

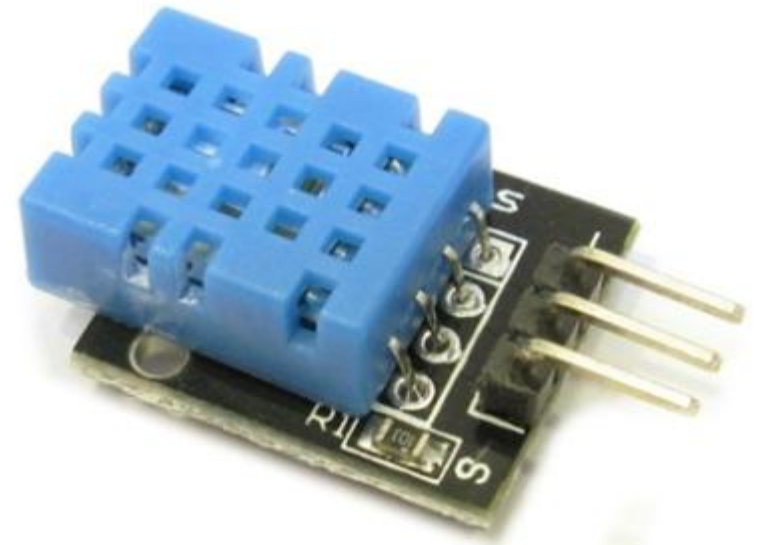
센서 라이브러리 활용
+ 퀴즈(10%성적반영)

목표

- 인터럽트를 이용한 화재 감지 실험
- 라이브러리를 이용한 센서 데이터 확인 및 활용

DHT11

- 동작 전압 (Power) 3~5 V
- 온도 측정 범위 (Temperature range) 0 ~ 50 °C (± 2 °C)
- 습도 측정 범위 (Humidity range) 20 ~ 80 % (± 5 %)
- 최대소비전력 (Max. current) 2.5 mA
- 데이터 주기 (sampling rate) 1 Hz



DHT11 라이브러리 사용

- <https://github.com/adafruit/DHT-sensor-library>

The screenshot shows the GitHub repository page for `adafruit/DHT-sensor-library`. The repository is public and has 162 watchers, 1.6k stars, and 1.2k forks. It contains 17 issues, 13 pull requests, and 142 commits. The latest commit is by `dherrada` bumping the version to 1.4.3, 29 days ago.

Files:

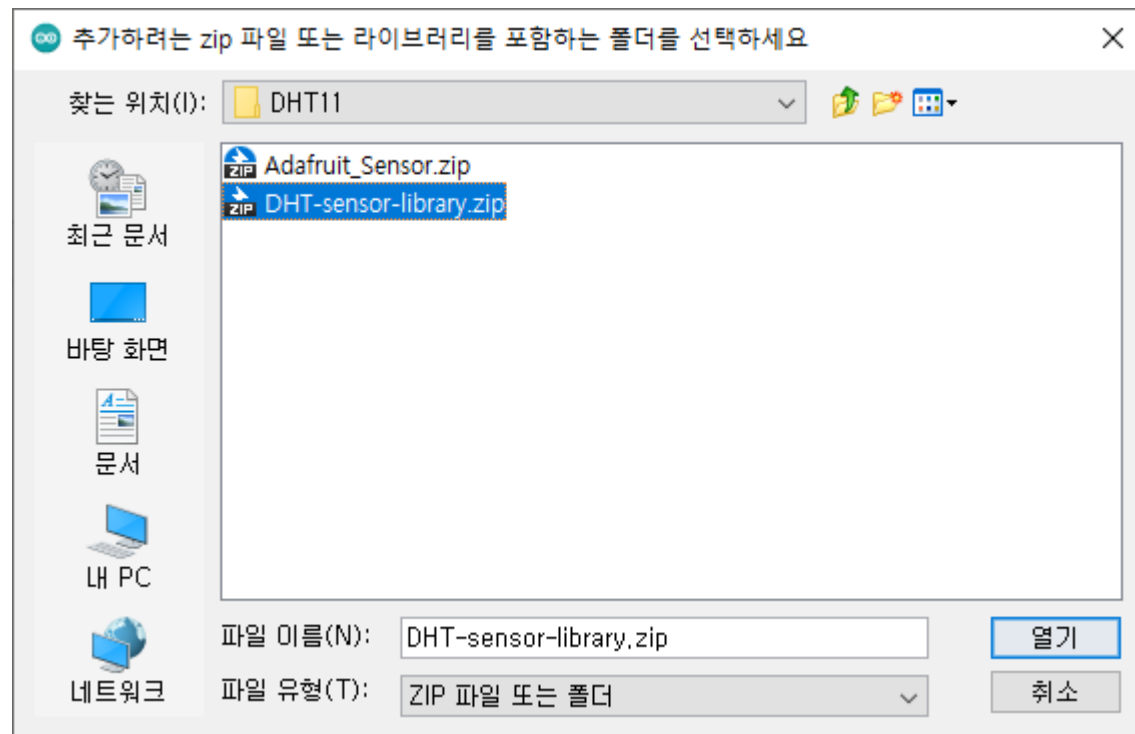
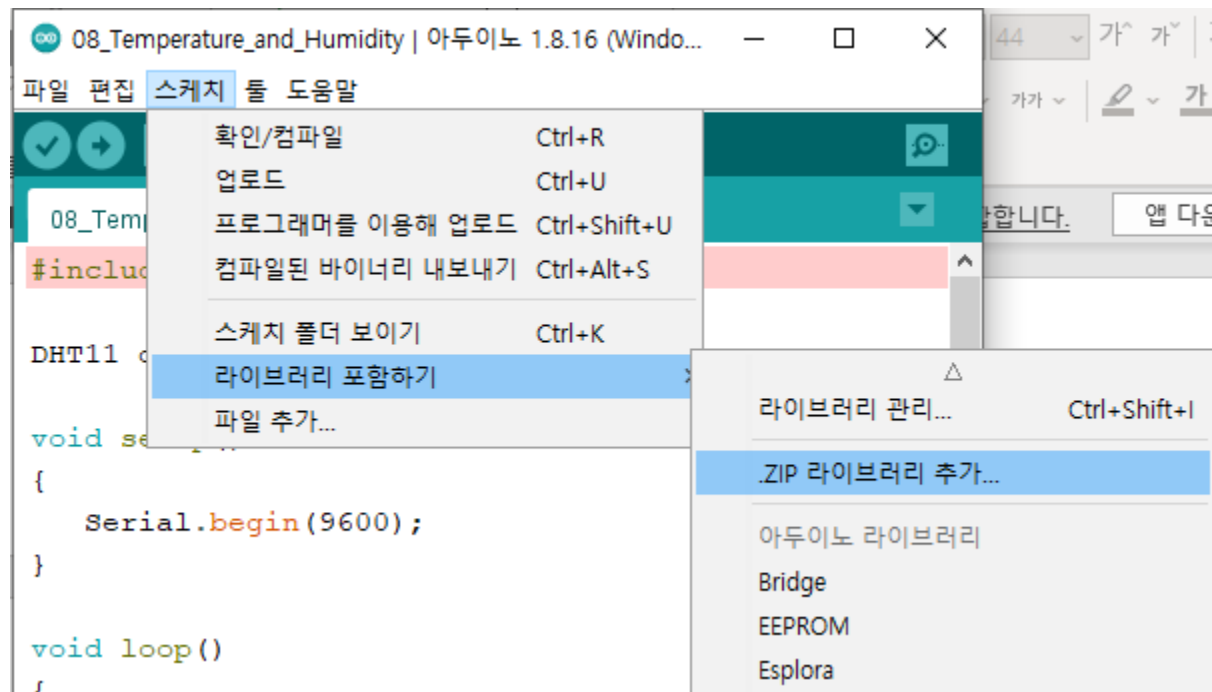
File	Description	Time
<code>.github</code>	actionified, formatted and doxy'd	2 years ago
<code>examples</code>	Updated comment on the Pin Out of the DHTxx Sensors	9 months ago
<code>.gitignore</code>	Add .gitignore	2 years ago
<code>CONTRIBUTING.md</code>	[Update URL]	2 years ago
<code>DHT.cpp</code>	Merge pull request #159 from Rotzbua/patch-1	2 years ago
<code>DHT.h</code>	Fix comment on DHT22 and DHT21 variables	29 days ago
<code>DHT_U.cpp</code>	actionified, formatted and doxy'd	2 years ago
<code>DHT_U.h</code>	actionified, formatted and doxy'd	2 years ago
<code>README.md</code>	actionified, formatted and doxy'd	2 years ago
<code>code-of-conduct.md</code>	actionified, formatted and doxy'd	2 years ago
<code>keywords.txt</code>	Use correct field separator in keywords.txt	3 years ago
<code>library.properties</code>	Bump to 1.4.3	29 days ago

About: Arduino library for DHT11, DHT22, etc Temperature & Humidity Sensors. Link: learn.adafruit.com/dht

Releases: 22 releases. Latest: 1.4.3 - Switched from defines to ... (29 days ago)

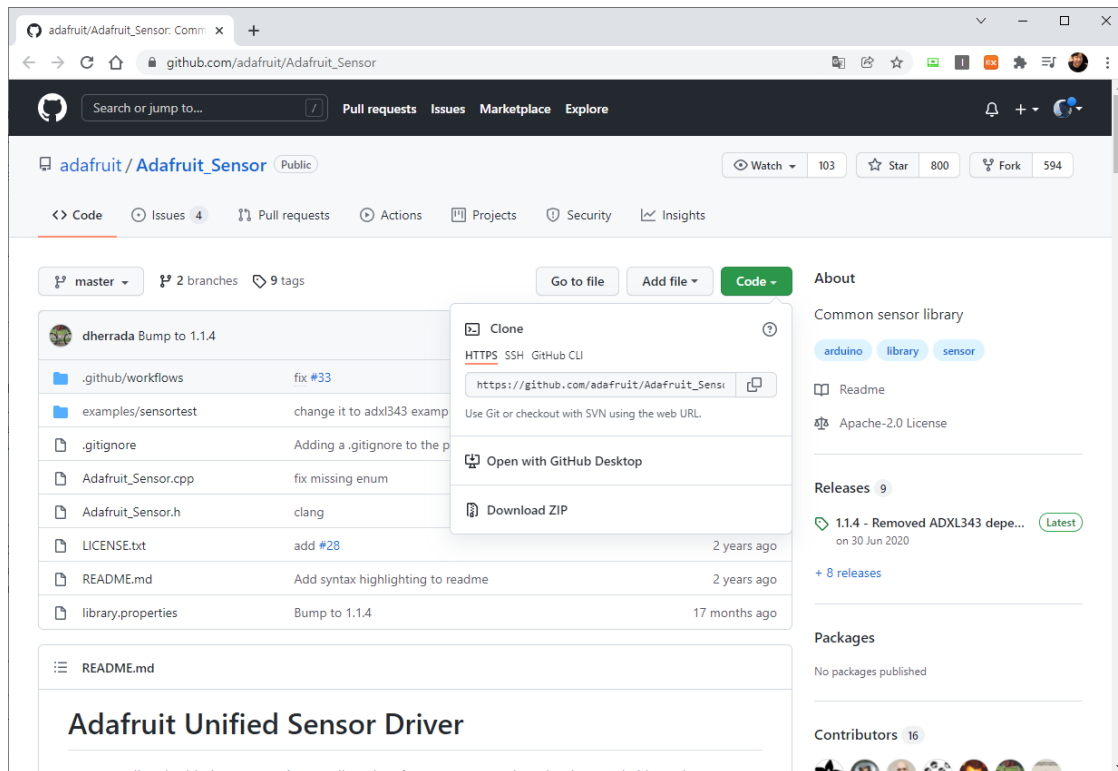
Packages: No packages published

DHT11 라이브러리 사용

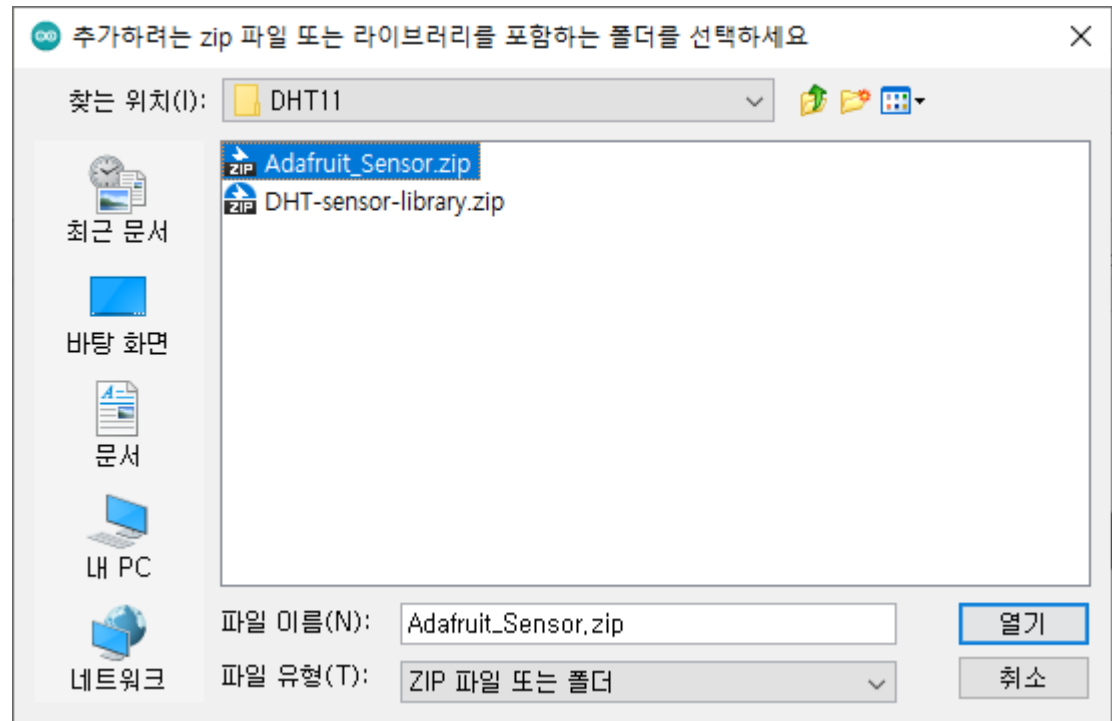
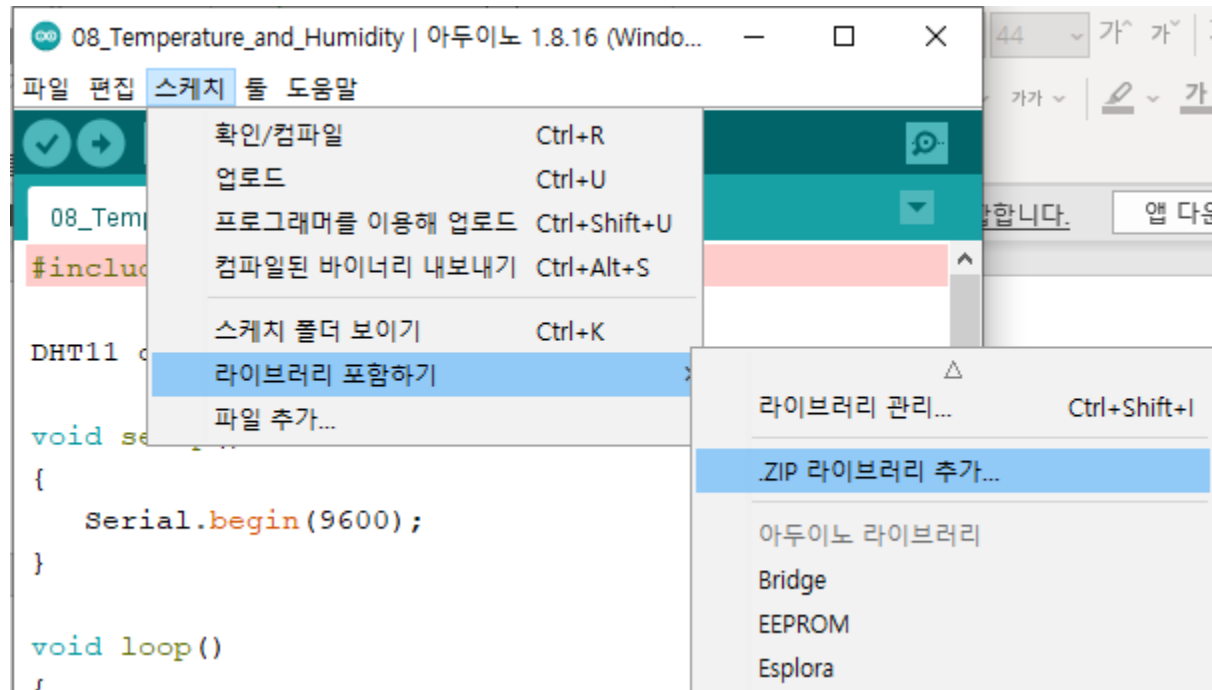


DHT11 라이브러리 사용

- adafruit_sensor.h no such file 에러 발생
- https://github.com/adafruit/Adafruit_Sensor 라이브러리 추가

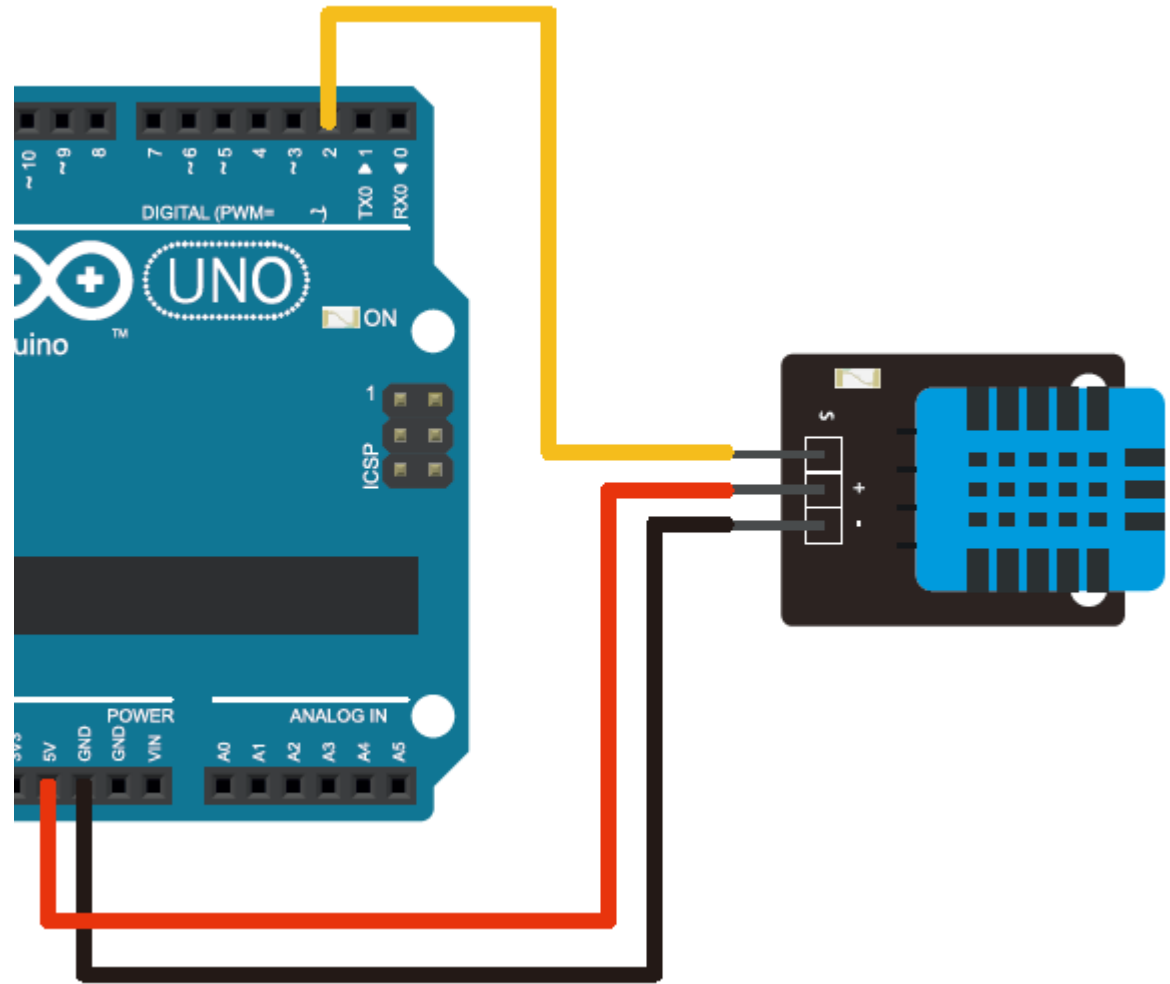


DHT11 라이브러리 사용

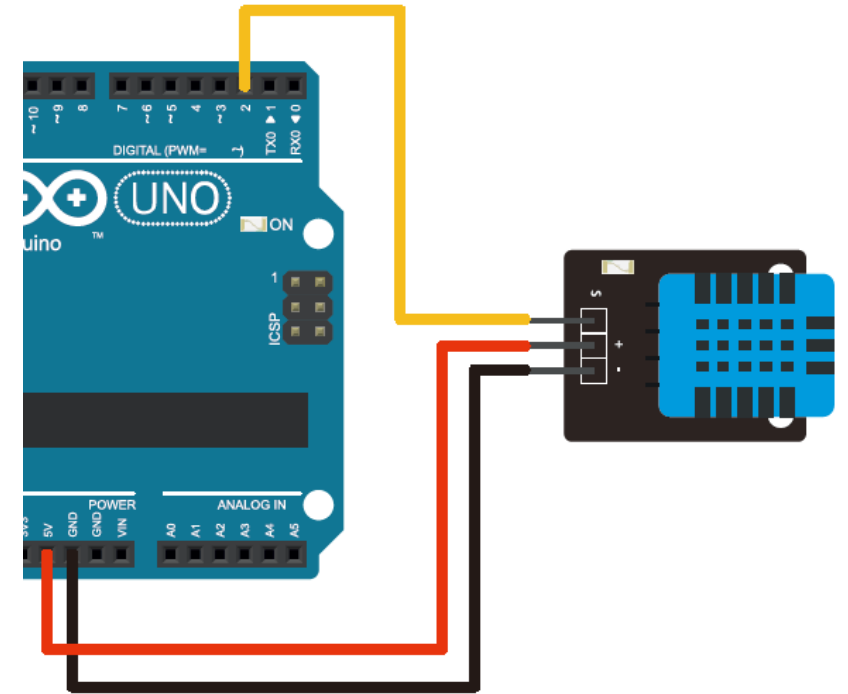
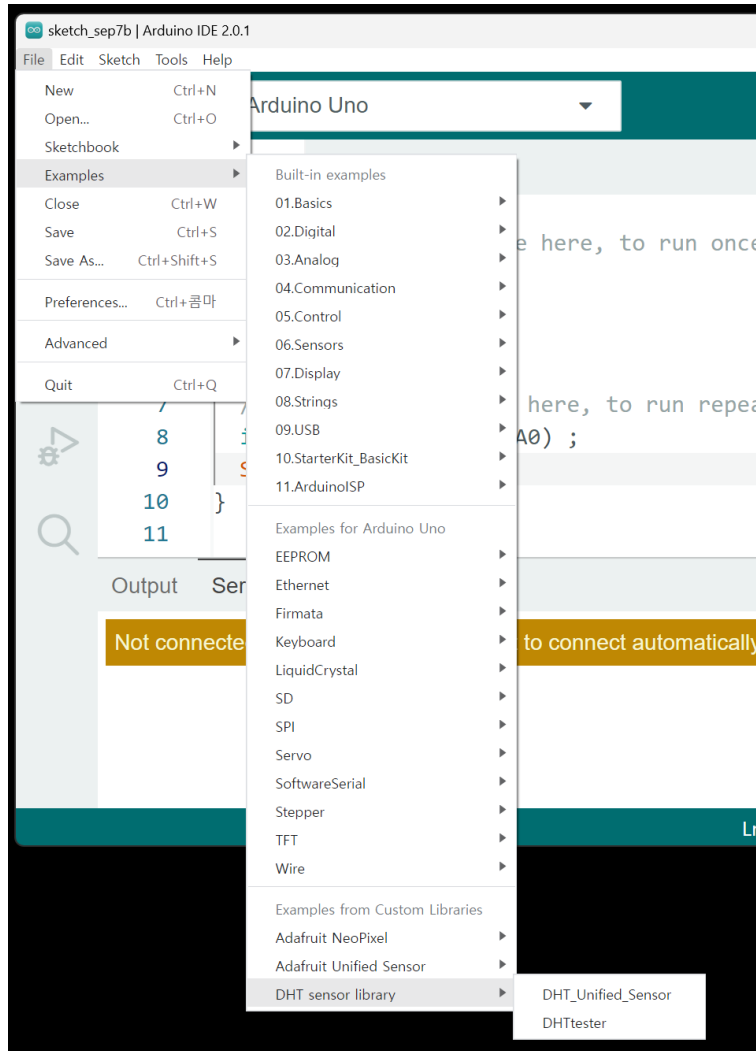


DHT11 아두이노 테스트

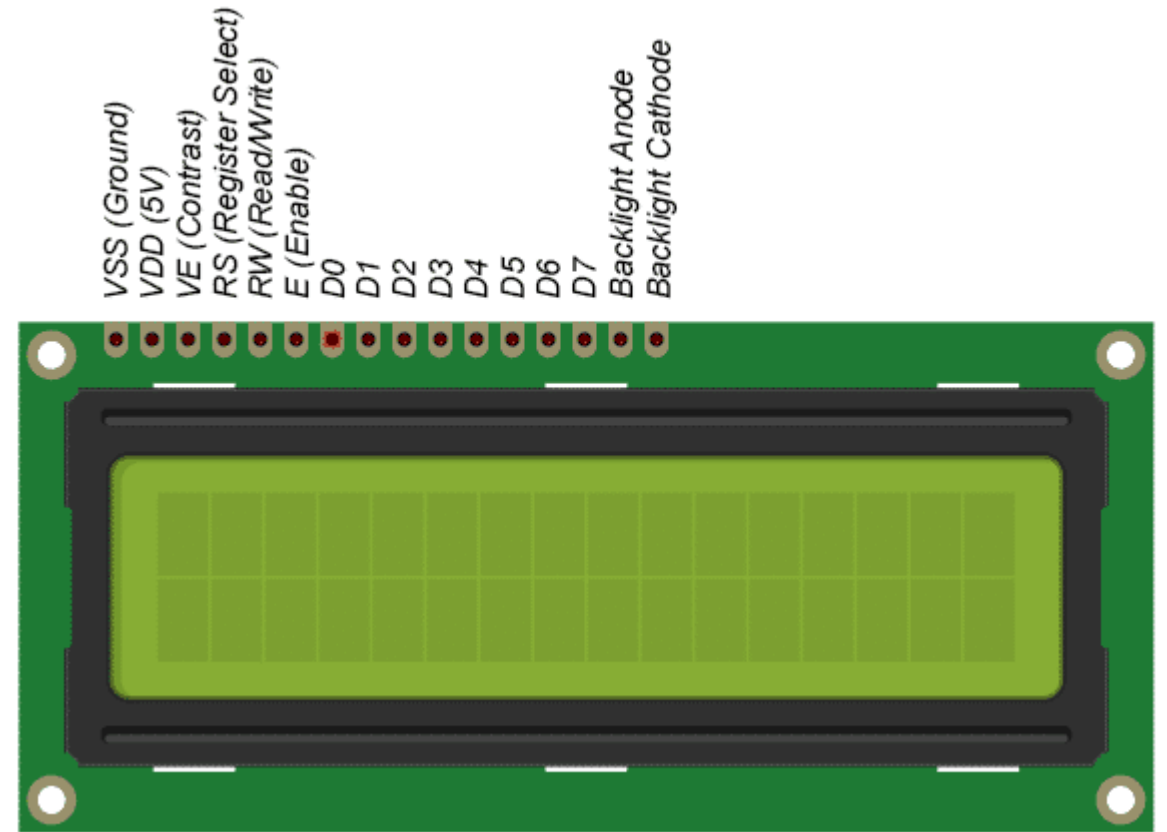
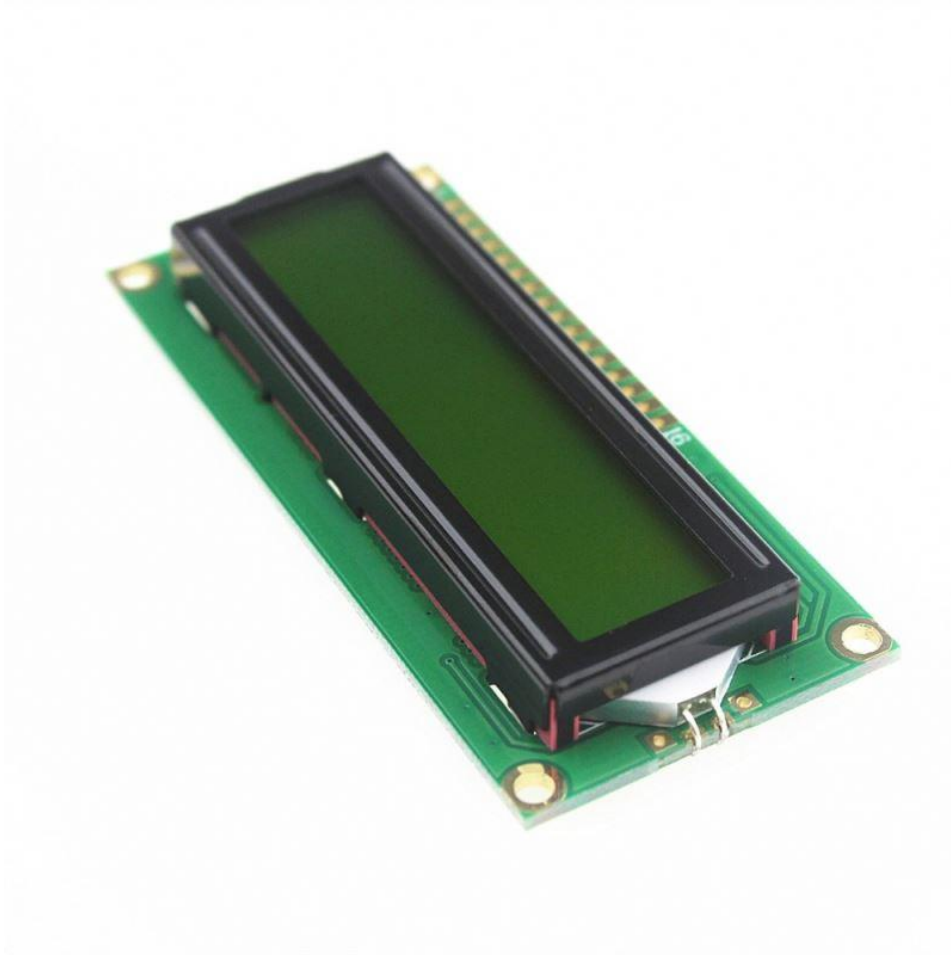
- S(signal) : 아두이노 2번핀
- + : VCC(5V)
- - : GND(0V)



DHT11 아두이노 테스트(Example 코드 활용)

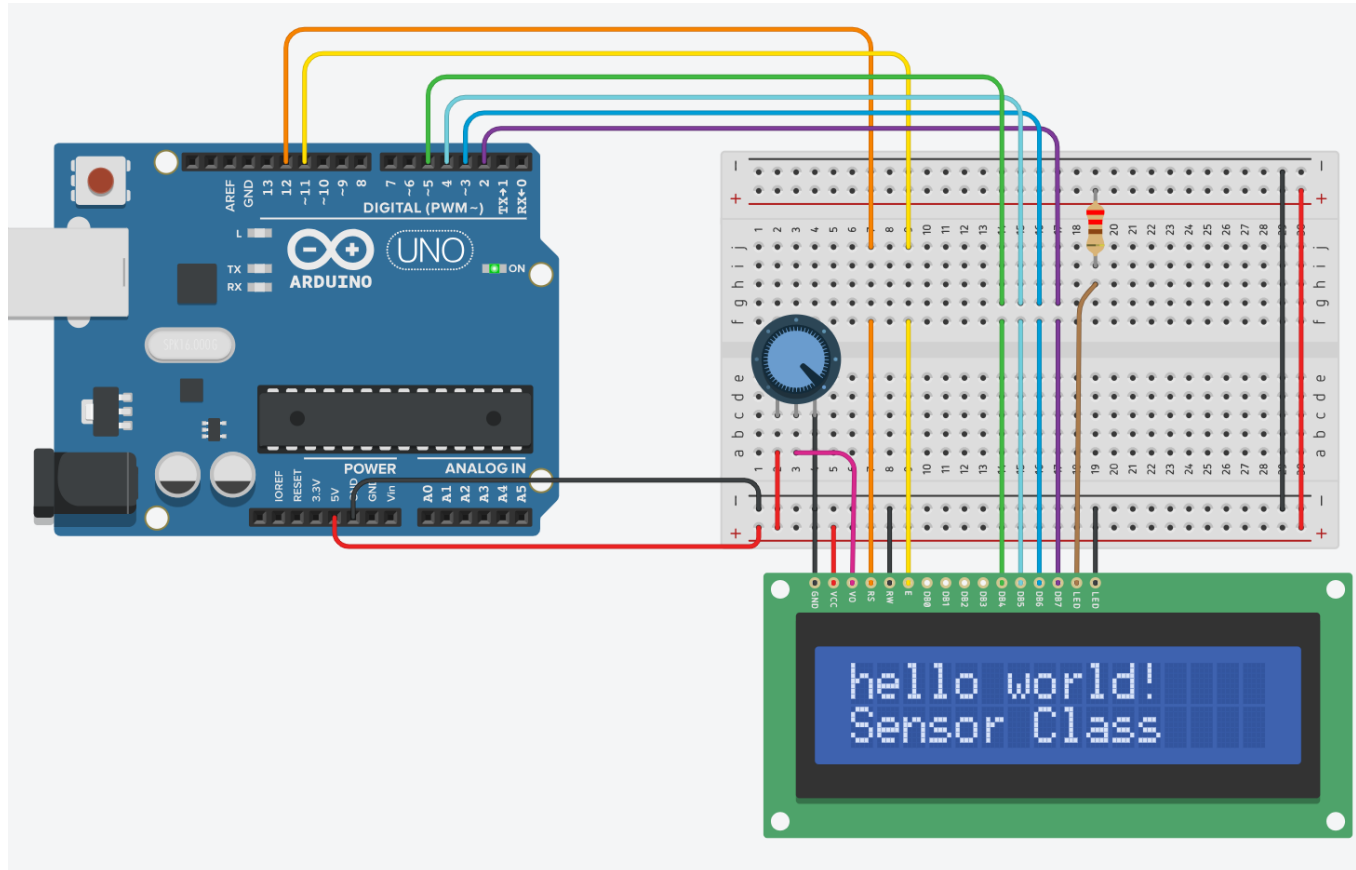


16x2 Character LCD



16x2 Character LCD 실험

- 아두이노 Example을 이용



```
#include <LiquidCrystal.h>

LiquidCrystal lcd_1(12, 11, 5, 4, 3, 2);

void setup()
{
  lcd_1.begin(16, 2);
}

void loop()
{
  lcd_1.setCursor(0, 0);
  lcd_1.print("hello world!");

  lcd_1.setCursor(0, 1);
  lcd_1.print("Sensor Class");
  delay(1000);
}
```

QUIZ

- LCD에 현재 온도와 습도를 표시하시오



불꽃감지센서 (Flame sensor)

- 불꽃 또는 화염은 사람의 눈으로 확인 할 수 없는 자외선과 적외선의 파장이 발생
- 불꽃감지센서는 적외선 감지센서로서 760nm ~ 1100nm파장을 감지한다.

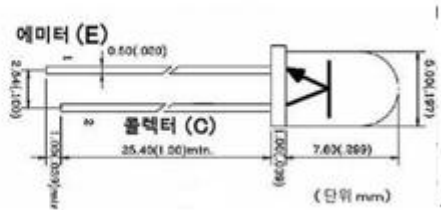
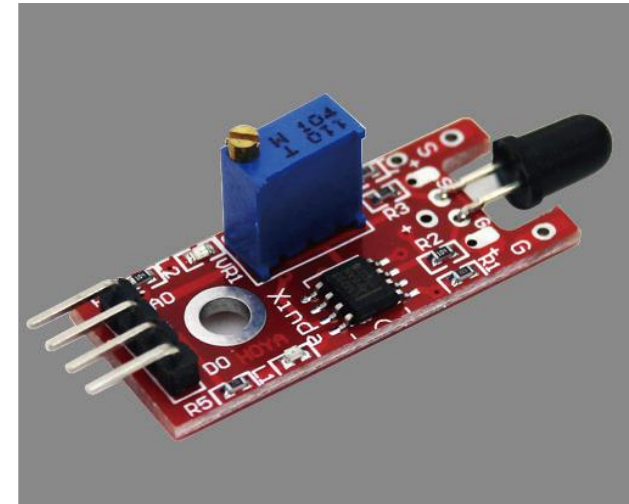
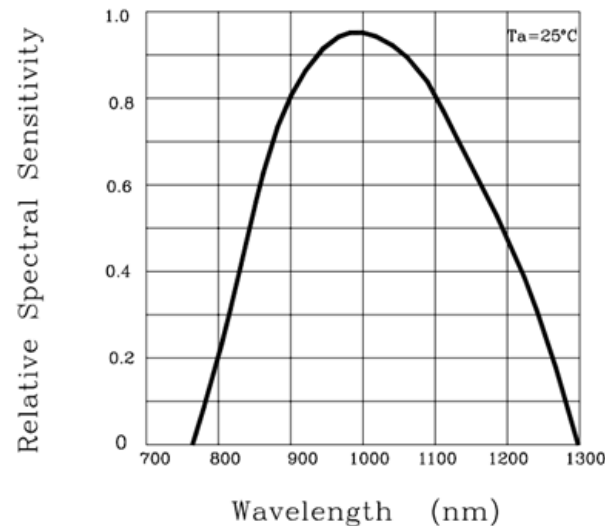
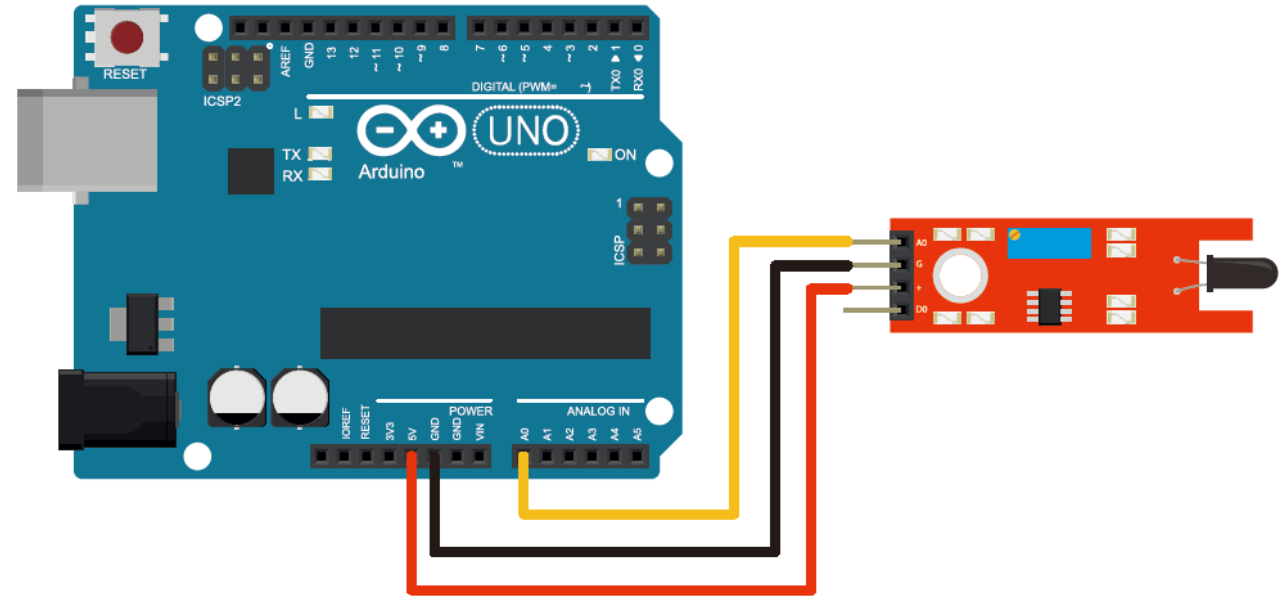


Fig. 5 Spectral Sensitivity



불꽃감지센서 (Flame sensor)

- 불꽃 감지 아두이노 실험 구성
 - 센서모듈 A0 <> 아두이노 A0
 - 센서모듈 G <> 아두이노 GND
 - 센서모듈 + <> 아두이노 5V



불꽃감지센서 (Flame sensor)

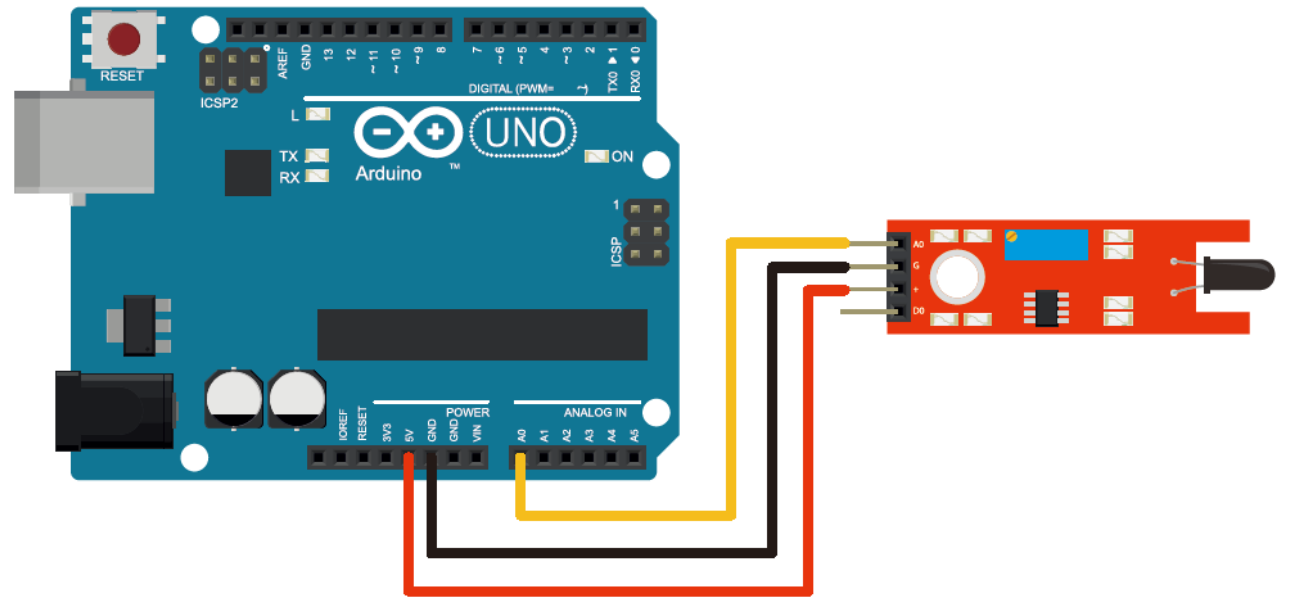
- 불꽃 감지 아두이노 실험 코드 작성

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

void loop()
{
  int analog_value = analogRead(A0);

  Serial.println(analog_value);

  delay(100);
}
```



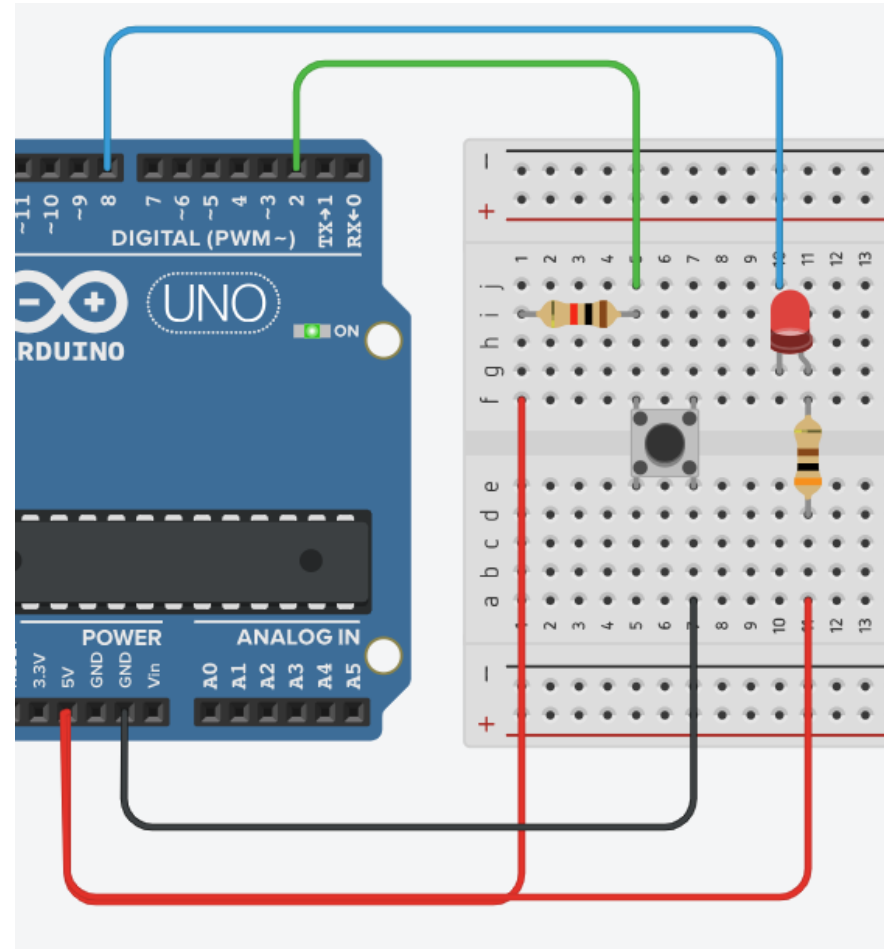
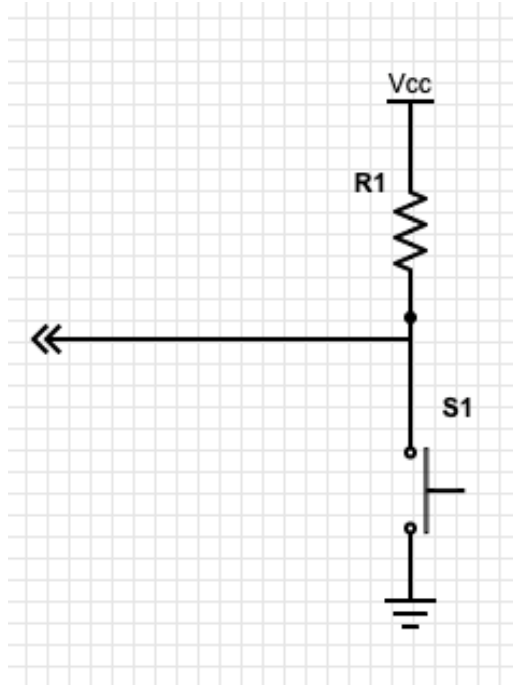
불꽃감지센서를 이용한 화재감지 응용

- 인터럽트를 이용하여 불꽃이 감지되면 소리와 빛을 내는 시스템을 제작하고 사진과 코드를 아래의 이메일 주소로 전송
- **juhong.park@farmily.ai**



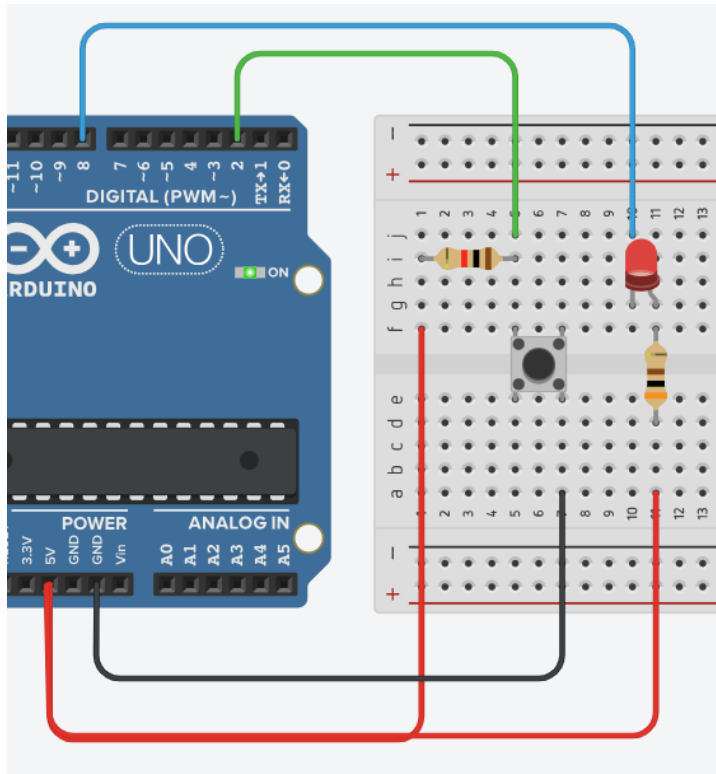
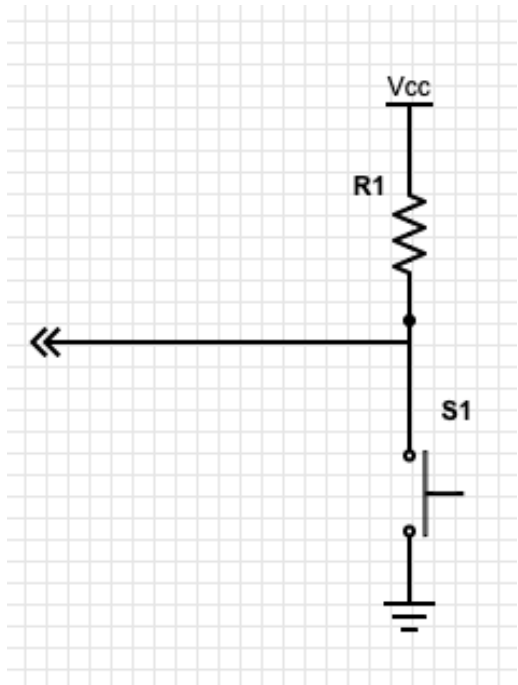
외부 인터럽트(External Interrupt)

- 펄싱 vs 인터럽트



외부 인터럽트(External Interrupt)

- 풀링 vs 인터럽트



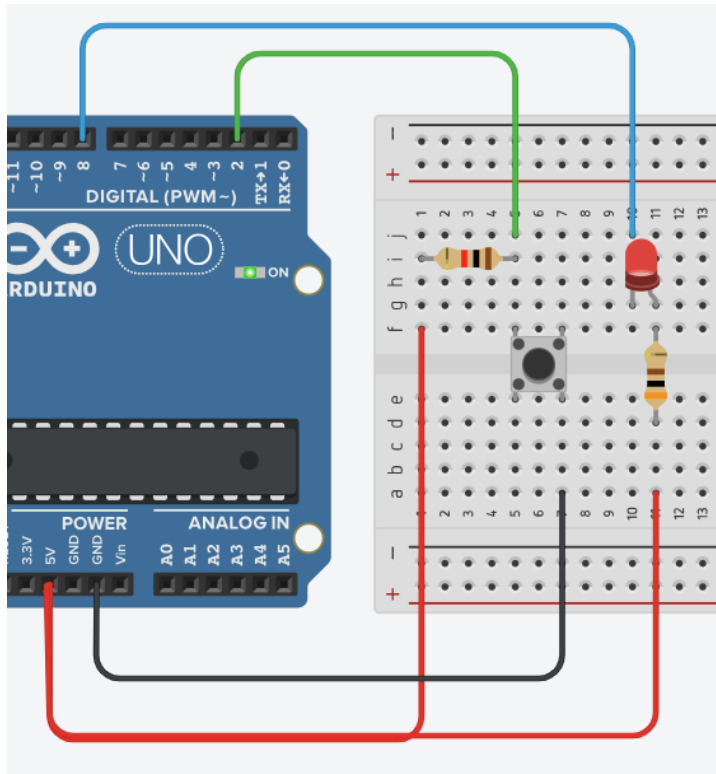
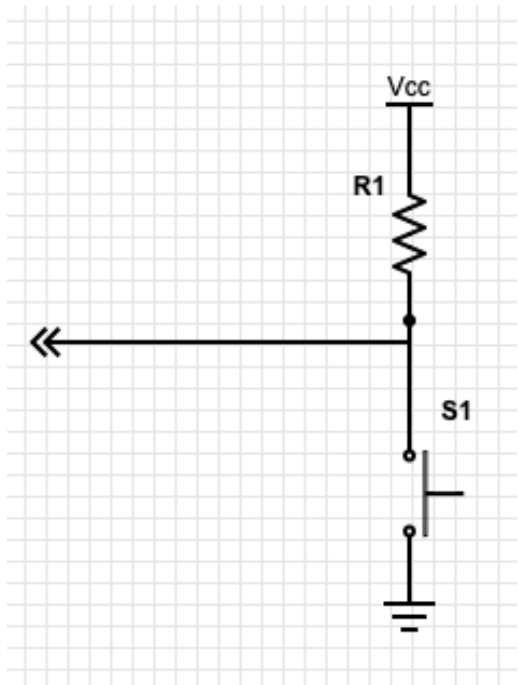
```
void setup()
{
  pinMode(2, INPUT) ;
  pinMode(8, OUTPUT) ;
}

void loop()
{
  int input = digitalRead(2) ;

  if( input == 0 )
  {
    digitalWrite(8, 0) ;
  }
  else
  {
    digitalWrite(8, 1) ;
  }
}
```

외부 인터럽트(External Interrupt)

- 풀링 vs 인터럽트



```
void setup()
{
  pinMode(2, INPUT) ;
  pinMode(8, OUTPUT) ;

  Serial.begin(9600) ;
}

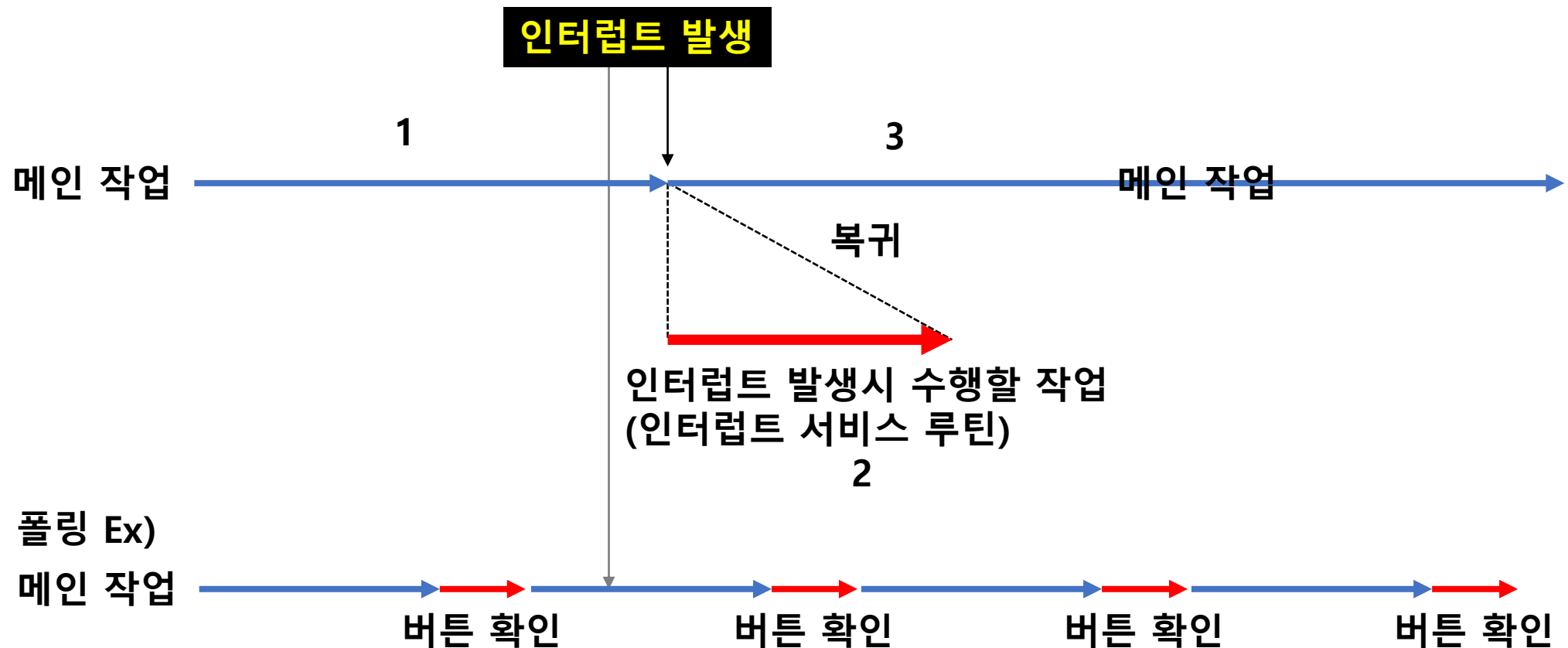
void loop()
{
  digitalWrite(8, 0) ;
  delay(1000) ;

  digitalWrite(8, 1) ;
  delay(1000) ;

  int input = digitalRead(2) ;
  if( input == 0 )
  {
    Serial.println("key") ;
  }
}
```

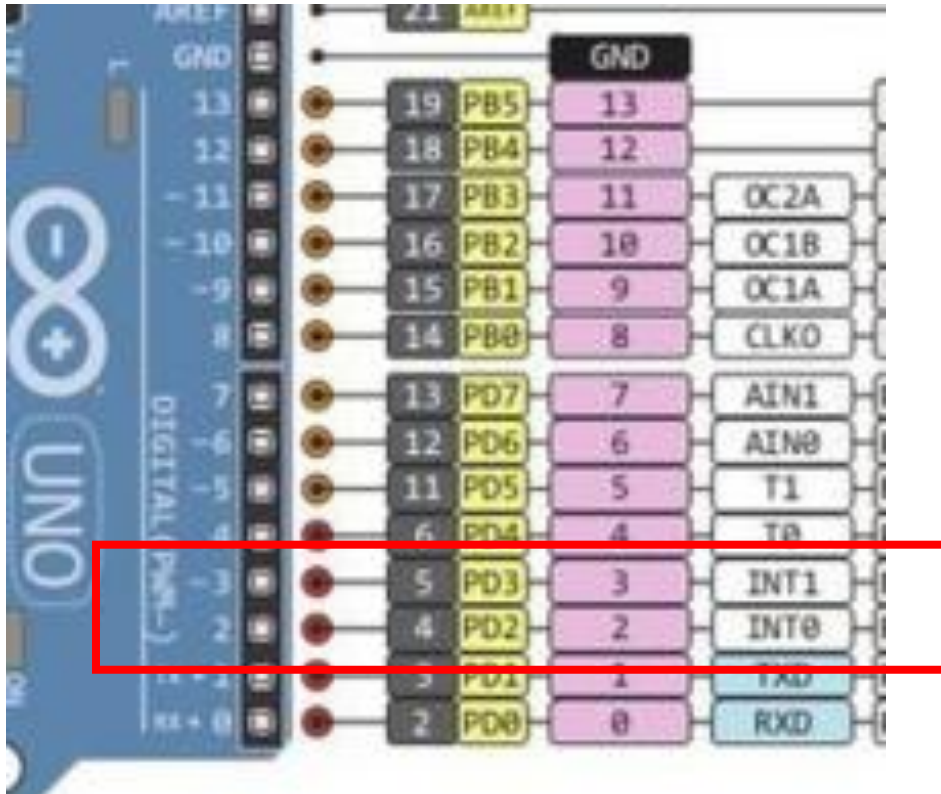
외부 인터럽트(External Interrupt)

- 폴링 vs 인터럽트



외부 인터럽트(External Interrupt)

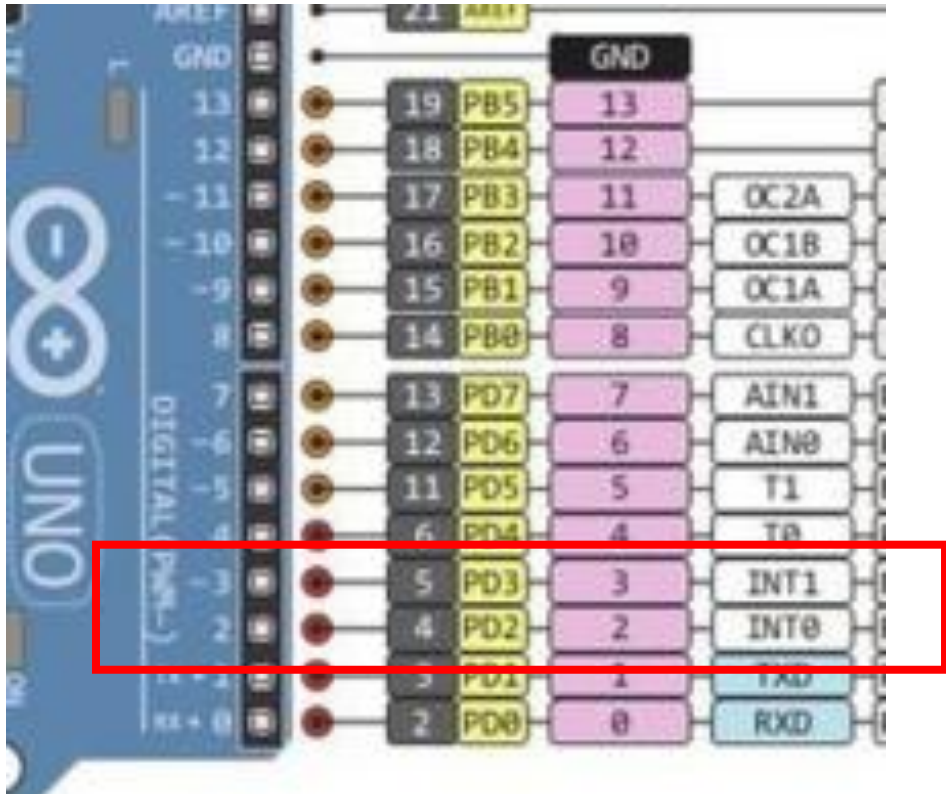
- 폴링 vs 인터럽트



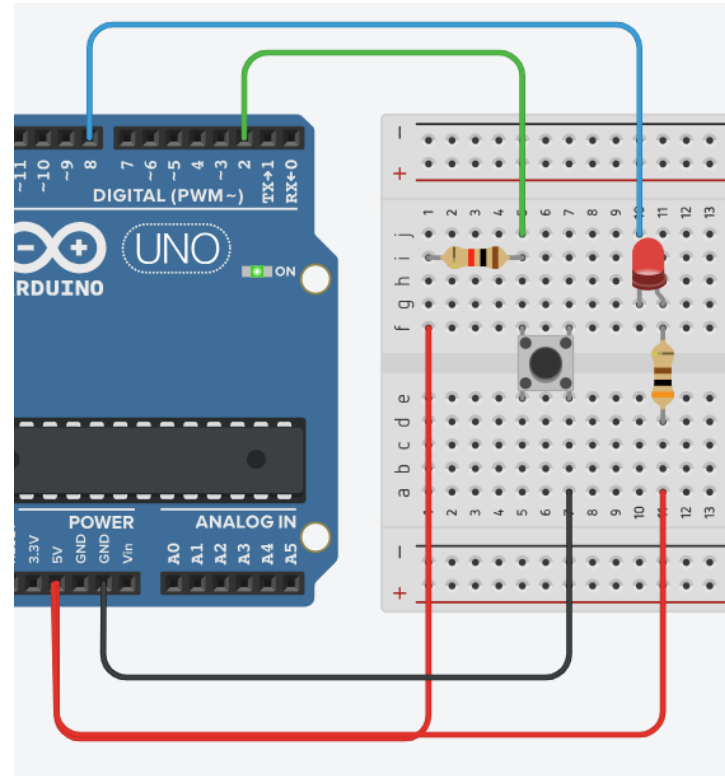
- INT1 : Interrupt #1
- INT0 : Interrupt #0

외부 인터럽트(External Interrupt)

- 폴링 vs 인터럽트



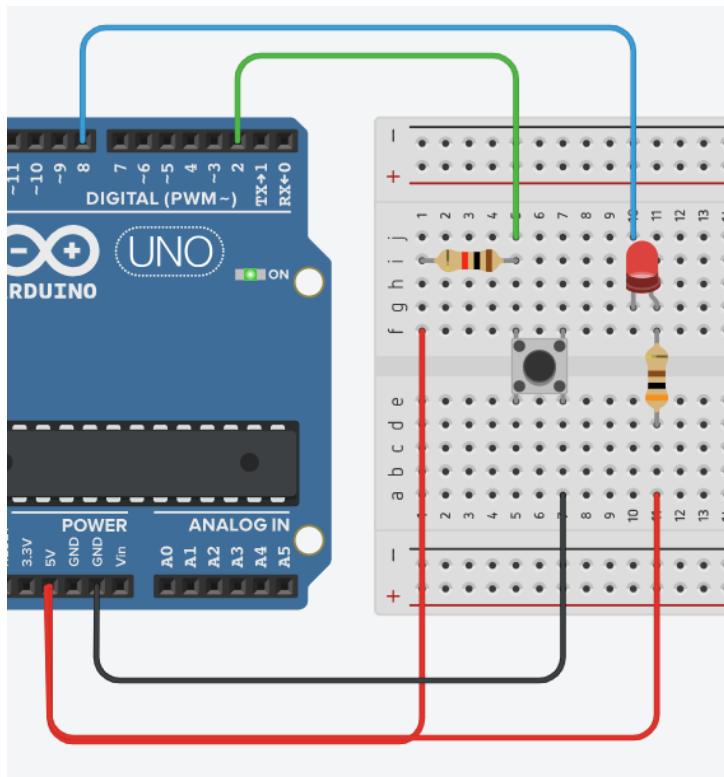
2 → INT0 : Interrupt #0



외부 인터럽트(External Interrupt)

- 폴링 vs 인터럽트

2 → INT0 : Interrupt #0



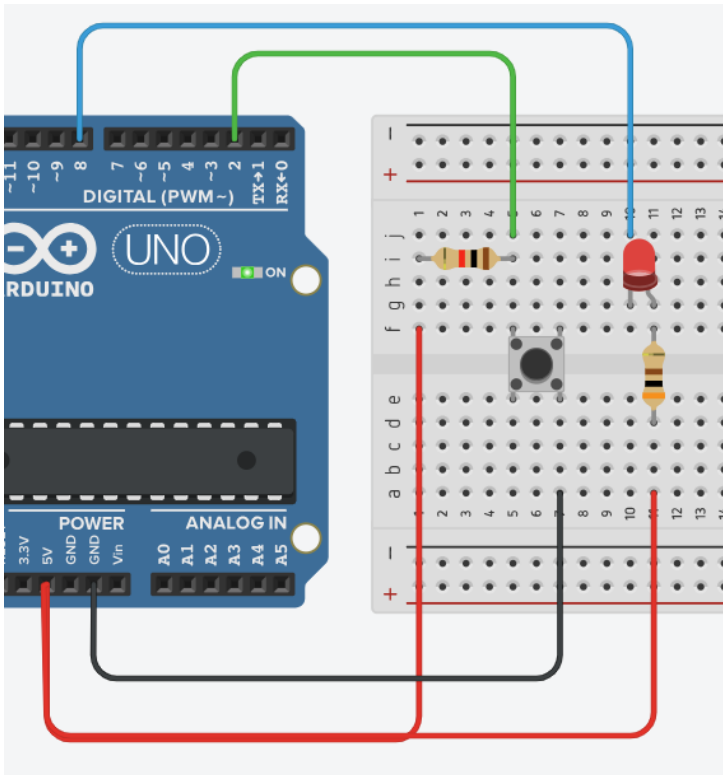
인터럽트 발동 조건 (mode)

모드	상태
LOW	핀이 LOW일때
CHANGE	LOW->HIGH or HIGH->LOW로 변할 때
RISING	LOW ->HIGH일때
FALLING	HIGH -> LOW일때
HIGH	핀이 HIGH일때

외부 인터럽트(External Interrupt)

- 폴링 vs 인터럽트

2 → INT0 : Interrupt #0



```
attachInterrupt( digitalPinToInterrupt(핀번호), 서비스루틴함수명, 모드 );
```

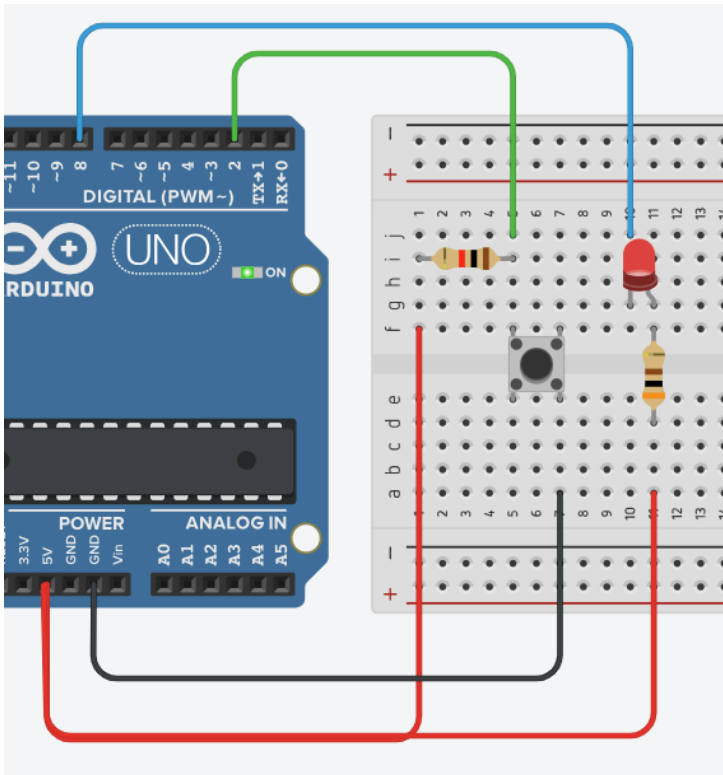
인터럽트 발동 조건 (mode)

모드	상태
LOW	핀이 LOW일때
CHANGE	LOW->HIGH or HIGH->LOW로 변할 때
RISING	LOW ->HIGH일때
FALLING	HIGH -> LOW일때
HIGH	핀이 HIGH일때

외부 인터럽트(External Interrupt)

• 폴링 vs 인터럽트

2 → INT0 : Interrupt #0



```
attachInterrupt( digitalPinToInterrupt(2), ExINT, FALLING );
```

```
attachInterrupt( digitalPinToInterrupt(핀번호), 서비스루틴함수명, 모드 );
```

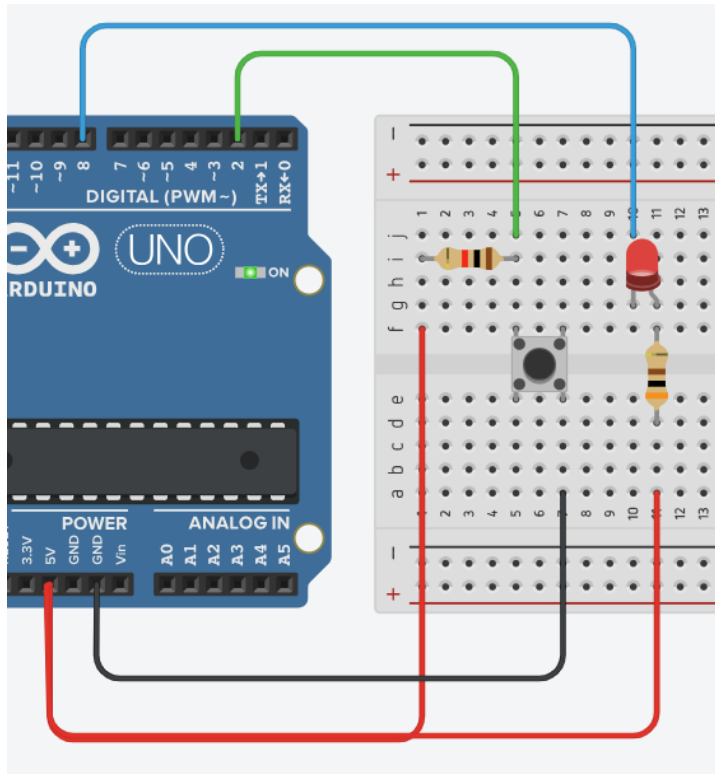
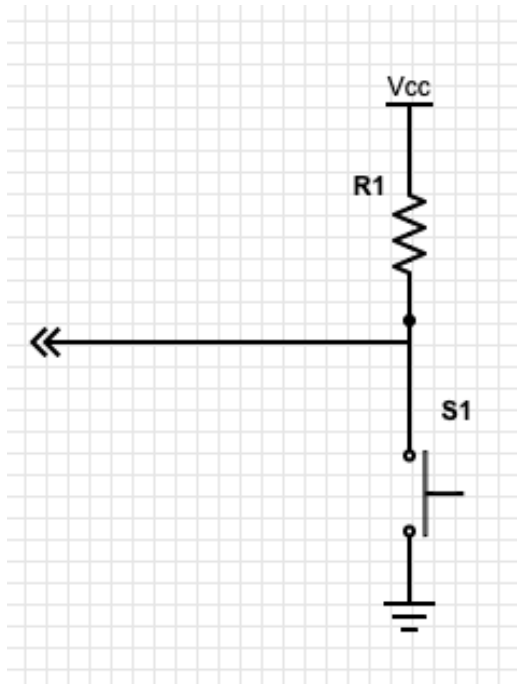
인터럽트 발동 조건 (mode)

모드	상태
LOW	핀이 LOW일때
CHANGE	LOW->HIGH or HIGH->LOW로 변할 때
RISING	LOW ->HIGH일때
FALLING	HIGH -> LOW일때
HIGH	핀이 HIGH일때

외부 인터럽트(External Interrupt)

• 플링 vs 인터럽트

`attachInterrupt(digitalPinToInterrupt(2), ExINT, FALLING);`



```
void setup()
{
  pinMode(8, INPUT) ;
  pinMode(2, OUTPUT) ;

  attachInterrupt( digitalPinToInterrupt(2), ExINT, FALLING );

  Serial.begin(9600) ;
}

void loop()
{
  digitalWrite(8, 0) ;
  delay(1000) ;

  digitalWrite(8, 1) ;
  delay(1000) ;
}

void ExINT()
{
  Serial.println("ExINT") ;
}
```