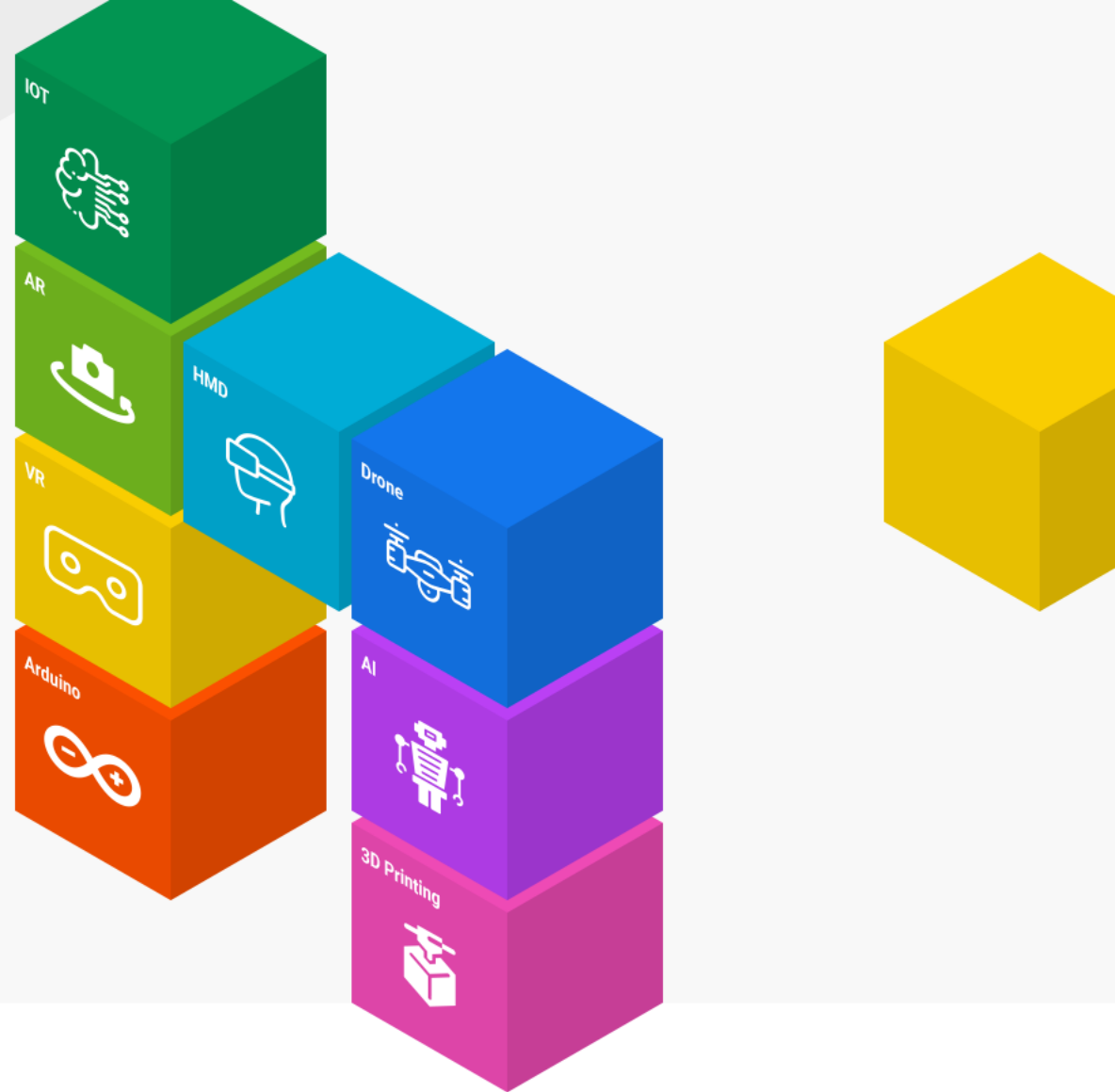


[아두이노 시뮬레이션 코딩]

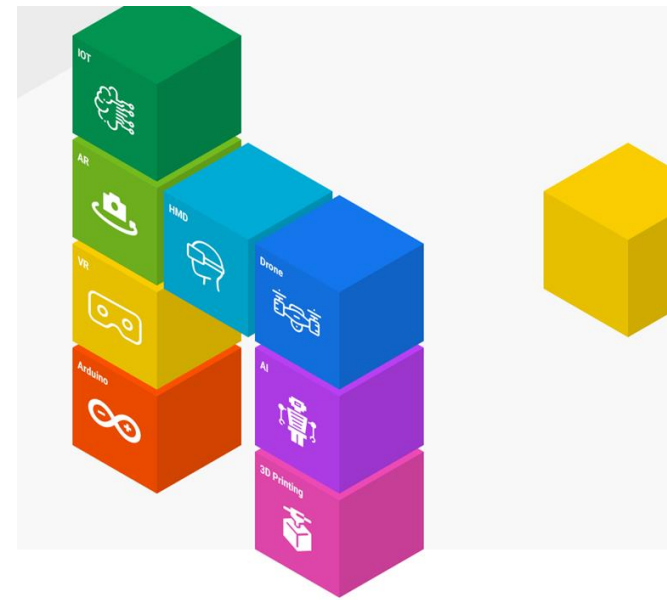
아두이노 기초 및 디지털 명령어



www.helloapps.co.kr

김 영 준 / 070-4417-1559 / splduino@gmail.com


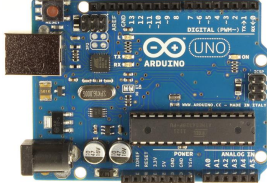
아두이노는 컴퓨터인가?



컴퓨터 vs 마이크로 컨트롤러




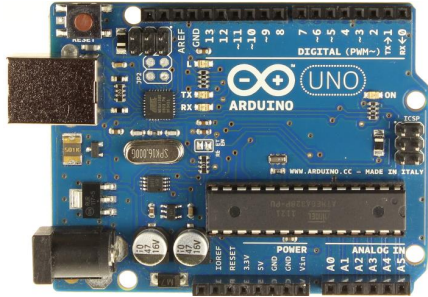
컴퓨터와 마이크로 컨트롤러 (마이컴)은 서로 다른 장치임

컴퓨터 (PC)	마이크로 컨트롤러 (마이컴)
<ul style="list-style-type: none">■ 운영체제(OS)를 가짐<ul style="list-style-type: none">- Windows, 리눅스, 안드로이드, iOS, Mac■ 일반적으로 키보드, 마우스, 모니터 등을 기본으로 지원■ 라즈베리파이는 리눅스 OS를 기반으로 하는 소형컴퓨터임 	<ul style="list-style-type: none">■ OS가 없음<ul style="list-style-type: none">- 펌웨어라 불리는 SW가 메모리에서 바로 실행■ 키보드, 마우스, 모니터를 기본으로 지원하지 않음■ 아두이노는 AVR C++ 언어로 개발되는 펌웨어로 실행되는 마이크로 컨트롤러임 

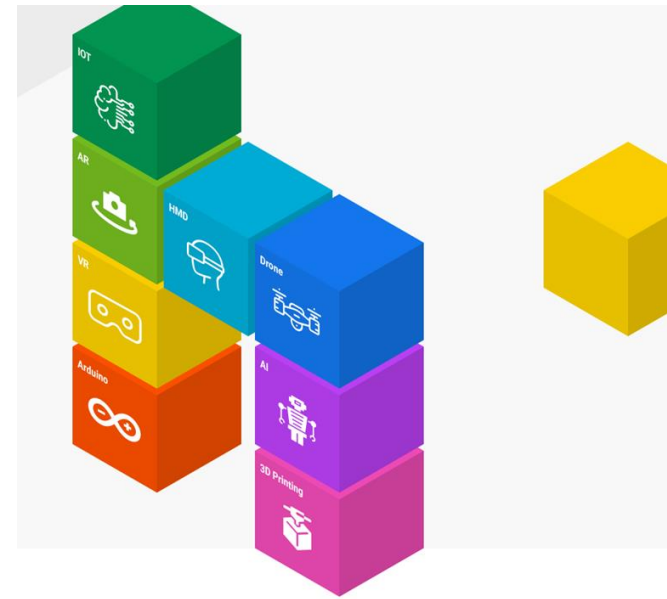
컴퓨터 vs 마이크로 컨트롤러



컴퓨터와 마이크로 컨트롤러 (마이컴)은 서로 다른 장치임

컴퓨터 (PC)	마이크로 컨트롤러 (마이컴)
<ul style="list-style-type: none">OS가 부팅된 후, 어플리케이션 SW가 실행됨 	<ul style="list-style-type: none">전원이 연결되자마자 메모리에 있는 SW가 바로 실행됨  <ul style="list-style-type: none">다양한 전자 기기에 내장되는 형태

아두이노는 H/W인가?

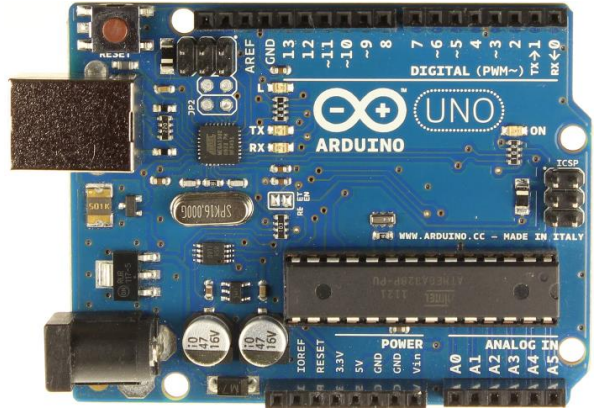


아두이노란?



2005년 이탈리아의 IDI라는 기관에서 초보자들이 디자인 작품을 쉽게 만들 수 있도록 고안

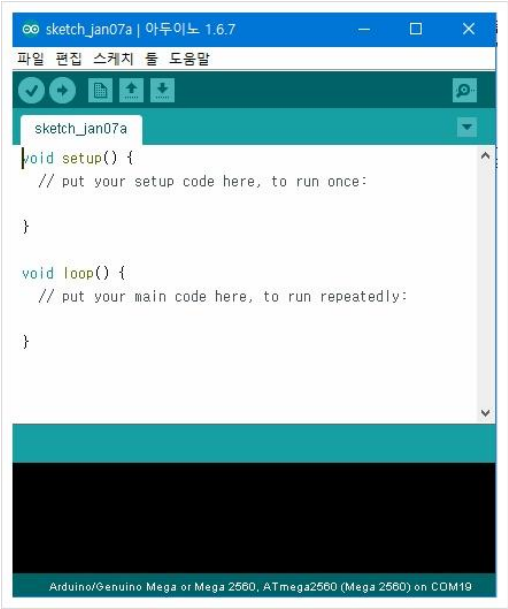
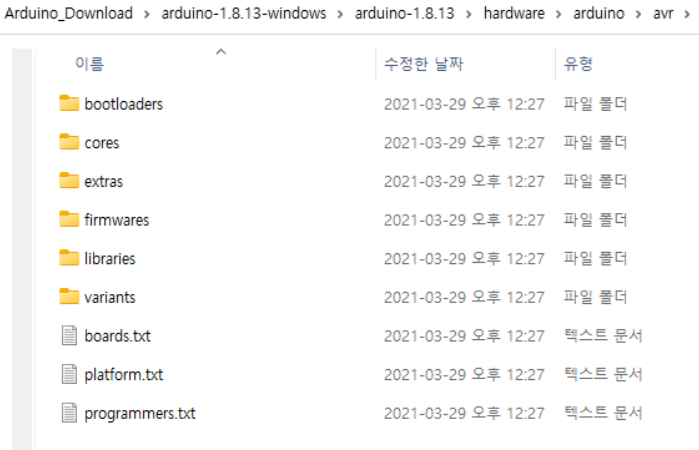
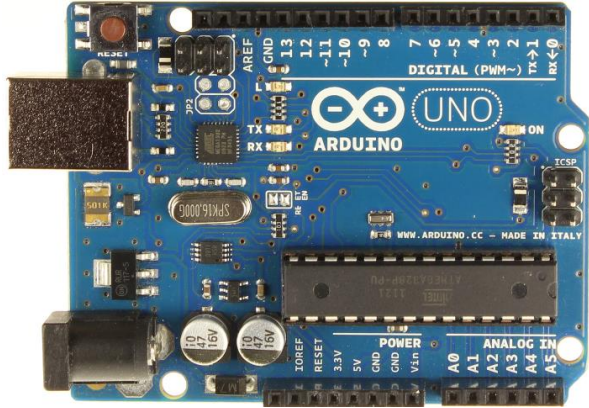
Arduino라는 이탈리아의 술집에서 모임을 가진 후, Arduino 이름 붙임
(원래 이 술집의 이름은 이탈리아의 왕의 이름을 따서 지었다고 함)

개발툴 IDE	SW 라이브러리	레퍼런스 보드
<ul style="list-style-type: none">MIT의 Processing 개발팀에서 오픈 소스로 개발한 개발용 툴을 가져다가 수정하여 사용함일반적으로 알고 있는 Sketch 코드는 아두이노에서 만든 이름이 아니라 프로세싱 툴에서 만들어진 이름임	<ul style="list-style-type: none">초보자들이 복잡한 AVR C++ 언어를 배우지 않고 쉬운 C언어 함수만으로 명령어를 사용할 수 있도록 라이브러리 형태로 정의함SW 라이브러리의 이름이 Arduino 임	<ul style="list-style-type: none">아두이노 우노 보드 외 다수 

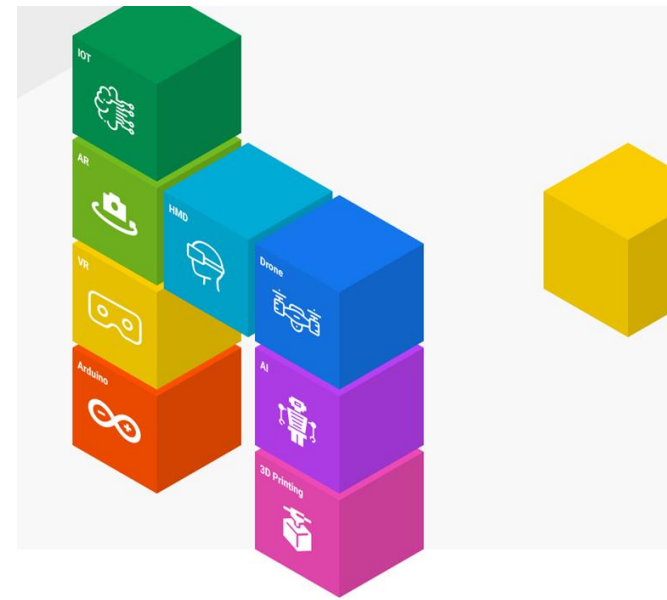
아두이노란?



아두이노는 개발툴, SW 라이브러리, 레퍼런스 하드웨어 보드로 구성됨

개발툴 IDE	SW 라이브러리	레퍼런스 보드																														
	 <table border="1"> <thead> <tr> <th>이름</th> <th>수정한 날짜</th> <th>유형</th> </tr> </thead> <tbody> <tr> <td>bootloaders</td> <td>2021-03-29 오후 12:27</td> <td>파일 폴더</td> </tr> <tr> <td>cores</td> <td>2021-03-29 오후 12:27</td> <td>파일 폴더</td> </tr> <tr> <td>extras</td> <td>2021-03-29 오후 12:27</td> <td>파일 폴더</td> </tr> <tr> <td>firmwares</td> <td>2021-03-29 오후 12:27</td> <td>파일 폴더</td> </tr> <tr> <td>libraries</td> <td>2021-03-29 오후 12:27</td> <td>파일 폴더</td> </tr> <tr> <td>variants</td> <td>2021-03-29 오후 12:27</td> <td>파일 폴더</td> </tr> <tr> <td>boards.txt</td> <td>2021-03-29 오후 12:27</td> <td>텍스트 문서</td> </tr> <tr> <td>platform.txt</td> <td>2021-03-29 오후 12:27</td> <td>텍스트 문서</td> </tr> <tr> <td>programmers.txt</td> <td>2021-03-29 오후 12:27</td> <td>텍스트 문서</td> </tr> </tbody> </table>	이름	수정한 날짜	유형	bootloaders	2021-03-29 오후 12:27	파일 폴더	cores	2021-03-29 오후 12:27	파일 폴더	extras	2021-03-29 오후 12:27	파일 폴더	firmwares	2021-03-29 오후 12:27	파일 폴더	libraries	2021-03-29 오후 12:27	파일 폴더	variants	2021-03-29 오후 12:27	파일 폴더	boards.txt	2021-03-29 오후 12:27	텍스트 문서	platform.txt	2021-03-29 오후 12:27	텍스트 문서	programmers.txt	2021-03-29 오후 12:27	텍스트 문서	
이름	수정한 날짜	유형																														
bootloaders	2021-03-29 오후 12:27	파일 폴더																														
cores	2021-03-29 오후 12:27	파일 폴더																														
extras	2021-03-29 오후 12:27	파일 폴더																														
firmwares	2021-03-29 오후 12:27	파일 폴더																														
libraries	2021-03-29 오후 12:27	파일 폴더																														
variants	2021-03-29 오후 12:27	파일 폴더																														
boards.txt	2021-03-29 오후 12:27	텍스트 문서																														
platform.txt	2021-03-29 오후 12:27	텍스트 문서																														
programmers.txt	2021-03-29 오후 12:27	텍스트 문서																														

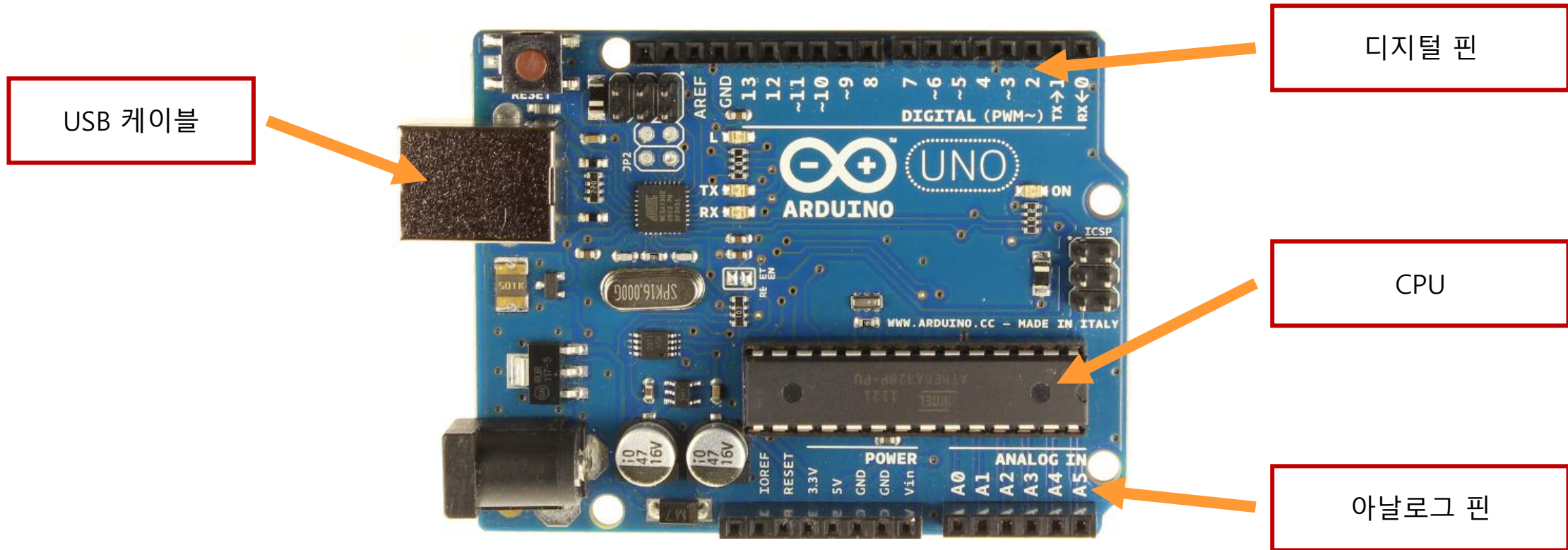
아두이노 보드의 구성



아두이노 보드의 구성



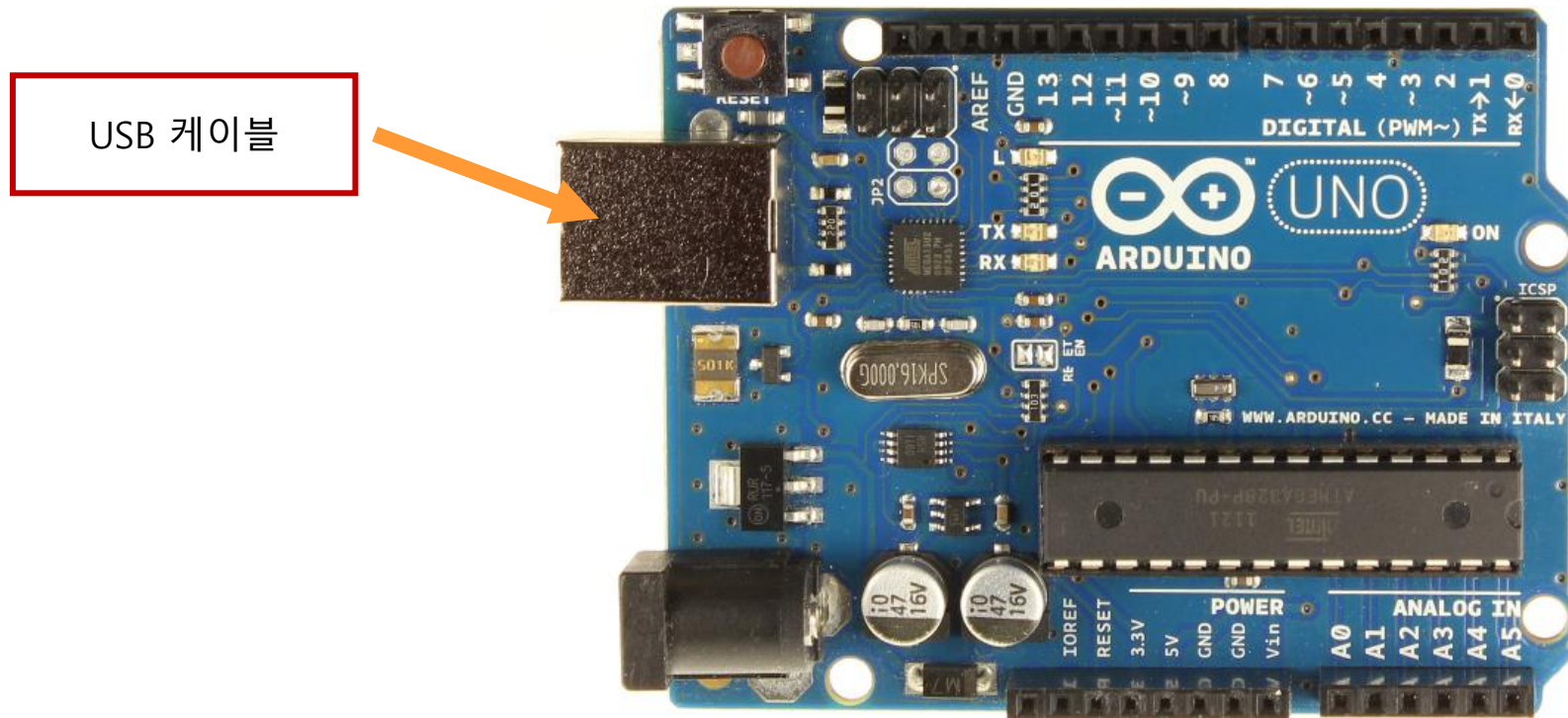
일반적으로 알고 있는 아두이노 우노 보드는 대표적인 레퍼런스 보드 중에 하나임



아두이노 보드의 구성



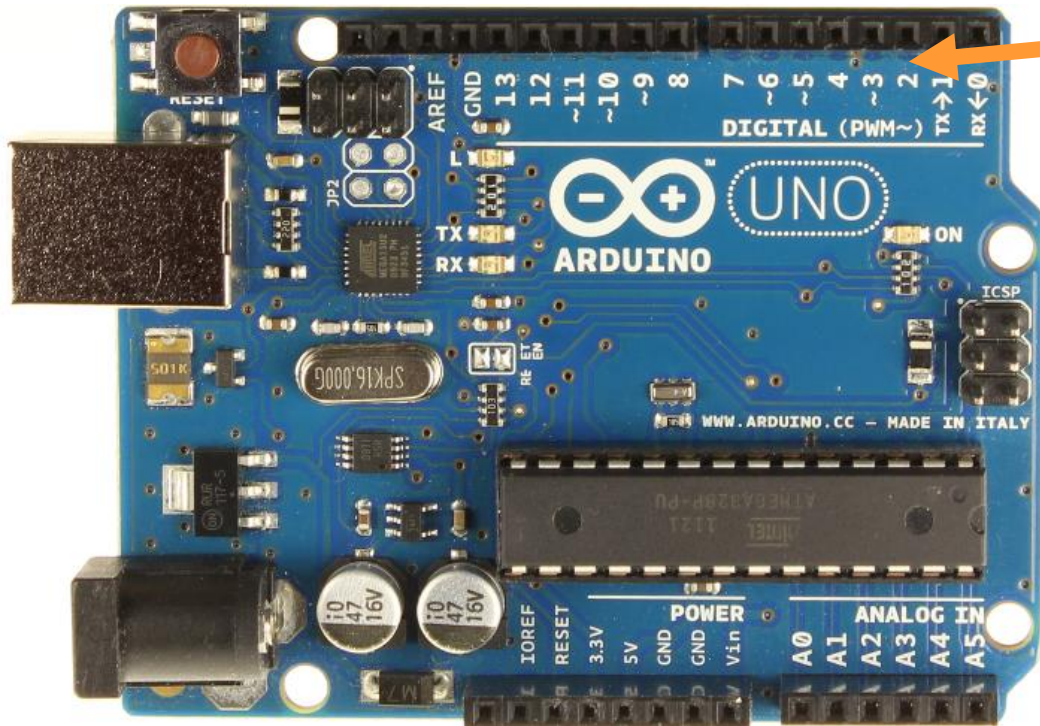
USB 케이블을 통해 프로그램을 업로드 하거나 PC와 데이터를 주고 받음



아두이노 보드의 구성



아두이노의 디지털 명령어를 사용하여 디지털 핀에 연결된 센서의 값을 읽거나 씀



디지털 핀

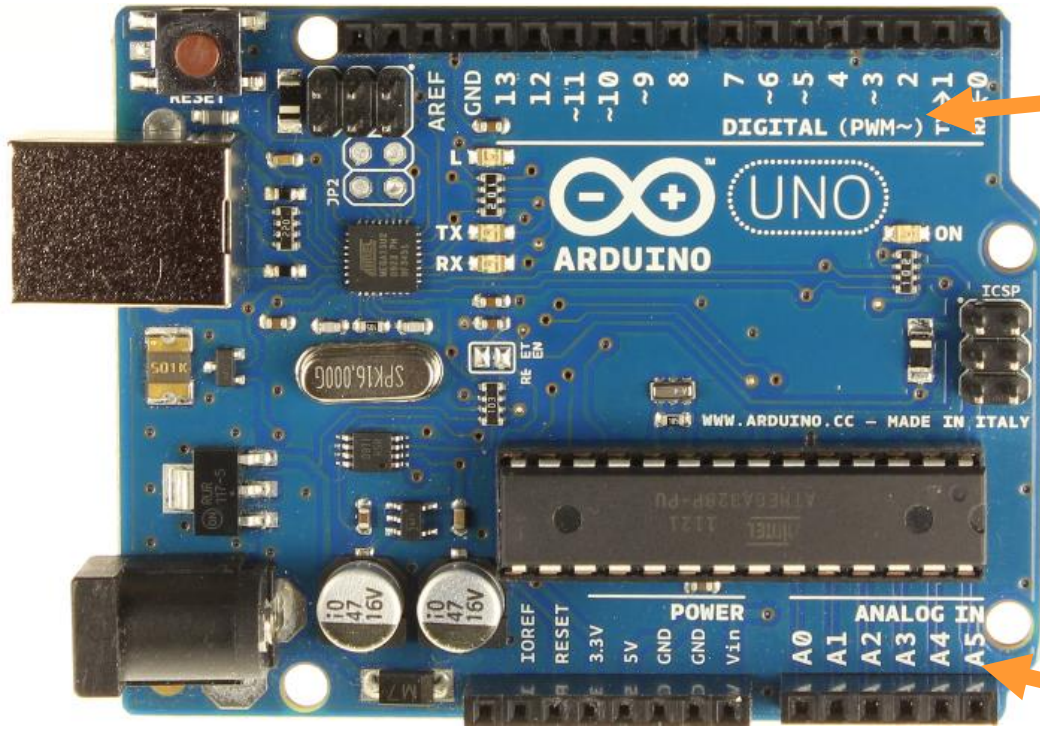
DigitalRead
DigitalWrite

AnalogWrite

아두이노 보드의 구성



아두이노의 디지털 명령어를 사용하여 디지털 핀에 연결된 센서의 값을 읽거나 씀

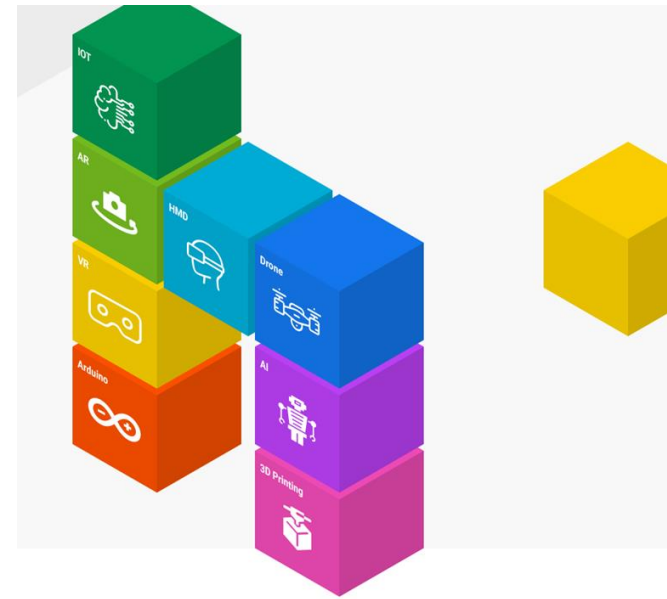


AnalogWrite

AnalogRead

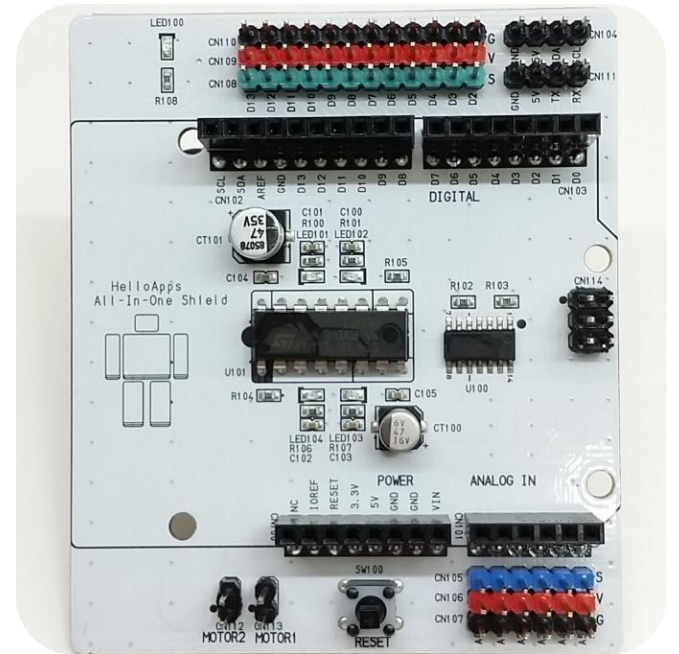
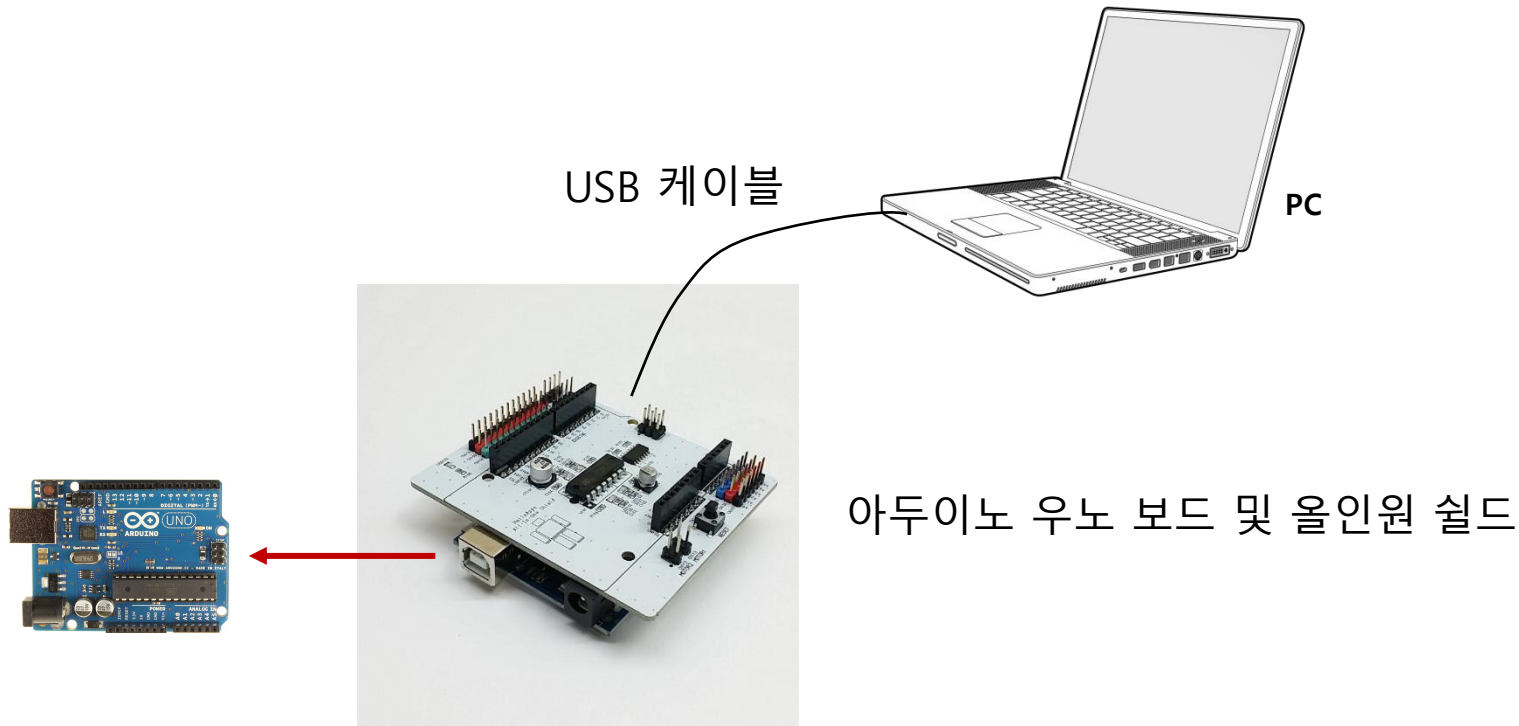
아날로그 핀

초보자용 올인원 쉴드를 이용한 부품 연결하기



아두이노 보드의 구성

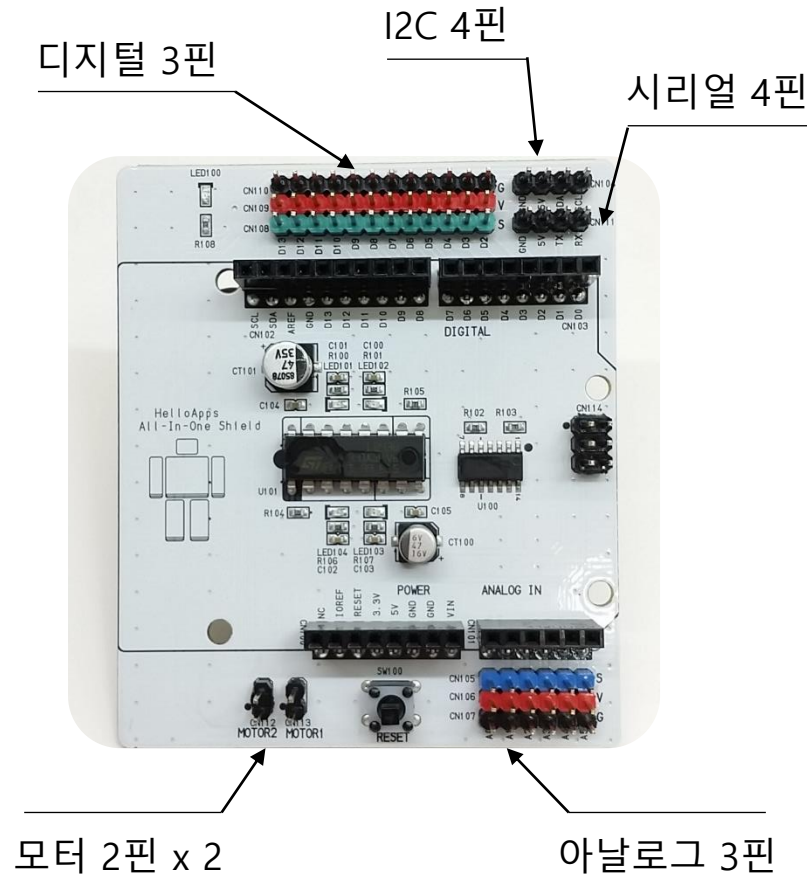
아두이노 우보 보드만 있는 경우, 센서들을 연결하기가 불편함
올인원 쉴드 보드를 사용할 경우, 간단하게 부품들을 연결할 수 있음



아두이노 보드의 구성

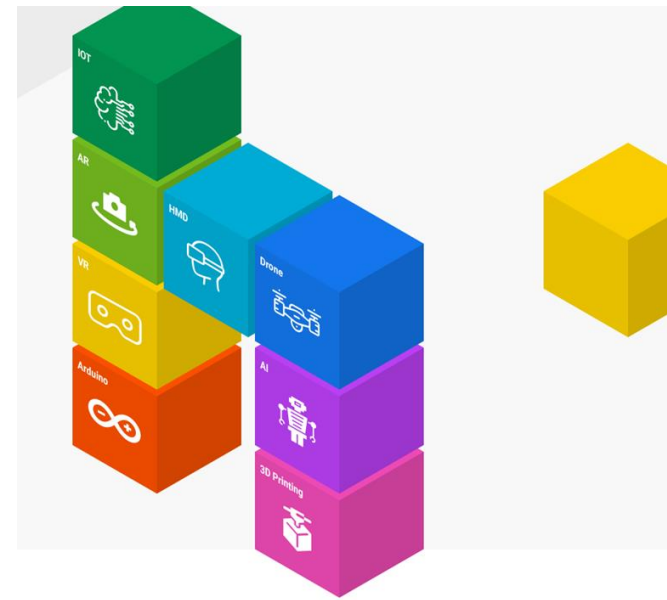


초보자용 센서 쉴드의 핀 이름



브레드보드가 필요없는
초보자용 아두이노 보드

디지털 부품 (LED) 연결하기



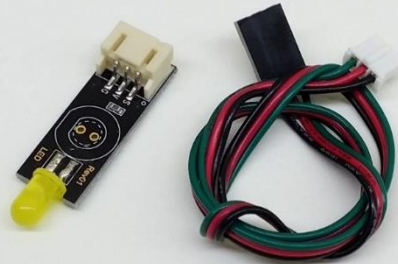
디지털 부품의 종류



디지털 부품은 On/Off 또는 High/Low 상태로 제어되는 부품임



디지털 버튼



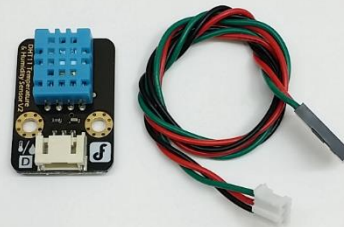
LED소자



서보모터



스피커



디지털 온습도센서

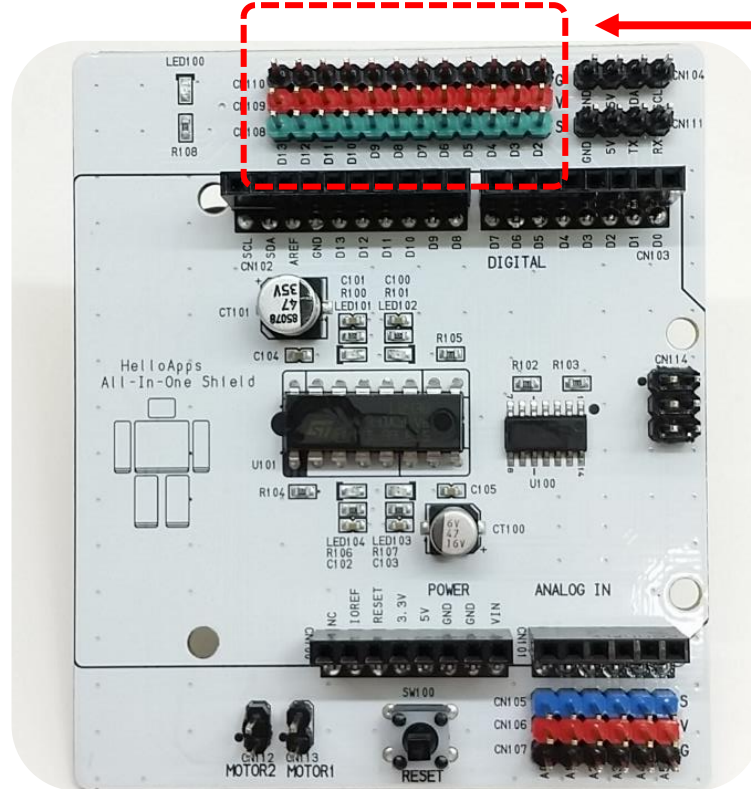


컬러 LED바

디지털 부품 (LED) 연결하기



디지털 부품은 디지털 핀에 연결합니다.



디지털 부품은 디지털
핀에 연결합니다.

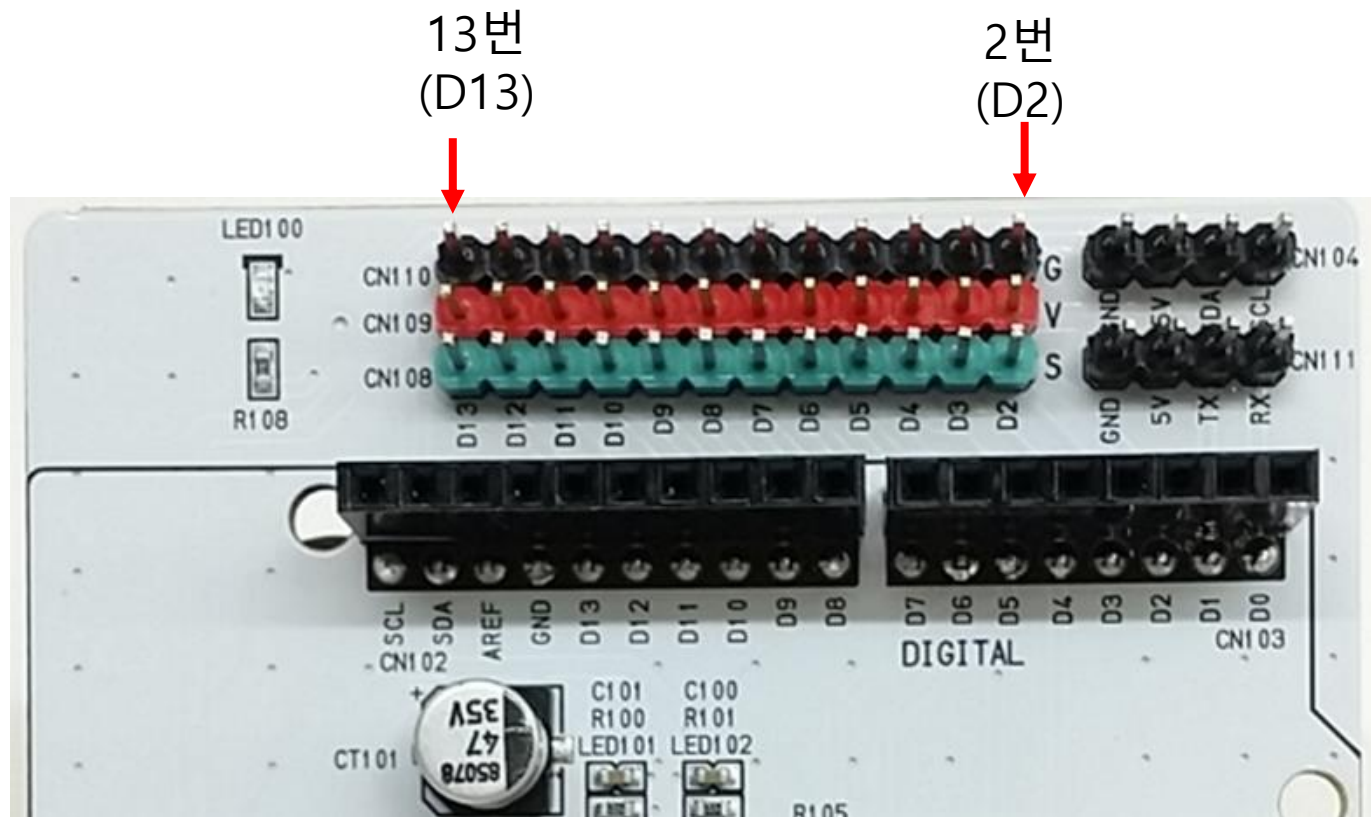
디지털 핀에는
2번 ~ 13번 까지
번호가 표시되어 있습니다.
(D2 ~ D13)

디지털 핀에 부품을 연결할
때에는 핀 번호를 확인해야
합니다.

디지털 부품 (LED) 연결하기



디지털핀은 0번 부터 13번 까지 14개이나, 실제 연결은 2번 부터 13번 까지만 가능

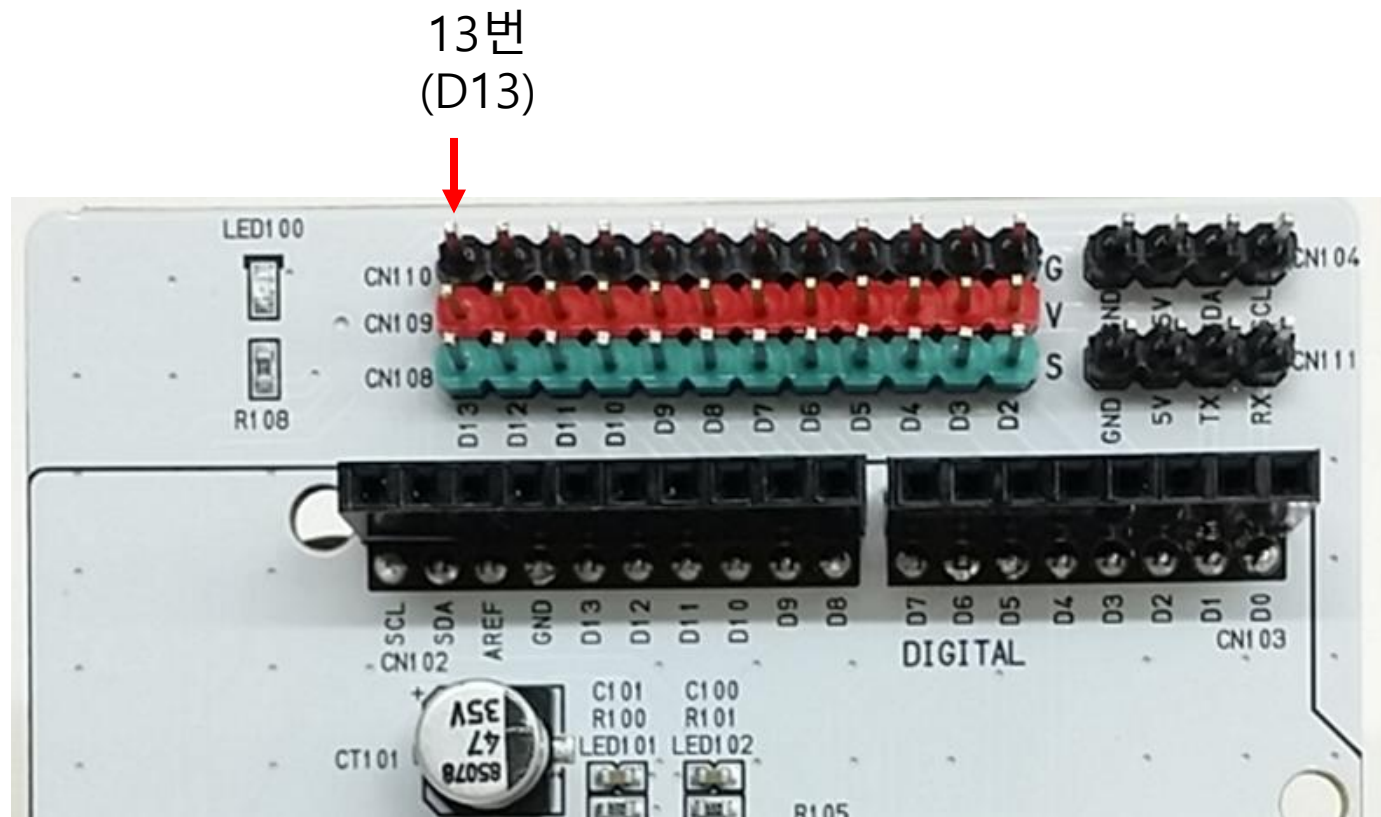


0번과 1번은 PC와 통신할 때 사용되기 때문에 2번 부터 13번 까지 12개의 핀을 사용할 수 있음

디지털 부품 (LED) 연결하기



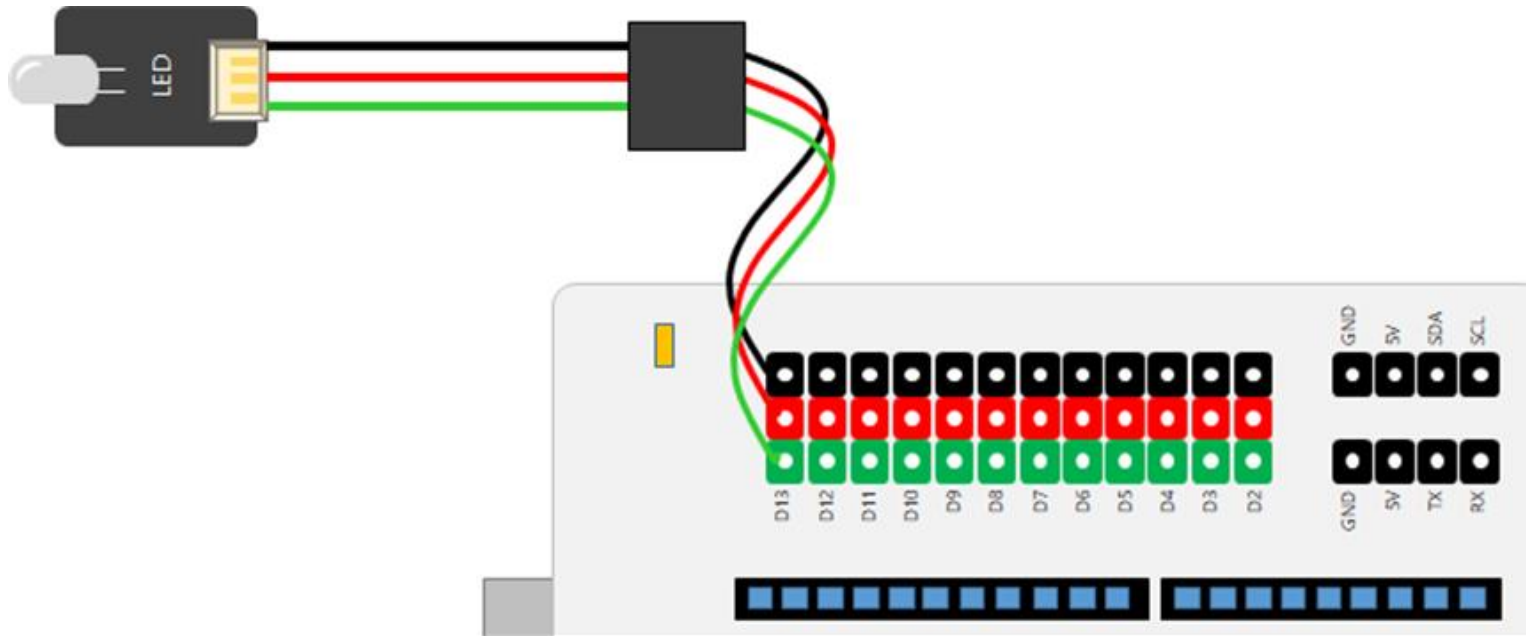
디지털 13번 핀에 LED 연결하기



디지털 부품 (LED) 연결하기



디지털 13번 핀에 LED 연결하기



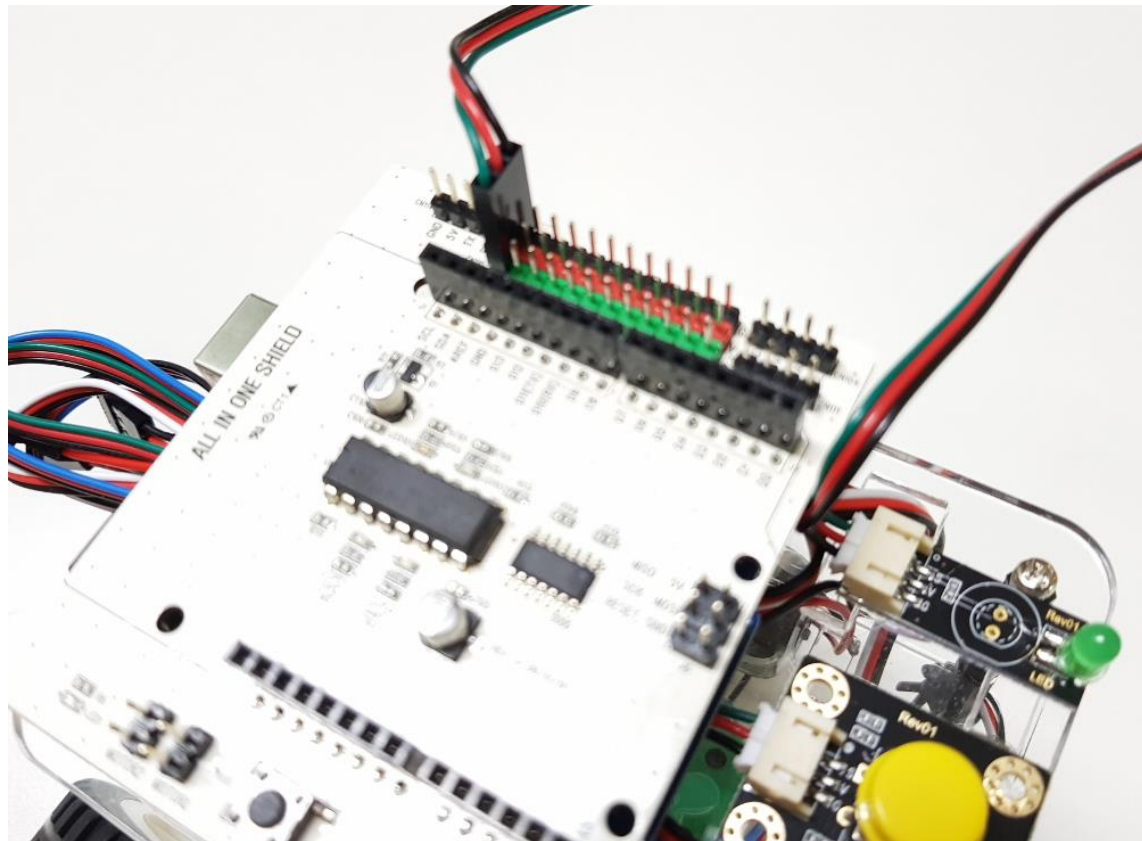
일반적으로 디지털 센서는
케이블이 녹색선으로
표시되어 있음

- GND (낮은 전압)
- VCC 또는 5V (높은 전압)
- 디지털 데이터 선

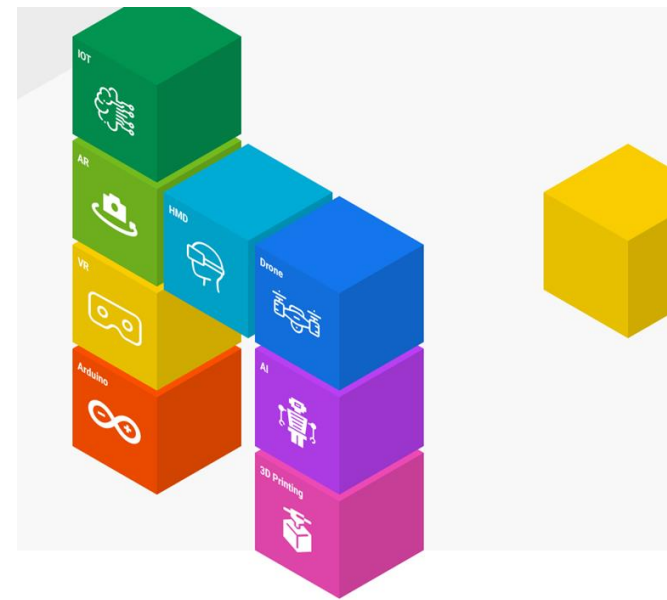
디지털 부품 (LED) 연결하기



디지털 13번 핀에 LED 연결하기

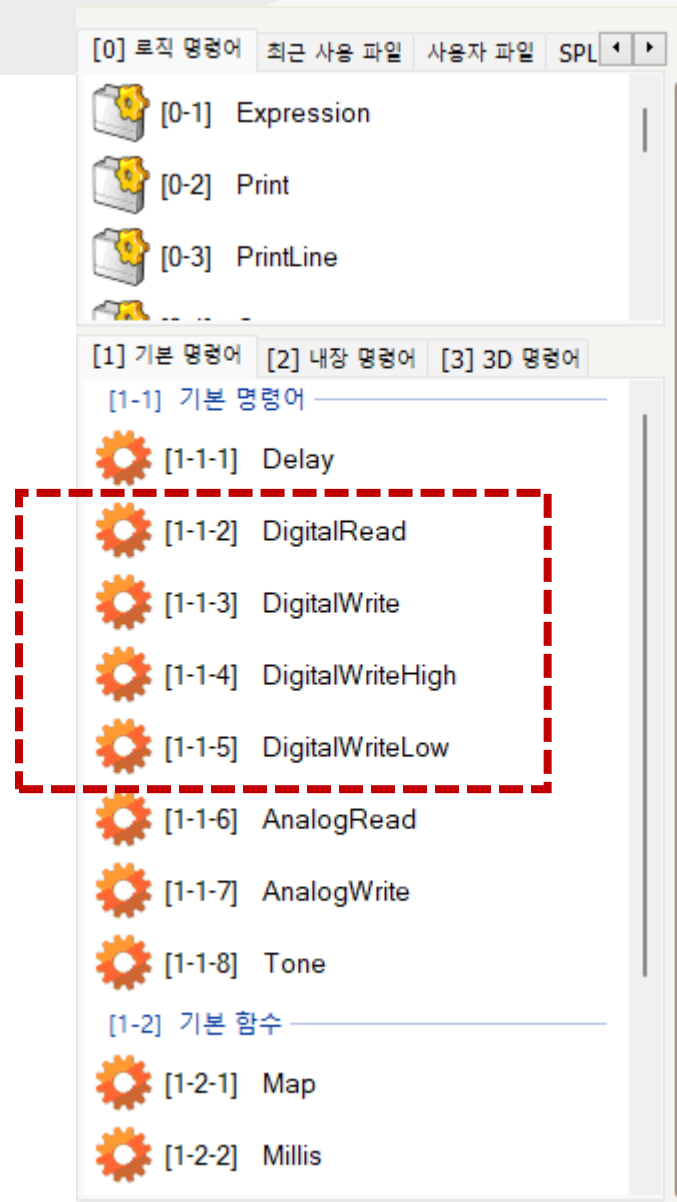


아두이노 디지털 명령어 기초



아두이노 디지털 명령어

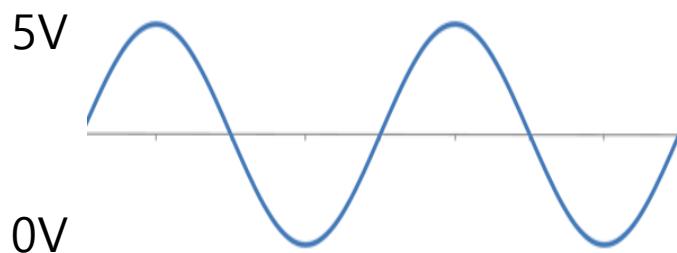
- 디지털 핀에 값을 쓸 때 사용하는 명령어
✓ DigitalWrite
- 디지털 핀에서 값을 읽어 올 때 사용하는 명령어
✓ DigitalRead



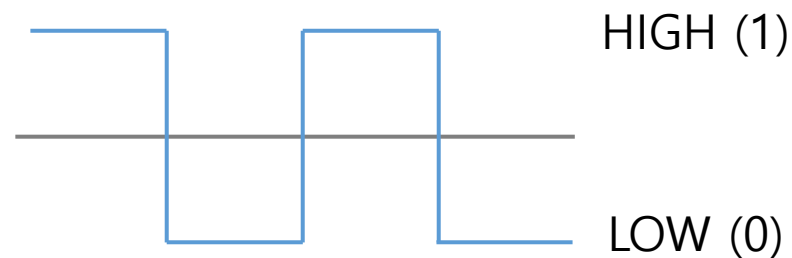
디지털 명령어에서의 값



- 0V와 5V 대신에 LOW와 HIGH라는 단어로 표시하는 이유



디지털 센서의 출력 또는 입력

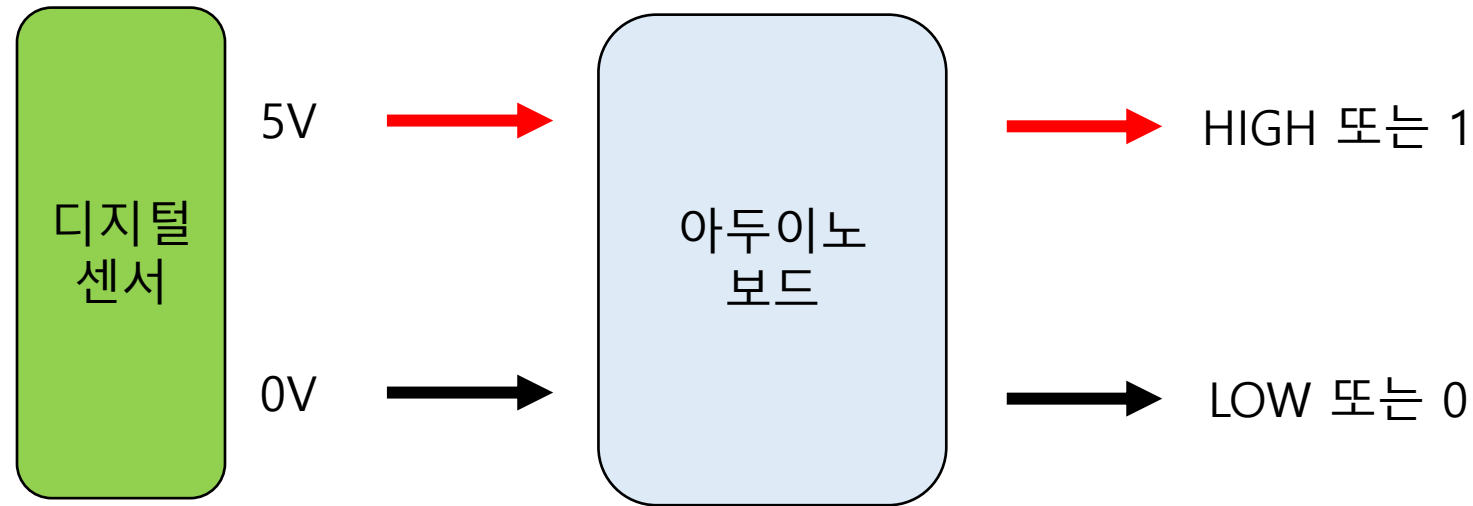


아두이노 보드에서의
디지털 센서값 처리

디지털 명령어에서의 값



- 아두이노 디지털 명령에서는 값이 0 또는 1을 사용하며, 0 대신에 LOW, 1 대신에 HIGH 라는 단어를 사용함



디지털 센서는 0V ~ 5V 사이의 전압값이 출력되며, 아두이노에서는 이 전압값이 0 또는 1로 처리됩니다.

0V ~ 2.5V 사이는 0으로 표시하고
2.5V ~ 5V 사이는 1로 표시합니다.

디지털 명령어에서의 값



- 13번 핀에 연결된 디지털 핀에 값을 쓰는 방법

아두이노에서 특별히 디지털 센서 값은 다음과 같이 예약어로 사용됨

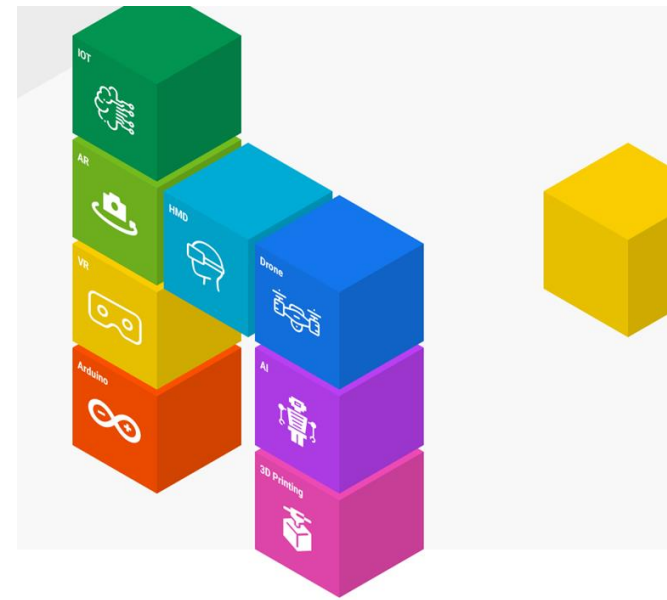
5V  HIGH 라는 단어를 사용합니다.

예) `DigitalWrite(13, HIGH)`

0V  LOW 라는 단어를 사용합니다.

예) `DigitalWrite(13, LOW)`

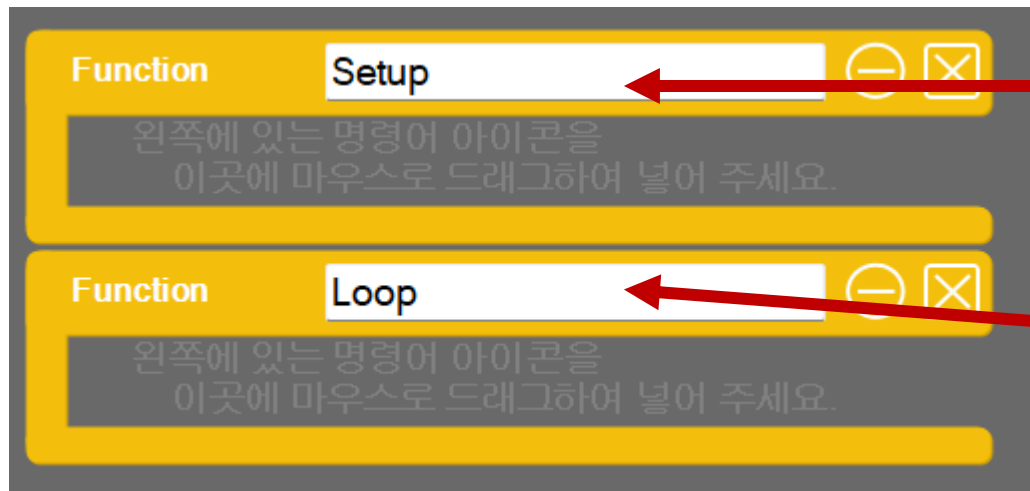
아두이노 기본 함수



Setup 함수와 Loop 함수



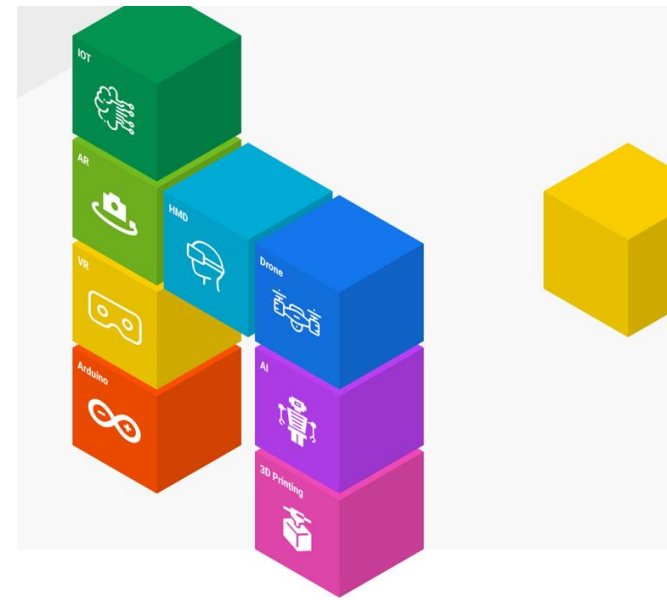
- Setup 함수는 맨처음 가장 먼저 실행되며, 한번만 실행되는 함수임
- Loop 함수는 Setup 함수 다음으로 실행되며, 전원이 꺼질 때까지 무한히 반복하여 실행됨



맨 처음 가장 먼저 한번만
실행됨

Setup 함수 실행이 끝난 후
Loop 함수가 계속 반복하여
실행됨

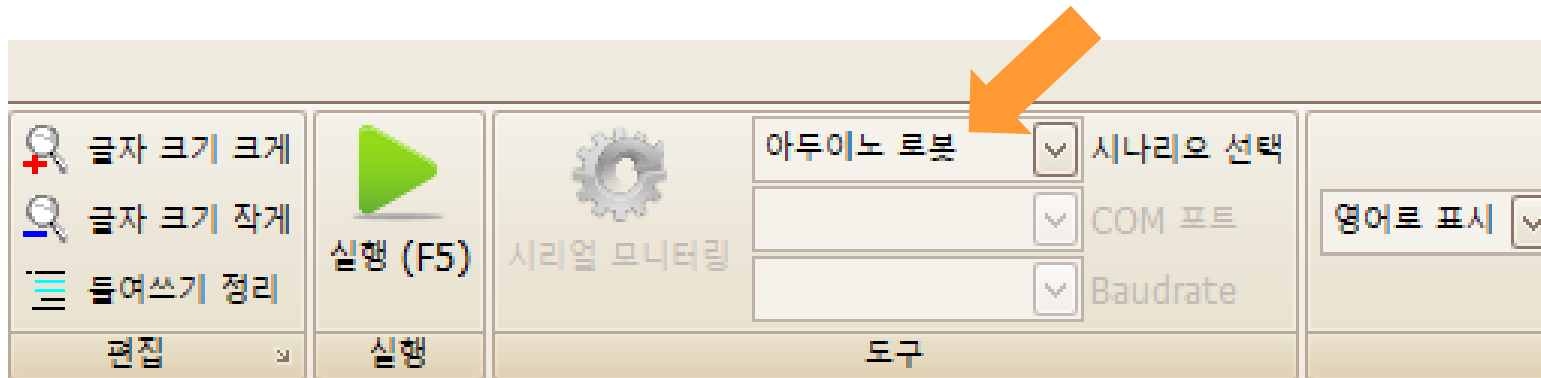
코딩으로 LED 제어하기



디지털 명령어에서의 값



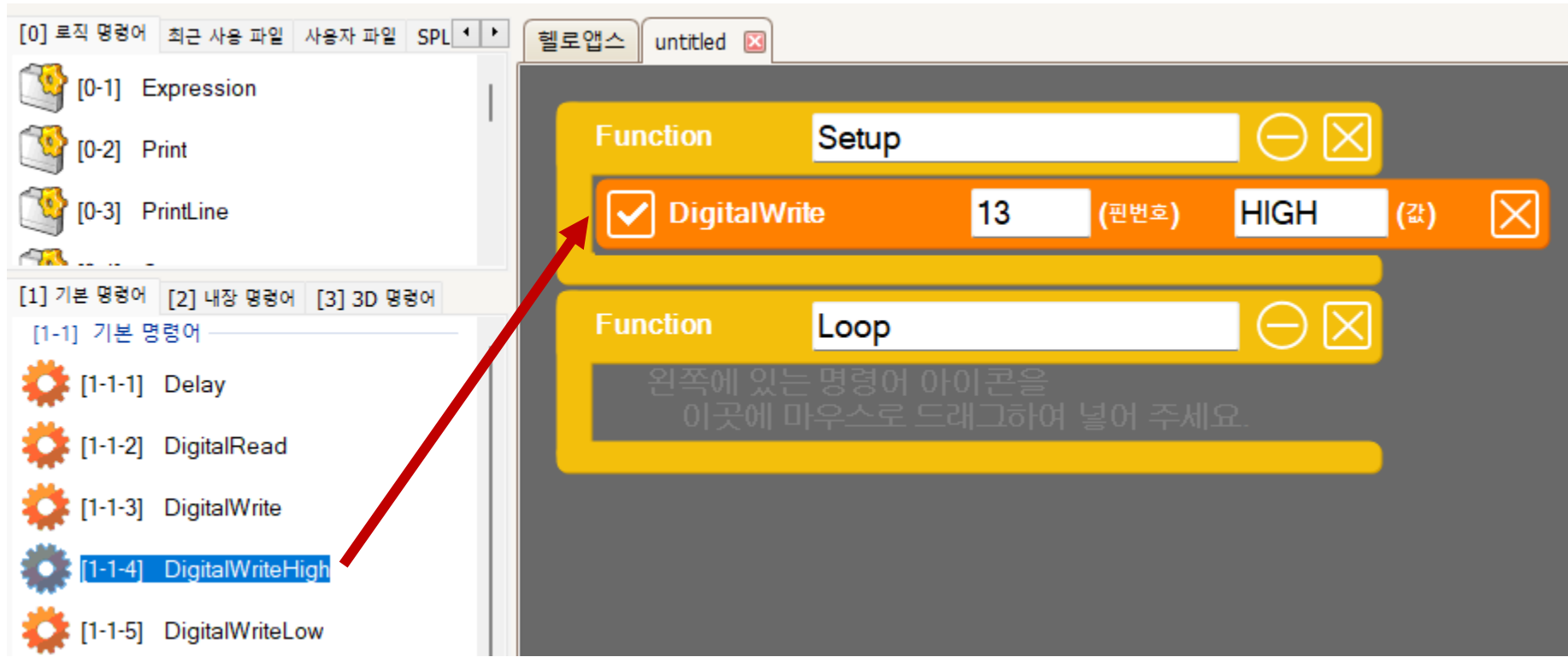
- 시나리오를 아두이노 로봇으로 선택합니다.



디지털 명령어에서의 값



- [1-1-4] DigitalWriteHigh 명령어를 Setup 함수 안에 드래그 하여 추가합니다.



DigitalWriteHigh 명령어의 의미



DigitalWrite 명령어에 HIGH 값을 자동으로 입력해 주는 명령어



디지털 13번 핀에
HIGH 값을 쓰라는 의미임

디지털 13번 핀에
연결된 부품에 5V 전압이
입력되도록 함

HIGH 값이 있으면
LED가 켜짐

HIGH 값은 반드시
대문자로만
입력해야 함

정상

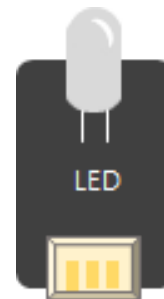
HIGH

잘못된 예

High

잘못된 예

high

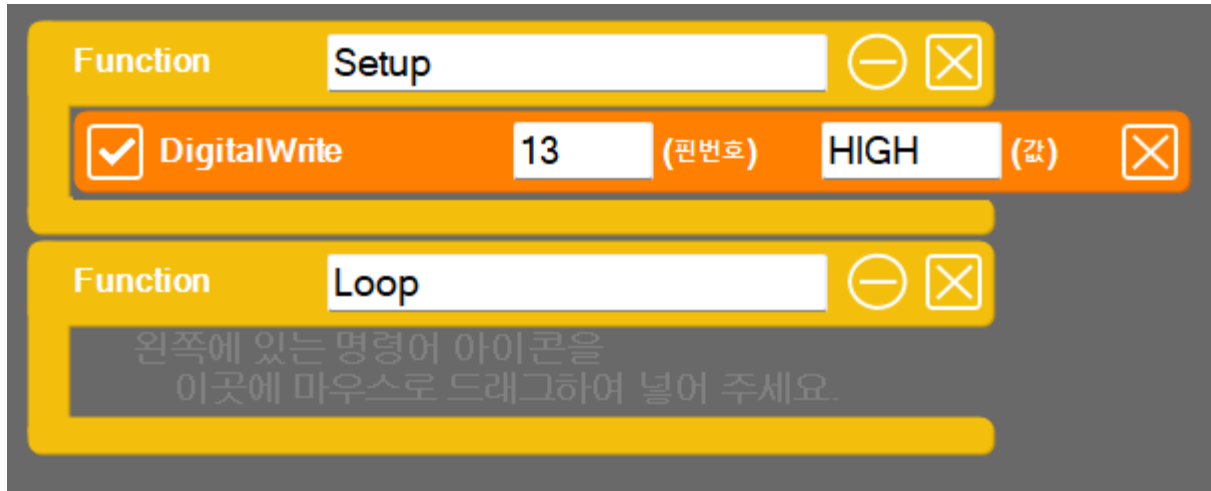


LED에 5V 전압이 입력되면 LED가 켜짐

디지털 명령어에서의 값



- 코드의 의미



한번만 실행되는 Setup 함수에
DigitalWrite 명령어가 있기 때문에
DigitalWrite 명령어는 한번만 실행됨

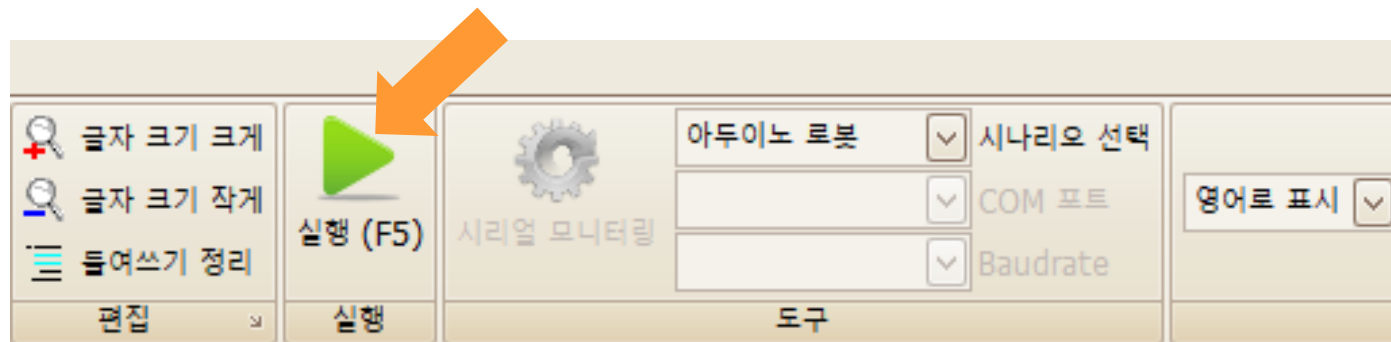
13번 핀에 HIGH 값을 출력하였기 때문에
LED에 5V 전압이 입력됨

예상되는 결과) 13번 핀에 연결된 LED가 계속 켜져 있어야 함

실행하기



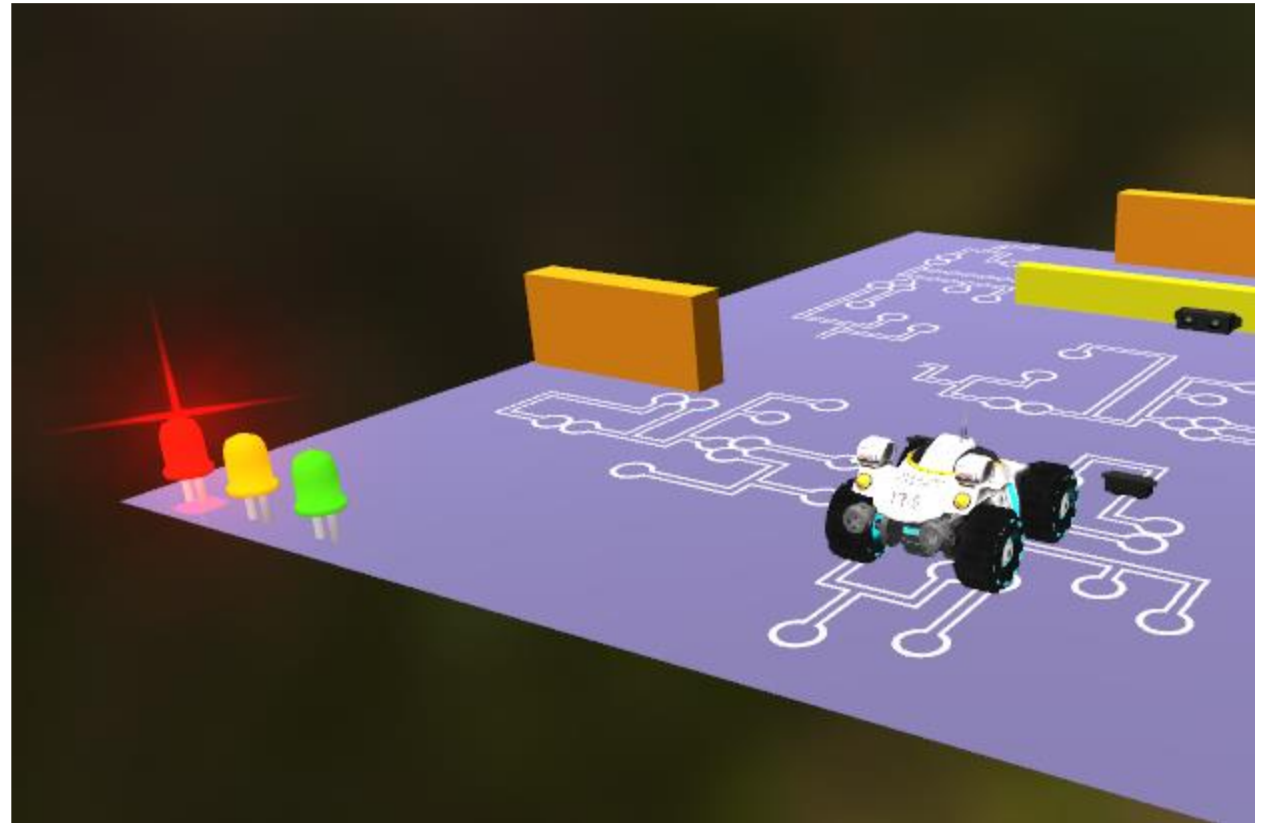
- 실행 버튼을 클릭합니다.



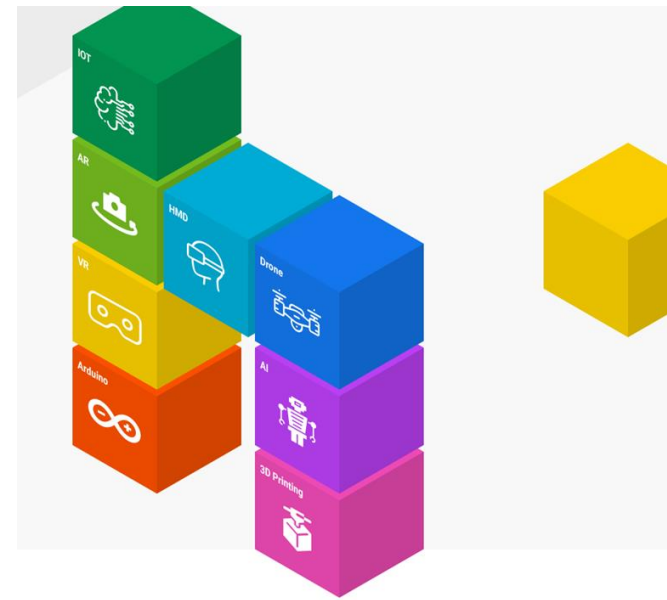
실행하기



RED 컬러 LED가 켜져 있게 됩니다.
마우스로 화면이 이동시킬 수 있습니다.



시뮬레이션 연결 환경 이해하기

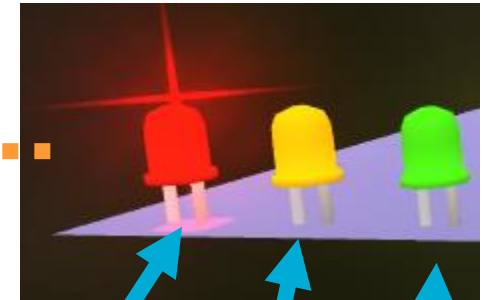
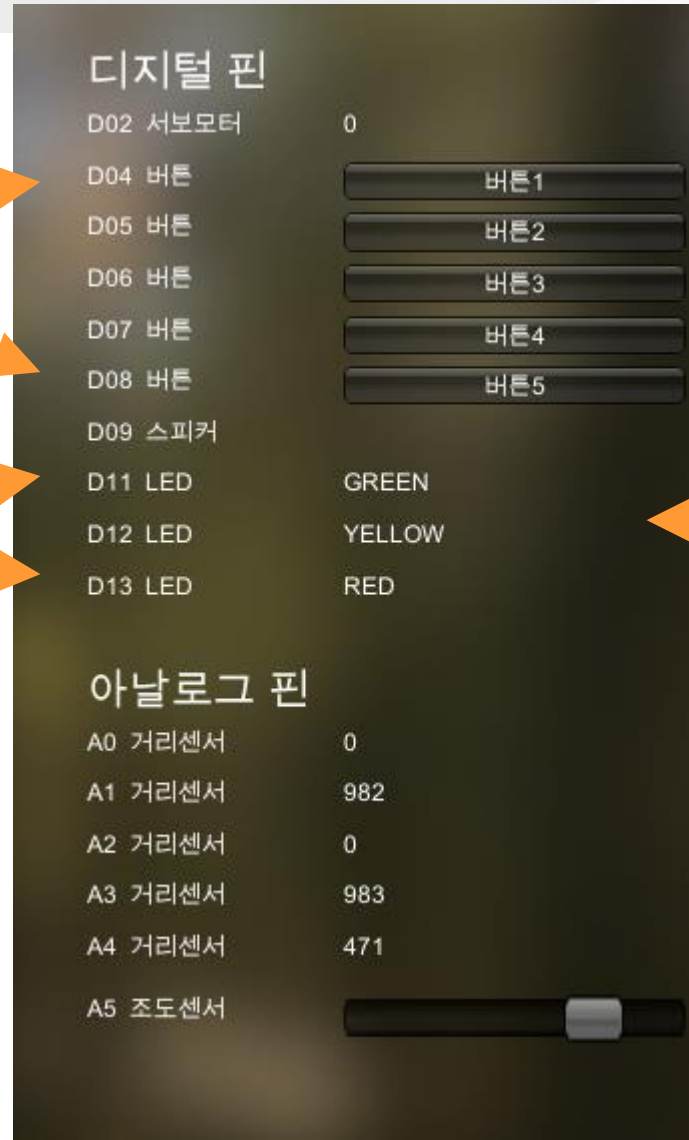


시뮬레이션 연결 환경 이해하기



4번부터 8번까지 5개의
버튼이 연결되어 있음

3개의 LED가
11번 부터 13번까지 연결되어
있음



13번

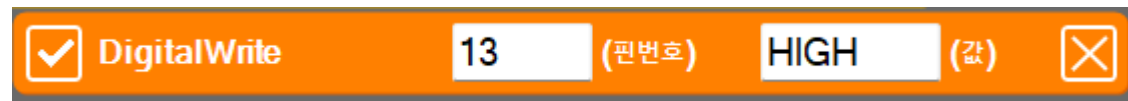
12번

11번

시뮬레이션 연결 환경 이해하기

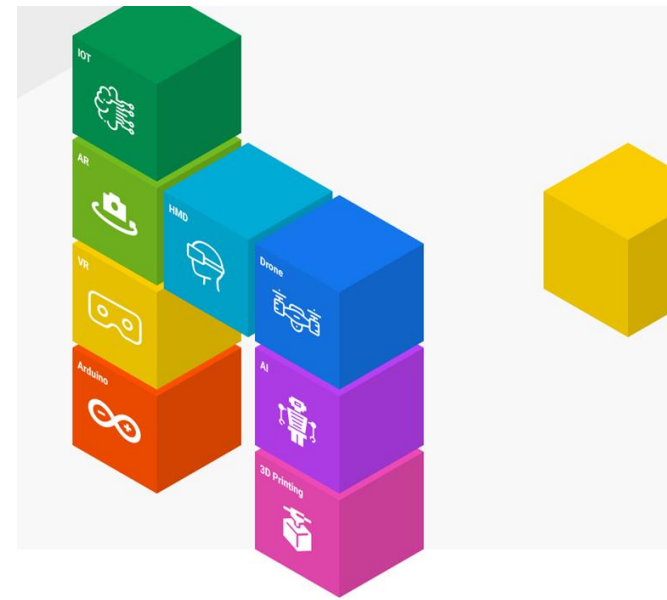


13번에 연결된 빨간색 LED가 켜진 상태로 표시됩니다.



13번

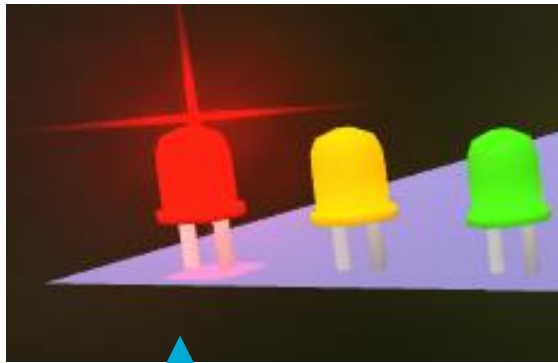
3초 후에 LED 끄기



3초 후에 LED 끄기

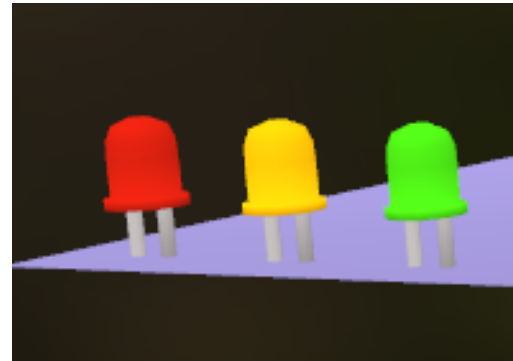


13번에 연결된 빨간색 LED를 프로그램 실행 후 자동으로 켜지게 한 후, 3초가 지나가면 다시 꺼진 상태로 만들어 보세요



13번

3초 후



Delay 명령어 (기다리기 명령어)



Delay 명령어를 주어진 밀리초 시간 동안 명령어 실행을 멈추는 기능을 수행함

Delay (밀리초)

밀리초	초
1,000	1초
3,000	3초
100	0.1초
10	0.01초

3초 후에 LED 끄기



[1-1-1] Delay 명령어를 마우스로 드래그 하여 DigitalWrite 명령어 아래에 추가합니다.

The screenshot shows the HelloApps IDE interface. On the left, a command palette lists various blocks. Under the '[1-1] 기본 명령어' (Basic Commands) category, the '[1-1-1] Delay' block is highlighted with a blue selection bar. A red arrow points from this block to the main workspace. The workspace displays a 'Function Setup' block containing two sub-blocks: 'DigitalWrite' (orange) and 'Delay' (green). The 'DigitalWrite' block is configured with pin number '13' and mode 'HIGH'. The 'Delay' block is configured with a duration of '1000' milliseconds. Below the 'Setup' block is a 'Function Loop' block, which is currently empty. A grey text box within the Loop block contains the instruction: '왼쪽에 있는 명령어 아이콘을 이곳에 마우스로 드래그하여 넣어 주세요.' (Please drag the command icon from the left into this area).

3초 후에 LED 끄기



Delay 명령어의 값을 1,000밀리초에서 3,000밀리초로 수정합니다.

The image shows a Scratch code editor with two function blocks. The first function is named 'Setup' and contains two blocks: 'DigitalWrite' with pin number '13' and mode 'HIGH', and a 'Delay' block with a value of '3000' (milliseconds). A red arrow points from a text box on the right to the '3000' value in the Delay block. The second function is named 'Loop' and contains a placeholder text: '왼쪽에 있는 명령어 아이콘을 이곳에 마우스로 드래그하여 넣어 주세요.'

3초를 기다림

3초 후에 LED 끄기



[1-1-5] DigitalWriteLow 명령어를 드래그 하여 Delay 명령어 아래에 추가합니다.

The screenshot shows the HelloApps IDE interface. On the left, there is a sidebar with a tree view of blocks. The '기본 명령어' (Basic Commands) category is expanded, showing a list of blocks including 'Delay', 'DigitalRead', 'DigitalWrite', 'DigitalWriteHigh', and 'DigitalWriteLow'. The 'DigitalWriteLow' block is highlighted with a blue selection bar. A red arrow points from this block to the 'Loop' function in the main workspace. The 'Loop' function is currently empty, with a text prompt in Korean: '왼쪽에 있는 명령어 아이콘을 이곳에 마우스로 드래그하여 넣어 주세요.' (Please drag the command icon from the left here with the mouse and put it in).

Function Setup

- ☒ DigitalWrite 13 (핀번호) HIGH (값)
- ☒ Delay 3000 (밀리초)
- ☒ DigitalWrite 13 (핀번호) LOW (값)

Function Loop

왼쪽에 있는 명령어 아이콘을
이곳에 마우스로 드래그하여 넣어 주세요.

DigitalWriteLow 명령어의 의미

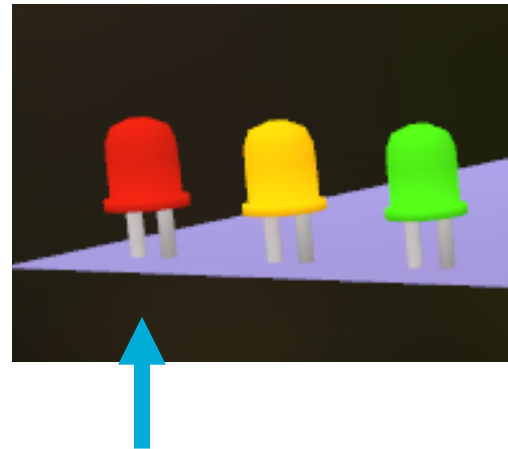


DigitalWrite 명령어에 LOW 값을 자동으로 입력해 주는 명령어



LOW 값이 입력되면
낮은 전압 (0V)이 LED에
전달되어 LED가
꺼지게 됨

LOW 값은 반드시
대문자로만
입력해야 함



13번

정상

LOW

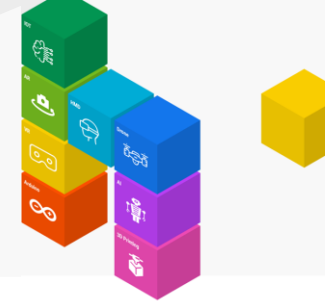
잘못된 예

Low

잘못된 예

low

실행하기



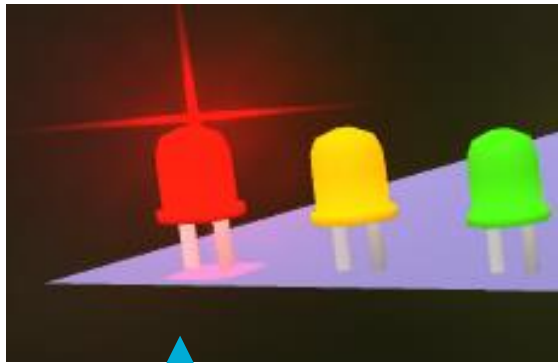
- 실행 버튼을 클릭합니다.



3초 후에 LED 끄기

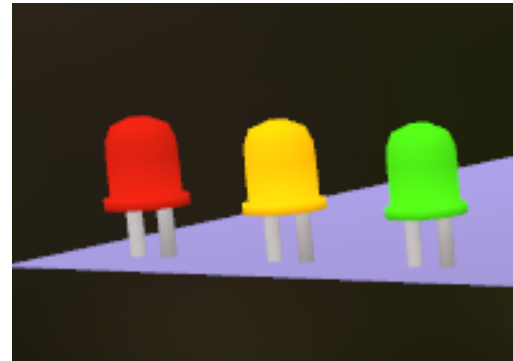


13번에 연결된 빨간색 LED를 프로그램 실행 후 자동으로 켜지며, 3초가 지나가면 자동으로 꺼지게 됩니다.

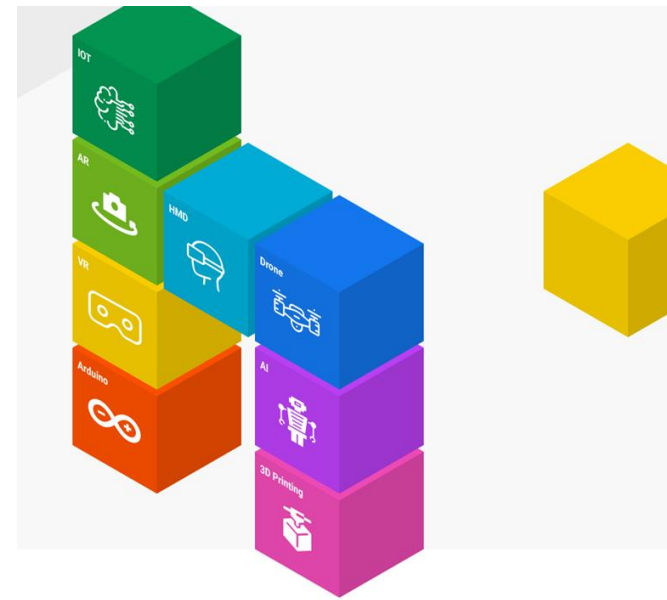


13번

3초 후



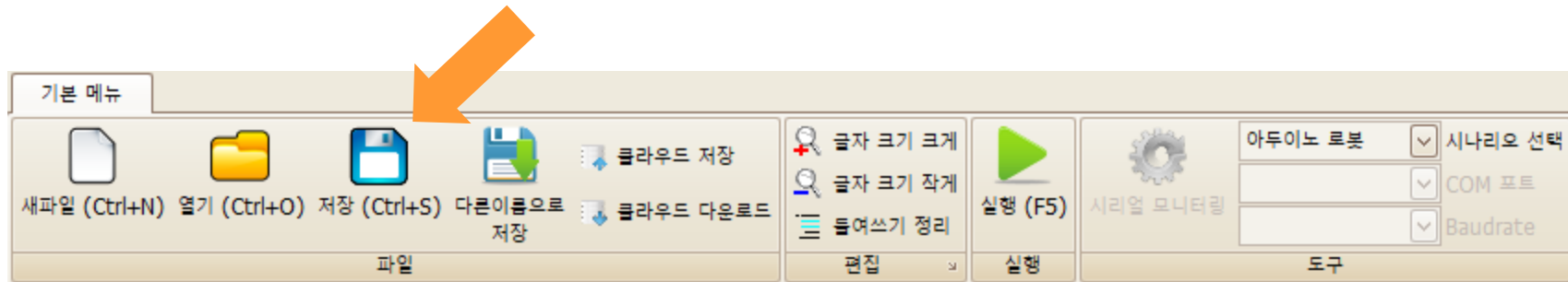
스크립트 변환 결과 확인하기



SPL 스크립트 변환 결과

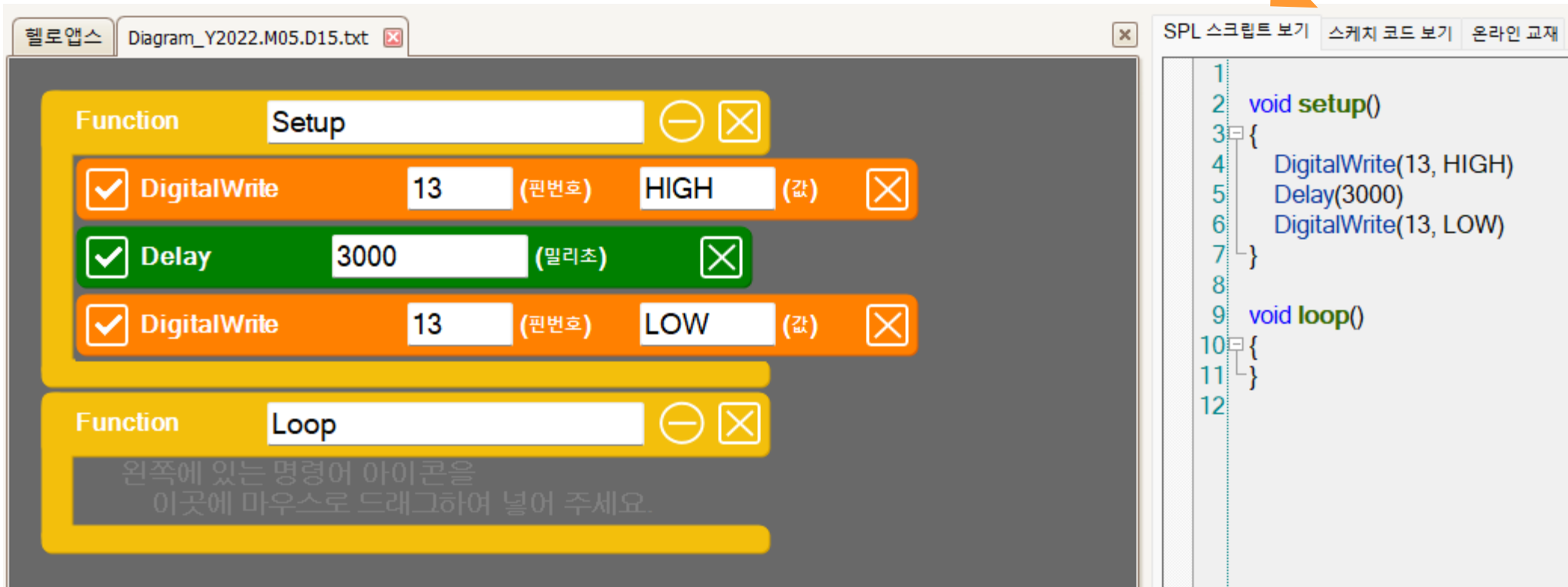


저장 버튼을 클릭하면 우측에 SPL 스크립트 변환 결과가 표시됩니다.



SPL 스크립트 변환 결과

저장 버튼을 클릭하면 우측에 SPL 스크립트 변환 결과가 표시됩니다.




The screenshot displays the HelloApps application window. The main workspace on the left contains a block-based script with two functions: 'Setup' and 'Loop'. The 'Setup' function includes three blocks: 'DigitalWrite' (pin 13, HIGH), 'Delay' (3000 ms), and 'DigitalWrite' (pin 13, LOW). The 'Loop' function is currently empty, with a placeholder text: '왼쪽에 있는 명령어 아이콘을 이곳에 마우스로 드래그하여 넣어 주세요.' (Drag the command icons from the left here with the mouse). On the right side, there are three tabs: 'SPL 스크립트 보기' (SPL Script View), '스케치 코드 보기' (Sketch Code View), and '온라인 교재' (Online Manual). The 'SPL 스크립트 보기' tab is active, showing the converted C++ code. An orange arrow points from the text above to this tab. The code is as follows:

```
1  
2 void setup()  
3 {  
4   digitalWrite(13, HIGH)  
5   delay(3000)  
6   digitalWrite(13, LOW)  
7 }  
8  
9 void loop()  
10 {  
11 }  
12
```

SPL 스크립트 변환 결과



SPL (Simple Programming Language)




```
SPL 스크립트 보기 | 스케치 코드 보기 | 온라인 교재
1
2 void setup()
3 {
4   DigitalWrite(13, HIGH)
5   Delay(3000)
6   DigitalWrite(13, LOW)
7 }
8
9 void loop()
10 {
11 }
12
```

SPL 스크립트는
C언어 문법을 초보자
용으로 간단히 표현한
스크립트 입니다.

SPL 스크립트 변환 결과



스케치 코드 보기



```
SPL 스크립트 보기 | 스케치 코드 보기 | 온라인 교재
1
2 void setup();
3 void loop();
4 void setup()
5 {
6     pinMode(13, OUTPUT);
7     digitalWrite(13, HIGH);
8     delay(3000);
9     digitalWrite(13, LOW);
10 }
11 void loop()
12 {
13 }
14
15
```

스케치 코드는 원래 아두이노에서
입력되는 소스 코드 형태입니다.

블록 코딩을 작성하면 자동으로
우측에 스케치 코드가 생성됩니다.

스케치 코드는 AVR C++ 언어로
되어 있어서 C 문법을 모르는
초보자가 작성하기 어렵습니다.