

Embedded System Design

아두이노를 이용한 임베디드 설계 전략



목차

- 아두이노 개요
- 아두이노를 이용한 제어
 - LED 제어를 통한 디지털/아날로그 출력
 - 스위치 제어를 통한 디지털 입력
 - 가변 저항 제어를 통한 아날로그 입력
 - 조도센서, 온도센서
 - 피에조부저
 - 7Segment/초음파 센서/모터 제어
- 블루투스 통신



Arduino 열풍

- Maker Movement
- Internet of Things

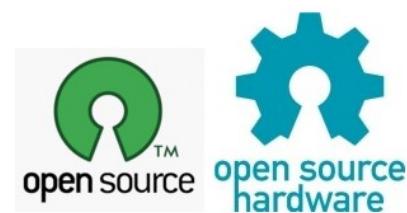


[참고] 아두이노를 활용한 IoT 프로젝트

<http://www.imaso.co.kr/?p=6002>

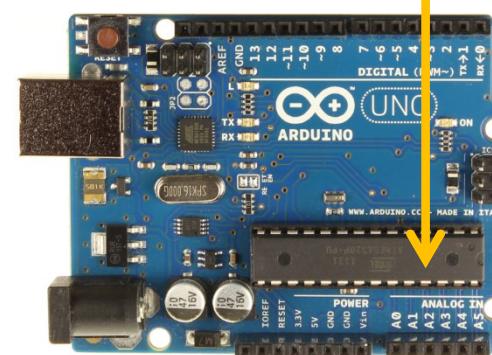
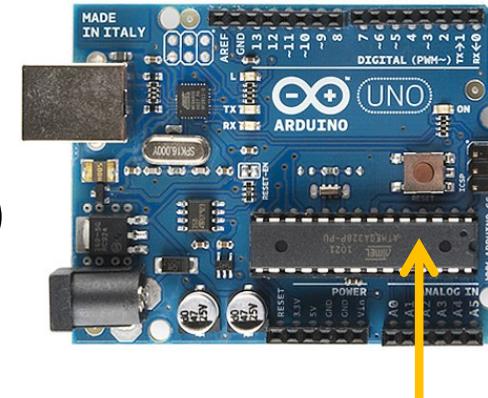
아두이노(Arduino)

- 오픈소스 기반의 마이크로컴퓨터
 - AVR 계열의 칩셋 사용 (Atmega328)
 - 멀티 플랫폼 지원(Windows, Mac, Linux 지원)
- 전문지식 없이 임베디드 시스템 구현이 가능한 설계 도구
 - IDE에서 프로그래밍 코드(Sketch) 작성하고 컴파일
 - USB를 통해 아두이노로 업로드
 - C/C++ 기반으로 개발되어 라이브러리를 통해 기능 확장 가능

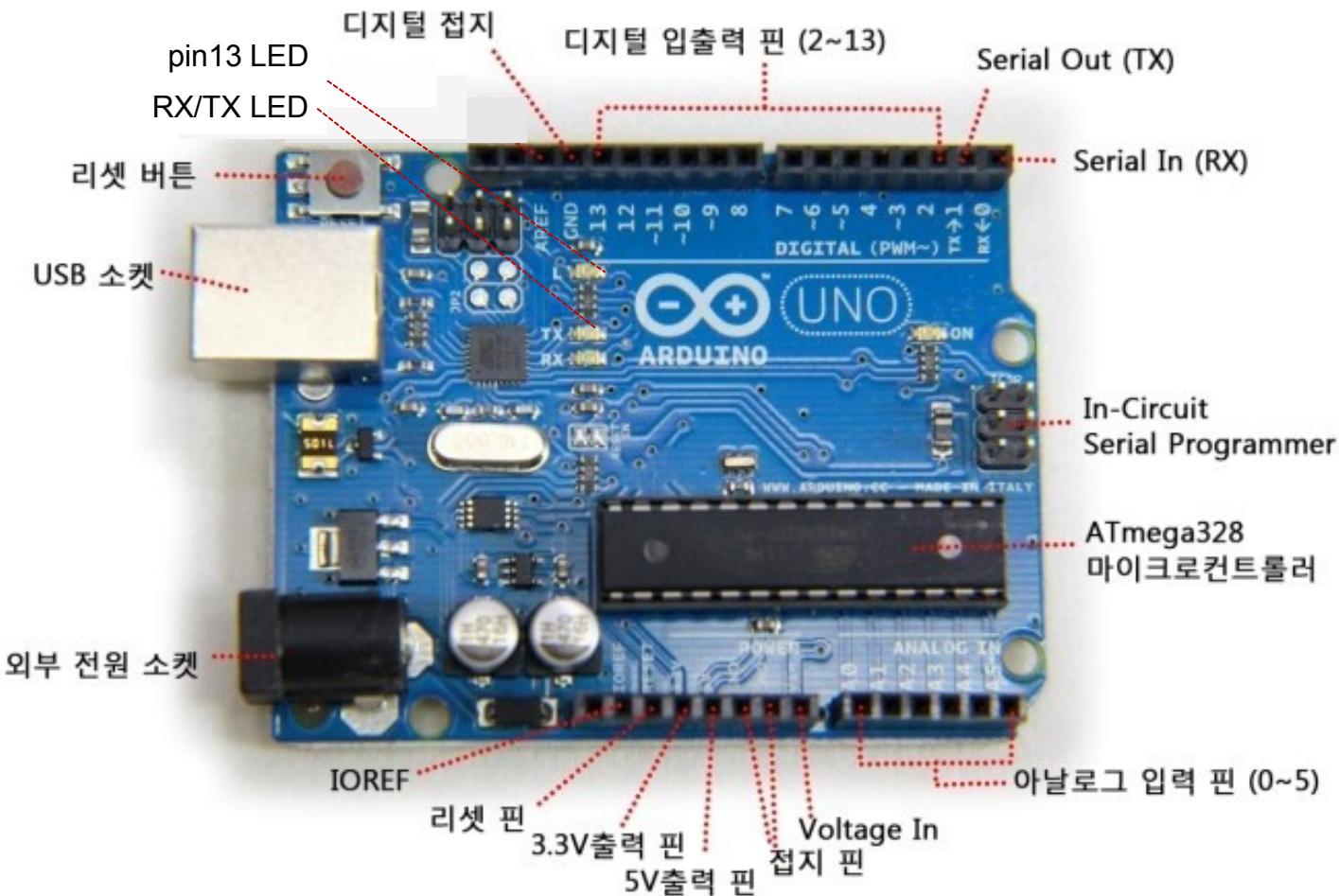


Arduino UNO

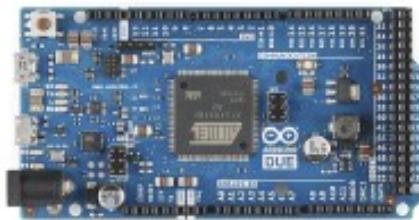
- **UNO 특징**
 - ATmega328 microcontroller
 - Input voltage: 7-12V
 - 14 Digital I/O Pins (6 PWM outputs)
 - 6 Analog Inputs
 - 32KB Flash Memory
 - 16MHz Clock Speed
- **UNO R3**
 - USB 통신칩 변경하여 전송속도 향상 (Atmega16U2 사용)
 - Linux와 Mac에서 드라이버 설치 없이 사용 가능
 - 헤더핀 3개 추가 및 리셋버튼 위치 변경



Arduino UNO R3



다양한 Arduino 보드



Arduino Due



Arduino Leonardo



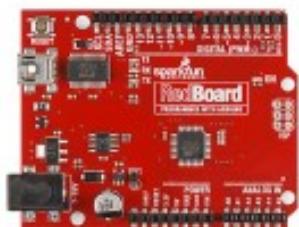
Arduino Uno R3



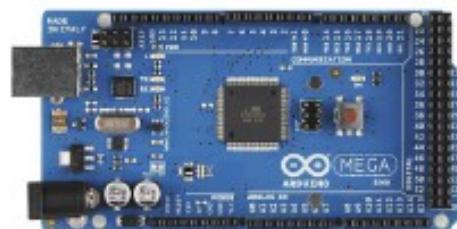
Mega Pro Mini 3.3V



Pro Micro



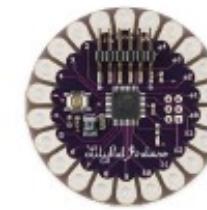
RedBoard



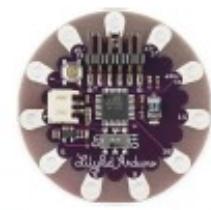
Arduino Mega 2560 R3



Arduino Pro



LilyPad Main



Lily Pad Simplicity



Mega Pro 5V



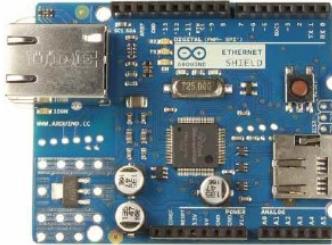
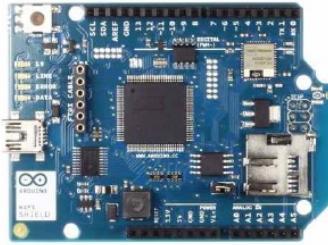
Arduino Fio



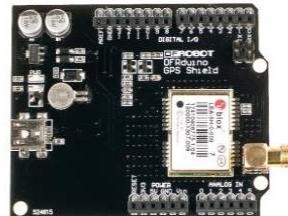
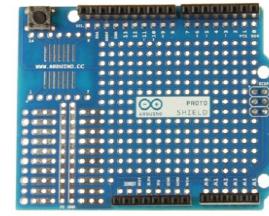
Arduino Mini

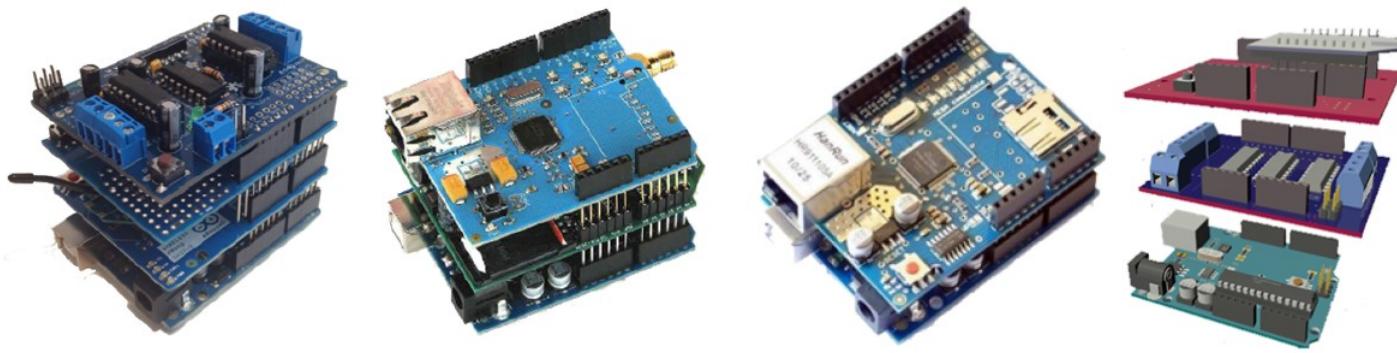
Arduino Shield

- 기본 보드 외에 기능 확장을 위해 사용
- 적층하여 사용

아두이노 쉴드	보드	기능
이더넷 쉴드 (Ethernet Shield)	 The image shows the Arduino Ethernet Shield, which is a blue printed circuit board (PCB) designed to be attached to an Arduino Uno or similar board. It features a W5100 Ethernet chip, an RJ45 port for network connection, and various pins for interfacing with the host Arduino.	LAN선을 연결하여 인터넷에 접속 가능 웹 서버 및 네트워크 제어에 사용
와이파이 쉴드 (WIFI Shield)	 The image shows the Arduino WiFi Shield, a blue PCB with a central ESP8266 WiFi module. It includes pins for power, ground, and serial communication, as well as a small antenna.	무선으로 네트워크에 접속 가능 네트워크를 이용한 제어에 사용
블루투스 쉴드 (Bluetooth Shield)	 The image shows the Arduino Bluetooth Shield, a green PCB featuring a Texas Instruments CC2540 Bluetooth module. It has a large array of pins for digital I/O and a central processing unit (CPU).	블루투스 통신을 이용한 제어에 사용

Arduino Shield

GPS 쉴드 (GPS Shield)		GPS 신호를 수신하여 제어에 사용
모터 쉴드 (MOTOR Shield)		모터제어를 위한 쉴드
기본 쉴드 (Proto Shield)		아두이노 보드와 호환이 되도록 만들어진 기본 쉴드



관련 SW

- 아두이노 통합개발환경 (sketch로 프로그래밍)
 - <http://arduino.cc/en/Main/Software>
- 회로 구성
 - fritzing(<http://fritzing.org>)
 - 브레드보드, 회로도, PCB 버전으로 작업 가능
- 온라인 시뮬레이션
 - 123D circuits(<https://circuits.io>)
 - 회로 및 코드 동작 시뮬레이션 가능
- 아두이노와 연동가능한 프로그래밍 환경
 - Processing (<https://processing.org>)
 - S4A (<http://s4a.cat>) / ScratchX (<http://scratchx.org>)
 - App Inventor (<http://appinventor.mit.edu>)
 - ArduBlock (<https://learn.sparkfun.com/tutorials/alternative-arduino-interfaces/ardublock>)



전류, 전압, 저항

기초 지식



전압, 전류, 저항

- **전류(I)**

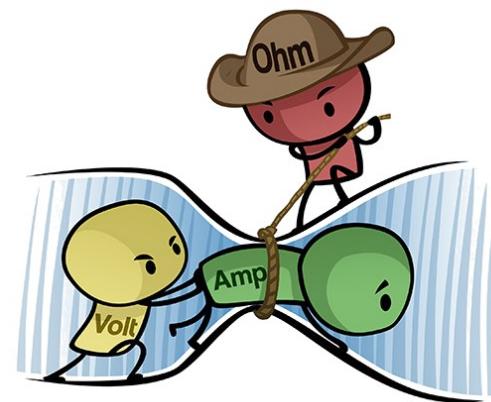
- 단위 시간동안 공간상의 어떤 점을 지나가는 전하의 양
- 단위 A(ampere)

- **전압(V)**

- 도체 내에 있는 두 점 사이의 전기적인 위치에너지 차
- 전류는 전위가 높은 곳에서 낮은 곳으로 흐름
- 단위 V(volt)

- **저항(R)**

- 전류의 흐름을 방해하는 성질
- 단위 Ω (ohm)



옴의 법칙

옴의 법칙(Ohm's law)은 **도체의 두 지점 사이에 나타나는 전위 차에 의해 흐르는 전류**가 일정한 법칙에 따르는 것을 말한다.

두 지점 사이의 도체에 전위차가 존재할 때, 도체의 저항의 크기와 전류는 반비례 관계를 갖는다

도체에 양단에 걸리는 전위차로 단위는 전압(V, volt), R은 도체의 전기저항(resistance)으로 단위는 옴(Ω , ohm)이다.

[출처 : Wiki백과]

$$I = V/R, V = I \times R, R = V/I$$

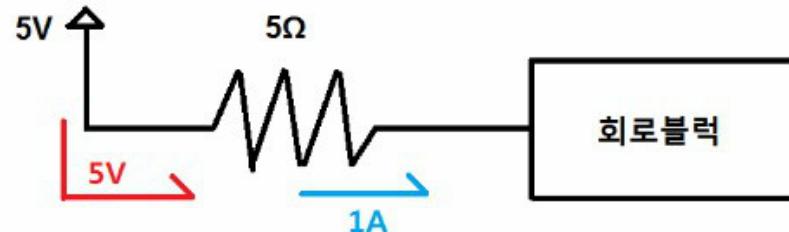
$$R = R_1 + R_2$$



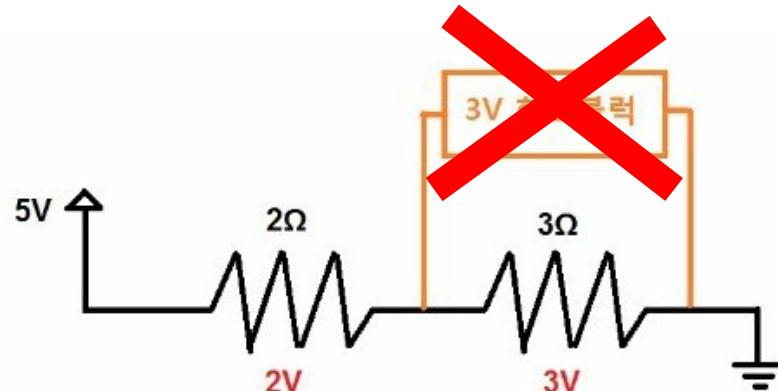
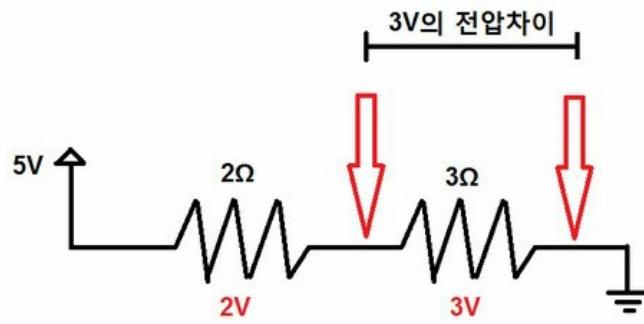
저항의 용도

- 전류 제어

- 과전류를 방지하여
전자부품 보호



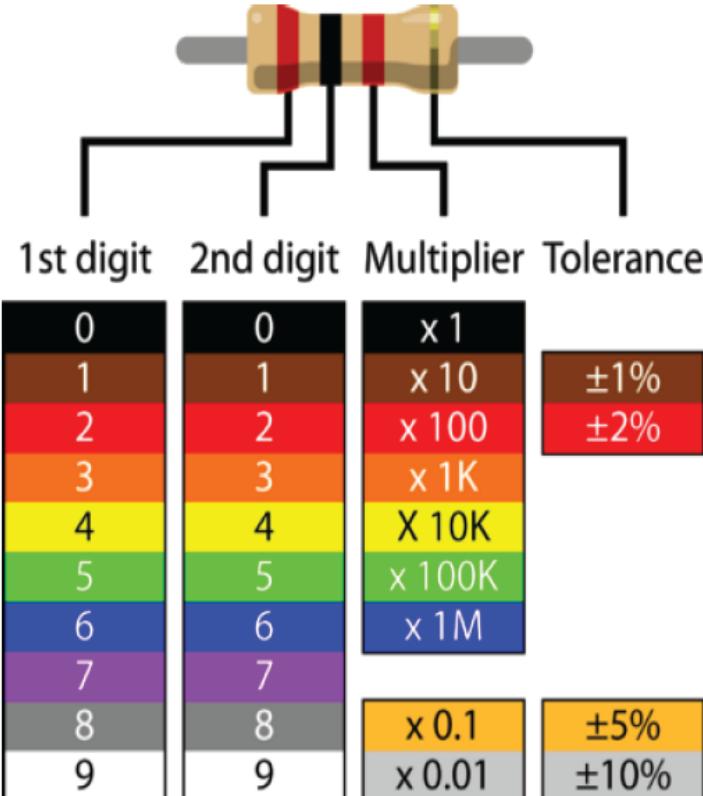
- 전압 제어



$$I = V/R, V = I \times R, R = V/I$$

$$R = R_1 + R_2$$

저항의 크기



- 띠의 개수와 색깔로 구분

- 4개 띠 또는 5개 띠
- 마지막 표시색 : 오차

- **4개 띠**

노랑, 보라, 주황, 금색

$$4 \text{ } 7 \times 1000 = 47000 \Omega \\ = 47K\Omega \text{ (오차 } \pm 5\%)$$

- **5개 띠**

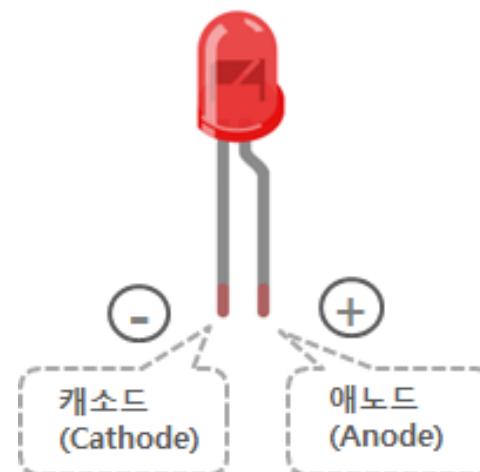
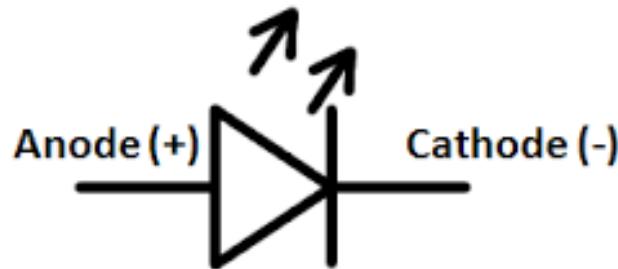
노랑, 보라, 주황, 적색, 은색

$$4 \text{ } 7 \text{ } 3 \times 100 = 47300 \Omega \\ = 47.3K\Omega \text{ (오차 } \pm 10\%)$$

LED (Light Emitting Diode)

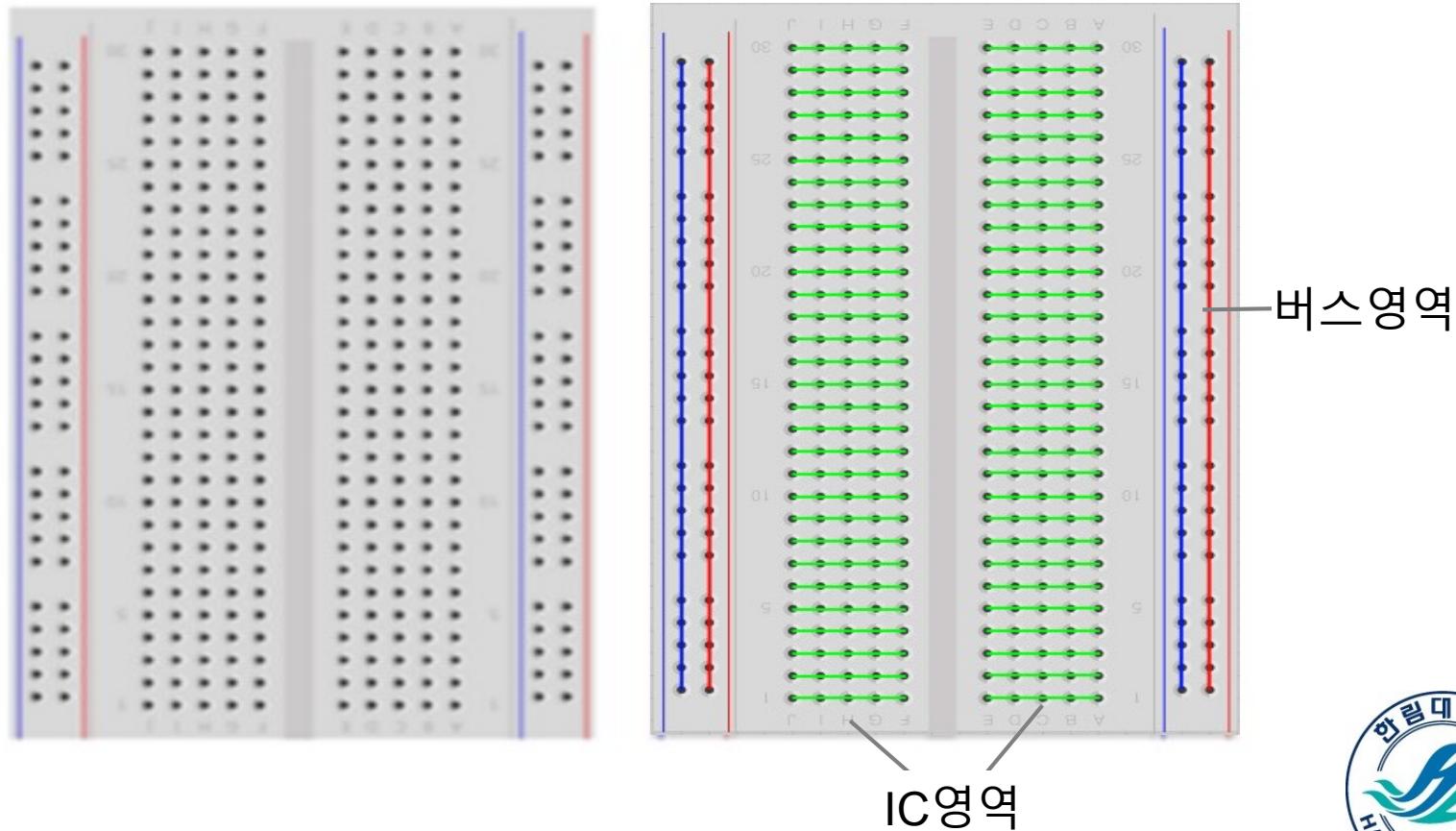
- **LED (Light Emitting Diode)**

- 빛을 내는 반도체 소자
- 반 영구적인 수명(기존 발광체에 비해 열 적게 발생)
- 극성이 있음
 - Anode : 긴 다리, VCC와 연결
 - Cathode : 짧은 다리, GND와 연결
- LED는 항상 저항과 함께 사용



브레드보드

- 쉽게 전자회로를 구성하도록 하는 도구
 - 납땜하지 않고 부품들을 연결



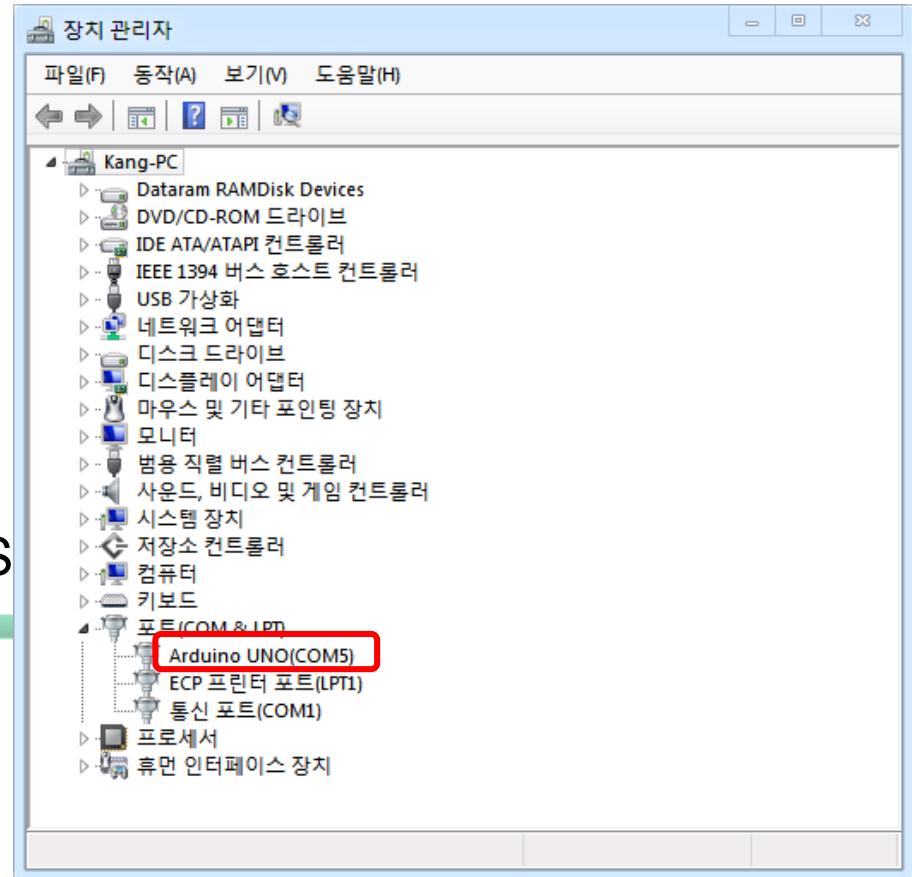
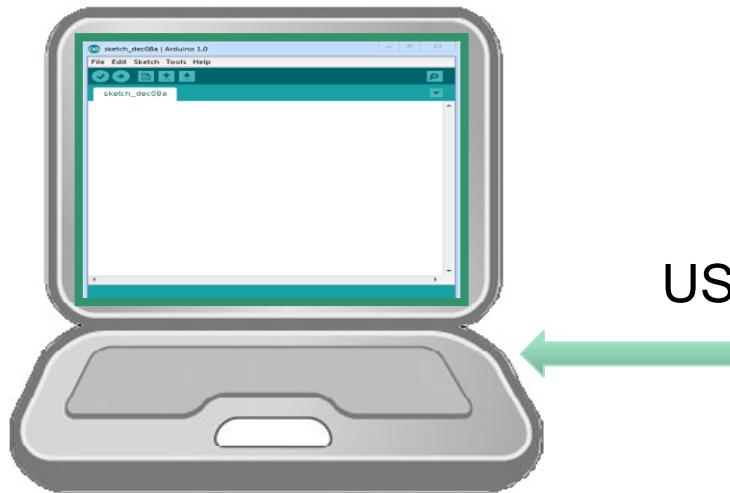
개발환경 구축 및 IDE 사용법

ARDUINO 개발환경



Arduino 개발환경 구축

- PC에 아두이노 보드 사용을 위한 IDE 설치
 - www.arduino.cc에서 IDE 다운로드



Arduino IDE

The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** Blink | 아두이노 1.6.7
- Menu Bar:** 파일 편집 스케치 툴 도움말
- Toolbar:** Includes icons for Open, Save, Upload, and others.
- Sketch Editor:** Displays the "Blink" sketch by Scott Fitzgerald. The code is highlighted with syntax coloring:

```
// the setup function runs once when you press reset or power the board
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

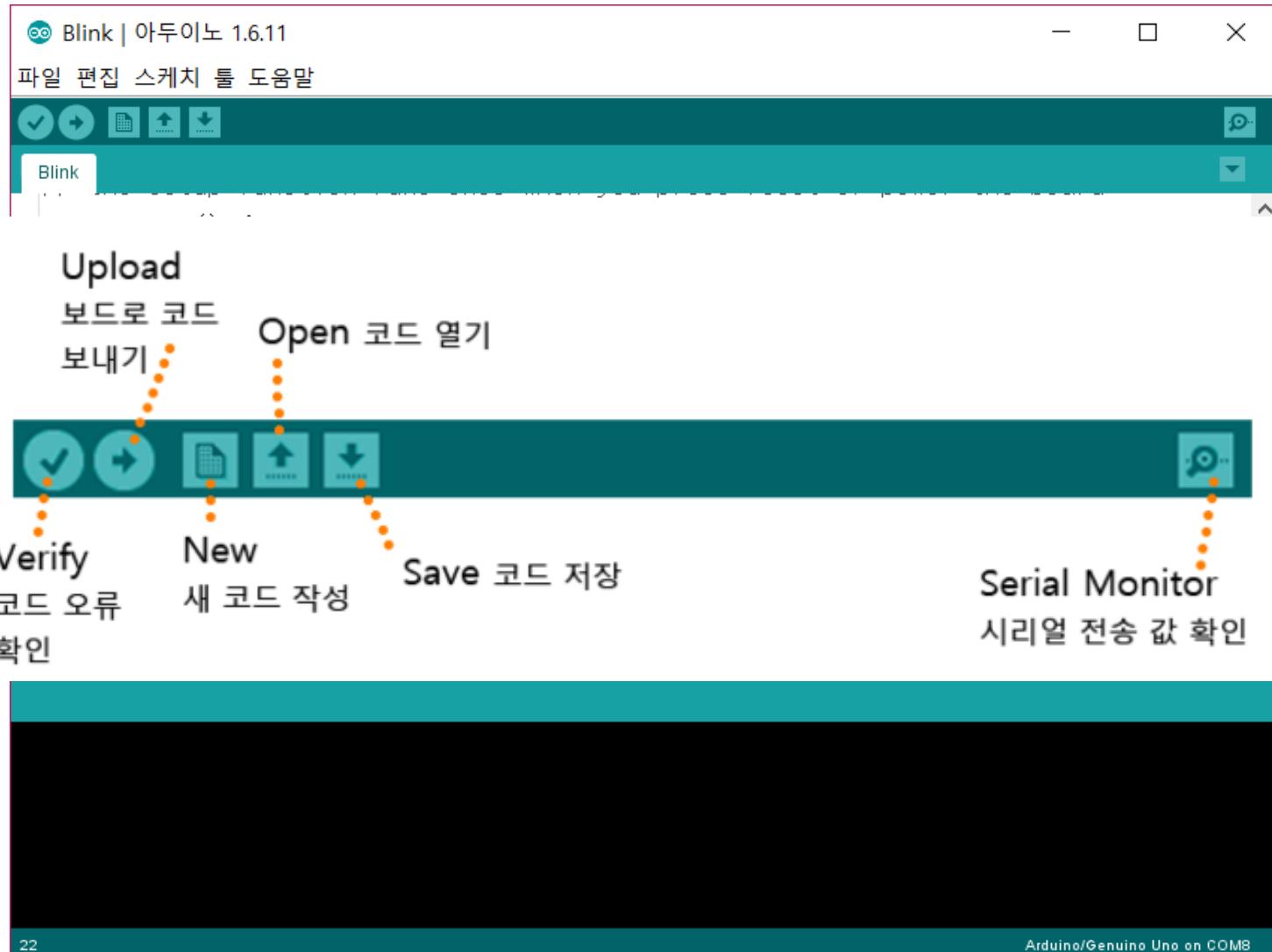
// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH);      // turn the LED on (HIGH is the voltage level)
    delay(1000);                // wait for a second
    digitalWrite(13, LOW);       // turn the LED off by making the voltage LOW
    delay(1000);                // wait for a second
}
```
- Status Bar:** Includes the message "컴파일 완료.", memory usage information ("스케치는 프로그램 저장 공간 (2%) 중 928 바이트를 사용, 최대 32,256 바이트.", "전역 변수는 (0%)의 동적 메모리중 9바이트를 사용, 2,039바이트의 지역변수가 남음."), and the connection status "Arduino/Genuino Uno on COM6".

sketch

Processing 언어에
기반을 둔 프로그래밍
언어



Arduino IDE



Arduino IDE

- **Verify / Compile**
 - 작성한 프로그램 코드가 제대로 되었는지 확인
 - 이상이 없으면 컴파일이라는 과정을 통해 기계가 이해할 수 있는 언어로 변경
- **Upload**
 - 기계가 이해할 수 있는 언어로 바뀐 코드를 아두이노 보드로 전송
- **New**
 - 새로운 스케치 작업을 할 때 사용
- **Open**
 - 기존에 작성된 스케치를 열 때 사용
- **Save**
 - 지금 작성하고 있는 스케치(프로그램 코드) 저장
- **Serial Monitor**
 - 시리얼로 보내고 받는 값을 확인할 때 사용



Arduino Basic Structure

```
void setup() {  
    // 초기화 루틴  
    // setup() 함수, 최초 1회만 실행  
}
```

```
void loop() {  
    // 반복 루틴  
    // loop() 함수, setup 이후 무한반복  
}
```



Arduino Basic Structure

```
#include <Arduino.h>

int main(void) {
    init();                                // 마이크로컨트롤러 초기화
    #if defined(USBCON)                     // USB 장치 연결 성공
        USBDevice.attach();
    #endif

    setup();                            // 사용자 함수 초기화

    for(;;) {
        loop();                         // 사용자 무한 반복 함수
        if(serialEventRun) serialEventRun(); // 시리얼데이터 처리
    }
    return 0;
}
```



Example - Blink

Blink | 아두이노 1.6.11

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
페이지 설정 Ctrl+Shift+P
인쇄 Ctrl+P
환경설정 Ctrl+Comma
종료 Ctrl+Q

```
digitalWrite(13, LOW);
delay(1000);
```

내장된 예제

- 01.Basics > **Blink**
- 02.Digital
- 03.Analog
- 04.Communication
- 05.Control
- 06.Sensors
- 07.Display
- 08.Strings
- 09.USB
- 10.StarterKit_BasicKit
- 11.ArduinoISP

Examples from Libraries

- Bridge
- EEPROM
- Ethernet
- Firmata
- SD
- SoftwareSerial



Example - Blink Sketch

```
void setup() {  
    // initialize the digital pin as an output.  
    // Pin 13 has an LED connected on most Arduino boards:  
    pinMode(13, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(13, HIGH); // set the LED on  
    delay(1000); // wait for a second  
    digitalWrite(13, LOW); // set the LED off  
    delay(1000); // wait for a second  
}
```



Reference

- **Structure**

- setup()
 - loop()

Please Check the SITE:

<http://arduino.cc/en/Reference/HomePage>

- **Functions**

- pinMode (pin, mode) : 사용 할 디지털 pin 번호, 사용 모드
 - digitalWrite (pin, value) : pin 번호에 value 값 출력
 - delay(ms) : 시간 지연 (ex) delay(1000) 1초동안 지연

- **Variables**

- HIGH, LOW, INPUT, OUTPUT



Example - Blink

```
// the setup function runs once when you press reset or power the
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

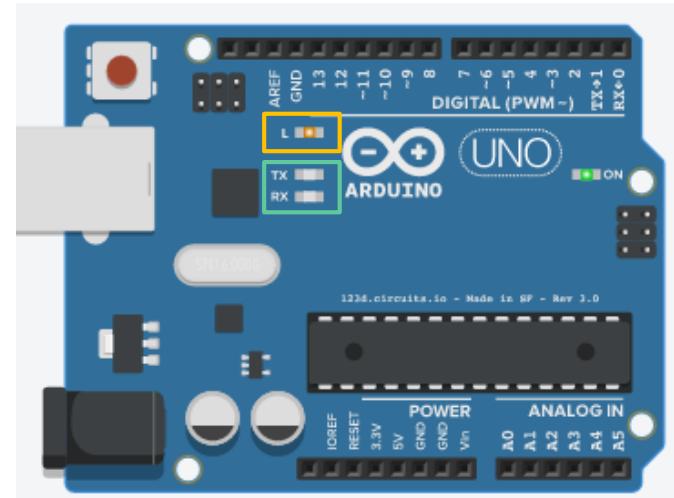
// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level
    delay(1000);           // wait for a second
    digitalWrite(13, LOW); // turn the LED off by making the voltage
    delay(1000);           // wait for a second
}
```

업로드 완료.

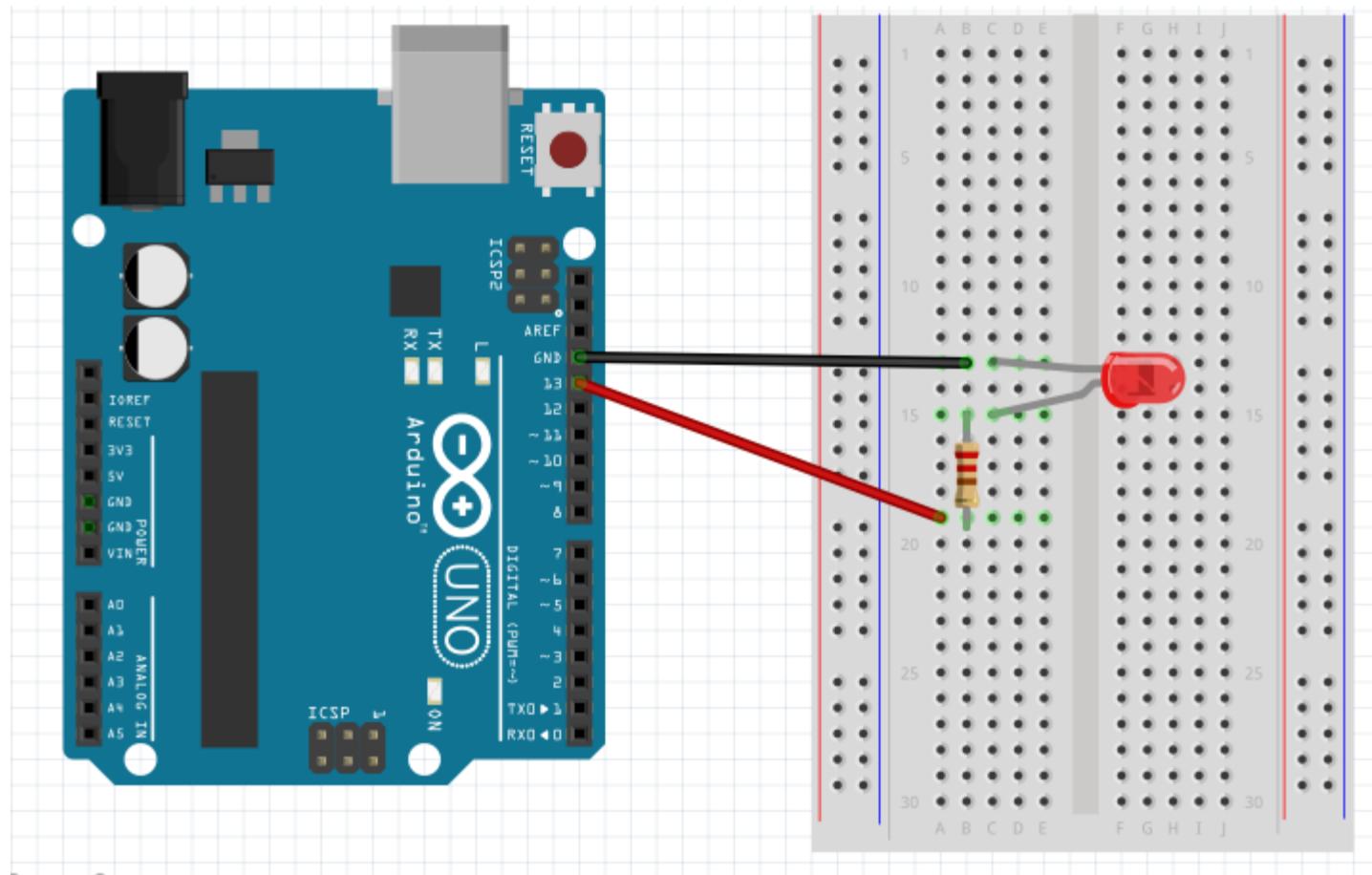
스케치는 프로그램 저장 공간 (2%) 중 928 바이트를 사용. 최대 32,256
전역 변수는 (0%)의 동적 메모리중 9바이트를 사용, 2,039바이트의 지정된
메모리를 사용합니다.

21 Arduino/Genuino Uno on COM3

- 1. Click Verify(compile) and check error**
- 2. Check if Arduino is connected**
- 3. Click Upload**



Example - Blink



TinkerCAD Circuits 01 용한 Blink

riverlike

안전함 | <https://circuits.io/circuits/3813011-led1/edit>

All changes saved

Code Editor Components Start Simulation

LED1

Resistor

Name	1
resistance	220

Search

All Components Grid All Components List Arduino Basic Kit DFRobot Beginner Kit

Resistor

LED

LED RGB

Light bulb



TinkerCAD Circuits 01용한 Blink

TINKER CAD

desk Circuits

LED1

All changes saved

Code Editor Components Start Simulation

Resistor

Name 1
resistance 220

1 (Arduino Uno R3) Upload & Run Libraries Download Code Debugger Serial Monitor

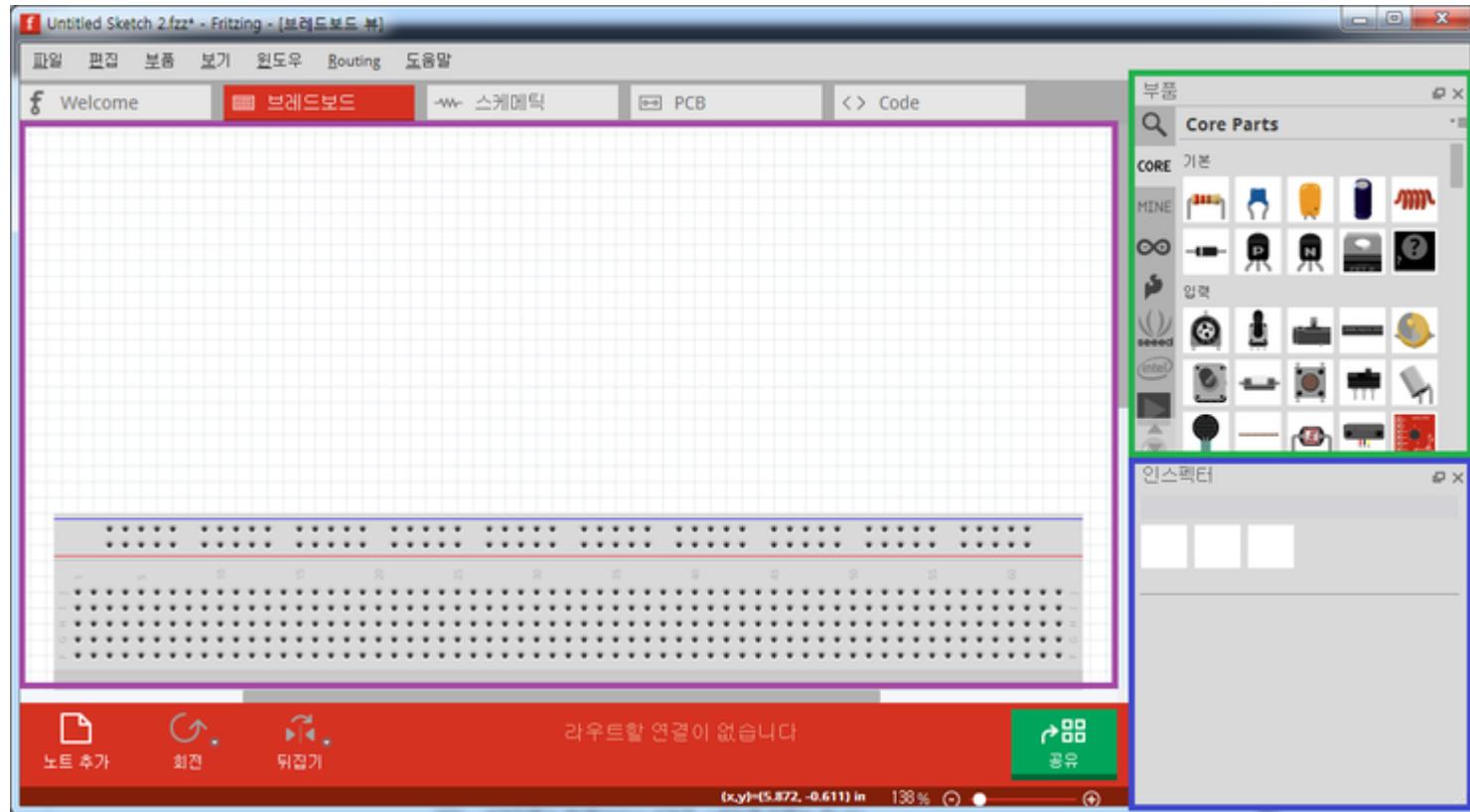
```

1 int led = 8;
2
3 void setup() {
4     // initialize the digital pin as an output.
5     pinMode(led, OUTPUT);
6 }
7
8 void loop() {
9     digitalWrite(led, HIGH);      // turn the LED on (HIGH is the voltage level)
10    delay(1000);                // wait for a second
11    digitalWrite(led, LOW);      // turn the LED off by making the voltage LOW
12    delay(1000);                // wait for a second
13 }
14

```



Fritzing 이용한 Blink



- **보라색영역**
 - 브래드보드, 회로 설계할 도면
- **초록색영역**
 - 부품, 그림을 그릴 때 사용할 부품
- **파란색영역**
 - 인스펙터, 선택한 부품의 크기와 값 등을 설정할 수 있는 영역

Fritzing 이용한 Blink

Led1.fzz* - Fritzing - [브레드보드 뷰]

파일 편집 보품 보기 원도우 Routing 도움말

Welcome 브레드보드 스케메틱 PCB Code

부품

Core Parts

CORE MINE

인스펙터

Breadboard1 v. 4

Placement

- location 1,826 -0,214 in
- rotation 90,0 degrees 고정

속성

- 패밀리 breadboard
- 사이즈 half+
- 부품 번호
- 태그 breadboard
- 연결

fritzing

노트 추가 회전 뒤집기

각우팅 완료

(x,y)=(0.904, -0.333) in 150% +

ALLYM UNIVERSITY

Fritzing 이용한 Blink

Led1.fzz* - Fritzing - [브레드보드 뷰]

파일 편집 Code 보기 윈도우 도움말

Welcome 브레드보드 스케메틱 PCB Code

```
led1.ino
```

```
int led = 8;

void setup() {
    // initialize the digital pin as an output.
    pinMode(led, OUTPUT);
}

void loop() {
    digitalWrite(led, HIGH);      // turn the LED on (HIGH is the voltage level)
    delay(1000);                // wait for a second
    digitalWrite(led, LOW);       // turn the LED off by making the voltage LOW
    delay(1000);                // wait for a second
}
```

부품

Core Parts

CORE MINE INFINITE SPARKLE

seed intel

총력

인스펙터

Breadboard1 v. 4

Breadboard1

Placement

location 1,826 -0,214 in
rotation 90,0 degrees 고정

속성

패밀리 breadboard
사이즈 half+
부품 번호
태그 breadboard
연결

*

새로만들기 Open 저장

Platform Arduino Arduino UNO 보드

Port COM3 Port Serial Monitor Upload

150 %

LED 제어를 통한 디지털/아날로그 출력



디지털 신호 출력

void setup() {	아두이노 보드의 핀 설정
pinMode(13, OUTPUT);	13번을 디지털출력으로 설정
}	setup() 함수 종료
void loop() {	loop() 함수의 시작, 내부의 명령이 무한 반복
digitalWrite(13, HIGH);	디지털입출력 핀 13번에 HIGH(1) 출력
delay(1000);	1000ms 동안 시간지연
digitalWrite(13, LOW);	디지털입출력 핀 13번에 LOW(0) 출력
delay(1000);	1000ms 동안 시간지연
}	loop() 함수의 끝

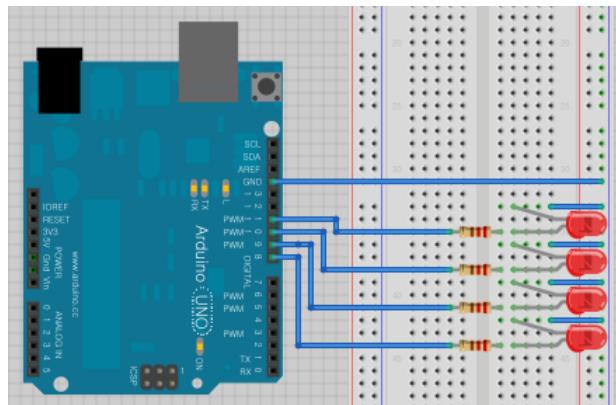


LED 4개 ON/OFF 제어

- 변수를 사용하여 입출력 핀과 시간지연 변경
 - loop() 밖에서 전역변수로 선언
- 4개의 LED 동시제어



- 디지털핀 8,9,10,11번
- 1초 간격으로 깜빡임



```

int digitalPin = 10;
int delayTime = 500;

void setup() {
    pinMode(digitalPin, OUTPUT);
}

void loop() {
    digitalWrite(digitalPin, HIGH);
    delay(delayTime);
    digitalWrite(digitalPin, LOW);
    delay(delayTime);
}

```

변수와 for문 이용

- LED 4개 ON/OFF 제어

int i;	for문에 사용하기 위하여 i를 정수형 변수로 선언
void setup() { for(i=8; i<12; i++) { pinMode(i, OUTPUT); } }	8~11번 핀 디지털출력으로 지정
void loop() { for(i=8; i<12; i++) { digitalWrite(i, HIGH); } delay(1000); for(i=8; i<12; i++) { digitalWrite(i, LOW); } delay(1000); }	8~11번 핀 HIGH출력 시간지연 1초 8~11번 핀 LOW출력 시간지연 1초



배열과 for문 이용

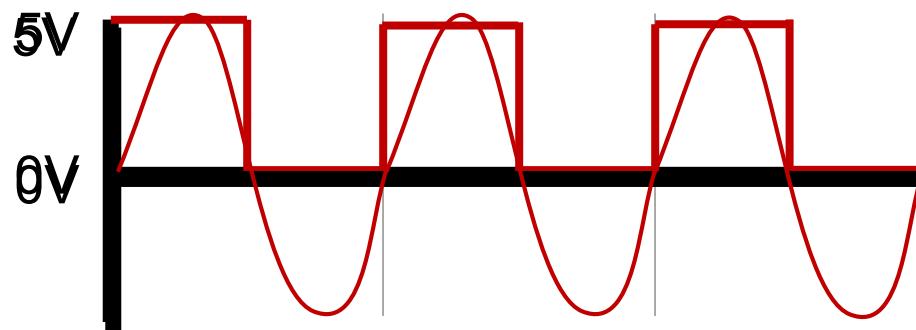
- LED 4개 ON/OFF 제어

<pre>int pinLED[] = {8, 9, 10, 11}; int i;</pre>	배열 pinLED[]를 정수형으로 선언하고 값을 지정
<pre>void setup() { for(i=0; i<4; i++) { pinMode(pinLED[i], OUTPUT); } }</pre>	pinLED[0]인 8번핀을 디지털출력으로 지정 pinLED[1]인 9번핀을 디지털출력으로 지정 pinLED[2]인 10번핀을 디지털출력으로 지정 pinLED[3]인 11번핀을 디지털출력으로 지정
<pre>void loop() { for(i=0; i<4; i++) { digitalWrite(pinLED[i], HIGH); } delay(1000); for(i=0; i<4; i++) { digitalWrite(pinLED[i], LOW); } delay(1000); }</pre>	8~11번 핀 HIGH 출력 시간지연 1초 8~11번 핀 LOW 출력 시간지연 1초



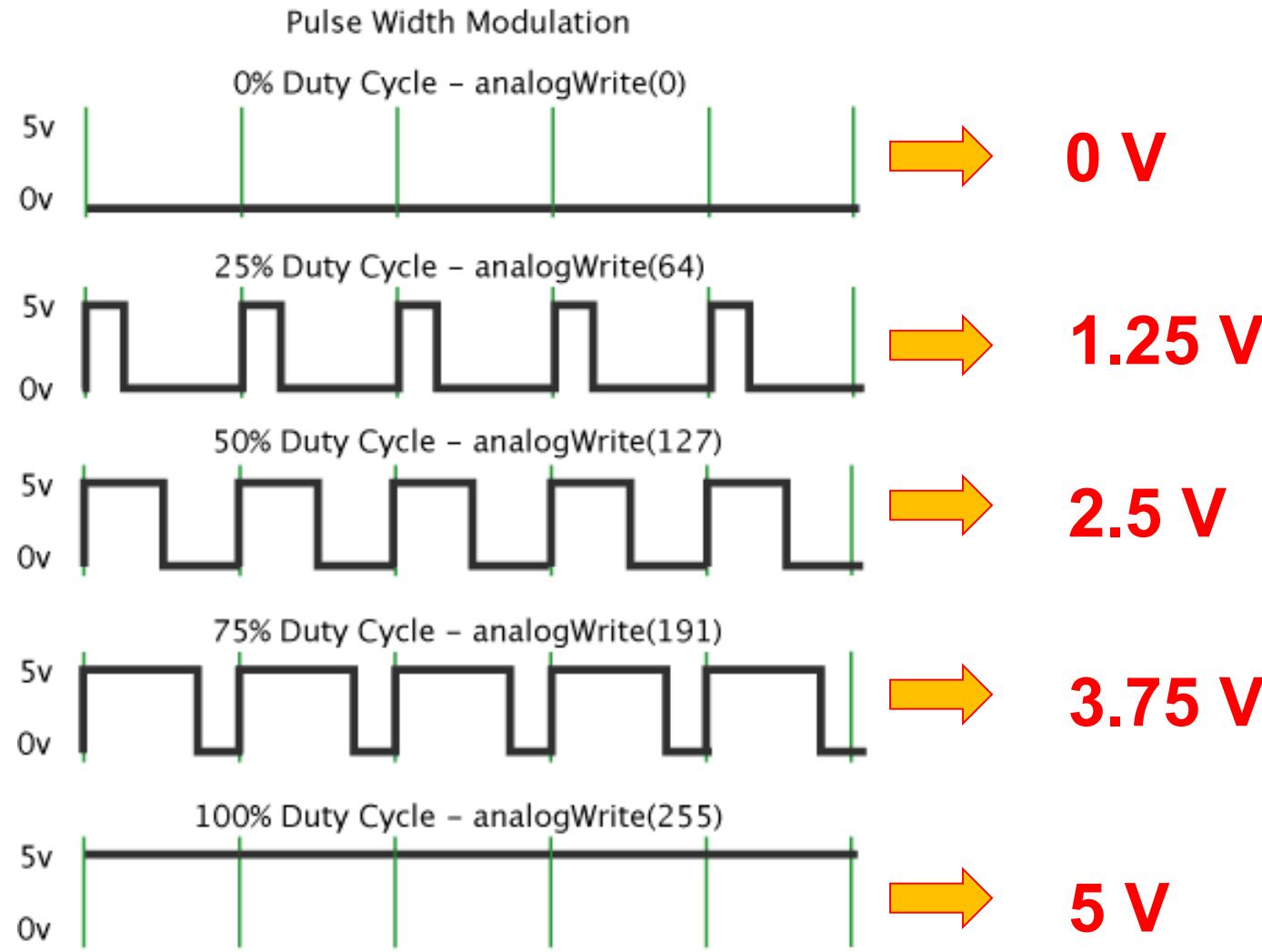
아날로그 신호 출력

- **PWM을 이용한 Fading**
 - 틸트(~)가 붙은 3,5,6,9,10,11번 핀
 - LED밝기 조절
 - 모터 속도 제어
- **PWM(Pulse Width Modulation)**
 - 디지털신호로 아날로그신호를 표현하는 방식
 - 펄스 폭 조절하여 전압의 세기 표현



PWM

- 한 주기 내의 ON상태인 시간의 비율 조절



LED 부드럽게 끄고 부드럽게 켜기

- **analogWrite()** 이용

- 0~255 사이의 8bit값을 PWM파형으로 출력

int ledPin=9; int fade=0;	
void setup() { pinMode(ledPin, OUTPUT); }	9번핀 출력모드로 설정(생략가능)
void loop() { for (fade=0; fade<=255; fade+=5) { analogWrite (ledPin, fade); delay(30); } for (fade=255; fade>=0; fade-=5) { analogWrite (ledPin, fade); delay(30); } }	8번 핀으로 analogWrite()로 0에서 255까지 서서히 켜지는 PWM 신호 출력 8번 핀으로 analogWrite()로 255에서 0까지 서서히 줄어드는 PWM 신호 출력



함수정리

	디지털 핀에 대하여 입력모드와 출력 모드를 지정
pinMode(pin, state);	<ul style="list-style-type: none"> - pin : 핀 번호 (0~13) - state : 출력은 OUTPUT, 입력은 INPUT 설정
	디지털출력으로 선언된 핀에 대하여 HIGH(1)와 LOW(0) 출력
digitalWrite(pin, state);	<ul style="list-style-type: none"> - pin : 디지털 출력으로 선언된 핀 번호 - state : HIGH 또는 1, LOW 또는 0
	시간지연 함수
delay(milli-seconds);	<ul style="list-style-type: none"> -milli-seconds : 숫자 값이 밀리초를 의미
	아날로그 출력핀을 통해 0~255사이 PWM 신호 출력
analogWrite(pin, value);	<ul style="list-style-type: none"> - pin : PWM 핀 번호 - value : 출력범위 0~255



Serial 통신, 버튼 SWITCH

SWITCH 제어를 통한
디지털입력



Serial 통신

- 두 기기 간의 일대일 통신
 - 아두이노 \Leftrightarrow PC 또는 아두이노 \Leftrightarrow 아두이노
 - 아두이노 보드는 하나의 직렬포트 제공
 - USB를 통해 디지털 핀 0(RX), 1(TX) 이용하여 통신
- 시리얼 모니터를 통해 통신 확인

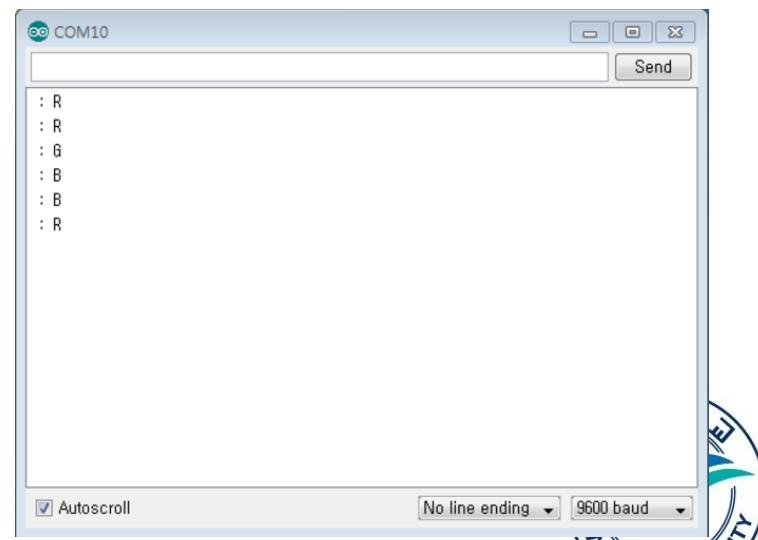
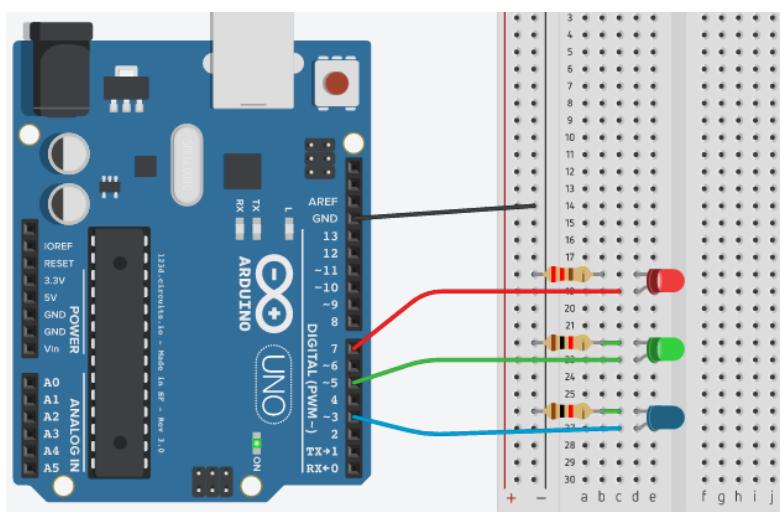
시리얼 모니터 

Serial.begin()	시리얼 통신을 위한 속도 설정
Serial.end()	시리얼 통신 종료
Serial.available()	시리얼포트에서 읽을 수 있는 바이트 수 가져옴
Serial.print()	시리얼 포트를 통해 데이터 송신 (사람이 읽을 수 있는 아스키코드로 해석해서 전송)
Serial.println()	개행문자와 함께 데이터 송신(문자출력 후 개행)
Serial.read()	시리얼 포트를 통해 데이터 수신



- 시리얼통신을 통해 R,G,B 3LED ON/OFF

- 빨강, 녹색, 파랑 LED : 7, 5, 3번에 디지털출력으로 연결
 - シリ얼모니터에 데이터 입력
 - 입력 문자 'R' 이면 빨강 LED 켜기
 - 입력 문자 'G' 이면 녹색 LED 켜기
 - 입력 문자 'B' 이면 파랑 LED 켜기
 - 입력 문자 'X' 이면 모든 LED 끄기



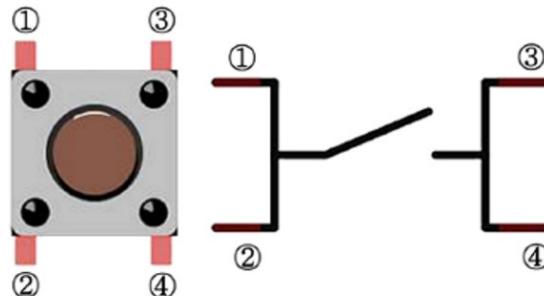
Serial 통신

#define LED_R 7 #define LED_G 5 #define LED_B 3 void setup() { Serial.begin(9600); pinMode(LED_R, OUTPUT); pinMode(LED_G, OUTPUT); pinMode(LED_B, OUTPUT); } void loop() { if(<i>Serial.available()</i>) { char com_in = <i>Serial.read()</i> ; switch(<i>com_in</i>) { case 'R': digitalWrite(LED_R, HIGH); break; case 'G': digitalWrite(LED_G, HIGH); break; case 'B': digitalWrite(LED_B, HIGH); break; case 'X': digitalWrite(LED_R, LOW); digitalWrite(LED_G, LOW); digitalWrite(LED_B, LOW); break; } Serial.println(<i>com_in</i>); } delay(100); }	3, 5, 7번핀 디지털 출력으로 설정 9600baud으로 통신 준비
	시리얼모니터에 입력이 있는 경우 수신버퍼에서 한 바이트 읽어 com_in에 저장 com_in이 'R' 이면 빨강 LED ON com_in이 'G' 이면 녹색 LED ON com_in이 'B' 이면 파랑 LED ON com_in이 'X' 이면 LED 모두 OFF



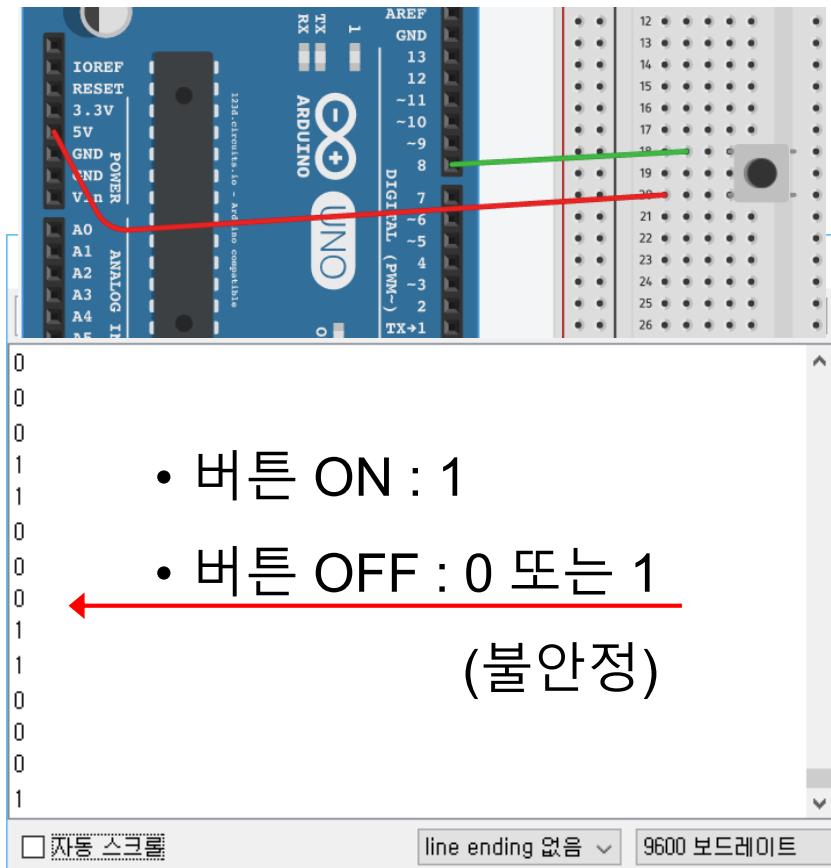
디지털 입력

- 버튼 스위치



- 디지털 신호 입력 받기
 - 8번 핀 버튼 스위치 연결

```
void setup() {  
    pinMode(8, INPUT);  
    Serial.begin(9600);  
}  
  
void loop() {  
    int state = digitalRead(8);  
    Serial.println(state);  
}
```

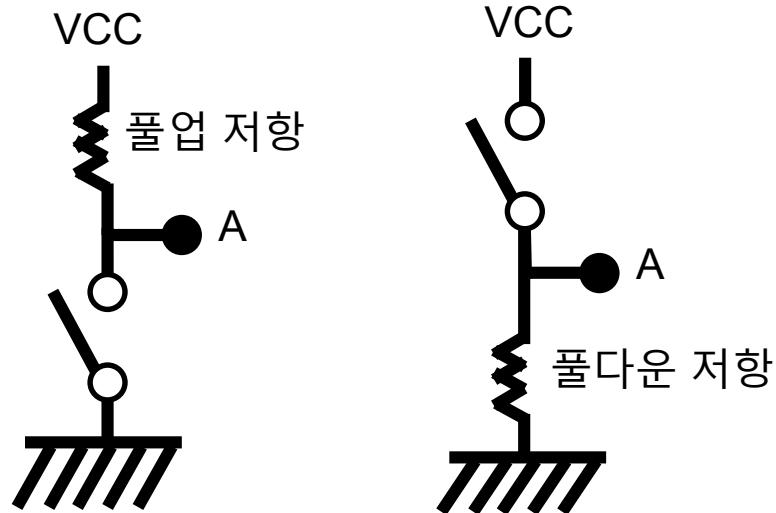


Floating 현상

- 전압 값이 불안정하여 오작동이 일어나는 현상
 - 1도 아니고 0도 아닌 애매한 상태

- 해결방법

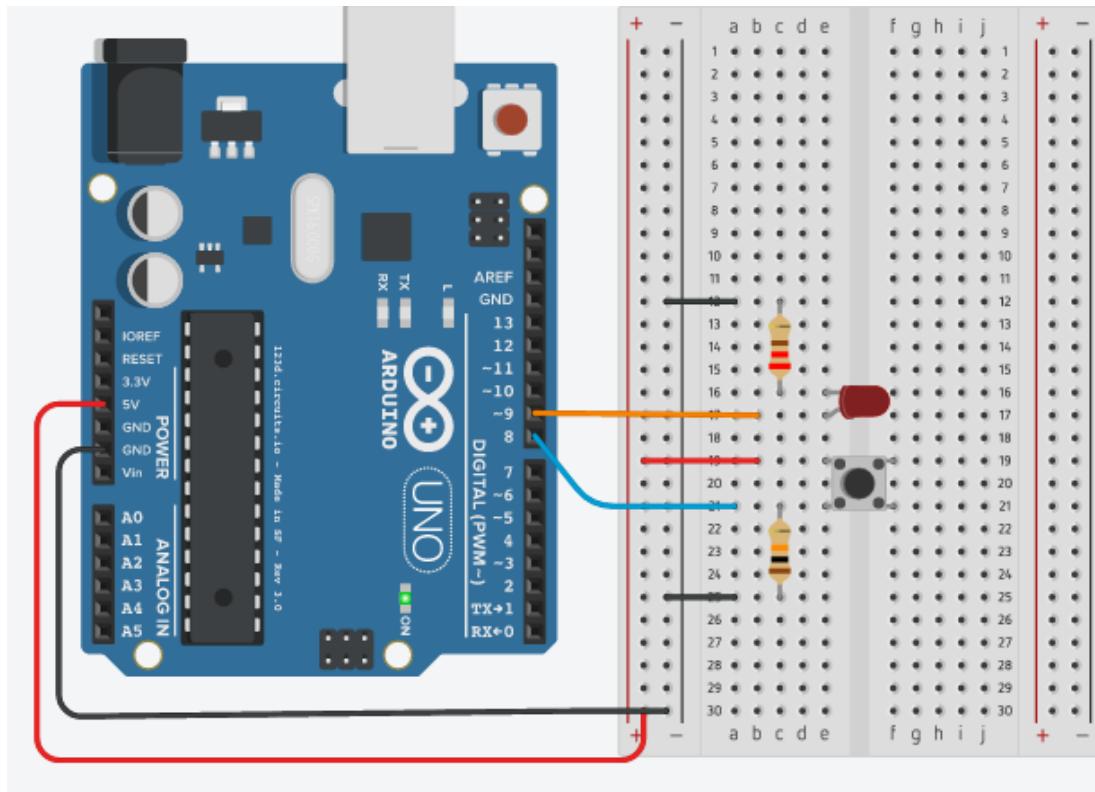
- 풀업 저항
 - VCC(5V) 쪽에 저항 연결
- 풀다운 저항
 - GND 쪽에 저항 연결
- MCU 내부저항
 - 별도의 저항 연결 불필요



	ON	OFF
Pull Up	0V	5V
Pull Dn	5V	0V

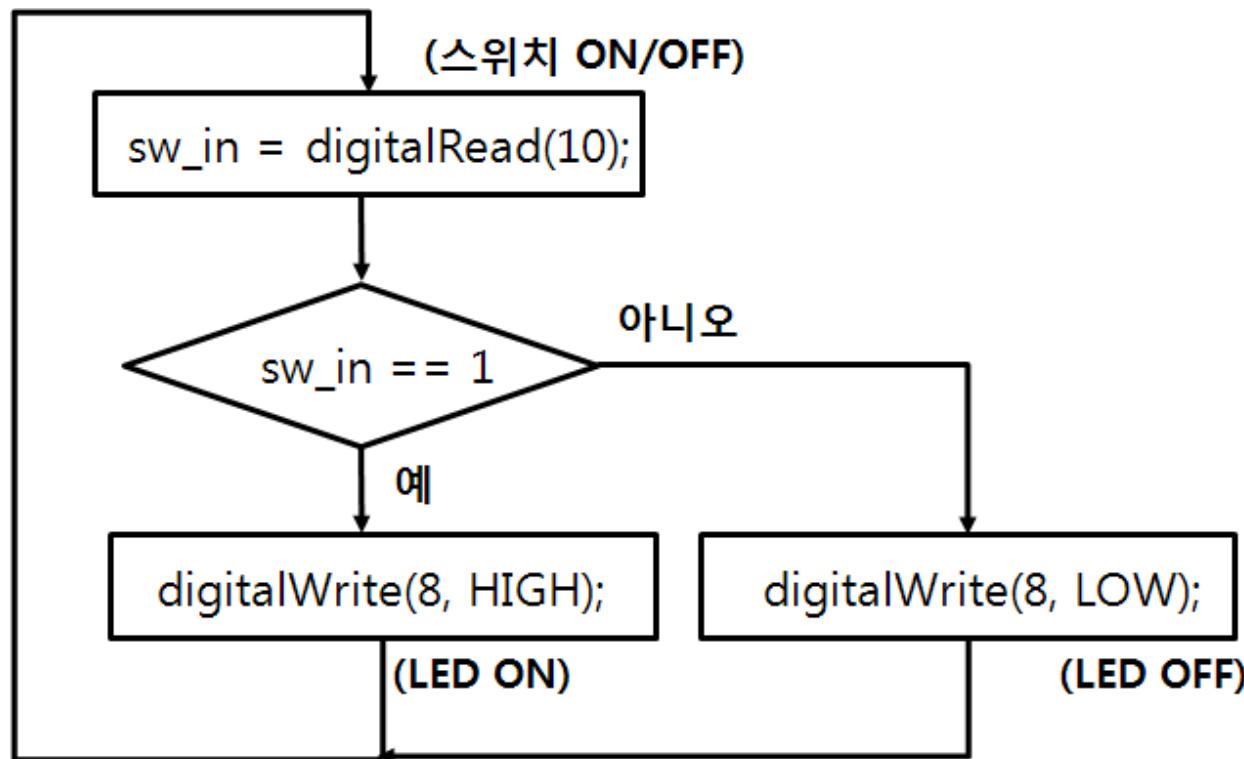
버튼스위치를 이용한 LED ON/OFF

- 버튼스위치 : 10번 핀
– Pull Down 저항 사용($10\text{K}\Omega$)
- LED : 8 번 핀(220Ω)



버튼스위치를 이용한 LED ON/OFF

- 버튼이 눌리면 LED ON, 그렇지 않으면 LED OFF



버튼스위치를 이용한 LED ON/OFF

int pin_led=8; int pin_sw=10;	정수형 변수 pin_led와 pin_sw 선언
void setup() { pinMode(pin_led, OUTPUT); pinMode(pin_sw, INPUT); }	8번(pin_led) 핀 디지털 출력으로 설정 10번(pin_sw) 핀 디지털 입력으로 설정
void loop() { int sw_in=digitalRead(pin_sw); if(sw_in==1) digitalWrite(pin_led, HIGH); else digitalWrite(pin_led, LOW); }	10번(pin_sw) 핀으로부터 값을 읽어 sw_in저장 sw_in이 1 이면 LED ON sw_in이 1 아니면 LED OFF



버튼스위치를 이용한 LED ON/OFF

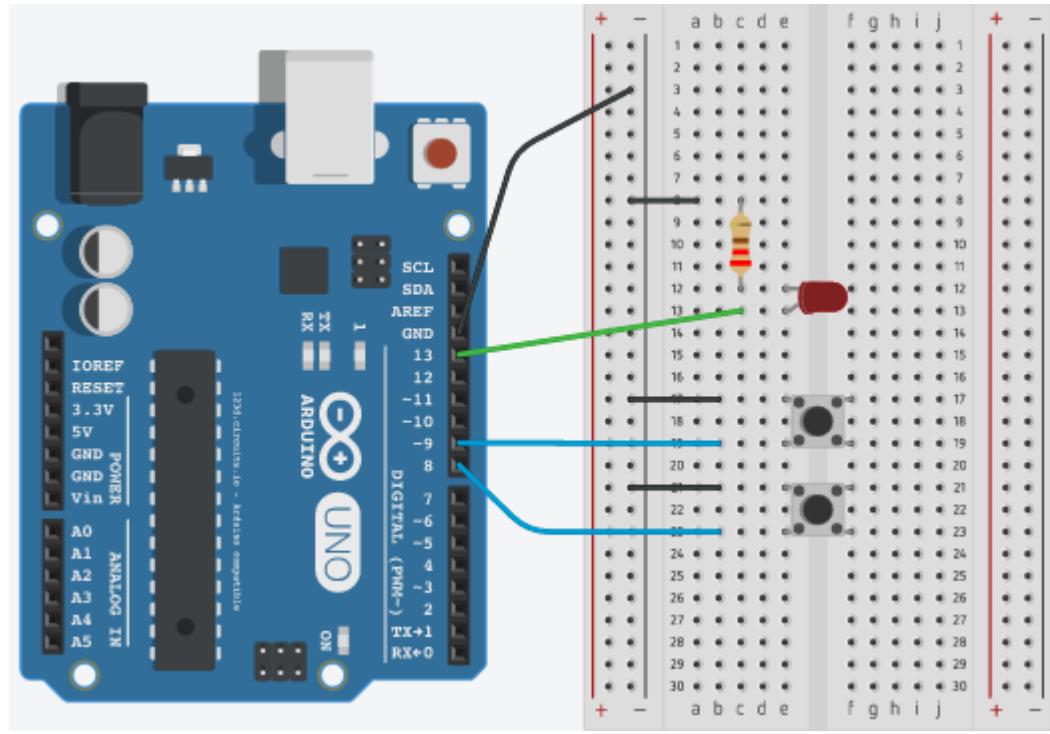
- 버튼이 눌리면 LED ON/OFF 상태 변화하기

<pre>int pin_led=8; int pin_sw=10; int pState = LOW</pre>	정수형 변수 pin_led와 pin_sw 선언 LED상태 pState의 초기값 LOW (OFF 상태)
<pre>void setup() { pinMode(pin_led, OUTPUT); pinMode(pin_sw, INPUT); }</pre>	8번(pin_led) 핀 디지털 출력으로 설정 10번(pin_sw) 핀 디지털 입력으로 설정
<pre>void loop() { int sw_in=digitalRead(pin_sw);</pre>	10번(pin_sw) 핀으로부터 값을 읽어 sw_in저장
<pre>if(sw_in==HIGH && pState==LOW) { digitalWrite(pin_led, HIGH); pState = HIGH; delay(100); } else if(sw_in==HIGH && pState==HIGH) { digitalWrite(pin_led, LOW); pState = LOW; delay(100); }</pre>	sw_in가 HIGH이고 LED가 Off상태이면, LED ON LED 상태 ON으로 변경 sw_in가 HIGH이고 LED가 On상태이면, LED OFF LED 상태 OFF로 변경
<pre>}</pre>	



내부 PULLUP저항

- 버튼 스위치 2개와 LED 1개 사용
 - 버트스위치에 별도의 저항 사용 X
 - 1번 버튼 스위치 누르면 LED ON
 - 2번 버튼 스위치 누르면 LED OFF



내부 PULLUP저항

<pre>int ledPin = 13; int buttonApin = 9; int buttonBpin = 8;</pre>	<p>정수형 변수 pin_led와 pin_sw 선언</p>
<pre>void setup() { pinMode(ledPin, OUTPUT); pinMode(buttonApin, INPUT_PULLUP); pinMode(buttonBpin, INPUT_PULLUP); }</pre>	<p>13번 핀 디지털 출력으로 설정 8번 핀 내부PULLUP 입력으로 설정 9번 핀 내부PULLUP 입력으로 설정</p>
<pre>void loop() { if(digitalRead(buttonApin)==LOW) digitalWrite(pin_led, HIGH); if(digitalRead(buttonBpin)==LOW) digitalWrite(pin_led, LOW); }</pre>	<p>8번 버튼이 눌리면(LOW) LED ON 9번 버튼이 눌리면(LOW) LED OFF LED OFF</p>



가변저항, 조도센서, 온도센서
피에조부저

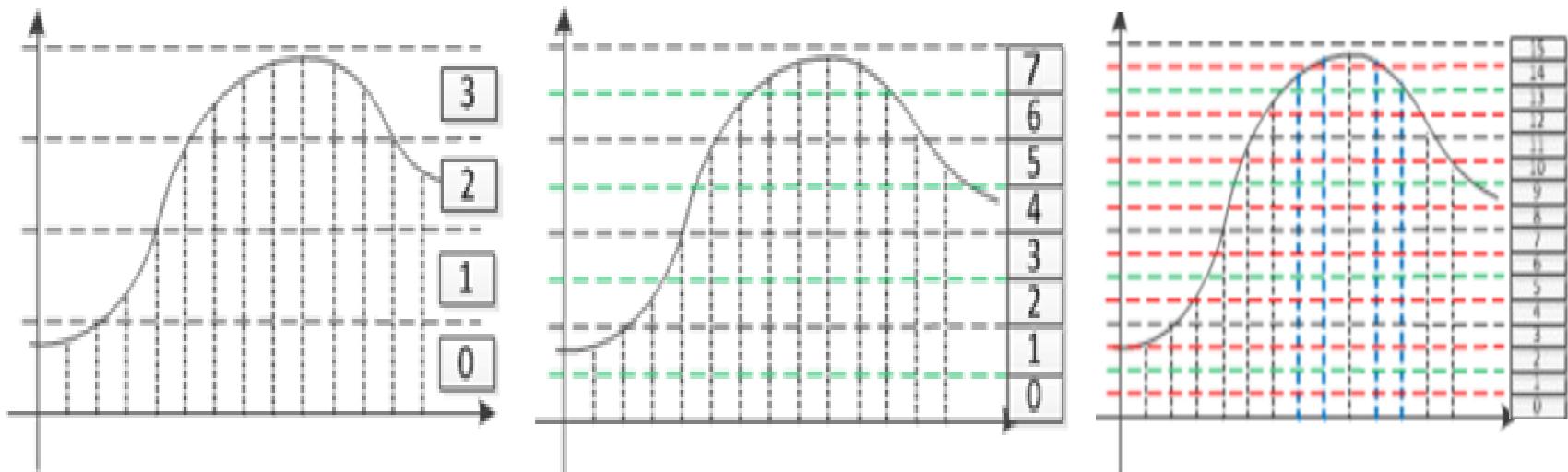
가변저항 제어를 통한 아날로그 입력



아날로그 입력

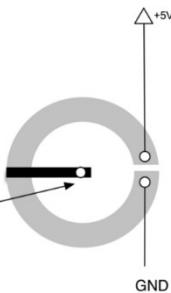
• ADC(Analog Digital Converter) 사용

- 아날로그 입력 핀을 통하여 외부로부터 0 ~ 5V 값 입력
- 10bit의 디지털 값으로 변환 : 0 ~ 1024 사이

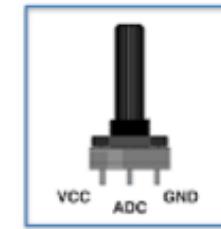
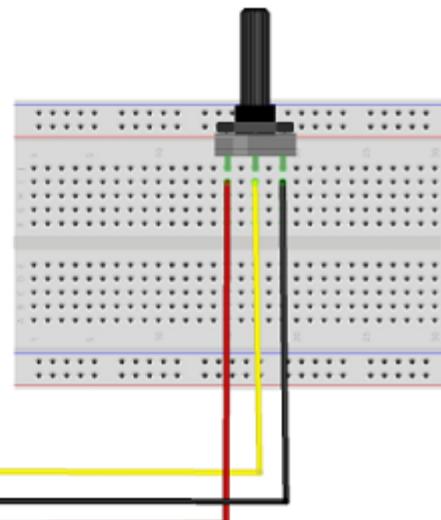
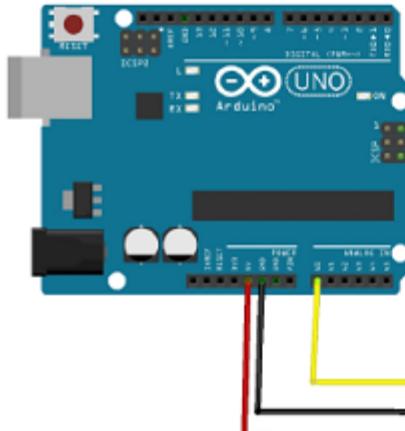


가변저항(Potentiometer)

- 저항을 임의로 바꿀 수 있는 저항기
 - 소리 크기 조절 및 빛의 밝기 조절에 사용
 - 손잡이의 회전각에 따라 저항값 변경

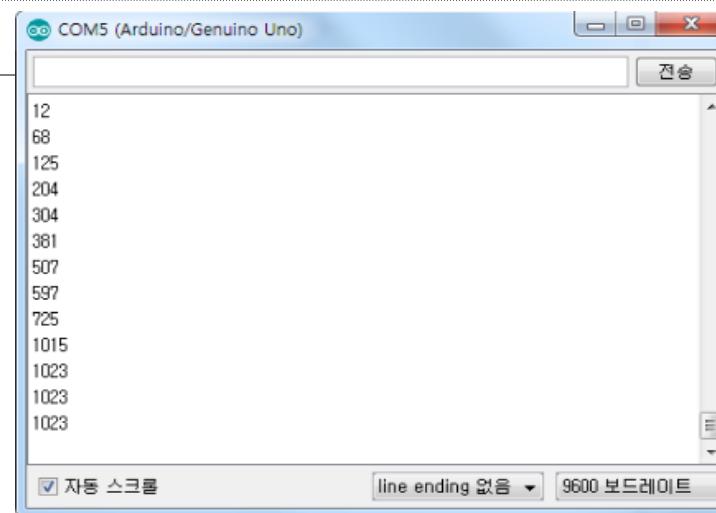


- 가변저항 읽기



가변저항(Potentiometer)

const int analogPin = 0;	
void setup() {	
Serial.begin(9600);	시리얼 통신을 위한 전송속도 9600 설정
}	
void loop() {	
int val; val = analogRead(analogPin);	아날로그 입력핀 2번을 통하여 입력된 전압값을 0~1023사이의 값으로 읽어 val에 저장
Serial.println(val);	시리얼 통신을 통하여 모니터에 val 값 출력
}	



가변저항으로 LED 밝기 조절

```

int analogPin = 0;
int ledPin = 9;

void setup() {
}

void loop() {
    int val;
    int light;

    val = analogRead(analogPin);

    light = map(val,0,1023,0,255);

    analogWrite(ledPin, light);
    delay(1000);
}

```

변수 선언

아날로그 입력핀 0번을 통해 입력된 전압값을
0~1023사이의 값으로 읽어 val에 저장

0~1023범위의 val값을 0~255사이의 값으로
변환하여 light에 저장

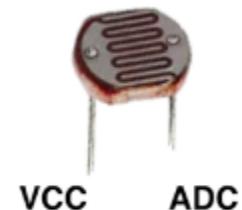
아날로그 출력 9핀을 PWM 형태로 LED 밝기 표현
analogWrite(9, 127);은 2.5V 전압 설정



센서값 읽기

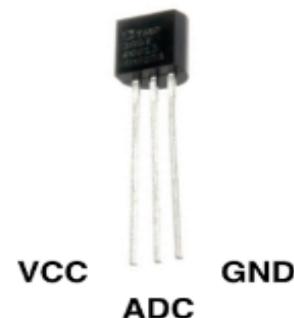
- 조도센서

- Photo Cell(Resistor) 또는 CDS(Cadmium Sulfide)
- 빛의 밝기 변화를 인식하는 센서
 - 저항($10k\Omega$) + 조도센서 : 가변저항처럼 동작
- 빛의 양에 따라 저항 값 변화
 - 주위가 밝으면 저항값 감소, 어두우면 저항값 증가



- 온도센서

- 물체의 온도를 감지하여 전기신호로 바꿔주는 센서
- TMP36 온도센서
 - 온도에 따른 전압의 변화량을 이용
 - $-40^{\circ}\text{C} \sim 120^{\circ}\text{C}$



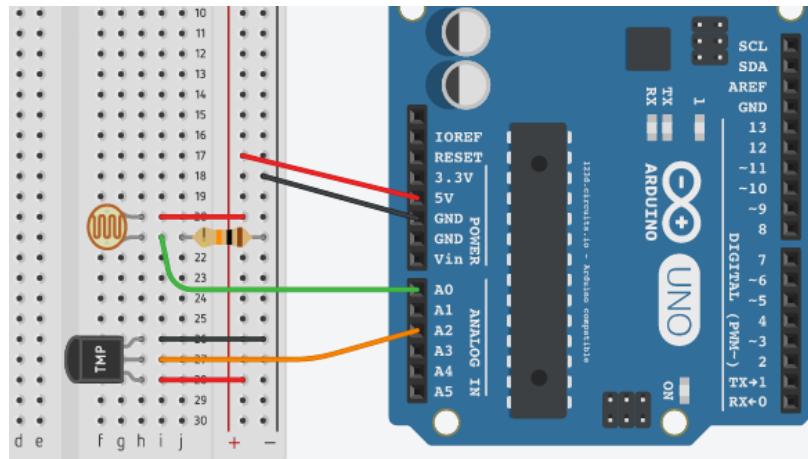
조도센서와 온도센서 값 확인하기

- 아날로그 입력 연결

- 조도센서 : A0
- 온도센서 : A2

- 시리얼모니터 출력

- 조도센서 값
- 온도센서 값

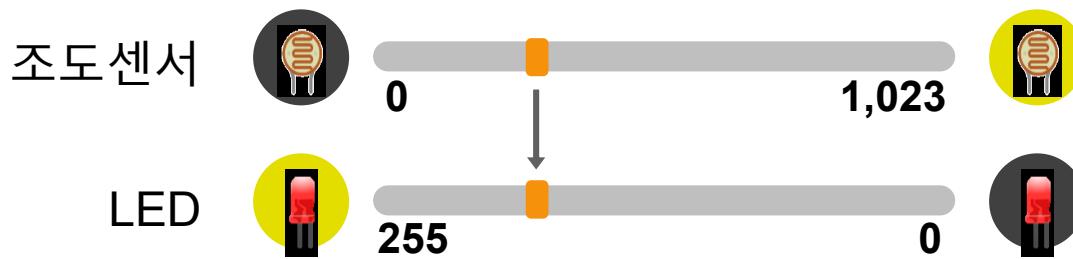


```
int cdsvalue = analogRead(CDS);
int tmpvalue = analogRead(TMP);
```

```
Serial.print("CDS : ");
Serial.print(cdsvalue);
Serial.print("₩t");
Serial.print("TMP : ");
Serial.println(tmpvalue);
```

조도센서와 LED

- 어두워지면 LED 켜기
 - 조도 측정 후 어두우면 LED 켜기
 - 측정값이 300 미만이면 LED ON, 아니면 LED OFF
- 주변밝기에 따라 LED 밝기 제어하기
 - 조도센서 값의 범위를 LED 출력의 범위로 변환



- 어두운 정도에 따라 LED 밝기 조절
 - 조도센서 300 ~ 500 → LED 255 ~ 0

어두워지면 LED켜기

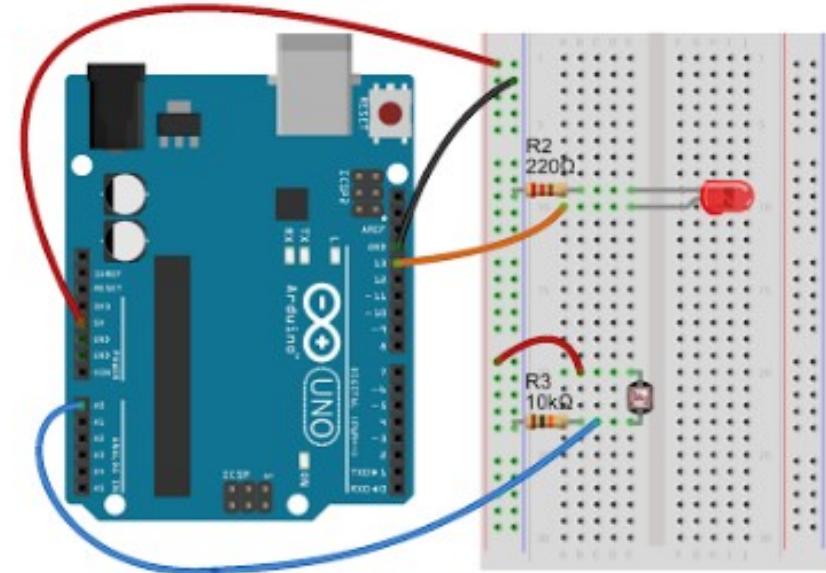
- 회로 연결
 - LED (+) : D13
 - 조도센서 입력 : A0 (10KΩ)

```

int ledPin = 13;
int sensorPin = 0;

void setup() {
    pinMode(ledPin, OUTPUT);
}

void loop() {
    int val = analogRead(sensorPin);
    if( val < 300 ) {
        digitalWrite(ledPin, HIGH);
    } else {
        digitalWrite(ledPin, LOW);
    }
}
  
```



A0핀 조도센서 값 읽어 val에 저장
val 값이 300 미만이면 LED ON
아니면 LED OFF



주변밝기에 따라 LED 밝기 제어하기

```

int ledPin = 13;
int cdsPin = 0;
int cdsValue = 0;

void setup() {
    pinMode(cdsPin, INPUT);
    pinMode(ledPin, OUTPUT);
    //Serial.begin(9600);
}

```

```

void loop() {

    cdsValue = analogRead(cdsPin);
    //Serial.print("pre-cdsValue: ");
    //Serial.print(cdsValue);

    cdsValue = map(cdsValue, 300, 500, 255, 0);
    cdsValue = constrain(cdsValue , 0, 255);
    analogWrite(ledPin, cdsValue);

    //Serial.print(", post-cdsValue: ");
    //Serial.println(cdsValue);

}

```

참고

constrain(value, min, max);

변수 value 값을 min~max 사이로 한정

A0핀 조도센서 값 읽어 cdsValue에 저장

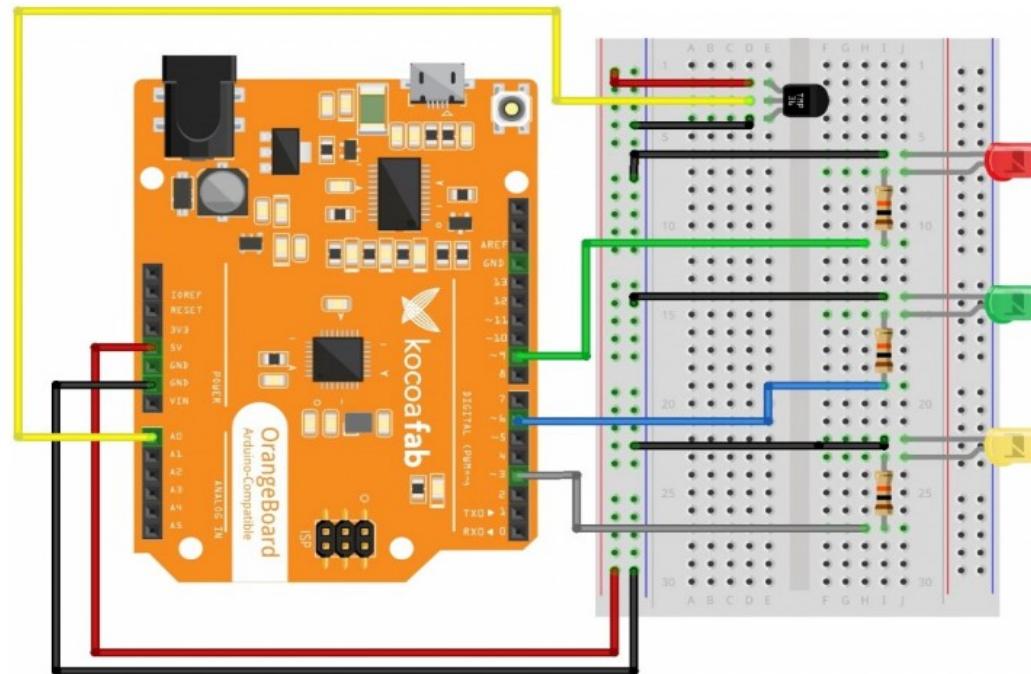
cdsValue 값 범위 변환 (300, 500)
 \rightarrow (255, 0)

아날로그핀 13번에 cdsValue 값 출력



온도센서와 LED

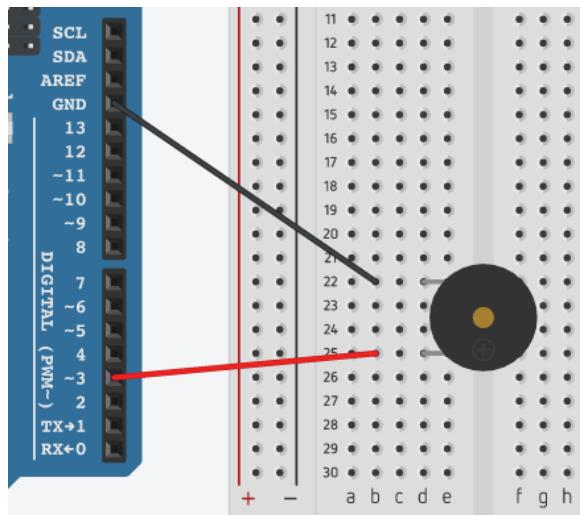
- 온도 측정 후 온도에 따라 다른 LED 켜기
 - 섭씨 온도로 변환
 - 20도 미만이면 빨강 LED, 20도 이상 30도 이하면 초록 LED, 30도 이상이면 노랑 LED 점등



<https://www.kocoafab.cc/tutorial/view/573>

Piezo Buzzer

- 주파수를 전기신호로 출력하는 전자부품
 - PWM 출력의 원리를 이용하여 주파수(소리의 파장)를 전기신호로 출력
 - 스피커와 유사한 구조(낮은 품질의 소리)



#define piezo 3	
void setup() { pinMode(piezo, OUTPUT); }	3번 핀을 디지털 출력으로 설정
void loop() { tone(piezo, 261.6, 1000); tone(piezo, 329.7, 2000); tone(piezo, 349.2, 1000); }	3번핀, 4옥타브 도를 1초 동안 출력 3번핀, 4옥타브 미를 2초 동안 출력 3번핀, 4옥타브 파를 1초 동안 출력



주파수에 따른 소리의 높낮이

음계 온타브	1	2	3	4	5	6	7	8
C(도)	32.7	65.4	130.8		523.3	1046.6	2093.0	4186.0
C#	34.6	68.3	138.6	277.2	554.4	1108.7	2217.5	4434.9
D#	36.7	73.4	146.8	293.7	587.3	1174.7	2349.3	4698.6
D(레)	38.9	77.8	115.6	311.1	622.3	1244.5	2489.0	4978.0
E(미)	41.2	82.4	164.8	329.6	659.3	1318.5	2637.0	5274.0
F(파)	43.7	87.3	174.6	349.2	698.5	1396.9	2793.8	5587.7
F#	46.2	92.5	185.0	370.0	740.0	1480.0	2960.0	5919.9
G(솔)	49.0	98.0	196.0	312.0	784.0	1568.0	3136.0	6271.9
G#	51.9	103.8	207.7	415.3	830.6	1661.2	3322.4	6644.9
A(라)	55.0	110.0	220.0	330.0	880.0	1760.0	3520.0	7040.0
A#	58.3	116.5	233.1	466.2	932.3	1864.7	3729.3	7040.0
B(사)	61.7	123.5	246.9	493.9	987.8	1975.5	3951.1	7902.1



학교 종이 땡땡땡 연주

```
#define C 262
#define D 294
#define E 330
#define F 349
#define G 392
#define A 440
#define B 494

int piezoPin = 8;
int tempo = 200; // tone 함수 자연 값 설정
int notes[25] = { G, G, A, A, G, G, E, G, G, E, E, D, G, G, A, A, G, G, E, G, E, D, E, C };

void setup() {
    pinMode(piezoPin, OUTPUT);
}

void loop() {

    for (int i = 0; i < 12; i++) {
        tone(piezoPin, notes[ i ], tempo);
        delay (300);
    }
    delay(100);
    for (int i = 12; i < 25; i++) {
        tone(piezoPin, notes[ i ], tempo);
        delay(300);
    }
}
```

도
레
미
파
솔
라
시

멜로디 중간에 짧게 멈
추는 용도



함수정리

`analogRead(pin);`

아날로그 입력 핀을 통해 0~1023 사이의 값 읽음

- pin : 아날로그 입력 핀 번호

`map(value, a1, a2, b1, b2);`

a1~a2사이의 value 값을 b1~b2 범위의 값으로 변환

- value : a1~a2사이의 정수형 값

`tone(pin, frequency, duration);`

a1~a2사이의 value 값을 b1~b2 범위의 값으로 변환

- pin : pwm 핀 번호
- frequency : 31~65535 범위의 주파수(hz)
- duration : 출력시간(단위 ms)

`noTone();`

주파수 출력 중지

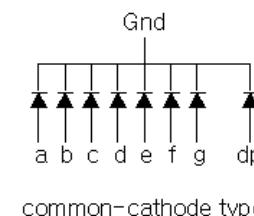
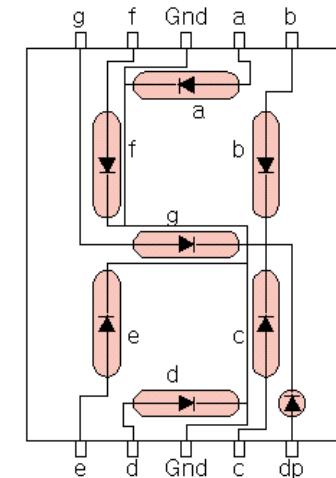
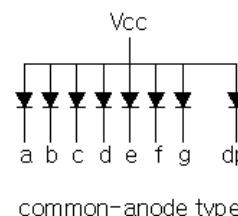
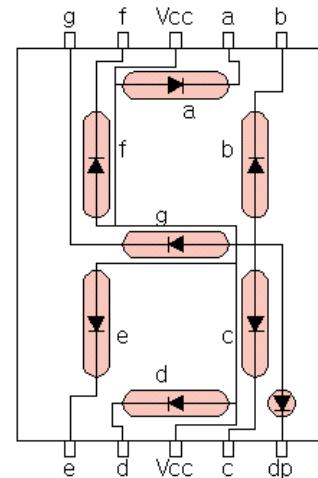


7Segment/초음파센서/모터제어



7-Segment

- 8개의 LED가 8(숫자) 모양으로 배치
 - a, b, c, d, e, f, g, dot
- 1byte 단위로 조합된 데이터로 숫자와 영문자 표시
- 2가지 종류
 - Common Anode 형
 - Common Cathod 형



7-Segment

- Common-Anode 형의 문자 표현

표현문자	a	b	c	d	e	f	g	DP
0	0	0	0	0	0	0	1	1
1	1	0	0	1	1	1	1	1
2	0	0	1	0	0	1	0	1
3	0	0	0	0	1	1	0	1
4	1	0	0	1	1	0	0	1
5	0	1	0	0	1	0	0	1
6	0	1	0	0	0	0	0	1
7	0	0	0	1	1	1	1	1
8	0	0	0	0	0	0	0	1
9	0	0	0	0	1	0	0	1

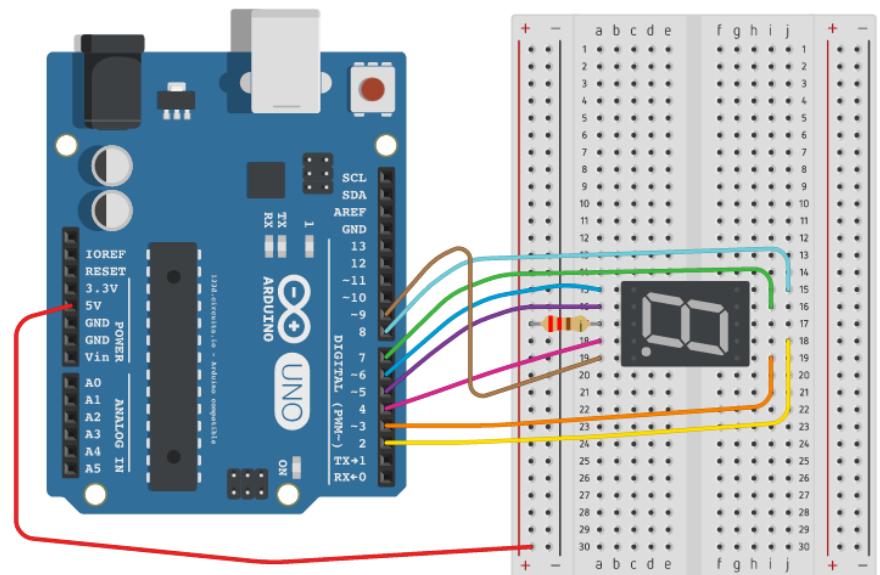
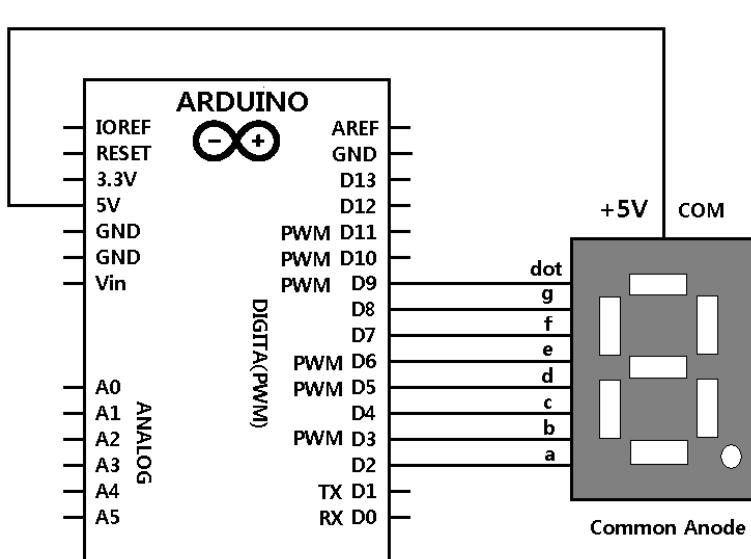


7-Segment에 숫자 0 표시하기

- Common Anode 형 세그먼트 사용

- a, b, c, d, e, f, g, dot에 아두이노 2~9번 핀 순서대로 연결
- 표시하고자 하는 글자에 맞게 해당 비트의 LED ON

표현문자	a	b	c	d	e	f	g	DP
0	0	0	0	0	0	0	1	1



7-Segment에 숫자 0 표시하기

```
void setup(){
    for (int i=2; i<=9; i++){
        pinMode(i, OUTPUT);
    }
}
```

세그먼트의 a, b, c, d, e, f, g, dot에
디지털입출력 핀 2~9번을 연결하고, 출력모드 설정

void loop() {	
digitalWrite(2, LOW);	a = 0
digitalWrite(3, LOW);	b = 0
digitalWrite(4, LOW);	c = 0
digitalWrite(5, LOW);	d = 0
digitalWrite(6, LOW);	e = 0
digitalWrite(7, LOW);	f = 0
digitalWrite(8, HIGH);	g = 1
digitalWrite(9, HIGH);	dot = 1
}	



7-Segment에 숫자 0 표시하기

- 배열 사용

<pre>int segPins[8]={2,3,4,5,6,7,8,9}; int degData0[8]={0,0,0,0,0,0,1,1};</pre>	<p>FND 세그먼트에 사용할 입출력 핀 저장 "0"을 표시하기 위한 데이터</p>
<pre>void setup() { for(int i=0; i<8; i++) { pinMode(segPins[i], OUTPUT); } }</pre>	<p>2~9번 핀 디지털 출력 핀으로 설정</p>
<pre>void loop() { for(int i=0; i<8; i++) { digitalWrite(segPins[i], degData0[i]); } }</pre>	<p>segPins[i]에서 7세그먼트에 연결된 핀 순서대로 호출 (a~f, dot) 각 핀에 대응하는 bit 데이터 degData0[i]를 출력</p>



7-Segment에 Count 표시하기

- 숫자 0, 1, 2, ..., 9 차례대로 표시하기
- 배열 사용

1차원 배열 구성

```
int digitData0[8]={0, 0, 0, 0, 0, 0, 1, 1};  
int digitData1[8]={1, 0, 0, 1, 1, 1, 1, 1};  
int digitData2[8]={0, 0, 1, 0, 0, 1, 0, 1};  
int digitData3[8]={0, 0, 0, 0, 1, 1, 0, 1};  
int digitData4[8]={1, 0, 0, 1, 1, 0, 0, 1};
```

2차원 배열 구성

```
int digitData[5][8]={  
    {0, 0, 0, 0, 0, 0, 1, 1}, 0  
    {1, 0, 0, 1, 1, 1, 1, 1}, 1  
    {0, 0, 1, 0, 0, 1, 0, 1}, 2  
    {0, 0, 0, 0, 1, 1, 0, 1}, 3  
    {1, 0, 0, 1, 1, 0, 0, 1}, 4  
    {0, 1, 0, 0, 1, 0, 0, 1} 5  
}; | | | | | | |  
a b c d e f g DP
```



7-Segment에 Count 표시하기

```
byte digits[10][7] = {
    {1, 1, 1, 1, 1, 1, 0}, // 0
    {0, 1, 1, 0, 0, 0, 0}, // 1
    {1, 1, 0, 1, 1, 0, 1}, // 2
    {0, 1, 1, 1, 1, 0, 1}, // 3
    {0, 0, 1, 1, 0, 1, 1}, // 4
    {0, 1, 1, 0, 1, 1, 1}, // 5
    {1, 1, 1, 0, 1, 1, 1}, // 6
    {0, 0, 1, 1, 1, 0, 0}, // 7
    {1, 1, 1, 1, 1, 1, 1}, // 8
    {0, 1, 1, 1, 1, 1, 1} // 9
};
```

```
void setup() {
    int pin= 2;
    for (pin = 2; pin<10; pin++)
        pinmode(pin, output);
    digitalWrite(9, high);
}
```

```
void writenum(byte digit) {
    int pin = 2;
    for (int segpin = 0; segpin < 7; segpin++) {
        digitalWrite(pin+segpin, digits[digit][segpin]);
    }
}
```

```
void loop() {
    for (byte count = 0; count < 10; count++) {
        writeNum(count);
        delay(1000);
    }
}
```

anode 형 세그먼트와 아두이노 핀 연결
 7-세그먼트 : a, b, c, d, e, f, g, dot
 아두이노핀 : 2, 3, 4, 5, 6, 7, 8, 9

2~9번 핀 디지털 출력 설정
 9번 핀은 dot로 1 설정

사용자 정의 함수
 digit이 5이면,
 2차원 배열에서 5는 {0, 1, 1, 0, 1, 1, 1}, 이므로
 2번핀에 digits[5][0] 값인 0 출력
 3번핀에 digits[5][1] 값인 1 출력
 5번핀에 digits[5][2] 값인 1 출력

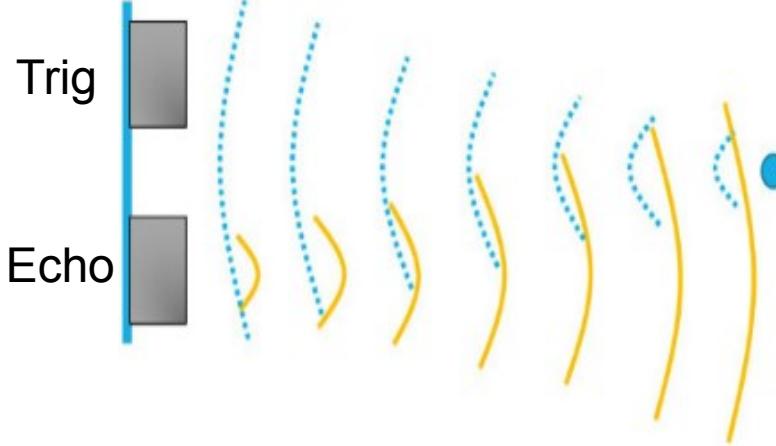
 8번핀에 digits[5][6] 값인 1 출력

켜진 led들이 보여야 하기 때문에 1초 기다린다.



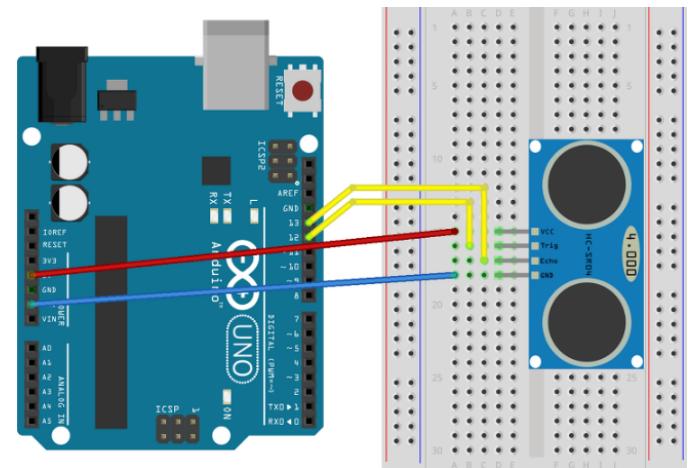
초음파 센서

- 초음파를 이용하여 사물까지의 거리를 알려주는 센서
 - 초음파를 발사한 후 반향되는 시간을 측정하여 거리 계산
 - 측정거리 : 2cm ~ 5cm



$$\text{Distance} = (\text{time} / 29.1) / 2$$

- 초음파 29.1us마다 1cm이동
- 왕복 거리



초음파센서	아두이노
Vcc	5V
GND	GND
Trig	디지털 출력 핀(12)
Echo	디지털 입력 핀(13)

초음파 센서로 거리 측정

int trigPin = 12; int echoPin = 13;	초음파 센서 출력핀(Trig) 설정 초음파 센서 입력핀(Echo) 설정
void setup() { Serial.begin(9600); pinMode(trigPin, OUTPUT); pinMode(echoPin, INPUT); }	시리얼 포트 9600으로 초기화 트리거 핀 출력으로 설정 에코핀 입력으로 설정
void loop() { Serial.print(UltrasonicMeasure()); Serial.println("cm"); delay(300); }	초음파 센서로 거리를 측정하여 시리얼모니터로 출력
long UltrasonicMeasure(void) { long duration, distance; digitalWrite(trigPin, HIGH); delayMicroseconds (10); digitalWrite(trigPin, LOW); duration = pulseIn (echoPin, HIGH); if(duration == 0) return; distance = duration / 58.2; return duration; }	초음파 센서로 거리를 출력하기 위한 함수 초음파를 발사하기 위해 trigPin에 HIGH 설정 초음파가 충분히 발사될 수 있도록 10us 이상 HIGH 유지 후 LOW로 설정 echoPin이 HIGH가 될 때까지 시간 측정 (LOW상태에서 초음파가 들어오면 HIGH가 됨) 1초가 지나도 초음파 들어오지 않으면 리턴 거리 = (측정시간 / 29.1) / 2 duration 리턴



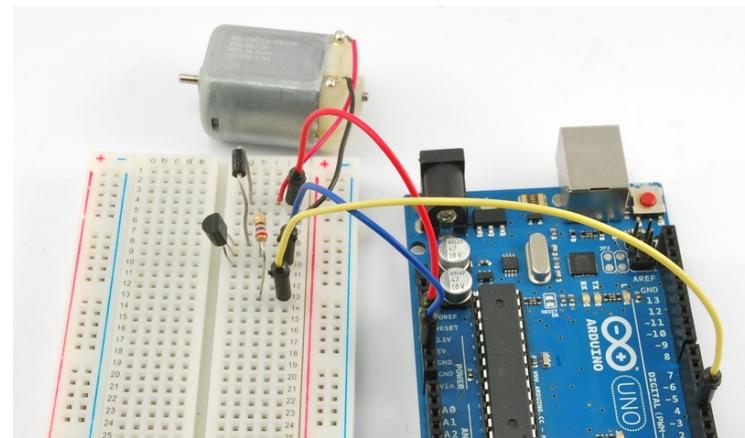
모터 제어

- DC 모터: 우리가 알고 있는 모터! 빙글 빙글!
- 서보 모터
- 스텝핑 모터



DC모터

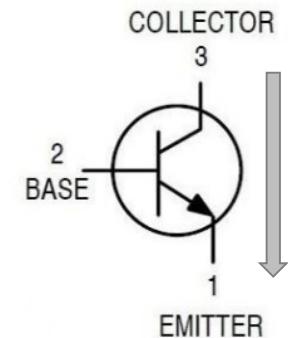
- **DC모터 제어**
 - 많은 전력 소모
 - 별도 회로구성 필요
 - PWM 이용하여 속도 조절
 - 0~255 사이값
- **DC모터 연결 회로 구성**
 - 트랜지스터(PN2222)
 - 다이오드(1N4001)
 - 1K옴 저항



DC모터

• 트랜지스터(PN2222)

- 적은 전류로 모터 구동에 필요한 많은 전류를 제어할 수 있게 함
- 적은 양의 전류를 B로 보내면 C→E로 전류 흐름

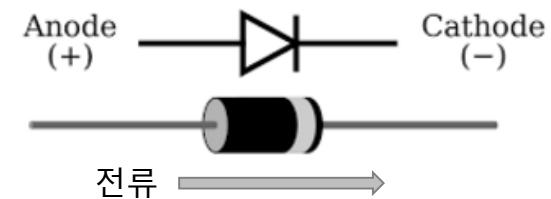


• 다이오드(1N4001)

- 역전압 발생으로 인한 아두이노 보드 손상 방지
- 전류가 한쪽 방향으로 흐르도록 제어

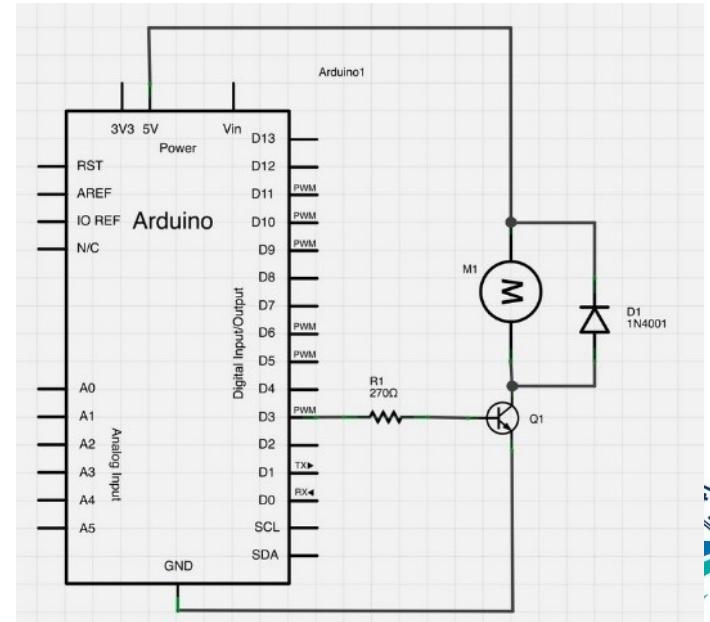
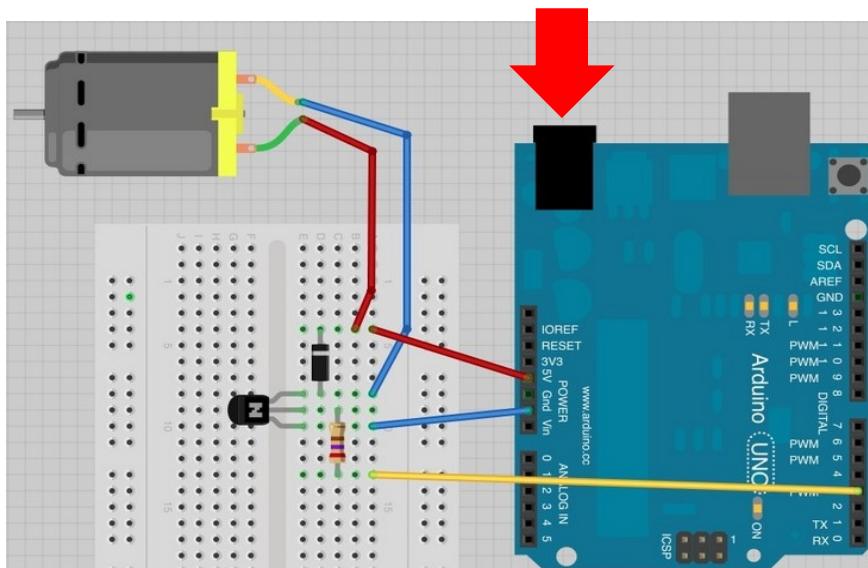
• 1K옴 저항

- 전류 제한



DC모터 제어하기

- 모터 속도 제어
 - 시리얼 모니터에서 0 ~ 255 사이 값 입력하여 모터 구동
- DC모터 회로 구성 시 주의
 - 트랜지스터 연결 : C-B-E 핀 연결
 - 다이오드 방향 : 줄무늬가 5V 전원 쪽에 위치



DC모터 제어하기

```

int motorPin = 3;

void setup() {
    pinMode(motorPin, OUTPUT);
    Serial.begin(9600);
    while (! Serial);
    Serial.println("Speed 0 to 255");
}

void loop() {
    if ( Serial.available() ) {
        long speed = Serial.parseInt();
        if (speed >= 0 && speed <= 255) {
            Serial.println("Current speed: " + String(speed));
            analogWrite(motorPin, speed);
        }
    }
}

```

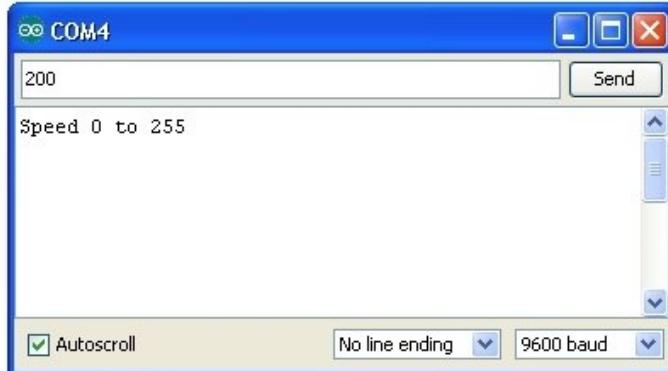
DC모터에 연결된 PWM 출력 가능 핀

시리얼 포트 9600으로 초기화

속도 입력범위 출력

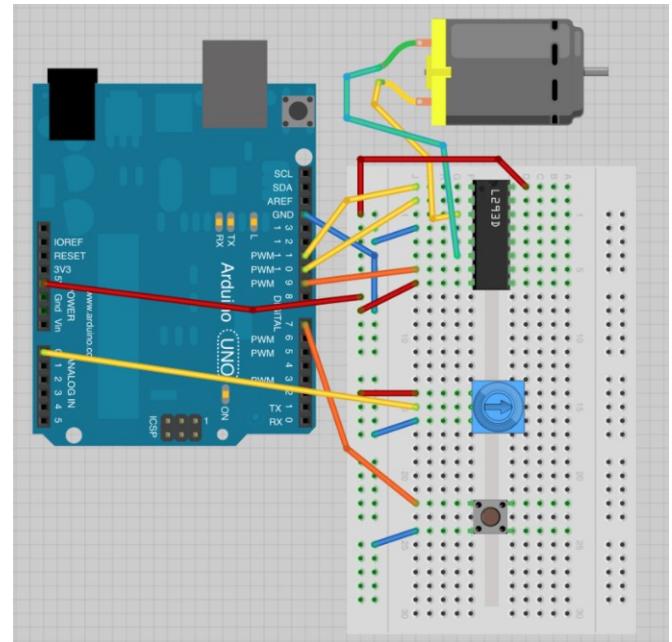
시리얼모니터로부터 입력이 있는 경우
정수값 입력
범위 이내 값이면

현재 속도 모니터에 출력
PWM신호로 DC모터 속도 조절



Advanced

- DC 모터 양방향으로 제어하기
 - L293D 모터 드라이버 칩 사용
 - 모터 속도와 동작 방향 제어

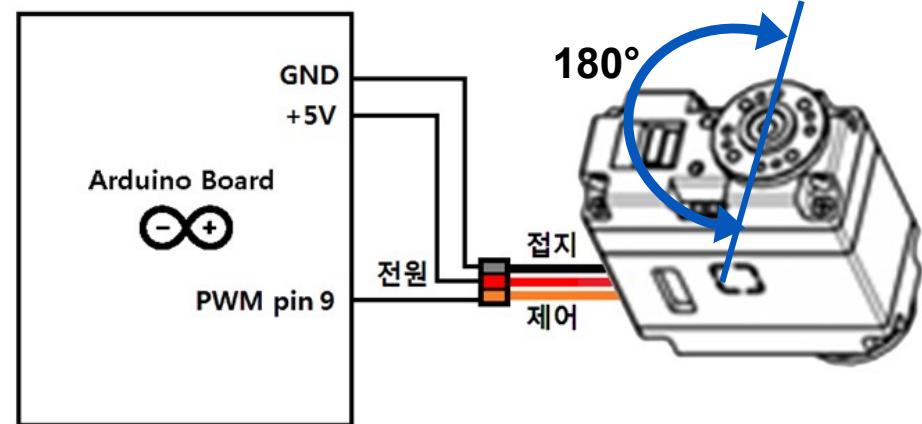


<http://wiki.vctec.co.kr/opensource/arduino/dcmotordirection>

서보모터

- $0^\circ \sim 180^\circ$ 범위의 각도를 정밀하게 이동
- 3개의 리드선

서보모터	아두이노
전원	5V
접지	GND
제어	PWM 출력 핀



- 서보 라이브러리 선언 후 사용

```
#include <Servo.h>
```

Example : Servo-Sweep

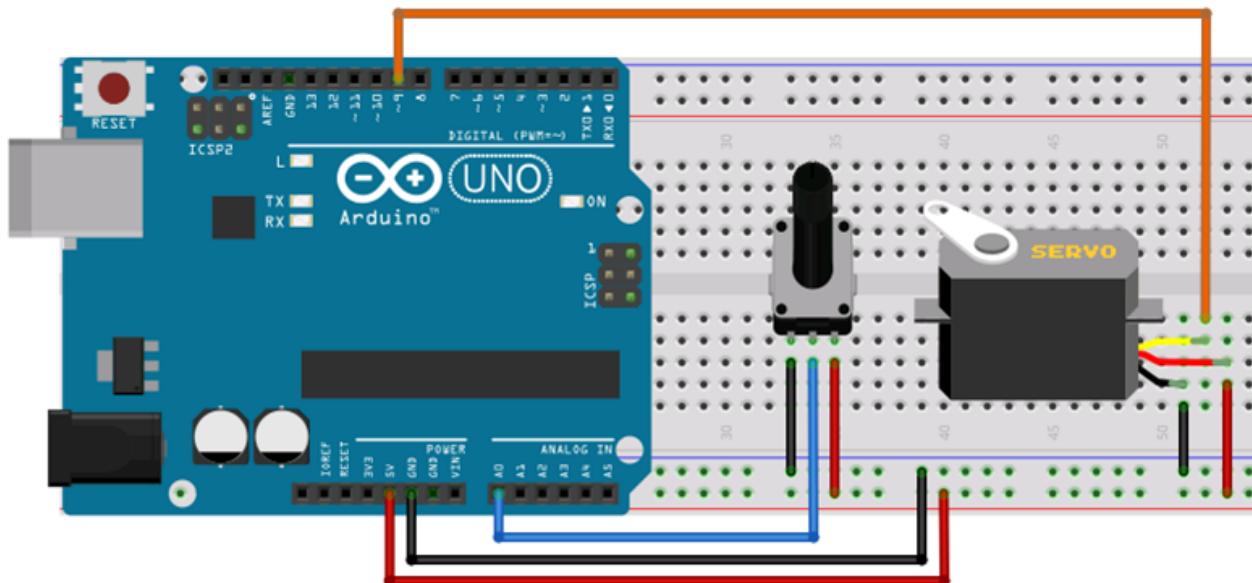
- 0~180도까지 1도씩 증가시키면서 회전

#include <Servo.h>	서보 라이브러리 포함 선언
Servo myservo; int pos=0;	Servo 변수를 myservo로 지정(12개까지 사용가능) 회전각도 pos의 초기값 지정
void setup() { myservo.attach(9); }	PWM핀 9번을 myservo에 지정
void loop() { for(pos = 0; pos <= 180; pos += 1) { myservo.write(pos); delay(15); } for(pos = 180; pos >= 0; pos -= 1) { myservo.write(pos); delay(15); } }	0도에서 180도까지 1도씩 증가 증가된 이동각 만큼 회전 서보모터가 회전하는 동안 지연 180도에서 0도까지 1도씩 감소 서보모터 회전 각도 출력 서보모터가 회전하는 동안 지연



Example : Servo-Knob

아두이노 UNO 보드		연결 장치
핀번호	setup() 설정	
PWM 핀 9번	설정 필요 없음	서보모터 제어판
아날로그 입력핀 5번	설정 필요 없음	가변저항



Example : Servo-Knob

- 가변저항을 사용하여 회전각도 제어

#include <Servo.h>	서보 라이브러리 포함 선언
Servo myservo; int potpin=0; Int servopin=9; int val=0;	Servo 변수를 myservo로 지정 가변저항 potpin에 아날로그 입력 A0 핀 지정 서보모터 servopin을 PWM 출력 9번 핀 지정
void setup() { myservo. attach (servopin); }	PWM핀 9번을 myservo에 지정
void loop() { val = analogRead (potpin); val = map (val, 0, 1023, 0, 180); myservo. write (val); delay(15); }	A0핀 가변저항의 값을 읽어 val에 저장 map함수로 0~180도 범위로 변환 변환된 각도만큼 서보모터 회전 시간 지연



함수정리

delayMicroseconds(delay);

시간지연 함수

- micro seconds : 마이크로 초 단위

pulseIn(pin, value);

에코 핀이 value에 지정된 신호가 될 때까지 시간(us) 측정
 (1초가 지나도 신호가 들어오지 않으면 0 리턴)
 - pin : 에코 핀 번호
 - value : 입력 핀의 신호 (LOW 또는 HIGH)

long Serial.parseInt();

시리얼 버퍼에 있는 정수(long 형)만 읽음

Servo myservo;
 myservo.attach(pin);

서보 객체에 9번 핀의 서보 모터에 연결
 - pin : 서보 모터에 연결할 핀 번호

myservo.write(angle);

지정된 각도만큼 서보 모터 회전
 - angle : 회전할 각도



Library 사용 및 추가

- **Library 사용**

- **Library 추가**

- 인터넷에서 다운로드

- Arduino.cc
 - GitHub

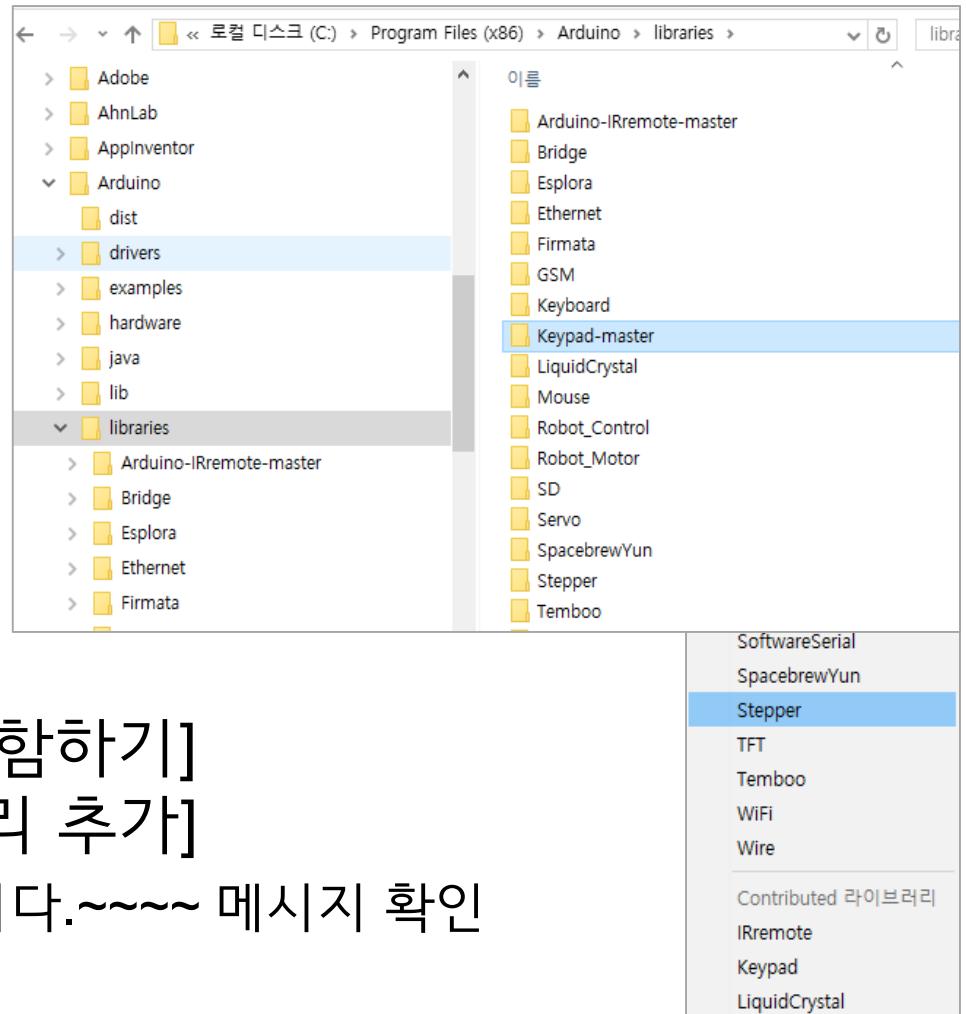
- [스케치]-[라이브러리 포함하기]

- [ZIP 라이브러리 추가]

- 라이브러리 추가되었습니다.~~~~ 메시지 확인

- 압축파일 해제 후

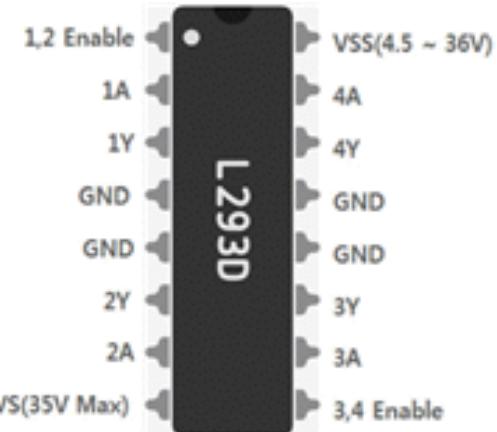
- [Arduino 설치 폴더 - libraries 폴더]에 복사



모터 드라이버

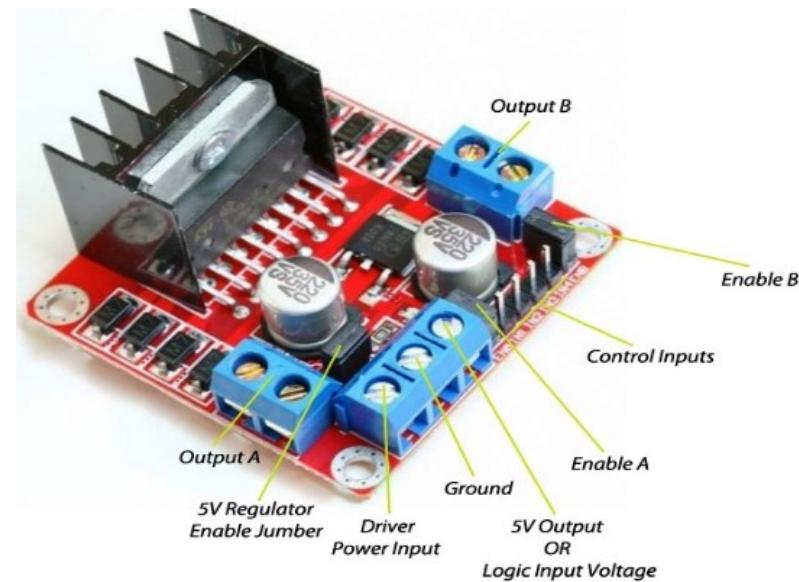
- **L293D**

1,2 Enable 3,4 Enable	모터의 속도 PWM핀에 연결
1A, 2A 3A, 4A	모터 회전방향 제어 아두이노 핀에 연결
1Y, 2Y 3Y, 4Y	모터의 (+), (-)에 연결



- **L298**

Enable A Enable B	모터의 속도 PWM핀에 연결
Control Input 1, 2 Control Input 3, 4	모터 회전방향 제어 아두이노 핀에 연결
Output A Output B	모터의 (+), (-)에 연결



HC-06 모듈 설정/ 가상 시리얼 통신

Bluetooth

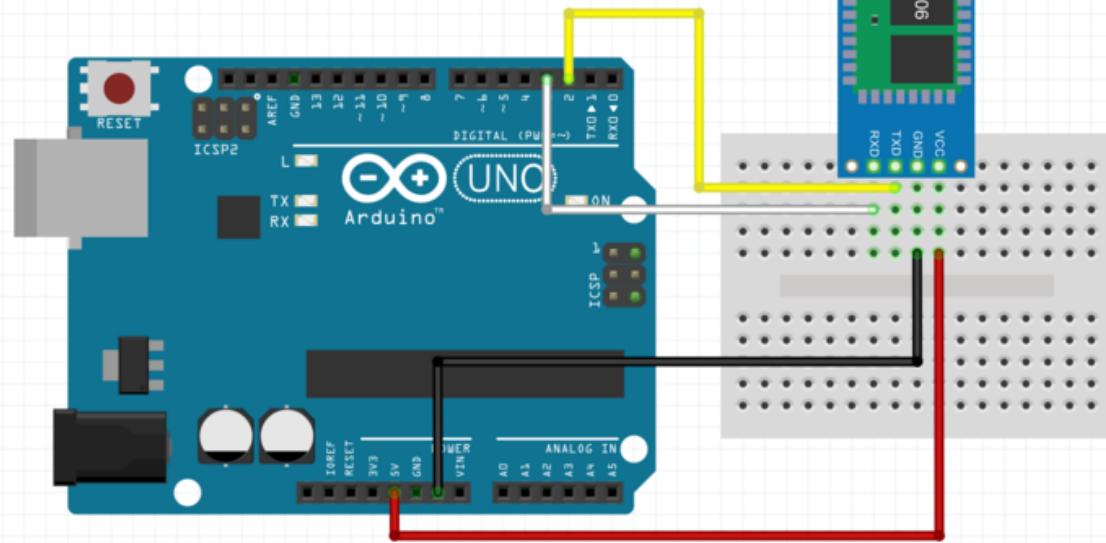


블루투스 통신

- HC-06 모듈 설정
- 소프트웨어シリ얼 라이브러리 선언 후 사용

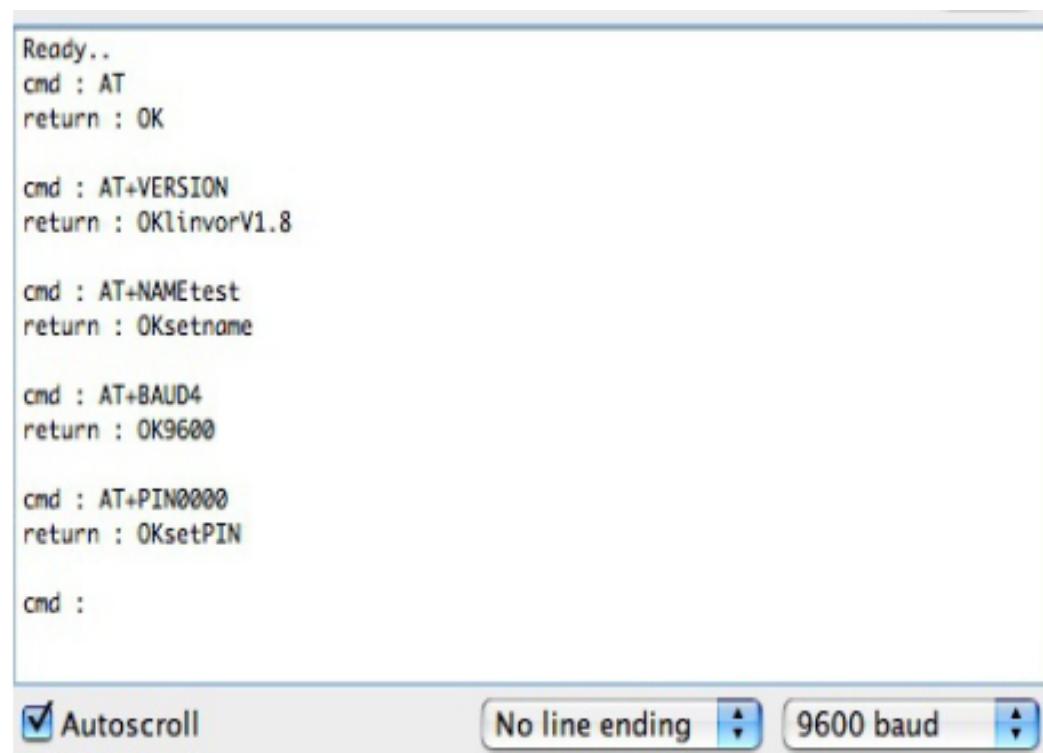
```
#include <SoftwareSerial.h>
```

HC-06	아두이노
Vcc	5V
GND	GND
TXD	2번 핀
RXD	3번 핀



AT 명령어

- **Serial Monitor을 통해 HC-06 설정값 변경**
 - AT
 - AT+VERSION
 - AT+NAME블루투스이름
 - AT+BAUD4



```
Ready..  
cmd : AT  
return : OK  
  
cmd : AT+VERSION  
return : OKlinvorV1.8  
  
cmd : AT+NAMEtest  
return : OKsetname  
  
cmd : AT+BAUD4  
return : OK9600  
  
cmd : AT+PIN0000  
return : OKsetPIN  
  
cmd :  
  
 Autoscroll      No line ending      9600 baud
```



```

#include <SoftwareSerial.h>
#define rxPin 2
#define txPin 3

SoftwareSerial SwSerial(rxPin, txPin); // swSerial(rx, tx) <-> 블루투스 모듈 (TX, RX)
char data;

void setup() {
    Serial.begin(9600);
    SwSerial.begin(9600);
    Serial.println("Ready..");
}

void loop() {
    Serial.flush();           // 시리얼 버퍼를 비운다.
    Serial.print("cmd : ");
    while (!Serial.available()); // 버퍼에 값이 들어올 때 까지 대기.

    while (Serial.available()) { // HC-06 으로 명령어를 날린다
        data = Serial.read();
        if(data == -1) break ;
        SwSerial.print(data);
        Serial.print(data);

        delay(1);      // 시리얼 통신에서는 9600bps 기준으로 read 를 사용할 때 1ms 의 딜레이를 줘야 한다.
    }
    Serial.println();

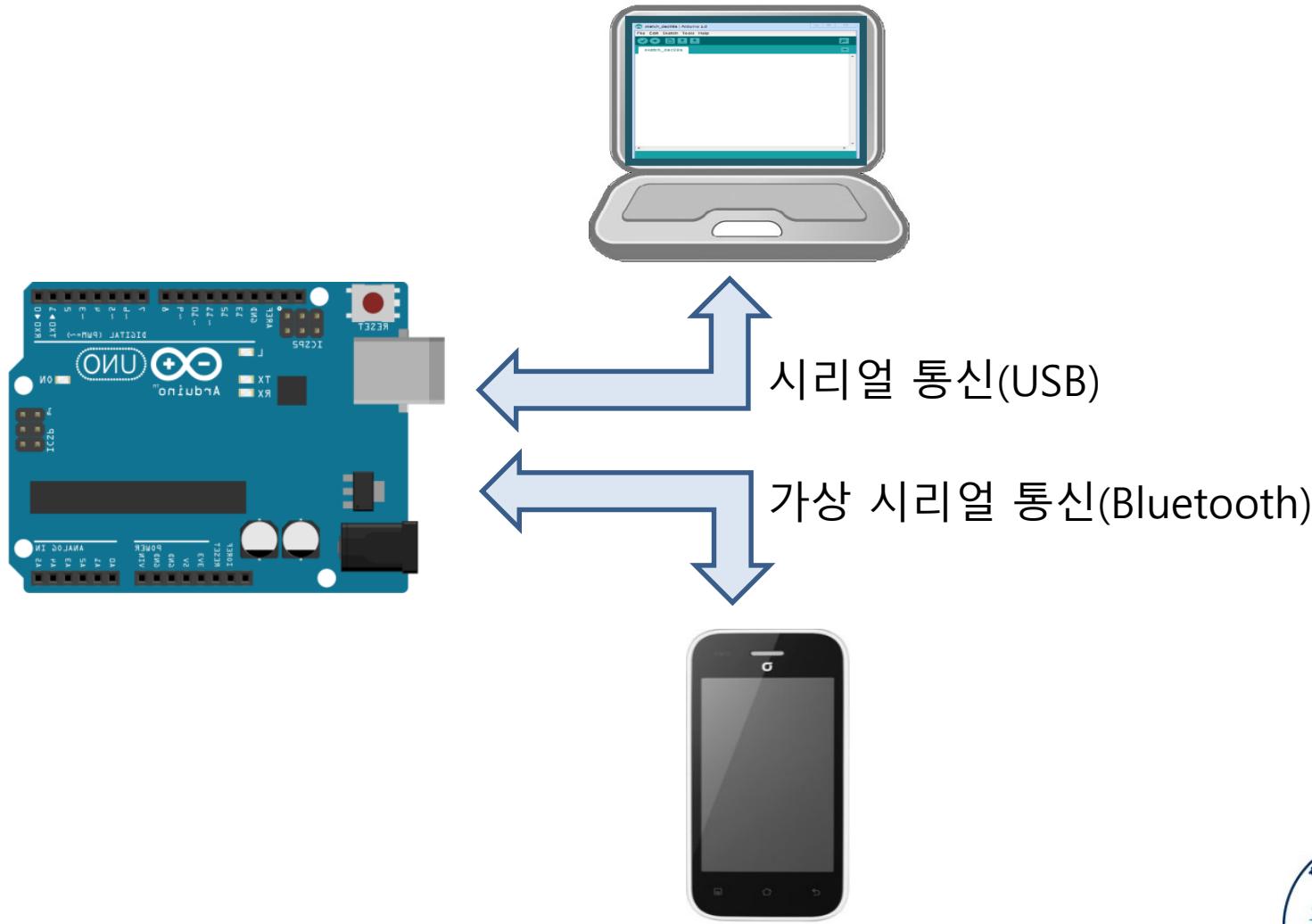
    delay(1000);          // HC-06에서 처리할 시간을 준다
    Serial.print("return : ");

    while (SwSerial.available()) { // HC-06 으로 부터 받아온 리턴 값을 출력한다
        data = SwSerial.read();
        if(data == -1) break ;
        Serial.print(data);
        delay(1);
    }
    Serial.println("");
}

```



블루투스를 이용한 통신

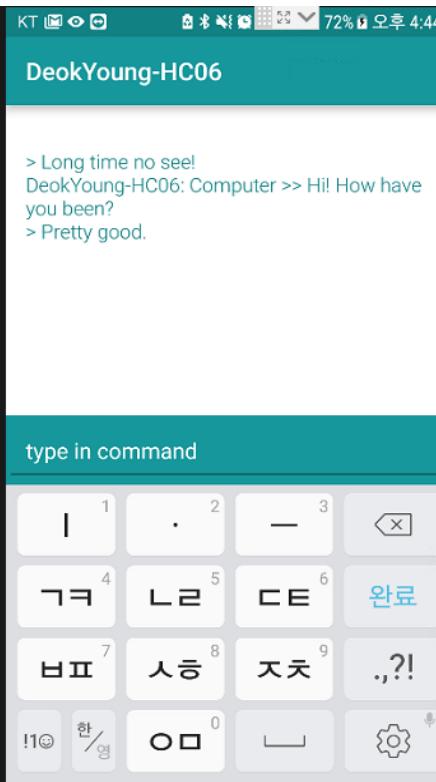


블루투스를 이용한 통신

```
#include <SoftwareSerial.h>
SoftwareSerial swSerial(2,3);
```

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

```
void loop() {
    if( Serial.available() ) {
        String str = Serial.readString();
        swSerial.println(str);
        Serial.print(str);
    }
    if( swSerial.available() ) {
        String str = swSerial.readString();
        Serial.println(str);
    }
}
```



소프트웨어 시리얼 라이브러리 포함 선언
swSerial(rx, tx) ⇔ 블루투스 모듈(TX, RX)

시리얼 통신
블루투스 통

// PC → (아드

시리얼 모니터

시리얼버퍼

가상 시리얼

시리얼 버퍼

// 핸드폰 →

블루통신 모

자동 스크롤

line ending 없음

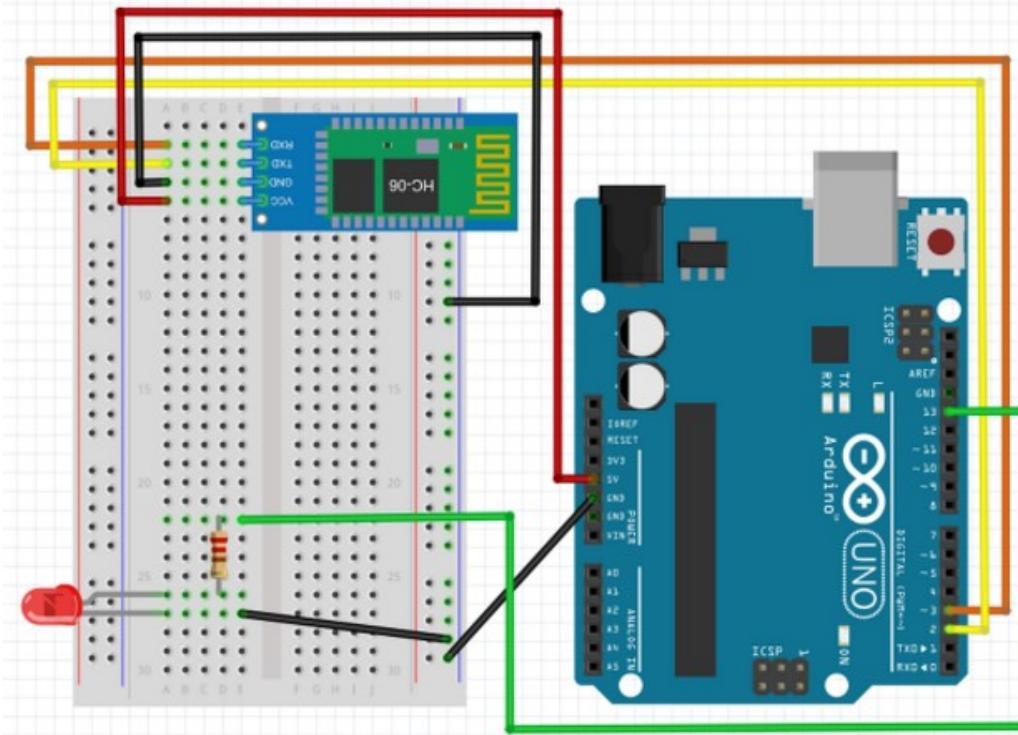
9600 보드레이트

가상 시리얼버퍼에서 문자열을 읽어 str에 저장
시리얼 버퍼에 str 출력



블루투스를 이용하여 LED ON/OFF

- 블루투스 모듈 연결 : 2번 핀-TX , 3번 핀-RX
 - LED : 13핀 연결
 - 블루투스 모듈에 수신된 값 ‘1’ : LED ON
‘0’ : LED OFF



블루투스를 이용한 LED ON/OFF

```
#include <SoftwareSerial.h>
SoftwareSerial swSerial(2,3);
```

```
void setup()
{
    swSerial.begin(9600);
    pinMode(13, OUTPUT);
}
```

```
void loop() {

    if ( swSerial.available() ) {
        char cmd = (char) swSerial.read();
        if(cmd == '1') {
            digitalWrite(13, HIGH);
        } else if(cmd == '0') {
            digitalWrite(13, LOW);
        }
    }
}
```

소프트웨어 시리얼 라이브러리 포함 선언
swSerial(rx, tx) ⇔ 블루투스 모듈(TX, RX)

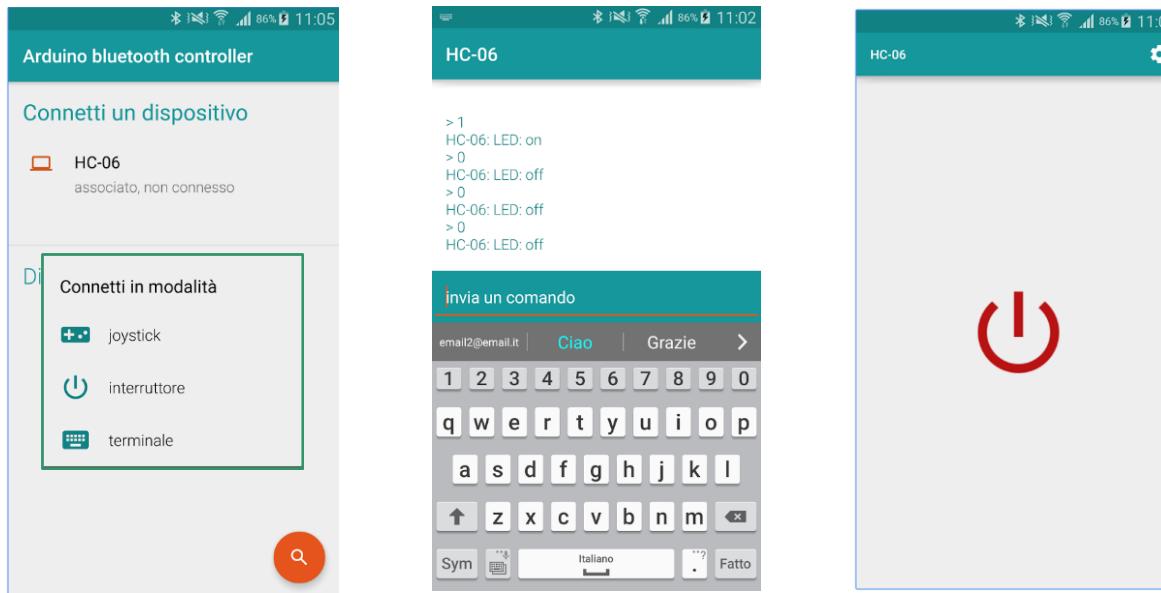
블루투스 통신 설정 (9600baud)
13번 핀 디지털 출력으로 설정

블루통신 모니터에 입력이 있는 경우
가상 시리얼버퍼에서 문자를 읽어 cmd에 저장
cmd 값이 '1' 이면 LED On
cmd 값이 '0' 이면 LED Off
이외 값이면 아무 동작 하지 않음



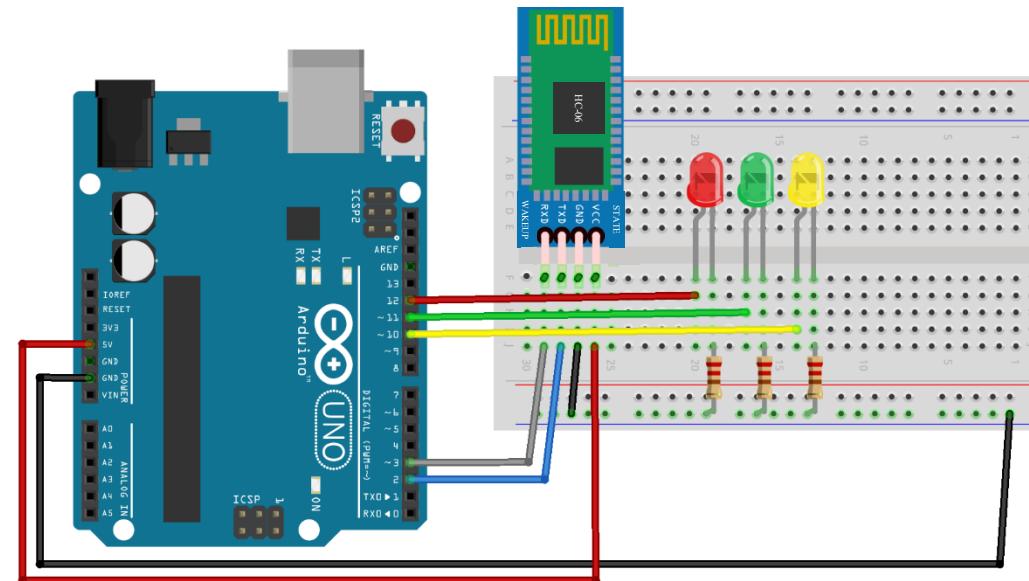
블루투스 제어 앱

- Google Play에서 블루투스제어 앱 검색 및 설치
- **arduino bluetooth controller**
 - 터미널 모드
 - 스위치 모드



앱인벤터로 블루투스제어 앱 작성

- ON/OFF 버튼으로 LED ON/OFF



fritzing



블루투스 이용한 LED 3개 ON/OFF

```
#include<SoftwareSerial.h>
SoftwareSerial swSerial(2,3);
void setup()
{
    Serial.begin(9600);
    swSerial.begin(9600);
    pinMode(12, OUTPUT);    // 12번 핀 : 빨강 LED
    pinMode(11, OUTPUT);    // 11번 핀 : 초록 LED
    pinMode(10, OUTPUT);    // 10번 핀 : 노랑 LED
}
void loop()
{
    long comm;
    if( swSerial.available() ) {                                // Bluetooth통신에서 들어온 값이 있는지 검사,
        comm = swSerial.parseInt();                            // long 타입 숫자 반환(수신 값이 숫자일 경우)
                                                                // 숫자 아닐 경우 0 반환
        switch( comm ) {
            case 1 : digitalWrite(12, HIGH); break;           // 1 수신 : 빨강 LED ON
            case 2 : digitalWrite(12, LOW); break;             // 2 수신 : 빨강 LED OFF
            case 3 : digitalWrite(11, HIGH); break;           // 3 수신 : 초록 LED ON
            case 4 : digitalWrite(11, LOW); break;             // 4 수신 : 초록 LED OFF
            case 5 : digitalWrite(10, HIGH); break;           // 5 수신 : 노랑 LED ON
            case 6 : digitalWrite(10, LOW); break;             // 6 수신 : 노랑 LED OFF
        }
    }
}
```

아두이노 코드



블루투스 이용한 LED 3개 ON/OFF

앱인벤터 코드

```
when lstBluetooth .BeforePicking
do set lstBluetooth .Elements to BluetoothClient1 . AddressesAndNames

when lstBluetooth .AfterPicking
do evaluate but ignore result call BluetoothClient1 .Connect
address lstBluetooth . Selection

when Clock1 .Timer
do if BluetoothClient1 . IsConnected
then set lstBluetooth . Enabled to false
set LabelState . Text to join lstBluetooth . Selection
" 연결되었습니다... "
else set lstBluetooth . Enabled to true
set LabelState . Text to " 연결해제되었습니다... "

when Disconn .Click
do call BluetoothClient1 .Disconnect
```



블루투스 이용한 LED 3개 ON/OFF

```
when RedOn .Click
do if BluetoothClient1 .IsConnected
then call BluetoothClient1 .Send1ByteNumber
      number 1
```

```
when RedOff .Click
do if BluetoothClient1 .IsConnected
then call BluetoothClient1 .Send1ByteNumber
      number 2
```

```
when YellowOn .Click
do if BluetoothClient1 .IsConnected
then call BluetoothClient1 .Send1ByteNumber
      number 5
```

```
when YellowOff .Click
do if BluetoothClient1 .IsConnected
then call BluetoothClient1 .Send1ByteNumber
      number 6
```

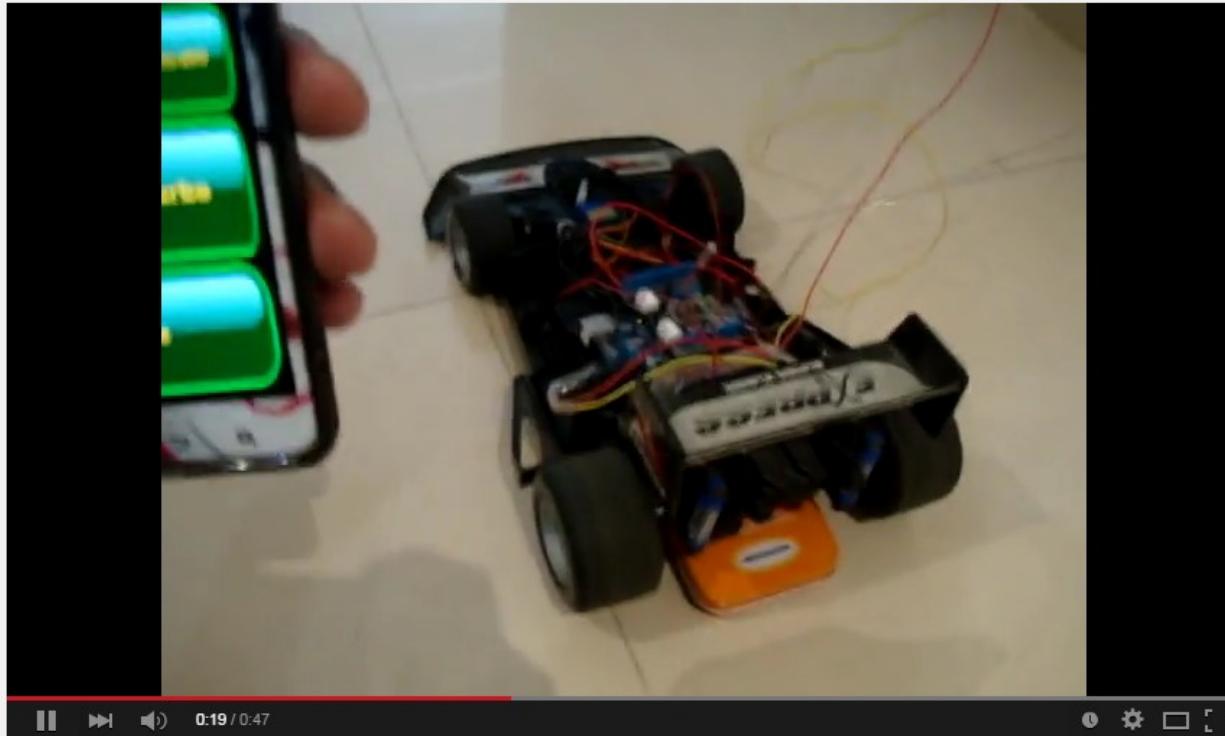
```
when GreenOn .Click
do if BluetoothClient1 .IsConnected
then call BluetoothClient1 .Send1ByteNumber
      number 3
```

```
when GreenOff .Click
do if BluetoothClient1 .IsConnected
then call BluetoothClient1 .Send1ByteNumber
      number 4
```

```
when Screen1 .BackPressed
do close application
```



아두이노 블루투스 자동차



Android Bluetooth Controlled RC car using arduino

https://www.youtube.com/watch?v=QZLNe4-_IuE



참고 사이트

- ODIY 강좌 (아두이노 및 라즈베리파이)
http://opensource.kofac.re.kr/edu/curriculum_list.do
- 아두이노 기초개념 및 예제
codingrun.com
- MAKESHARE 아두이노 기초 강좌
<http://makeshare.org>
- 아두이노 강좌
<http://wiki.vctec.co.kr/opensource/arduino>



결론 !

- 가능한 많은 **센서의 종류**를 아는 것이 당신의 아이디어를 구현할 수 있는 좋은 방법 !
- 만드는 것은 어렵지 않다! 너의 아이디어가 중요하다!
- 이제는 **Maker**의 시대!

