

INDEX



마이크로컨트롤러



아두이노



RGB LED



부저 모듈



DC 모터



진동 감지 센서



시리얼통신



Timer함수

아 두 이 노 스 케 치 단 계 별 코딩 교육자료

네오 캐논



아두이노 스케치 교육 과정에서 네오아두보드의 나노보드에 새로운 아두이노 스케치를 업로드 하게 되면 출고 시 업로드 되어 있던 'neo_cannon' 스케치가 지워지게 되므로 교육 과정이 끝난 이후에 반드시 'neo_cannon' 스케치를 다시 업로드 해주어야 네오캐논이 정상 동작하게 됩니다. 스케치는 홈페이지 교육자료 또는 블로그에서 다운로드 받으시기 바랍니다.

모든 스케치 업로드 과정에서는 블루투스 모듈을 분리한 후 업로드를 진행 해주시기 바랍니다.



마이크로컨트롤러

MCU, Micro Controller Unit

제어 장치 제작을 목적으로 중앙 처리 장치에 입출력 및 메모리 장치를 포함하여
하나의 칩으로 구현한 **마이크로 프로세서의 일종**

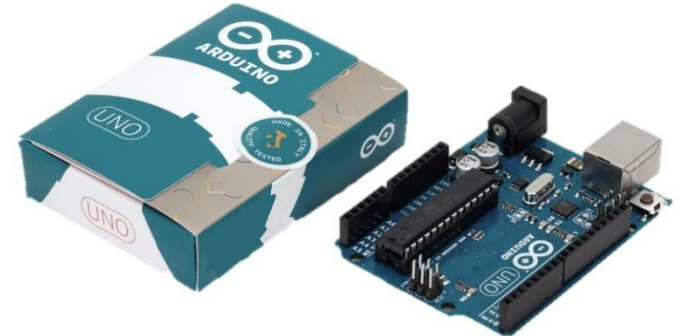


마이크로컨트롤러

MCU, Micro Controller Unit

MCU: 연산 + 주변 장치 제어

MPU: 연산 위주 작업



MCU 예시. 아두이노 우노



MPU 예시. 라즈베리파이



아두이노
Arduino

마이크로 컨트롤러 보드와 소프트웨어 개발환경까지 함께 이르는 말

마이크로 컨트롤러
MCU

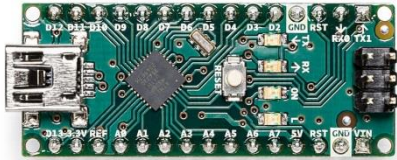
오픈 소스

하드웨어

소프트웨어



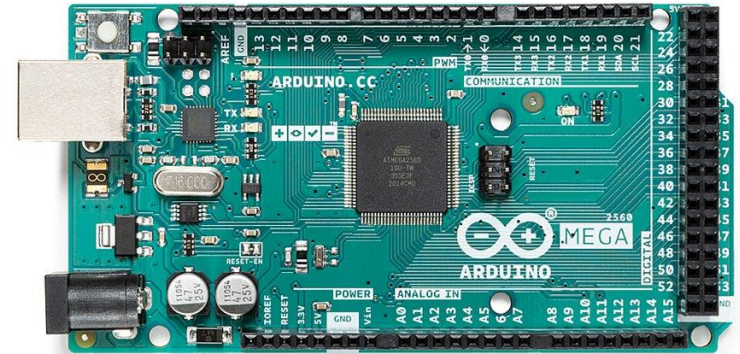
아두이노 Arduino



아두이노 나노



아두이노 우노



아두이노 메가

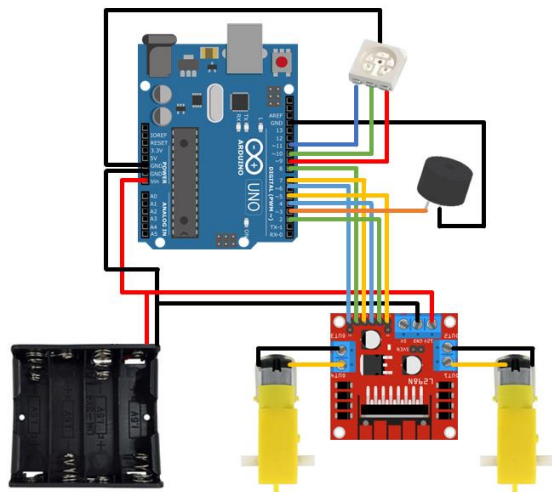
이 외에도 아두이노 프로, 아두이노 레오나르도, 아두이노 듀에 등 다양하게 존재하며
사용 목적에 맞게 선택하여 작업하면 됩니다.

우리가 사용할 네오아두보드는 아두이노 나노 보드를 사용하기 편하게 변형하여 만든 보드입니다.



네오아두보드 NeoArduBoard

기성품 아두이노 우노 사용



네오아두보드 + 나노 사용

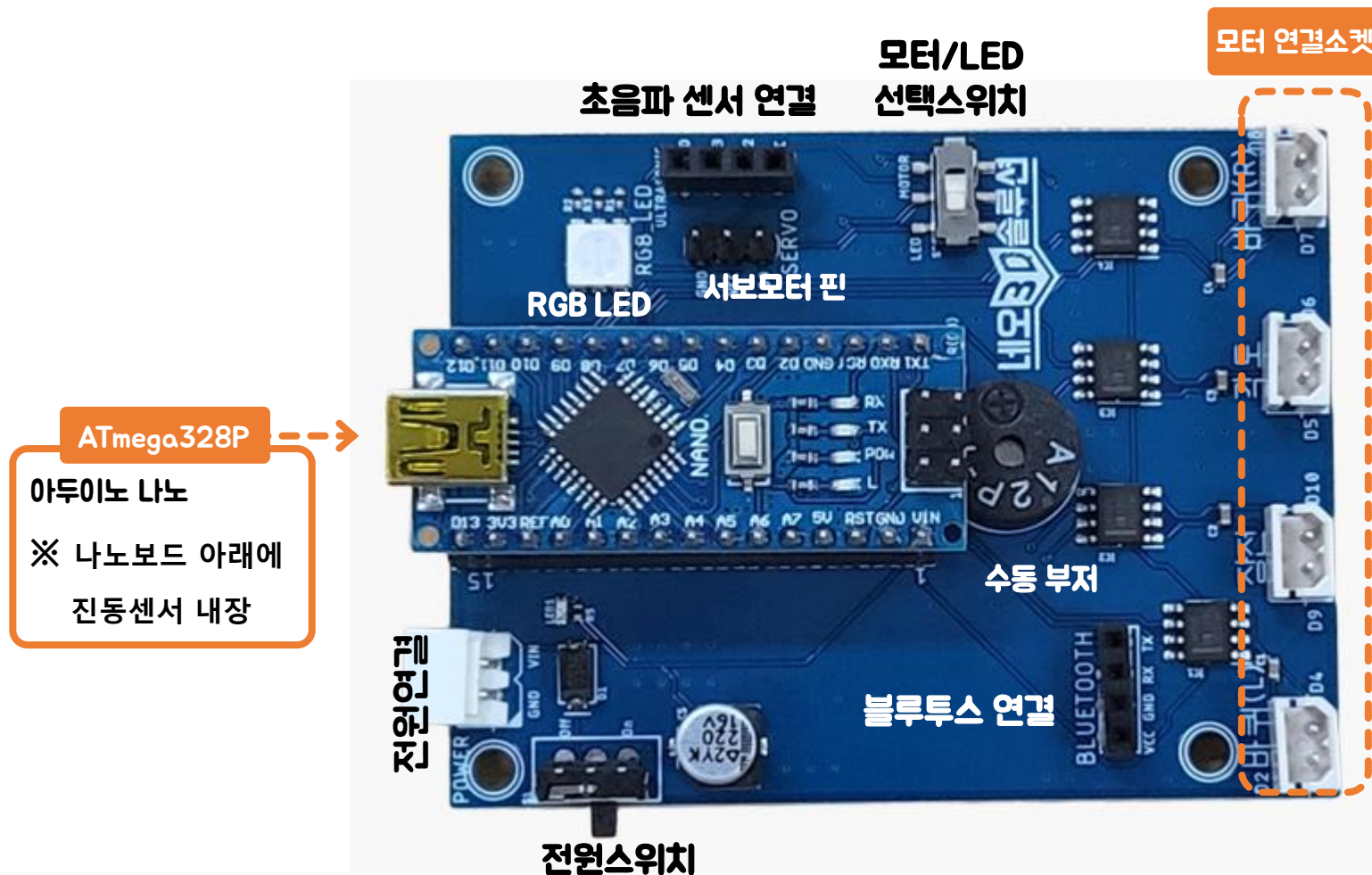


기성품 아두이노 우노와 네오아두보드를 사용하여 RGB LED / 부저 / 모터 2개를
제어하고자 할때의 비교 => 전선 사용 최소화



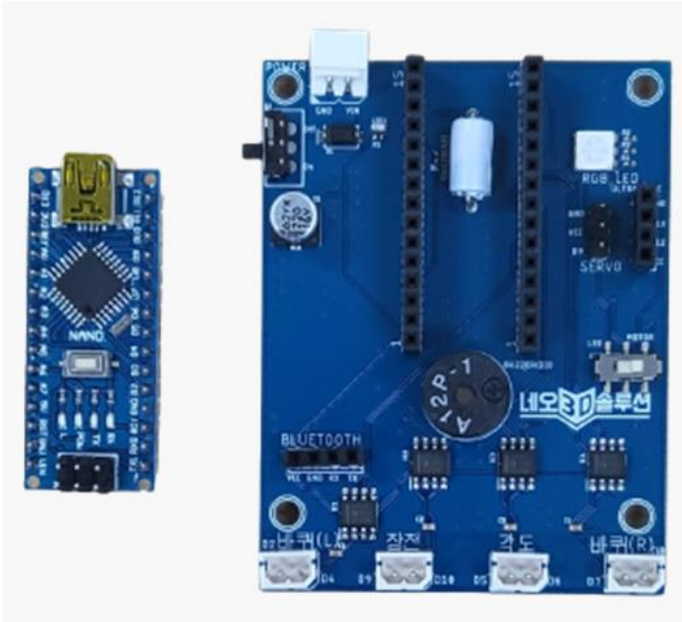
네오아두보드 NeoArduBoard

[네오캐논 제품에는 사진과 같은 네오아두보드가 적용되어 있습니다.]





네오아두보드 NeoArduBoard



Digital 입출력: D0 ~ D13

Analog 입력: A0 ~ A7

RGB LED 내장 (D9: 빨강, D10: 초록, D11: 파랑)

모터 드라이버(L9110)

- (D2, D4), (D5, D6), (D7, D8), (D9, D10)

서보 모터 전용 헤더핀 (D9)

초음파센서 전용 소켓 (D12, D13)

부저 내장 (D3) / 진동센서 내장 (A0)

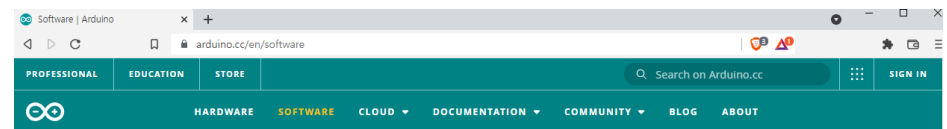
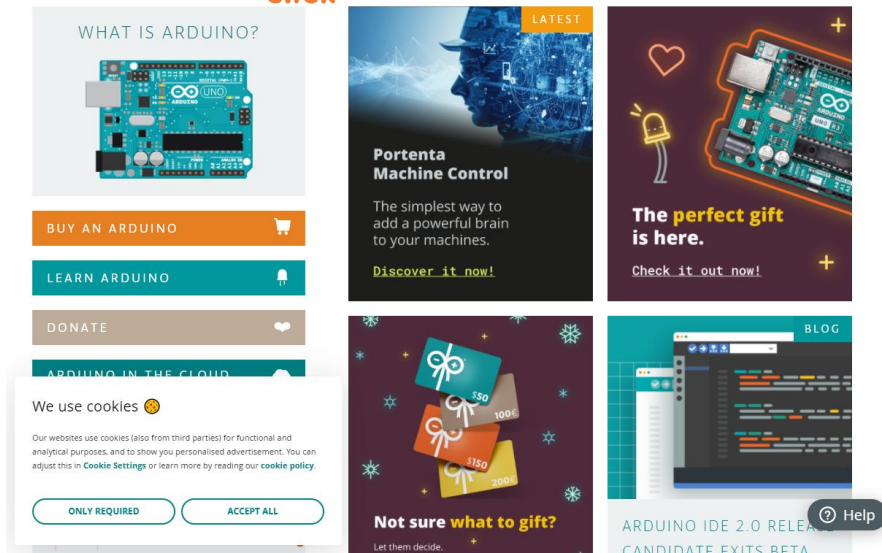
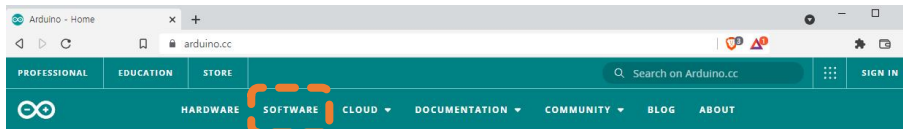
블루투스 전용 소켓 (D0, D1)

(블루투스 모듈의 TX / RX 핀 회로가 아두이노의 D0 / D1 핀으로
연결되어 있기 때문에 아두이노에 새로운 스케치를
업로드 하려면 블루투스 모듈을 분리하시기 바랍니다)



아두이노 IDE 설치 Install Arduino IDE

<https://arduino.cc>



설명에는 Windows Win7 and newer로 진행하겠습니다.



아두이노 IDE 설치 Install Arduino IDE

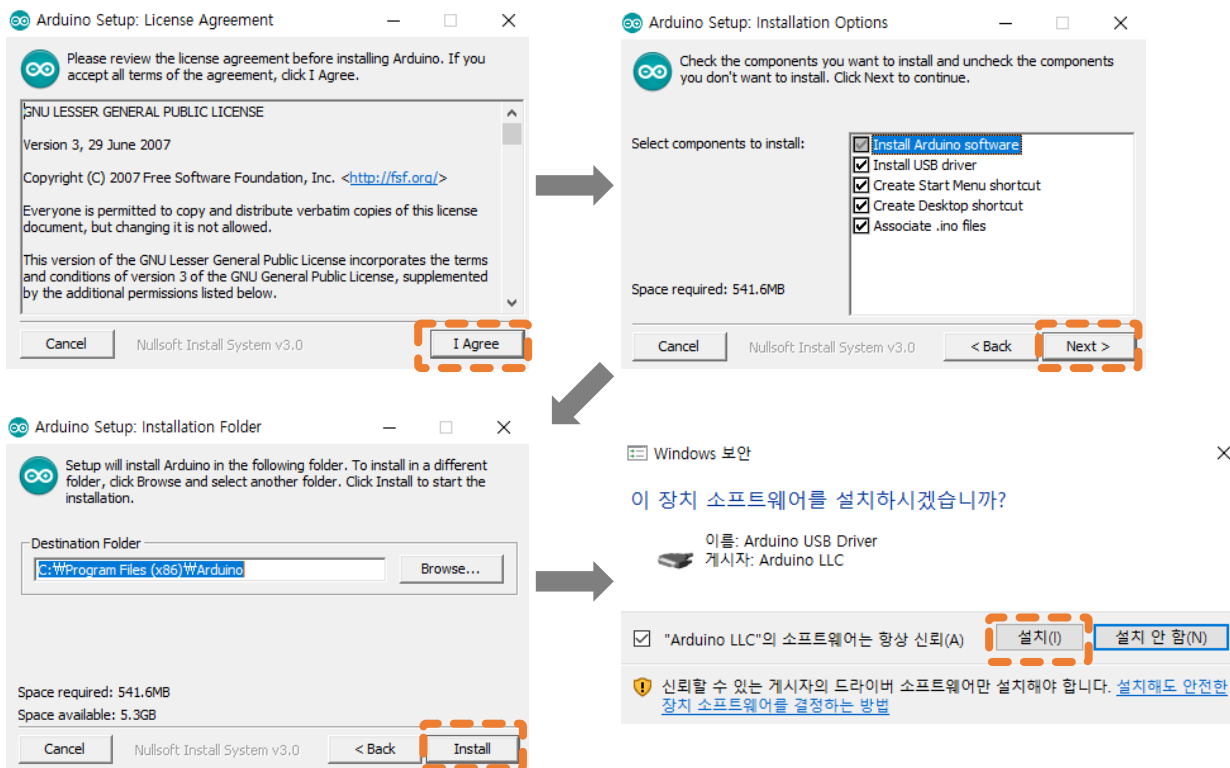


내용을 동일하게 실행하면
바탕화면에 왼쪽과 같은 아이콘 생성 됩니다.
해당 아이콘을 실행하면 아두이노 IDE를 실행할 수 있습니다.

클릭시 해당 그림처럼 나오게 됩니다.



다운 받아진 파일을 실행하고 보안경고가 나오면 '확인'



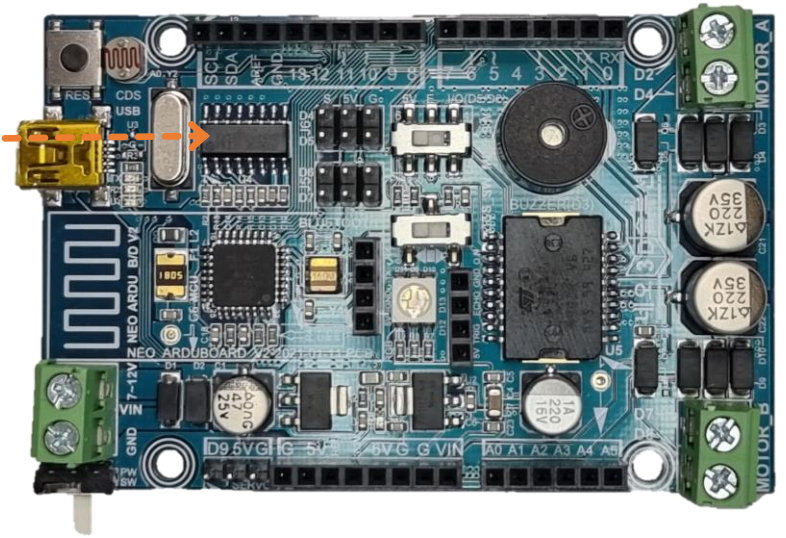


드라이버 설치 Install Driver

우리가 사용하는 드라이버 칩은 CH340 Chip이기에
해당 드라이버를 설치해 주도록 하겠습니다.
다른 드라이버 칩이라면 다른 드라이버를 설치하셔야 합니다.

<https://blog.naver.com/neo3ds/222666062937>

CH340 Chip



해당 링크에서
CH340 알집 파일을 제공하고 있습니다. (제공하는 파일은 윈도우용)
해당 파일을 다운받아 주세요.

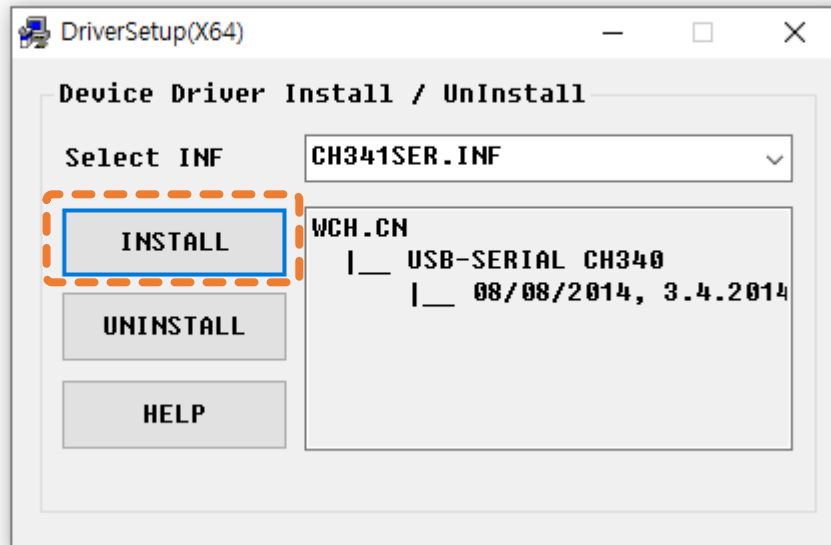
다운이 안되신다면 <https://sparks.gogo.co.nz/ch340.html> 해당 사이트를
이용 해주셔도 됩니다.



드라이버 설치 Install Driver

압축 파일을 다운 받으셨다면, 압축을 풀어주세요.

CH34x_Install_Windows_v3_4.EXE 2017-01-24 오후 1:17 응용 프로그램 238KB



압축 푸신 파일을 실행하게 되시면
하나의 프로그램이 실행되게 됩니다.
INSTALL을 클릭하여 주세요.

만약 Fail이 뜨게 된다면 UNINSTALL을 누르고
다시 INSTALL을 실행해주세요.

INSTALL이 성공했다면 이 창을 꺼주셔도 됩니다.



Comport 확인 Check Comport

장치와 연결이 되었다면 Comport 번호를 알아야 시리얼 통신을 할 수 있습니다.



+



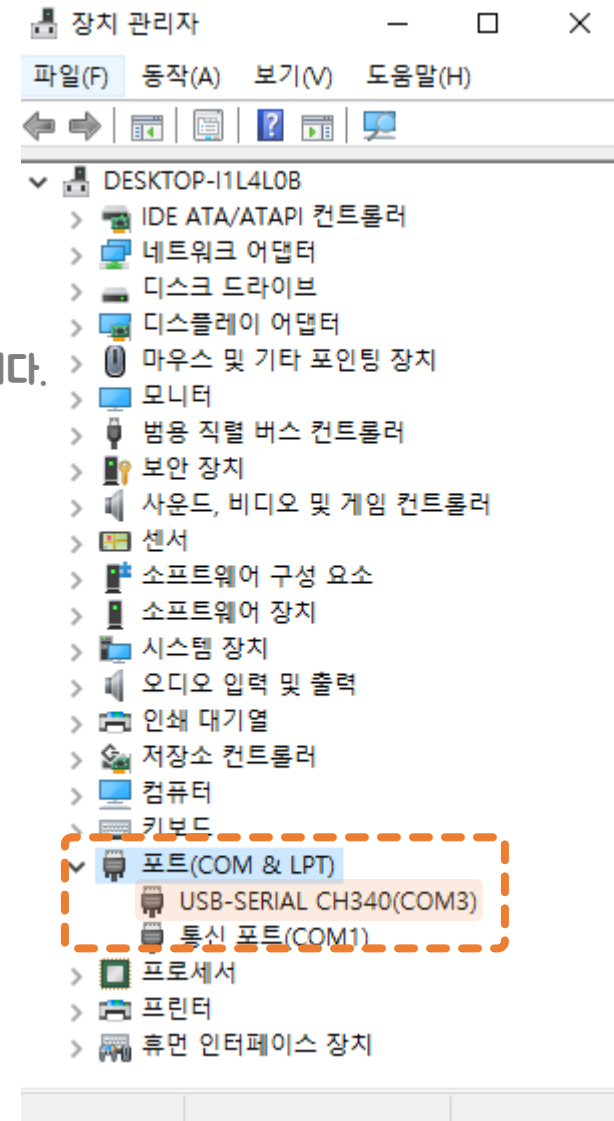
키를 동시에 누른 후에 m 키를 누르면 **장치 관리자** 창이 나오게 됩니다.



장치 관리자
제어판

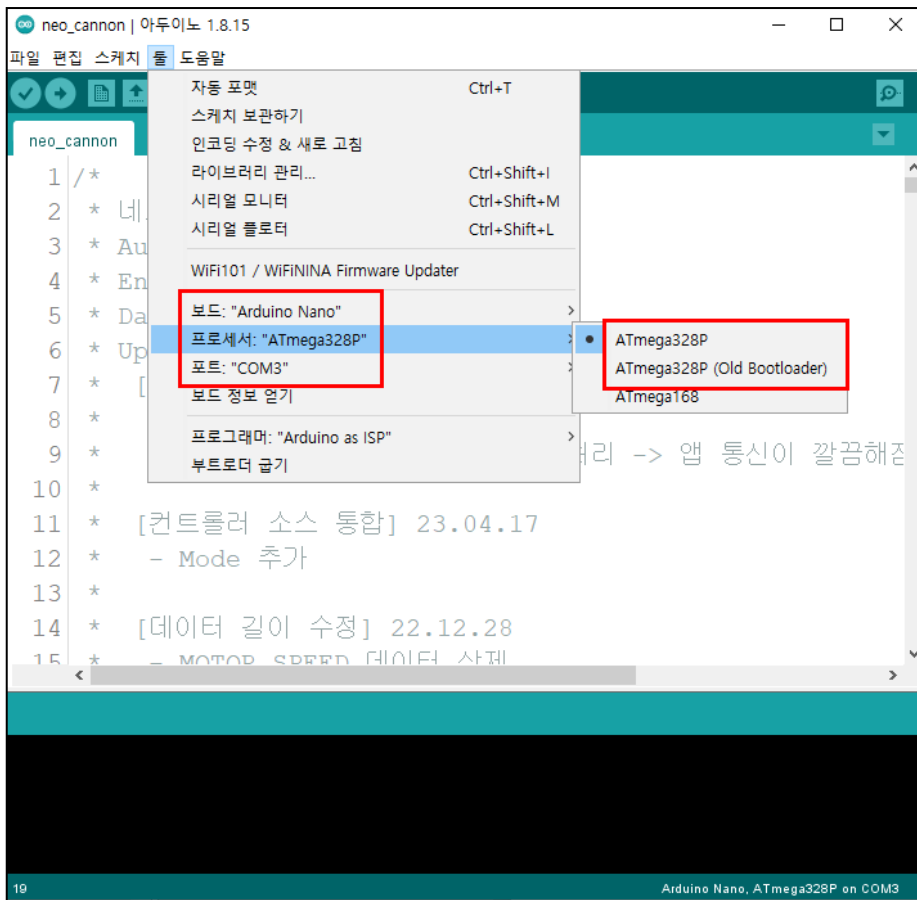
포트 내용 중에 **USB-SERIAL CH340(COM번호)**

여기에 COM번호를 통해서 시리얼 통신을 진행할 것입니다.





IDE 세팅 Setting IDE



아두이노 나노를 사용할 준비가 되었는지 확인하기 위해
세팅을 확인하면서 안되어 있다면 변경해주도록 하겠습니다.
내용은 아래와 같습니다.

메뉴 부분에 '툴'

→ 보드 Arduino Nano

→ 프로세서 : Atmega328p 또는 Atmega328p (Old Bootloader)

→ 포트: 연결할 컴포트

프로세서는 Atmega328p 를 선택하여 업로드 해보도 업로드가 되지 않으면
Atmega328p (Old Bootloader) 를 선택하여 업로드 하기 바랍니다.

연결할 컴포트는 장치 관리자를 통해 알아낸
USB-SERIAL CH340의 Com숫자 번호 입니다.



IDE 세팅

Setting IDE

환경설정



설정 네트워크

스케치북 위치:

C:\Users\Wjune\Documents\Arduino\

찾아보기

에디터 언어:

System Default

(아두이노를 재시작해야 함)

에디터 글꼴 크기:

12

Interface scale:

☒ 자동

100%

(아두이노를 재시작해야 함)

테마:

디폴트 테마

(아두이노를 재시작해야 함)

다음 동작중 자세한 출력 보이기:

☐ 컴파일

☐ 업로드

컴파일러 경고:

None

☐ 줄 번호 표시

☐ 코드 폴딩 사용하기

☒ 업로드 후 코드 확인하기

☐ 외부 에디터 사용

☒ 시작시 업데이트 확인

☒ 검증 또는 업로드 할 때 저장하기

☐ Use accessibility features

추가적인 보드 매니저 URLs



추가적인 환경 설정은 파일에서 직접 편집할 수 있습니다

C:\Users\Wjune\AppData\Local\Arduino15\preferences.txt

(아두이노가 실행되지 않는 경우에만 수정 가능)

확인

취소

메뉴 '파일' → 환경설정

환경설정에 들어가서 글꼴 크기를 바꾼다거나,

코드의 줄 번호를 볼 수 있게 설정한다거나

환경을 변경할 수 있습니다.



발광 다이오드

LED, Light-Emitting Diode



LED + / - 구분하는 방법

1. 다리 길이 구별법

길이가 긴 쪽 +
짧은 쪽

2. 금속판 폭 구별법

폭이 좁은 곳 +
폭이 넓은 곳 -

금속판 폭을 통해 구별하는 것이 확실한 방법입니다.

우리가 사용하고자 하는 LED는 RGB LED SMD Type입니다.



발광 다이오드

LED, Light-Emitting Diode



왼쪽의 사진에 예시처럼

LED는 우리의 일상 속에서 흔하게 볼 수 있습니다.

그러면 LED를 사용하는 이유는 무엇일까요?





발광 다이오드

LED, Light-Emitting Diode

일반 조명과 LED 조명 비교



자료: 지식경제부

LED을 사용하는 이유는 다음과 같습니다.

1. 에너지 효율이 좋다
2. 친환경적이다
3. 다루기 쉽다

다루기 쉽다라고 되어 있는데 네오아두보드를 통해 얼마나 다루기 쉬운지 알아보도록 하겠습니다.



아두이노 문법

Arduino

setup()

아두이노 보드에 전원이 켜지고
초기에 한번 실행하는 영역(함수)
cf) 보통 설정하는 코드를 작성합니다.

ex) `pinMode()`, `Serial.begin()` 등

loop()

아두이노 보드에 전원이 켜지고
지속적으로 실행하는 영역(함수)



아두이노 문법

Arduino

pinMode()

pinMode(핀번호, 입출력 설정);

핀번호

입출력 설정: OUTPUT / INPUT

- OUTPUT: 출력
- INPUT: 입력

digitalWrite()

digitalWrite(핀번호, 전압 설정);

핀번호

전압 설정: HIGH / LOW

- HIGH: 5V 전압 인가
- LOW: 0V 전압 (=GND 사용 가능)

digitalRead()

digitalRead(핀번호);

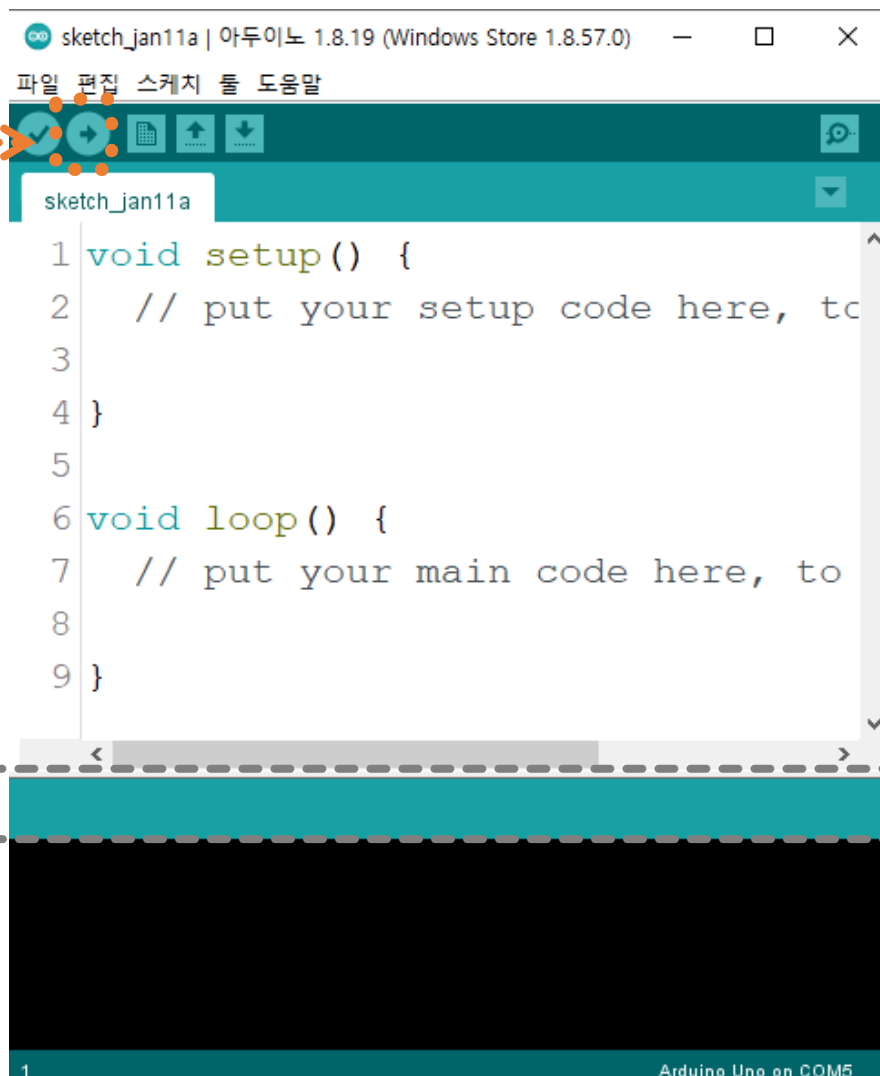
해당 핀의 센서값(0~1)을
가져옵니다.

핀번호



업로드 Upload

업로드 버튼

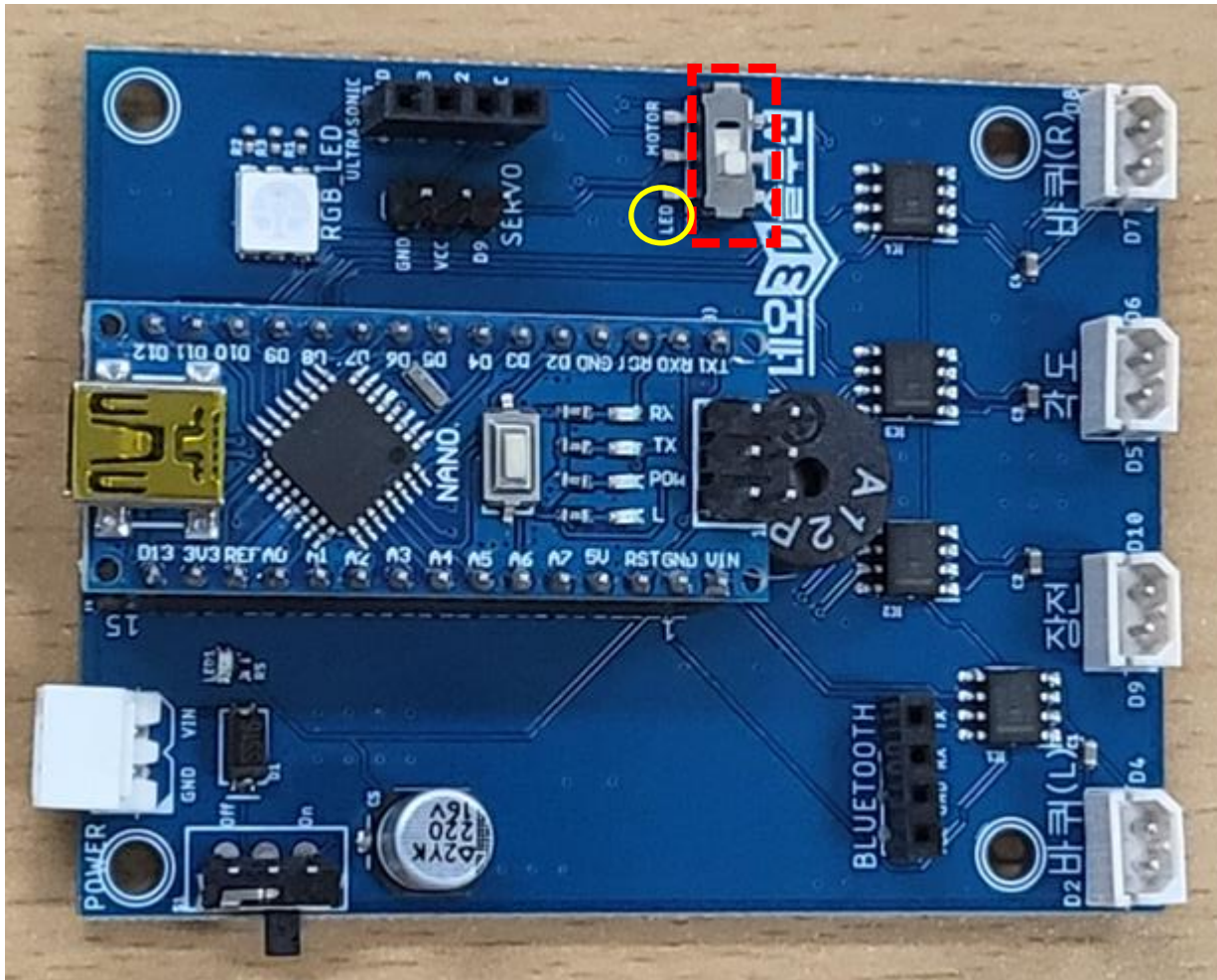


업로드 버튼을 누르면

오른편에 진행상태바가 나오고 완료가 되면

왼쪽편에 '업로드 완료'라고 나오게 됩니다.

LED ↔ MOTOR 선택 스위치



RGB LED를 제어하기
위해서는 하얀색 스위치를
사진과 같이 아래쪽 (LED
글씨) 으로 선택하여
진행해주세요.



발광 다이오드

LED, Light-Emitting Diode

- LED ON

LED를 켜주세요.



발광 다이오드

LED, Light-Emitting Diode

- LED ON

LED를 켜주세요.

```
1 void setup() {  
2     // put your setup code here, to run once:  
3     pinMode(9, OUTPUT);  
4     digitalWrite(9, HIGH);  
5 }  
6  
7 void loop() {  
8     // put your main code here, to run repeatedly:  
9  
10 }
```

// 이 앞에 있으면 한 줄 주석입니다.

주석은 설명을 작성할 때 사용합니다.

코드 실행시에 작동되는 문법은 아닙니다.



아두이노 문법

Arduino

`delay()`

`delay(ms);`

프로그램을 ms 시간만큼 멈추게 합니다.

ms: milliseconds

ex) 1000(ms) -> 1초

Delay를 쓰면 여러가지 동작을 하면서
지속적인 센서의 값을 받거나 얻어야 할 때 문제가 생길 수 있습니다.
이러한 방법을 해결하기 위해서는
millis나 다른 방법을 사용해야 합니다.



발광 다이오드

LED, Light-Emitting Diode

- LED Blink

LED를 1초 간격으로 켜졌다가 꺼졌다가를 반복하게 해주세요.



발광 다이오드

LED, Light-Emitting Diode

- LED Blink

LED를 1초 간격으로 켜졌다가 꺼졌다가를 반복하게 해주세요.

```
1 void setup() {  
2   pinMode(9, OUTPUT);  
3 }  
4  
5 void loop() {  
6   digitalWrite(9, HIGH);  
7   delay(1000);  
8   digitalWrite(9, LOW);  
9   delay(1000);  
10 }
```



PWM이란

디지털 신호 -> 아날로그 신호

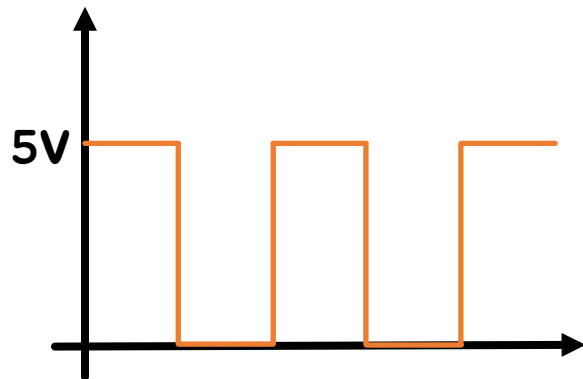
D: 0%

출처: https://ko.wikipedia.org/wiki/듀티_사이클



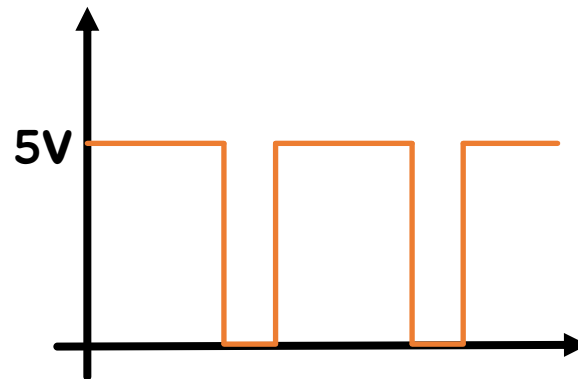
PWM이란

디지털 신호 → 아날로그 신호



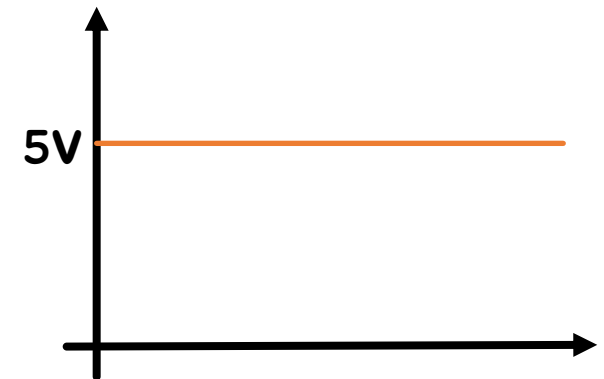
50% Duty Cycle

2.5V



75% Duty Cycle

3.75V



100% Duty Cycle

5V



아두이노 문법

Arduino

analogWrite()

analogWrite(핀번호, 세기값);

핀번호

세기값: 0~255

analogRead()

analogRead(핀번호);

해당 핀의 센서 값(0~1023)을 가져옵니다.

핀번호

analogWrite나 analogRead를 사용할 때는 pinMode를 사용하실 필요가 없습니다.

코드 가독성을 위해서 사용하기도 하니 이점은 이해하시고 넘어가시기만 하면 됩니다.

cf) 아날로그 핀을 디지털 입/출력으로 사용하기 위해서는 pinMode을 사용해주셔야 합니다.



발광 다이오드

LED, Light-Emitting Diode

- 세기 조정

LED의 밝기를 변경해서 보여주세요.



발광 다이오드

LED, Light-Emitting Diode

- 세기 조정

LED의 밝기를 변경해서 보여주세요.

```
1 void setup() {  
2   analogWrite(9, 30);  
3   delay(500);  
4   analogWrite(9, 100);  
5   delay(500);  
6   analogWrite(9, 255);  
7   delay(500);  
8 }  
9  
10 void loop() {  
11 }
```



변수 variable

변수

변수는 '변하는 수' 라는 의미로

프로그래밍에서는 숫자나 문자 같은 데이터를 저장하는 공간을 의미합니다.

$x = x + 1$ 이라 하면 수학적으로는 모순이지만,

프로그래밍에서는 x 라는 값에 1을 더한 값을 다시 x 에 넣는 식입니다.

쉬운 예시로, 우리의 나이가 1년이 지나면 한 살 먹는 것과 유사하다고 보시면 됩니다.

$x = x + 5$ 라면 x 라는 값에 5을 더한 값을 다시 x 에 넣어주는 것이겠죠?!



변수 variable

변수 명명규칙

1. 변수의 이름은 영문자(대소문자), 숫자, 언더스코어(_)로만 구성
2. 변수의 이름은 숫자로 시작될 수 없음
3. 변수의 이름 사이에는 공백을 포함할 수 없음
4. 변수의 이름으로 C언어에서 미리 정의된 키워드(keyword) 사용할 수 없음

auto, break, case, char, const, continue, default,
do, double, else, enum, extern, float, for,
goto, if, int, long, register, return, short,
signed, sizeof, static, struct, switch, typedef, union
unsigned, void, volatile, while



변수 variable

Camel Case

서로 다른 단어가 합쳐질 때
뒤에 첫 단어의 맨 앞에 스펠링이 대문자로 쓰이는 작성 방법

Example)

digital + write = digitalWrite

analog + read = analogRead

pin + mode = pinMode





상수 Constant

상수

변하지 않는 값

상수를 변수명으로 정한다면 모든 스펠링을 대문자로 작성

Example)

HIGH, LOW, OUTPUT, INPUT 등



반복문 for문

for문

```
for (초기값; 조건식; 스텝) {  
    수행문장  
}
```

초기값: 시작값

조건식: 참/거짓 값이 나와야하며, 참일 때만 반복 진행

스텝: 반복할때 마다 증감할 정도

ex) 변수 num에 0~100의 값을 할당

```
int num = 0;  
for (int val = 0; val < 100; val++) {  
    num = val  
}
```



발광 다이오드

LED, Light-Emitting Diode

- 점점 밝아지게

LED 세기가 0에서 1씩 커져서 밝아지게 해주세요. (시간 간격: 0.02초)



발광 다이오드

LED, Light-Emitting Diode

- 점점 밝아지게

LED 세기가 0에서 1씩 커져서 밝아지게 해주세요. (시간 간격: 0.02초)

```
1 int redLed = 9;
2
3 void setup() {
4     pinMode(redLed, OUTPUT);
5     for (int power=0; power < 256; power++) {
6         analogWrite(redLed, power);
7         delay(20);
8     }
9 }
10
11 void loop() {
12 }
```



발광 다이오드

LED, Light-Emitting Diode

실습1. 점점 어두워지게

LED 세기가 최대치에서 1씩 줄어들어 꺼지게 해주세요.

(시간 간격: 0.02초)



발광 다이오드

LED, Light-Emitting Diode

실습1. 점점 어두워지게

LED 세기가 최대치에서 1씩 줄어들어 꺼지게 해주세요.
(시간 간격: 0.02초)

```
1 int redLed = 9;
2
3 void setup() {
4     pinMode(redLed, OUTPUT);
5     for (int power=256; power >= 0; power--) {
6         analogWrite(redLed, power);
7         delay(20);
8     }
9 }
10
11 void loop() {
12 }
```



오버플로우/언더플로우

Overflow/Underflow

analogWrite의 범위가 0~255라고 하였는데 세기값에 범위 외의 값을 넣으면 어떻게 될까?
-10이나 256을 넣고 확인해보자.



오버플로우/언더플로우

Overflow/Underflow

-1을 넣으면 255의 세기와 동일하게 켜지게 될 것이고,
256을 넣으면 0의 세기와 같게 꺼지게 될 것이다.

이러한 현상을 오버플로우/언더플로우 라고 합니다.

오버플로우는 설정값을 넘어선 경우, 언더플로우는 설정값 미만인 경우 발생합니다.

대표적인 예로 싸이 강남스타일의 유튜브 조회수를 볼 수 있습니다.

<https://www.youtube.com/watch?v=oDUdh9ToeJ4>

즉, 범위를 넘어서서 세기값이 이뤄지는 것은 문제가 되는 프로그래밍입니다.

아두이노에서 잡아주기에 오류 없이 실행이 되는 것이지 잡아주지 않는다면 원하지 않는 결과를 얻기 때문입니다.
정확히는 C언어의 컴파일러

이와 같게 각 자료형의 타입은 범위를 가지고 있습니다.

구분	자료형	범위	바이트
정수형	char	-128 ~ 127	1(8)
	unsigned char	0 ~ 255	1(8)
	short	-32768 ~ 32767	2(16)
	int	-2,147,483,648 ~ 2,147,483,647	4(32)
	long	-2,147,483,648 ~ 2,147,483,647	4(32)
	unsigned short	0 ~ 65535	2(16)
실수형	float	$8.4 \times 10^{-37} \sim 3.4 \times 10^{38}$	4(32)
	double	$2.2 \times 10^{-308} \sim 1.8 \times 10^{308}$	8(64)
나열형	enum	정수를 대신하여 사용하는 별명, int형의 크기	
무치형	void	실제 자료는 없음을 명시적으로 선언	

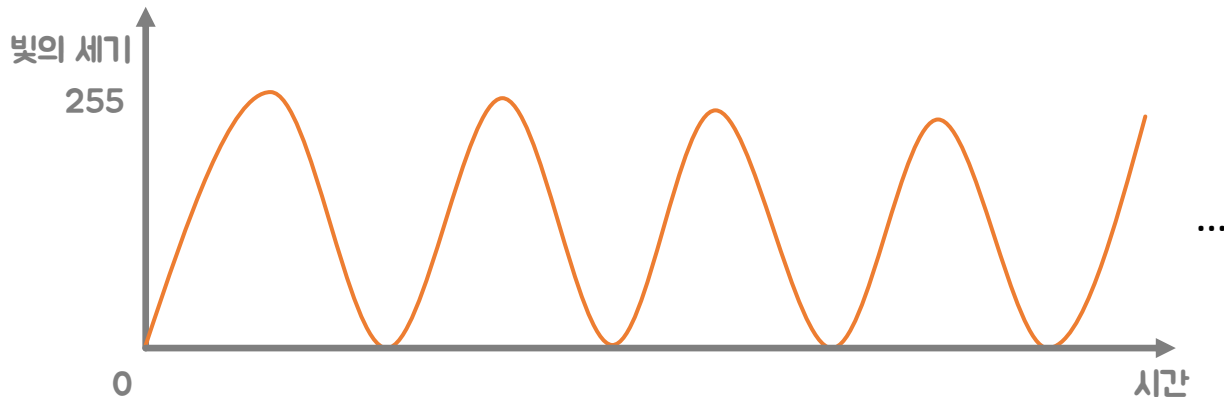


발광 다이오드

LED, Light-Emitting Diode

실습2. 점점 밝아졌다 어두워지게 반복

LED 세기가 1씩 밝아졌다가 최대치 세기가 되면 점점 1씩 어두워지게
0이 되어 꺼지면 다시 점점 밝아지게 이 로직을 반복하여 계속 점점 밝아졌다가
어두워졌다가를 반복하는 프로그램을 작성해주세요. (시간 간격: 0.01초)





발광 다이오드

LED, Light-Emitting Diode

실습2. 점점 밝아졌다 어두워지게 반복

쉬운 방식으로 '점점 밝아지게'와
'점점 어두워지게'를 합쳐서 loop문에 두어
계속 반복하게 작성한 코드입니다.

보다 효율적이지만 어려운 방식으로는
시간 복잡도라는 개념을 고려해 loop문에
반복문을 사용하지 않고 가정문으로
처리하는 방식이 있습니다.

```
1 int redLed = 9;
2
3 void setup() {
4   pinMode(redLed, OUTPUT);
5 }
6
7 void loop() {
8   for (int power=0; power < 256; power++) {
9     analogWrite(redLed, power);
10    delay(10);
11  }
12  for (int power=255; power >= 0; power--) {
13    analogWrite(redLed, power);
14    delay(10);
15  }
16 }
```



가정문 if문

if문

```
if (조건식) {  
    수행문장;  
}
```

조건식: 참/거짓 값이 나와야하며, 참일 때만 중괄호 안에 수행문장 실행



가정문 if문

If~else문

```
if (조건식) {  
    수행문장A;  
} else {  
    수행문장B;  
}
```

조건이 참일 때는 수행문장A 실행

조건이 거짓일 경우는 else문에 있는 수행문장B 실행

If~else if~else문

```
if (조건식A) {  
    수행문장A;  
} else if(조건식B) {  
    수행문장B;  
} else {  
    수행문장C;  
}
```

조건A가 참일 때는 수행문장A 실행

조건A가 거짓일 경우 조건B 참/거짓 판단

조건B가 참이면 수행문장B 실행

조건B도 거짓이면 else문에 있는 수행문장C 실행



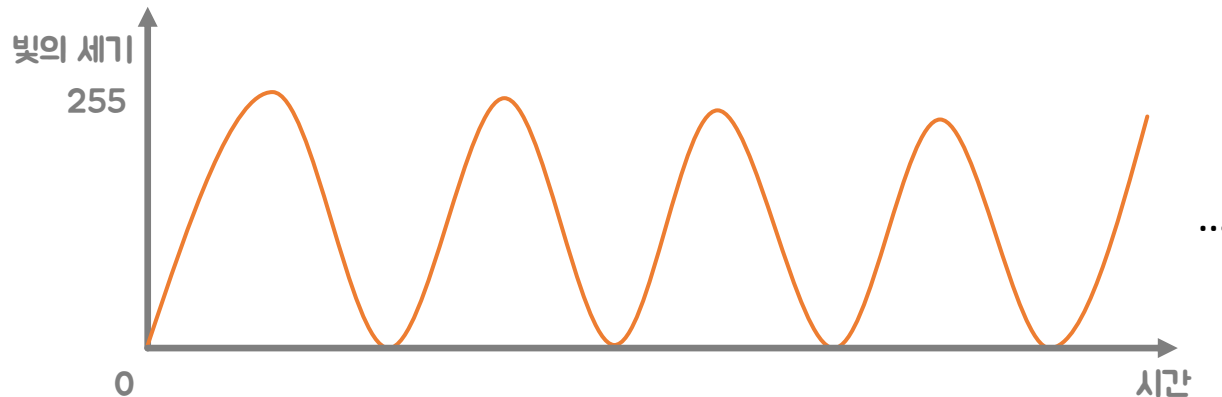
발광 다이오드

LED, Light-Emitting Diode

실습2 심화: 가정문 사용-알고리즘

심화1. 점점 밝아졌다 어두워지게 반복

LED 세기가 1씩 밝아졌다가 최대치 세기가 되면 점점 1씩 어두워지게 0이 되어 꺼지면 다시 점점 밝아지게 이 로직을 반복하여 계속 점점 밝아졌다가 어두워졌다가를 반복하는 프로그램을 작성해주세요. (시간 간격: 0.01초)





발광 다이오드

LED, Light-Emitting Diode

실습2 심화: 가정문 사용-알고리즘

심화1. 점점 밝아졌다 어두워지게 반복

```
1 int redLed = 9;
2 int power = 0;
3 int sw = 1;
4
5 void setup() {
6     pinMode(redLed, OUTPUT);
7 }
8
9 void loop() {
10    analogWrite(redLed, power);
11    delay(10);
12    power += sw;
13
14    if (power == 255) sw = -1;
15    else if (power == 0) sw = 1;
16 }
```



발광 다이오드

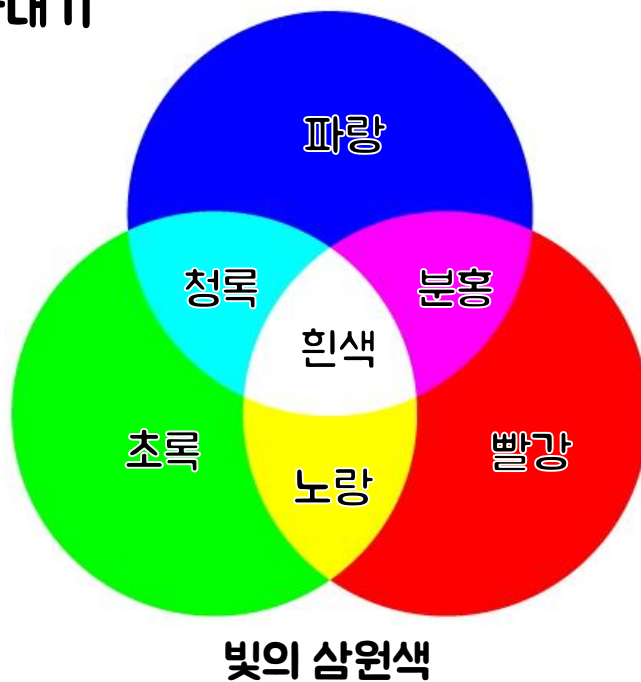
LED, Light-Emitting Diode

- RGB LED

RGB LED를 통해 다양한 색상 나타내기

PIN MAP

RGB LED		Neo Arduino
RED		D9
GREEN		D10
BLUE		D11





발광 다이오드

LED, Light-Emitting Diode

- RGB LED

RGB LED를 통해 다양한 색상 나타내기

이전 색상이 켜있는 상태에서 다른 색상을 켜게 되면
빛의 삼원색에 따라 합쳐진 색상이 나오는 것을 확인할 수 있습니다.

그렇다면 단색을 나타내기 위해서는 이전의 색상을
꺼주어야 한다는 것입니다.

```
1 int redLed    = 9;
2 int greenLed  = 10;
3 int blueLed   = 11;
4
5 void setup() {
6   pinMode(redLed, OUTPUT);
7   pinMode(greenLed, OUTPUT);
8   pinMode(blueLed, OUTPUT);
9
10  // 초록
11  digitalWrite(greenLed, HIGH);
12  delay(2000);
13  // 노랑
14  digitalWrite(redLed, HIGH);
15  delay(2000);
16  // 흰색
17  digitalWrite(blueLed, HIGH);
18 }
19
20 void loop() {
21 }
```




발광 다이오드

LED, Light-Emitting Diode

실습3. RGB LED

RGB LED를 통해 다양한 색상 나타내기

빨 → 초 → 파 순서로 켜기 (시간 간격 2초)



발광 다이오드

LED, Light-Emitting Diode

실습3. RGB LED

RGB LED를 통해 다양한 색상 나타내기

빨 → 초 → 파 순서로 켜기 (시간 간격 2초)

```
1 int redLed    = 9;
2 int greenLed  = 10;
3 int blueLed   = 11;
4
5 void setup() {
6     pinMode(redLed, OUTPUT);
7     pinMode(greenLed, OUTPUT);
8     pinMode(blueLed, OUTPUT);
9
10    // 빨강
11    digitalWrite(redLed, HIGH);
12    delay(2000);
13    // 초록
14    digitalWrite(redLed, LOW);
15    digitalWrite(greenLed, HIGH);
16    delay(2000);
17    // 파랑
18    digitalWrite(greenLed, LOW);
19    digitalWrite(blueLed, HIGH);
20 }
21
22 void loop() {
23 }
```



함수 Function

함수는 특정 용도의 코드들을 한 곳에 모아 사용하기 편하게 하는 문법입니다.

우리가 지금까지 당연하게 사용한 `setup`과 `loop`도 함수입니다.

해당 함수는 아두이노IDE에서 꼭 필수적인 함수라는 점이 다른 함수들과는 다를 뿐입니다.

(C언어 계열에서는 `main`함수가 필수적입니다. → 아두이노IDE 내부적으로 `setup`과 `loop`문은 `main`함수 안에서 동작되게 프로그래밍 되어 있습니다.)

함수의 개념은 처음에 이해하기 어려울 수 있습니다.

그러니 어렵다고 포기하지 마시고 추후 다시 살펴봐도 됩니다.



함수 Function

RGB LED를 간단하게 사용하기 위한 함수를 만들어 보겠습니다.

```
digitalWrite(redLed, HIGH/LOW );  
digitalWrite(greenLed, HIGH/LOW );  
digitalWrite(blueLed, HIGH/LOW );
```

RGB 색상을 하나 만들기 위해서는 위의 3줄의 코딩이 필요합니다.

이것을 지속적으로 써주면 줄이 너무 길어지기에 함수로 처리하면 간단해집니다.



함수 Function

void

void는 반환값이 없는 함수를 나타냅니다.

함수는 값을 반환을 할 수가 있고, 반환을 하지 않을 수도 있습니다.

반환을 하기 위해서는 반환할 자료형을 사용해주면 됩니다. (ex. int, float)



함수 Function

반환형 함수 예시

```
float getDistance(int trigPin, int echoPin) {  
    pinMode(trigPin, OUTPUT);  
    pinMode(echoPin, INPUT);  
    digitalWrite(trigPin, LOW);  
    delayMicroseconds(2);  
    digitalWrite(trigPin, HIGH);  
    delayMicroseconds(10);  
    digitalWrite(trigPin, LOW);  
    float duration = pulseIn(echoPin, HIGH);  
    return duration / 29 / 2;  
}
```

return을 통해서 반환할 값을 설정하면 됩니다.

단, 반환 타입과 반환 값의 타입은 맞춰주어야 합니다.

해당 부분은 매개변수(parameter)라고 합니다.

매개변수는 함수 영역에서 사용할 수 있는 변수를 지정했다고 생각하시면 됩니다.

매개변수의 값은 함수를 호출할 때 사용합니다.

현재 지금은 함수를 정의한 것입니다.

```
void loop() {  
    float distance = getDistance(12, 13);  
    if (distance > 20) {  
        digitalWrite(redLed, LOW);  
    } else {  
        digitalWrite(redLed, HIGH);  
    }  
}
```

함수 호출



발광 다이오드 LED

실습4. 함수

RGB LED를 제어하는 함수를 만들어
원하는 색상을 켜보세요.



발광 다이오드 LED

실습4. 함수

```
1 int redLed    = 9;
2 int greenLed  = 10;
3 int blueLed   = 11;
4
5 void rgbCtrl(boolean redState, boolean greenState, boolean blueState) {
6     digitalWrite(redLed, redState);
7     digitalWrite(greenLed, greenState);
8     digitalWrite(blueLed, blueState);
9 }
10
11 void setup() {
12     pinMode(redLed, OUTPUT);
13     pinMode(greenLed, OUTPUT);
14     pinMode(blueLed, OUTPUT);
15
16     rgbCtrl(true, false, false); // 빨강
17     delay(2000);
18     rgbCtrl(true, true, false); // 노랑
19     delay(2000);
20     rgbCtrl(false, true, true); // 청록
21     delay(2000);
22     rgbCtrl(true, true, true); // 흰색
23 }
24
25 void loop() {
26 }
```

rgbCtrl이라는 함수를 만들어서
간단하게 기능을 만들었습니다.



부저 Buzzer



수정이나 세라믹 같은 결정체의 성질(압전 물질)을 이용한 소자
압전 물질에 얇은 판을 대어 **압전 효과**에 의해 소리 발생



부저 Buzzer

피에조 부저는 능동 부저와 수동 부저로 나누어져 있습니다.

능동 부저는 단일음만 낼 수 있고, 수동 부저는 피아노처럼 음계를 나타낼 수 있습니다.



능동 부저 / 수동 부저 구분하는 방법

1. 다리 길이 구별법

다리 길이 같음: 능동 부저

다리 길이 다름: 수동 부저

2. 스티커 유무 구분법

스티커 있음: 능동 부저

스티커 없음: 수동 부저

위의 구분 방법이 정확하진 않습니다. 코드상으로 확인 하는 방법이 가장 확실한 방법 입니다.

tone함수로 소리가 음계 소리가 나면 수동 부저입니다.



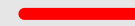
부저 Buzzer

PIN MAP

Buzzer

Neo
Arduino

+



D3

-



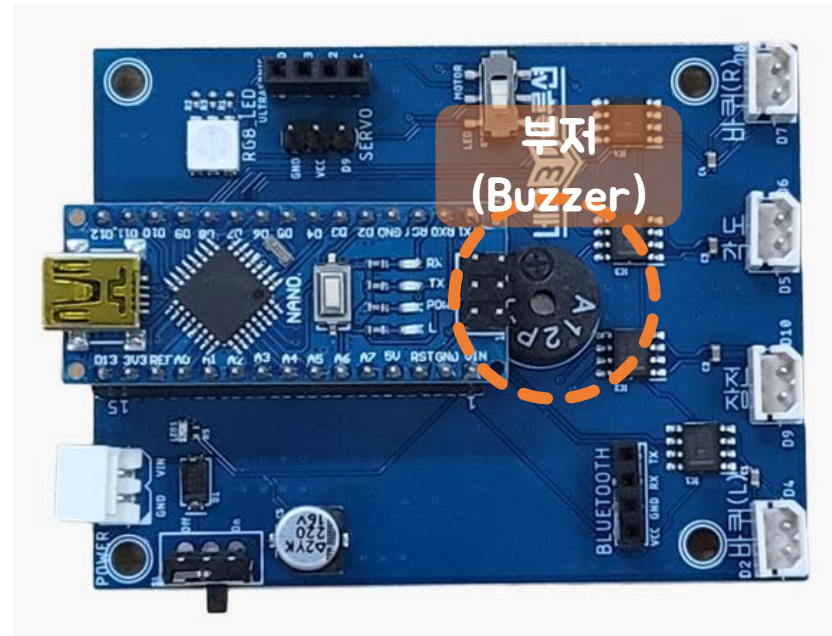
GND

1. 수동부저와 능동부저의 차이점

- 단일음 / 음계표현
- 다리길이/스티커

2. 부저 멜로디

- tone을 통한 멜로디 만들기





부저 Buzzer

tone(핀번호, 주파수, 시간ms)

생략 가능 → 생략시 계속 주파수 음 출력

noTone(핀번호)

: Tone()에 의해 시작된 구형파 발생 멈춤

시간ms를 설정하지 않아서 지속적인 음을 출력했을 때는 적당한 시간을 준 후에

noTone을 사용하면 음 출력을 정지할 수 있습니다.



부저 Buzzer

실습1. 멜로디 만들기

수동 부저를 이용하여 만들고 싶은 멜로디를 완성해주세요.

음계 ↓

→ 옥타브

	1	2	3	4	5	6	7	8
C(도)	33	65	131	262	523	1047	2093	4186
D(레)	37	73	147	294	587	1175	2349	4699
E(미)	41	82	165	330	659	1319	2637	5274
F(파)	44	87	175	349	698	1397	2794	5588
G(솔)	49	98	196	392	784	1568	3136	6272
A(라)	55	110	220	440	880	1760	3520	7040
B(시)	62	123	247	494	988	1976	3951	7902



부저 Buzzer

실습1. 멜로디 만들기

수동 부저를 이용하여 만들고 싶은 멜로디를 완성해주세요.

학교종이 땡땡땡 일부

```
1 int buzzer = 3;
2
3 void setup() {
4   tone(buzzer, 392, 300);
5   delay(500);
6   tone(buzzer, 392, 300);
7   delay(500);
8   tone(buzzer, 440, 300);
9   delay(500);
10  tone(buzzer, 440, 300);
11  delay(500);
12  tone(buzzer, 392, 300);
13  delay(500);
14  tone(buzzer, 392, 300);
15  delay(500);
16  tone(buzzer, 330, 300);
17  delay(500);
18
19  tone(buzzer, 392, 300);
20  delay(500);
21  tone(buzzer, 392, 300);
22  delay(500);
23  tone(buzzer, 330, 300);
24  delay(500);
25  tone(buzzer, 330, 300);
26  delay(500);
27  tone(buzzer, 294, 300);
28  delay(500);
29 }
30
31 void loop() {
32 }
```

tone함수의 울릴 시간ms + 쉼 시간을 주어야 음 간격을 줄 수 있습니다.
즉, delay 값에 tone의 ms시간 + 쉼 시간을 넣어 주셔야 합니다.



부저 Buzzer

실습2. 경보 알림 만들기

부저와 LED를 이용하여 경보 알림 기능을 만들어주세요.

예시) 경찰차 사이렌



부저 Buzzer

실습2. 경보 알림 만들기

부저와 LED를 이용하여 경보 알림 기능을
만들어주세요. 예시) 경찰차 사이렌

```
1 int redLed = 9;
2 int blueLed = 11;
3 int buzzer = 3;
4
5 void setup() {
6   pinMode(redLed, OUTPUT);
7   pinMode(blueLed, OUTPUT);
8   pinMode(buzzer, OUTPUT);
9 }
10
11 void loop() {
12   digitalWrite(blueLed, LOW);
13   digitalWrite(redLed, HIGH);
14   tone(buzzer, 1004, 300);
15   delay(500);
16   digitalWrite(redLed, LOW);
17   digitalWrite(blueLed, HIGH);
18   tone(buzzer, 1524, 300);
19   delay(500);
20 }
```




tone함수

PWM 영향

tone()

[Advanced I/O]

설명

핀에 특정 주파수(50% 듀티 사이클)의 구형파를 발생시킵니다. 지속 시간을 정할 수 있으며, 따로 정하지 않는다면 `noTone()`을 부를 때까지 구형파가 지속됩니다. 핀을 피에조 버저 또는 스피커에 연결하여 tone을 연주할 수 있습니다.

한번에 한 tone만 발생시킬 수 있습니다. 다른 핀에서 tone이 이미 연주되고 있으면, tone()을 새로 불러도 아무 일도 일어나지 않을 것입니다. 같은 핀에서 tone이 연주되고 있으면, 주파수가 새로 설정될 것입니다.

tone() 함수의 사용은 (Mega 이외의 보드에서) 3번과 11번 핀에서의 PWM 출력을 방해할 것입니다. 31HZ보다 낮은 tone을 발생시키는 것은 불가능합니다. 기술적인 세부 사항은, [Brett Hagman's notes](#)를 보십시오.

출처: <https://www.arduino.cc/reference/ko/language/functions/advanced-io/tone/>



배열 Array

1	MSG워너비 (M.O.M)	3:32
2	Next Level - aespa	3:41
3	신호등 - 이무진	3:52
4	Permission to Dance - 방탄소년단	3:07
5	Butter - 방탄소년단	2:44
6	Weekend - 태연 (TAEYEON)	3:53
7	헤븐 우연 - 헤이즈 (Heize)	3:13
8	치맛바람 (Chi Mat Ba Ram) - 브레이브걸스	3:37
9	나를 아는 사람 - MSG워너비 (정상동기)	4:52
10	Dun Dun Dance - 오마이걸	3:40
11	Peaches (feat. Daniel Caesar & Giveon) - Drake	3:18
12	롤린 (Rollin') - 브레이브걸스 (Brave Girls)	3:17
13	종아종아 - 조정석	3:36
14	Alcohol-Free - TWICE (트와이스)	3:30
15	Dynamite - 방탄소년단	3:19
16	라일락 - 아이유 (IU)	3:34
17	비와 당신 - 이무진	4:22
18	ASAP - STAYC (스테이씨)	3:14
19	안녕 (Hello) - 조이 (JOY)	3:38
20	Celebrity - 아이유	3:15
21	운전만해 (We Ride) - 브레이브걸스	3:09
22	상상더하기 - MSG워너비 TOP 8	4:01
23	호미들 - 사이렌 리믹스	3:18

배열을 설명하기에 앞서 멜론 플레이리스트를 가져와보겠습니다.

여기서 플레이리스트 할 때 리스트로 보아 우리는 **배열**이 무엇인지 유추가 됩니다.

가장 앞에 있는 번호가 플레이리스트의 순서입니다.

이와 같이 배열은 순서를 담당하는 부분이 있습니다.

우리는 이 순서를 인덱스(index)라고 할 것입니다.

즉, **인덱스는 값이 놓여진 위치를 의미**하는 것입니다.

그리고 해당 인덱스에 쓰여져 있는 부분이 값입니다.

배열은 값들을 나열 해놓은 자료형이며, 인덱스와 값으로 이루어져 있습니다.



배열 Array

{"바라만 본다", "Next Level", "신호등", "Permission to Dance", "Butter"}

인덱스

0

1

2

3

4

값

바라만 본다

Next Level

신호등

Permission to Dance

Butter

프로그래밍에서 인덱스는 0부터 시작합니다.

배열에 나열된 순서에 따라 인덱스는 1씩 증가하게 됩니다.



배열 Array

String MelonChart[5] =

{"바라만 본다", "Next Level", "신호등", "Permission to Dance", "Butter"}

문제의 인덱스의 시작은 0이라고 두고 시작한다.

Q1. MelonChart 배열의 3번째 위치의 값은 무엇인가?

Q2. MelonChart 배열에 " Butter"는 어느 위치에 있는가?

Q2. MelonChart 배열에 " butter"값이 있는가?



배열 Array

String MelonChart[5] =

{"바라만 본다", "Next Level", "신호등", "Permission to Dance", "Butter"}

문제의 인덱스의 시작은 0이라고 두고 시작한다.

Q1. MelonChart 배열의 3번째 위치의 값은 무엇인가? "Permission to Dance"

Q2. MelonChart 배열에 " Butter"는 어느 위치에 있는가? 4

Q2. MelonChart 배열에 " butter"값이 있는가? 없다

[설명] 프로그래밍에서 영어의 소문자와 대문자는 구별됩니다.



부저 Buzzer

실습1 심화: 배열 사용

심화1. 배열을 이용해서 원하는 노래 만들기

자신이 원하는 노래를 만들어서
친구들에게 자랑해보세요.



부저 Buzzer

실습1 심화: 배열 사용 방법1: For문

심화1. 배열을 이용해서 원하는 노래 만들기

sizeof함수는 배열의 총 바이트 수를 반환합니다.

아두이노에서 int는 2byte로 이루어져 있습니다.

배열의 구성요소 개수를 구하려고 한다면

배열의 총 바이트 수를 구한 값에서

배열의 한 요소에서 바이트 사이즈를 나눠주면

구성요소 개수를 나타낼 수 있습니다.

현재 예시로, notes의 구성 요소가 12개가 있습니다.

배열의 총 요소 개수 int의 byte수
sizeof(notes)의 값은 $(12) * (2) = 24$

sizeof(notes[0])의 값은 $1 * 2 = 2$ 가 나옵니다.

notes배열의 첫번째 요소의 값
이를 이용해 구성 요소 12개의 값을 얻을 수 있습니다.

```
1 int buz = 3;
2
3 int notes[] = {
4   392, 392, 440, 440, 392,
5   392, 330, 392, 392, 330,
6   330, 294
7 };
8
9 void setup() {
10  for (int idx = 0; idx < (sizeof(notes)/sizeof(notes[0])); idx++) {
11    tone(buz, notes[idx], 300);
12    delay(500);
13  }
14 }
15
16 void loop() {
17
18 }
```



부저 Buzzer

실습1 심화: 배열 사용 방법2: For~each문

심화1. 배열을 이용해서 원하는 노래 만들기

배열과 같은 문법에서 사용할 수 있는

For~Each문을 사용하여 보다 쉽게 처리할 수 있습니다.

배열의 한 요소씩 탐색하면서 진행하기에 변수는 탐색한 요소값이 됩니다.
해당 요소값을 통하여 원하는 수행문장을 작성할 수 있습니다.

```
for ( 변수 : 배열변수 ) {  
    수행문장  
}
```

```
1 int buzzer = 3;    학교종이 땡땡땡 일부  
2 int notes[12] = {  
3     392, 392, 440, 440, 392,  
4     392, 330, 392, 392, 330,  
5     330, 294  
6 };  
7  
8 void setup() {    For~each문  
9     for (int note : notes) {  
10         tone(buzzer, note, 300);  
11         delay(500);  
12     }  
13 }  
14  
15 void loop() {  
16 }
```




부저 Buzzer

실습2 심화: 삼화 연산자

심화2. 경보장치 효과 내기

LED를 빨간색과 파란색을 왔다갔다 하면서
2개의 다른 음을 내주세요.



부저 Buzzer

실습2 심화: 삼화 연산자

심화2. 경보장치 효과 내기

삼항연산자란, 조건식이 참이면 참일 때 지정한 값을 적용하고,
조건식이 거짓이면 거짓일 때 지정한 값을 적용하는 문법입니다.

조건식 ? 참일때의 값 : 거짓일때의 값

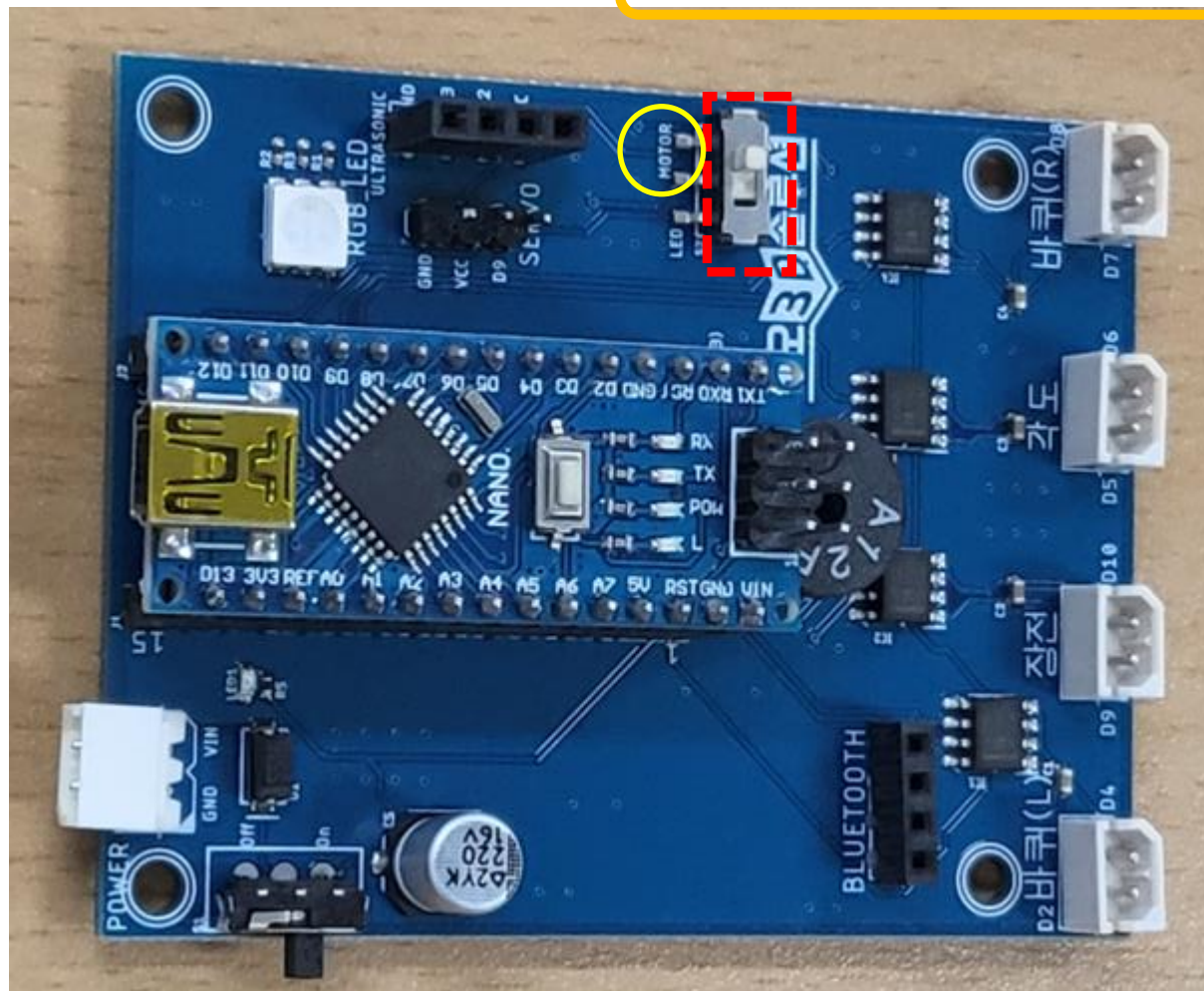
```
1 int redLed = 9;
2 int blueLed = 11;
3 int buzzer = 3;
4 boolean isState = false;
5
6 void setup() {
7   pinMode(redLed, OUTPUT);
8   pinMode(blueLed, OUTPUT);
9   pinMode(buzzer, OUTPUT);
10 }
11
12 void loop() {
13   int note = isState? 1524 : 1004; // 삼항연산자
14   digitalWrite(blueLed, isState);
15   digitalWrite(redLed, !isState);
16   tone(buzzer, note, 300);
17   delay(500);
18   isState = !isState;
19 }
```



직류 전동기

DC Motor, Direct Current Motor

LED ↔ MOTOR 선택 스위치



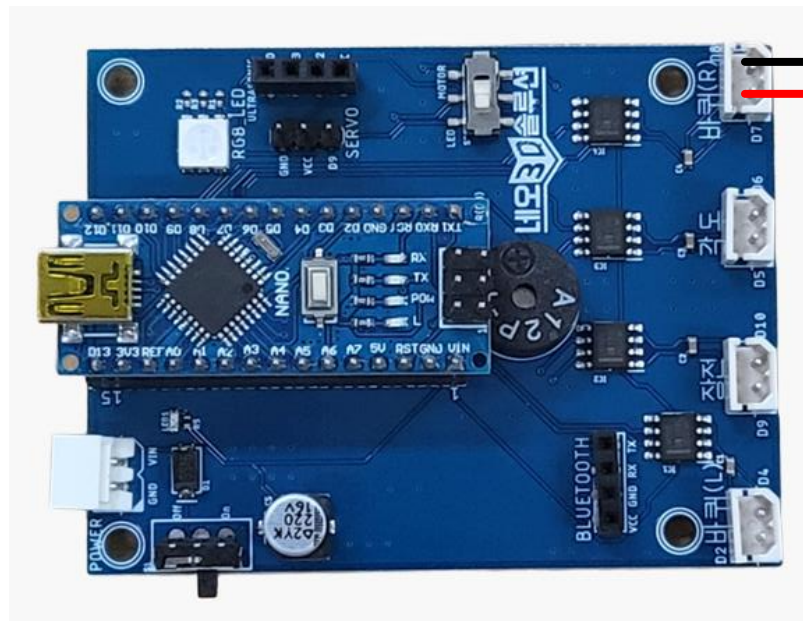
모터를 제어하기
위해서는
스위치를 사진과 같이
위쪽 (MOTOR 글씨) 으로
선택하여 진행해주세요.

(학습이 끝나면 선택 스위치는
모터쪽으로 선택 되어 있어야
게임 시 네오캐논을 동작시킬
수 있습니다.)



직류 전동기

DC Motor, Direct Current Motor





직류 전동기

DC Motor, Direct Current Motor

- 회전/역회전

DC 모터를 회전 / 역회전으로 돌려보기



직류 전동기 DC Motor

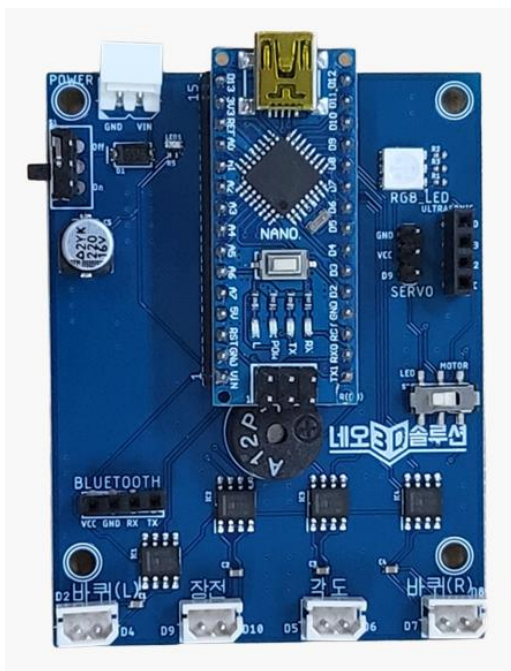
- 회전/역회전

DC 모터를 정회전 / 역회전으로 돌려보기

```
1
2 int lfw = 2; // left forward
3 int lbw = 4; // left backward
4
5 void setup() {
6   pinMode(lfw, OUTPUT);
7   pinMode(lbw, OUTPUT);
8
9   digitalWrite(lfw, HIGH); // 정회전
10  digitalWrite(lbw, LOW);
11  delay(2000);
12
13  digitalWrite(lfw, LOW); // 역회전
14  digitalWrite(lbw, HIGH);
15  delay(2000);
16
17  digitalWrite(lfw, LOW); // 멈춤
18  digitalWrite(lbw, LOW);
19 }
20
21 void loop() {
22 }
```



진동 감지 센서 Vibration Sensor



[참고]

진동센서는 위 그림에서와 같이 나노보드 아래에 내장되어 있습니다.
눈에 잘 보이지 않는 곳에 숨어 있지만 아주 미세한 진동도 잘 감지합니다.
진동센서와 LED / 부저를 사용하여 지진 감지장치를 만들어 보겠습니다.



시리얼 통신

Serial Communication

시리얼 통신을 통해 컴퓨터와 아두이노 보드간 데이터를 송수신 할 수 있습니다.

해당 통신을 위해서는 기본적인 설정이 필요합니다.

1byte 데이터를 주고 받는 직렬 통신입니다.

1byte = 8bit

```
1 int vibe = A0;
2
3 void setup() {
4   Serial.begin(9600);
5   pinMode(vibe, INPUT);
6 }
7 void loop() {
8   int value = analogRead(vibe);
9   Serial.print("Vibe Value : ");
10  Serial.println(value);
11 }
```

시리얼 통신 속도 지정

보드레이트 → 1초에 9600 bit 송수신

print → 출력

println → 출력 후 줄바꿈

('\' 은 소문자 'L')

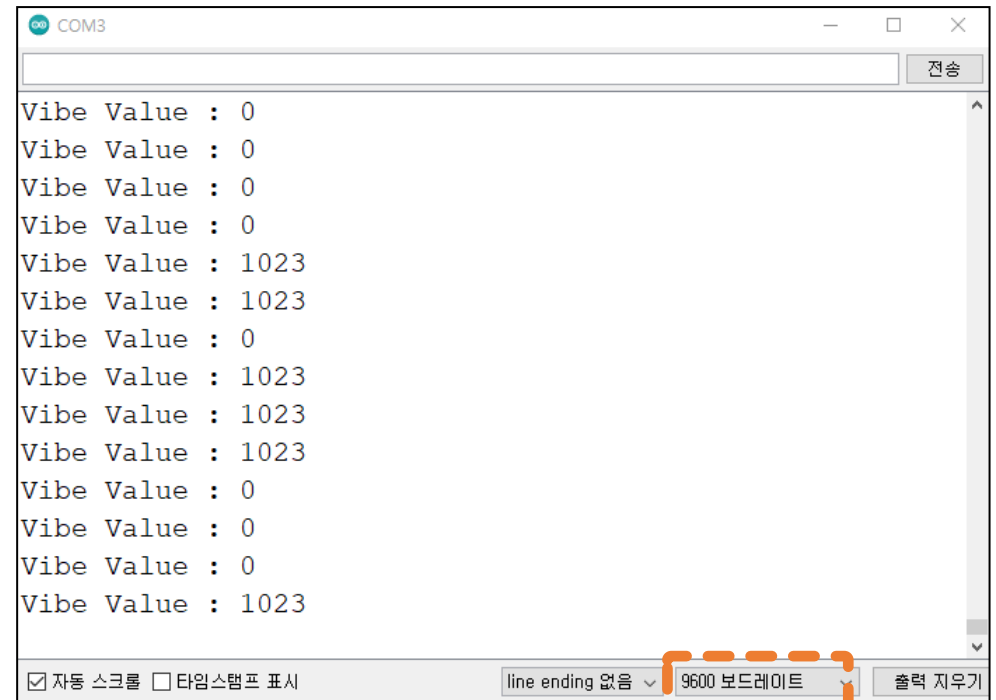


시리얼 모니터 Serial Monitor

시리얼 모니터

```
sketch_aug21a | 아두이노 1.8.15
파일 편집 스케치 툴 도움말
시리얼 모니터
sketch_aug21a $
1 int vibe = A0;
2
3 void setup() {
4   Serial.begin(9600);
5   pinMode(vibe, INPUT);
6 }
7 void loop() {
8   int value = analogRead(vibe);
9   Serial.print("Vibe Value : ");
10  Serial.println(value);
11 }
12
13
스케치는 프로그램 저장 공간 1902 바이트(6%)를 사용. 최대 30720 바이트
전역 변수는 동적 메모리 202바이트(9%)를 사용, 1846바이트의 지역변
13 Arduino Nano, ATmega328P on COM3
```

시리얼 모니터 창



보드레이트 설정

Serial.begin(값)과 동일해야 합니다.



진동 감지 센서

Vibration Sensor

실습1. 진동 감지시 알림 사이렌 만들기

진동 감지시에 0.3초 초록색 LED가 켜지고 소리가 나며,
0.2초 LED가 꺼지고 소리가 안 나게
총 5초 동안 작동하는 프로그램을 만들어 주세요.



진동 감지 센서 Vibration Sensor

실습1. 진동 감지시 알람 사이렌 만들기

진동 감지시에 0.3초 초록색 LED가 켜지고 소리가 나며,
0.2초 LED가 꺼지고 소리가 안 나게
총 5초 동안 작동하는 프로그램을 만들어 주세요.

```
1 int buzzer = 3;
2 int blueLed = 11;
3 int vibe = A0;
4
5 void setup() {
6   pinMode(buzzer, OUTPUT);
7   pinMode(blueLed, OUTPUT);
8   pinMode(vibe, INPUT);
9 }
10
11 void loop() {
12   int vibeData = analogRead(vibe);
13   if(vibeData > 500) {
14     for(int cnt=0; cnt<10; cnt++){
15       digitalWrite(blueLed, HIGH);
16       tone(buzzer, 523, 300);
17       delay(300);
18       digitalWrite(blueLed, LOW);
19       noTone(buzzer);
20       delay(200);
21     }
22   }
```



시리얼 통신

Serial Communication

- 시리얼 통신을 통해 LED 제어

시리얼 통신을 통해서 센서, 모듈들을 제어할 수도 있습니다.

시리얼 모니터에 0이라는 문자를 작성하면 파랑LED가 꺼지고,

1이라는 문자를 작성하면 파랑LED가 켜지게 해보겠습니다.



시리얼 통신

Serial Communication

- 시리얼 통신을 통해 LED 제어

자동 형변환

: 기본 타입에서 자신보다 큰 타입이면 컴파일러가
자동으로 큰 타입으로 맞춰주는 것을 자동 형변환이라고 합니다.
현재 Serial.read()의 타입은 byte 형태인데 char 형태로
자동 형변환 되어 문자형태로 받아드릴 수 있습니다.

아두이노의 시리얼모니터를 열고 0 과 1 을 입력 후 전송(엔터)
시켜 보시기 바랍니다.

```
1 int blueLed = 11;
2
3 void setup() {
4     Serial.begin(115200);
5     pinMode(blueLed, OUTPUT);
6 }
7
8 void loop() { 시리얼 통신이 가능하면
9     if (Serial.available()) {
10         char cmd = Serial.read(); 자동 형변환
11         if (cmd == '0') { 시리얼 통신에서 읽은 값
12             digitalWrite(blueLed, LOW);
13         } else if (cmd == '1') {
14             digitalWrite(blueLed, HIGH);
15         }
16     }
17 }
```



타이머 함수 millis()

Delay를 쓰면 아두이노가 모든 동작을 멈추기 때문에 led 를 깜빡이는 동작을 하면서 센서 감지 동작 또는 모터 회전 동작 등 다른 기능을 멈추지 않게 하기 위해 millis 함수를 사용합니다. 예제에서 led 가 1초 간격으로 깜빡이지만 모터는 멈추지 않고 계속 회전하도록 처리 하였습니다.

```
1 int blueLed = 11;
2 int mtfw = 7;
3 int mtbw = 8;
4 unsigned long currentTime;
5 boolean ledState;
6
7 void setup() {
8   pinMode(blueLed, OUTPUT);
9   pinMode(mtfw, OUTPUT);
10  pinMode(mtbw, OUTPUT);
11 }
12
13 void loop() {
14   if(millis() - currentTime > 1000){
15     currentTime = millis();
16     ledState = !ledState;
17     digitalWrite(blueLed, ledState);
18   }
19   digitalWrite(mtfw, HIGH);
20   digitalWrite(mtbw, LOW);
21 }
```

millis() 함수

아두이노 보드가 프로그램을 돌리기 시작한 후

현재까지 흐른 시간 값을 가져오는 함수

대략 50일이 지나면 overflow 발생하여 0으로 초기화

