



Arduino-IoT

[wk03]

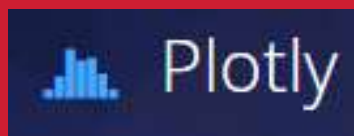
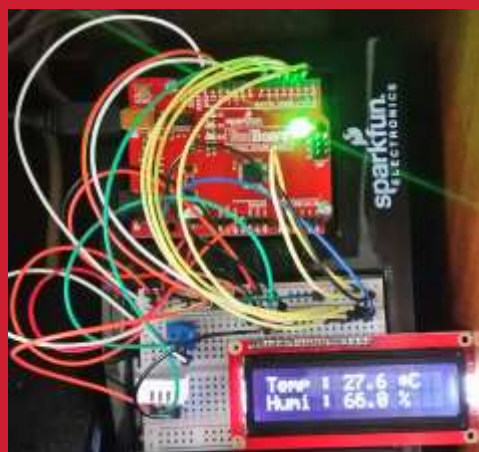
node.js express

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**



[Practice]

◆ [wk02]

- Node module : **aanninfo.js**
- Upload folder: **aann-rpt02**



[practice] local module : aanninfo.js

index_aann.js uses local module **aanninfo.js** in start subfolder.

```
1 // index_aann.js
2
3 var myinfo = require('./aanninfo');
4
5 myinfo("aa00", "Redwoods", '010-1234-5678');
6
7 myinfo("aa55", "Comsi", '010-5678-1234');
```

My Info
ID : aa00
Name : Redwoods
Phone : 010-1234-5678

My Info
ID : aa55
Name : Comsi
Phone : 010-5678-1234

[Finished in 0.2s]

Save as
AAnn_info.png



[practice] local module : aanninfo.js

How to make aanninfo.js in start subfolder.

1. Make local module – aanninfo.js
2. Call aanninfo.js from index_aann.js.
3. Capture your result.

```
index_aann.js  x  aanninfo.js  x
1 // aanninfo.js
2
3 module.exports = function(id, name, phone) {
4     console.log("My Info");
5     console.log("ID : " + id);
6     console.log("Name : " + name);
7     console.log("Phone : " + phone + "\n");
8 }
```

[\[참고\] Node local module 만들기](#)

◆ [Target of this week]

My Info using node module – aanninfo.js

Upload folder : aann-rpt02

- 제출할 파일들

- ① **AAnn_package.png**
- ② **AAnn_HTTP.png**
- ③ **AAnn_TCP_Log.png**
- ④ **AAnn_Upload.png**
- ⑤ **AAnn_info.png**
- ⑥ **start folder**
- ⑦ **server folder**

Purpose of AA

주요 수업 목표는 다음과 같다.

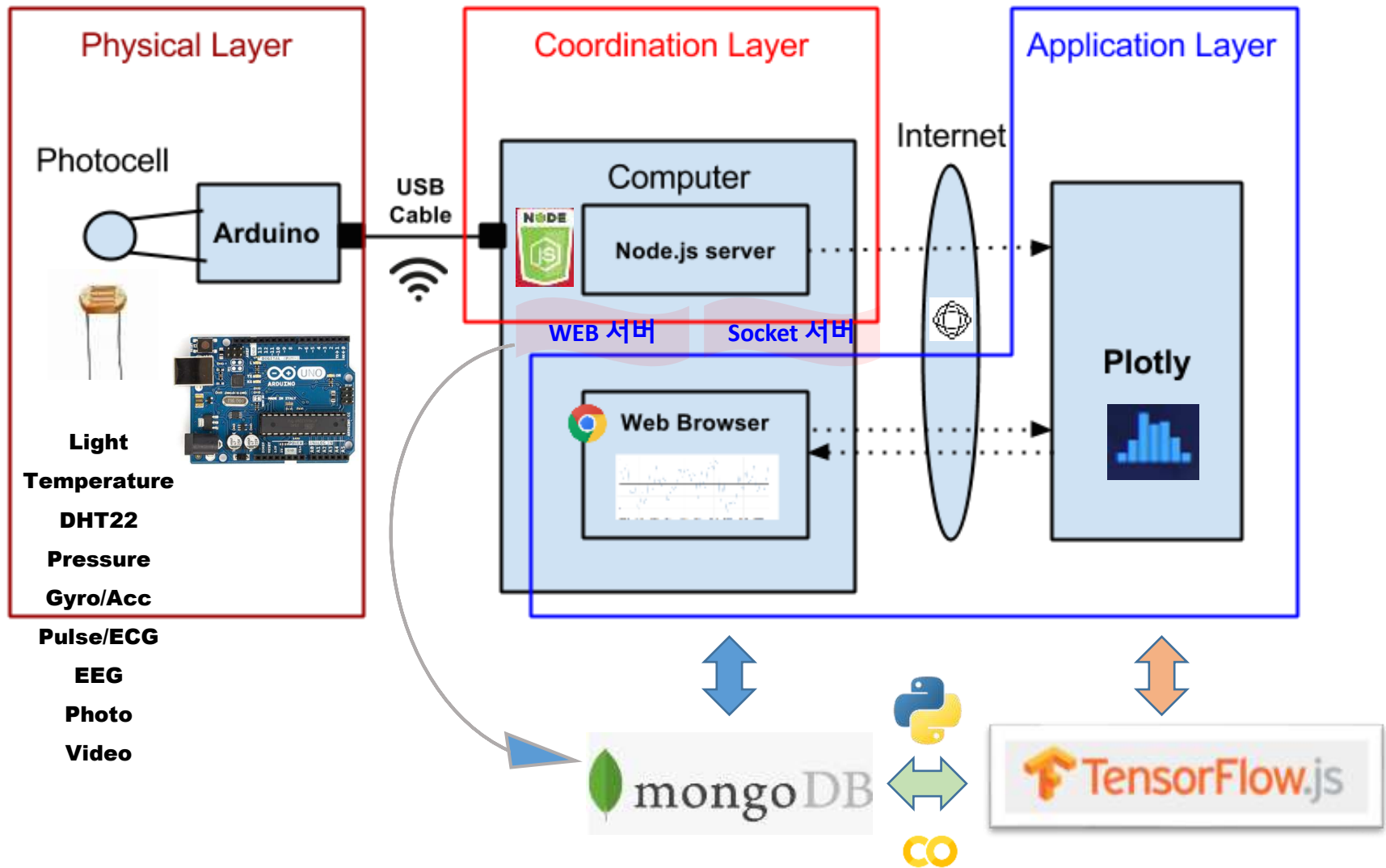
1. Node.js를 이용한 아두이노 센서 신호 처리
2. Plotly.js를 이용한 아두이노 센서 신호 시각화
3. MongoDB에 아두이노 센서 데이터 저장 및 처리



4. 저장된 IoT 데이터의 마이닝 (파이썬 코딩)



Layout [H S C]





Node.js Server

1. http, tcp, file

2. Express



6. Node Server

Node Server I.

- 1. HTTP server**
- 2. TCP server**
- 3. File upload**



7. Node Server

Node Server II.

- 1. Express server**
2. Full Express App
3. My Express App



7.1.1 Express server test

Step 1 : npm init

Step 2 : npm install --save express

Step 3 : Write Express code

Step 4 : Run app.js

Step 5 : <http://localhost:3000>

Step 6 : Routing test



7.1.2 Express server test

탐색기 ... 문제 출력 디버그 콘솔 터미널

▽ 열려 있는 편집기

▽ 제목 없음(작업 영역)

▽ aann

 ▽ aann-rpt01

 JS aann-hello.js

 > aann-rpt02

 > aann-rpt03

 AA_수업계획서...

Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

D:\aann\aann-rpt03>

새 파일
새 폴더
파일 탐색기에 표시 <Shift> + <Alt> + R
통합 터미널에서 열기
폴더에서 찾기... <Shift> + <Alt> + F
잘라내기 Ctrl+X
복사 Ctrl+C
붙여넣기 Ctrl+V

1. Make folder aann-rpt03

**2. Go to the folder,
"aann-rpt03"**

> npm init



7.1.2 Express server test: `npm init`

```
D:\aann\aann-rpt03>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 init`` for definitive documentation on these fields
and exactly what they do.

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

Press `^C` at any time to quit.

```
package name: (aann-rpt03) express-test
```

```
version: (1.0.0)
```

```
description: test express server
```

```
entry point: (index.js) app.js
```

```
test command:
```

```
git repository:
```

```
keywords:
```

```
author: aa00
```

```
license (ISC) MIT
```

package.json

탐색기

열려 있는 편집기

package.json aa...

제목 없음...

aann

aann-rpt01

JS aann-hello.js

aann-rpt02

aann-rpt03

package.json

AA_수업계획서....

package.json X

aann > aann-rpt03 > package.json > .

```

1 {
2   "name": "express-test",
3   "version": "1.0.0",
4   "description": "test express server",
5   "main": "app.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "author": "aa00",
10  "license": "MIT"
11 }
12

```

문제

출력

디버그 콘솔

터미널

D:\aann\aann-rpt03 디렉터리

```

2021-09-14 오후 05:12 <DIR> .
2021-09-14 오후 05:12 <DIR>
2021-09-14 오후 05:12 229 package.json
                        1개 파일                229 바이트
                        2개 디렉터리  2,467,576,836,096 바이트 남음

```



7.1.4 Express server test: express module install

npm install --save express

문제

출력

디버그 콘솔

터미널

cmd

```
D:\aann\aann-rpt03>npm install --save express
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN express-test@1.0.0 No repository field.
```

```
+ express@4.17.1
added 50 packages from 37 contributors and audited 50 packages in 2.942s
found 0 vulnerabilities
```

```
D:\aann\aann-rpt03>dir
D 드라이브의 볼륨: DATA
볼륨 일련 번호: 82D1-4852
```

```
D:\aann\aann-rpt03 디렉터리
```

| | | | |
|------------|----------|-------|--------------------------|
| 2021-09-14 | 오후 05:16 | <DIR> | . |
| 2021-09-14 | 오후 05:16 | <DIR> | .. |
| 2021-09-14 | 오후 05:16 | <DIR> | node_modules |
| 2021-09-14 | 오후 05:16 | | 14,349 package-lock.json |
| 2021-09-14 | 오후 05:16 | | 279 package.json |
| | 2개 파일 | | 14,628 바이트 |
| | 3개 디렉터리 | | 2,467,574,353,920 바이트 남음 |

package.json

package.json X

./ aann > aann-rpt03 > package.json > ...

```

1  {
2    "name": "express-test",
3    "version": "1.0.0",
4    "description": "test express server",
5    "main": "app.js",
6    "scripts": {
7      "test": "echo \"Error: no test specified\" && exit 1"
8    },
9    "author": "aa00",
10   "license": "MIT",
11   "dependencies": {
12     "express": "^4.17.1"
13   }
14 }
15

```

프로젝트 폴더 내의 **node_modules** subfolder에 **express server modules**들이 저장되어 서버 기능을 지원.
그리고 **package.json**에 **express** 모듈 정보가 “**dependencies**” 속성에 저장.

7.1.6 Express server test: app.js

JS app.js

./ aann > aann-rpt03 > JS app.js > ...

```

1  // app.js
2  var express = require('express');
3  var app = express();
4  var port = 3000;

5
6  app.get('/', function(req, res) {
7    res.send('<a href="/hello">Hello Page</a>');
8  });
9
10 app.get('/hello', function(req, res) {
11   res.send('Hello aa00');
12 });
13
14 app.get('/comsi', function(req, res) {
15   res.send('Hello Comsi!');
16 });
17
18 var server = app.listen(port, function() {
19   console.log('Listening on port %d', server.address().port);
20 });
  
```

Express server

routing

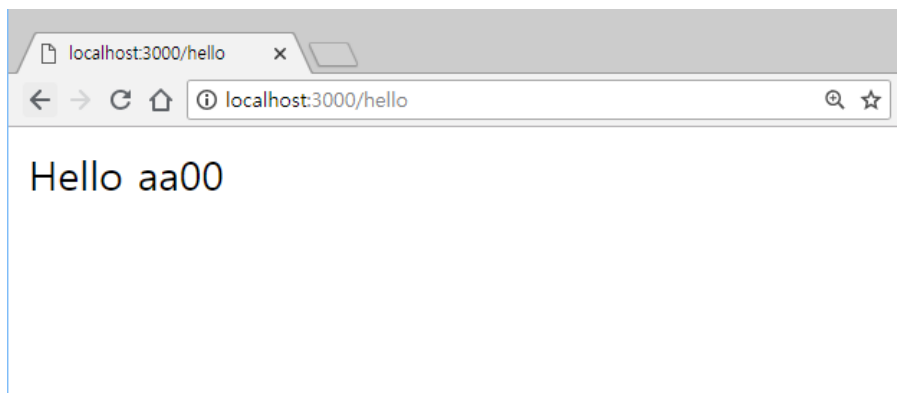
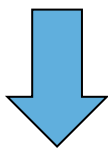


7.1.7 Express server test: run **app.js**

```
D:\aann\aann-rpt03>node app  
Listening on port 3000  
█
```



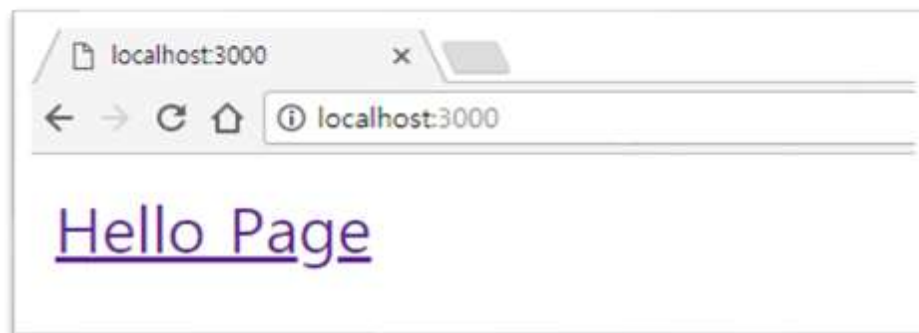
[Hello Page](#)



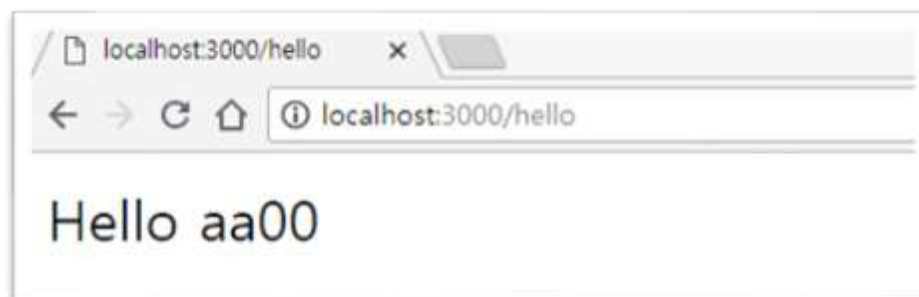


7.1.8 Express server test: test server **routing**

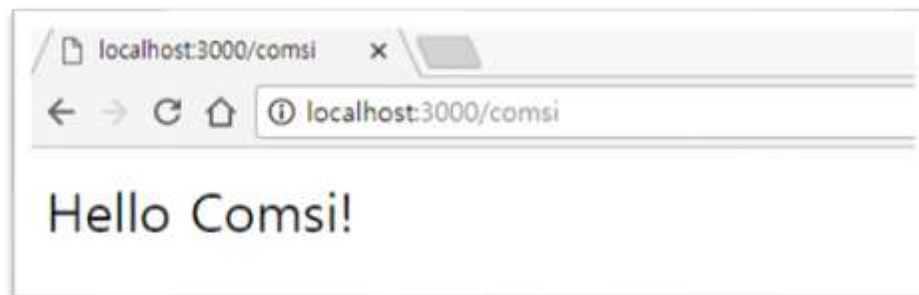
localhost:3000



localhost:3000/hello



localhost:3000/comsi

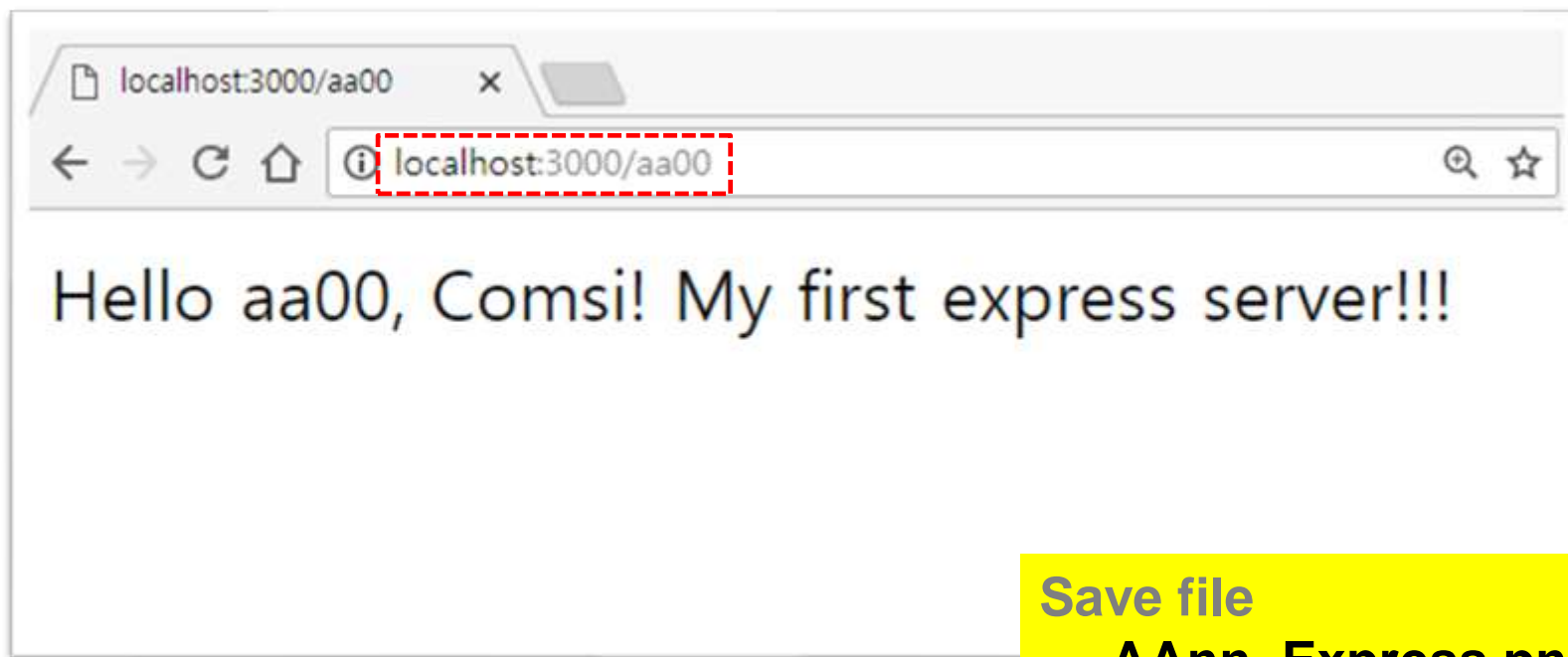




7.1.9 Express server test: run **app.js** – DIY

[DIY] My ID routing → **localhost:3000/aann**

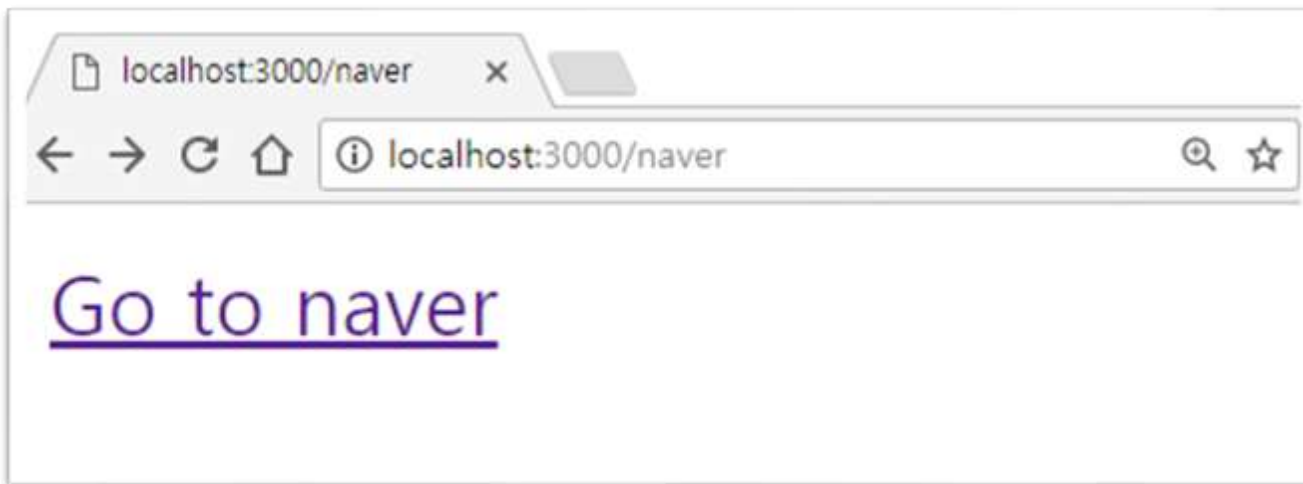
```
app.get('/aa00', function(req, res) {  
  res.send('Hello aa00, Comsi! My first express server!!!');  
});
```



Save file
AAnn_Express.png

Routing: 라우팅은 애플리케이션 엔드 포인트(**URI**)의 정의, 그리고 **URI**가 클라이언트 요청에 응답하는 방식

[DIY] Go to naver.com



[Hint] `Go to naver`



7.1.12 Express web server test: app2.js

```
JS app2.js
aann > aann-rpt03 > JS app2.js > ...
1 // app.js
2 var express = require('express');
3 var app = express();
4 var port = 3030;
5
6 var path = require('path');
7
8 app.get('/', function(req, res) {
9   res.send('<a href="/hello">Hello Page</a>');
10 });
11 app.get('/hello', function(req, res) {
12   res.send('Hello aa00');
13 });
14 app.get('/comsi', function(req, res) {
15   res.send('Hello Comsi!');
16 });
17
18 app.use(express.static(path.join(__dirname, 'public')));
19
20 var server = app.listen(port, function() {
21   console.log('Listening on port %d', server.address().port);
22 });
```

Save file

AAnn_Express_2server.png

문제 출력 디버그 콘솔 터미널

```
D:\aann\aann-rpt03>node app
Listening on port 3000
```

```
D:\aann\aann-rpt03>node app2
Listening on port 3030
```



[Practice]

◆ [wk03]

- Express server
- Upload folder: aann-rpt03
- Use repo “aann” in github

◆ [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**



Arduino



Home

Buy

Download

Products ▼

Learning ▼

Forum

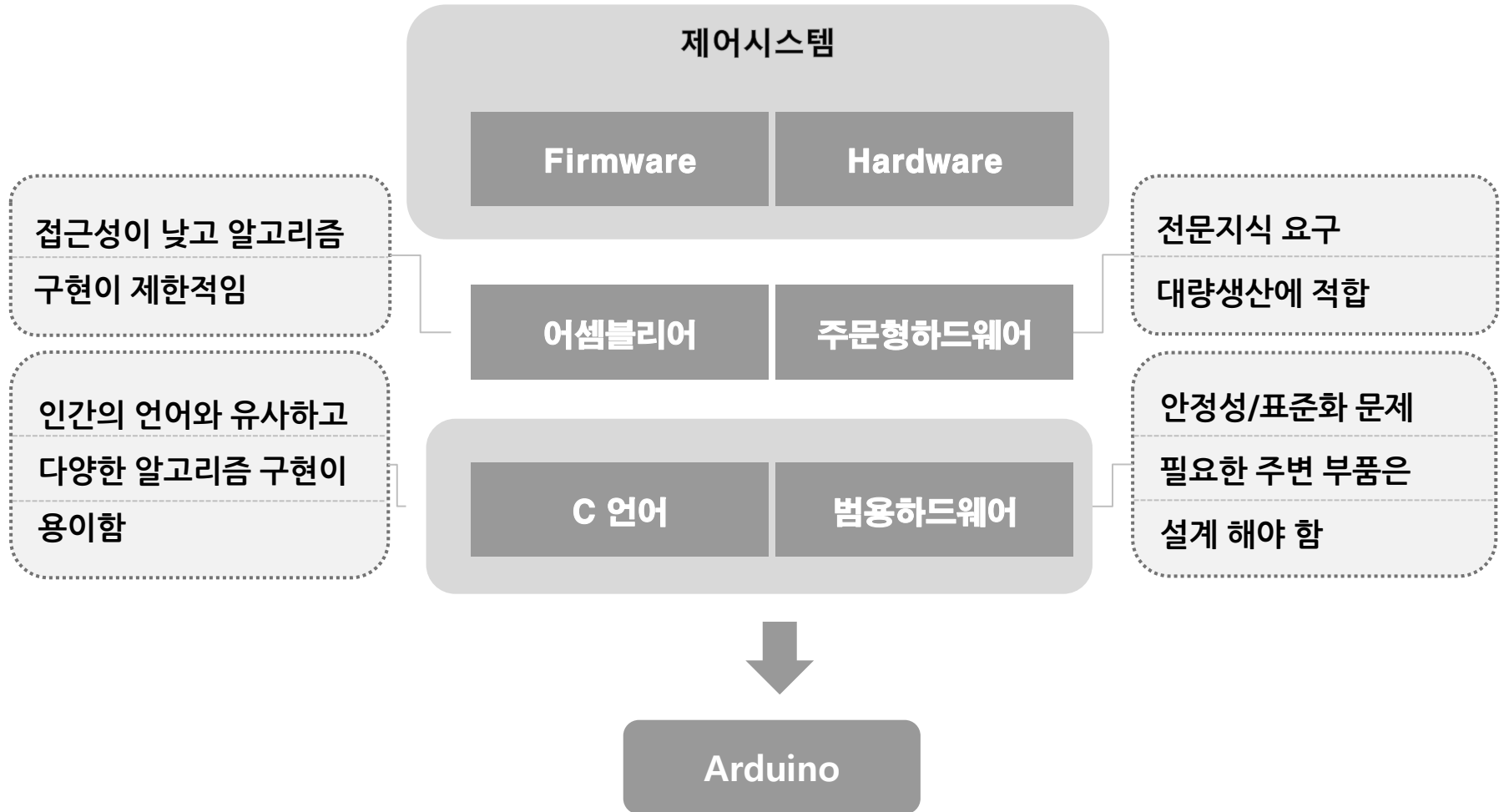
Support ▼

Blog

<https://www.arduino.cc/>



0.1 Arduino 란?



0.1 Arduino 란?

2005년 Italy의 Massimo Banzi & David Cuatielles에 의해 개발

예술가
취미생활
학생

전자공학
교육

누구나
쉽게
사용
가능한
제어장치

오픈소스
하드웨어

GSM Wifi
Ethernet
Motor drive
등의 쉘드
제공

다양한
라이브러리



LabView

MATLAB

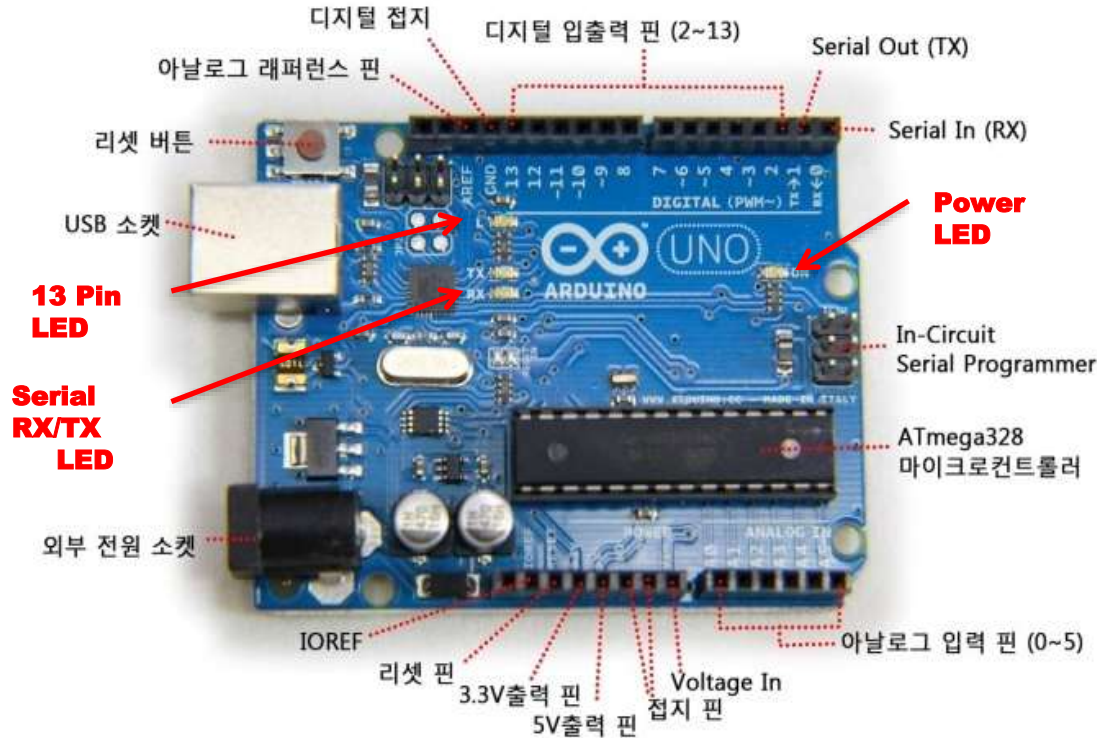
Node.js
Plot.ly

Mongo DB

Tensorflow.js

범용
하드웨어
IOT 의
표준

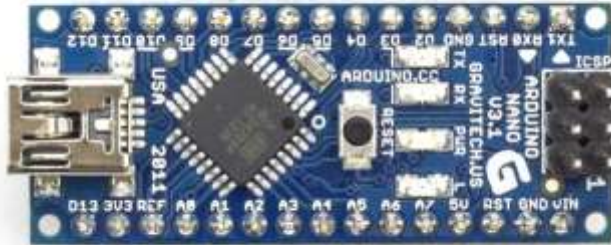
0.2 Arduino hardware



✓ Arduino UNO R3

- ATmega328 microcontroller
- Input voltage: 7~12V
- 14 Digital I/O Pins (6 PWM outputs)
- 6 Analog Inputs
- 32KB Flash Memory
- 16Mhz Clock Speed

0.2.1 Arduino hardware: Nano, Nano33



✓ Arduino Pro NANO

- ATmega168/328 microcontroller
- Input voltage: 7~12V
- 14 Digital I/O Pins (6 PWM outputs)
- 8 Analog Inputs
- 16KB Flash Memory
- 16Mhz Clock Speed



✓ Arduino NANO33 BLE SENSOR

- ◆ Color, brightness, proximity and gesture sensor
- ◆ Digital microphone
- ◆ Motion, vibration and orientation sensor
- ◆ Temperature, humidity and pressure sensor
- ◆ Arm Cortex-M4 microcontroller and BLE module

0.2.2 Arduino hardware

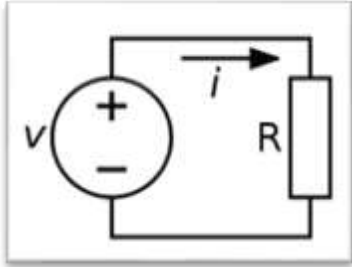


✓ Arduino Uno WiFi

ESP8266 Wi-Fi Module

- ATmega328p microcontroller
- Input voltage: 7~12V
- 14 Digital I/O Pins (6 PWM outputs)
- 8 Analog Inputs
- ESP8266 Wi-Fi
- 16Mhz Clock Speed

0.3 전압, 전류, 저항



전압
[V]

- ✓ 전위가 높은 쪽과 낮은 쪽의 차이
- ✓ 1쿨롱(coulomb: 전하의 단위)의 전하가 갖고 있는 에너지
- ✓ Arduino에서는 직류 3.3[V]와 5[V]를 지원

전류
[A]

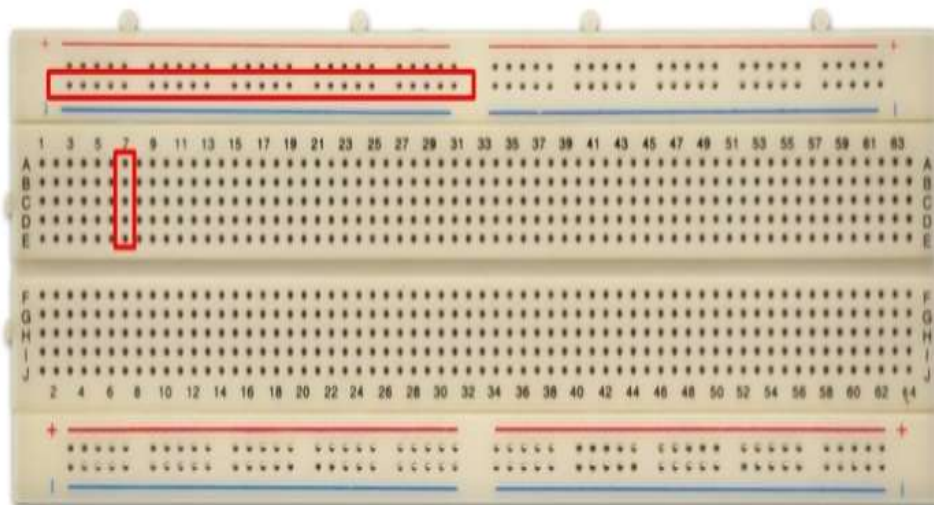
- ✓ 1초당 1쿨롱의 전하가 단위 면적을 통과했을 때를 1[A]로 정의
- ✓ Arduino에서는 1/1000[A] 단위인 [mA]를 사용

저항
[Ω]

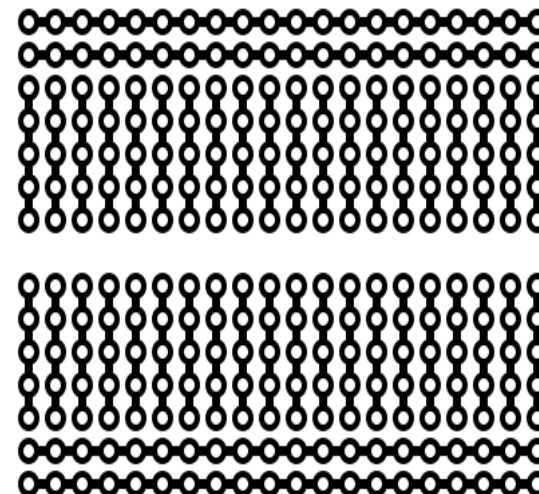
- ✓ 전류의 흐름을 방해하는 정도를 나타냄
- ✓ 색 띠나 숫자로 값을 표시
- ✓ Arduino에서는 칩 (chip) 형태의 저항이 사용

0.4 브레드 보드 (Bread board)

시제품 제작이나 실험용 와이어를 보드에 꽂아 사용



빨간색 묶음 홀끼리 내부회로가 연결되어 있음



내부 결선



Arduino SW

<http://fritzing.org/home/>

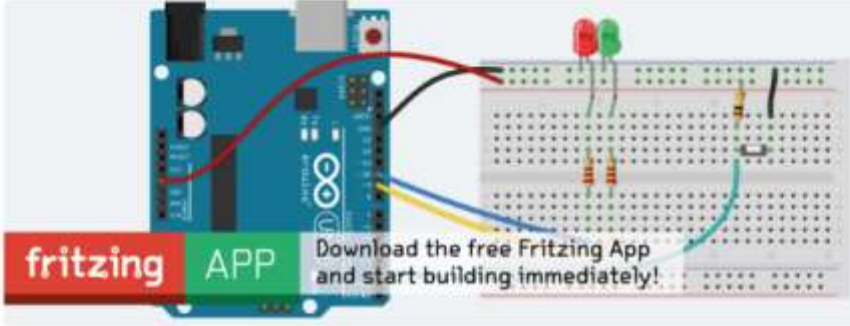
fritzing.org · Fritzing Fritzing

fritzing

electronics made easy

Projects · Parts · Download · Learning · Services · Contribute

FORUM FAB



fritzing APP Download the free Fritzing App and start building immediately!

Fritzing is an open-source hardware initiative that makes electronics accessible as a creative material for anyone. We offer a software tool, a community website and services in the spirit of Processing and Arduino, fostering a creative ecosystem that allows users to document their prototypes, share them with others, teach electronics in a classroom, and layout and manufacture professional pcbs.

Download and Start

Download our latest version 0.9.3b released on June 2, 2016 and start right away.

Produce your own board

With Fritzing Fab you can easily and inexpensively turn your circuit into a real, custom-made PCB. Try it out now!

Participate

Fritzing can only act as a creative platform if many



Fritzing configuration

fritzing electronics
made easy

Projects Parts **Download** Learning Services Contribute

FORUM

FAB

Fritzing is open source, free software. Be aware that the development of it depends on the active support of the community.

Select the download for your platform below.

Version **0.9.3b** was released on June 2, 2016.

Windows 32 bit

Windows 64 bit

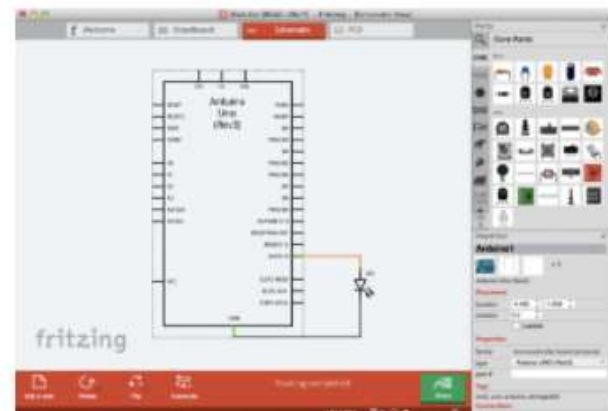
Mac OS X 10.7 and up

Linux 32 bit

Linux 64 bit

Source Github

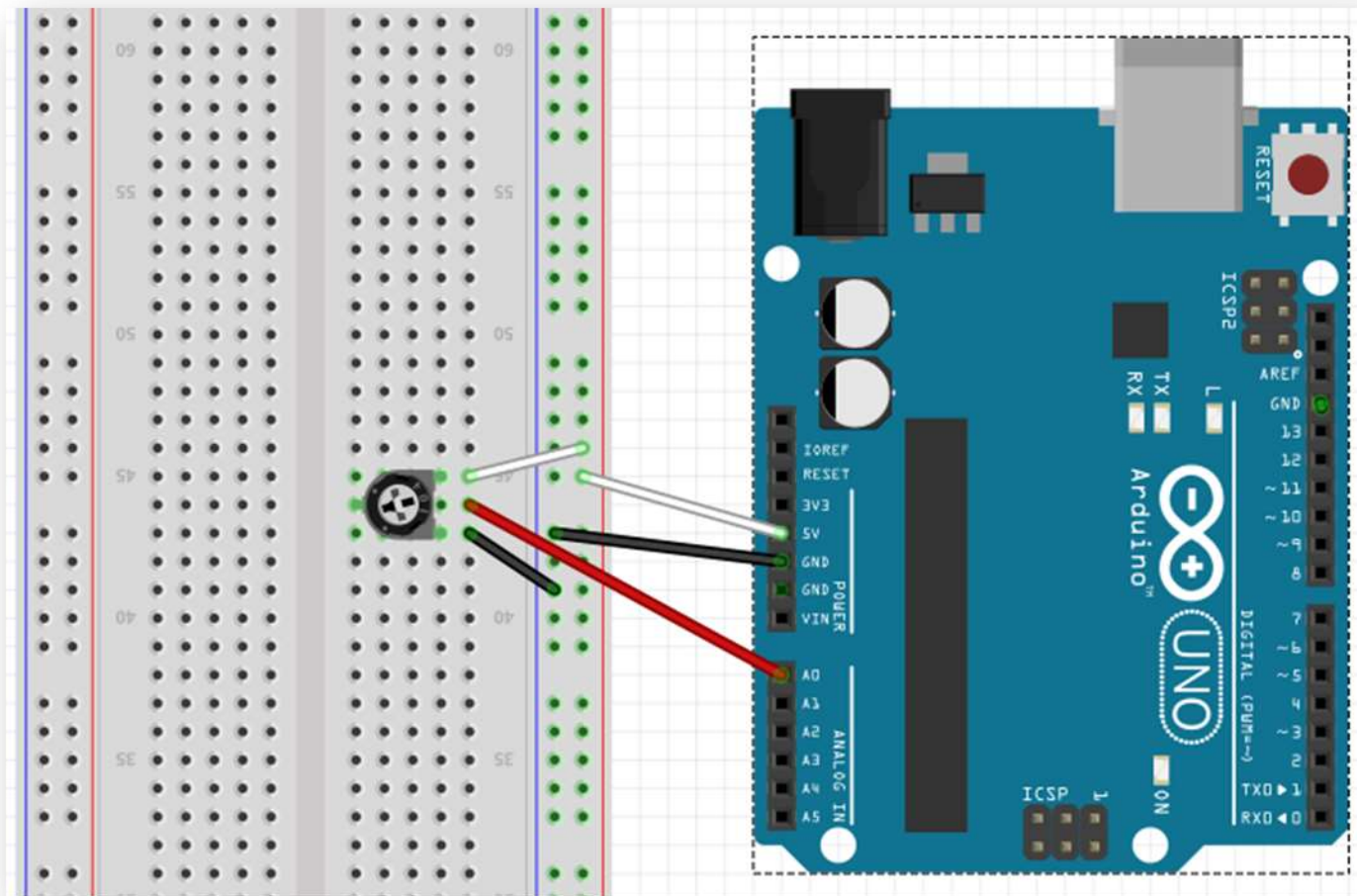
Downloaded 2578877 times.





Arduino circuits

Potentiometer (가변 저항기)

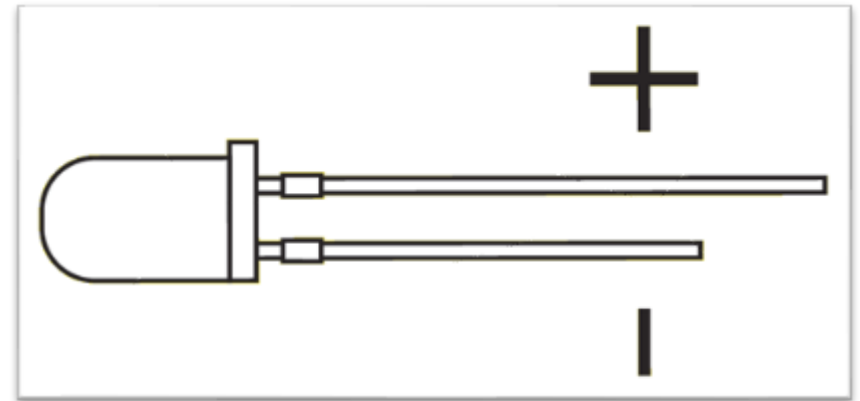


Parts : 가변저항기

Polarity of Diode and LED



The diode circuit symbol, with the anode and cathode marked.

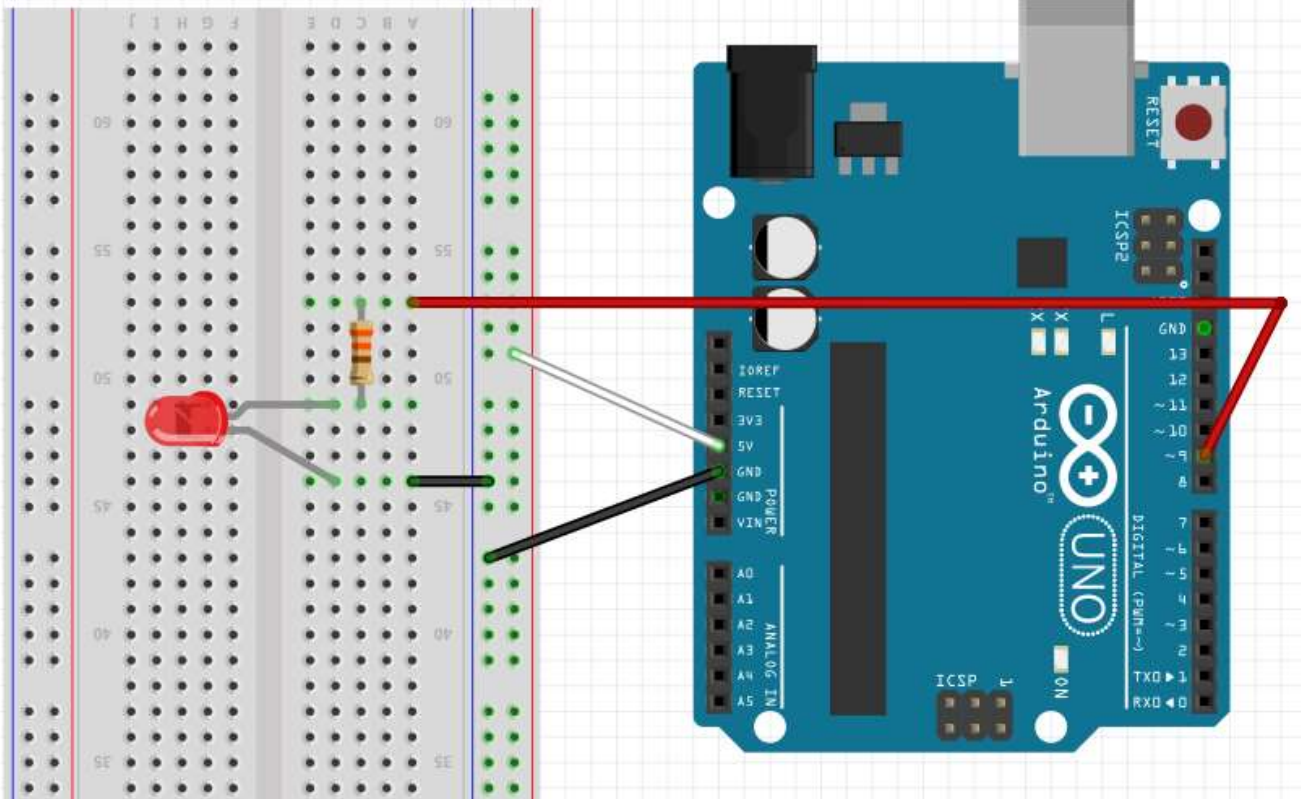


Find the longer leg, which should indicate the positive, anode pin.

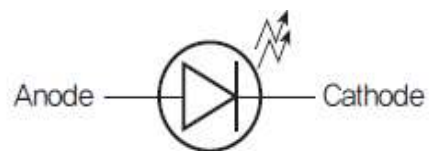
<https://learn.sparkfun.com/tutorials/polarity/diode-and-led-polarity>



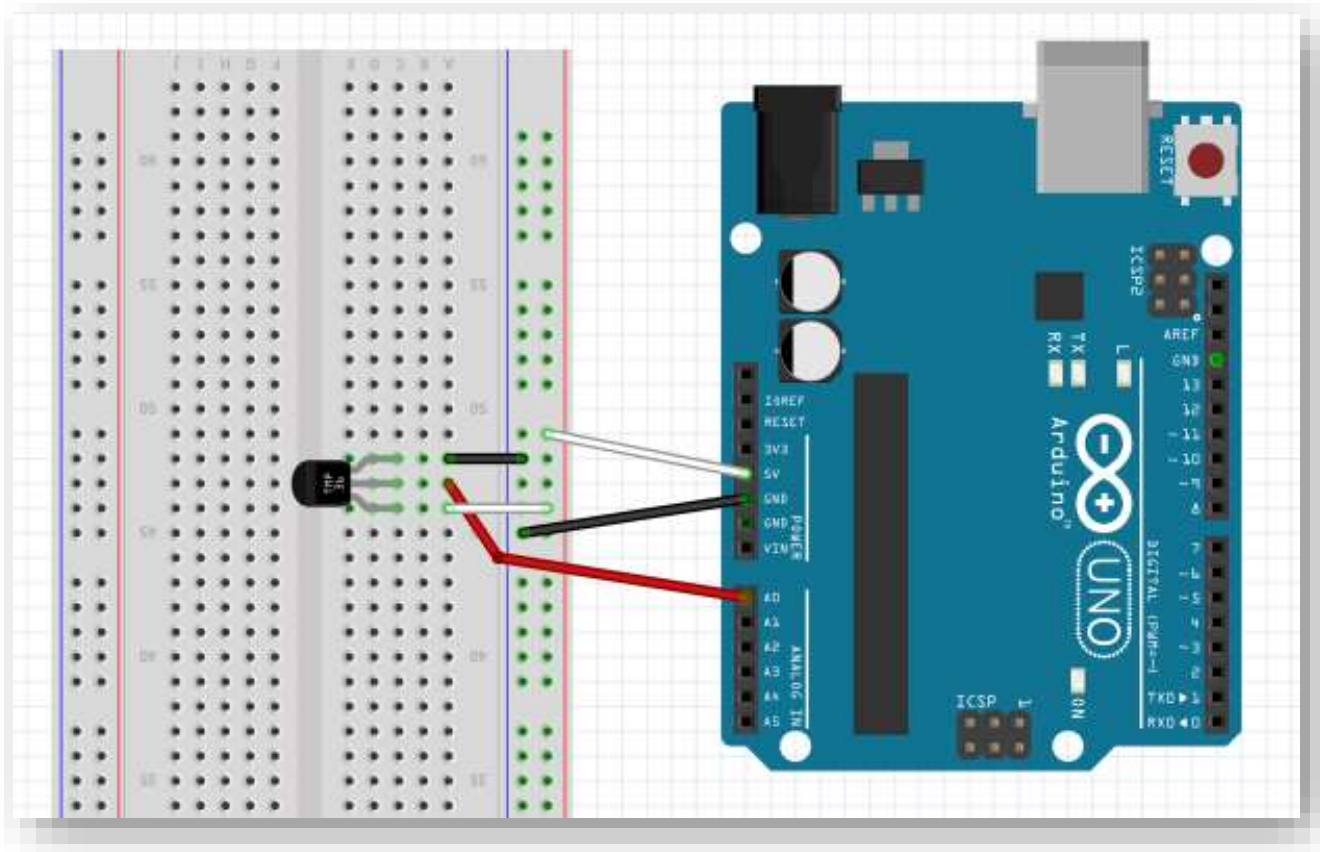
single LED



Parts : LED (1), R (330 Ω X 1)

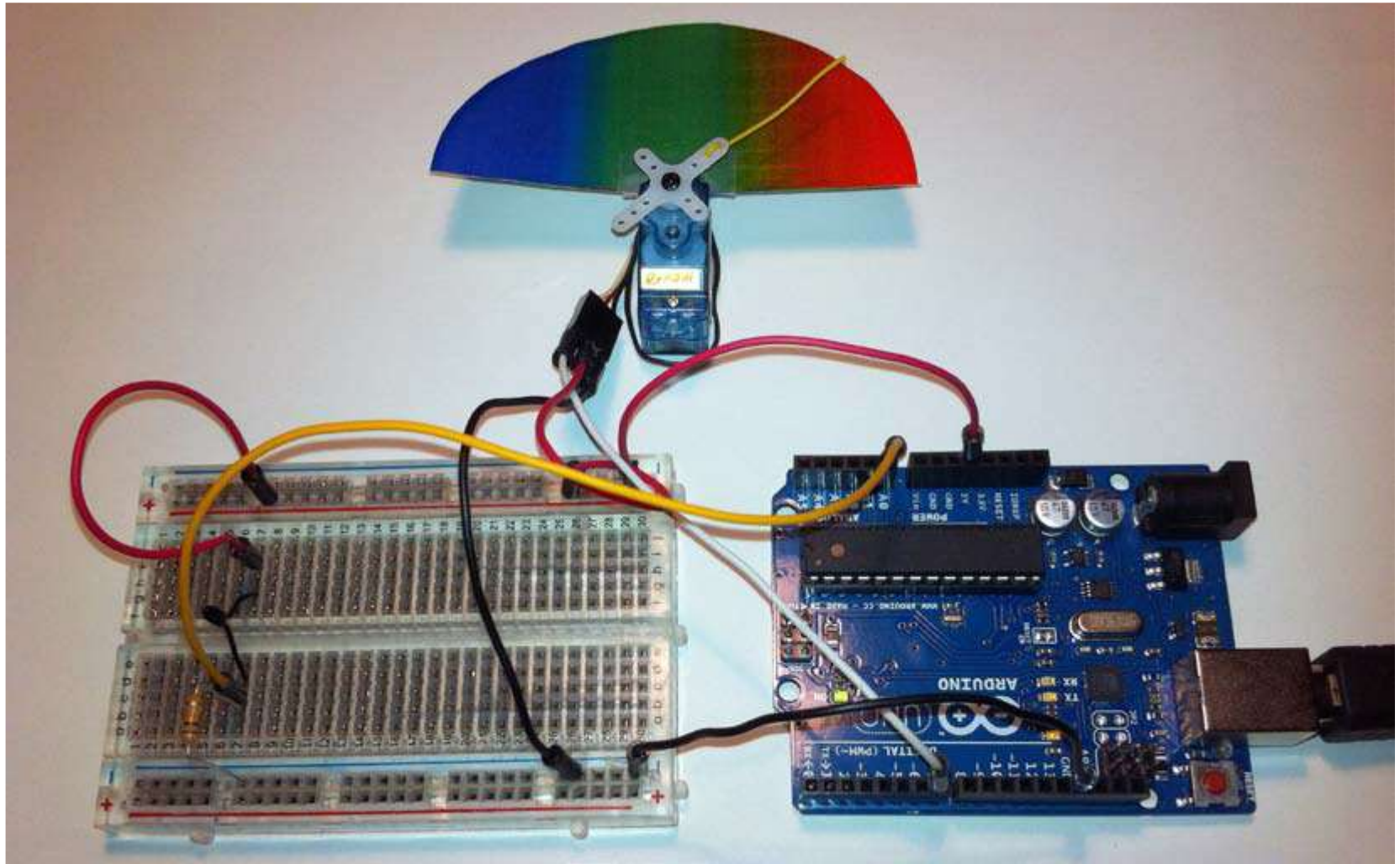


Temperature sensor (TMP36)



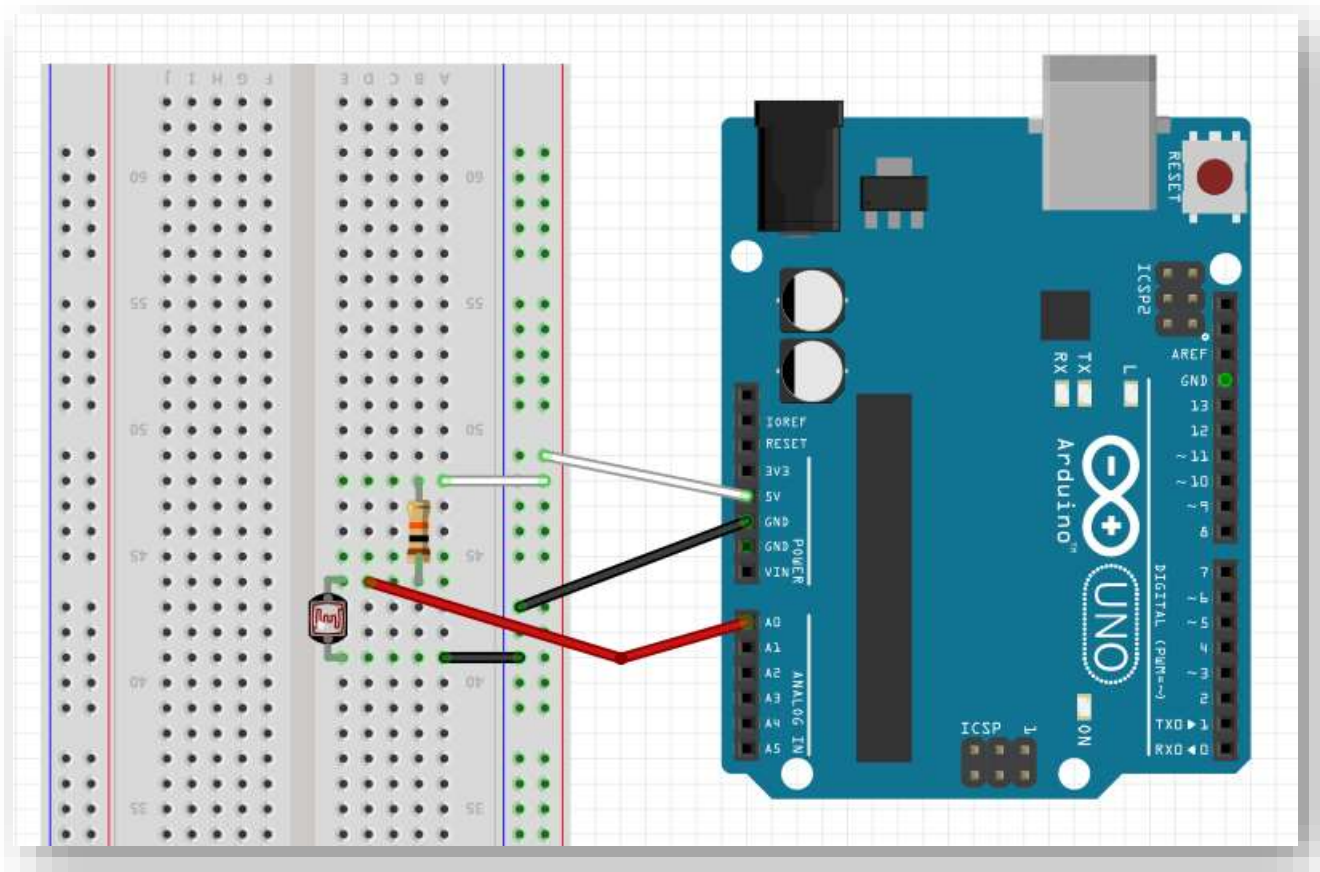
Parts : Temperature sensor (TMP36)
A0 : analog signal input

DIY3 Servo





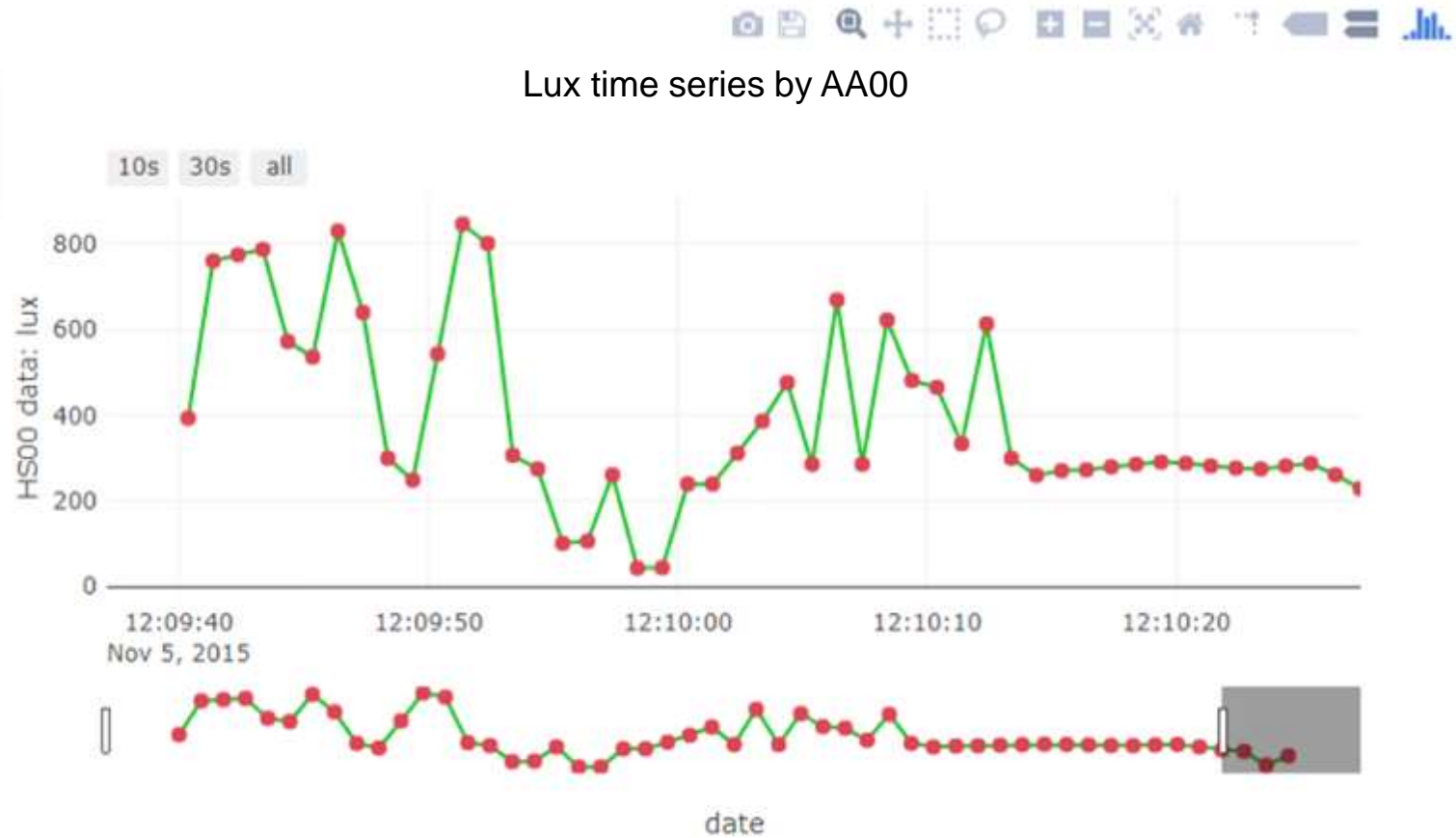
Luminosity sensor : photo cell LDR



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

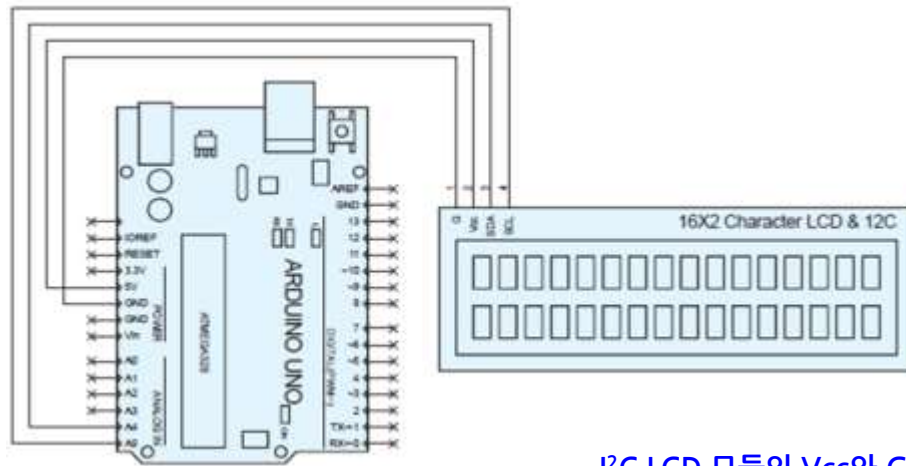
Arduino data + plotly

Time series by AA00





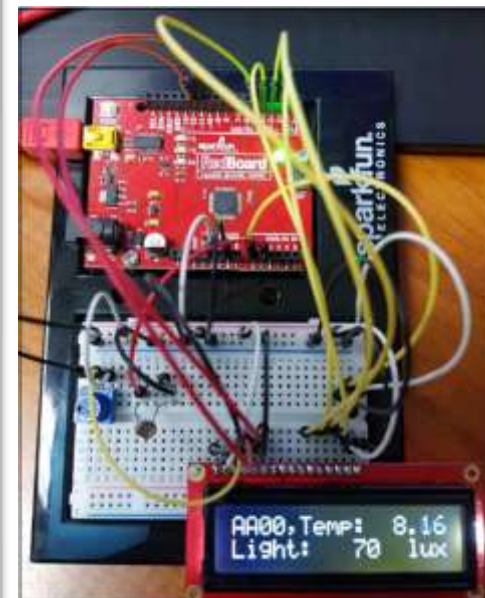
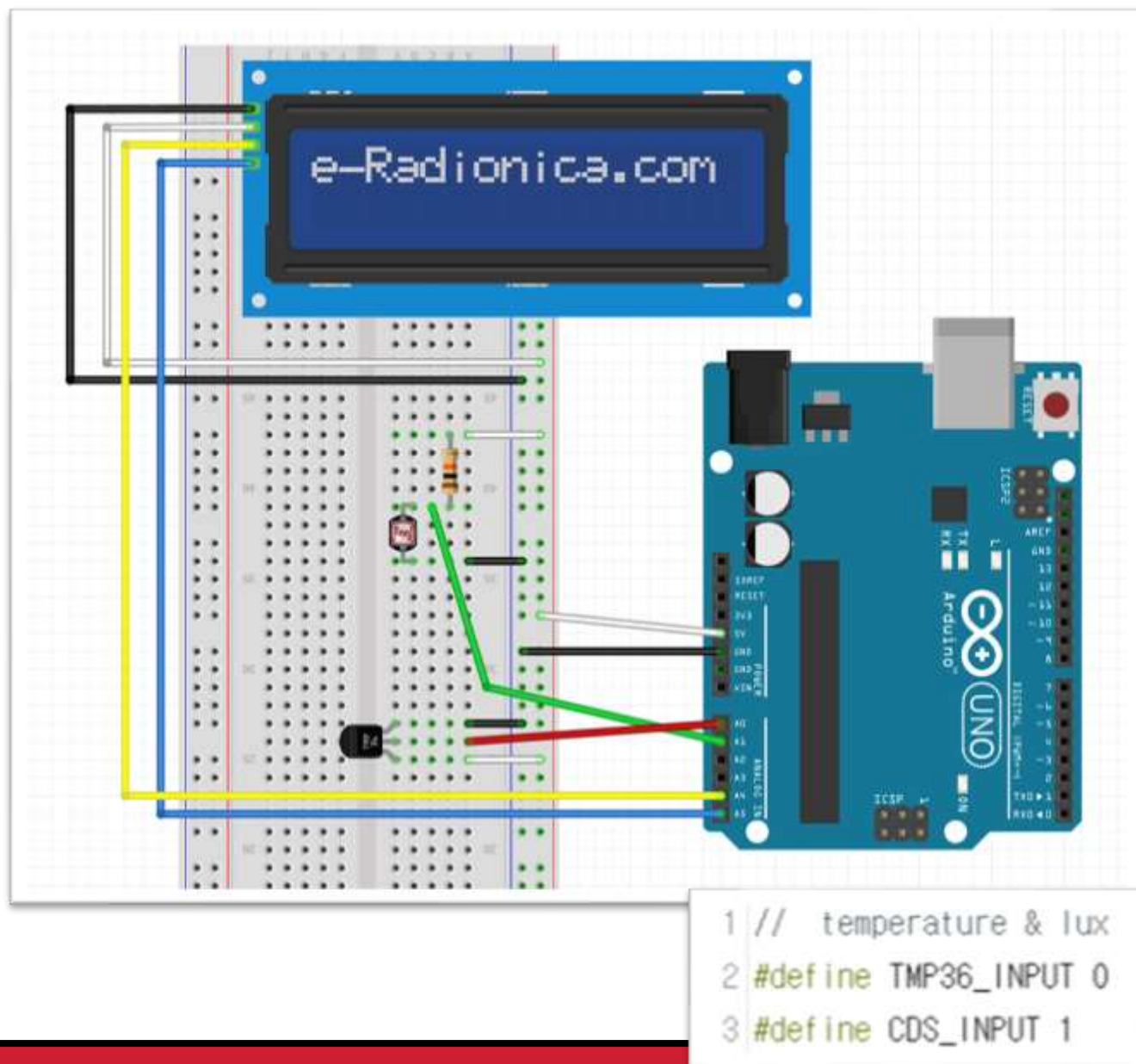
I2C LCD module



I²C LCD 모듈의 Vcc와 GND를 Arduino의 5V와 GND에 연결한다.
SDA는 A4에, SCL은 A5에 연결한다.

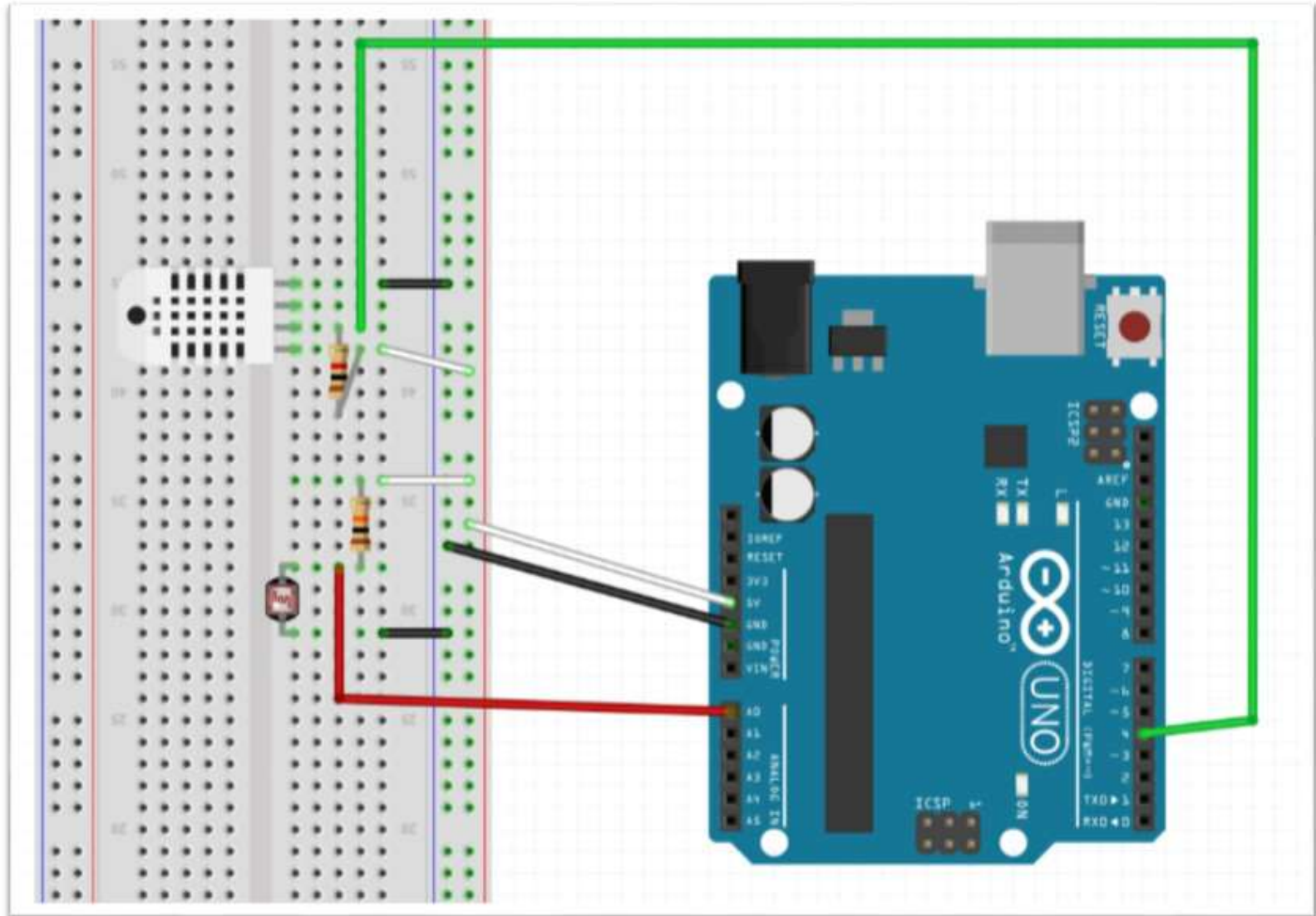


TMP36 + CdS + LCD : circuit





0.A7 DHT22 & CdS

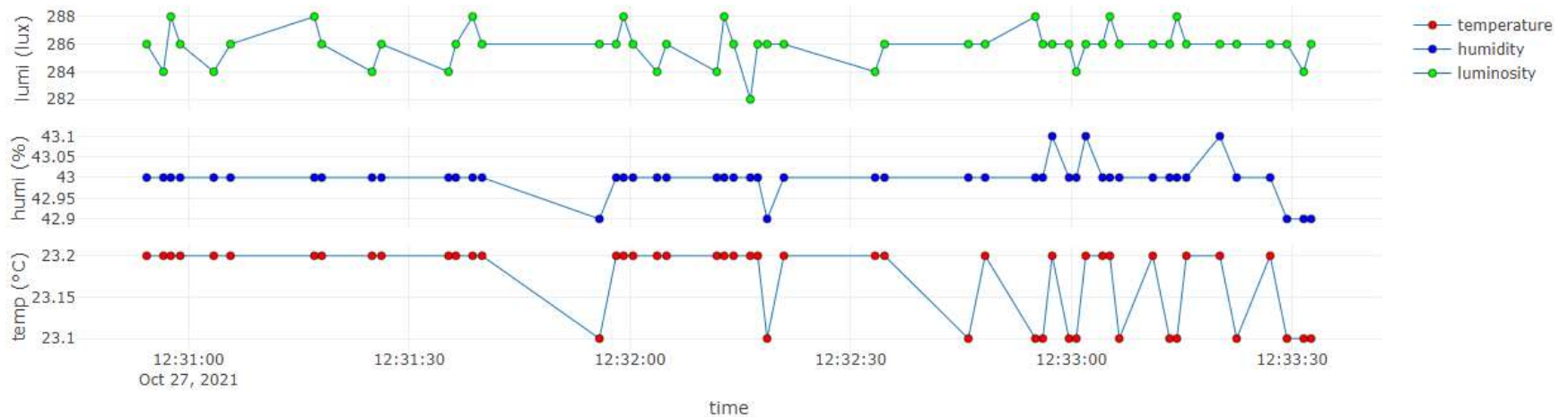


WEB client : client_cds_dht22.html

Real-time Weather Station from sensors

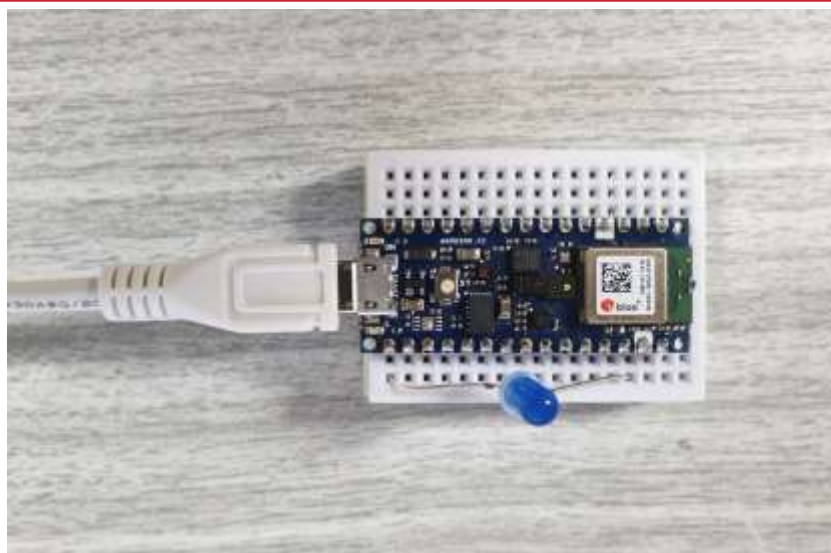


on Time: 2021-10-27 12:33:32.600



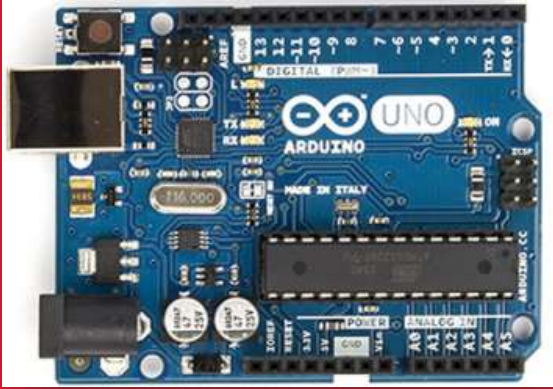


0.A8 Nano33 BLE Sensor



Real-time Weather Station from nano 33 BLE sensors

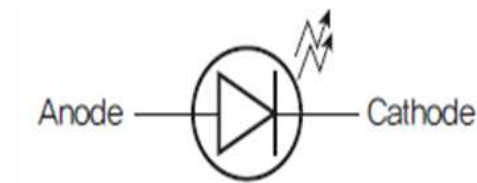




LED

LED (Light Emitting Diode)

- ✓ 전기 신호를 빛으로 출력하는 반도체 소자
- ✓ 고효율, 반영구적 수명
- ✓ 가정용 실내등, 산업용 특수등, 자동차용 전조등 및 실내등에 사용





A2.1.1 Blink [digitalWrite()]

sketch01_start | 아두이노 1.8.5

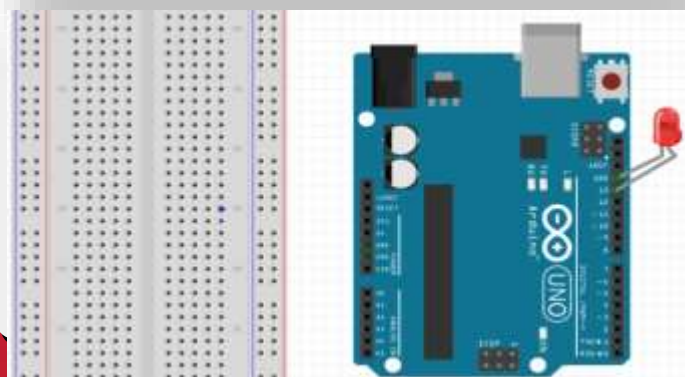
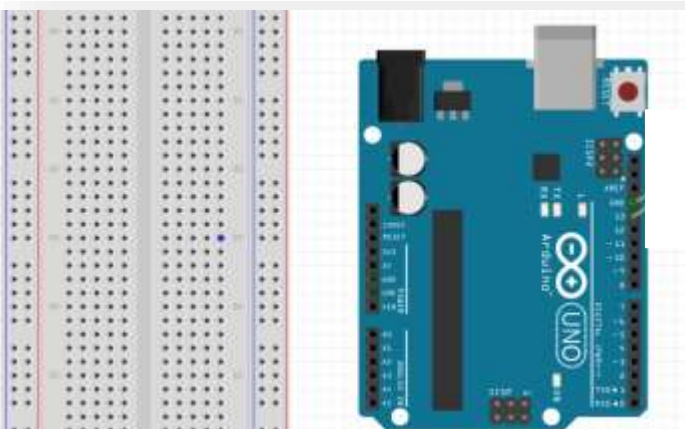
파일 편집 스케치 툴 도움말

새 파일 Ctrl+N
열기... Ctrl+O
최근 파일 열기
스케치북
예제
닫기 Ctrl+W
저장 Ctrl+S
다른 이름으로 저장... Ctrl+Shift+S
레이아웃 설정 Ctrl+Shift+P
인쇄 Ctrl+P
환경설정 Ctrl+Comma

내장된 예제

01. Basics
02. Digital
03. Analog
04. Communication
05. Control
06. Sensors

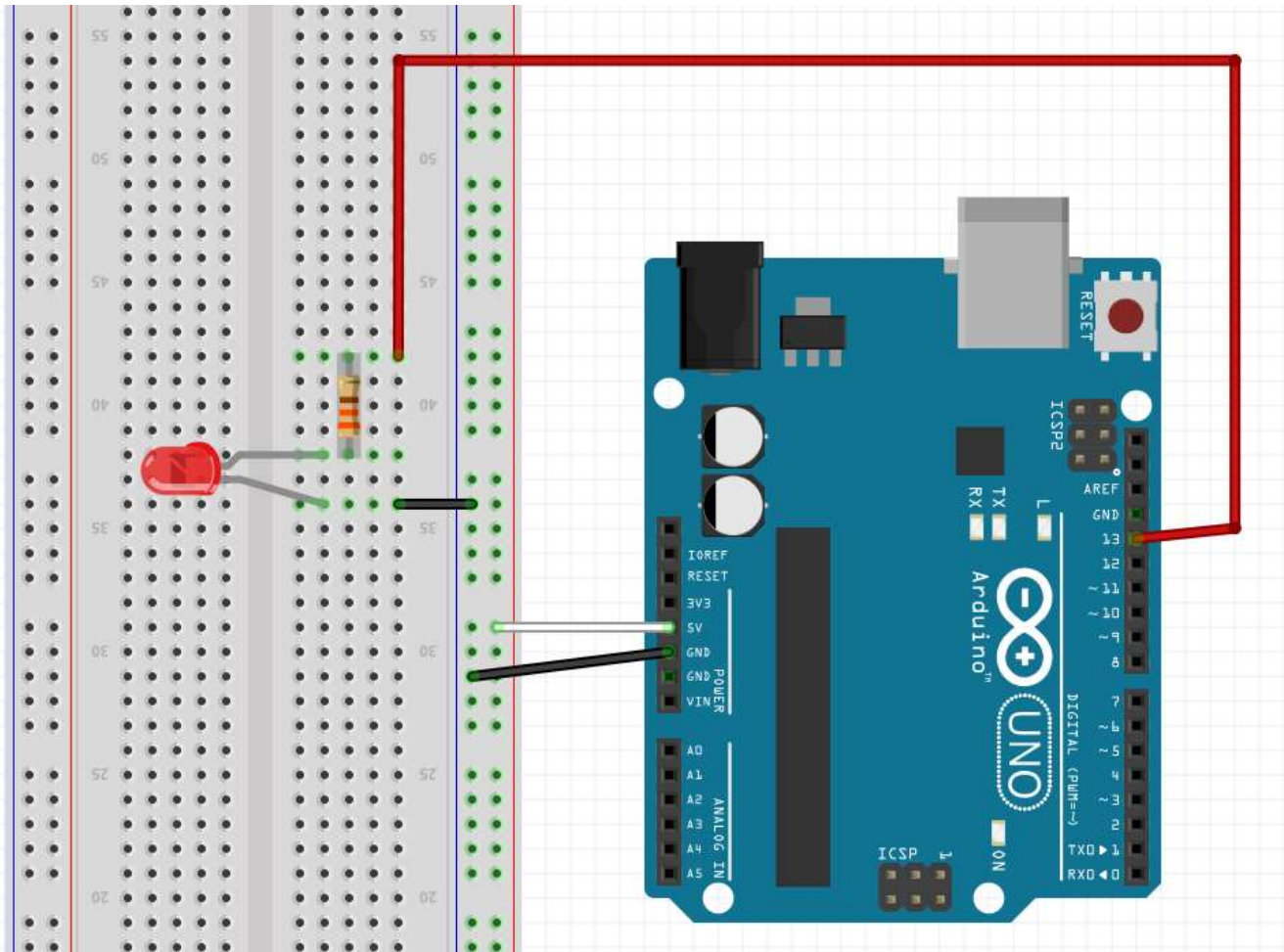
AnalogReadSerial
BareMinimum
Blink
DigitalReadSerial
Fade
ReadAnalogVoltage



Blink

```
1 /*  
2  Blink  
3  Turns an LED on for one second, then off for one second, repeatedly.  
4  */  
5  
6 // the setup function runs once when you press reset or power the board  
7 void setup() {  
8   // initialize digital pin LED_BUILTIN as an output.  
9   pinMode(LED_BUILTIN, OUTPUT);  
10 }  
11  
12 // the loop function runs over and over again forever  
13 void loop() {  
14   digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)  
15   delay(1000); // wait for a second  
16   digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW  
17   delay(1000); // wait for a second  
18 }
```

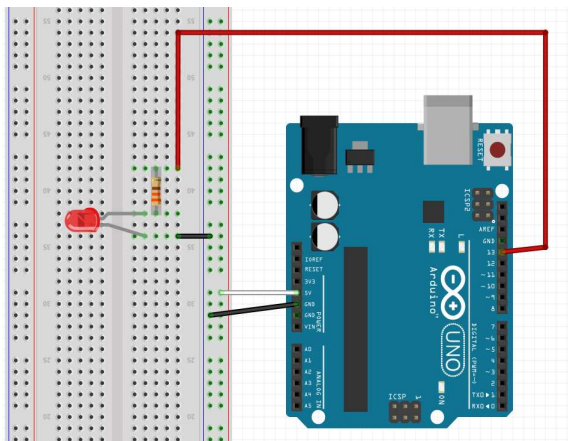
A2.1.2 blink circuit



**Connect LED to D13 & GND
with 330 Ω**



A2.1.3 blink [modified your code, save it]



**Connect LED to
D13 & GND
with 330 Ω**

aa00

이름

sketch01_blink

libraries

sketch01_blink

sketch01_start

sketch01_blink\$

```
1 /*
2  Blink by AA00
3  Turns an LED on for one second, then off for one second, repeatedly.
4  */
5  int pinNum = 13; // D13
6
7  // the setup function runs once when you press reset or power the board
8  void setup() {
9    // initialize digital pin 13 as an output.
10    pinMode(pinNum, OUTPUT);
11  }
12
13  // the loop function runs over and over again forever
14  void loop() {
15    digitalWrite(pinNum, HIGH); // turn the LED on (HIGH is the voltage level)
16    delay(1000);                // wait for a second
17    digitalWrite(pinNum, LOW);  // turn the LED off by making the voltage LOW
18    delay(1000);                // wait for a second
19  }
```



A2.2.1 LED control – 밝기 조절

밝기 조절 : 디밍 (Dimming)

- ✓ LED에 입력되는 전력은 **PWM (Pulse Width Modulation)**을 이용하여 조절.
- ✓ PWM : 고속의 스위칭으로 High와 Low 신호의 비율을 조절하여
LED의 밝기, 모터의 회전 등을 조절하는 방법
- ✓ Arduino에서는 **analogWrite()** 명령어로 구현
- ✓ Arduino UNO의 경우 **3, 5, 6, 9, 10, 11 번 핀이 PWM을 지원**한다.



A2.2.2 LED control – 밝기 조절: PWM

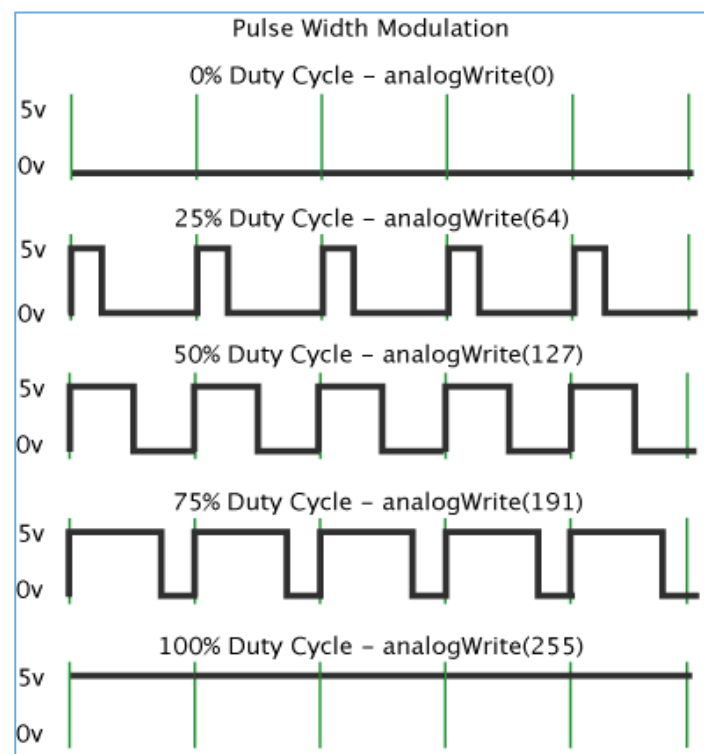
PWM (Pulse Width Modulation)

Using [analogWrite\(pin, pwm_value\)](#) function in fading an LED off and on. AnalogWrite uses [pulse width modulation \(PWM\)](#), turning a digital pin on and off very quickly with different ratio between on and off, to create a fading effect.

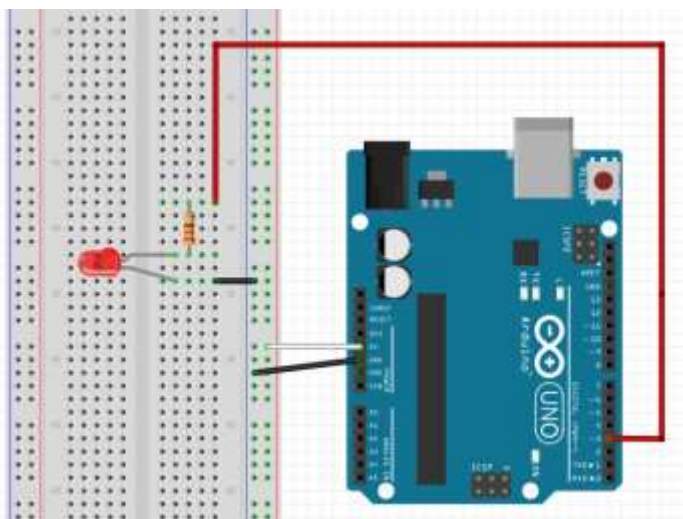
A call to [analogWrite\(\)](#) is on a scale of **0 - 255**, such that `analogWrite(255)` requests a 100% duty cycle (always on), and `analogWrite(127)` is a 50% duty cycle (on half the time)

PWM frequency = 500 Hz

<https://www.arduino.cc/en/Tutorial/PWM>



A2.2.3 LED control – 밝기 조절: PWM



▶ 스케치 구성

1. LED의 핀 번호를 pwm 핀으로 설정한다. **D3**
2. 아날로그 출력에는 `setup()`에서의 핀 설정이 필요 없다.
3. `loop()`에서 마구잡이 수를 하나 발생시켜서 `analogWrite()` 함수로 LED의 밝기를 0.01초 간격으로 반복해서 변화시킨다.



A2.2.4 LED control – 밝기 조절: PWM

▶ 사용 함수

- **analogWrite**(핀번호, 값)

정해진 핀에 아날로그 출력을 한다. '값'에는 0~255의 값을 넣는다.

- **random**(시작값, 종료값)

시작 값과 종료 값 사이의 정수를 마구잡이로 하나 만들어 반환한다.

- **pwmLed**(핀번호, 값)

정해진 PWM 출력 핀에 0~255의 pwm 값으로 아날로그 출력을 하는 사용자 정의 함수이다.



A2.2.5 LED control – 밝기 조절: code

▶ 아두이노 코드 : sketch02_pwm_led.ino

```
int pwm = 0;
int led = 3; // D3

void setup() {
  // 아날로그 출력에서 핀 모드 설정이 필요 없다.
}

void loop() {

  pwm = random(0,255);
  pwmLed(led , pwm);
}

void pwmLed(int led, int pwmValue) {

  analogWrite(led, pwmValue);
  delay(10);
}
```

실습 결과

LED의 밝기가

0.01초 간격으로 마구

잡이로 변하는 것을 확

인



신호 발생 및 모니터링

Serial monitor & plotter



A2.3 시리얼 통신 (serial comm.)

시리얼 통신

UART (Universal Asynchronous Receiver/Transmitter)

RS-232

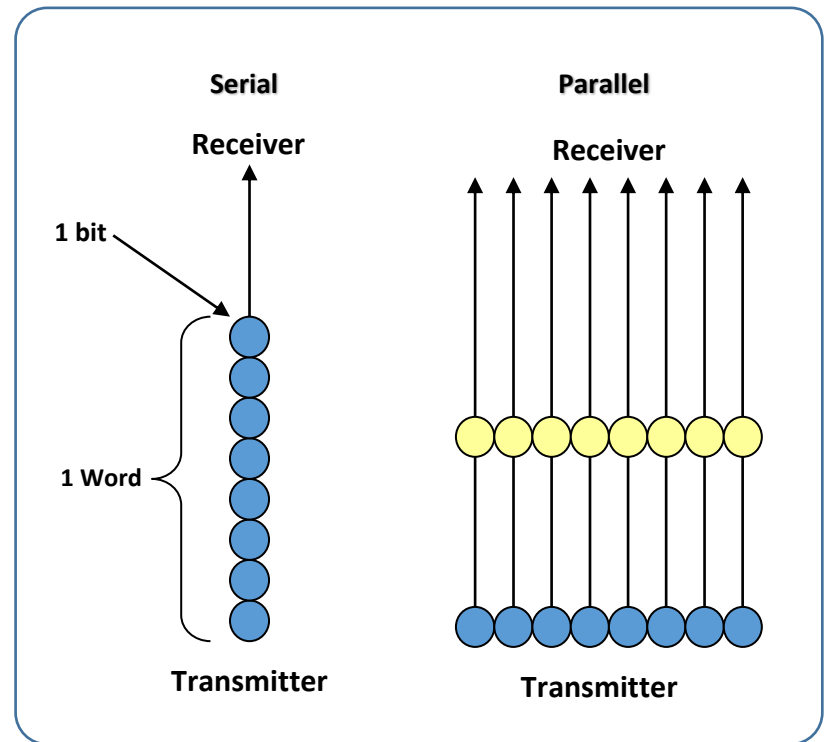
RS-422

RS-485

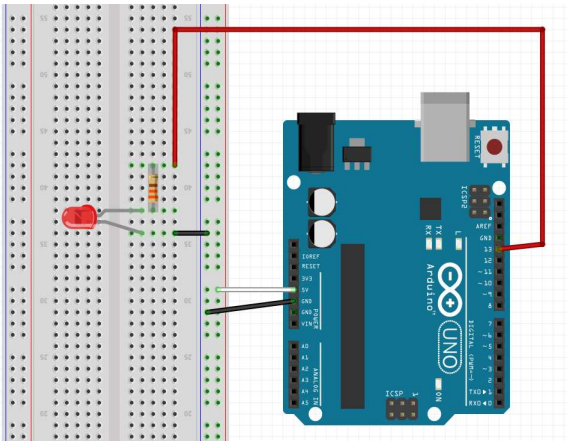
Arduino에서는 다음과 같은 목적으로 사용

Debugging : 프로그램의 오류를 수정하는 작업

데이터 통신 : Arduino와 컴퓨터 혹은 다른 장치와의 통신



A2.3.1 LED 밝기 조정 및 모니터링 - 스케치



▶ 스케치 구성

1. LED의 핀 번호를 pwm 핀으로 설정한다.
2. setup()에서 직렬 통신 속도를 9600 bps 로 설정하고 컴퓨터와 연결한다.
3. loop()에서 마구잡이 수를 하나 발생시켜서 analogWrite() 함수로 LED의 밝기를 반복해서 변화시키면서 직렬 통신으로 pwm 값을 전송한다.



A2.3.2 LED 밝기 조정 및 모니터링 - 함수

▶ 사용 함수

- **Serial.begin(전송속도)**

직렬 통신 포트를 컴퓨터와 연결한다. 전송속도는 bps (bits per sec)로 일반적으로 9600으로 설정한다. 19200, 57600, 115200 등의 값을 설정할 수 있다.

- **Serial.print(전송내용)**

괄호 안의 내용을 직렬 통신으로 전송한다. 따옴표로 구분된 부분은 텍스트를 직접 전송하고 따옴표 없이 변수를 써주면 변수의 값이 전송된다.

- **Serial.println(전송내용)**

‘Serial.print’와 같으나 전송 뒤 줄 바꿈을 한다.



A2.3.3 LED 밝기 조정 및 모니터링 – code

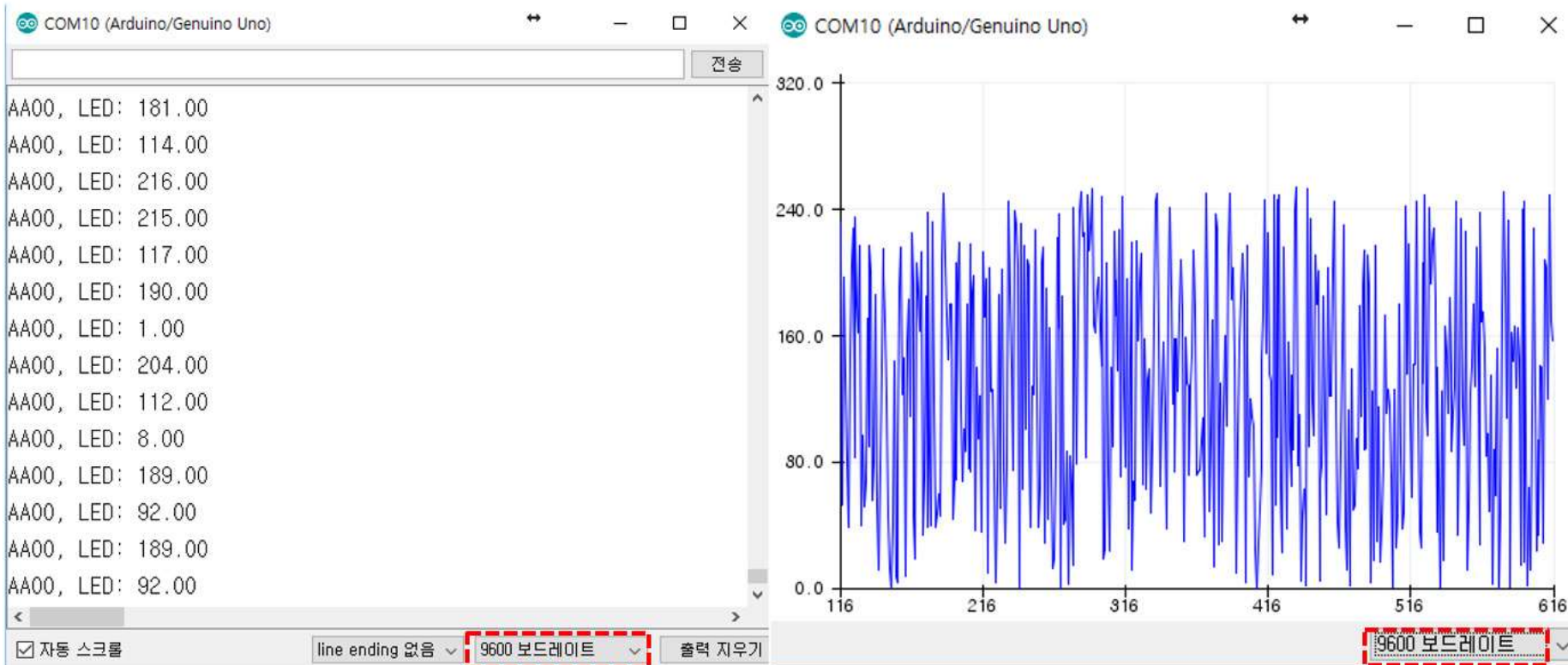
```
sketch03_pwm_led_serial
1 // sketch03_pwm_led_serial.ino
2 int pwm = 0;
3 int led = 3;
4
5 void setup() {
6   Serial.begin(9600);
7 }
8
9 void loop() {
10  // put your main code here:
11  pwm = random(0,255);
12  pwmLed(led , pwm);
13
14  Serial.print("AA00, LED: ");
15  Serial.println(pwm);
16  delay(10);
17 }
18
19 void pwmLed(int led, int pwmValue) {
20   analogWrite(led, pwmValue);
21   delay(10);
22 }
```



A2.3.4 LED 밝기 조정 및 모니터링 – 결과

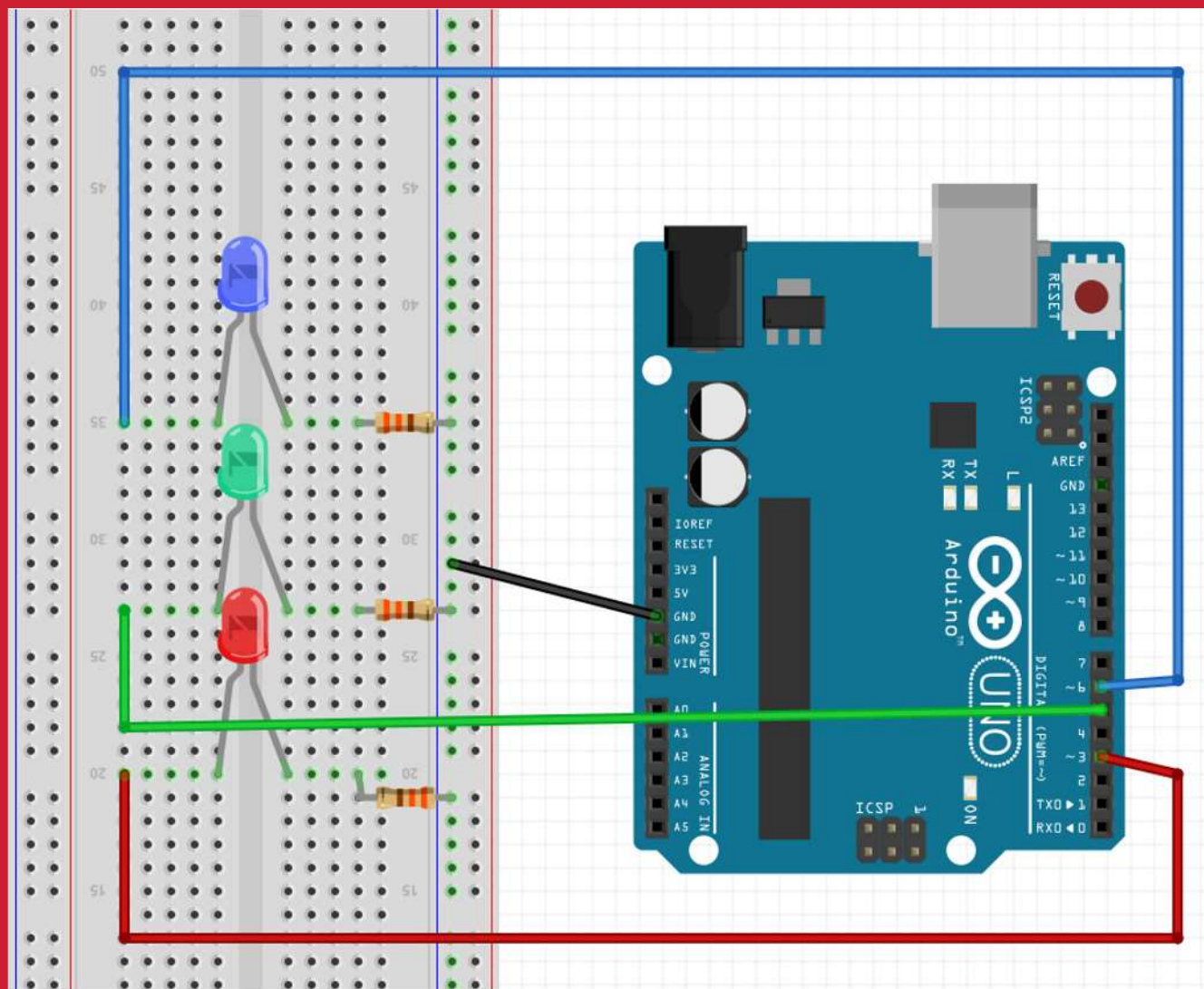
실습 결과

LED의 밝기가 0에서 255 단계로 마구잡이로 변하는 것을 확인할 수 있으며
직렬모니터와 직렬플로터로 pwm의 값의 변화를 모니터링 할 수 있다.

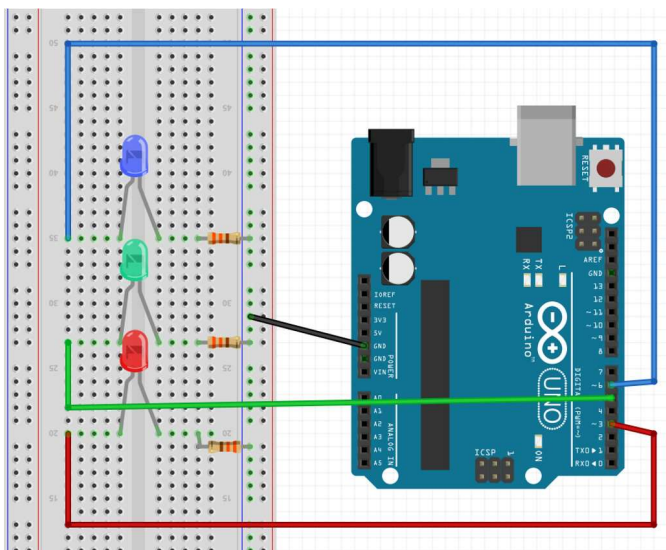




3 LED 모니터링



A2.4.1 3개의 LED 밝기 조정 및 모니터링 - 스케치



▶ 스케치 구성

1. 3 개의 LED의 핀 번호를 각각 다른 pwm 핀 (3, 5, 6)으로 설정한다.
2. setup()에서 직렬 통신 속도를 9600 bps 로 설정하고 컴퓨터와 연결한다.
3. loop()에서 마구잡이 수를 세 개 발생시켜서 analogWrite() 함수로 세 개의 LED의 밝기를 각각 반복해서 변화시킨다.
4. 직렬 통신으로 3 개의 pwm 값을 한 줄로 컴퓨터로 전송한다.

A2.4.2 3개의 LED 밝기 조정 및 모니터링 – code

sketch04_pwm_3_leds

```
1 // pwm_3_leds.ino
2 int pwm1 = 0;
3 int pwm2 = 0;
4 int pwm3 = 0;
5
6 int ledR = 3;
7 int ledG = 5;
8 int ledB = 6;
9
10 void setup() {
11
12   Serial.begin(9600);
13 }
```

```
15 void loop() {
16
17   pwm1 = random(0,255);
18   pwm2 = random(0,255);
19   pwm3 = random(0,255);
20   pwmLed(ledR, pwm1);
21   pwmLed(ledG, pwm2);
22   pwmLed(ledB, pwm3);
23
24   Serial.print("AA00, LED_R: ");
25   Serial.print(pwm1);
26   Serial.print(" , LED_G: ");
27   Serial.print(pwm2);
28   Serial.print(" , LED_B: ");
29   Serial.println(pwm3);
30   delay(10);
31 }
32
33 void pwmLed(int led, int pwmValue) {
34   analogWrite(led, pwmValue);
35   delay(10);
36 }
```



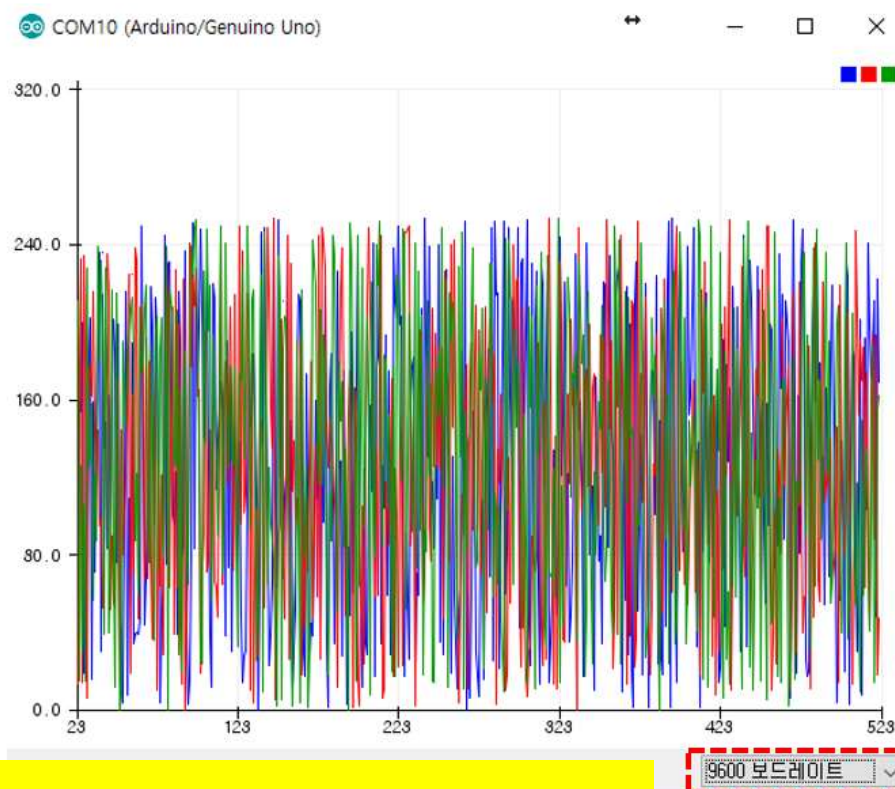
A2.4.3 3개의 LED 밝기 조정 및 모니터링 – 결과

실습 결과

세 개의 **LED**의 밝기가 각각 **0**에서 **255** 단계로 마구잡이로 변하는 것을 확인할 수 있다. 직렬모니터와 직렬플로터로 세 개의 **pwm**의 값의 변화를 모니터링 한다.

```
COM10 (Arduino/Genuino Uno)
전송
AA00, LED_R: 46 , LED_G: 190 , LED_B: 208
AA00, LED_R: 250 , LED_G: 209 , LED_B: 173
AA00, LED_R: 145 , LED_G: 180 , LED_B: 180
AA00, LED_R: 44 , LED_G: 67 , LED_B: 206
AA00, LED_R: 69 , LED_G: 192 , LED_B: 219
AA00, LED_R: 115 , LED_G: 68 , LED_B: 101
AA00, LED_R: 87 , LED_G: 180 , LED_B: 76
AA00, LED_R: 218 , LED_G: 74 , LED_B: 187
AA00, LED_R: 198 , LED_G: 37 , LED_B: 140
AA00, LED_R: 95 , LED_G: 122 , LED_B: 36
AA00, LED_R: 213 , LED_G: 172 , LED_B: 196
AA00, LED_R: 195 , LED_G: 10 , LED_B: 203
AA00, LED_R: 125 , LED_G: 126 , LED_B: 72
AA00, LED_R: 4 , LED_G: 64 , LED_B: 186
AA00, LED_R: 51 , LED_G: 121 , LED_B: 
```

☒ 자동 스크롤 line ending 없음 9600 보드레이트



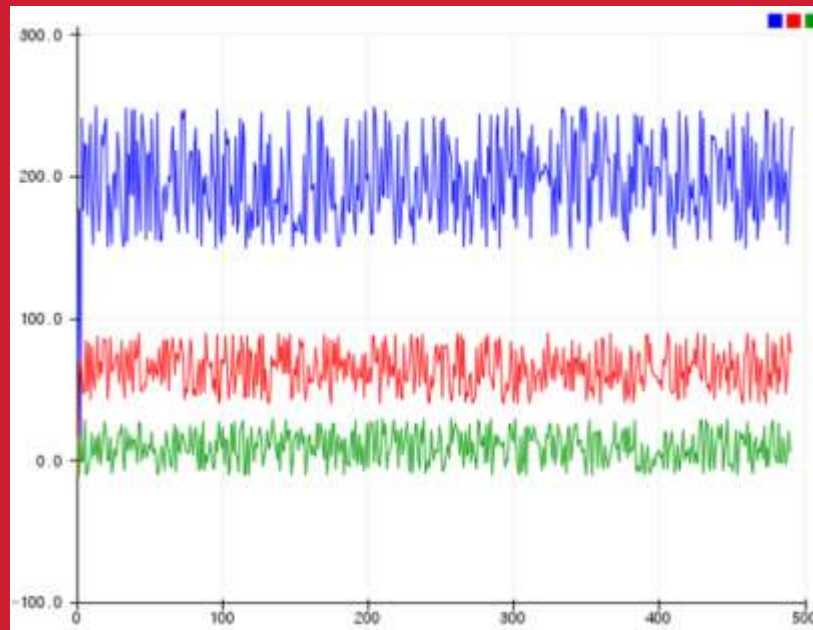
Save as

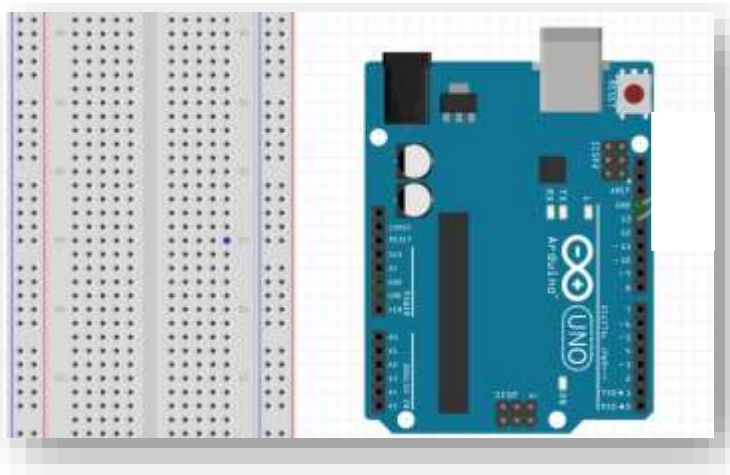
AAnn_multi_Monitoring.png



[DIY] Multi-signals

다중신호 시뮬레이션 및 모니터링





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 }
```

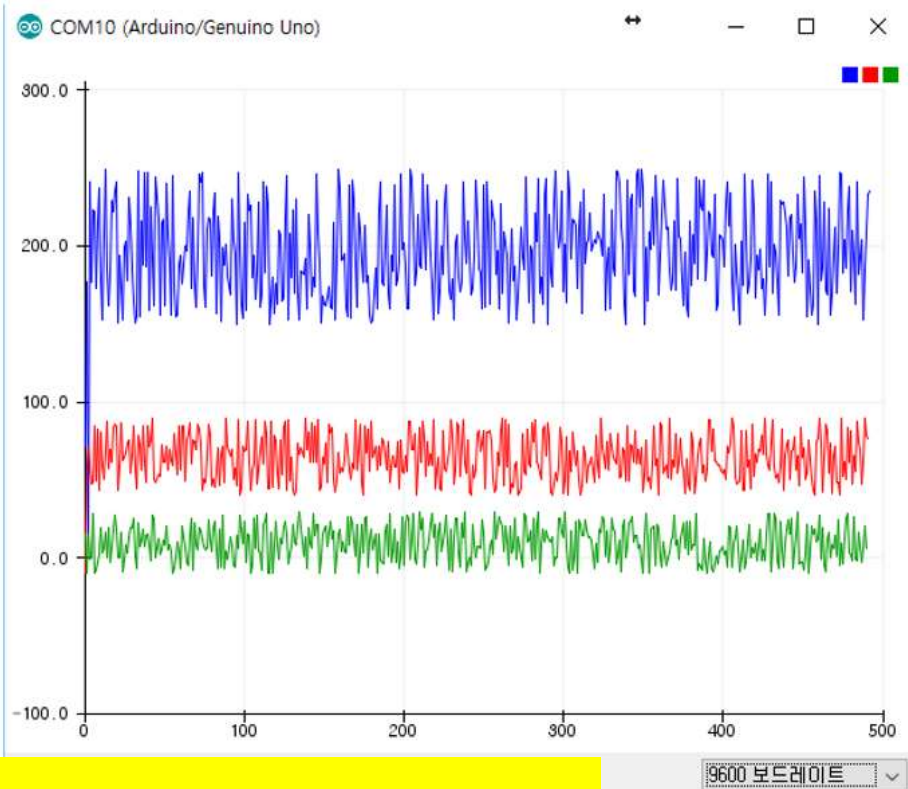


DIY - result

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

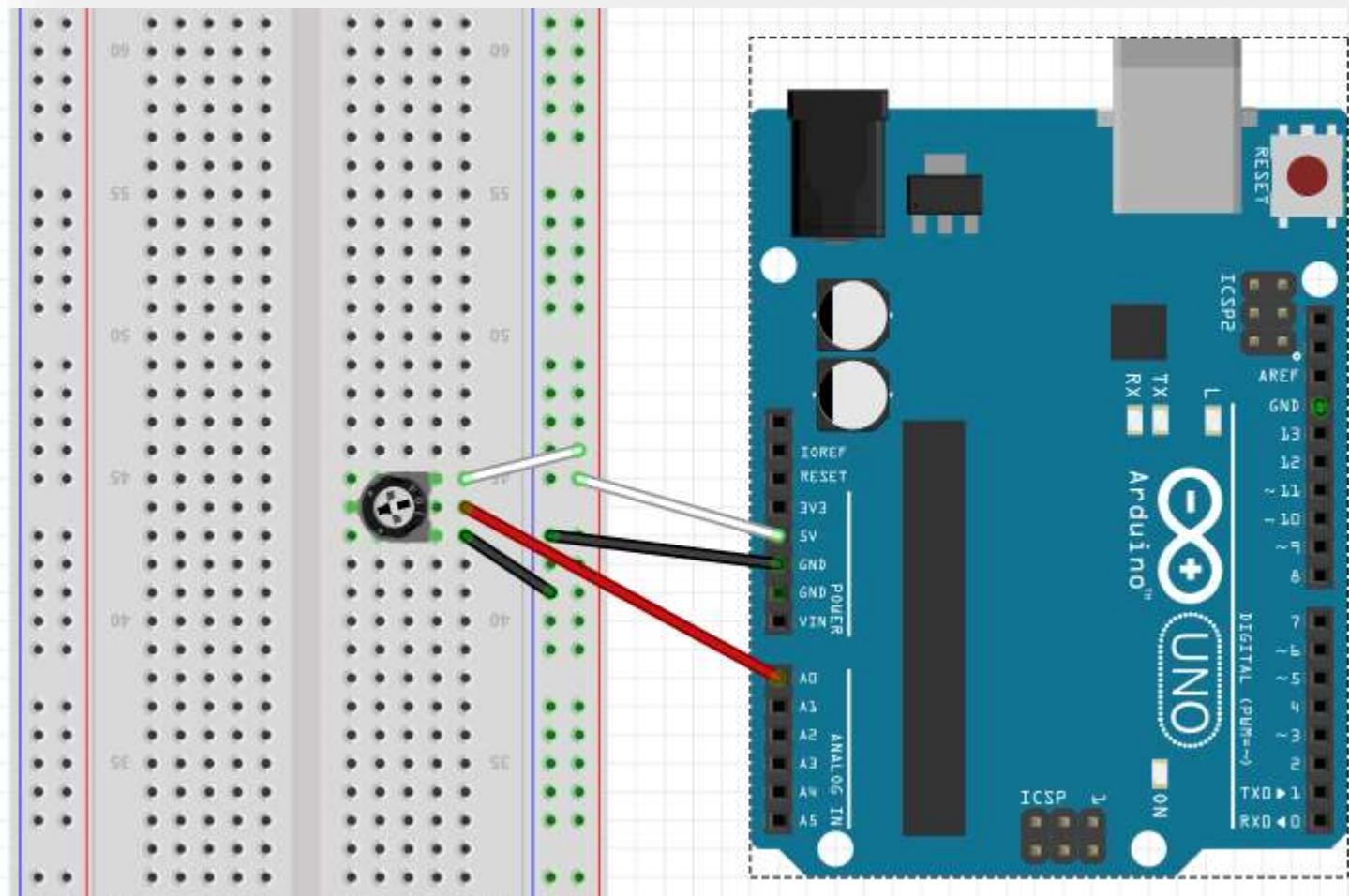
AAnn_multi_Signals.png

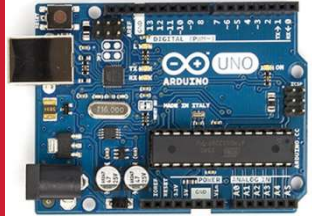


Analog Signal

A2.5.1 AnalogReadSerial (circuit)

Standard potentiometer (가변 저항기)





[Practice]

◆ [wk04]

- **Arduino basic circuits**
- **Complete your project**
- **Upload folder: aann-rpt04**
- **Use repo “aann” in github**

◆ [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**

● 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
- ✓ <https://www.anaconda.com/> Anaconda
- ✓ <http://www.github.com> GitHub
- ✓ <https://colab.research.google.com/> Colab