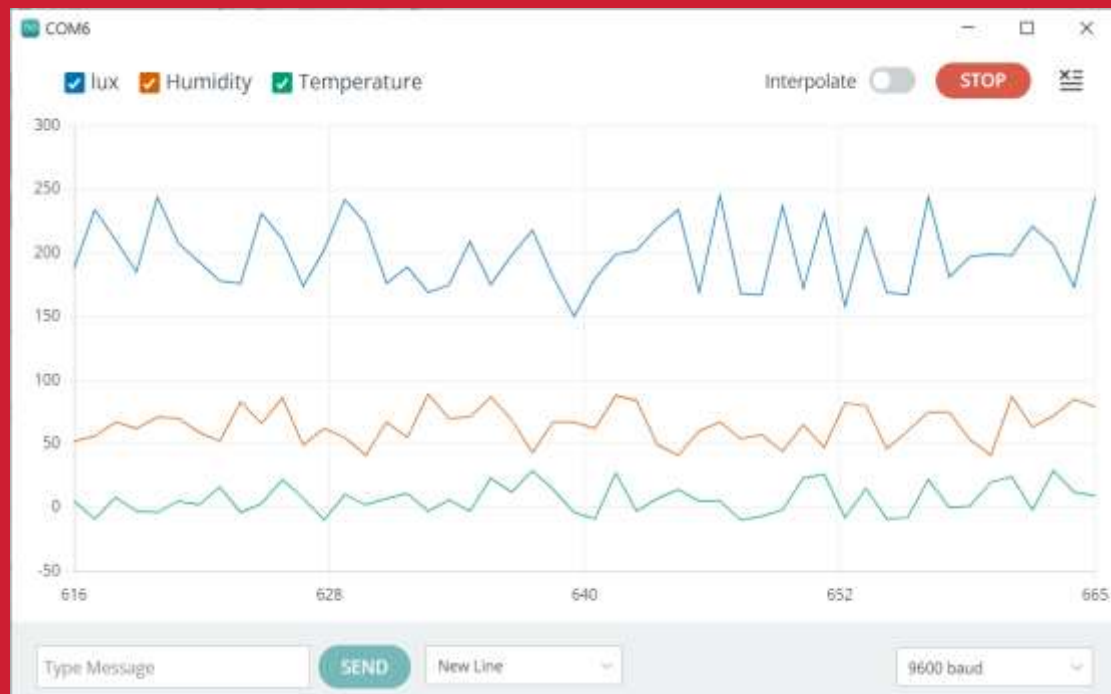
Layout [H S C]



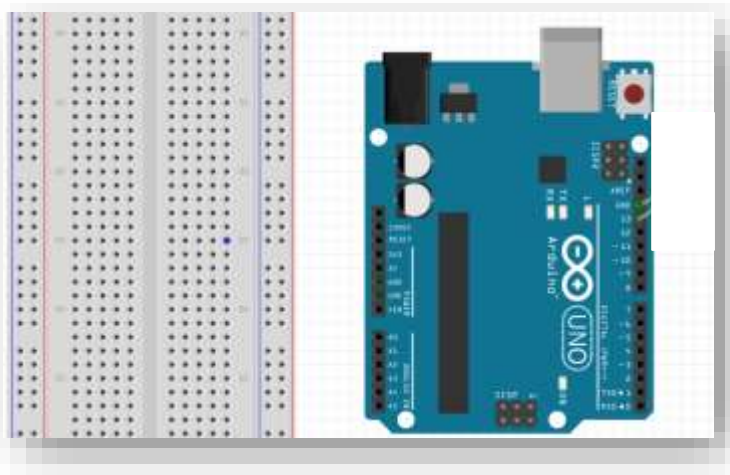


[DIY] Multi-signals

다중신호 시뮬레이션 및 모니터링



DIY - 스케치



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

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

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

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

▶ 스케치 구성

1. 3 개의 신호를 담은 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

```
1 /*
2   Multi Signals
3   Simulation of multiple random signals
4 */
5 // signals
6 int humi=0;
7 int temp=0;
8 int lux=0;
9
```

```
10 // the setup routine runs once when you press reset:
11 void setup() {
12   // Initialize serial communication at 9600 bits per second:
13   Serial.begin(9600);
14 }
15
16 // the loop routine runs over and over again forever:
17 void loop() {
18   // Multi signals
19   humi = random(40,90);
20   temp = random(-10, 30);
21   lux = random(150,250);
22   Serial.print("AA00, Ambient lux: ");
23   Serial.print(lux);
24   Serial.print(" , Humidity: ");
25   Serial.print(humi);
26   Serial.print(" , Temperature: ");
27   Serial.println(temp);
28   delay(500);      // delay in between reads for stability
29 }
```

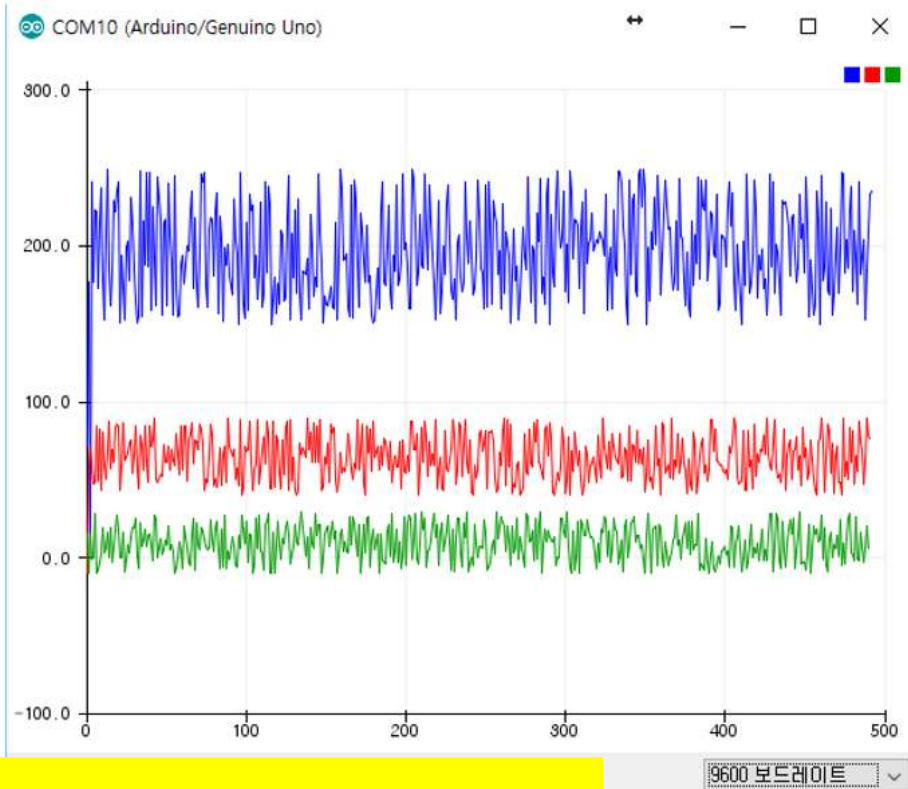


DIY - result

DIY 결과

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

```
COM10 (Arduino/Genuino Uno)
| 전송
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
< >
[ ] 자동 스크롤 line ending 없음 9600 보드레이트
```



Save as

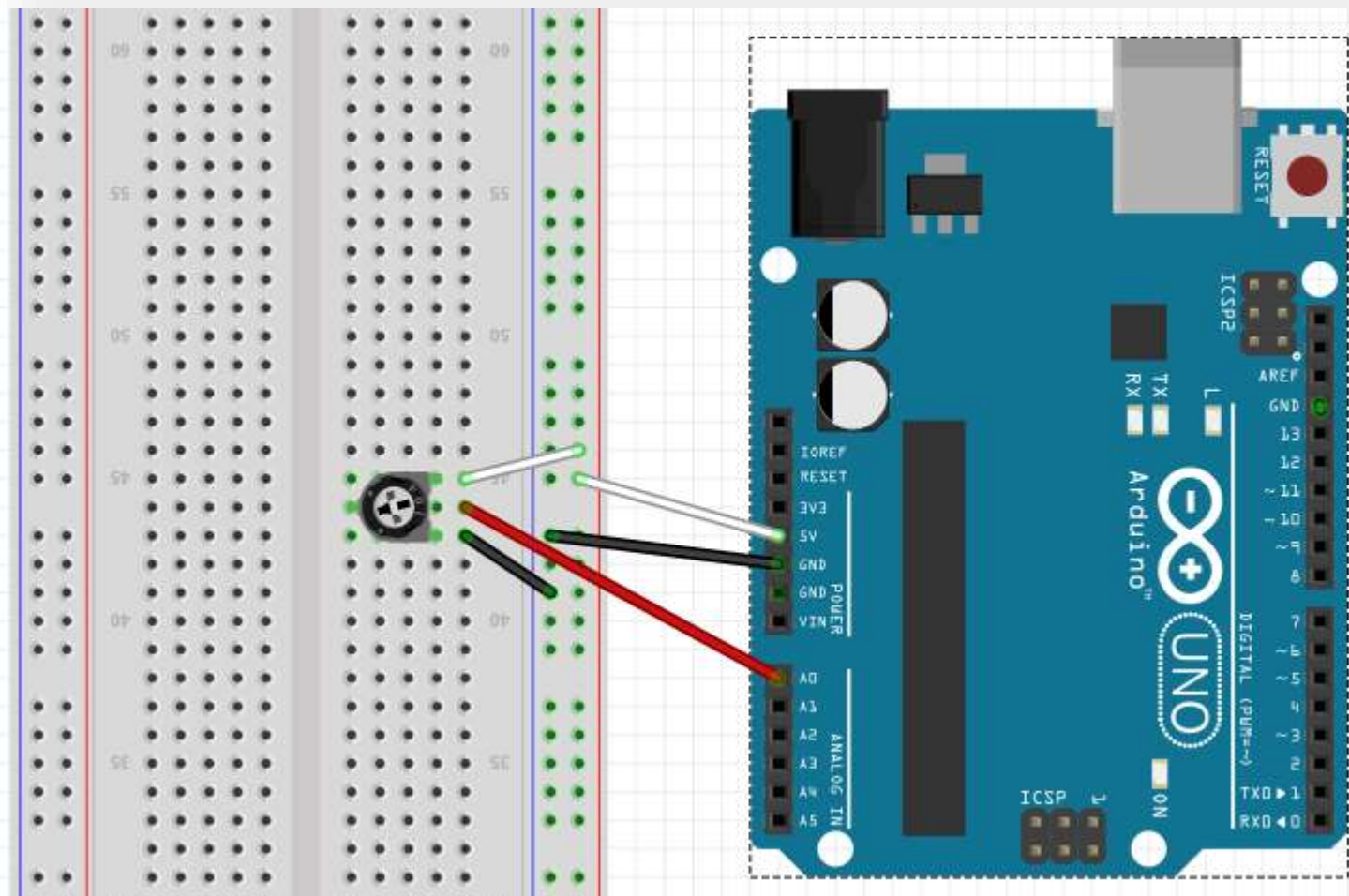
AAnn_multi_Signals.png



Analog Signal

A2.5.1 AnalogReadSerial (circuit)

Standard potentiometer (가변 저항기)



A2.5.2 AnalogReadSerial (code)

▶ 스케치 구성 (코드 4-1)

1. setup()에서 직렬 통신 속도를 9600 bps 로 설정하고 컴퓨터와 연결한다.
2. loop()에서 **analogRead()** 함수로 A0 핀에서 측정되는 값을 읽어 들인다.
3. 직렬 통신으로 A0 측정값을 한 줄로 0.5 초 마다 컴퓨터로 전송한다.

▶ 아두이노 코드 : sketch06_analog_read.ino

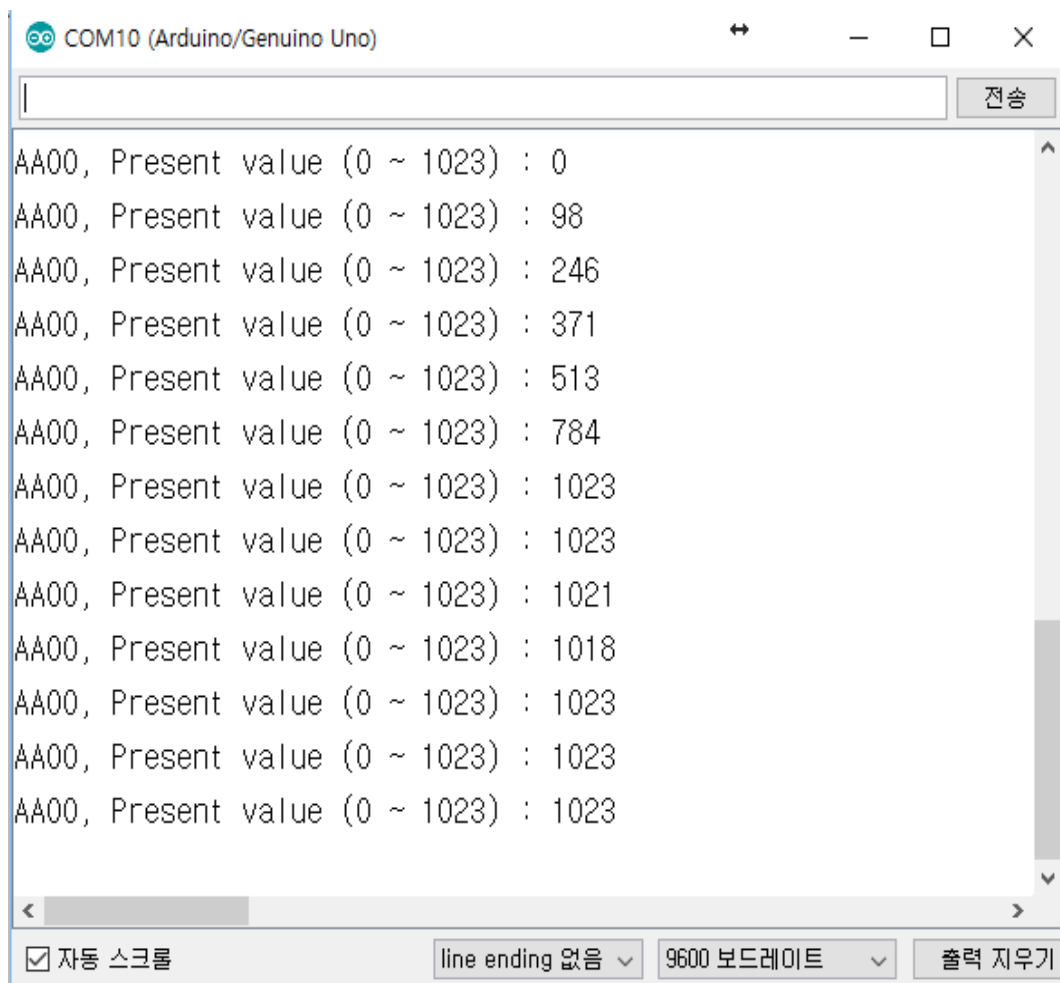
```
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  Serial.print("AA00, Present value (0 ~ 1023) : ");
  Serial.println(sensorValue);
  delay(500);    // 2 Hz sampling
}
```



A2.5.3 ReadAnalogValue

Serial monitor : $0 < \text{value} < 1023$



아날로그 값을 저항 및 전압으로 변환

▶ 저항 또는 전압 환산

$$1. \text{저항} = 10.0 * A0 / 1023 \text{ (k}\Omega\text{)}$$

$$2. \text{전압} = 5.0 * A0 / 1023 \text{ (V)}$$

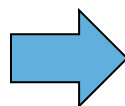
A0: 아날로그 핀 A0에서의 측정값 (0 ~ 1023)

A2.5.5 Analog value to Resistance

Serial monitor : Resistance ($0 < R < 10 \text{ k}\Omega$)

```
COM10 (Arduino/Genuino Uno)
전송

AA00, Present value (0 ~ 1023) : 0
AA00, Present value (0 ~ 1023) : 98
AA00, Present value (0 ~ 1023) : 246
AA00, Present value (0 ~ 1023) : 371
AA00, Present value (0 ~ 1023) : 513
AA00, Present value (0 ~ 1023) : 784
AA00, Present value (0 ~ 1023) : 1023
AA00, Present value (0 ~ 1023) : 1023
AA00, Present value (0 ~ 1023) : 1021
AA00, Present value (0 ~ 1023) : 1018
AA00, Present value (0 ~ 1023) : 1023
AA00, Present value (0 ~ 1023) : 1023
AA00, Present value (0 ~ 1023) : 1023
```



```
COM10 (Arduino/Genuino Uno)
전송

AA00, Present R (0 ~ 10.0) : 0.00
AA00, Present R (0 ~ 10.0) : 0.12
AA00, Present R (0 ~ 10.0) : 2.68
AA00, Present R (0 ~ 10.0) : 3.45
AA00, Present R (0 ~ 10.0) : 4.15
AA00, Present R (0 ~ 10.0) : 5.34
AA00, Present R (0 ~ 10.0) : 6.68
AA00, Present R (0 ~ 10.0) : 7.50
AA00, Present R (0 ~ 10.0) : 8.43
AA00, Present R (0 ~ 10.0) : 10.00
AA00, Present R (0 ~ 10.0) : 9.98
AA00, Present R (0 ~ 10.0) : 9.96
AA00, Present R (0 ~ 10.0) : 10.00
```

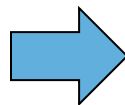
```
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.print("AA00, Present R (0 ~ 10.0) : ");
  float resistance = sensorValue*(10.0/1023.0); // kΩ
  Serial.println(resistance);
  delay(500);    // 2 Hz sampling
}
```

A2.5.6 Analog value to Voltage

Serial monitor : Voltage ($0 < V < 5 \text{ V}$)

```
COM10 (Arduino/Genuino Uno)
전송

AA00, Present value (0 ~ 1023) : 0
AA00, Present value (0 ~ 1023) : 98
AA00, Present value (0 ~ 1023) : 246
AA00, Present value (0 ~ 1023) : 371
AA00, Present value (0 ~ 1023) : 513
AA00, Present value (0 ~ 1023) : 784
AA00, Present value (0 ~ 1023) : 1023
AA00, Present value (0 ~ 1023) : 1023
AA00, Present value (0 ~ 1023) : 1021
AA00, Present value (0 ~ 1023) : 1018
AA00, Present value (0 ~ 1023) : 1023
AA00, Present value (0 ~ 1023) : 1023
AA00, Present value (0 ~ 1023) : 1023
```



```
COM10 (Arduino/Genuino Uno)
전송

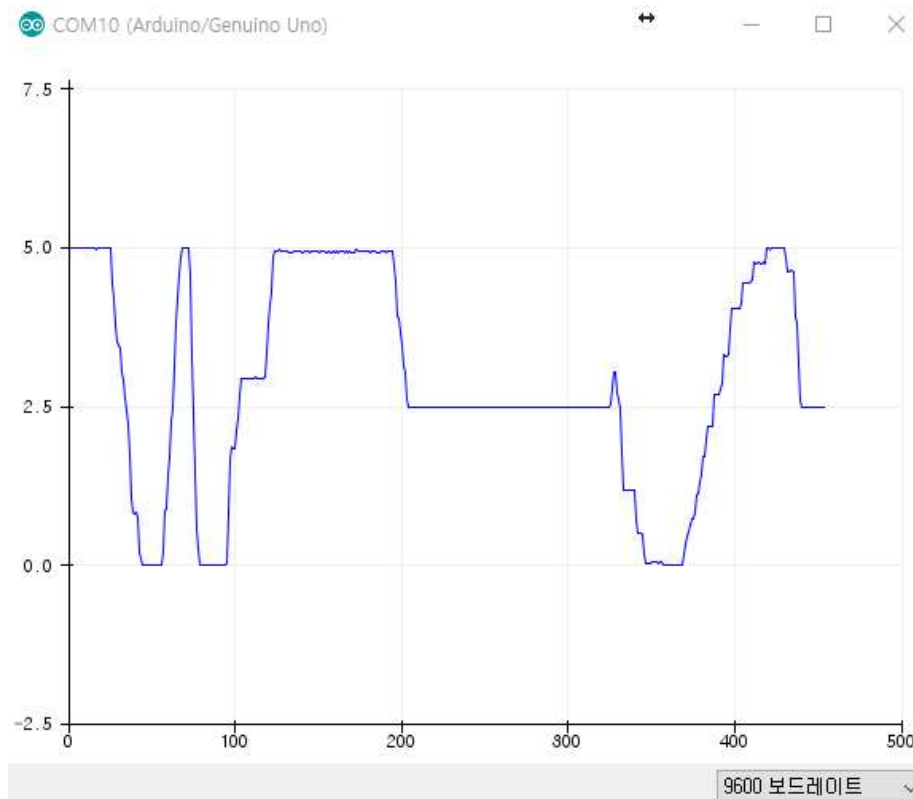
AA00, Present V (0 ~ 5.0) : 0.00
AA00, Present V (0 ~ 5.0) : 0.25
AA00, Present V (0 ~ 5.0) : 0.75
AA00, Present V (0 ~ 5.0) : 1.73
AA00, Present V (0 ~ 5.0) : 2.26
AA00, Present V (0 ~ 5.0) : 2.61
AA00, Present V (0 ~ 5.0) : 3.37
AA00, Present V (0 ~ 5.0) : 4.20
AA00, Present V (0 ~ 5.0) : 4.81
AA00, Present V (0 ~ 5.0) : 5.00
AA00, Present V (0 ~ 5.0) : 4.99
AA00, Present V (0 ~ 5.0) : 5.00
AA00, Present V (0 ~ 5.0) : 5.00
```

```
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.print("AA00, Present V (0 ~ 5.0) : ");
  float voltage= sensorValue*(5.0/1023.0); // V
  Serial.println(voltage);
  delay(500);    // 2 Hz sampling
}
```

A2.5.7 ReadAnalogVoltage

Result

```
COM4
\\A00, Present voltage (0.0 ~ 5.0) : 5.00
\\A00, Present voltage (0.0 ~ 5.0) : 3.68
\\A00, Present voltage (0.0 ~ 5.0) : 2.42
\\A00, Present voltage (0.0 ~ 5.0) : 1.37
\\A00, Present voltage (0.0 ~ 5.0) : 0.00
\\A00, Present voltage (0.0 ~ 5.0) : 0.00
\\A00, Present voltage (0.0 ~ 5.0) : 0.88
\\A00, Present voltage (0.0 ~ 5.0) : 1.47
\\A00, Present voltage (0.0 ~ 5.0) : 2.11
\\A00, Present voltage (0.0 ~ 5.0) : 2.79
\\A00, Present voltage (0.0 ~ 5.0) : 3.38
\\A00, Present voltage (0.0 ~ 5.0) : 3.99
\\A00, Present voltage (0.0 ~ 5.0) : 4.91
\\A00, Present voltage (0.0 ~ 5.0) : 5.00
\\A00, Present voltage (0.0 ~ 5.0) : 5.00
\\A00, Present voltage (0.0 ~ 5.0) : 4.68
\\A00, Present voltage (0.0 ~ 5.0) : 3.88
\\A00, Present voltage (0.0 ~ 5.0) : 3.35
```



Save as

AAnn_AnalogVoltage.png

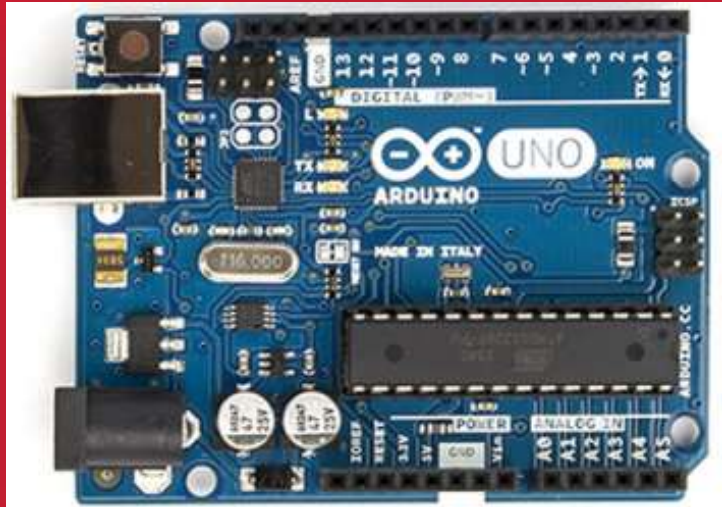
A2.5.8 ReadAnalogVoltage using f_map()

Hint code : f_map() instead of map()

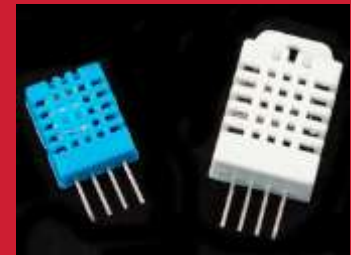
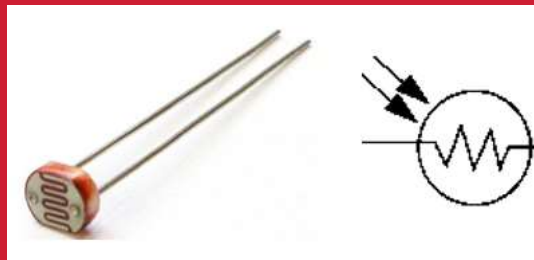
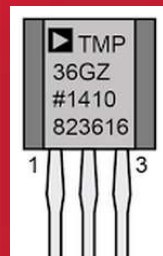
```

1 // the setup routine runs once when you press reset:
2
3 void setup() {
4   // initialize serial communication at 9600 bits per second:
5   Serial.begin(9600);
6 }
7
8 // the loop routine runs over and over again forever:
9 void loop() {
10   // read the input on analog pin 0:
11   int sensorValue = analogRead(A0);
12   //float voltage = map(sensorValue, 0, 1023, 0.0, 5.0); // map 0~1023 to 0~5
13   // float voltage = sensorValue*(5.0/1023.0);
14   float voltage = f_map(sensorValue, 0, 1023, 0.0, 5.0); // map 0~1023 to 0~5
15   // print out the value you read:
16   Serial.print("A00, Present voltage (0.0 ~ 5.0) : ");
17   Serial.println(voltage);
18   delay(500);      // delay in between reads for stability
19 }
20
21 float f_map(long x, long in_min, long in_max, float out_min, float out_max)
22 {
23   return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
24 }

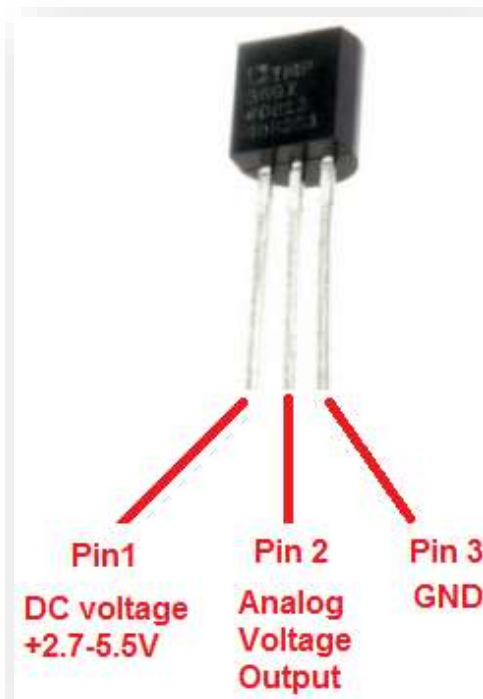
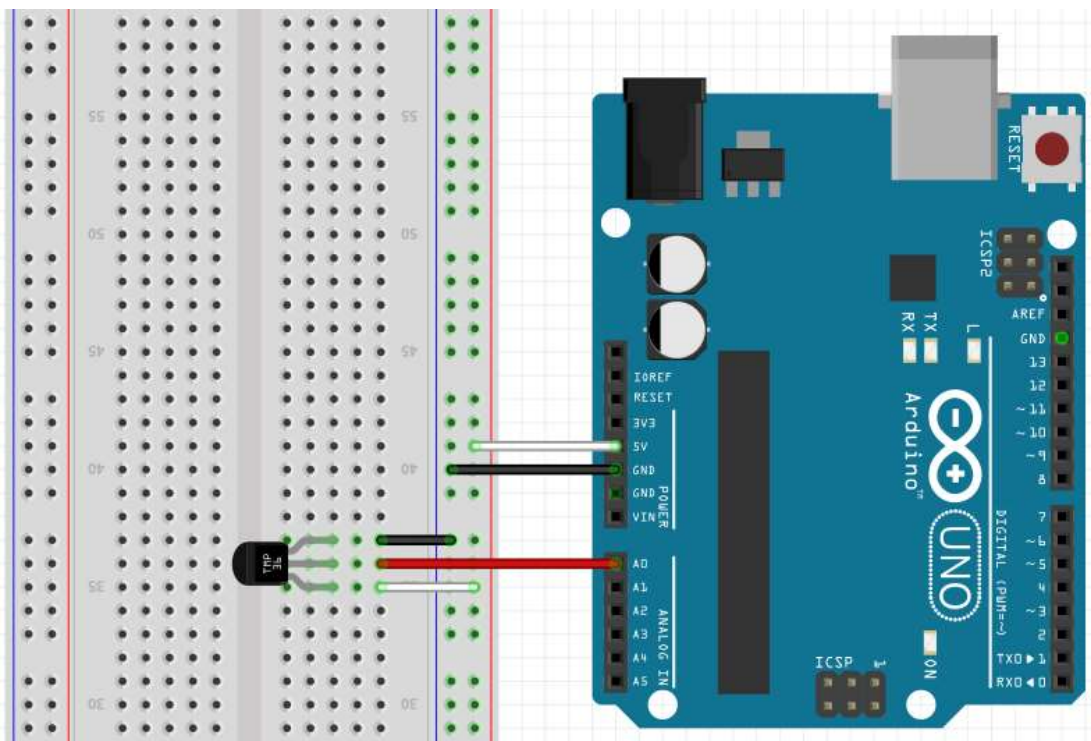
```



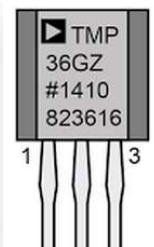
Arduino Sensors



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

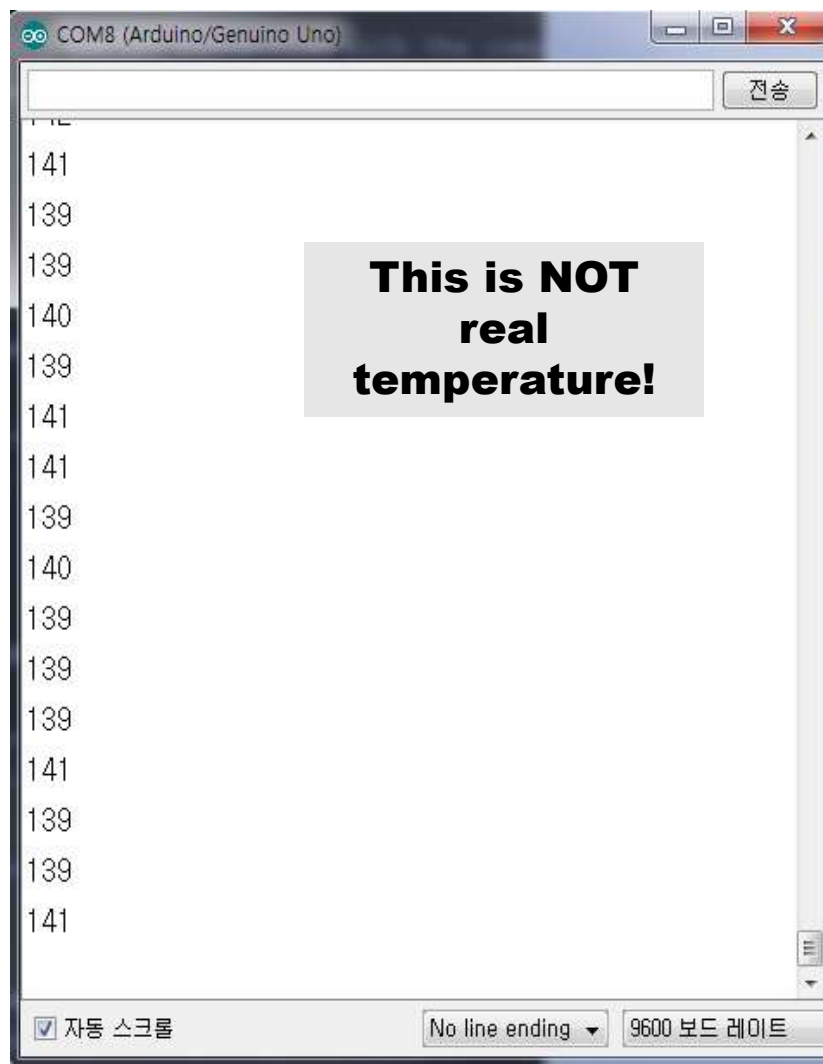


A3.1.2 Temperature sensor [TMP36]

Simple code

```
TMP36 $
1 //
2 //  AA00, 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)



A3.1.3 Temperature sensor [TMP36]

Sensor property

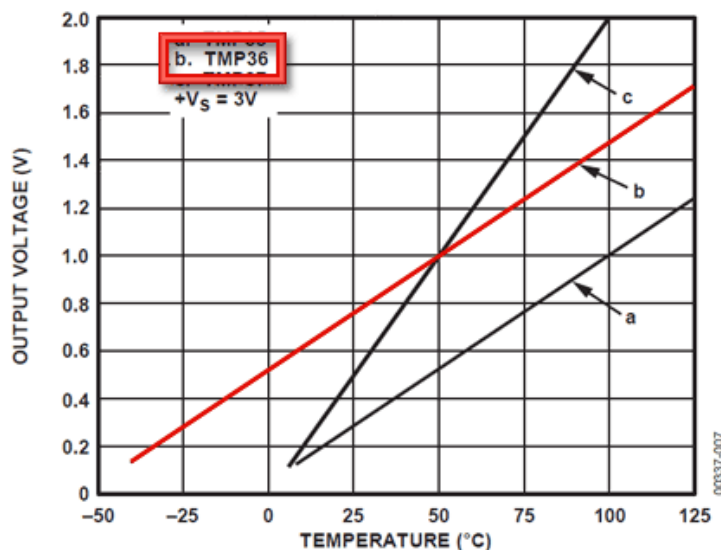


Figure 6. Output Voltage vs. Temperature

Temperature conversion

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

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

$$(0 \leq \text{value} \leq 1023)$$



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

https://github.com/Redwoods/Arduino/blob/master/ar-iot/py-ml/tmp36_LR.ipynb



A3.1.4 Temperature sensor [TMP36]

Working code

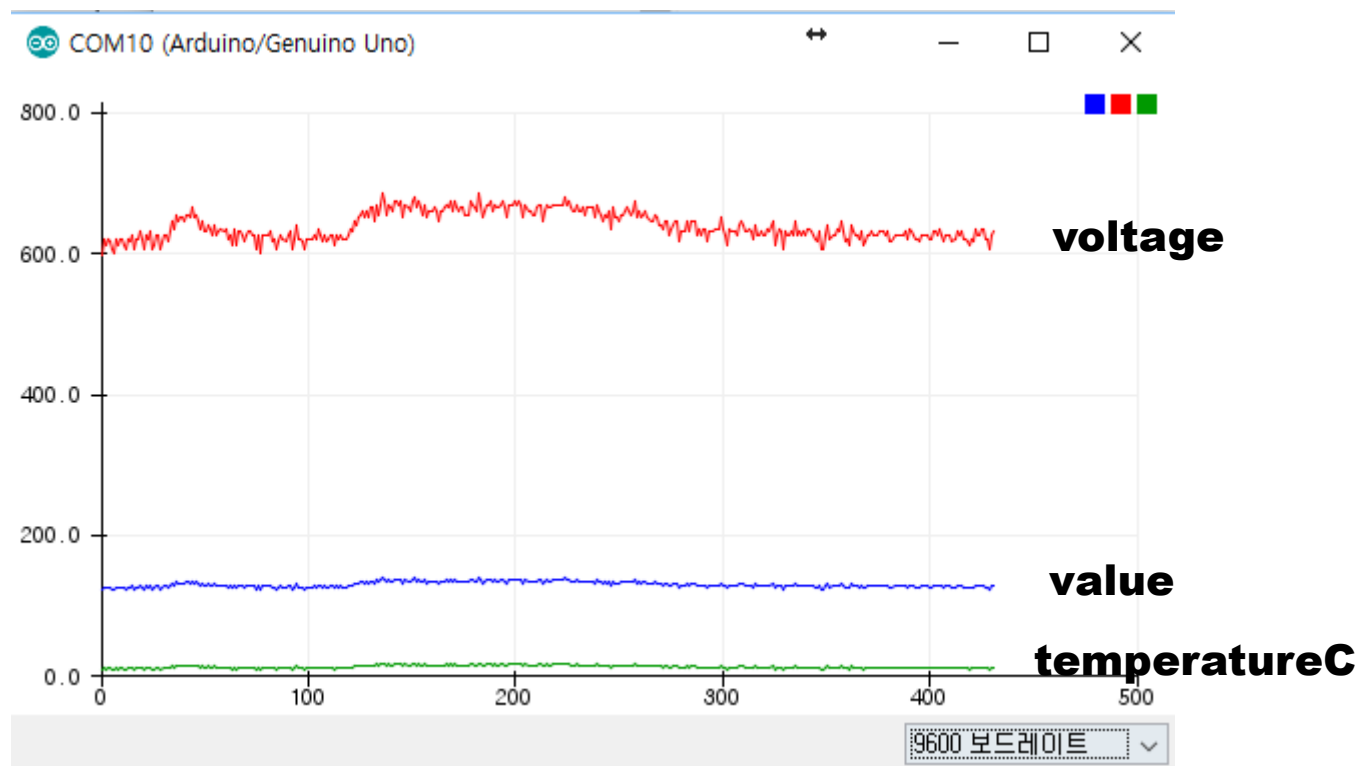
```
TMP36
10 }
11
12 void loop() {
13     //getting the voltage reading from the temperature sensor
14     int value = analogRead(TEMP_INPUT);
15     Serial.print("AA00, value = ");
16     Serial.print(value);
17     Serial.print(" : ");
18
19     // converting that reading to voltage
20     float voltage = value * 5.0 * 1000; // in mV
21     voltage /= 1023.0;
22
23     // print out the voltage
24     Serial.print(voltage);
25     Serial.print(" mV, ");
26
27     // now print out the temperature
28     float temperatureC = (voltage - 500) / 10 ;
29     Serial.print(temperatureC);
30     Serial.println(" degrees C");
31
32     delay(1000);
33 }
```

Serial output (°C)

```
COM4
AA00, value = 131 : 640.27 mV, 14.03 degrees C
AA00, value = 130 : 635.39 mV, 13.54 degrees C
AA00, value = 132 : 645.16 mV, 14.52 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 129 : 630.50 mV, 13.05 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 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
```

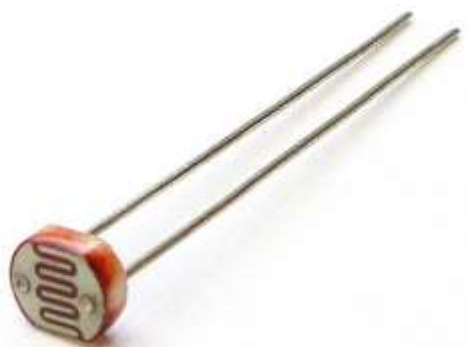


A3.1.5 Temperature sensor [TMP36]

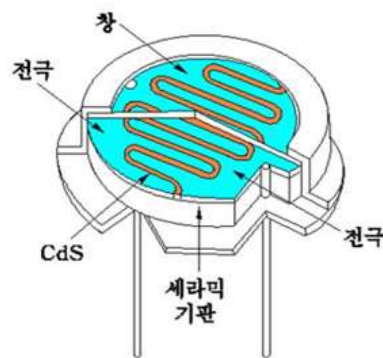


Save as
AAnn_TMP36.png

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

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.

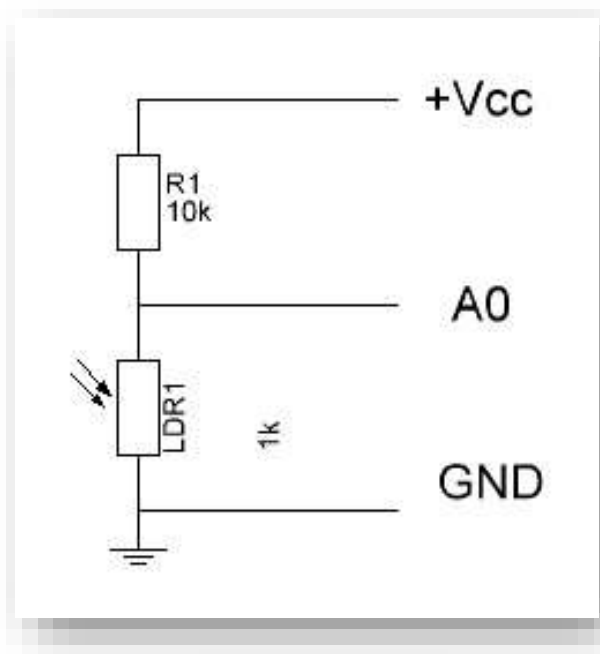
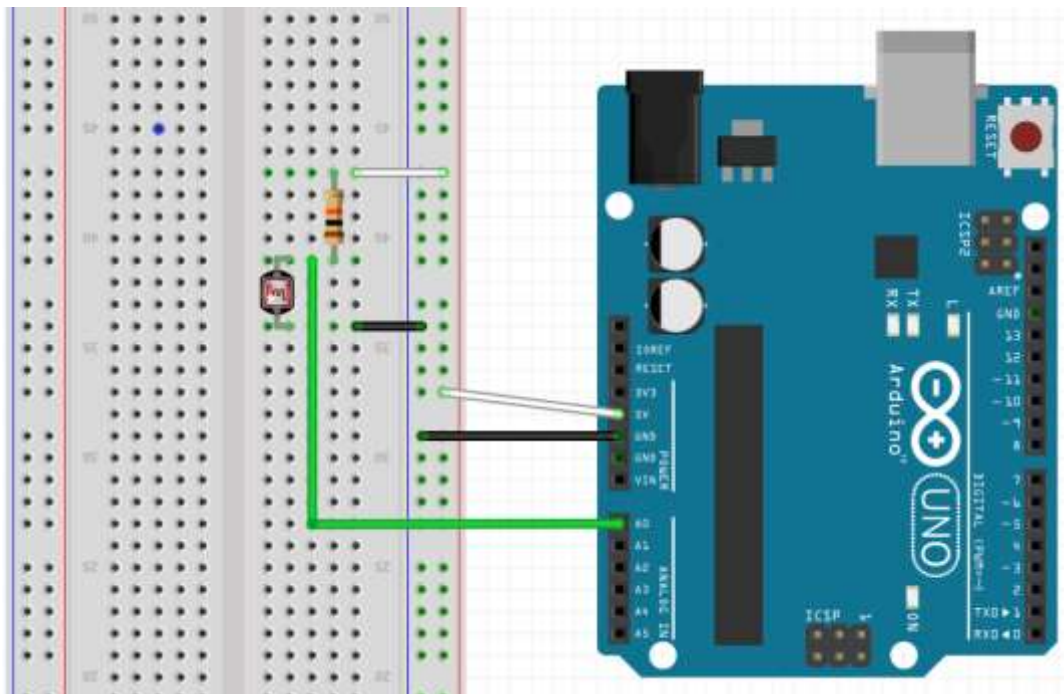
럭스(lux, 기호 **lx**)는 빛의 **조명도**를 나타내는 SI 단위이다. 럭스는 루멘에서 유도

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

럭스의 예 [\[편집\]](#)

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

CdS 센서 회로



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

광센서에서의 전압 강하 값을 **A0**로 측정



▶ 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.



CdS 센서 회로 – 측정 1.

CdS_start

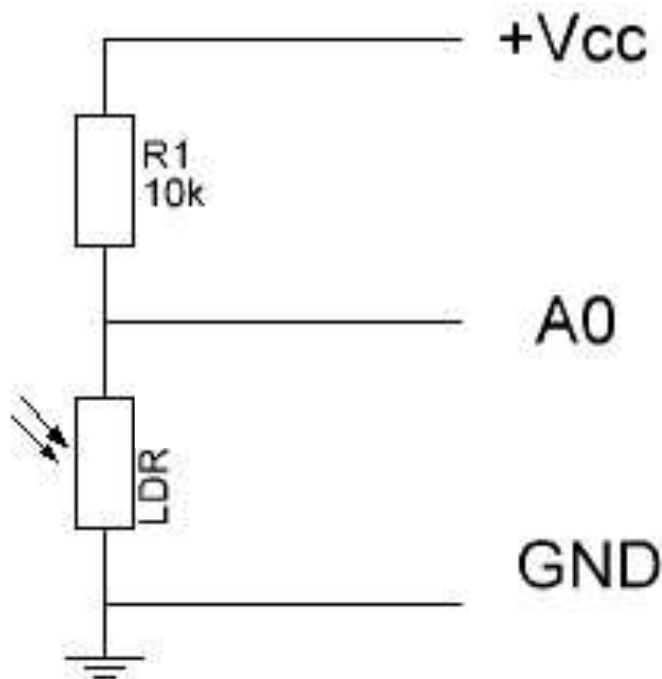
```
1 #define CDS_INPUT 0
2
3 void setup() {
4   Serial.begin(9600);
5 }
6
7 void loop() {
8
9   int value = analogRead(CDS_INPUT);
10  Serial.println(value);
11
12  delay(1000);
13 }
14
```

COM11 (Arduino/Genuino Uno)

672		어두울 때
672		
671		
669		
209		
205	밝을 때	
207		
207		
205		
207		
62		어두울 때
59		
53		

어두우면 측정 값이 커지고 밝을수록 값이 작아진다 ???

CdS 센서 회로 분석 (1/2)



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

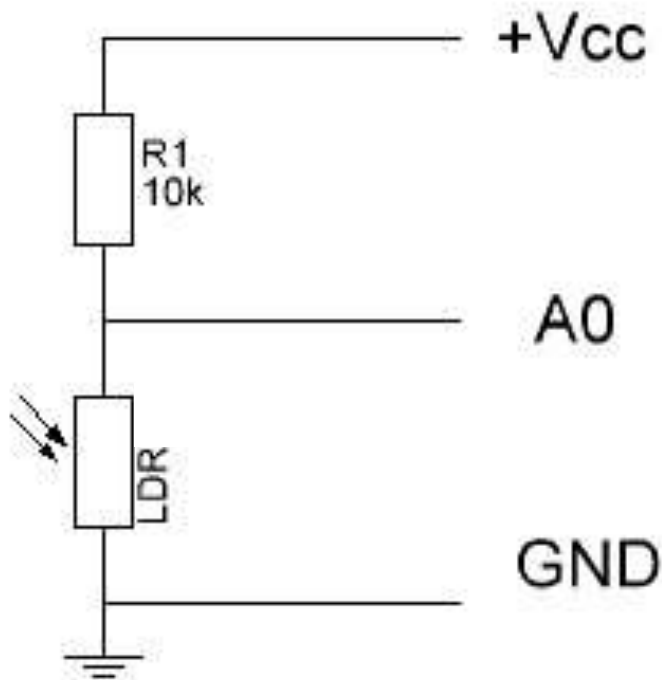
If you would use the LDR as the lower part of a voltage divider, then in darkness there would be a high voltage over the LDR, while in bright light, there would be a low voltage over that resistor.

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

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

A0에서 측정되는 LDR
양단의 전압 = V_{out}

CdS 센서 회로 분석 (2/2)



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

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

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

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

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

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

A0에서 측정되는 **LDR**
양단의 전압 = **V_{out}**

A3.2.5 Luminosity sensor [sketch-2]

▶ 스케치 구성

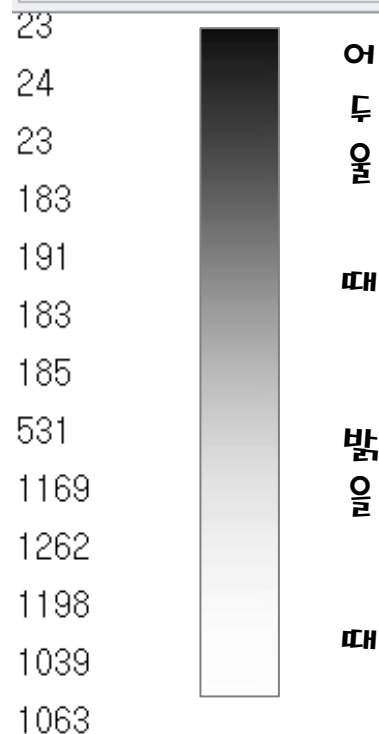
1. A0 핀을 CdS 조도 센서의 입력으로 설정한다.
2. setup()에서 직렬 통신 속도를 9600 bps 로 설정하고 컴퓨터와 연결한다.
3. loop()에서 **analogRead()** 함수로 A0 핀에서 측정되는 값을 읽어 들인다.
4. A0 측정값 (0 ~ 1023)을 전압 (0 ~ 5 V)으로 환산한다.
5. 전압을 조도로 환산한 후, A0 측정값, 환산 전압, 환산 조도 를 한 줄로 1 초 마다 컴퓨터로 전송한다.

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 Yout=RawADC0*5.0/1023; // 5/1023 (Vin = 5 V)
16   double lux=(2500/Yout-500)/10;
17   // lux = 500 / Rldr, Yout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
18   return lux;
19 }
  
```

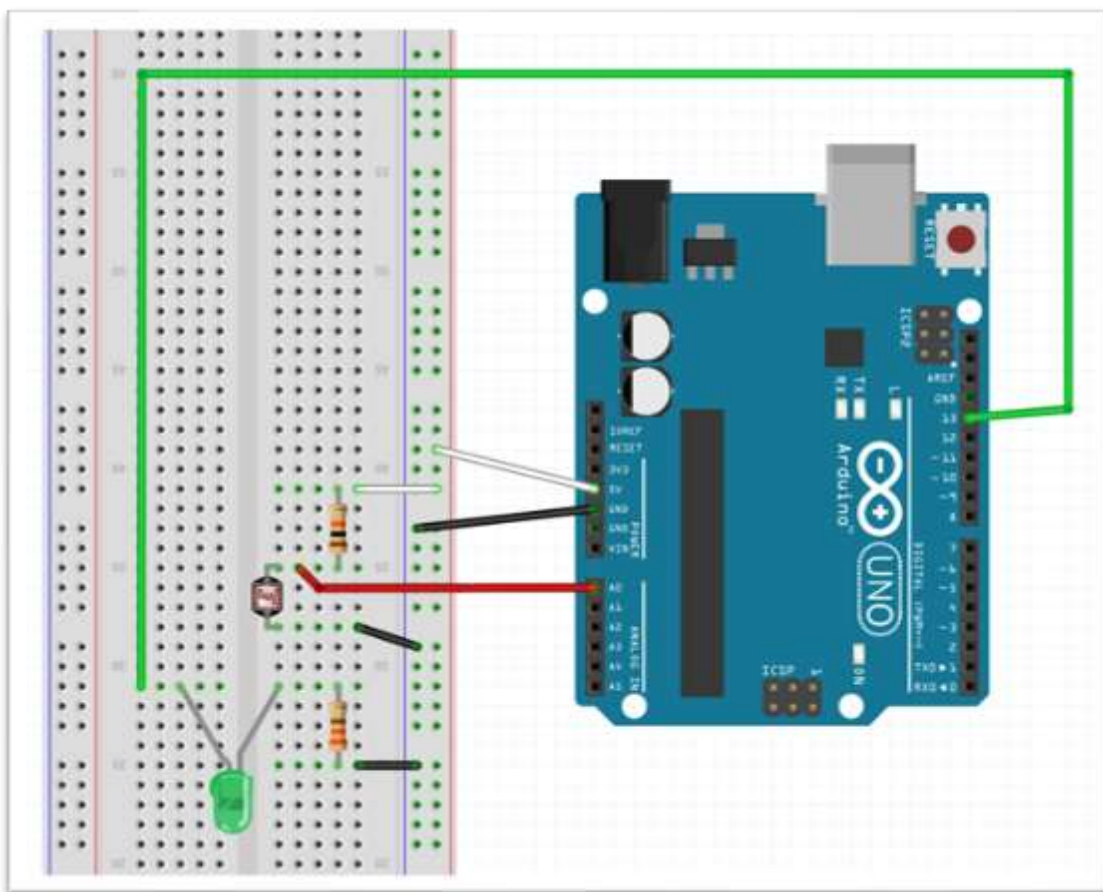
COM11 (Arduino/Genuino Uno)



밝을수록 측정 값이 커지고
어두울수록 값이 작아진다 !!!

DIY 조도 값에 따라 LED를 켜고 끄는 코드를 만드시오.

- 단색 LED의 anode를 D13번, cathode를 220 Ω (330 Ω) 저항에 연결 후 GND에 연결하시오.
- 조도 값이 문턱 값 이상이면 LED를 OFF, 그렇지 않으면 ON.





DIY Code

Write down your code here to complete the task that turns on LED when luminosity of ambient light becomes lower than a threshold.

조도 값이 문턱 값 이상이면 LED를 OFF, 그렇지 않으면 ON.

DIY Code

```

Cds_LED
1 // lux
2 #define CDS_INPUT 0
3 // LED pin
4 const int ledPin = 13;
5
6 int threshold = 70;
7
8 void setup() {
9   pinMode(ledPin, OUTPUT);
10  Serial.begin(9600);
11 }

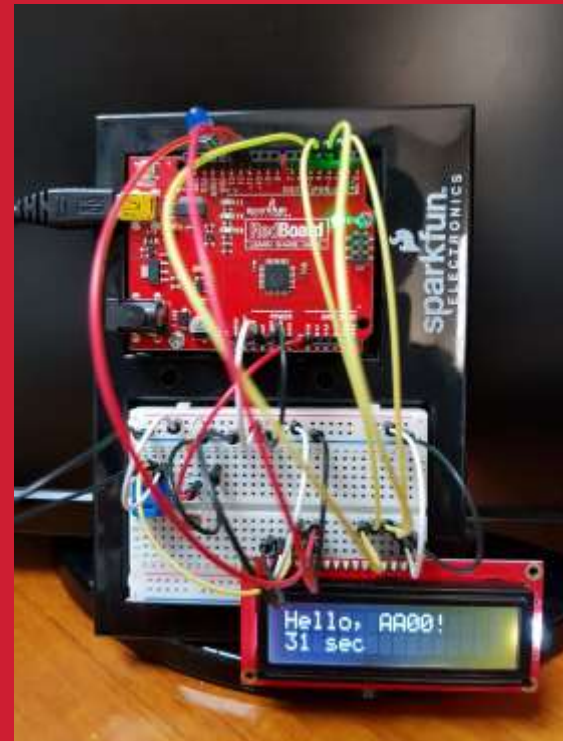
12
13 void loop() {
14   int value = analogRead(CDS_INPUT);
15   int lux = int(luminosity(value))
16   Serial.println(lux);
17
18   // If lux is lower than a threshold, LED is set ON.
19   if(lux >= threshold)
20     digitalWrite(ledPin, LOW);
21   else
22     digitalWrite(ledPin, HIGH);
23
24   delay(1000);
25 }
26 //Voltage to Lux
27 double luminosity (int RawADC0){
28   double Vout=RawADC0*5.0/1023; // 5/1023 (Vin = 5 V)
29   double lux=(2500/Vout-500)/10;
30   // lux = 500 / Rldr, Vout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
31   return lux;
32 }

```

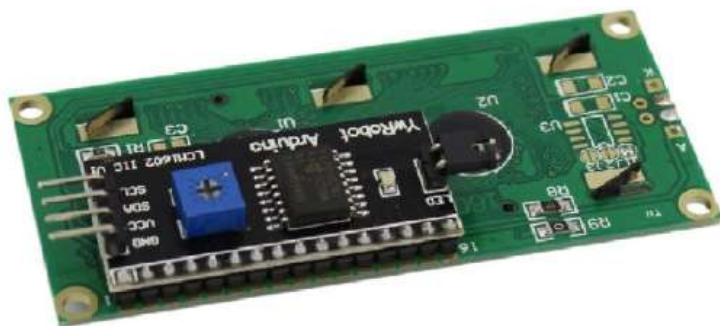
AAnn_CdS_LED.ino



Signal Monitoring via LCD



2. I²C를 이용한 LCD 출력



I²C(아이스퀘어드시, **Inter-Integrated Circuit**)는 필립스에서 개발한 직렬 버스이다. 마더보드, 임베디드 시스템, 휴대 전화 등에 저속의 주변 기기를 연결하기 위해 사용된다.

I²C는 풀업 저항이 연결된 직렬 데이터(**SDA**)와 직렬 클럭(**SCL**)이라는 두 개의 양 방향 오픈 컬렉터 라인을 사용한다. 최대 전압은 **+5 V**이며, 일반적으로 **+3.3 V** 시스템이 사용되지만 다른 전압도 가능하다.

<https://ko.wikipedia.org/wiki/I%C2%B2C>

<http://www.ifuturetech.org/product/16x2-lcd-i2c-lcd/>



I²C (Inter Integrated Circuit)

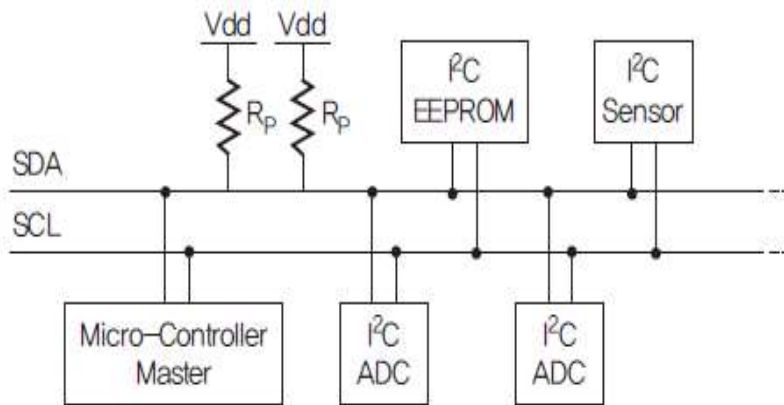


그림 3.2 I²C를 이용한 네트워크

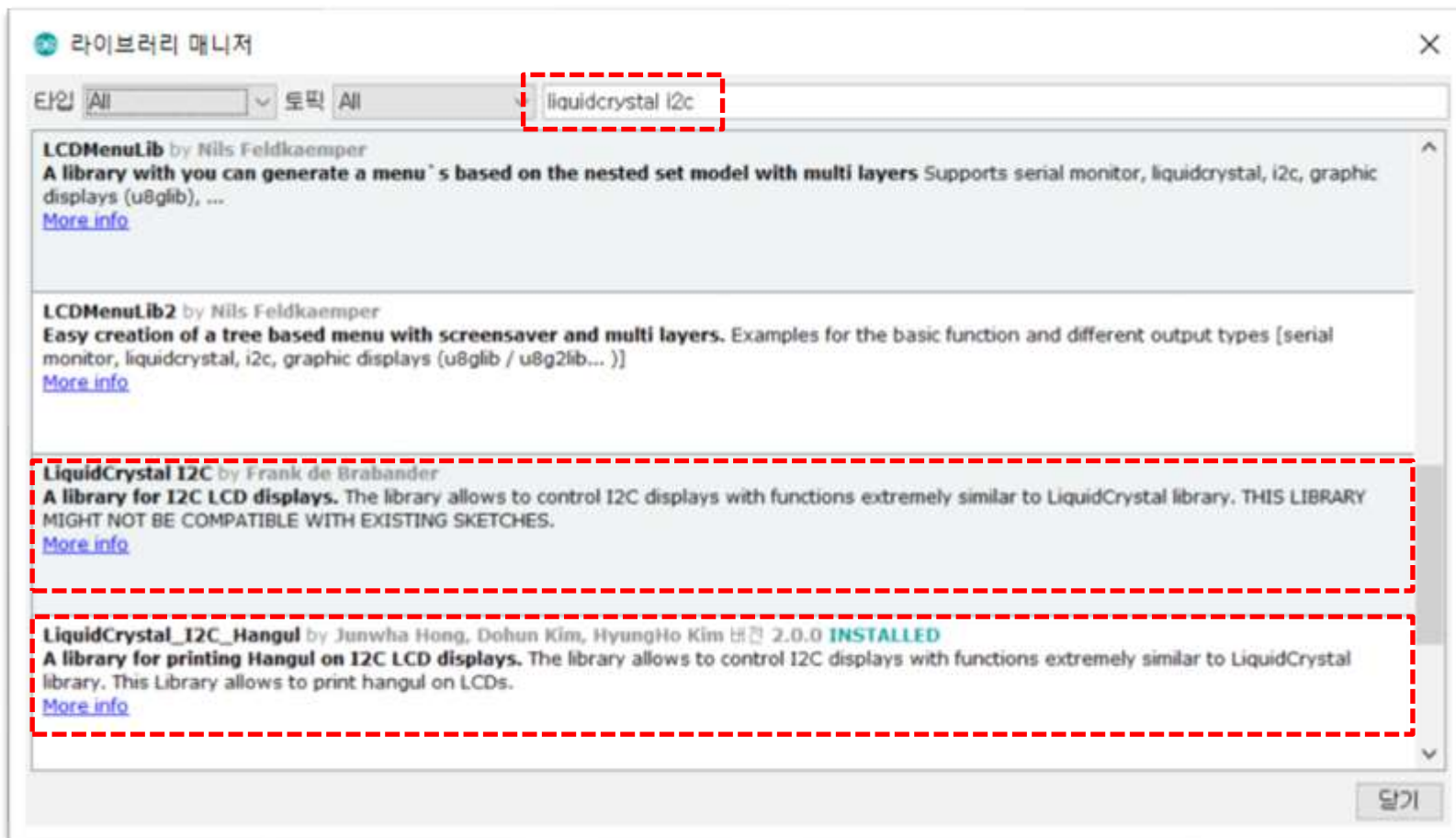
- ✓ Phillips사에서 개발된 규격이며 TWI라고도 함.
- ✓ SDA(Serial Data line), SCL(Serial Clock Line) 두 선으로 통신
- ✓ Master와 Slave로 구분되어 Master에서 통신을 주관
- ✓ 최대 112개의 노드를 연결 가능하고 최고 3.4Mbps의 속도

- ✓ LCD 모듈을 I²C 통신으로 제어하기 위해선 PCF8574 IC를 사용
- ✓ SDA, SCL 두 개의 입출력 핀만 필요

I²C를 이용한 LCD 출력 - 라이브러리 설치

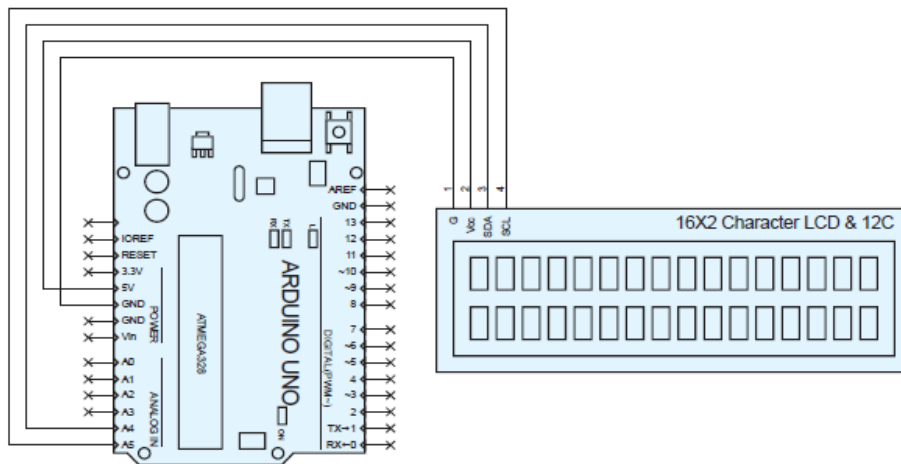
라이브러리 매니저를 이용하여 I²C LCD용 라이브러리(LiquidCrystal I2C)를 설치

스케치 > 라이브러리 포함하기 > 라이브러리 관리

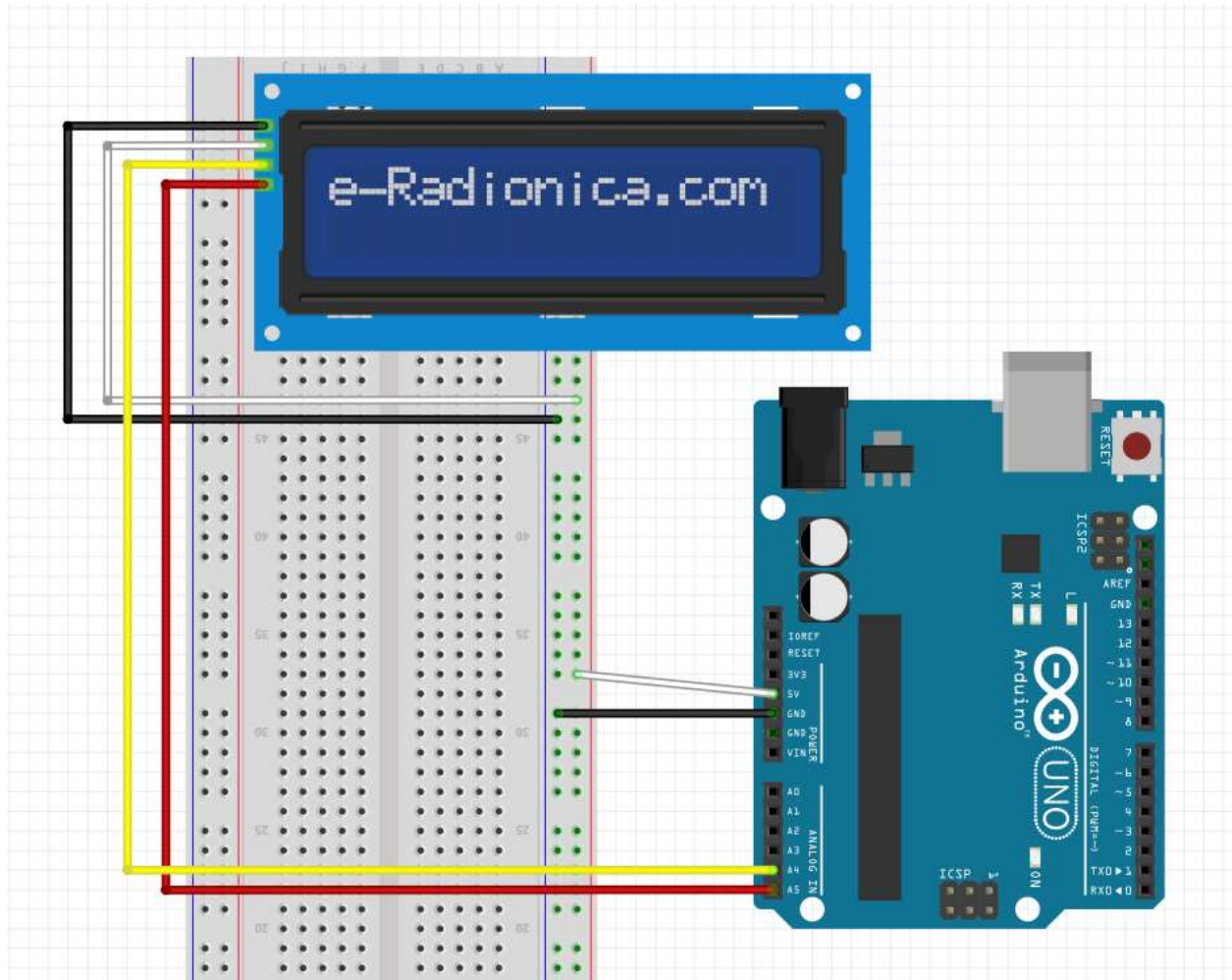


I²C를 이용한 LCD 출력 회로

- Hardware
1. I²C LCD 모듈과 Arduino는 전원핀 Vcc, GND와 I²C 통신핀 SDA, SCL이 연결되어야 한다.
 2. I²C LCD 모듈의 Vcc와 GND를 Arduino의 5V와 GND에 연결한다.
 3. SDA는 A4에, SCL은 A5에 연결한다.



I²C를 이용한 LCD 출력 회로



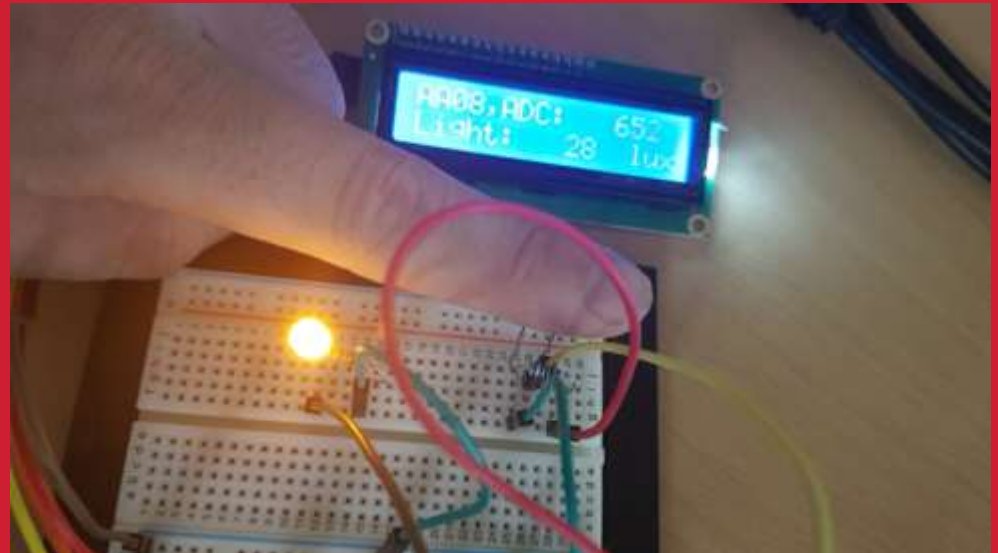
Commands

- LiquidCrystal_I2C(I2C 주소, 가로 글자수, 세로 글자수)
LCD 모듈이 연결된 I2C 주소와 LCD의 가로, 세로 글자수를 설정한다.
- lcd.init(); LCD 모듈을 설정한다.
- lcd.clear(): lcd란 이름의 LCD 모듈의 화면의 모든 표시를 지우고 커서를 왼쪽 위로 옮긴다.
- lcd.home(): lcd란 이름의 LCD 모듈의 커서를 왼쪽 위로 옮긴다.
- lcd.setCursor(행, 열): lcd란 이름의 LCD 모듈의 커서를 원하는 위치로 이동시킨다.
- lcd.print(데이터): lcd란 이름의 LCD 모듈에 데이터를 출력한다.
- lcd.noBacklight(): lcd란 이름의 LCD 모듈의 백라이트를 소등한다.
- lcd.backlight(): lcd란 이름의 LCD 모듈의 백라이트를 점등한다.

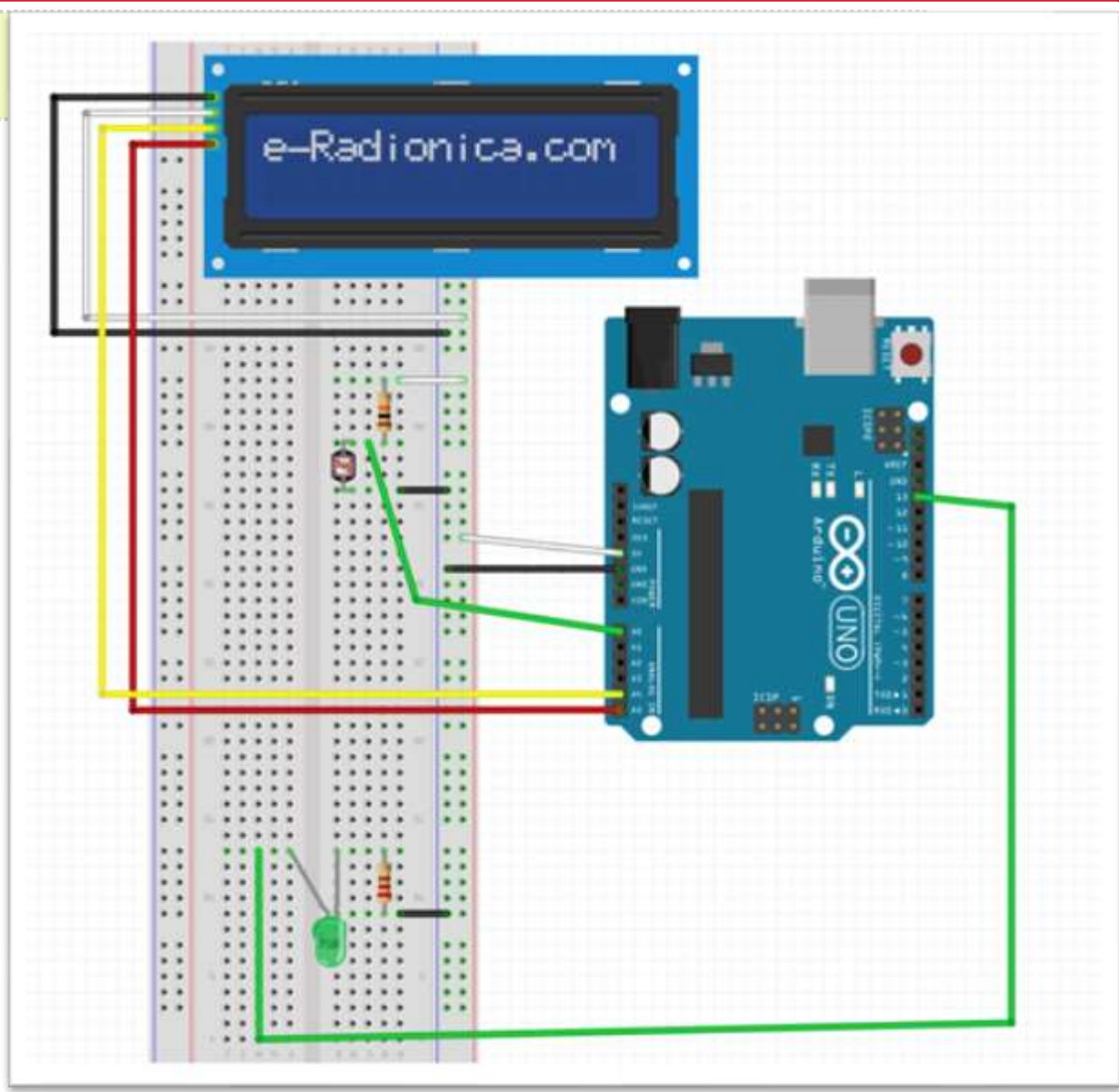


CdS LCD Project

LCD에 조도 값을
표시하면서
조도에 따라 **LED**를
ON/OFF



CdS_I2C_LCD_LED.fzz





CdS-LCD project

Set CdS-LCD project

Project

CdS 셀을 이용하여 조도를 측정해 보자.

1. CdS 셀로 측정된 조도를 아날로그 핀을 통하여 0~1023 범위로 읽는다.
2. ADC 값을 LCD 모듈로 **lux**로 출력한다. (빛의 밝기)
3. lux 값에 따라 D13에 연결된 단색 LED의 ON/OFF를 조정한다.

Hardware

1. LCD를 연결한다.
2. CdS셀과 10k Ω 저항을 연결한 뒤 저항의 한쪽 끝은 5V에
CdS셀의 한쪽 끝은 GND에 연결한다.
3. 저항과 CdS셀 사이를 아날로그 입력핀 A0에 연결한다.
4. 단색 LED를 220 또는 330 Ω 저항을 연결해서 디지털 입력핀 D13과 GND에 연결한다.



CdS-LCD project : new code

CdS 센서 LCD 회로 – code: AAnn_LCD_lux.ino

AAnn_LCD_lux_start §

```

1 /*
2 빛 입력 LCD 모니터링 및 제어
3 */
4 // LCD 라리브러리 설정
5 #include <LiquidCrystal_I2C.h>
6 #include <Wire.h>
7 // LCD 설정
8 LiquidCrystal_I2C lcd(0x27,16,2); // 0x3F
9 // 0번 아날로그핀을 CdS 셀 입력으로 설정한다.
10 const int CdSPin = 0; // CdS => A0
11 const int ledPin = 13; // LED pin => D13
12
13 // LED OFF above threshold lux
14
15 void setup() {
16   pinMode(ledPin, OUTPUT);
17   // 16X2 LCD 모듈 설정하고 백라이트를 켜다.
18   lcd.init();
19   lcd.backlight();
20   // 모든 메시지를 삭제한 뒤
21   // 숫자를 제외한 부분들을 미리 출력시킨다.
22   lcd.clear();
23   lcd.setCursor(0,0);
24   lcd.print("A000,ADC: ");
25   lcd.setCursor(0,1);
26   lcd.print("Light: ");
27   lcd.setCursor(13,1);
28   lcd.print("lux"); //
29 }

```

```

30 void loop(){
31   int adcValue; // 실제 센서로부터 읽은 값 (0~1023)
32   int illuminance; // 현재의 밝기. 0~100%
33   int lux; // 현재의 밝기. lux
34
35   // CdS cell을 통하여 입력되는 전압을 읽는다.
36   adcValue = analogRead(CdSPin);
37   // luminosity() 함수를 이용해서 Lux 를 계산한다.
38   lux = int(luminosity(adcValue));
39
40   // 전에 표시했던 내용을 지운다.
41   lcd.setCursor(12,0);
42   lcd.print(" ");
43   // ADC 값을 표시한다
44   lcd.setCursor(12,0);
45   lcd.print(adcValue);
46   // 전에 표시했던 내용을 지운다.
47   lcd.setCursor(9,1);
48   lcd.print(" ");
49   // 밝기를 표시한다
50   lcd.setCursor(9,1);
51   lcd.print(lux);
52
53   // On/Off LED by threshold
54
55
56   delay(1000);
57 }

```

LED ON/OFF

기능을 추가해서

Code를 완성 후,

AAnn_LCD_lux.

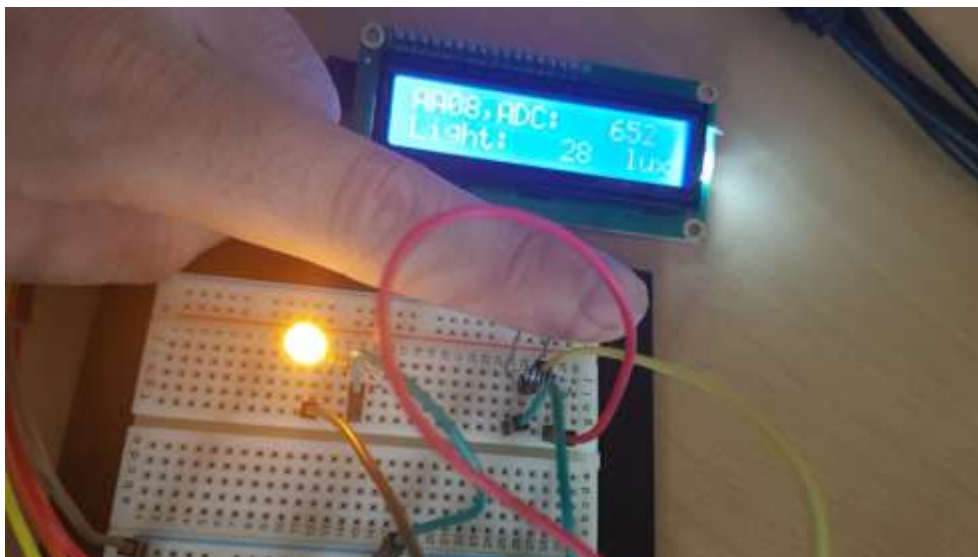
ino

로 저장...

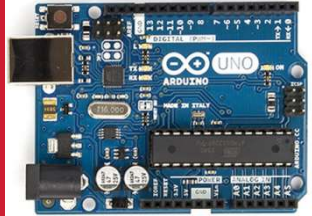
CdS 센서 LCD 회로 - 측정 결과

주변의 조도에 따라
어두우면 **LED**가
켜지고, 밝으면
LED가 꺼지도록
코드를 수정하시오.

LED가 켜진
화면을 폰으로
촬영해서 그림을
제출하시오.



조도에 따라 **LED**가 **ON/OFF** 되는 것을 확인 받고
결과 화면 촬영: [AAnn_LCD_lux.png](#) 로 저장...



[Practice]

◆ [wk04]

- **Arduino sensors**
- **Complete your project**
- **Upload folder: aann-rpt05**
- **Use repo “aann” in github**

◆ [Target of this week]

- Complete your works
- Save your outcomes and upload figures in github

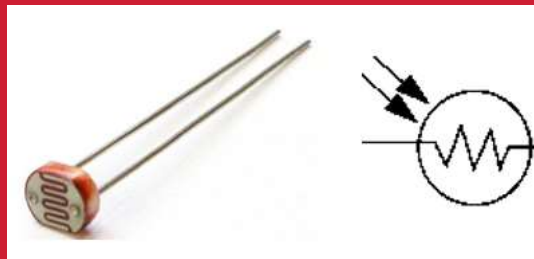
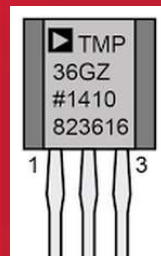
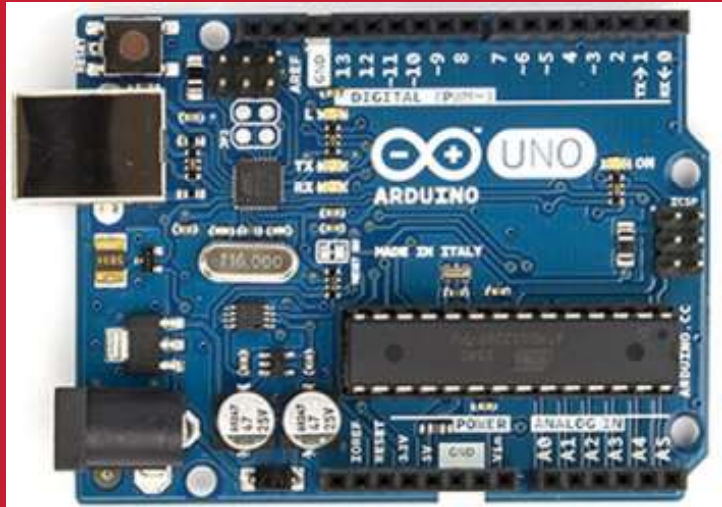
Upload folder : aann-rpt05

- 제출할 파일들

- ① **AAnn_AnalogVoltage.png**
- ② **AAnn_TMP36.png**
- ③ **AAnn_LCD_lux.png**
- ④ **All *.ino**



Arduino Sensors + Node.js





IOT: HSC

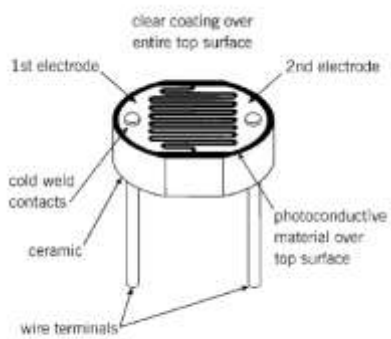
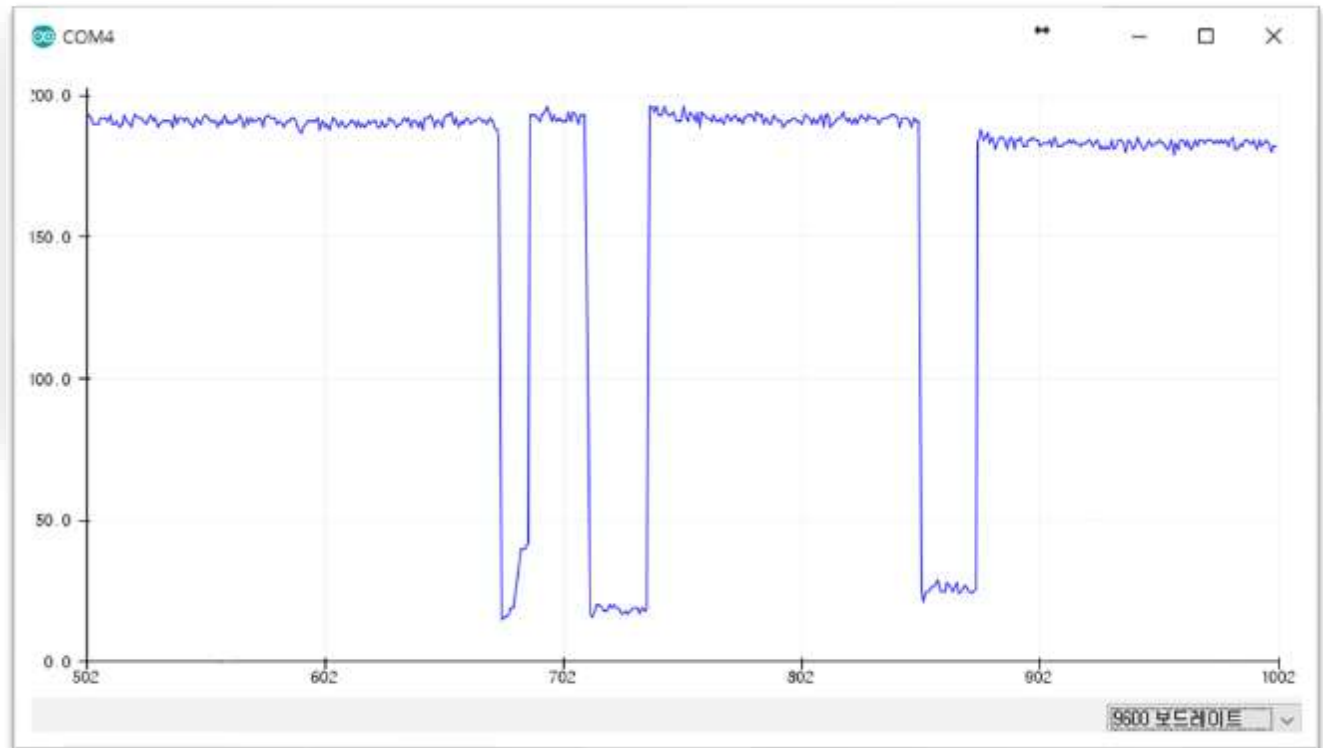
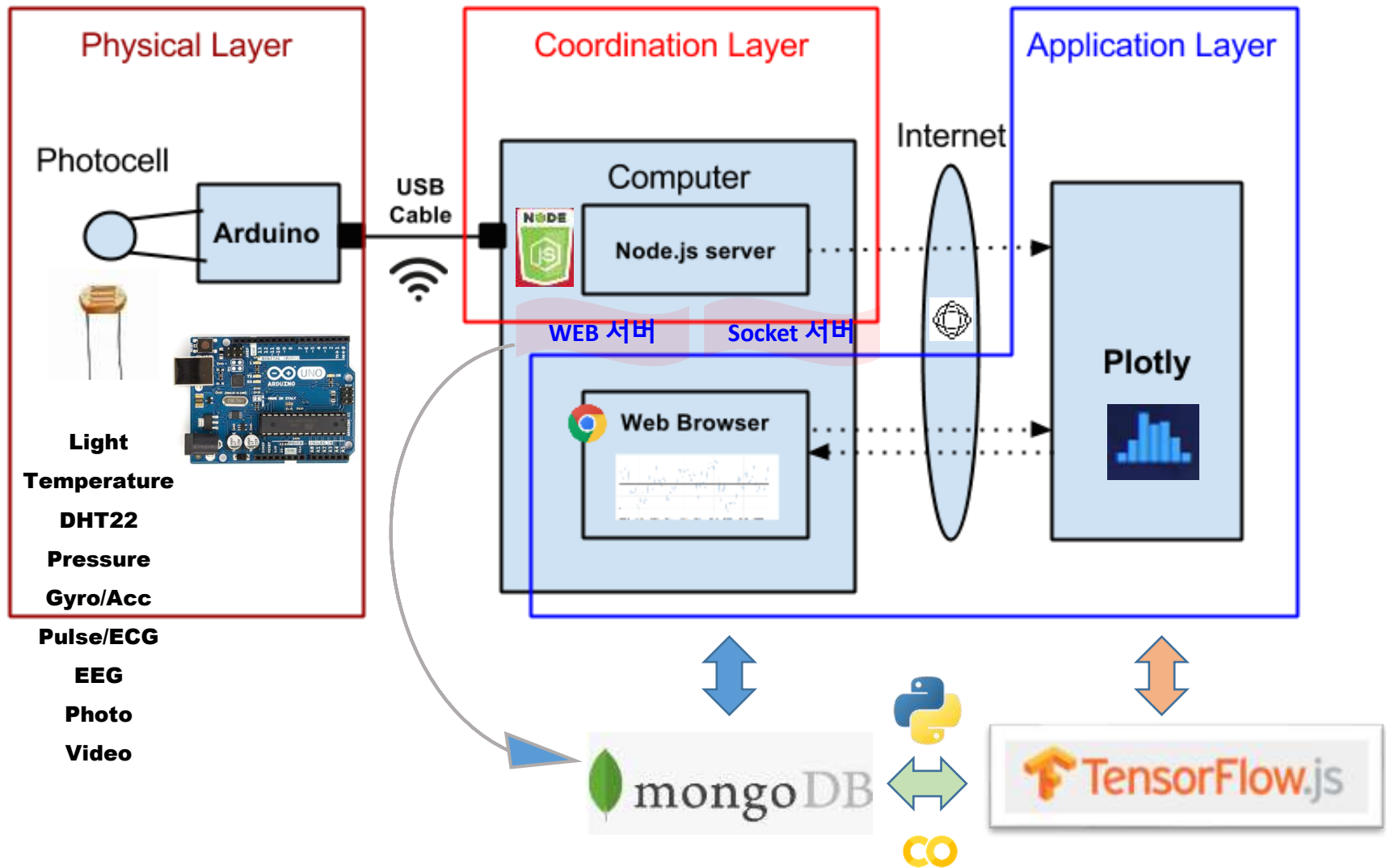


Figure 3
Typical Construction of a Plastic Coated Photocell



Layout [H S C]



on WEB monitoring Arduino data

IoT Signal from Arduino

Real-time Signals

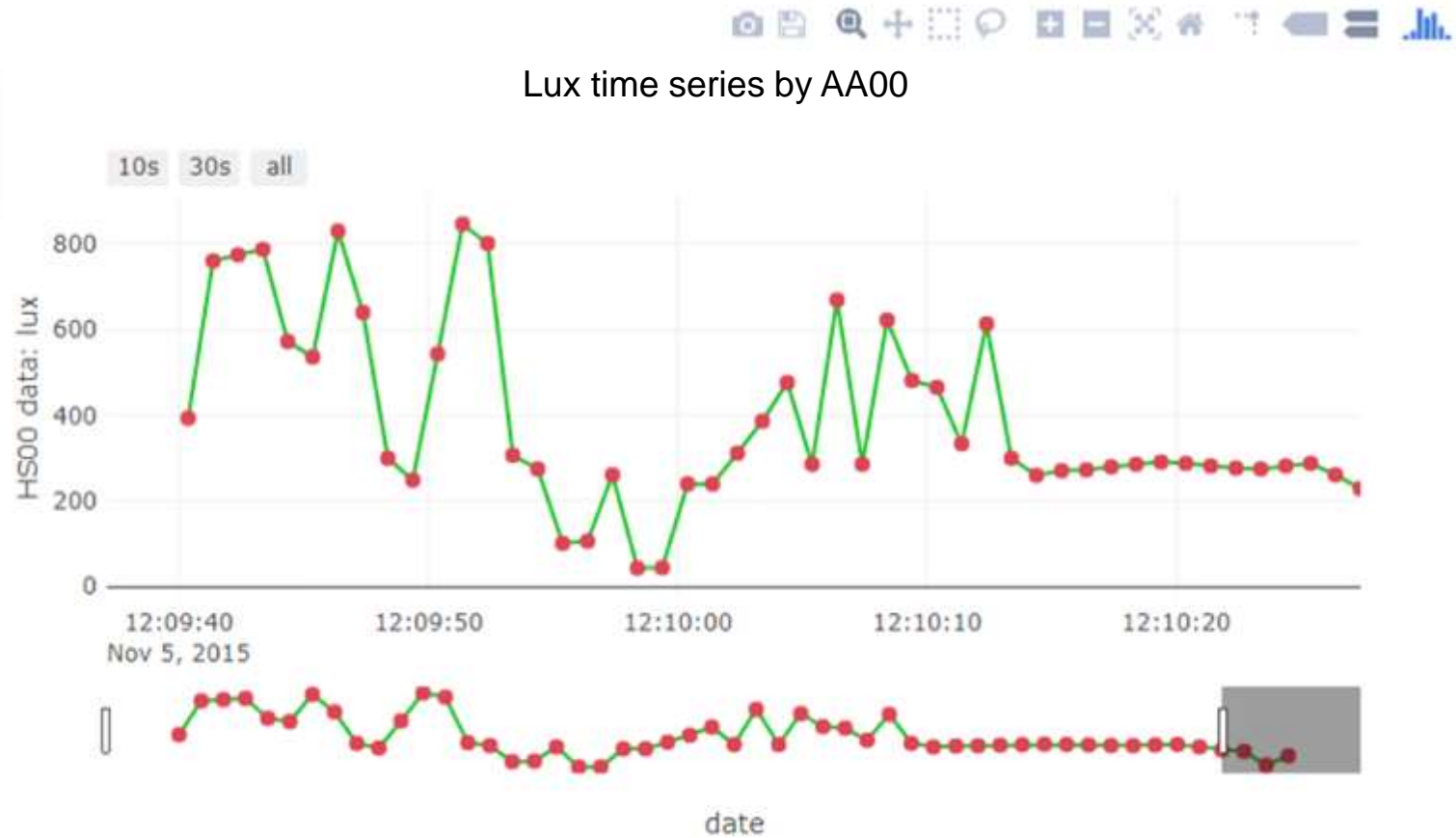
on Time: 2021-10-06 09:49:49.818

Signals (조도, 습도, 온도) : 166,60,-5

http://chaos.inje.ac.kr:3030/iot_multi.html

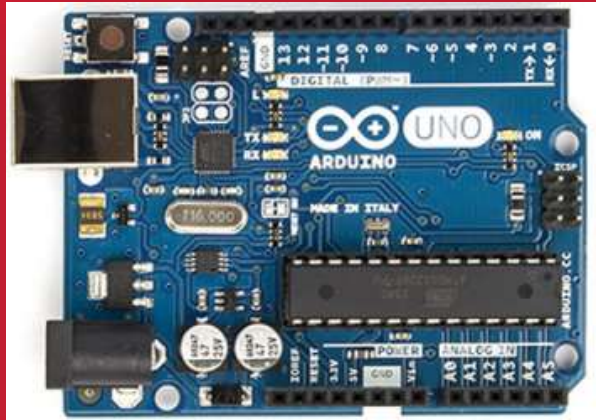
Arduino data + plotly

Time series by AA00



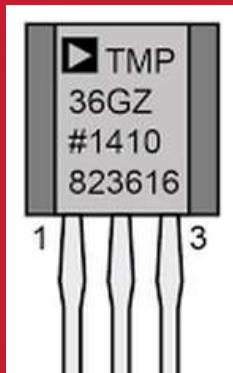


Single sensor: tmp36

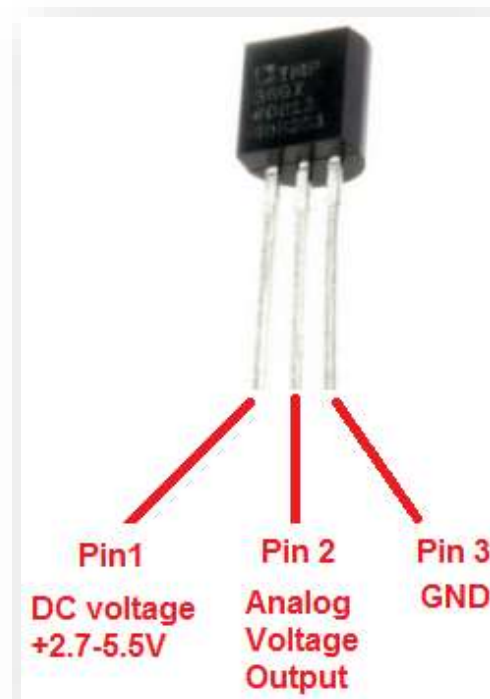
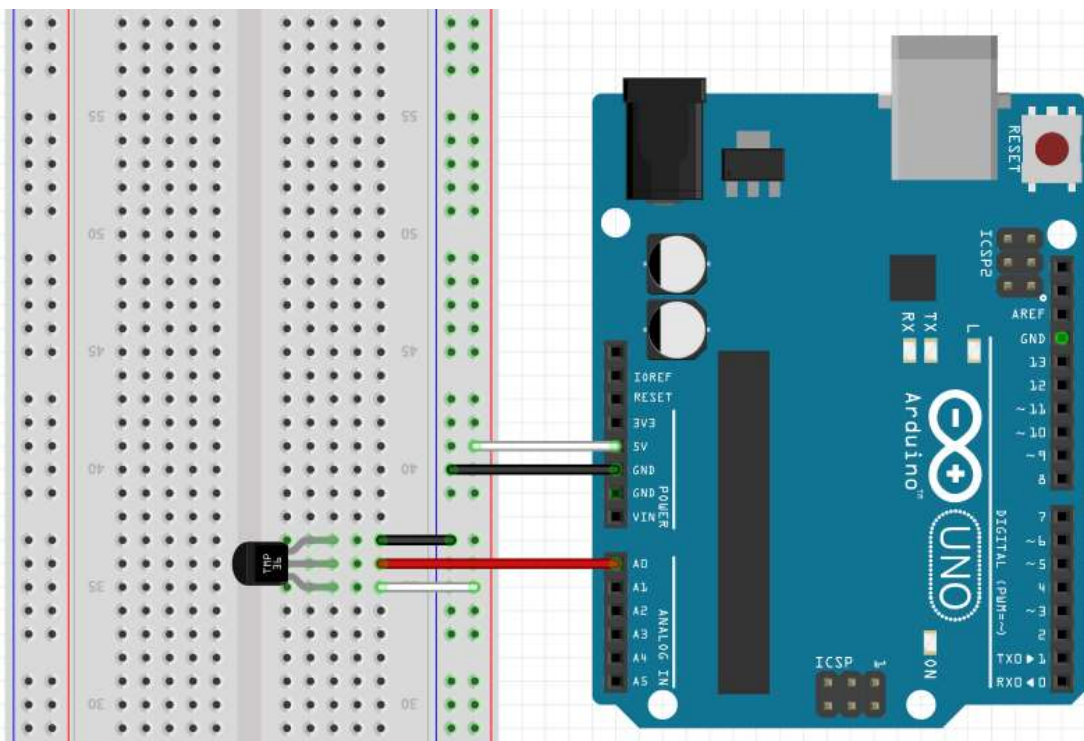


TMP36

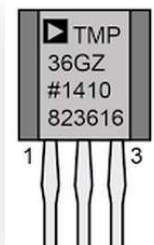
Node project



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: **aann-rpt06**
2. `md iot & cd iot`
3. `md tmp36`
4. `cd tmp36`
5. Open terminal
6. `npm init`





A4.1.3 tmp36 node project: package.json

탐색기

열려 있는 편집기

× npm package.json aann-rpt06...

제목 없음(작업 ...)

✓ aann

- > aann-rpt01
- > aann-rpt02
- > aann-rpt03
- > aann-rpt04
- > aann-rpt05
- ✓ aann-rpt06
 - > Arduino
 - ✓ iot\ tmp36
 - npm package.json
 - > Node
 - AA_수업계획서.pdf

npm package.json ×

./ aann > aann-rpt06 > iot > tmp36 > npm 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 }
17
```



A4.1.4 tmp36 node project: install modules

npm install --save serialport

```
D:\aann\aann-rpt06\iot\tmp36>npm install --save serialport
```

```
> @serialport/bindings@9.2.4 install D:\aann\aann-rpt06\iot\tmp36\node_modules\@serialport\bindings  
> prebuild-install --tag-prefix @serialport/bindings@ || node-gyp rebuild
```

npm notice created a lockfile as package-lock.json. You should commit this file.

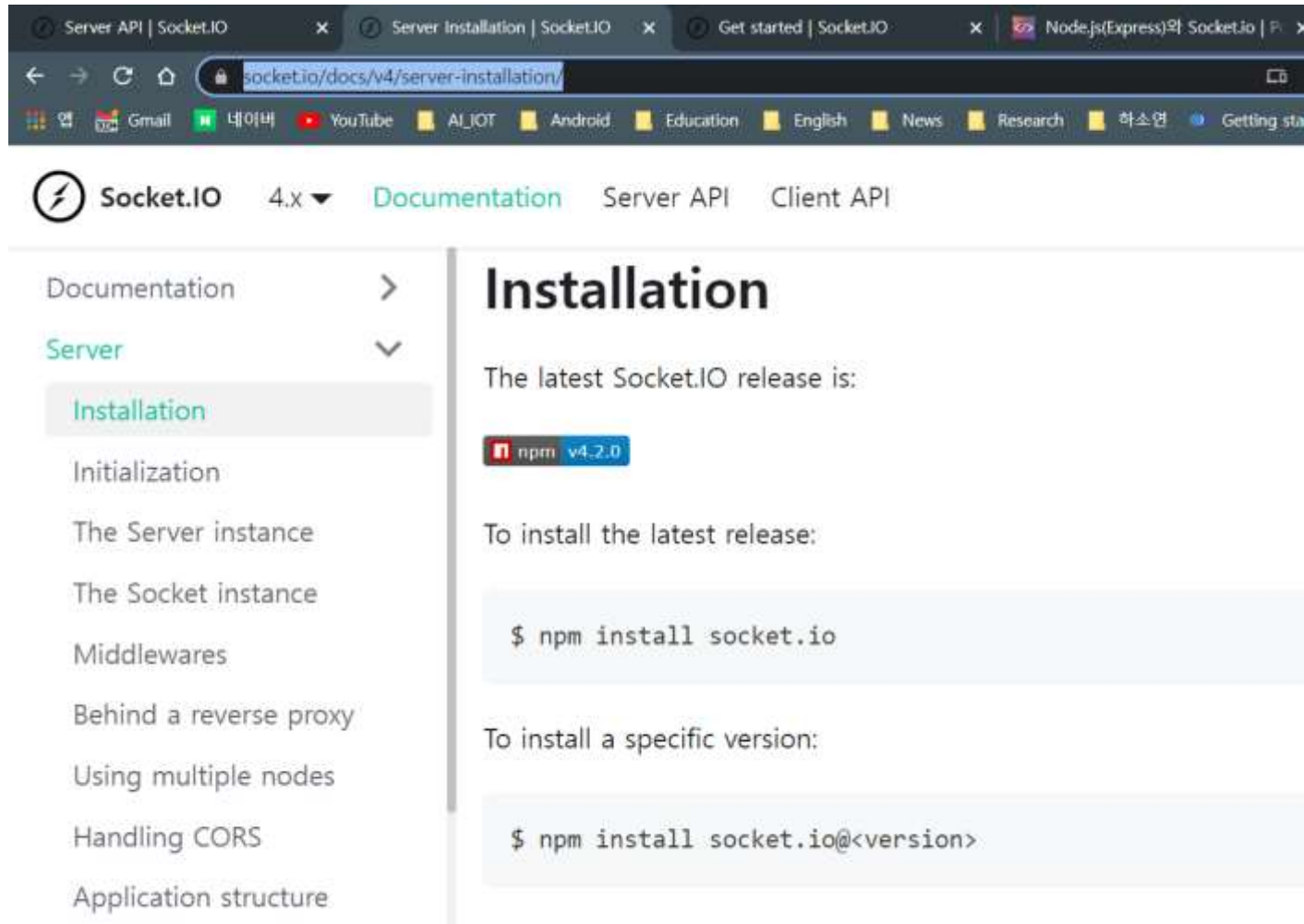
npm WARN tmp36@1.0.0 No repository field.

```
+ serialport@9.2.4  
added 74 packages from 45 contributors and audited 74 packages in 11.774s
```

17 packages are looking for funding
run `npm fund` for details

found 0 vulnerabilities

socket.io



Server API | Socket.IO x Server Installation | Socket.IO x Get started | Socket.IO x Node.js(Express)와 Socket.io | P x

socket.io/docs/v4/server-installation/

Socket.IO 4.x Documentation Server API Client API

Documentation >

Server >

Installation

Initialization

The Server instance

The Socket instance

Middlewares

Behind a reverse proxy

Using multiple nodes

Handling CORS

Application structure

Installation

The latest Socket.IO release is:

npm v4.2.0

To install the latest release:

```
$ npm install socket.io
```

To install a specific version:

```
$ npm install socket.io@<version>
```

<https://socket.io/docs/v4/server-installation/>



A4.1.4 tmp36 node project: install modules

npm install --save socket.io@2.3.0

```
D:\aann\aann-rpt06\iot\tmp36>npm install --save socket.io@2.3.0
npm WARN tmp36@1.0.0 No repository field.
```

```
+ socket.io@2.3.0
added 52 packages from 33 contributors and audited 126 packages in 3.878s
```

```
17 packages are looking for funding
  run `npm fund` for details
```

```
found 4 vulnerabilities (2 moderate, 1 high, 1 critical)
  run `npm audit fix` to fix them, or `npm audit` for details
```

D:\aann\aann-rpt06\iot\tmp36 디렉터리

2021-10-05	오전 10:23	<DIR>	.
2021-10-05	오전 10:23	<DIR>	.
2021-10-05	오전 10:23	<DIR>	node_modules
2021-10-05	오전 10:23		28,477 package-lock.json
2021-10-05	오전 10:23		367 package.json
	2개 파일		28,844 바이트
	3개 디렉터리		2,424,474,251,264 바이트 남음



A4.1.4 tmp36 node project: install modules

npm install --save socket.io@2.3.0

found 4 vulnerabilities (2 moderate, 1 high, 1 critical)
run `npm audit fix` to fix them, or `npm audit` for details

```
D:\aann\aann-rpt06\iot\tmp36>npm audit fix  
npm WARN tmp36@1.0.0 No repository field.
```

+ socket.io@2.4.1

added 1 package, removed 11 packages, updated 11 packages and moved 1 package in 1.33s

17 packages are looking for funding
run `npm fund` for details

fixed 1 of 4 vulnerabilities in 126 scanned packages
1 package update for 3 vulnerabilities involved breaking changes
(use `npm audit fix --force` to install breaking changes; or refer to
`npm audit` for steps to fix these manually)



A4.1.4 tmp36 node project: install modules

npm install --save socket.io [N.A.]

4.x 버전 설치는 좀 더 검토가 필요.

```
D:\aann\aann-rpt06\iot\tmp36>npm install --save socket.io
npm WARN tmp36@1.0.0 No repository field.
```

```
+ socket.io@4.2.0
added 20 packages from 66 contributors and audited 94 packages in 2.046s
```

```
17 packages are looking for funding
  run `npm fund` for details
```

```
found 0 vulnerabilities
```

D:\aann\aann-rpt06\iot\tmp36 디렉터리

2021-10-05	오전 10:23	<DIR>	.
2021-10-05	오전 10:23	<DIR>	.
2021-10-05	오전 10:23	<DIR>	node_modules
2021-10-05	오전 10:23		28,477 package-lock.json
2021-10-05	오전 10:23		367 package.json
	2개 파일		28,844 바이트
	3개 디렉터리		2,424,474,251,264 바이트 남음

정상 동작 버전을 설치!

npm install --save serialport

npm install --save socket.io@2.4.1

```
"author": "aa00",
"license": "MIT",
"dependencies": {
  "serialport": "^9.2.4",
  "socket.io": "^2.4.1"
}
```




A4.1.5 tmp36 node project (Arduino code)

AAnn_TMP36_NodeJS.ino

```
12 void loop() {  
13   //getting the voltage reading from the temperature sensor  
14   int value = analogRead(TEMP_INPUT);  
15   Serial.print("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 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  
value = 150 : 733.14 mV, 23.31 degrees C  
value = 149 : 728.25 mV, 22.83 degrees C  
value = 150 : 733.14 mV, 23.31 degrees C  
value = 149 : 728.25 mV, 22.83 degrees C  
value = 150 : 733.14 mV, 23.31 degrees C  
value = 150 : 733.14 mV, 23.31 degrees C  
value = 149 : 728.25 mV, 22.83 degrees C
```

☒ 자동 스크롤 ☐ 타임스탬프 표시 line ending 없음



A4.1.6 tmp36 node project (node code)

tmp36_node_start.js

```
3  var serialport = require("serialport");
4  var portName = "COM3"; // check your COM port!!
5  var port = process.env.PORT || 3000;
6
7  var io = require("socket.io").listen(port);
8
9  const Readline = require("@serialport/parser-readline");
10 // serial port object
11 var sp = new serialport(portName, {
12   baudRate: 9600, // 9600 38400
13   dataBits: 8,
14   parity: "none",
15   stopBits: 1,
16   flowControl: false,
17   parser: new Readline("\r\n"),
18 });
```

```
20 const parser = sp.pipe(new Readline({ delimiter: "\r\n" }));
21
22 // Read the port data
23 sp.on("open", () => {
24   console.log("serial port open");
25 });
26
27 var tdata = []; // Array
28
29 parser.on("data", (data) => {
30   // call back when data is received
31   // raw data only
32   //console.log(data);
33
34   tdata = data; // data
35   console.log("AA00," + tdata);
36   io.sockets.emit("message", tdata); // send data to all clients
37 });
```



A4.1.7 tmp36 node project (node cmd message)

[Terminal] [node tmp36_node.js](#)

```
D:\aann\aann-rpt06\iot\tmp36>node tmp36_node
serial port open
67.35 mV, 26.74 degrees C
67.35 mV, 26.74 degrees C
7 : 767.35 mV, 26.74 degrees C
7 : 767.35 mV, 26.74 degrees C
AA00, value = 157 : 767.35 mV, 26.74 degrees C
AA00, value = 157 : 767.35 mV, 26.74 degrees C
AA00, value = 157 : 767.35 mV, 26.74 degrees C
AA00, value = 157 : 767.35 mV, 26.74 degrees C
AA00, value = 155 : 757.58 mV, 25.76 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 = 156 : 762.46 mV, 26.25 degrees C
AA00, value = 156 : 762.46 mV, 26.25 degrees C
```



A4.1.8 tmp36 node project (all messages)

tmp36_node.js

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

```
parser.on("data", (data) => {
  // call back when data is received
  // raw data only
  // console.log(data);
```

```
  dStr = getDateString();
  tdata[0] = dStr;
  tdata[1] = data; // data
  console.log("AA00," + tdata.toString());
  io.sockets.emit("message", tdata); // send
});
```

```
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;
}
```

[Terminal] node tmp36_node

```
D:\aann\aann-rpt06\iot\tmp36>node tmp36_node
serial port open
AA00,2021-10-05 11:21:24.062,lue = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:24.062,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:24.062,value = 157 : 767.35 mV, 26.74 degrees C
AA00,2021-10-05 11:21:24.062,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:24.063,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:24.063,value = 157 : 767.35 mV, 26.74 degrees C
AA00,2021-10-05 11:21:25.644,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:26.648,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:27.651,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:28.651,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:29.655,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:30.658,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:31.662,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:32.661,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:33.665,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:34.669,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:35.672,value = 156 : 762.46 mV, 26.25 degrees C
AA00,2021-10-05 11:21:36.676,value = 155 : 757.58 mV, 25.76 degrees C
AA00,2021-10-05 11:21:37.675,value = 156 : 762.46 mV, 26.25 degrees C
```



AAnn_tmp36_message.png
로 저장



A4.1.9 tmp36 node project (only data)

AA00_TMP36_NodeJS.ino 수정

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 (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



A4.1.10 tmp36 node project (date & data → IOT)

[Terminal] node tmp36_node

```
D:\aann\aann-rpt06\iot\tmp36>node tmp36_node
```

```
serial port open
```

```
AA00,2021-10-05 11:31:03.941,26.25
```

```
AA00,2021-10-05 11:31:04.944,26.25
```

```
AA00,2021-10-05 11:31:05.945,26.25
```

```
AA00,2021-10-05 11:31:06.948,26.25
```

```
AA00,2021-10-05 11:31:07.951,26.25
```

```
AA00,2021-10-05 11:31:08.951,26.25
```

```
AA00,2021-10-05 11:31:09.954,25.76
```

```
AA00,2021-10-05 11:31:10.954,26.25
```

```
AA00,2021-10-05 11:31:11.958,26.25
```

```
AA00,2021-10-05 11:31:12.957,26.25
```

```
AA00,2021-10-05 11:31:13.961,26.25
```

```
AA00,2021-10-05 11:31:14.964,26.25
```

```
AA00,2021-10-05 11:31:15.964,26.25
```

시간 , 온도

IOT data format

시간, data

시간, 온도

AAnn_tmp36_IOT_data.png

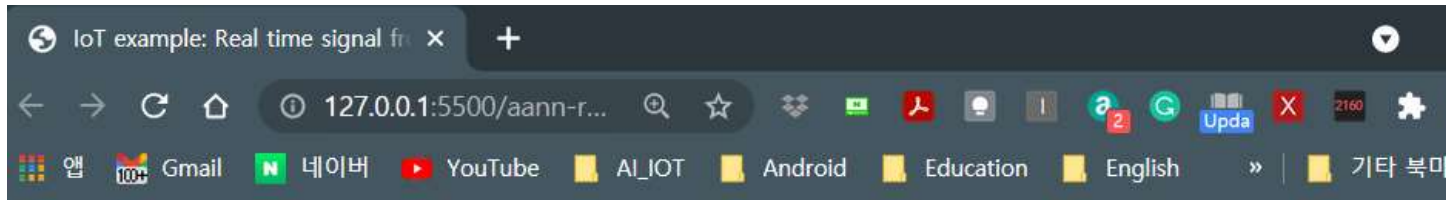
로 저장

공백없이 “,”로
시간과 온도 구분



A4.1.11 tmp36 node project (web monitoring)

[Web monitoring] [client_signal_tmp36.html](#)

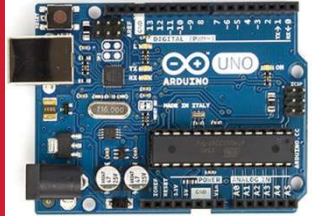


IoT Signal from Arduino

Real-time Signals

on Time: 2021-10-05 11:47:53.803

Signal (temp) : 25.76



[Practice]

◆ [wk04]

- **Arduino sensors + Node.js**
- **Complete your project**
- **Upload folder: aann-rpt06**
- **Use repo “aann” in github**

wk04 : Practice : aann-rpt06

◆ [Target of this week]

- Complete your works
- Save your outcomes and upload outputs in github repo.

제출폴더명 : **aann-rpt06**

- 압축할 파일들

- ① **AAnn_tmp36_message.png**
- ② **AAnn_tmp36_IOT_data.png**
- ③ **AAnn_tmp36_IOT_WEB.png**
- ④ **All *.ino**
- ⑤ **All *.js**
- ⑥ **NO node_modules folder**

● 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

Target of this class

Real-time Weather Station from nano 33 BLE sensors



on Time: 2020-09-09 10:27:17.321

