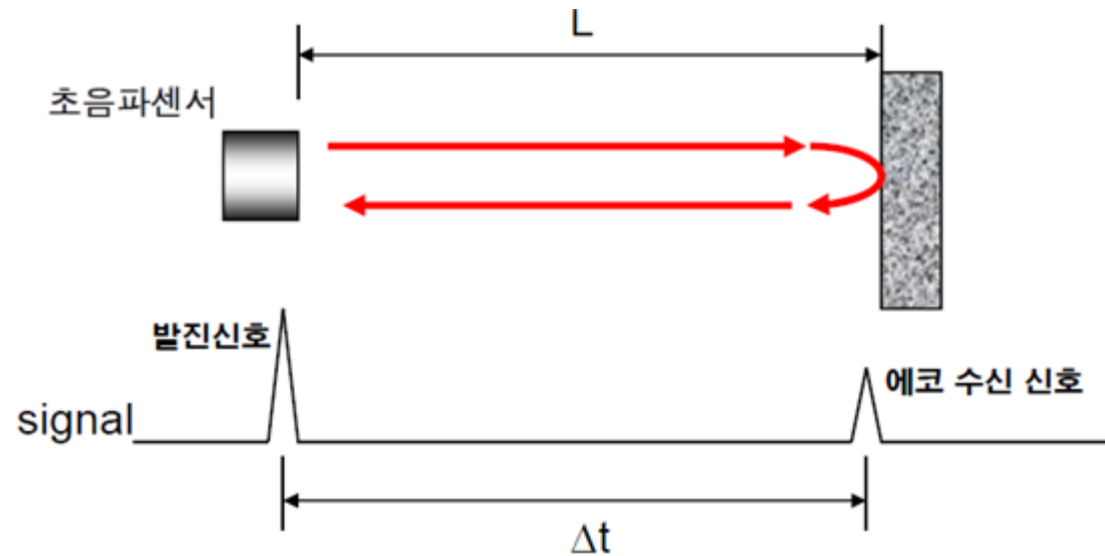


IR센서/초음파센서 실험



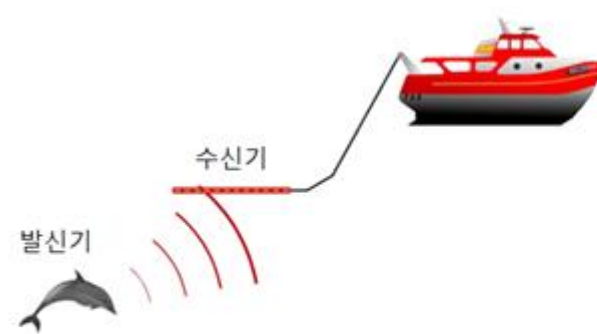
ToF(Time of Flight)

- **ToF**는 피사체를 향해 발사한 빛이나 소리가 반사돼 돌아오는 시간으로 거리를 계산해 사물의 입체감이나 공간 정보, 움직임 등을 인식하는 3D 센싱 기술이다



초음파 센서란?

- 초음파 센서는 인간이 들을 수 있는 범위를 벗어나 20,000Hz 이상의 음파를 사용해 센서로부터 지정된 목표 물체까지의 거리를 측정 및 계산하는 산업용 제어 장치.
- 음파는 기본적으로 고체, 액체 및 기체를 통과해 이동하는 압력파이고 거리를 측정하거나 표적이 있고 없음을 감지하기 위해 산업용 응용 분야에서 사용할 수 있다.



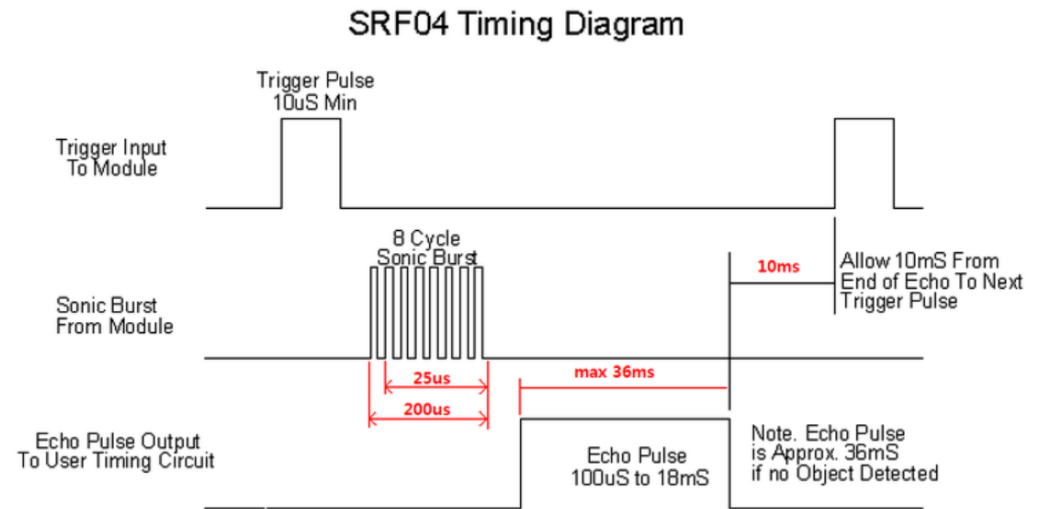
(a) 수동 소나(passive sonar)



(b) 능동 소나(active sonar)

초음파 센서 모듈

- SRF04 초음파 모듈을 사용하여 장애물까지의 거리 측정



초음파를 이용한 거리 측정

$$t = \frac{2 \times L(\text{물체와의 거리m})}{V_s(\text{음속m/s})}$$

t: 신호가 되돌아 올때까지 걸리는 시간(s)

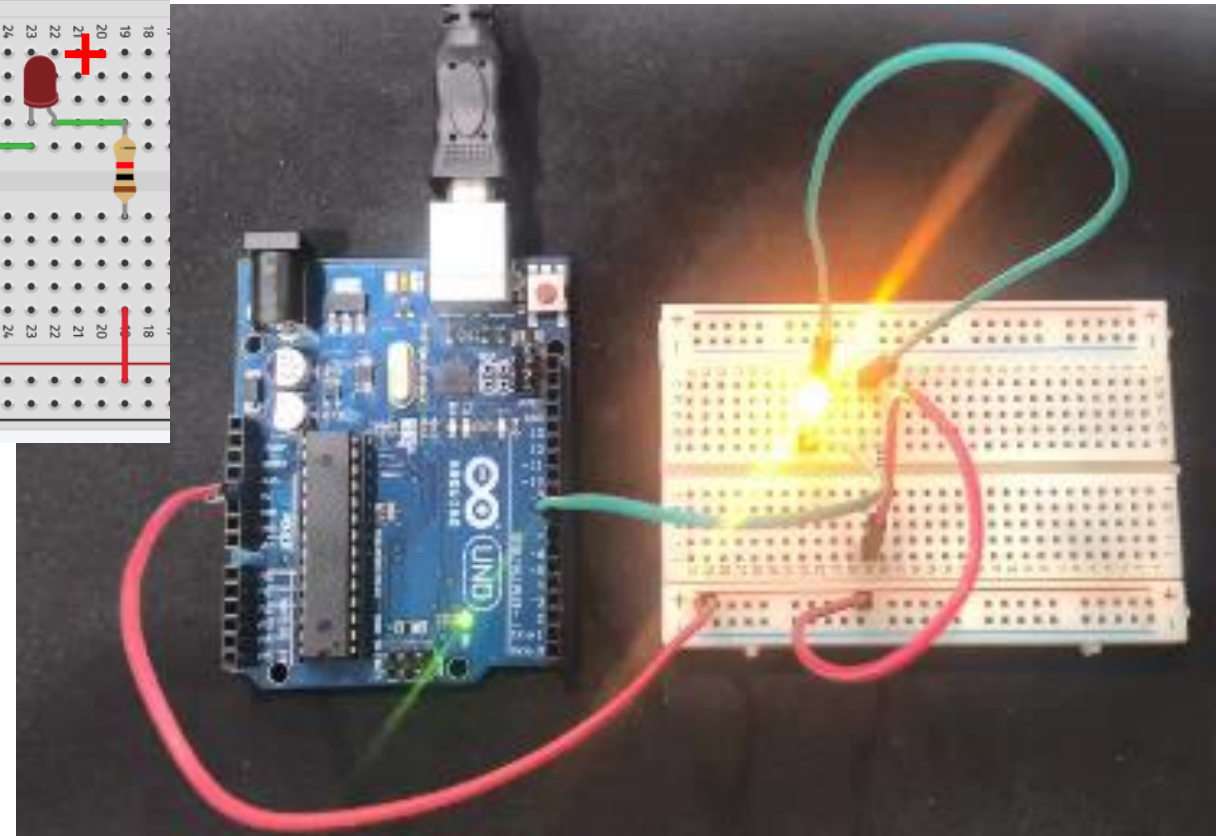
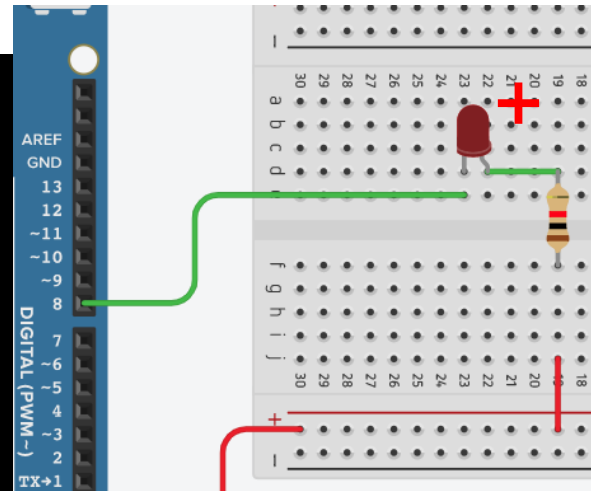
재료	속도 (m/s)
공기 (0℃)	331
공기 (20℃)	344
물 (25℃)	1498
목재 (소나무)	3300
유리	5000
철	5000
화강암	6000

LED를 이용한 digitalWrite 실험

- Arduino LED ON/OFF 실행

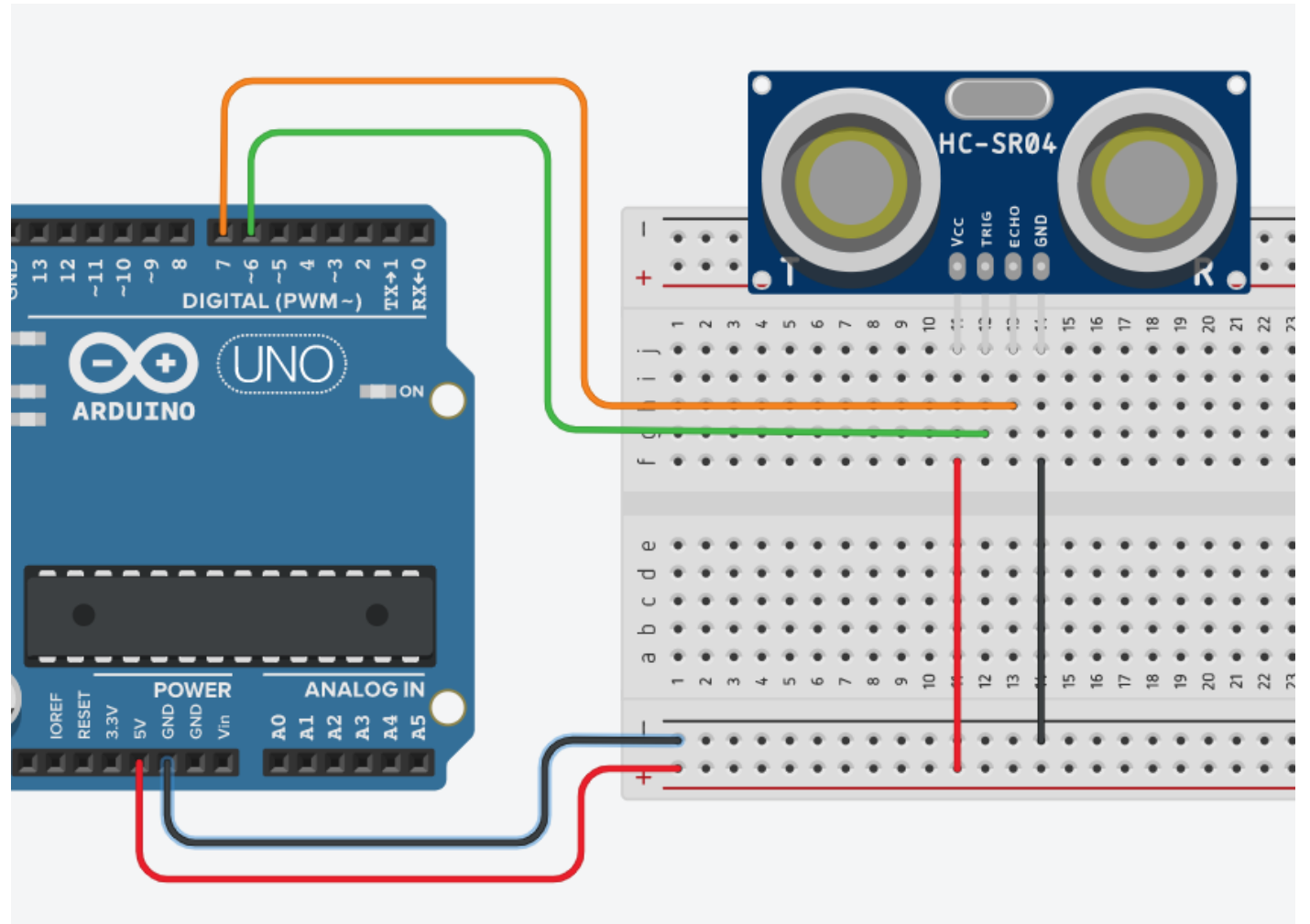
```
// C++ code
//
void setup()
{
  pinMode(8, OUTPUT);
}

void loop()
{
  digitalWrite(8, HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(8, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```



아두이노를 이용한 초음파 센서 실험

- VCC ↔ 아두이노 5V
- GND ↔ 아두이노 GND
- TRIG ↔ 아두이노 6
- ECHO ↔ 아두이노 7



아두이노를 이용한 초음파 센서 실험

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

    pinMode(6, OUTPUT);      //6 : Trigger
    pinMode(7, INPUT);       //7 : Echo
}

void loop()
{
    //trigger 발생
    digitalWrite(6, LOW) ;
    delayMicroseconds(2) ;
    digitalWrite(6, HIGH) ;
    delayMicroseconds(10) ;
    digitalWrite(6, LOW) ;

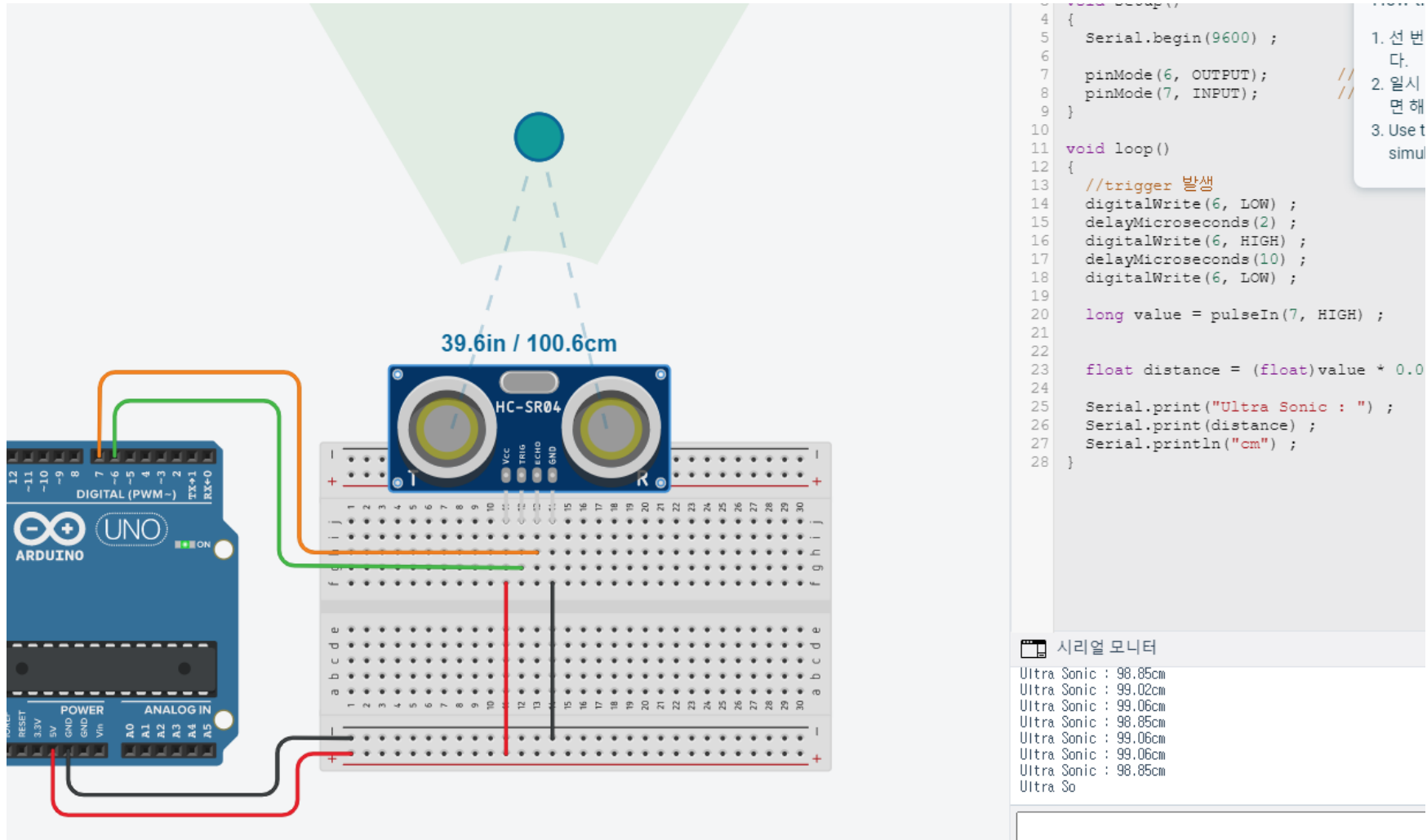
    long value = pulseIn(7, HIGH) ;

    float distance = (float)value * 0.01723 ;

    Serial.print("Ultra Sonic : ") ;
    Serial.print(distance) ;
    Serial.println("cm") ;
}
```

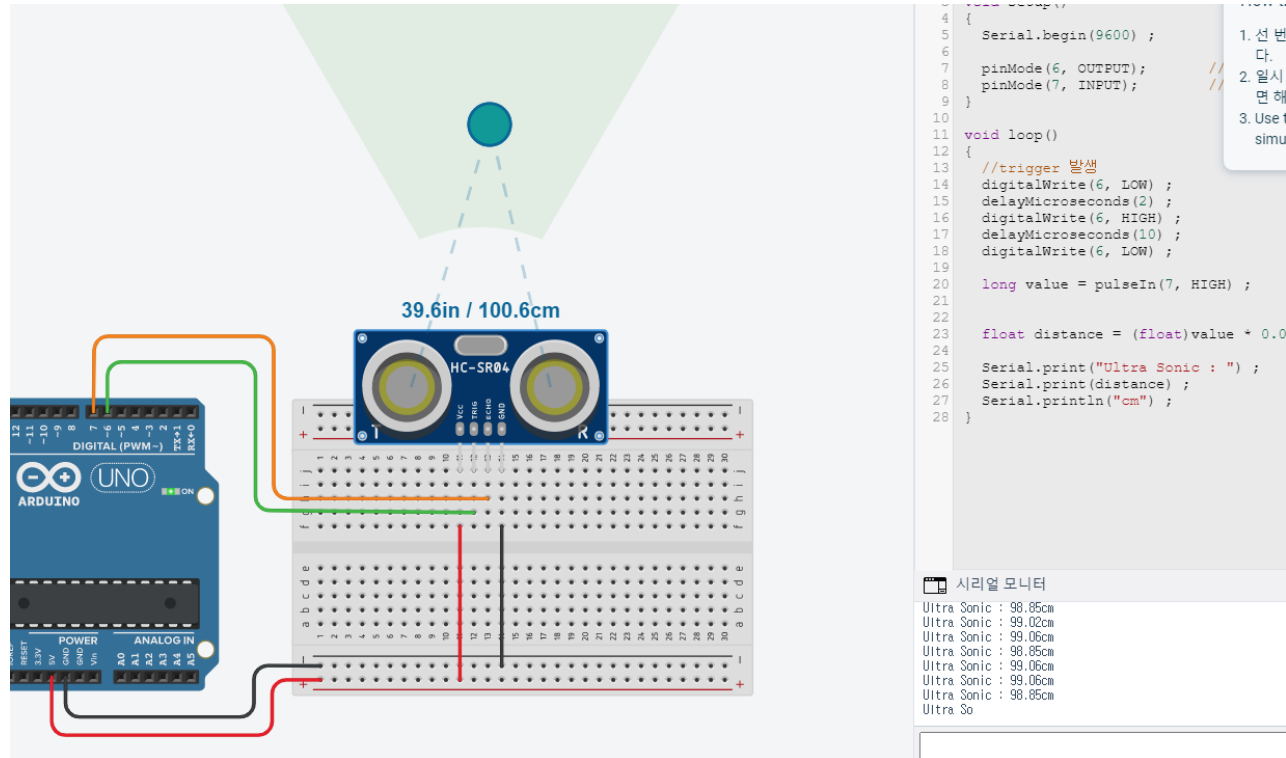
```
1  // C++ code
2  //
3  void setup()
4  {
5      Serial.begin(9600) ;
6
7      pinMode(6, OUTPUT);      //6 : Trigger
8      pinMode(7, INPUT);       //7 : Echo
9  }
10
11 void loop()
12 {
13     //trigger 발생
14     digitalWrite(6, LOW) ;
15     delayMicroseconds(2) ;
16     digitalWrite(6, HIGH) ;
17     delayMicroseconds(10) ;
18     digitalWrite(6, LOW) ;
19
20     long value = pulseIn(7, HIGH) ;
21
22
23     float distance = (float)value * 0.01723 ;
24
25     Serial.print("Ultra Sonic : ") ;
26     Serial.print(distance) ;
27     Serial.println("cm") ;
28 }
```


아두이노를 이용한 초음파 센서 실험



아두이노를 이용한 초음파 센서 실험

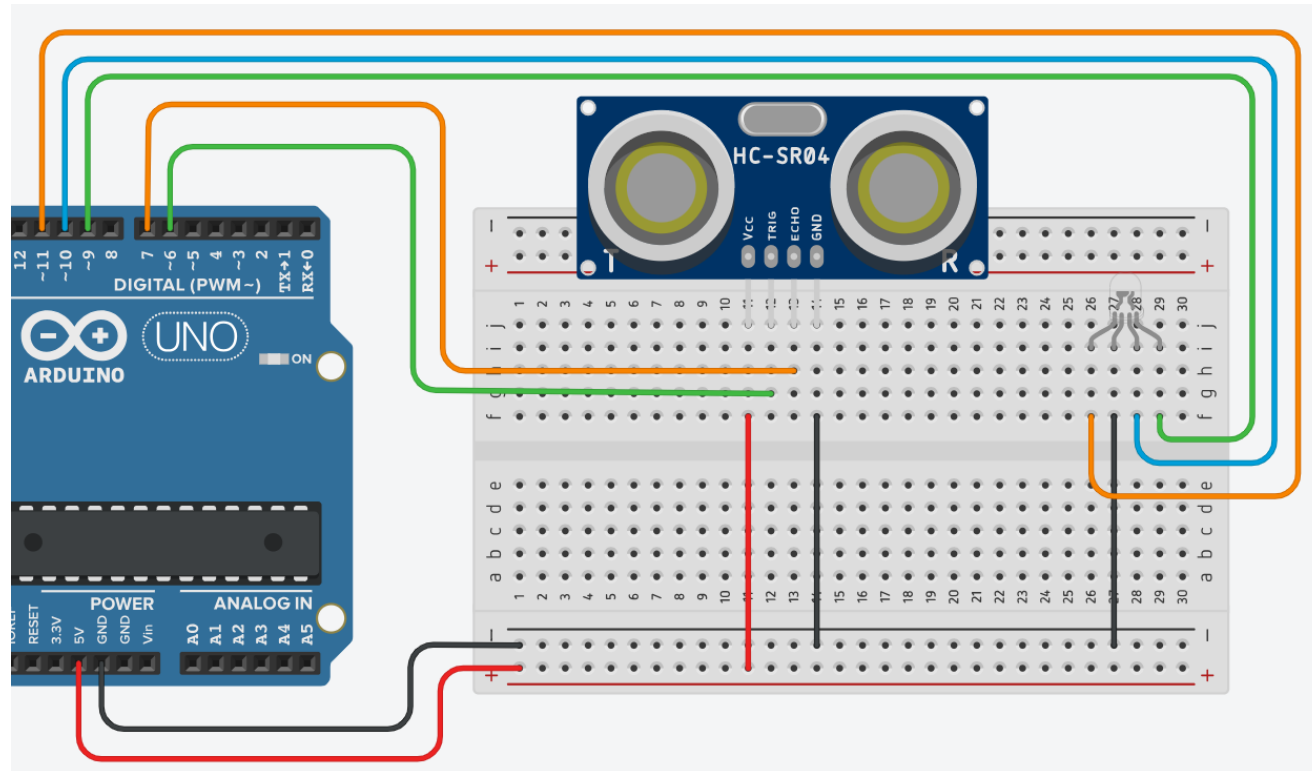
- QUIZ : 초음파 센서로 10cm이내에 장애물이 감지 되면 LED를 켜고 그렇지 않으면 LED를 끄는 회로와 프로그램을 완성 하시오.



아두이노를 이용한 초음파 센서 실험

• QUIZ :

- 초음파 센서로 10cm이내에 장애물이 감지 되면 **RED** LED를 켜고
- 초음파 센서로 20cm이내에 장애물이 감지 되면 **BLUE** LED를 켜고
- 그렇지 않으면 **GREEN** LED를 켜는 회로와 프로그램을 완성 하시오.



• 초음파센서

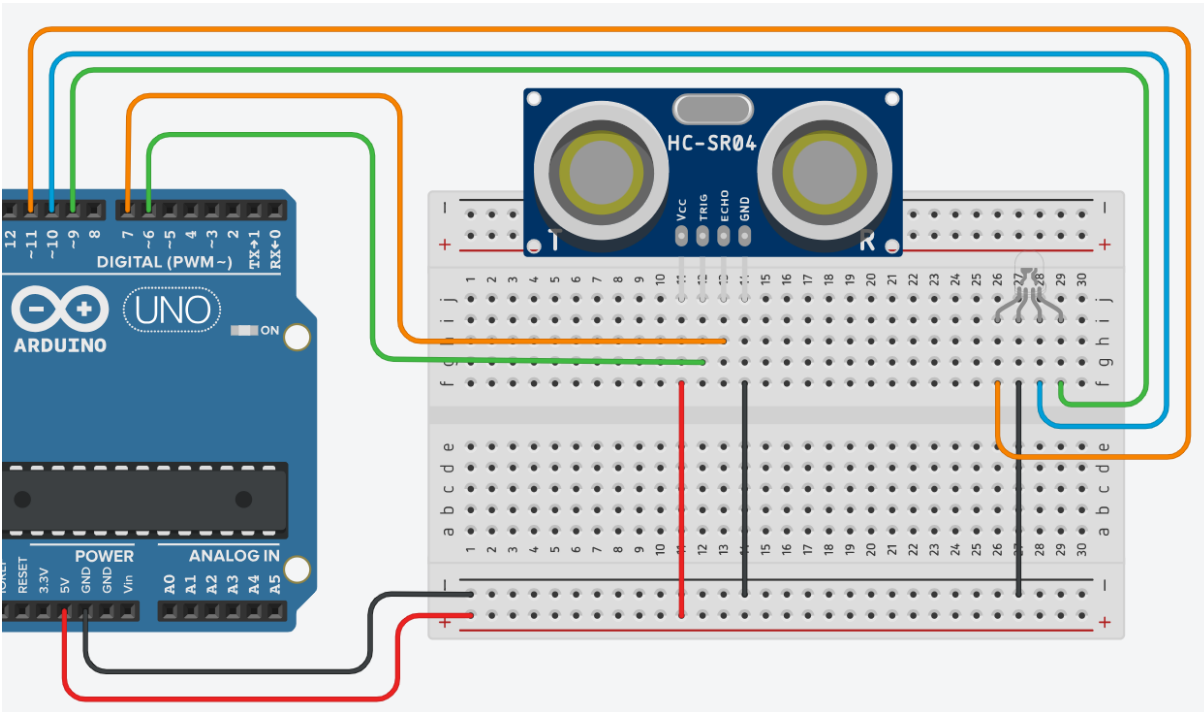
- VCC ↔ 아두이노 5V
- GND ↔ 아두이노 GND
- TRIG ↔ 아두이노 6
- ECHO ↔ 아두이노 7

• LED

- RED ↔ 아두이노 11번핀
- BLUE ↔ 아두이노 10번핀
- GREEN ↔ 아두이노 9번핀

• QUIZ :

- 초음파 센서로 10cm이내에 장애물이 감지 되면 **RED** LED를 켜고
- 초음파 센서로 20cm이내에 장애물이 감지 되면 **BLUE** LED를 켜고
- 그렇지 않으면 **GREEN** LED를 켜는 회로와 프로그램을 완성 하시오.



```
void setup() {  
  pinMode(6, OUTPUT);    //6 : Trigger  
  pinMode(7, INPUT);    //7 : Echo  
  pinMode(11, OUTPUT);  //RED LED  
  pinMode(10, OUTPUT);  //BLUE LED  
  pinMode(9, OUTPUT);   //GREEN LED  
}
```

```
void loop() {  
  digitalWrite(6, LOW) ;  
  delayMicroseconds(2) ;  
  digitalWrite(6, HIGH) ;  
  delayMicroseconds(10) ;  
  digitalWrite(6, LOW) ;
```

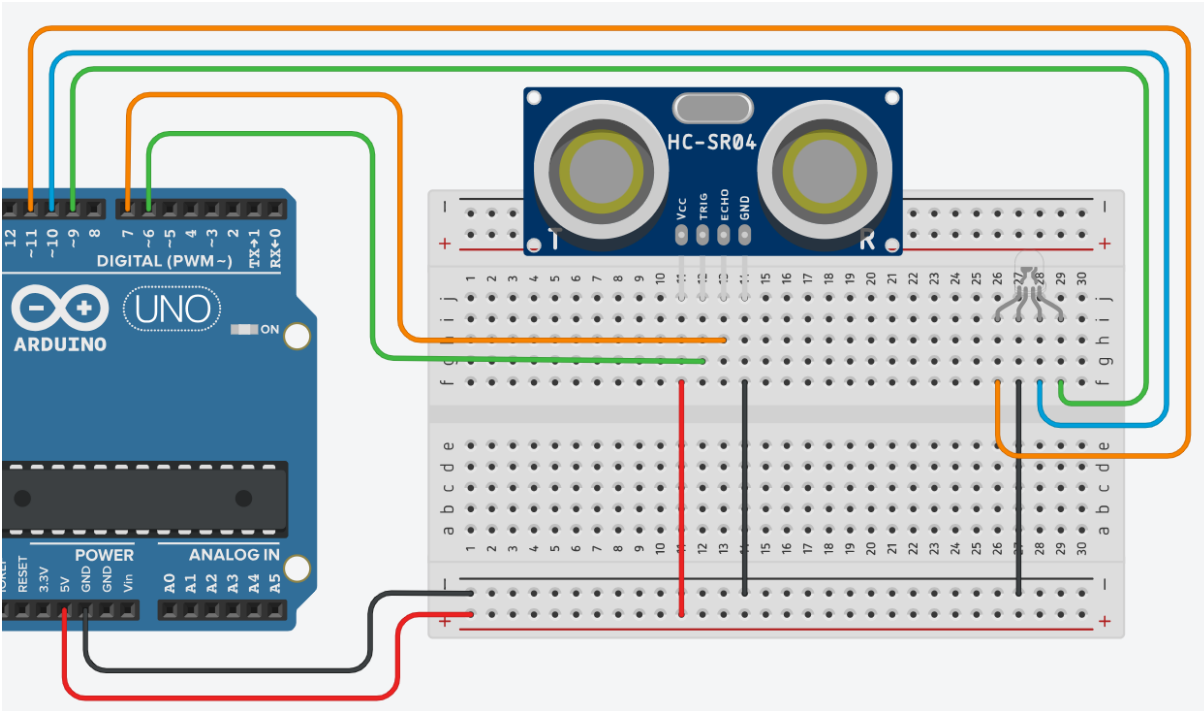
```
  long value = pulseIn(7, HIGH) ;
```

```
  float distance = (float)value * 0.01723 ;
```

```
  if( distance < 10 ) {  
    digitalWrite(11, HIGH) ;           //RED ON  
    digitalWrite(10, LOW) ;           //BLUE OFF  
    digitalWrite(9, LOW) ;           //GREEN OFF  
  }  
  else if( distance < 20 ) {  
    digitalWrite(11, LOW) ;           //RED OFF  
    digitalWrite(10, LOW) ;           //BLUE ON  
    digitalWrite(9, HIGH) ;           //GREEN OFF  
  }  
  else {  
    digitalWrite(11, LOW) ;           //RED OFF  
    digitalWrite(10, HIGH) ;          //GREEN ON  
    digitalWrite(9, LOW) ;            //blue off  
  }  
}
```

• QUIZ :

- 초음파 센서로 10cm이내에 장애물이 감지 되면 **RED** LED를 켜고
- 초음파 센서로 20cm이내에 장애물이 감지 되면 **BLUE** LED를 켜고
- 그렇지 않으면 **GREEN** LED를 켜는 회로와 프로그램을 완성 하시오.



```
void setup() {  
  pinMode(6, OUTPUT);          //6 : Trigger  
  pinMode(7, INPUT); //7 : Echo  
  pinMode(11, OUTPUT); //RED LED  
  pinMode(10, OUTPUT); //BLUE LED  
  pinMode(9, OUTPUT); //GREEN LED  
}
```

```
void loop() {  
  digitalWrite(6, LOW) ;  
  delayMicroseconds(2) ;  
  digitalWrite(6, HIGH) ;  
  delayMicroseconds(10) ;  
  digitalWrite(6, LOW) ;
```

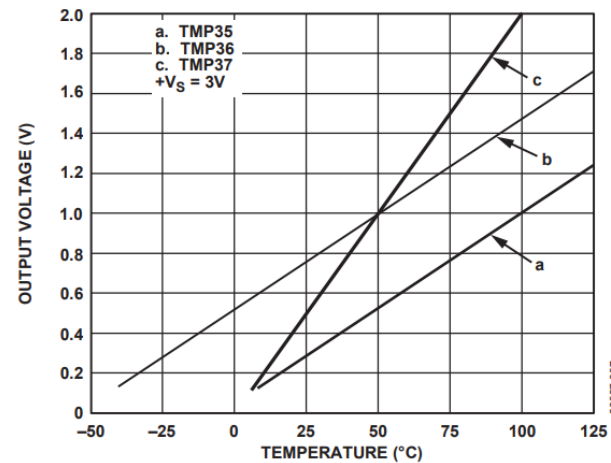
long value = pulseIn(7, HIGH) ;

float **distance** = (float)value * 0.01723 ;

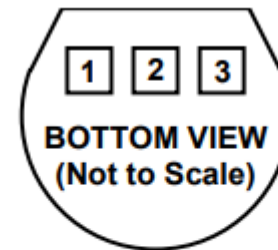
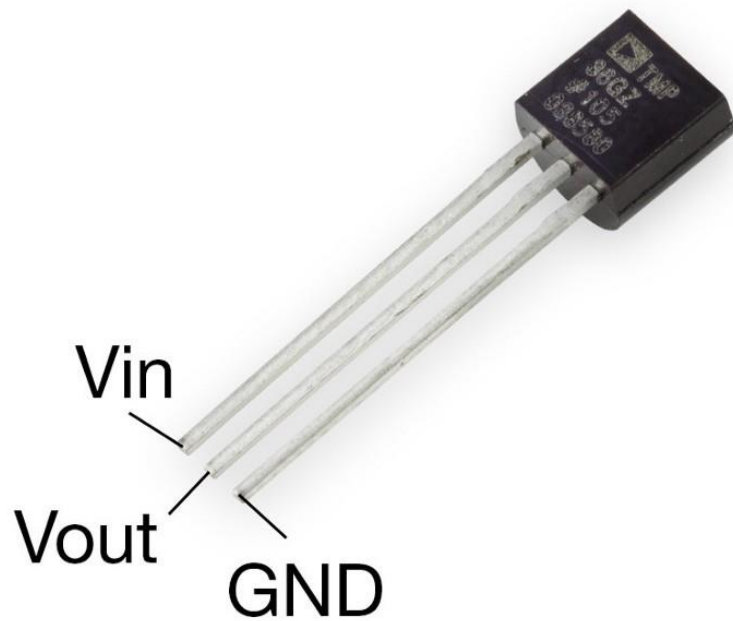
```
if( distance < 10 ) {  
  analogWrite(11, 234) ;  
  analogWrite(10, 63) ;  
  analogWrite(9, 247) ;  
}  
else {  
  analogWrite(11, 255) ;  
  analogWrite(10, 254) ;  
  analogWrite(9, 145) ;  
}  
}
```

TMP36

- 온도센서는 온도를 감지해 전기신호로 바꿔주는 센서를 의미
- TMP36
 - 상온에서 대략 750mV를 출력
 - 온도 1 °C가 변화하면 10mV의 출력 전압이 변화 함
 - 정밀도는 ± 1 °C로 정밀한 온도 감지는 어려움.
 - 사용하기 쉽고 저렴하여 정밀한 온도 감지가 필요 없는 어플리케이션이 많이 사용 됨.



TMP36 핀연결



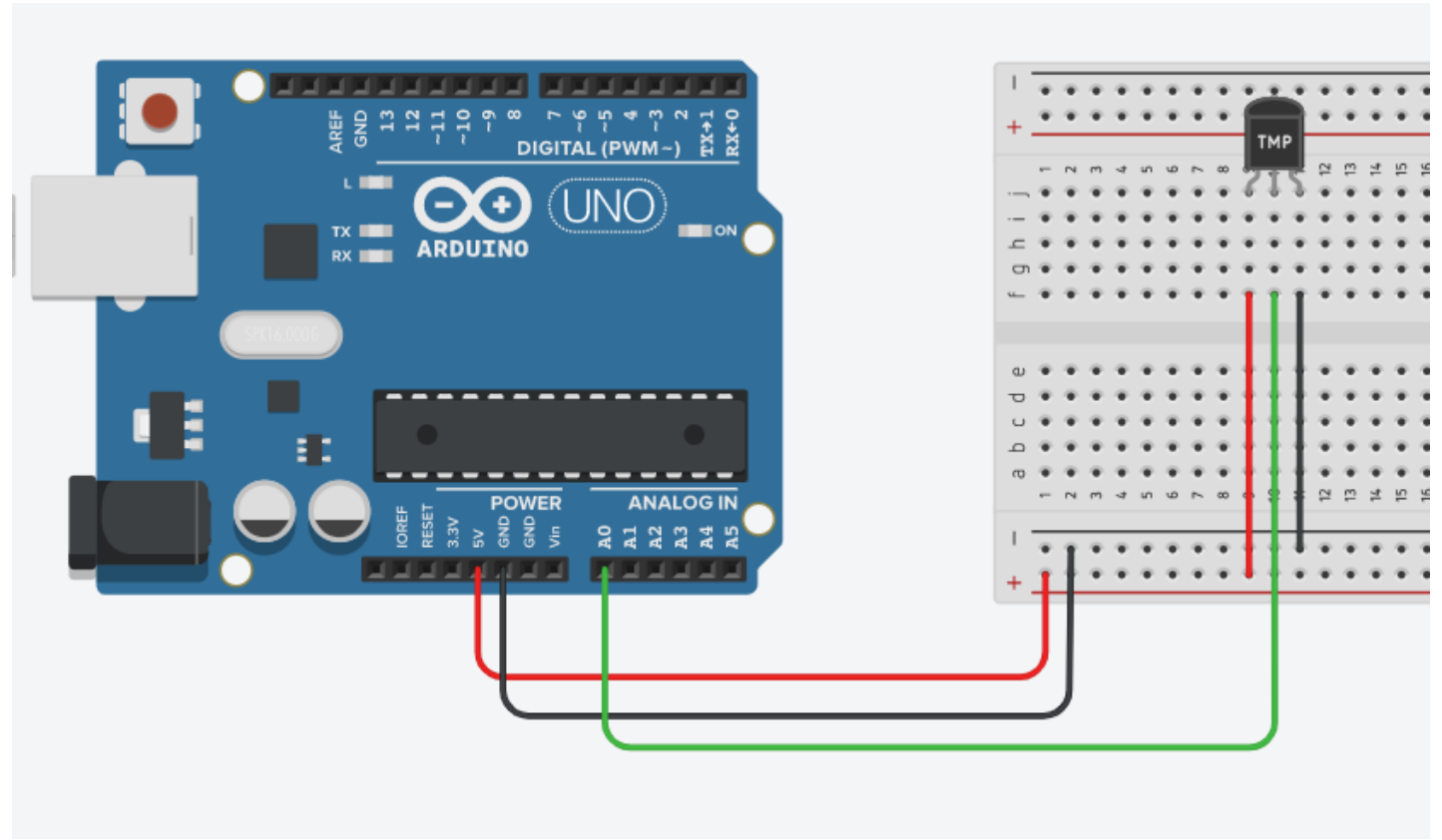
PIN 1, $+V_S$; PIN 2, V_{OUT} ; PIN 3, GND

Figure 4. T-3 (TO-92)

00337-004

TMP36 + 아두이노 실험

- TMP Vin <> 아두이노 5V
- TMP Vout <> 아두이노 A0
- TMP GND <> 아두이노 GND



코드 작성

void setup()

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

void loop()

```
{  
  int reading = analogRead(A0);  
  Serial.println(reading);  
}
```

The screenshot displays an Arduino IDE interface. At the top, a blue header reads '온도 센서 [TMP36]'. Below it, a text box contains '이름 온도센서'. The central part of the image shows a breadboard circuit. A TMP36 temperature sensor is connected to a breadboard. Its VCC pin is connected to a red wire leading to the positive terminal of a 5V power source. Its GND pin is connected to a green wire leading to the negative terminal. Its AO pin is connected to a black wire leading to analog pin A0 on the Arduino. A potentiometer is also connected to the breadboard, with its wiper connected to A0. The right side of the IDE shows the code for the project, which is identical to the code provided in the text blocks. Below the code editor, the '시리얼 모니터' (Serial Monitor) window is open, displaying the output of the program. The output shows a repeating sequence of values: 0.83 volts, 33.01 degrees C, and 91.41 degrees F.

```
1 void setup()  
2 {  
3   Serial.begin(9600);  
4 }  
5  
6 void loop()  
7 {  
8   int reading = analogRead(A0);  
9  
10  float voltage = reading * 5.0;  
11  voltage /= 1024.0;  
12  
13  Serial.print(voltage); Serial.println(" volts");  
14  
15  float temperatureC = (voltage - 0.5) * 100 ;  
16  Serial.print(temperatureC); Serial.println(" degrees C");  
17  
18  float temperatureF = (temperatureC * 9.0 / 5.0) + 32.0;  
19  Serial.print(temperatureF); Serial.println(" degrees F");  
20  
21  delay(1000);  
22 }
```

시리얼 모니터

```
0.83 volts  
33.01 degrees C  
91.41 degrees F  
0.83 volts  
33.01 degrees C  
91.41 degrees F  
0.83 volts  
33.01 degrees C  
91.41 degrees F  
0.83 volts  
33.01 degrees C  
91.41 degrees F  
0.83 volts  
33.01 degrees C  
91.41 degrees F
```

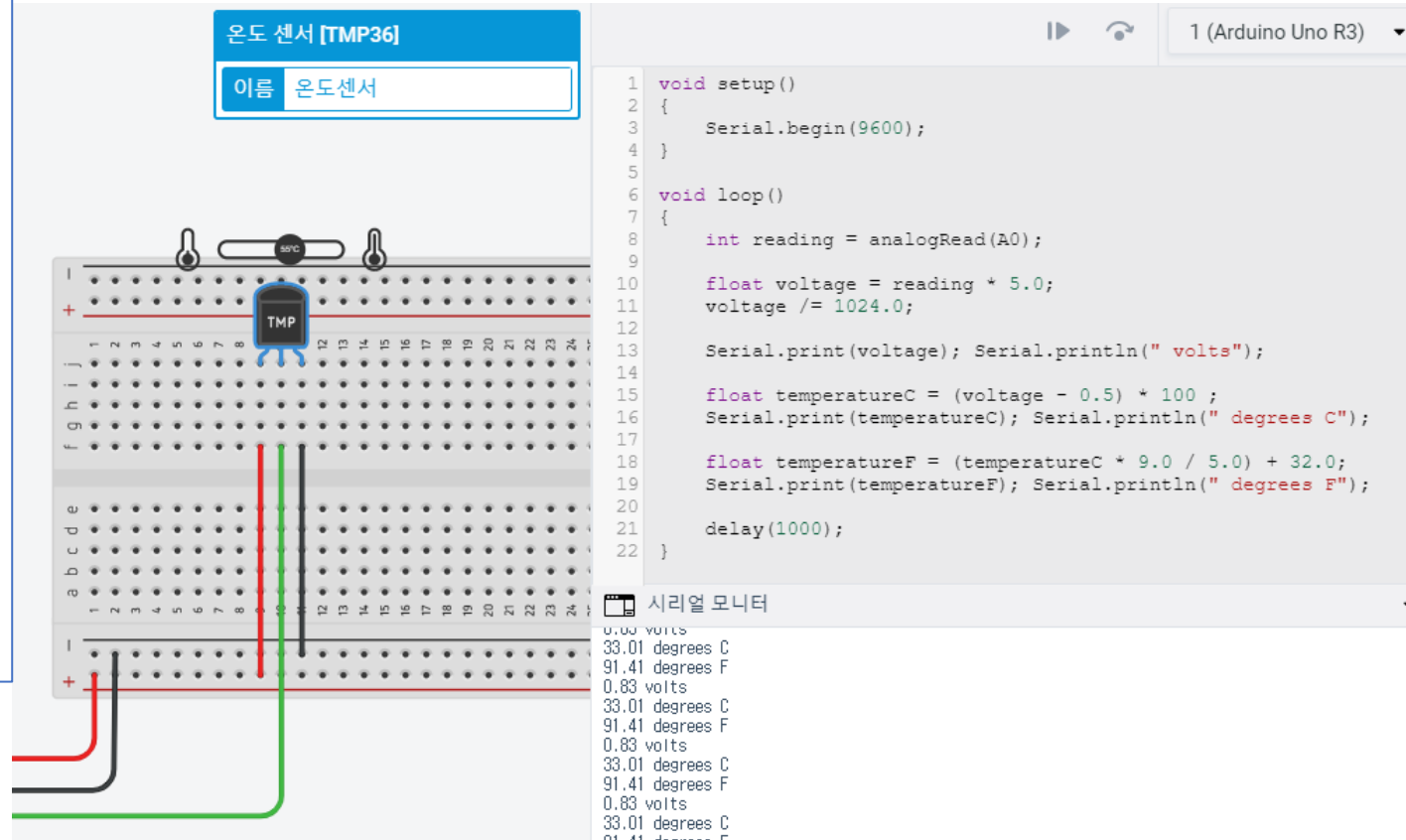
코드 작성

void setup()

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

void loop()

```
{  
  int reading = analogRead(A0);  
  
  float voltage = (reading / 1024.0) * 5.0;  
  
  Serial.print(voltage); Serial.println(" volts");  
  
  delay(1000);  
}
```



온도 센서 [TMP36]

이름 온도센서

```
1 void setup()  
2 {  
3   Serial.begin(9600);  
4 }  
5  
6 void loop()  
7 {  
8   int reading = analogRead(A0);  
9  
10  float voltage = reading * 5.0;  
11  voltage /= 1024.0;  
12  
13  Serial.print(voltage); Serial.println(" volts");  
14  
15  float temperatureC = (voltage - 0.5) * 100;  
16  Serial.print(temperatureC); Serial.println(" degrees C");  
17  
18  float temperatureF = (temperatureC * 9.0 / 5.0) + 32.0;  
19  Serial.print(temperatureF); Serial.println(" degrees F");  
20  
21  delay(1000);  
22 }
```

시리얼 모니터

0.00 volts
33.01 degrees C
91.41 degrees F
0.83 volts
33.01 degrees C
91.41 degrees F
0.83 volts
33.01 degrees C
91.41 degrees F
0.83 volts
33.01 degrees C
91.41 degrees F

코드 작성

void setup()

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

void loop()

```
{  
  int reading = analogRead(A0);  
  
  float voltage = (reading / 1024.0) * 5.0;  
  
  Serial.print(voltage); Serial.println(" volts");  
  
  float temperatureC = (voltage - 0.5) * 100 ;  
  Serial.print(temperatureC); Serial.println(" degrees C");  
  
  delay(1000);  
}
```

The screenshot displays an Arduino IDE interface. At the top, a blue label reads '온도 센서 [TMP36]' (Temperature Sensor [TMP36]), with a dropdown menu showing '이름 온도센서' (Name: Temperature Sensor). Below this, a breadboard circuit is shown. A TMP36 temperature sensor is connected to a breadboard. Its VCC pin is connected to a red wire leading to a 5V pin on the breadboard. Its GND pin is connected to a green wire leading to a GND pin. Its AO pin is connected to a black wire leading to an analog input pin (A0) on the breadboard. A potentiometer is also connected to the breadboard. The main code editor shows the following code:

```
1 void setup()  
2 {  
3   Serial.begin(9600);  
4 }  
5  
6 void loop()  
7 {  
8   int reading = analogRead(A0);  
9  
10  float voltage = reading * 5.0;  
11  voltage /= 1024.0;  
12  
13  Serial.print(voltage); Serial.println(" volts");  
14  
15  float temperatureC = (voltage - 0.5) * 100 ;  
16  Serial.print(temperatureC); Serial.println(" degrees C");  
17  
18  float temperatureF = (temperatureC * 9.0 / 5.0) + 32.0;  
19  Serial.print(temperatureF); Serial.println(" degrees F");  
20  
21  delay(1000);  
22 }
```

At the bottom, the '시리얼 모니터' (Serial Monitor) window is open, showing the following output:

```
0.00 volts  
33.01 degrees C  
91.41 degrees F  
0.83 volts  
33.01 degrees C  
91.41 degrees F  
0.83 volts  
33.01 degrees C  
91.41 degrees F  
0.83 volts  
33.01 degrees C  
91.41 degrees F
```

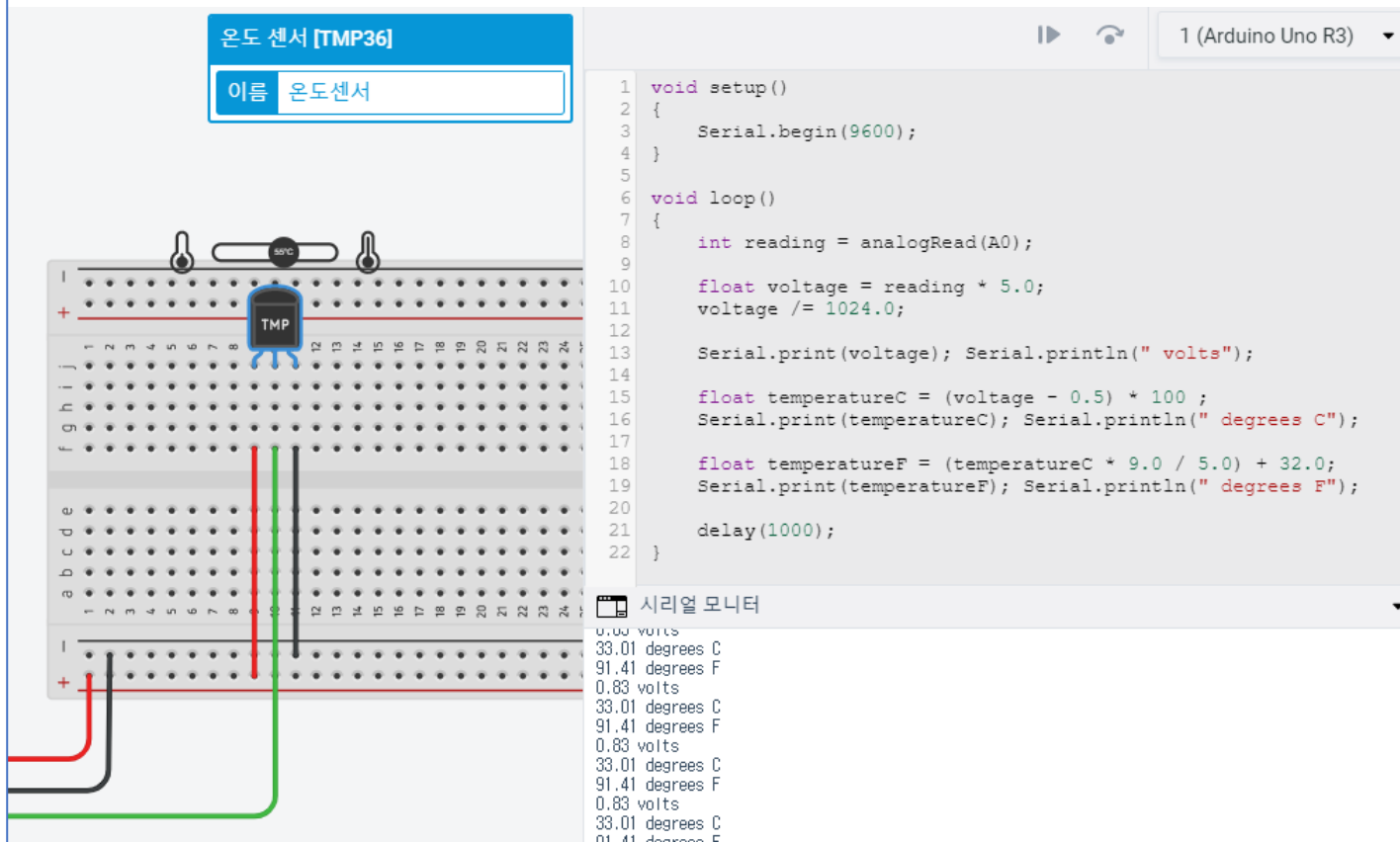
코드 작성

void setup()

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

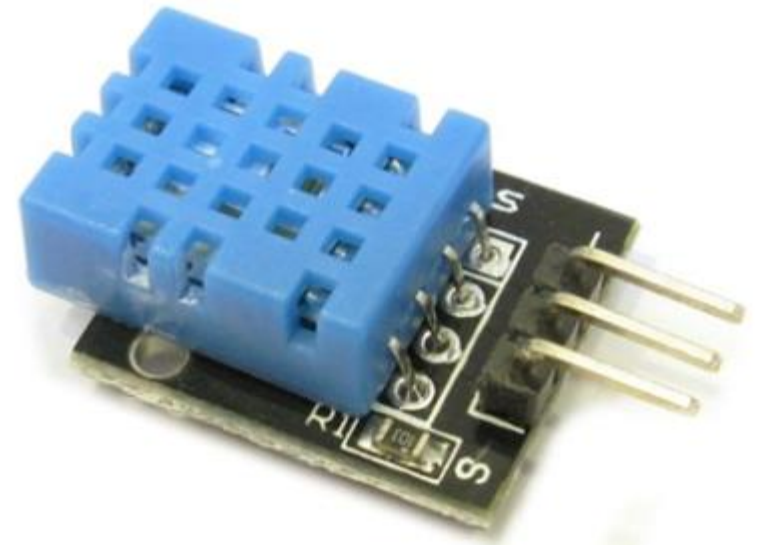
void loop()

```
{  
  int reading = analogRead(A0);  
  
  float voltage = (reading / 1024.0) * 5.0;  
  
  Serial.print(voltage); Serial.println(" volts");  
  
  float temperatureC = (voltage - 0.5) * 100 ;  
  Serial.print(temperatureC); Serial.println(" degrees C");  
  
  float temperatureF = (temperatureC * 9.0 / 5.0) + 32.0;  
  Serial.print(temperatureF); Serial.println(" degrees F");  
  
  delay(1000);  
}
```



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>

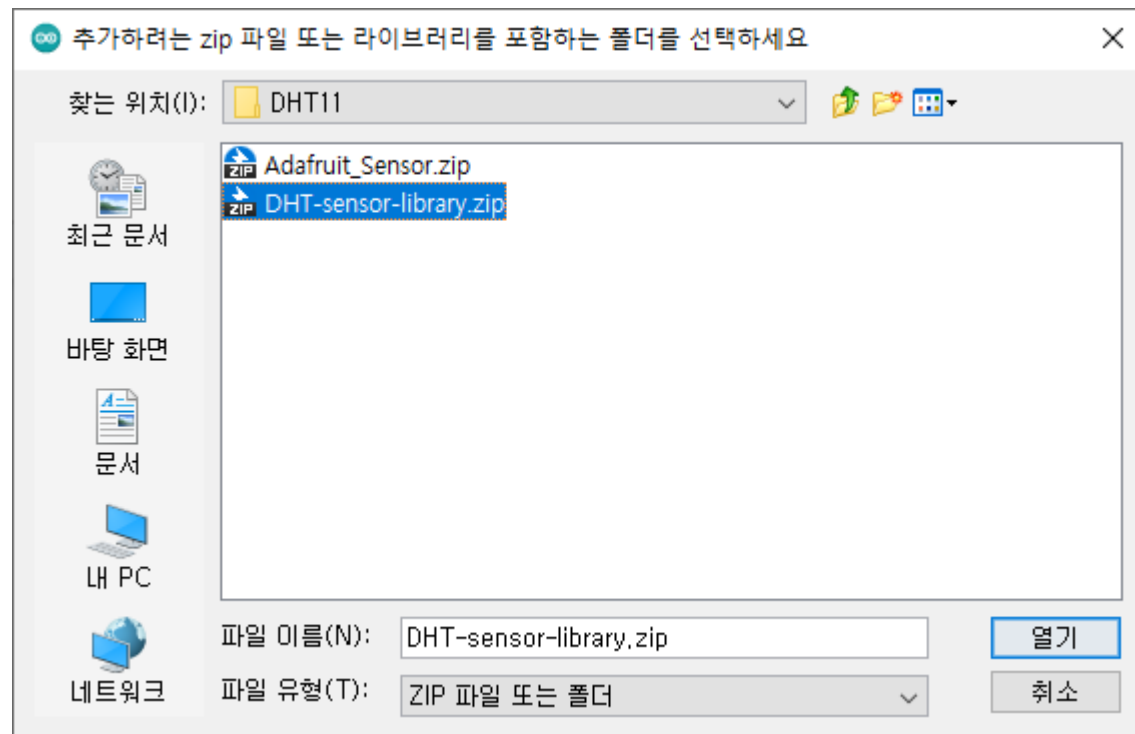
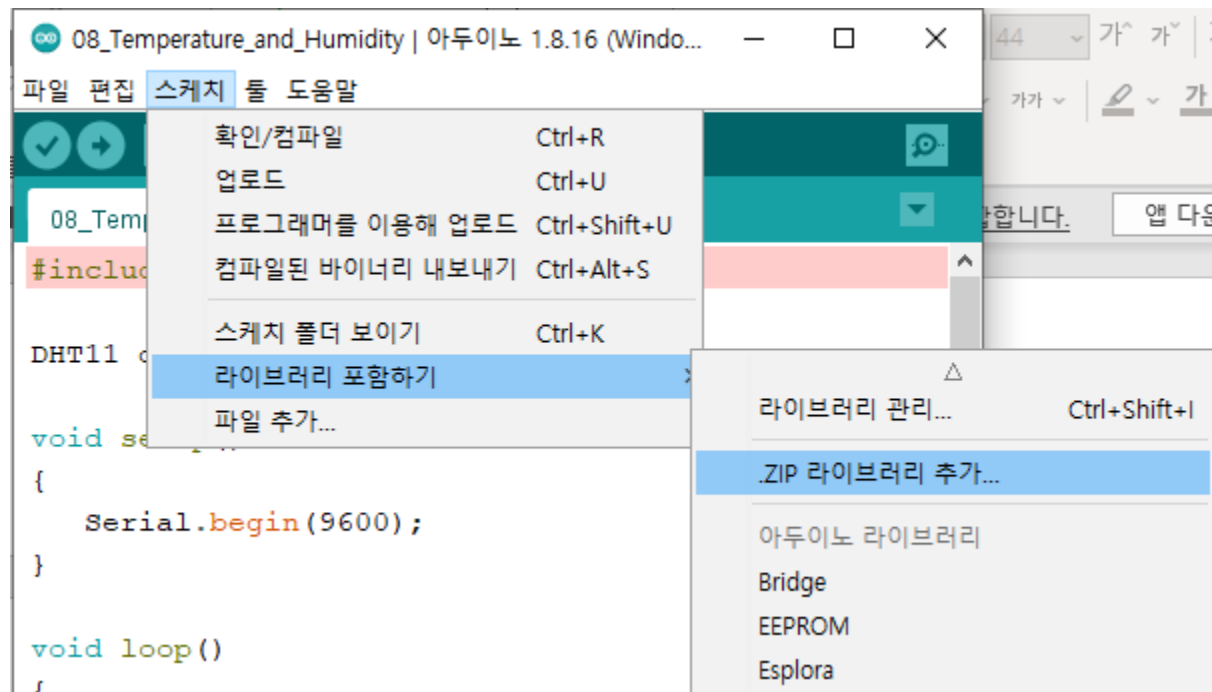
The screenshot shows the GitHub repository page for `adafruit/DHT-sensor-library`. The repository is public and has 162 watchers, 1.6k stars, and 1.2k forks. It contains 17 issues, 13 pull requests, and 142 commits. The latest commit is by `dherrada` bumping the version to 1.4.3, 29 days ago.

The file list includes:

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

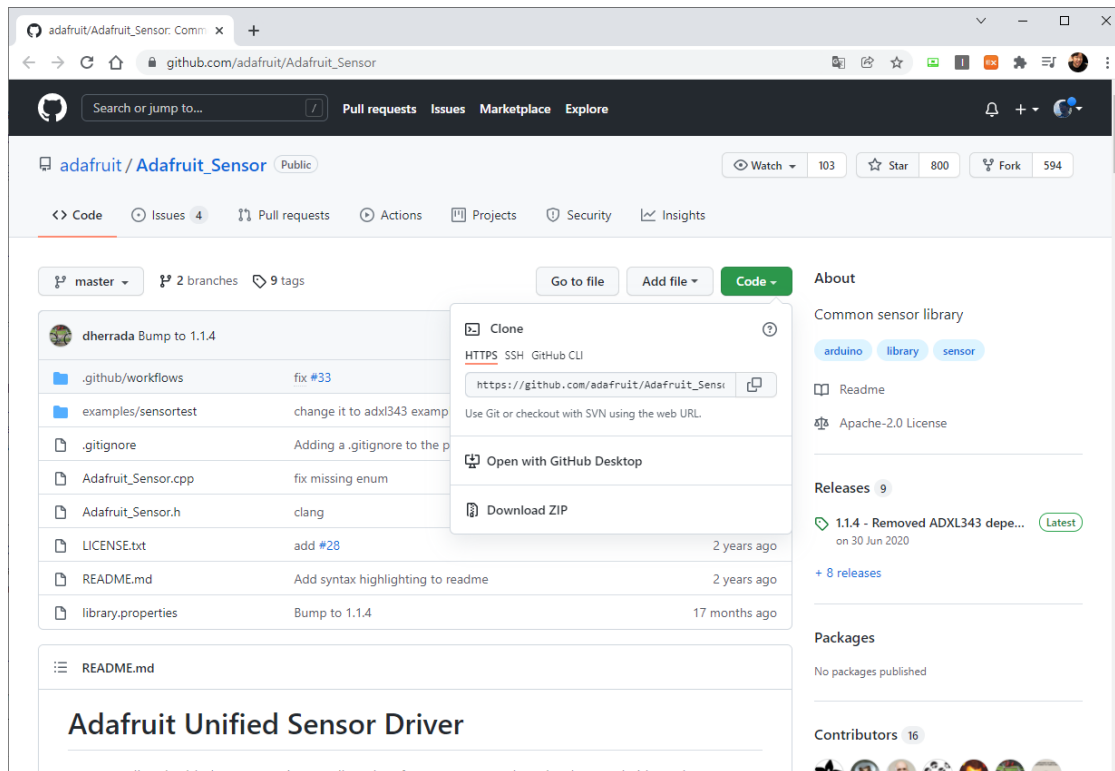
The right sidebar shows the `About` section with the description: "Arduino library for DHT11, DHT22, etc Temperature & Humidity Sensors". It also includes links to `learn.adafruit.com/dht`, `Readme`, `MIT License`, and `Code of conduct`. The `Releases` section shows 22 releases, with the latest being `1.4.3 - Switched from defines to ...` (29 days ago). The `Packages` section shows "No packages published".

DHT11 라이브러리 사용

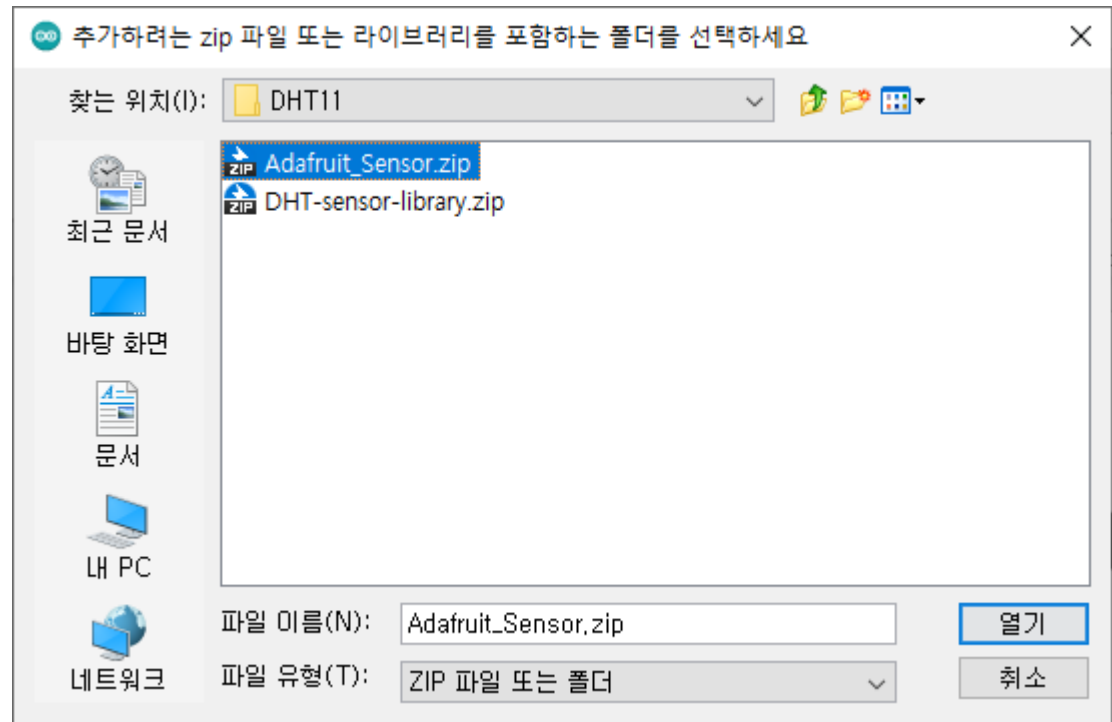
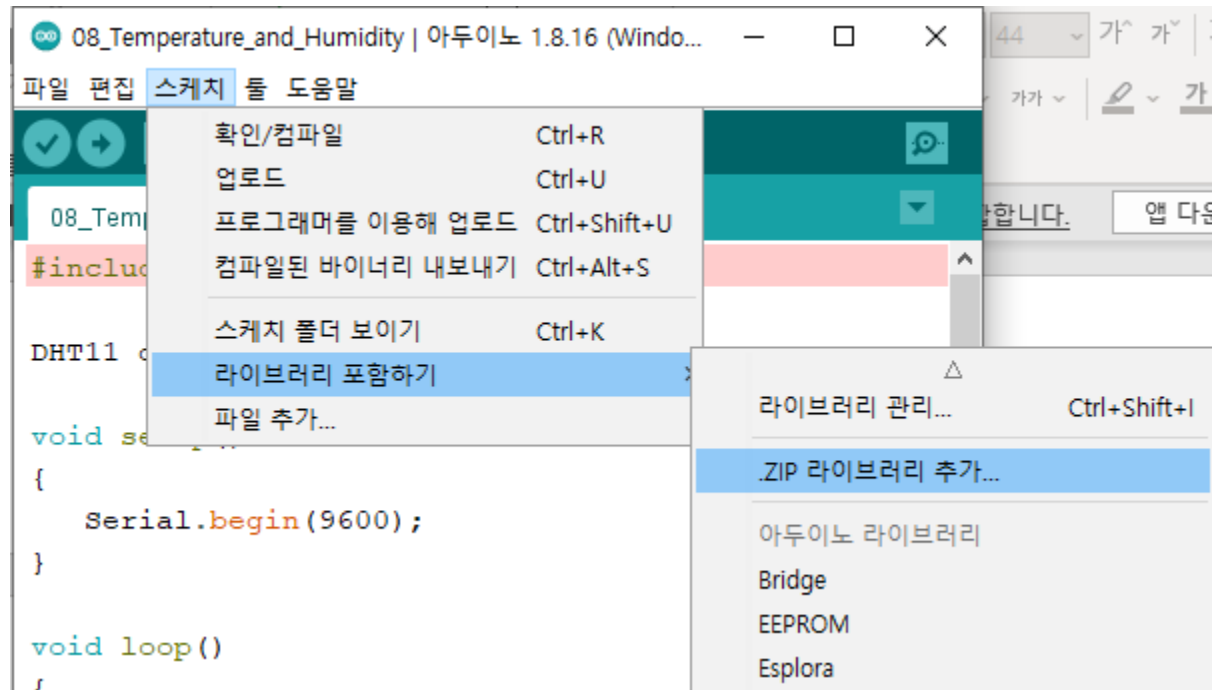


DHT11 라이브러리 사용

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

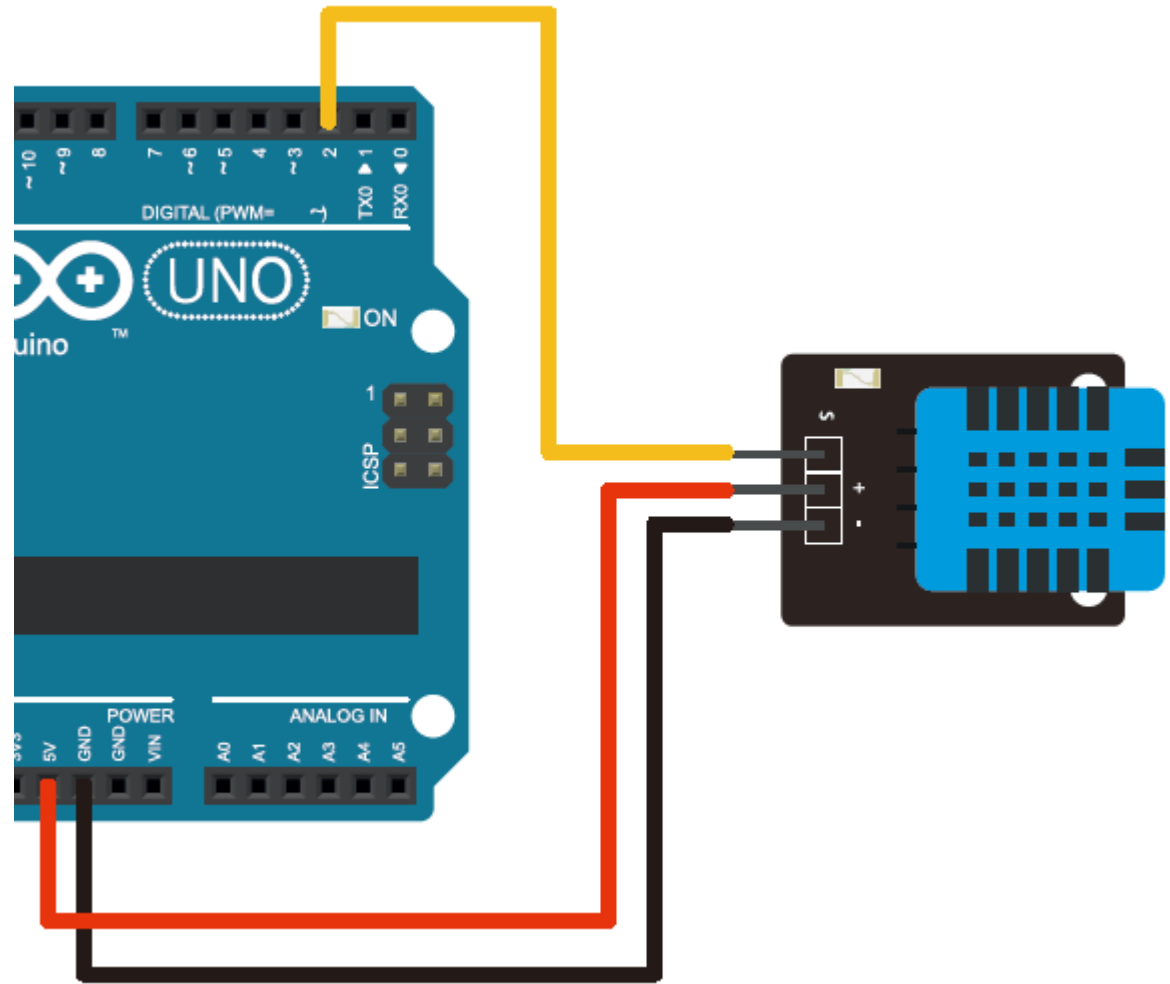


DHT11 라이브러리 사용



DHT11 아두이노 테스트

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



DHT11 아두이노 테스트

예제 : 08_Temperature_and_Humidity

```
#include "DHT.h"

#define DHTPIN 2
#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  Serial.println(F("DHTxx test!"));

  dht.begin();
}

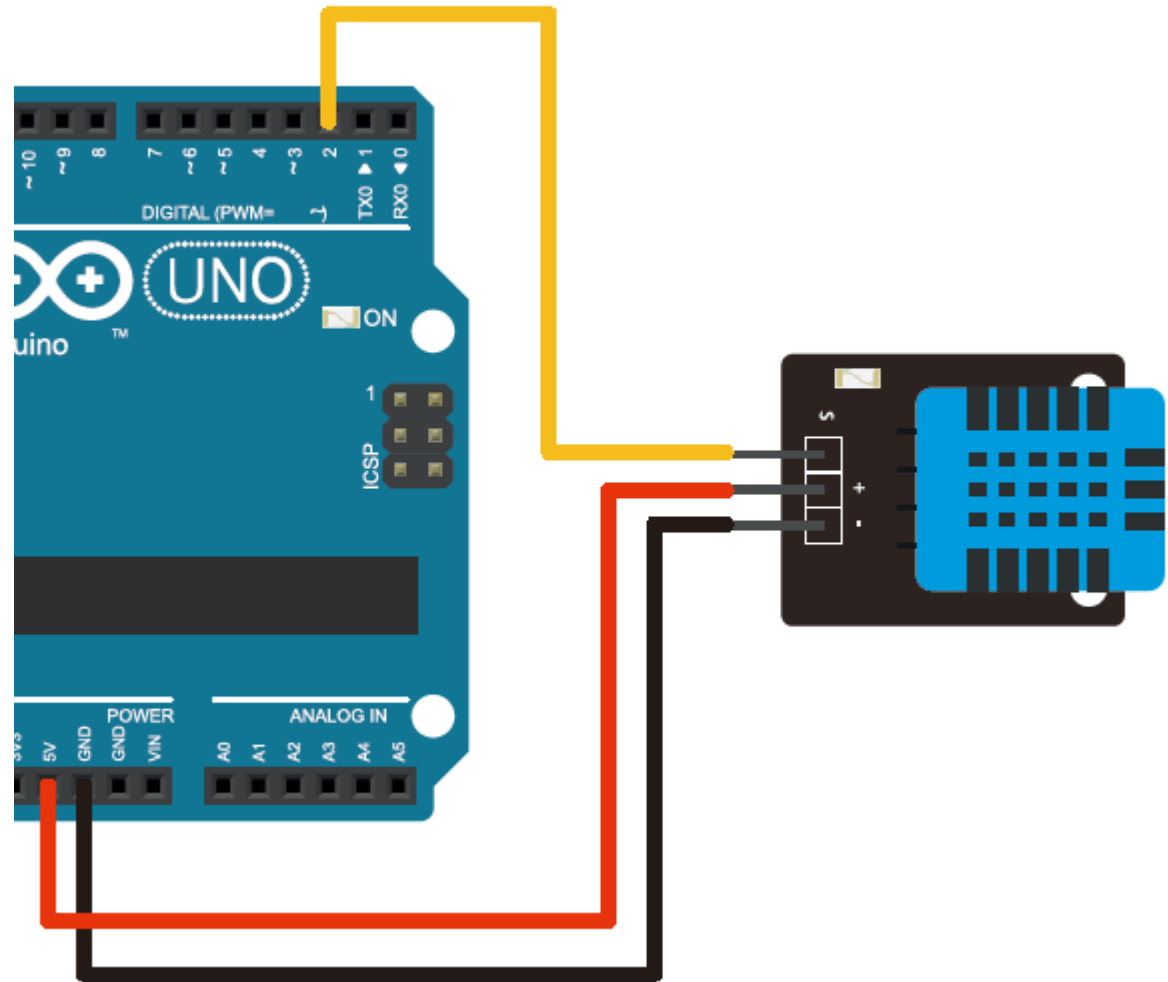
void loop() {
  delay(2000);

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

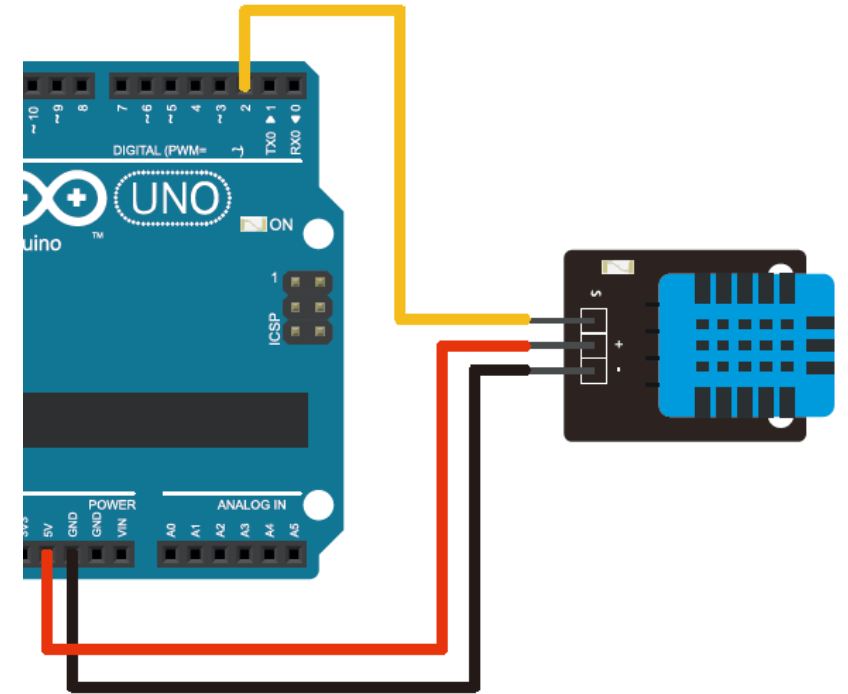
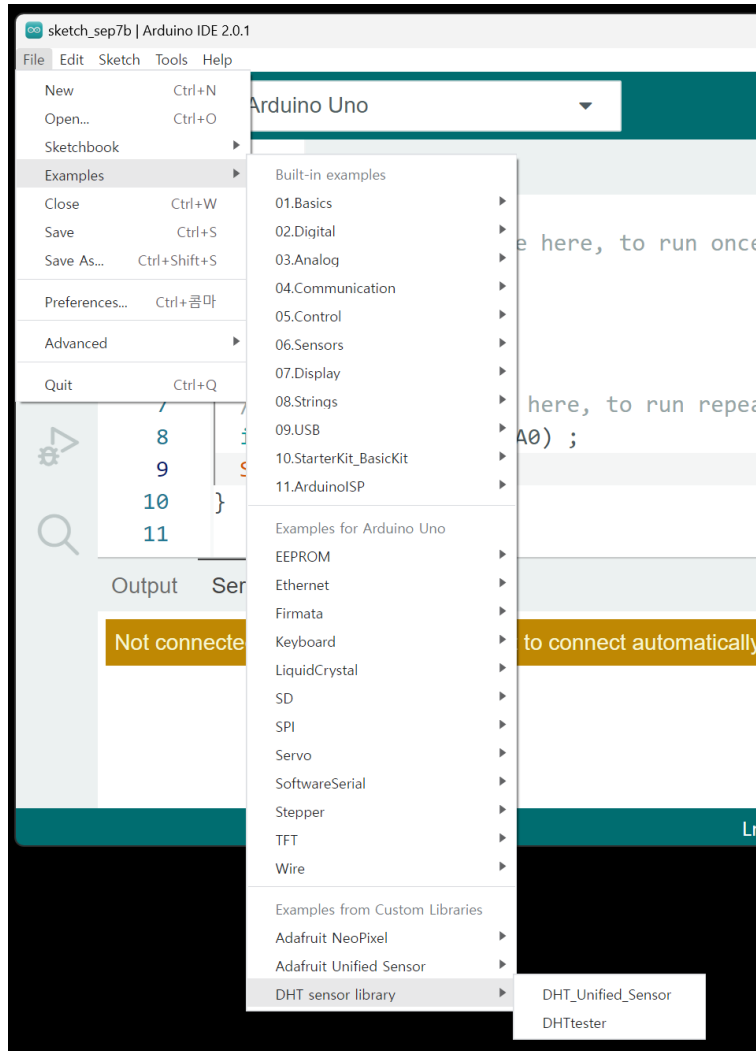
  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  // Compute heat index in Fahrenheit (the default)
  float hif = dht.computeHeatIndex(f, h);
  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("% Temperature: "));
  Serial.print(t);
  Serial.print(F("°C "));
  Serial.print(f);
  Serial.print(F("°F Heat index: "));
  Serial.print(hic);
  Serial.print(F("°C "));
  Serial.print(hif);
  Serial.println(F("°F"));
}
```



DHT11 아두이노 테스트(Example 코드 활용)



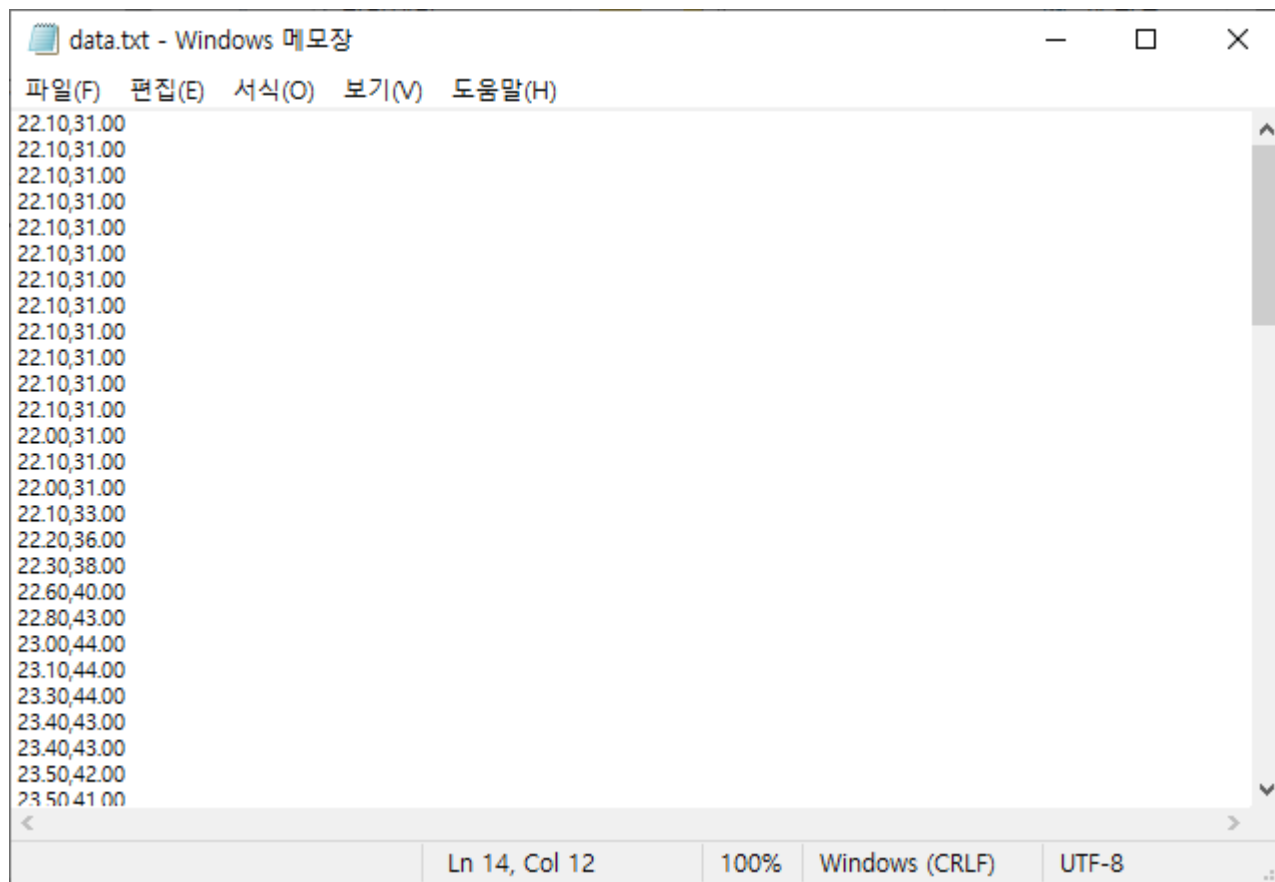
DHT11

시계열 데이터 분석(time series analysis)

- STEP1 : 출력 데이터 정리
 - 온도 RAW데이터, 습도 RAW데이터만 시리얼모니터에 출력

DHT11 시계열 데이터 분석(time series analysis)

- STEP2 : 출력 데이터 저장(data.txt)



```
data.txt - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
22.10,31.00
22.10,31.00
22.10,31.00
22.10,31.00
22.10,31.00
22.10,31.00
22.10,31.00
22.10,31.00
22.10,31.00
22.10,31.00
22.10,31.00
22.10,31.00
22.00,31.00
22.10,31.00
22.00,31.00
22.10,33.00
22.20,36.00
22.30,38.00
22.60,40.00
22.80,43.00
23.00,44.00
23.10,44.00
23.30,44.00
23.40,43.00
23.40,43.00
23.50,42.00
23.50,41.00
Ln 14, Col 12 100% Windows (CRLF) UTF-8
```

DHT11 시계열 데이터 분석(time series analysis)

- STEP3 : 출력 데이터를 엑셀에서 읽기 & 그래프 분석

