

아두이노 프로그래밍

1일차 - Part3 디지털 명령어

강사: 김영준 헬로앱스 대표
헬로앱스 (www.helloapps.co.kr)

아두이노 명령어

▶ 명령어의 기본 규칙

아두이노 명령어는 크게 디지털 명령어와 아날로그 명령어로 구분되며, 핀으로 값을 출력할 경우에는 뒤에 Write 단어가 붙고, 값을 읽어 올 때에는 뒤에 Read 단어가 붙습니다.

- DigitalRead
- DigitalWrite
- AnalogRead
- AnalogWrite

실제 사용되는 아두이노 명령어 더 많지만, 일단 위의 규칙만 알아도 다양한 주제의 아두이노 창작 작품을 구현할 수 있습니다.

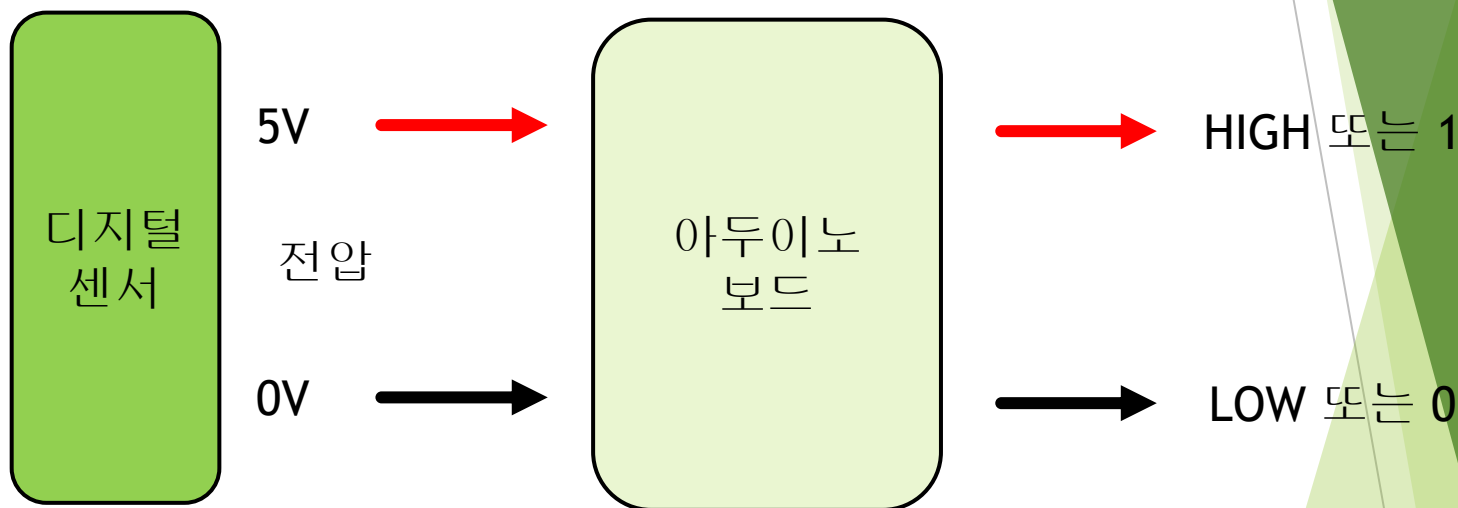
아두이노 명령어 구성

- ▶ 디지털 명령어
 - ▶ 0 (LOW) 또는 1 (HIGH) 값을 가지는 부품 제어
- ▶ 아날로그 명령어
 - ▶ 0 ~ 1023 사이의 값을 가지는 센서 제어
- ▶ 소리 생성 명령어
 - ▶ 소리를 발생시키는 명령어
- ▶ 모터 제어 명령어
 - ▶ 서보모터 및 다양한 모터 제어
- ▶ LCD 명령어
 - ▶ 문자 출력 장치
- ▶ 통신 명령어
 - ▶ 인터넷, 블루투스 등의 통신

아두이노에서의 센서값

디지털 센서

▶ 디지털 센서 값



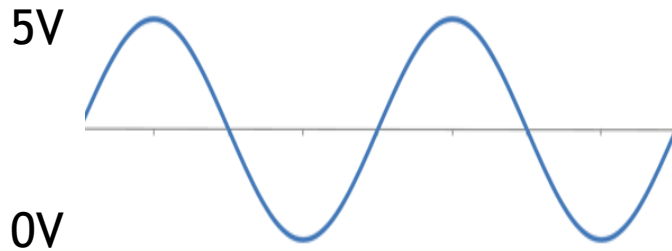
디지털 센서는 **0V ~ 5V** 사이의 전압값이 출력되며, 아두이노에서는 이 전압값이 **0** 또는 **1**로 처리됩니다.

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

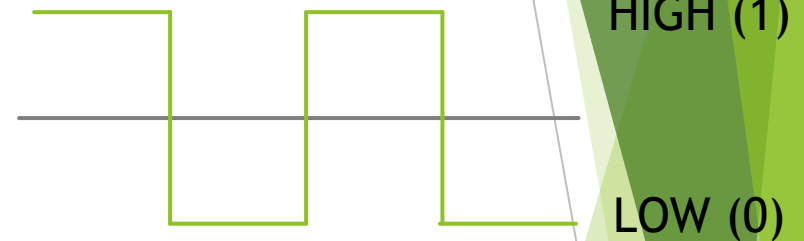
아두이노에서의 센서값

디지털 센서

▶ 디지털 센서 값



디지털 센서의 출력 또는 입력



아두이노 보드에서의
디지털 센서값 처리

▶ 디지털 센서 값

아두이노에서 특별히 디지털 센서 값은 다음과 같이 예약어로 사용됩니다.

5V  HIGH 라는 단어를 사용합니다.

예) `DigitalWrite(13, HIGH)`

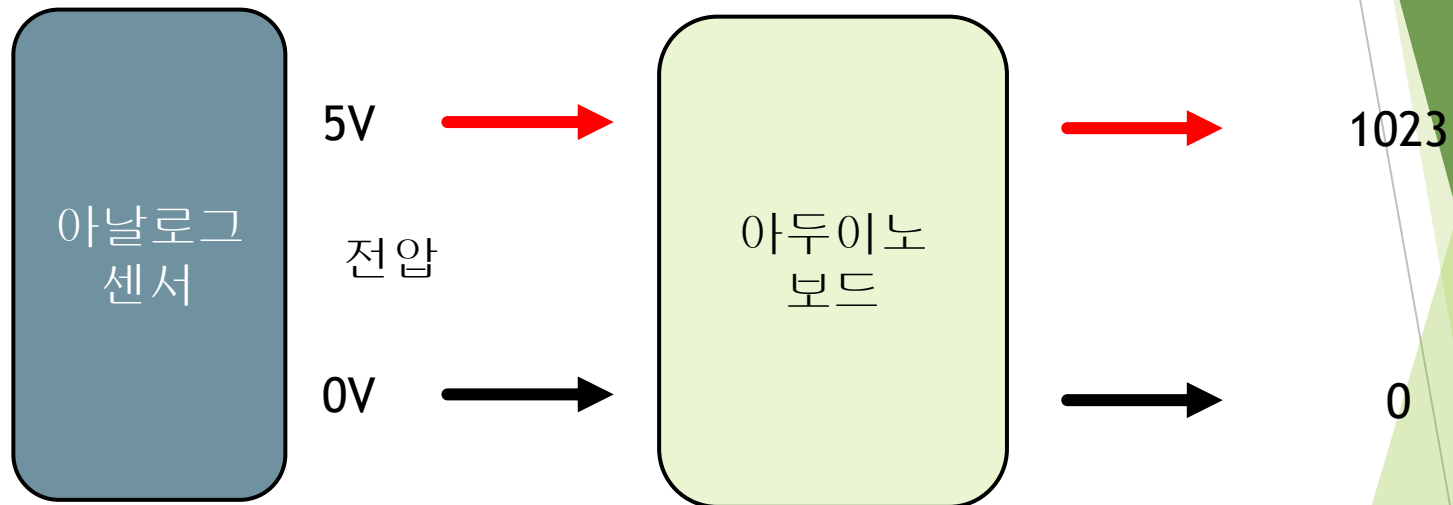
0V  LOW 라는 단어를 사용합니다.

예) `DigitalWrite(13, LOW)`

아두이노에서의 센서값

아날로그 센서

▶ 아날로그 센서 값

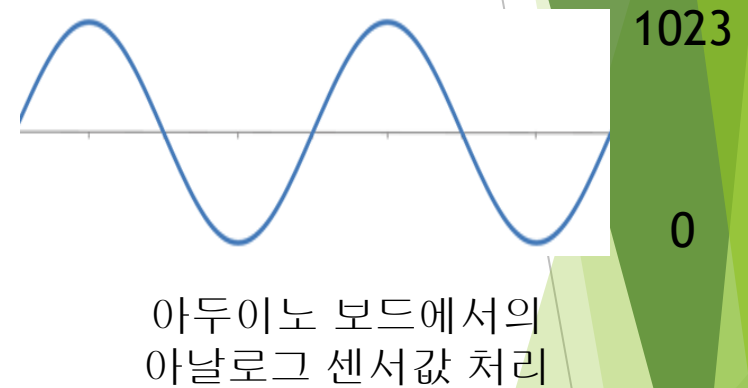
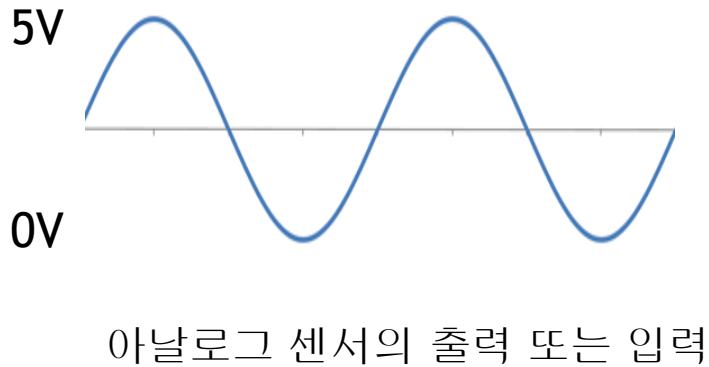


아날로그 센서는 0V ~ 5V 사이의 전압값이 출력되며, 아두이노에서는 이 전압값이 0 ~ 1023 사이의 숫자로 변환됩니다.

아두이노에서의 센서값

아날로그 센서

▶ 아날로그 센서 값

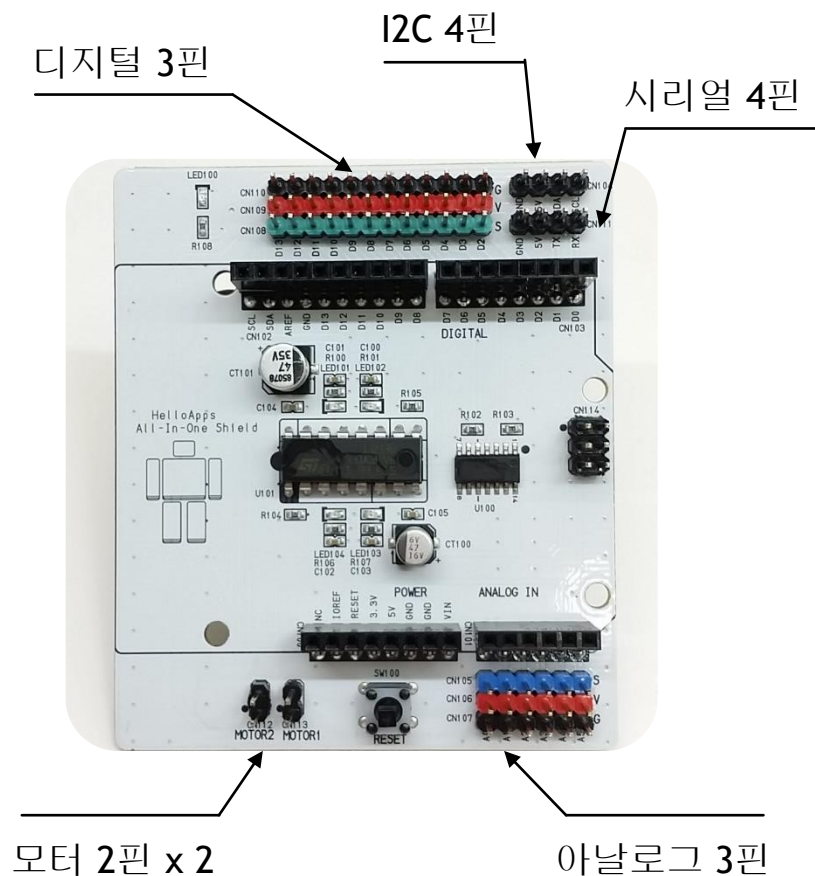


디지털 센서의 연결

아두이노 올인원 센서 쉴드

디지털 센서

▶ 초보자용 아두이노 올인원 센서 쉴드



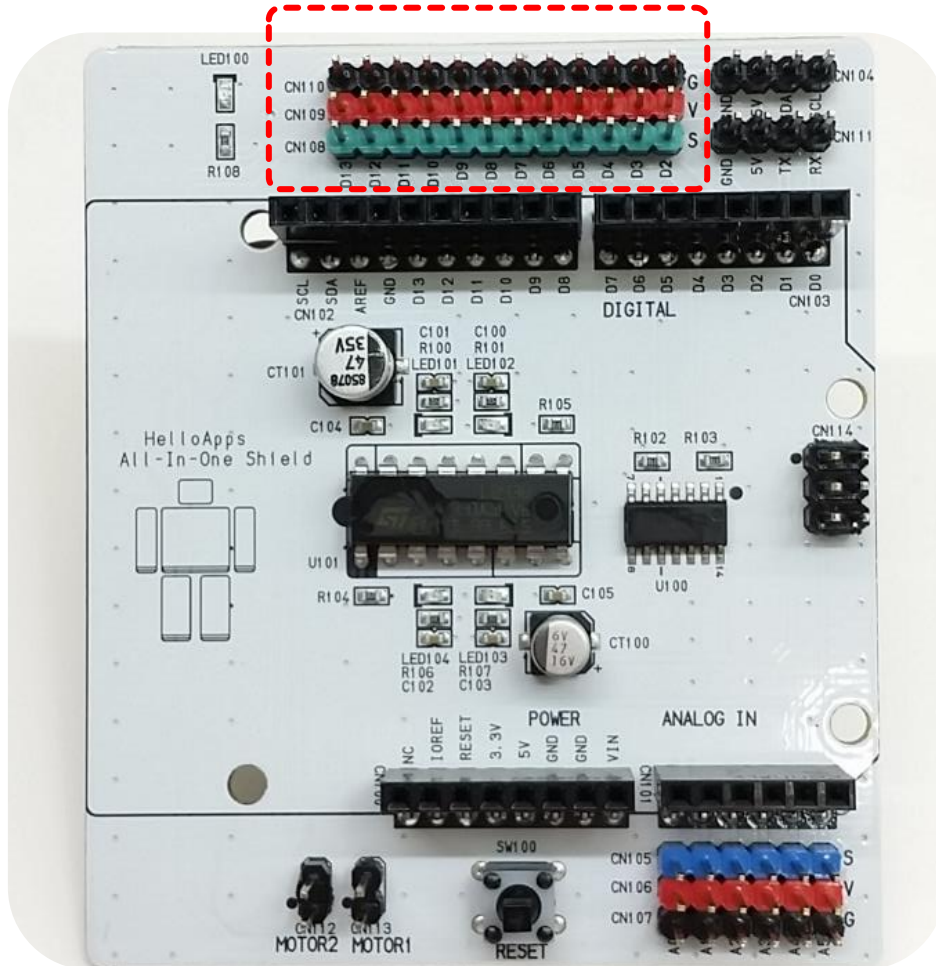
브레드 보드가 필요없는
초보자용 아두이노 보드

디지털 센서 연결하기

디지털 센서

▶ 디지털 핀

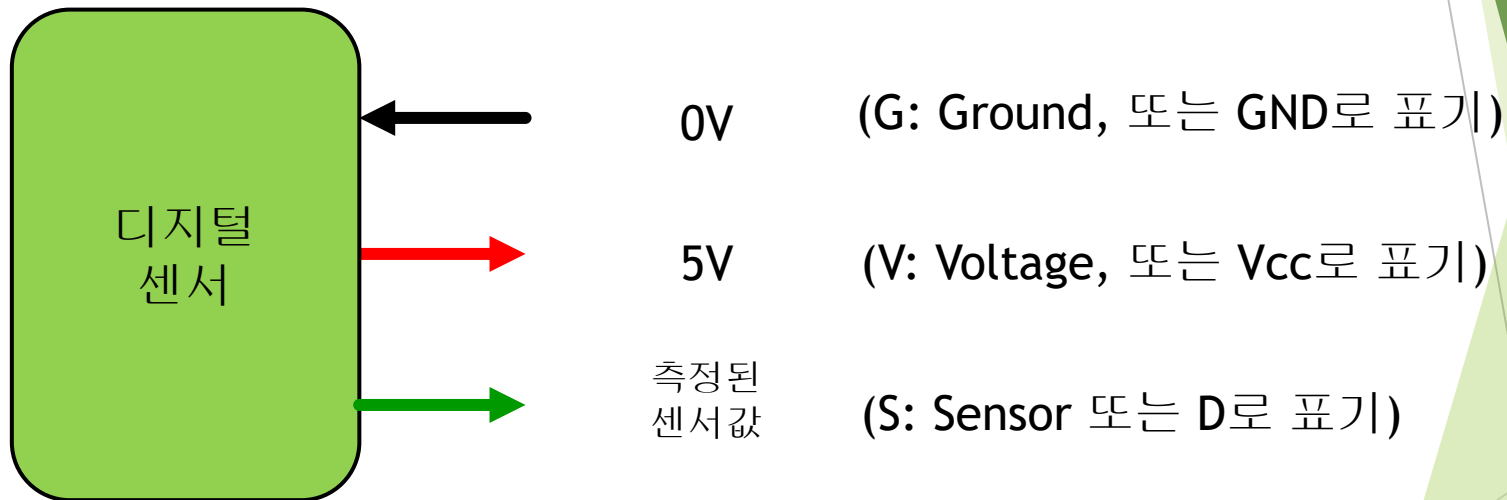
디지털 부품은 디지털 핀에 연결합니다.



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

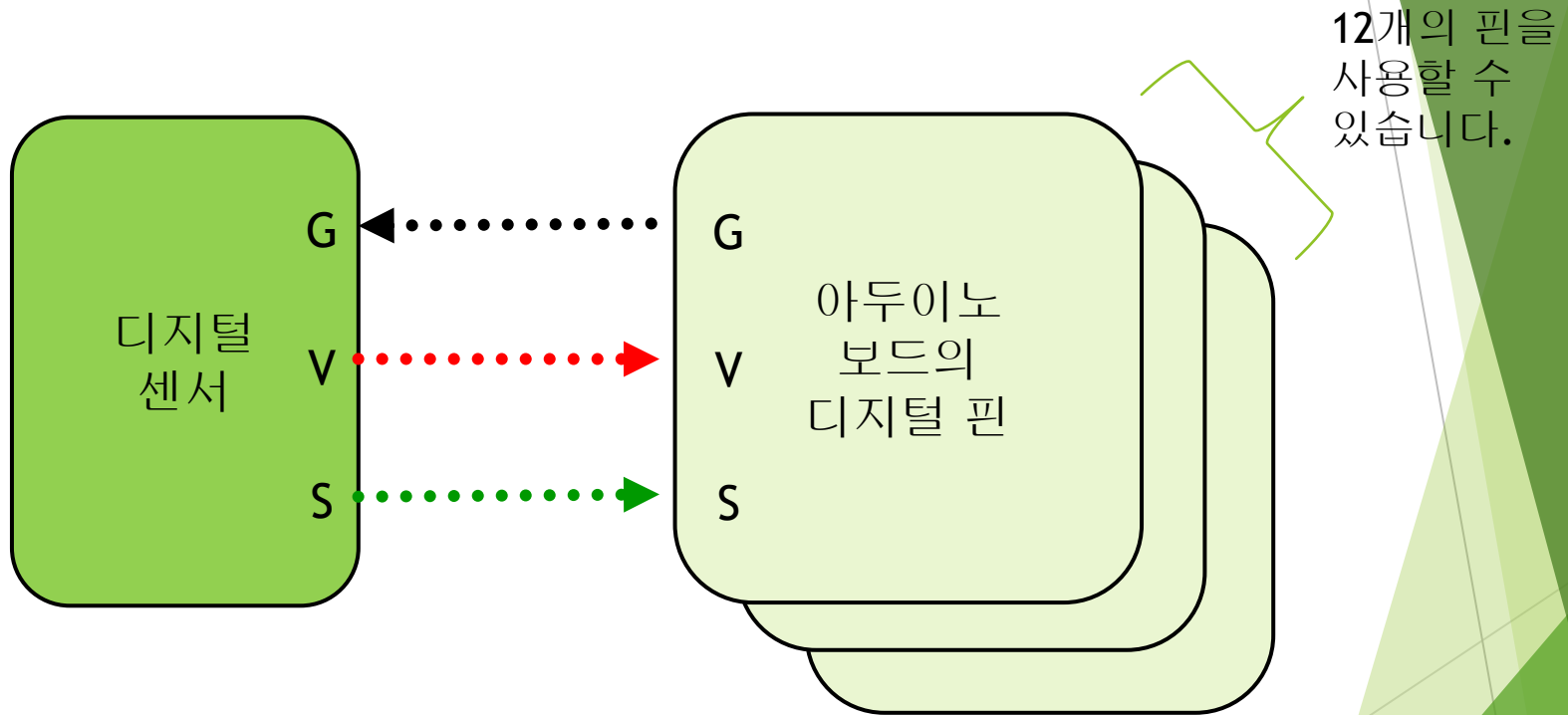
디지털 핀에 부품을 연결할
때에는 핀 번호를 확인해야
합니다.

▶ 디지털 3핀 센서 핀의 구조



디지털 센서 핀

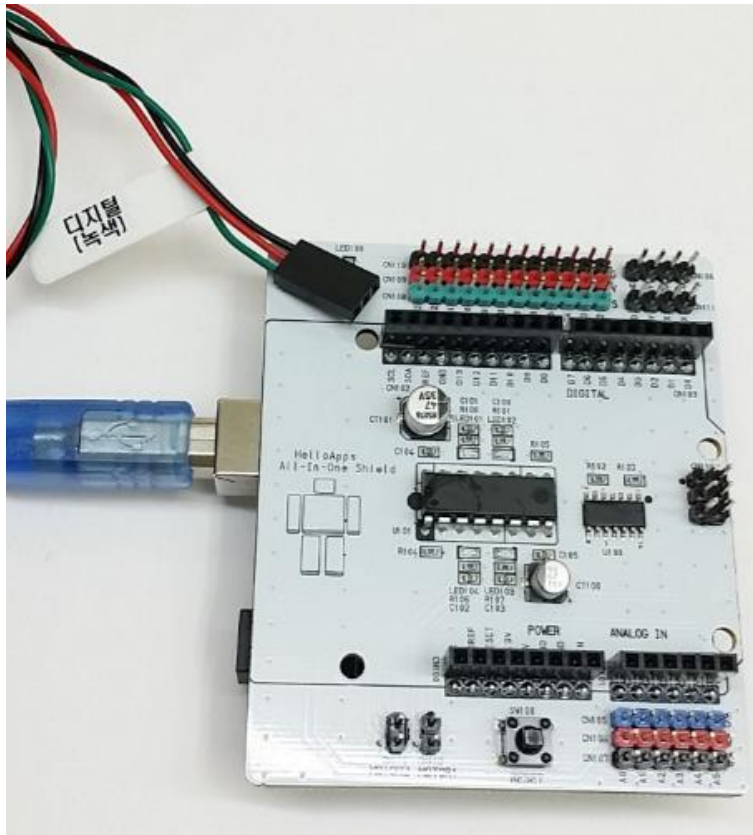
- ▶ 디지털 3핀 센서를 아두이노 보드에 연결하는 방법



아두이노 보드에는 총 14개의 디지털 핀(0번 핀 ~ 13번 핀)이 있으나 이중에서 0번과 1번은 통신용으로 사용되고 실제로는 2번 핀 부터 13번 핀까지 총 12개의 핀을 사용할 수 있습니다.

디지털 센서 연결하기

▶ 디지털 센서 케이블



디지털 센서는
케이블이 녹색선으로
표시되어 있습니다.

- GND
- VCC 또는 5V
- 디지털 데이터 선

LED 소자 연결 실습

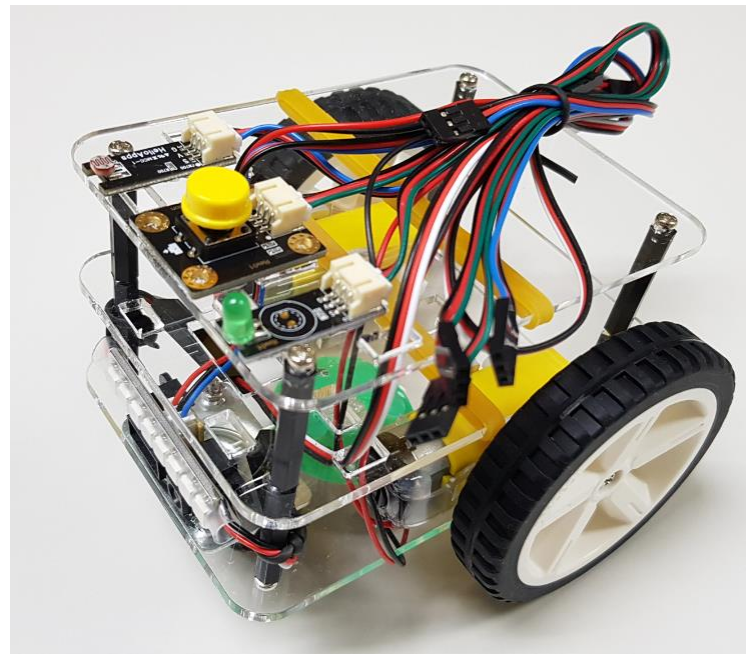
LED 소자 연결 실습

실습

- ▶ 실습) 아래의 LED 소자를 디지털 13번 핀에 연결해 봅니다.



LED소자

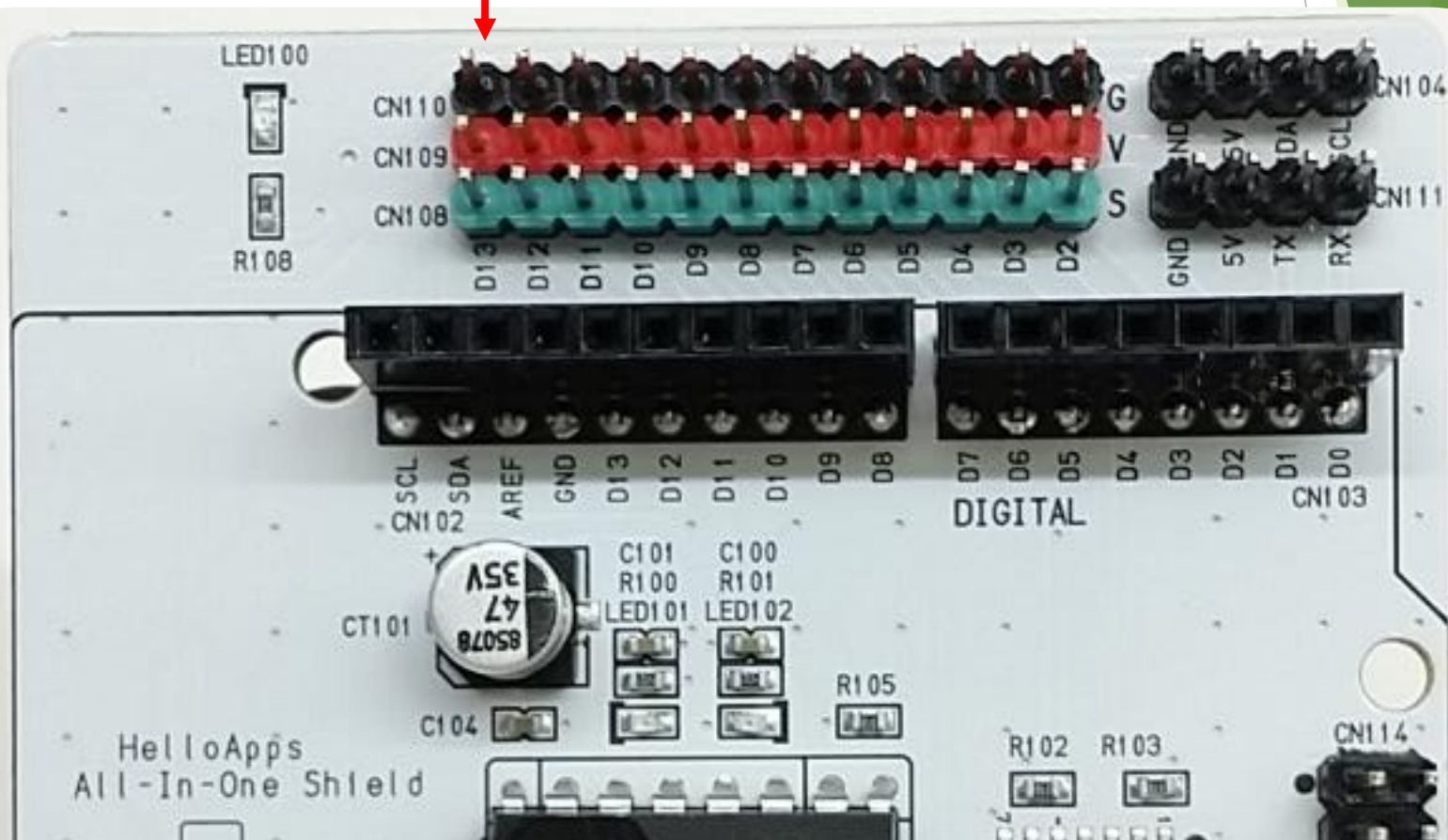


LED 소자 연결 실습

실습

- ▶ 실습) 디지털 13번 핀을 찾아 보세요.

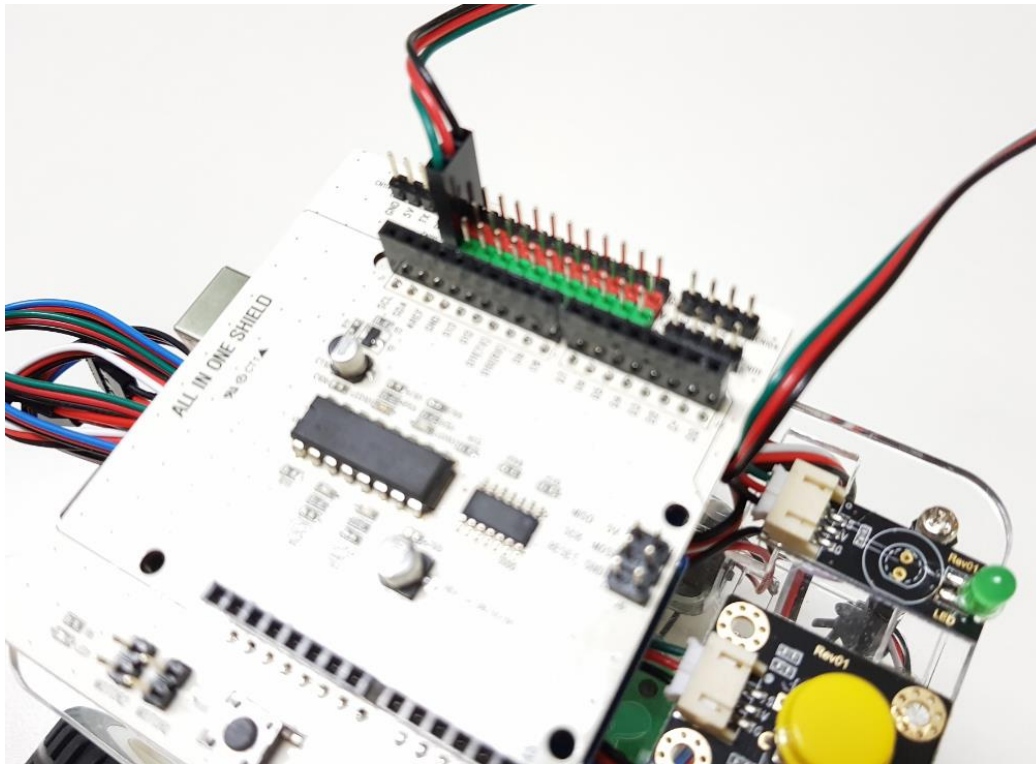
13번
(D13)



LED 소자 연결 실습

실습

▶ 디지털 부품의 연결



디지털 부품을 연결할 때에는 선의 색상 (검정, 빨강, 초록) 순서 및 핀 위치를 확인하여 연결합니다.

왼쪽은 LED소자를 디지털 13번 핀에 연결한 예입니다.

디지털 명령어

- ▶ 디지털 핀에 값을 쓸 때 사용하는 명령어

DigitalWrite

- ▶ 디지털 핀에서 값을 읽어 올 때 사용하는 명령어

DigitalRead

- ▶ 디지털 핀에 값을 쓸 때에는 핀번호와 값이 필요함

DigitalWrite (핀번호, 값)



- ▶ 디지털 핀에서 값을 읽어 올 때에는 핀번호와 값을 저장할 변수가 필요함

a = DigitalRead (핀번호)



- ▶ Delay 명령어는 주어진 시간 (밀리초) 동안 실행을 멈추는 명령어입니다.

Delay (밀리초)



Delay 1000 (ms)

1000 밀리초는 1초를 의미합니다.

100밀리초는 몇초일까요?

3500 밀리초는 몇초일까요?

LED 소자 제어 실습

실습) 13번 핀의 LED 점멸시키기

실습

- ▶ 디지털 13번 핀에 연결된 LED를 1초 간격으로 점멸시키는 프로그램을 작성해 보세요.

실습) 13번 핀의 LED 점멸시키기

실습

- ▶ 블록으로 작성한 프로그램 예)

The image shows a block-based programming interface with two main function blocks: 'Setup' and 'Loop'.

- Function Setup:** A yellow block labeled 'Function' with the name 'Setup'. It contains a grey instruction box with Korean text: '왼쪽에 있는 명령어 아이콘을 이곳에 마우스로 드래그하여 넣어 주세요.' (Please drag the command icons from the left into this area with the mouse).
- Function Loop:** A yellow block labeled 'Function' with the name 'Loop'. It contains four stacked blocks:
 - Block 1:** A blue block with a checkmark icon, labeled 'DigitalWrite'. It has a text input field containing '13' with '(Pin)' next to it, and another text input field containing 'HIGH' with '(Value)' next to it. There is a close button (X) on the right.
 - Block 2:** A red block with a checkmark icon, labeled 'Delay'. It has a text input field containing '1000' with '(ms)' next to it. There is a close button (X) on the right.
 - Block 3:** A blue block with a checkmark icon, labeled 'DigitalWrite'. It has a text input field containing '13' with '(Pin)' next to it, and another text input field containing 'LOW' with '(Value)' next to it. There is a close button (X) on the right.
 - Block 4:** A red block with a checkmark icon, labeled 'Delay'. It has a text input field containing '1000' with '(ms)' next to it. There is a close button (X) on the right.

실습) 13번 핀의 LED 점멸시키기

실습

- ▶ 스크립트로 작성한 프로그램 예)

```
void setup()
{
}

void loop()
{
    digitalWrite(13, HIGH)
    delay(1000)
    digitalWrite(13, LOW)
    delay(1000)
}
```

실습) 13번 핀의 LED 점멸시키기

실습

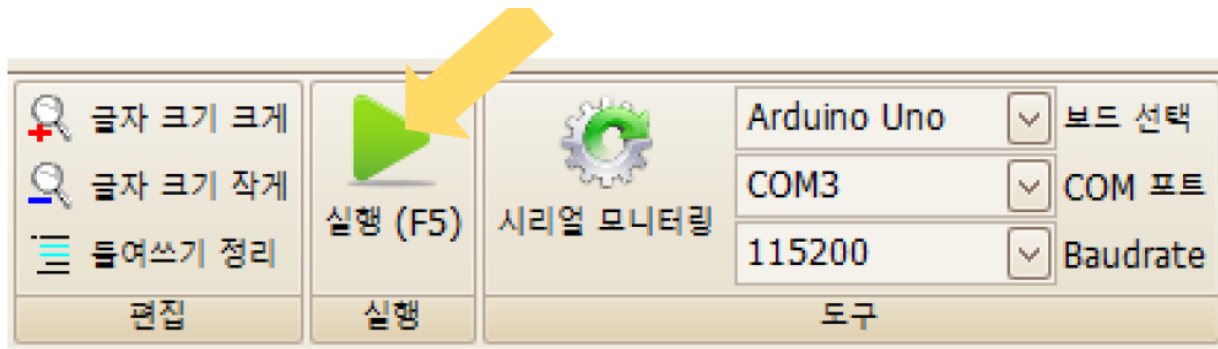
- ▶ C언어 문법(스케치 코드)으로 작성한 프로그램 예

```
void setup()
{
    pinMode(13, OUTPUT);
}
void loop()
{
    digitalWrite(13, HIGH);
    delay(1000);
    digitalWrite(13, LOW);
    delay(1000);
}
```

프로그램 실행하기

실습

- ▶ 상단 메뉴의 가운데 있는 실행 버튼을 클릭하여 프로그램을 아두이노에 업로드 시킵니다.



실행 버튼을 클릭하여 프로그램을 아두이노 보드에 업로드 시킨다

응용 실습

- ▶ 아래의 실습 주제를 수행해 봅니다.

실습) LED의 점멸 주기를 더 짧게 수정하기

LED의 점멸 주기를 더 짧게 수정한 후 업로드해 보기 바란다.

실습) LED의 점멸 주기로 활용될 수 있는 작품은?

회전하는 선풍기 날개에 쓰여진 글자가 정지 영상으로 보이게 하기 위해서는 LED의 점멸이 어떻게 활용되어야 하는지 생각해 보자.

LED 제어 응용 실습

블록 코드

- ▶ LED의 점멸주기를 수정하려면 아래 코드에서 어느 부분을 수정해 주어야 할까요?

The image shows a block-based programming interface with two function blocks: 'Setup' and 'Loop'. The 'Setup' block is currently empty. The 'Loop' block contains a sequence of four blocks: a 'DigitalWrite' block (Pin 13, HIGH), a 'Delay' block (1000 ms), another 'DigitalWrite' block (Pin 13, LOW), and a final 'Delay' block (1000 ms). Each block has a checkmark on the left and a close button on the right. A grey instruction box above the 'Setup' block says '왼쪽에 있는 명령어 아이콘 이곳에 마우스로 드래그하여 넣어 주세요.' (Drag the command icons from the left into this area with the mouse).

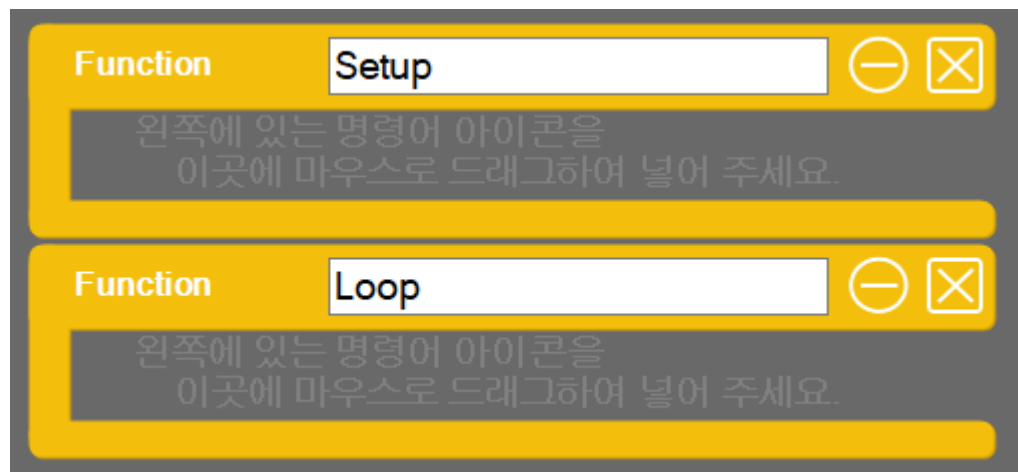
- ▶ LED의 점멸주기를 수정하려면 아래 코드에서 어느 부분을 수정해 주어야 할까요?

```
void setup()
{
    pinMode(13, OUTPUT);
}
void loop()
{
    digitalWrite(13, HIGH);
    delay(1000);
    digitalWrite(13, LOW);
    delay(1000);
}
```

아두이노 프로그램의 구조

아두이노 프로그램의 구조

- ▶ 아두이노 프로그램은 기본적으로 아래의 구조를 가집니다.



블록 코드

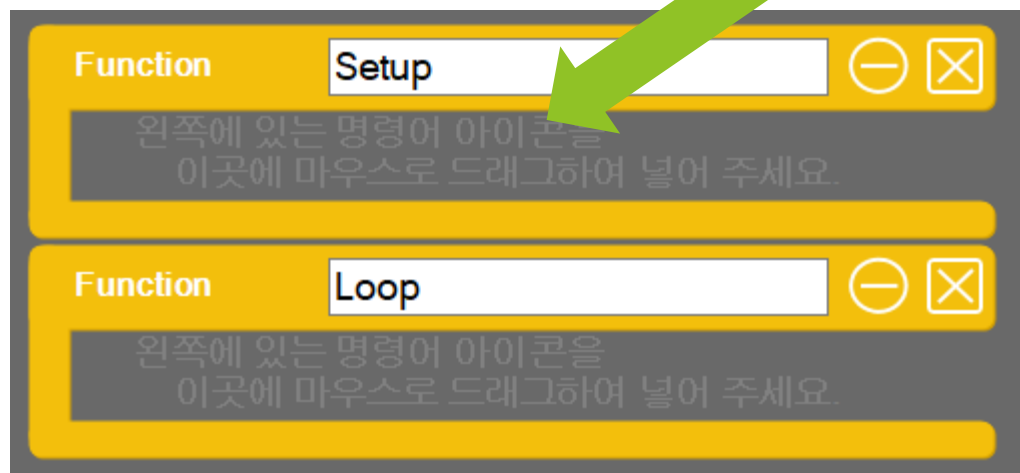
```
void setup()
{
}
void loop()
{
}
```

스케치 코드

아두이노 프로그램의 구조

프로그램 구조

- ▶ Setup 함수는 가장 먼저 실행되며 한번만 실행됩니다.



블록 코드

```
void setup()  
{  
}  
void loop()  
{  
}  
}
```

A green arrow points from the right towards the 'void setup()' code block.

스케치 코드

Setup 함수

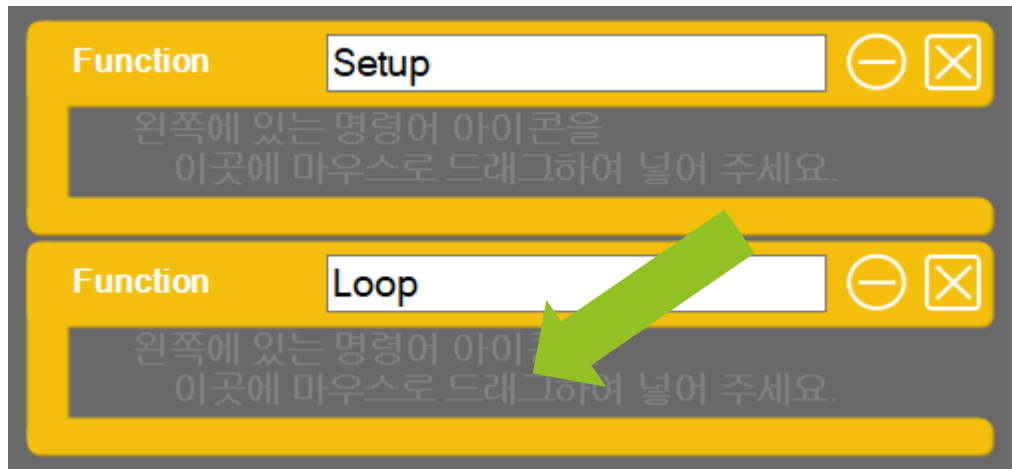
처음 SPL 편집기를 실행하면 기본적으로 Setup 이름이 포함된 블록과 Loop 이름이 포함된 블록을 볼 수 있다. Setup 함수는 아두이노 프로그램 실행시 가장 먼저 실행되어야 하는 명령어들이 포함되는 프로그램 그룹이다. 보통 전역 변수를 초기화 하거나 LCD의 백라이트를 켜는 등의 작업을 할 때 사용된다.

블록 방식으로 코딩을 한 후, 파일을 저장하면 우측에 스크립트와 C언어 형태의 스케치 코드를 볼 수 있다. 이 때 변환된 스케치 코드를 보면 Setup 함수에 자동으로 추가되는 명령어들을 볼 수 있는데, 이러한 명령어들은 원래 사용자가 항상 추가해주어야 하는 명령어이지만, 초보자를 위해 SPL 편집기가 자동으로 추가를 하고 있다.

아두이노 프로그램의 구조


프로그램 구조

- ▶ **Loop** 함수는 **Setup** 함수 실행이 끝난 후 실행되며, 아두이노 보드에 전원이 공급되는 한 무한히 반복하여 실행됩니다.



블록 코드

```
void setup()  
{  
  
}  
void loop()  
{  
  
}
```



스케치 코드

Loop 함수

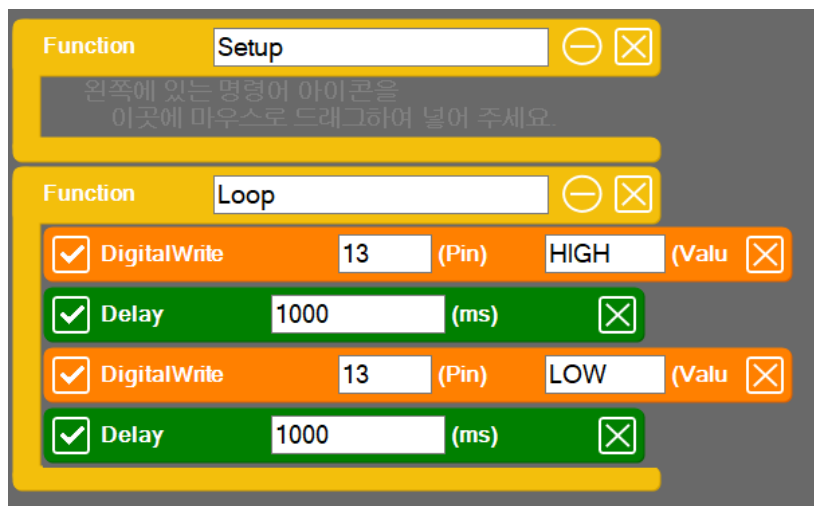
Loop 함수는 Setup 함수 명령이 끝나고 난 후, 이어서 무한히 반복되는 명령어 그룹입니다. 아두이노는 보드에 전원이 들어오는 동안 내내 Loop 함수 안에 포함되어 있는 프로그램을 계속 반복하여 실행시킨다.

응용 실습

아두이노 프로그램의 구조

블록 코드

- ▶ 디지털 13번 핀에 연결된 LED 소자를 0.5초간 3번 점멸 한 후, 항상 꺼져 있도록 기존 프로그램을 수정해 봅시다.



아두이노 프로그램의 구조

스크립트 코드

- ▶ 디지털 13번 핀에 연결된 LED 소자를 0.5초간 3번 점멸 한 후, 항상 꺼져 있도록 기존 프로그램을 수정해 봅시다.

```
void setup()  
{  
}
```

```
void loop()  
{
```

```
  DigitalWrite(13, HIGH)  
  Delay(1000)  
  DigitalWrite(13, LOW)  
  Delay(1000)
```

```
}
```



?