

사물인터넷 (Internet of Things)

김태운

목차

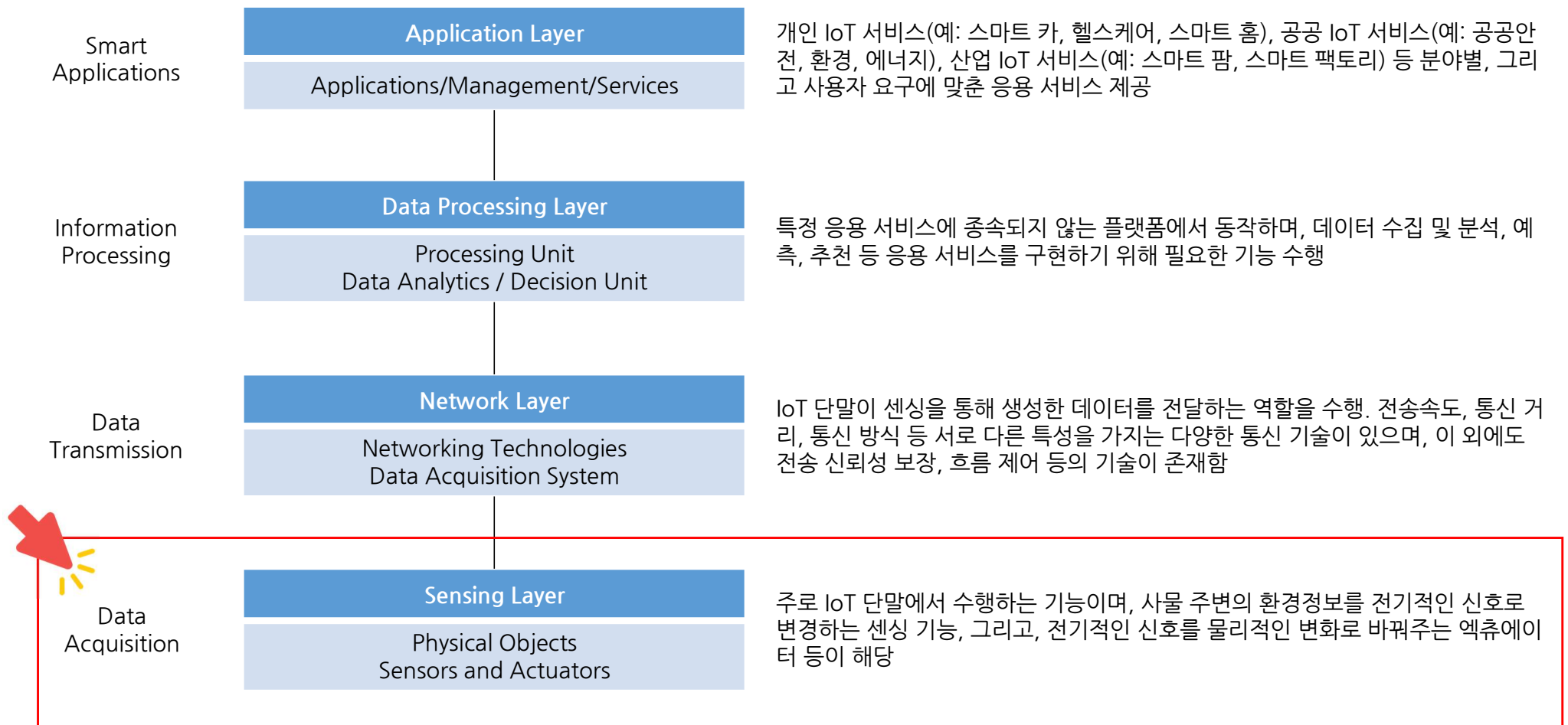
- Sensing Layer: 센서 및 센싱/분석기술
 - IoT 디바이스
 - 센서 및 액추에이터
 - 데이터 정제

Sensing Layer: IoT 디바이스

Sensing Layer: IoT 디바이스

IoT 디바이스는 Sensing Layer에 해당함

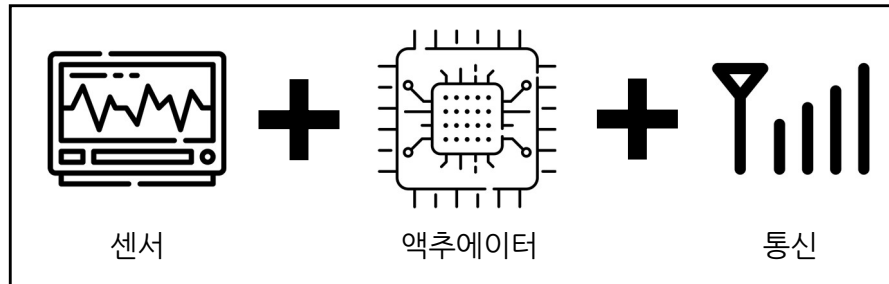
- 참고: 강의 2, 기능적 측면에서의 IoT 시스템 4-Layer 아키텍처



Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스

- 인터넷을 통해 연결되고 센싱, 통신 등이 가능한 하나의 기기
- 상황을 인지할 수 있는 센서, 간단한 기능을 수행하는 경량의 프로세서 및 소프트웨어, 수신 받은 명령을 물리적으로 실행하는 액추에이터가 내장됨
- 일반적인 IoT 디바이스 구성



- 센서: 주변의 환경정보를 전기적인 신호로 변환
- 액추에이터: 전기적인 신호를 물리적인 변화로 변환
- 통신: 디바이스간 또는 디바이스와 통신 인프라간 정보를 주고받기 위한 모듈

- 일상생활에서 존재하는 (거의) 모든 기기가 사물인터넷 디바이스가 될 수 있음
 - 아두이노, 라즈베리파이, 젯슨 나노, 구글 코랄 데브 보드 등...
 - 스마트폰, 스마트워치, VR/AR 글래스 등...
 - 냉장고, TV, 조명기기, IP 카메라 등...
- IoT 디바이스의 주요 역할: 주위 상황을 감지한 후, 이를 디지털 데이터로 변환하고, 응용 서비스를 제공하기 위해 네트워크를 통해 클라우드 서버 등으로 데이터를 전송
- 참고
 - “자율형” 디바이스: 사물의 주변 환경을 감지하고 유/무선 통신, 자동 접속, 상호 연동, 자율 판단/행동을 통해 실감/지능/융합형 서비스를 제공할 수 있는 지능형 디바이스를 의미

Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스 개발 환경 구축하기

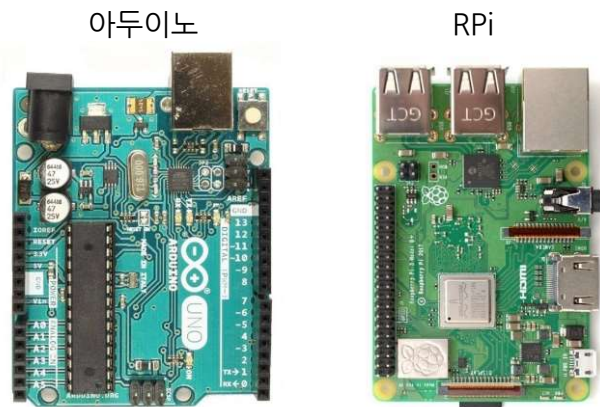
- 사물인터넷 서비스 개발에 널리 활용되는 아두이노(UNO) 및 라즈베리파이(4b) 개발 환경 구축 방법 살펴보기

직접 구축해 보지는 않고, 구축하는 방법만 살펴보겠습니다 ^^



• 참고: 아두이노와 라즈베리파이 차이점

- 아두이노
 - 간단한 연산을 통한 외부 기기 제어에 특화된 마이크로컨트롤러
 - 외부 IDE 프로그램 및 C/C++ 프로그래밍 언어를 통해서 프로그램을 작성하고, 컴파일된 실행 코드를 아두이노로 업로드
 - 이후, 아두이노는 업로드 된 작업만을 반복적으로 실행
- 라즈베리파이 (RPI)
 - 복잡한 연산 및 다양한 개발환경을 제공하는 단일 보드 컴퓨터 (Single Board Computer)
 - ‘라즈비안’이라는 리눅스 기반의 운영체제를 설치한 후, 일반적인 PC처럼 사용하여 센서/액추에이터 연동 가능(다양한 IDE 환경 및 다양한 프로그래밍 언어 사용 가능)



사물인터넷

Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스 개발 환경 구축

- 사물인터넷 서비스 개발에 널리 활용되는 아두이노(UNO) 및 라즈베리파이(4b) 개발 환경 구축 방법 살펴보기

- 참고: 아두이노와 라즈베리파이 차이점

	Raspberry Pi 3 Model B+	Arduino Uno R3
System	Single Board Computer (SBC)	Micro-Controller
Operating System	Run on an operating system	No operating system
Connections	PC based connections like HDMI, USB, ... and GPIOs	Hardware oriented connections like GPIOs, power jack, USB plug
Data Transfer	Files can be transferred any time via FTP, USB or the SD card	Data can only be transferred via the flash of the microcontroller
Program Execution	Multiple programs can run at the same time	Only one program can run at once
Processor	Broadcom BCM2837B0	AVR ATmega328p
Clock Speed	1.4GHz	16MHz
Register Width	64-bit	8-bit
RAM	1 GB	2 kB
GPIO Ports	5V, 3.3V and Ground Digital I/O pins but no analog pin	5V, 3.3V and Ground Digital I/O pins along with analog pins
# GPIO	40	20
I/O Maximal Current	50 mA	50 mA
Power consumption	700 mW	175 mW

Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스 개발 환경 구축

- 사물인터넷 서비스 개발에 널리 활용되는 아두이노(UNO) 및 라즈베리파이(4b) 개발 환경 구축 방법 살펴보기
 - 참고: 아두이노와 라즈베리파이 차이점

Advantages	Highly flexible with different operating systems and able to run multiple applications at the same time.	Perfect to read sensor values and control actors like motors. Easy to program with a lot of tutorials.
Disadvantages	More expensive and can not read analog sensor values	Usage is limited due to lack of computation power and no operating system
Area of Application	Connection of multiple devices and run multiple applications at the same time	Connection of sensors and actors to the internet and enable IoT applications
Price	\$35 to \$52	\$12 to \$18

Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스 개발 환경 구축

- 사물인터넷 서비스 개발에 널리 활용되는 아두이노(UNO) 및 라즈베리파이(4b) 개발 환경 구축 방법 살펴보기

- 참고: 아두이노와 라즈베리파이 차이점

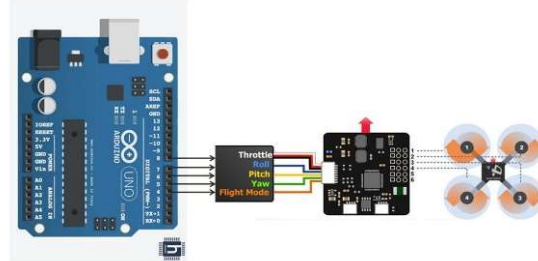
- 아두이노는 사물인터넷 분야에 최적화 된 디바이스로 센싱/액추에이터의 동작을 주로 하며, 간단한 수준의 연산만을 수행함



스마트 홈



스마트 팜



드론 제어

- 라즈베리파이는 고수준의 연산/학습/추론이 필요한 사물인터넷 분야에 활용되며, 그 외에도 다양한 목적으로 활용 가능



AI 자동차



Smart Mirror



비트코인 채굴기



NAS 서버

Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스 개발 환경 구축

- 사물인터넷 서비스 개발에 널리 활용되는 아두이노(UNO) 개발 환경 구축 방법 살펴보기

- 준비물

- 아두이노 UNO
- USB 케이블
- 개발용 PC/노트북



- 개발환경 구축 1

- 아두이노 홈페이지(www.arduino.cc/en/software)에 접속하여 아두이노 IDE 프로그램 다운로드 및 설치



Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.


Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is **hosted by GitHub**. See the instructions for **building the code**. Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file

Windows app Win 8.1 or 10 

Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits

Mac OS X 10.10 or newer

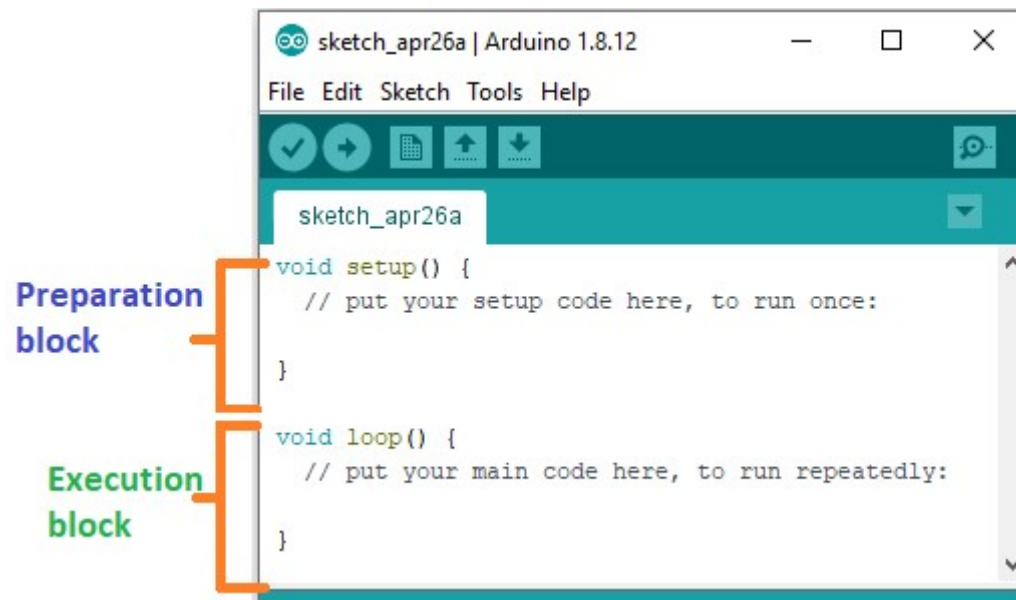
[Release Notes](#)

[Checksums \(sha512\)](#)

Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스 개발 환경 구축

- 사물인터넷 서비스 개발에 널리 활용되는 아두이노(UNO) 개발 환경 구축 방법 살펴보기
 - 개발환경 구축 2
 - USB 케이블로 아두이노와 노트북을 연결한 후, 아두이노 IDE 프로그램 실행



- 아두이노 프로그램은 기본적으로 setup 함수와 loop 함수로 구성
 - setup 함수: 아두이노가 시작될 때 한 번 호출되며, 초기화와 관련된 작업을 수행하는 코드를 입력하는 부분
 - loop 함수: setup 함수 실행 이후, 반복적으로 실행할 주요 로직을 코딩하는 부분
 - 프로그래밍 언어는 C/C++ 사용

Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스 개발 환경 구축

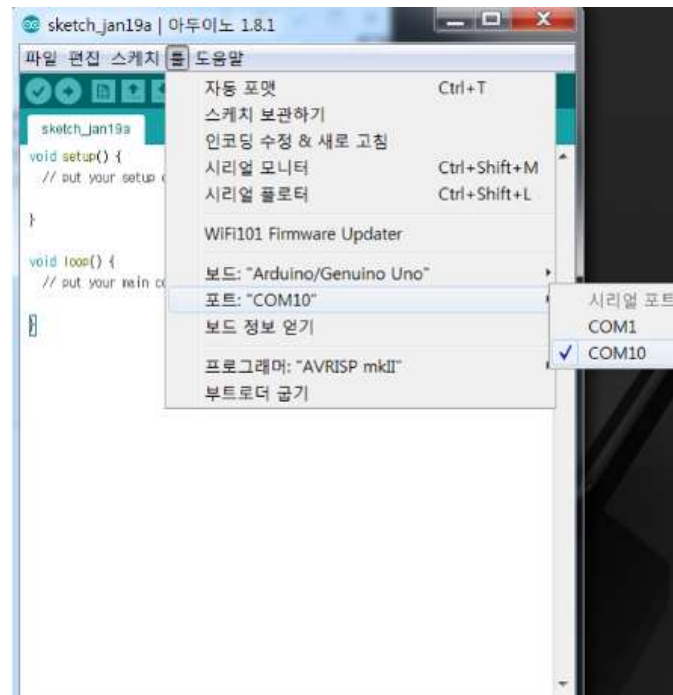
- 사물인터넷 서비스 개발에 널리 활용되는 아두이노(UNO) 개발 환경 구축 방법 살펴보기

- 개발환경 구축 3

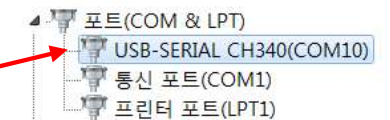
- 개발에 사용하는 아두이노 보드 종류 설정 및 시리얼 연결 포트 설정



보드 종류 설정



연결된 포트 설정

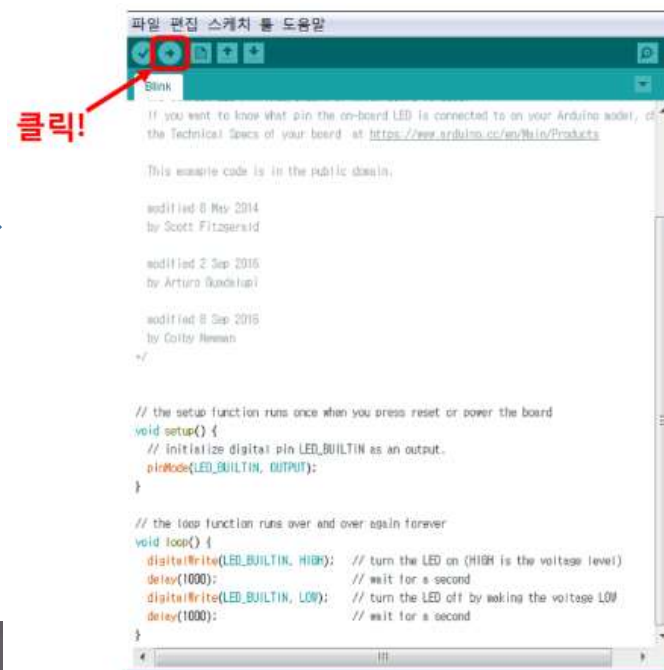
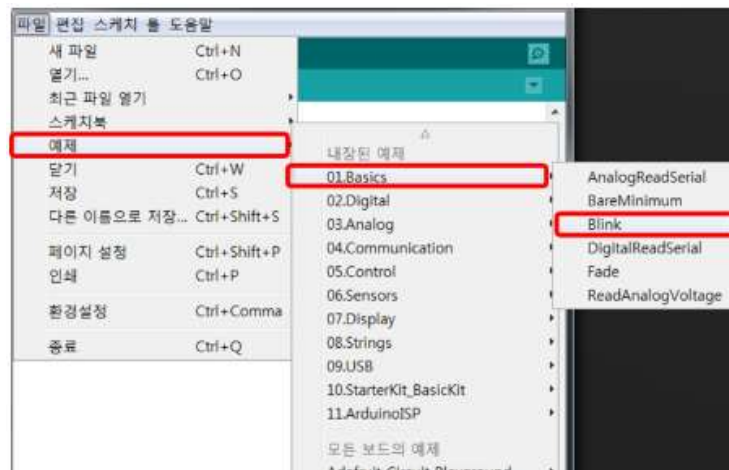


포트 번호는 [제어판 > 장치관리자 > 포트] 에서 확인 가능

Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스 개발 환경 구축

- 사물인터넷 서비스 개발에 널리 활용되는 아두이노(UNO) 개발 환경 구축 방법 살펴보기
 - 개발환경 구축 4: 예제 코드를 실행하여 정상 동작 검증
 - 아두이노 IDE 프로그램에서 기본으로 제공하는 예제 코드를 실행하여 검증



실행 결과



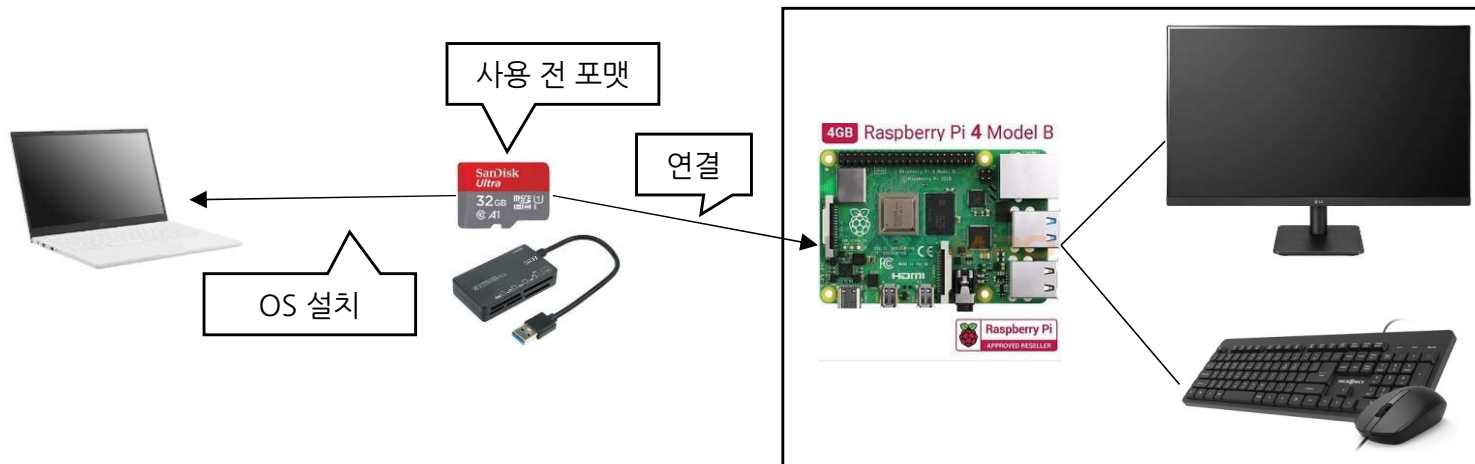
Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스 개발 환경 구축

• 사물인터넷 서비스 개발에 널리 활용되는 라즈베리파이 개발 환경 구축 방법 살펴보기

• 준비물

- 라즈베리파이, 모니터, 키보드, 마우스
- Micro SD 카드(라즈베리파이의 하드디스크 역할) 및 멀티카드 리더기(mSD 카드를 PC에 연결하는 용도)
- PC/노트북(mSD카드에 라즈비안 OS 설치용)



일반 PC와 동일한 구성 및 방식으로 사용

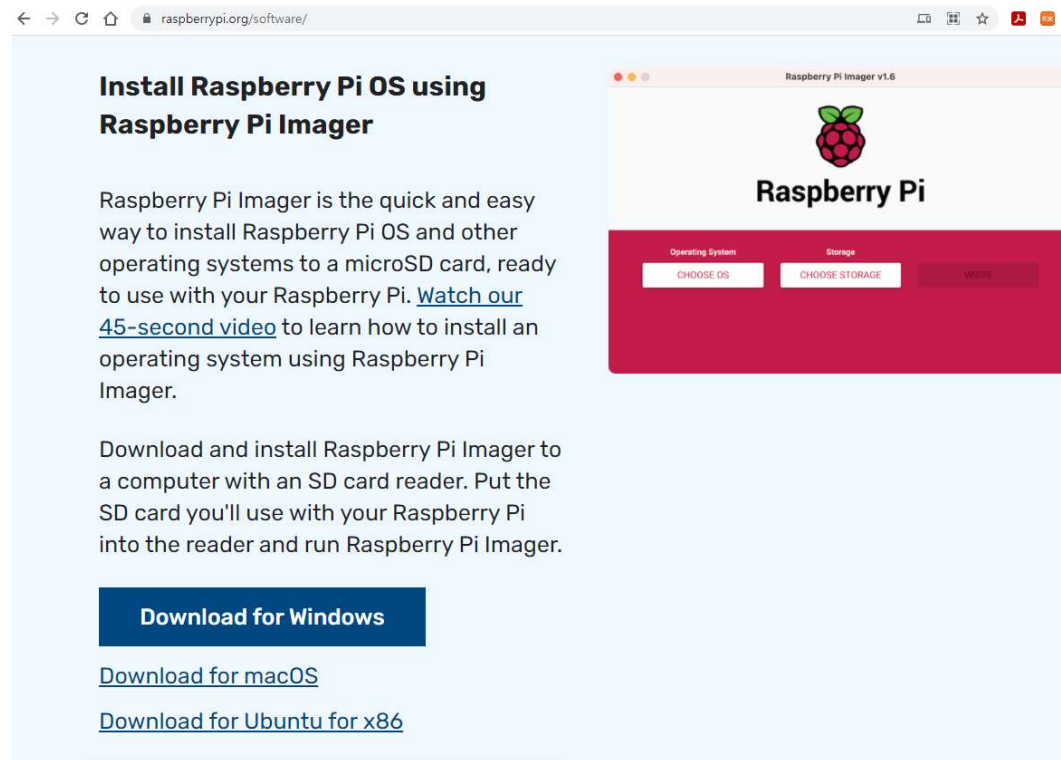
Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스 개발 환경 구축

- 사물인터넷 서비스 개발에 널리 활용되는 라즈베리파이 개발 환경 구축 방법 살펴보기

- 개발환경 구축 1

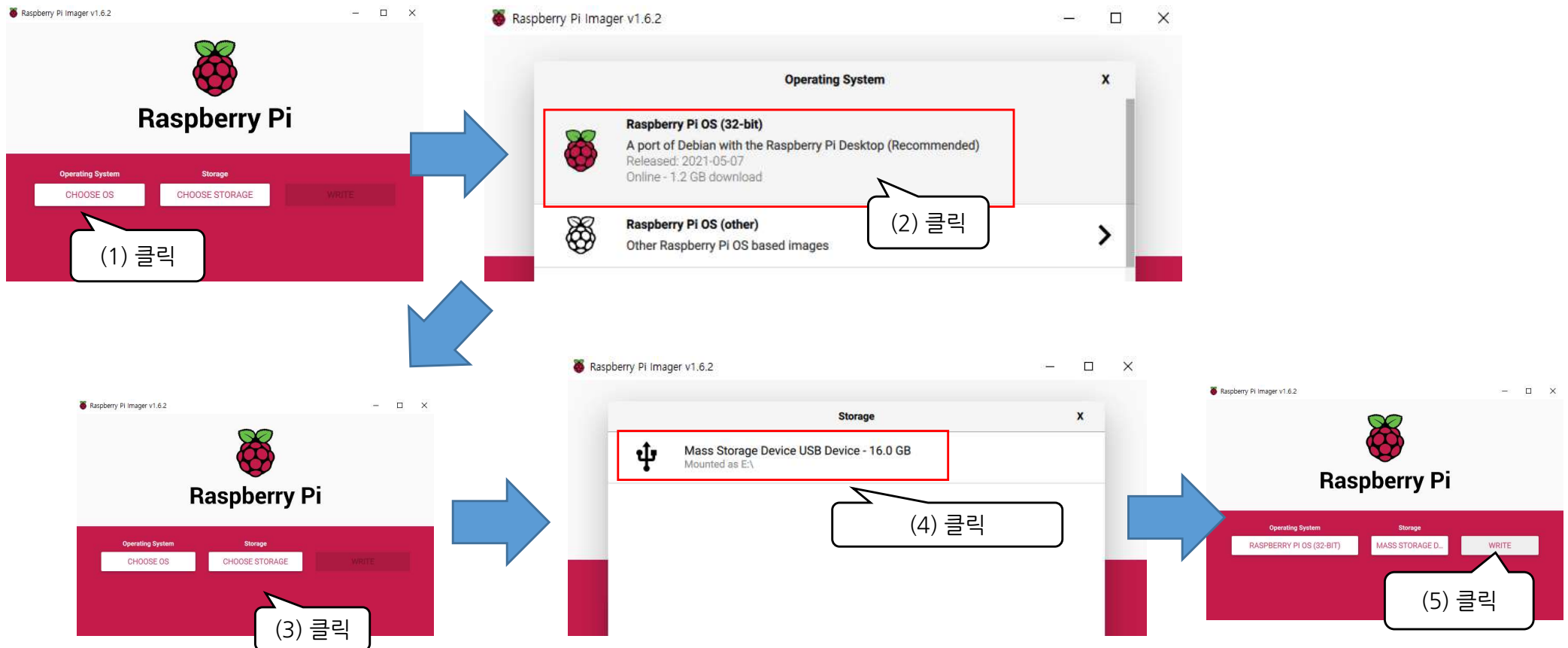
- 라즈베리파이 공식 홈페이지에서 전용 OS(라즈비안) 설치 프로그램 다운로드 및 설치
 - URL: <https://www.raspberrypi.org/software/>
 - Raspberry Pi Imager: 라즈비안 OS 설치용 프로그램



Sensing Layer: IoT 디바이스

■ 사물인터넷 디바이스 개발 환경 구축

- 사물인터넷 서비스 개발에 널리 활용되는 라즈베리파이 개발 환경 구축 방법 살펴보기
 - 개발환경 구축 2
 - 멀티카드 리더기를 사용하여 mSD 카드를 PC/노트북에 연결하고 Raspberry Pi Imager 프로그램 실행
 - mSD 카드에 라즈비안 OS 설치



Sensing Layer: IoT 디바이스

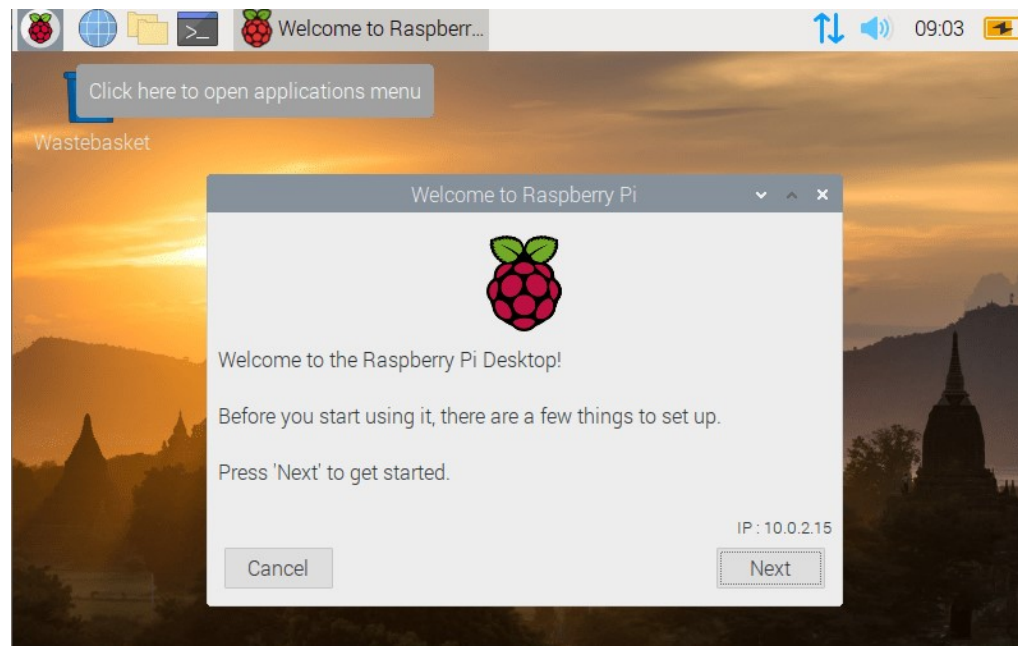
■ 사물인터넷 디바이스 개발 환경 구축

• 사물인터넷 서비스 개발에 널리 활용되는 라즈베리파이 개발 환경 구축 방법 살펴보기

• 개발환경 구축 3

- 라즈비안 OS가 설치된 mSD카드를 라즈베리파이 mSD 슬롯에 장착
- 라즈베리파이에 전원 공급 및 실행화면 확인

Wi-Fi, 블루투스 모듈이 기본 장착되어 있음



준비 완료!



- 일반 리눅스를 사용하는 것과 마찬가지로, 개발자가 원하는 프로그래밍 언어와 컴파일러 설치하고, 필요시 IDE를 설치하여 개발 환경 구축을 완료함

Sensing Layer: 센서 및 액추에이터

Sensing Layer: 센서 및 액추에이터

IoT 센서

- 스마트폰, 비디오 카메라 등 사물인터넷 기기는 다양한 센서를 탑재하고 있음
 - 예: 이미지 센서, 온도/습도 센서, 열화상 이미지 센서, 모션감지 센서, 가스 센서, 공기질 센서, 유량 센서, 압력 센서 등
- IoT 센싱의 기본은 인간의 오감(시각, 청각, 후각, 미각, 촉각)을 대신하여 주변을 측정하는 것
- IoT 센서의 종류 및 사용 예

종류	유형(type)	사용 목적
시각 센서	IR motion sensor	장애물 탐지
	ultrasonic distance sensor	센서로부터 물체의 거리 측정
	camera sensor	센서로부터 물체의 거리 측정
	speed sensor	도플러효과를 이용한 거리 측정
	light sensor	빛의 강도 측정
	position sensor	지자기장 센서와 가속도계를 함께 사용하여 자북극에 대한 상대적인 기기 위치를
	tilt sensor	물체의 기울기 측정
	GPS	위치(위도, 경도) 측정
청각 센서	sound sensor	마이크 등 음성 신호 인식 (소리 파형을 전기신호로 변환)
후각 센서	smoke detection sensor	연기 탐지
미각 센서	biosensor	다양한 생물학적 신호 측정
촉각/접촉 센서	temperature/humidity sensor	온도, 습도 측정
	pressure sensor	압력 측정
	pushbutton sensor	버튼을 누르는 힘을 측정
	weight sensor	무게 측정
	fingerprint sensor	지문 인식
	rain sensor	우량 측정

Sensing Layer: 센서 및 센싱 기술

IoT 센서

- 여러 분야별 디바이스나 시설 특성에 따라 탑재되는 다양한 종류의 센서 유형

구분	센서 유형								
	관성 센서	자기센서	압력센서	온도센서	음향센서	화학센서	습도 센서	질량유량 센서	기타 센서
자동차	○	○	○	○		○	○	○	○
도시 (환경관리)				○	○	○	○		○
온도조절 장치			○				○		○
완구	○	○	○						
흡연/탄소센 서(소비자용)				○		○	○	○	○
헬스케어 모니터링 (소비자용)	○	○	○	○		○	○		
스마트워치	○	○							
스마트의류	○	○	○	○	○		○		○
스마트 전자 소켓 및 어댑 터		○		○			○		○

Sensing Layer: 센서 및 센싱 기술

■ IoT 센서

- 센서는 이미지, 동작, 소리, 빛, 열, 가스, 온도, 습도 등 주변의 물리/화학/생물학적 정보를 감지하여 전기적 신호로 변환하는 모든 장치를 의미
- 데이터를 센싱하고 이를 신호 처리한 후, 인터페이스를 통해 전달하는 기능을 수행
- 최근에는 미세전자제어기술(MEMS), 여러 반도체 부품이 하나로 집적되는 반도체 SoC기술, 임베디드 소프트웨어 기술의 발전으로 과거보다 지능화된 스마트 센서가 널리 활용됨
- IoT기술 활성화를 위해 필요한 센서의 중요한 요소
 - 저가격
 - 저전력
 - 소형화
 - 무선 연결성(wireless connectivity)

Sensing Layer: 센서 및 센싱 기술

■ IoT 액추에이터

- 어떠한 (물리적인) 동작을 일으키는 디바이스를 의미
- (센서로부터 얻은 정보를 기반으로, 상황에 맞는 의사결정에 따라) 행동을 취하는 역할을 수행
- 특히, IoT에서는 AI 등에 기반한 플랫폼에서의 데이터 분석 및 예측에 기반하여 행동을 취함
- 액추에이터의 동작을 위해서는 제어 신호와 에너지가 필요 하며, 보통은 전기 에너지를 물리적 움직임으로 변환함
- 액추에이터의 분류
 - 기계적 액추에이터(mechanical actuator)
 - 기계적 운동을 주로 하는 디바이스로서, 스크류(screw)나 체인(chain)을 사용하여 회전운동(rotaty motion)을 선형운동(linear motion)으로 변환함
 - 전기적 액추에이터(electrical actuator)
 - 전류가 흐르는 도체가 자기장 속에서 받는 힘을 이용하여 전기에너지를 기계적에너지로 바꾸는 장치
 - 일반적으로 전기모터(electric motor)를 말하며, 부가되는 전원의 종류에 따라 직류전동기와 교류 전동기로 분류
 - 압력 액추에이터(pressure actuator)
 - 압력을 가하는 매체에 따라 수압과 공기압으로 나뉨
 - 수압 액추에이터: 파스칼의 법칙을 이용하여 밀폐된 공간(실린더)에 유체가 들어있을 때 그 유체에 압력을 가하면 그 압력이 공간 내에서 작용하는 원리를 이용한 것
 - 공기압 액추에이터: 액체 대신 압축공기를 사용한 것

Sensing Layer: 센서 및 센싱 기술

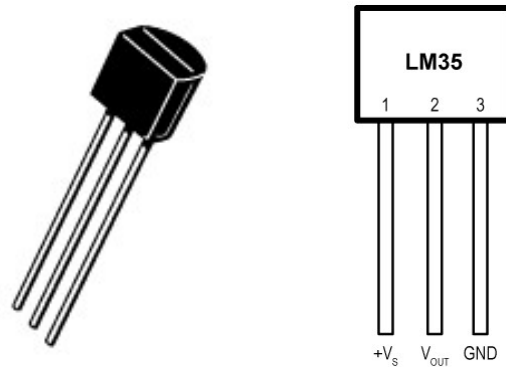
■ IoT 액추에이터

- 주변에서 쉽게 볼 수 있는 액추에이터의 예로는 스피커, 히터, 냉각기 등이 있음
- 액추에이터의 종류별 사용 예

유형	사용 예	에너지 변환 방법
sound	speaker	전기에너지 => 음파
relay	electromechanical switch	전기에너지 => 기계적 운동
valve	액체의 흐름 제어	
mechanical	기어	회전운동 => 선형운동
thermal	히터	
electrical	모터	전기에너지 => 기계적 운동
hydraulic	산업용 공정 제어	기계에너지 => 선형/회전운동
pneumatic	자동화 제어	압력 => 힘

Sensing Layer: 센서 및 센싱 기술

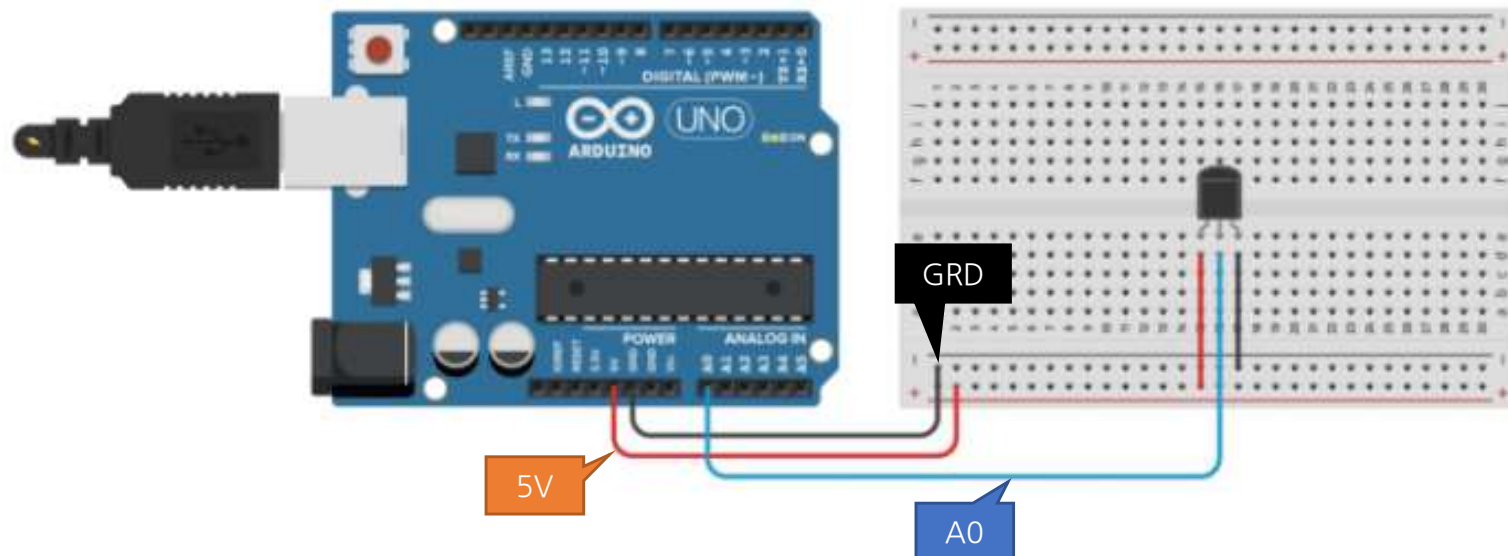
- 아두이노를 사용한 IoT 센서 활용 예: 온도 센서
 - 온도 센서: LM35 analog temperature sensor (TO-92)



Name	Pin	Description
+V _S	1	Positive power supply pin (4 – 30 V)
V _{OUT}	2	Temperature sensor analog output
GND	3	Device ground pin, connect to power supply negative terminal

온도 값이 리턴 되는 핀

- 온도 센서를 아두이노 보드에 연결하기(wiring)



Sensing Layer: 센서 및 센싱 기술

- 아두이노를 사용한 IoT 센서 활용 예: 온도 센서
 - 아두이노 IDE 코딩

```
// Define to which pin of the Arduino the output of the LM35 is connected:  
#define sensorPin A0
```

```
void setup() {
```

```
    // Begin serial communication at a baud rate of 9600:  
    Serial.begin(9600);  
}
```

아두이노에 연결된 PC/노트북과의 시
리얼 통신 시작

```
void loop() {
```

```
    // Get a reading from the temperature sensor:  
    int reading = analogRead(sensorPin);
```

LM35 센서로 부터 값 읽어오기

```
    // Convert the reading into voltage:  
    float voltage = reading * (5000 / 1024.0);
```

```
    // Convert the voltage into the temperature in degree Celsius:  
    float temperature = voltage / 10;
```

LM35 센서 리턴 값을 온도로 변환하는
작업(참고: LM35 스펙문서)

```
    // Print the temperature in the Serial Monitor:  
    Serial.print(temperature);  
    Serial.print(" \xC2\xB0"); // shows degree symbol  
    Serial.println("C");
```

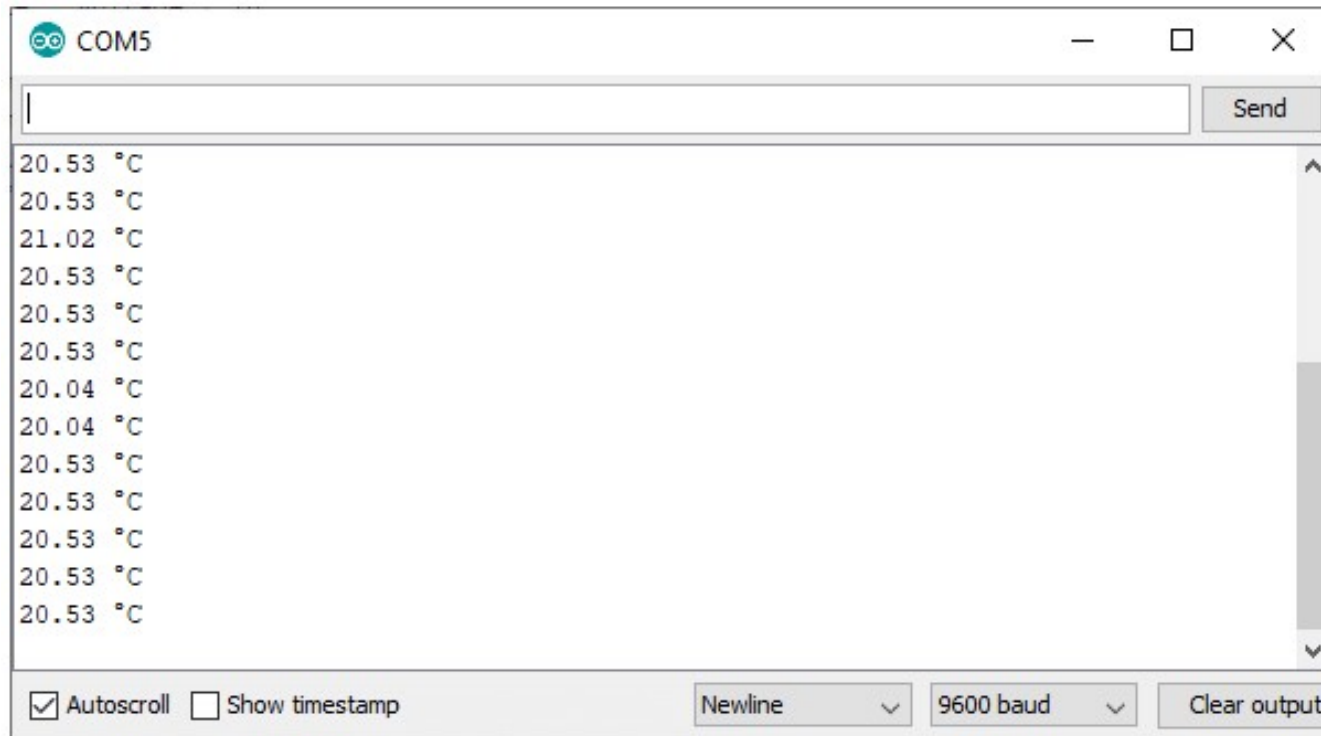
아두이노에 연결된 PC/노트북에
온도 값을 전달(시리얼 통신)

```
    delay(1000); // wait a second between readings  
}
```

??

Sensing Layer: 센서 및 센싱 기술

- 아두이노를 사용한 IoT 센서 활용 예: 온도 센서
 - 아두이노에 연결된 PC/노트북에서 결과값 확인

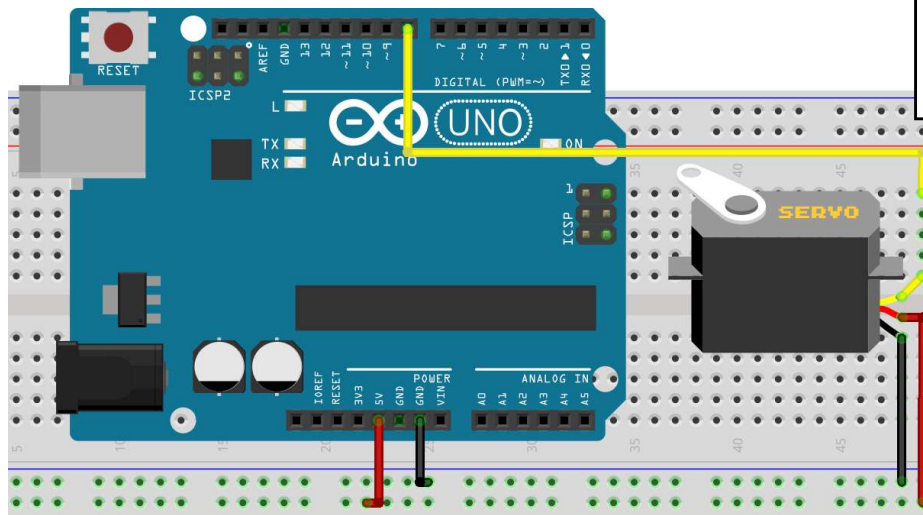


Sensing Layer: 센서 및 센싱 기술

- 아두이노를 사용한 IoT 액추에이터 활용 예: 서보 모터
 - 모터 디바이스: SG90 Servo Motor



- 서보 모터를 아두이노 보드에 연결하기(wiring)



Sensing Layer: 센서 및 센싱 기술

- 아두이노를 사용한 IoT 액추에이터 활용 예: 서보 모터
 - 아두이노 IDE 코딩

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

```
Servo servo;  
int angle = 0;
```

```
void setup() {
```

```
    servo.attach(8);  
    servo.write(angle);  
}
```

모터를 아두이노 보드 8번 핀에 연결
하고 0도에 맞추어 모터를 이동(초기화)

```
void loop()
```

```
{
```

```
    // scan from 0 to 180 degrees
```

```
    for(angle = 0; angle <= 180; angle++)
```

```
    {
```

```
        servo.write(angle);
```

```
        delay(15);
```

```
    }
```

```
    // now scan back from 180 to 0 degrees
```

```
    for(angle = 180; angle >= 0; angle--)
```

```
    {
```

```
        servo.write(angle);
```

```
        delay(15);
```

```
    }
```

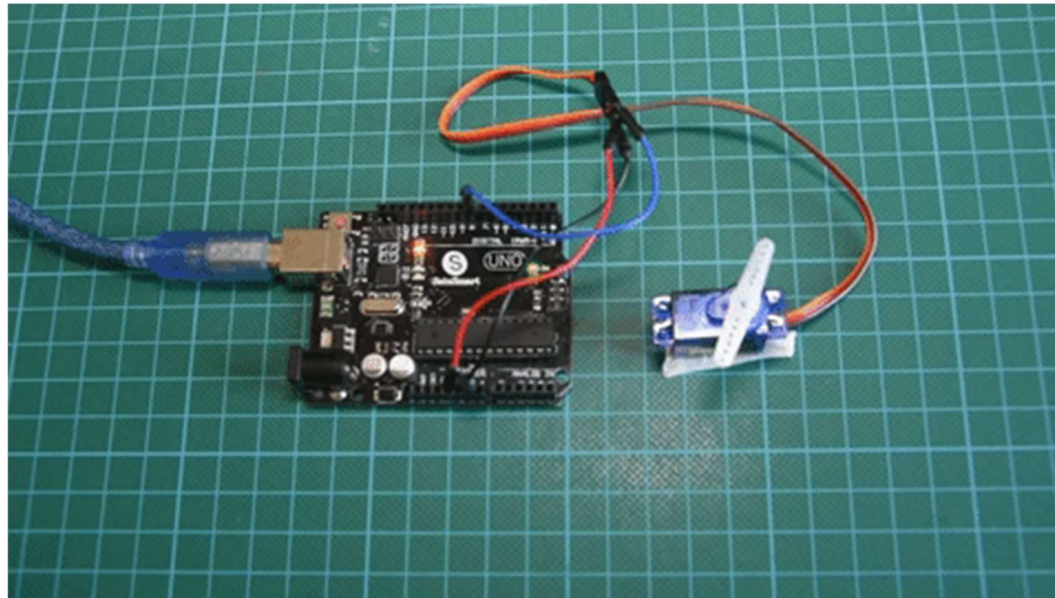
```
}
```

0도~180도까지 1도씩 증가하는 방향
으로 모터를 제어

180도~0도까지 1도씩 감소하는 방향
으로 모터를 제어

Sensing Layer: 센서 및 센싱 기술

- 아두이노를 사용한 IoT 액추에이터 활용 예: 서보 모터
 - 서보 모터 제어 결과 확인



gif

Sensing Layer: 센서 및 센싱 기술

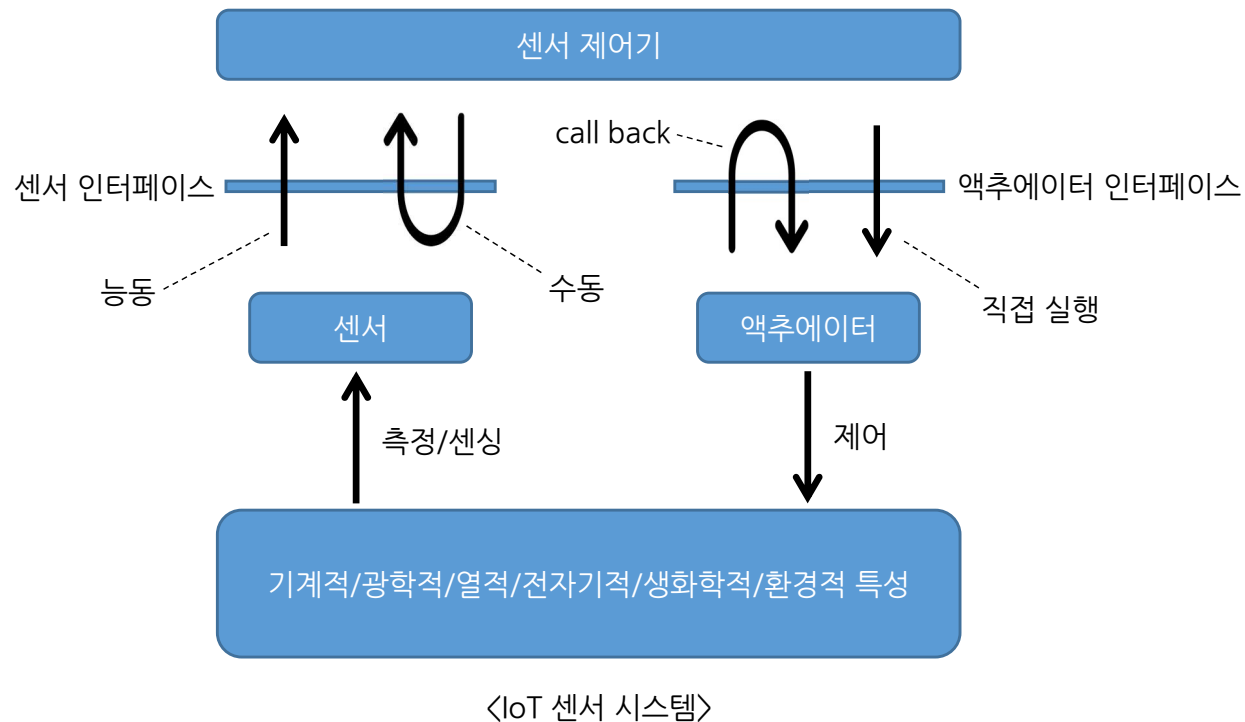
- 사물인터넷 기기 개발 사례
 - 삼성 Ballie: 사물인터넷 로봇



Sensing Layer: 센서 및 센싱 기술

IoT 센서 시스템과 요소 기술

- 일반적으로 IoT에서 센서 디바이스는 주변 환경에 대한 정보를 센싱하는 기능만 가지는 단품으로 구현되기 보다는, 다양한 기능이 결합된 하나의 시스템으로 구현됨
- 이것을 IoT 센서 시스템이라고 하며, 아래와 같은 구조를 가짐

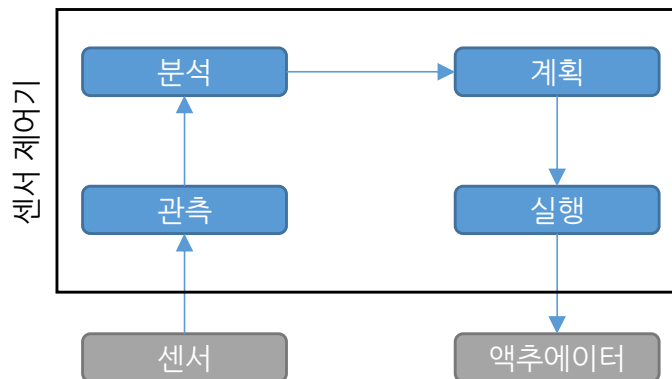
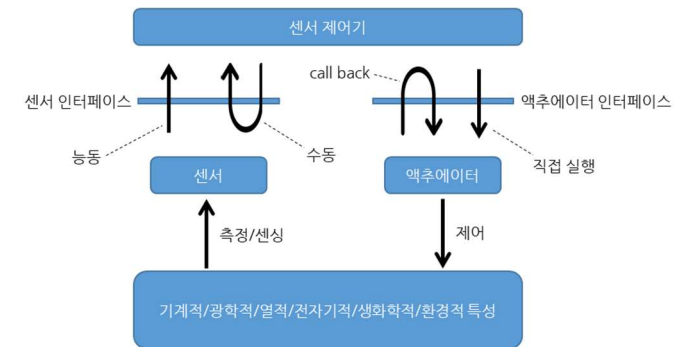


Sensing Layer: 센서 및 센싱 기술

IoT 센서 시스템과 요소 기술

IoT 센서 시스템은 센서, 액추에이터, 센서 제어기로 구성됨

- 센서
 - 기계적/광학적/열적/전자기적/생화학적/환경적 특성을 인식/센싱하는 디바이스
- 액추에이터
 - 어떠한 동작을 일으키는 디바이스로, 기계적/전기적/압력 액추에이터 등이 있음
- 센서 제어기
 - IoT 센서 제어기는 센서로부터 전달되는 데이터를 관측, 분석하고, 계획/실행 로직을 통해 액추에이터에게 명령을 내리는 역할을 수행
 - 이를 위하여 IoT 센서 제어기는 소형 기억장치와 프로세서를 내장하고 있으며 외부 네트워크와 접속할 수 있도록 표준화된 네트워크 접속 모듈을 탑재
 - 센서 제어기는 센서와 일체형 또는 분리형으로 구현할 수 있음
 - 일체형: 센서 시스템 내에 데이터를 관측, 분석, 계획 및 실행하는 모든 기능을 장착해야 하므로 센서의 크기 및 시스템 부하가 크고 그에 따라서 가격도 증가하게 되며 전력 소모도 많음. 단, 일체형의 경우에는 센싱과 제어 기능이 함께 구현되므로 데이터 처리속도가 빠르고(지연이 적고), 장거리 통신을 할 필요가 없으므로 전송 오류나 보안의 위험이 적다는 것이 장점
 - 분리형: 제어기의 역할을 플랫폼(클라우드 또는 엣지)에 맡기는 등 센서 자체에 제어 기능을 구현할 필요가 없음. 단, 분리형은 수집된 데이터를 플랫폼의 제어기까지 보내서 액추에이터를 구동하는 데 지연도 발생하고 전송 오류나 보안의 우려가 클 수 있음
 - 센서 제어기의 구조

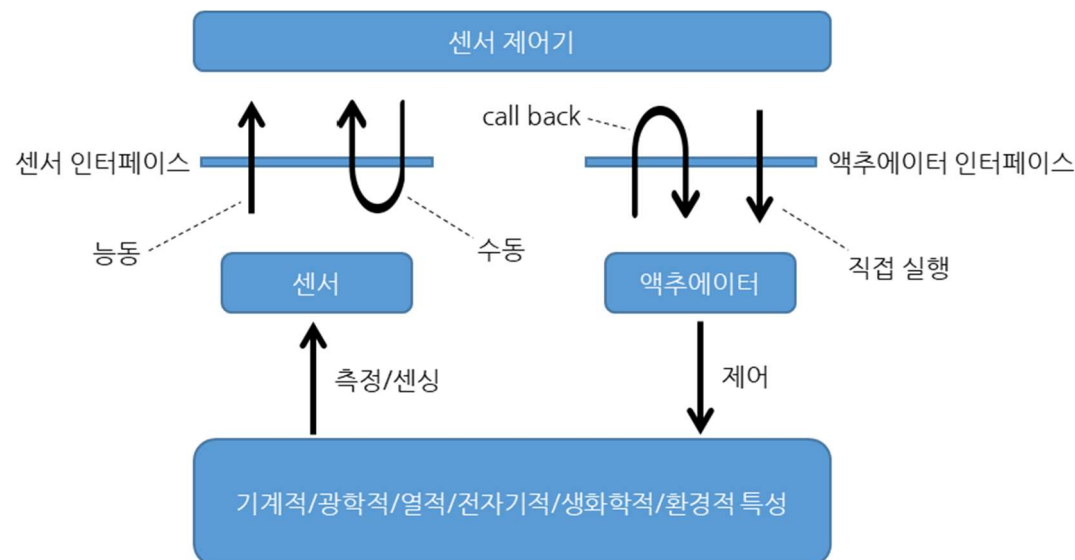


Sensing Layer: 센서 및 센싱 기술

IoT 센서 시스템과 요소 기술

IoT 센서 시스템의 동작 방식 및 순서

1. 현장에서 발생하는 다양한 현상을 센서가 측정함
2. 센서가 측정한 정보는 센서 인터페이스를 통하여 제어기로 전달되며, 전달 방식은 두 가지 경우로 나눌 수 있음: 센서가 데이터를 능동적으로 전송하는 경우(=능동 모드) 및 제어기가 주도적으로 센서로부터 데이터를 가져오는 경우(= 수동 모드)
3. 이어서, 제어기에서는 관측된 데이터를 분석하고 제어 계획을 수립함
4. 마지막으로 제어의 실행 단계에서는 제어기가 액추에이터 인터페이스를 통하여 직접 액추에이터에게 제어 명령을 실행하는 경우(= 직접 실행)와 액추에이터가 제어기에게 요구를 하는 경우에 한해서 실행하는 경우(= call back)가 있음

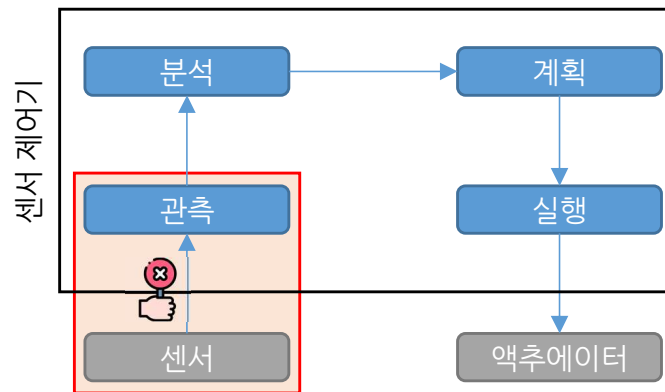


예) Actuator Device의 가용성 (availability)이 확보된 상황에서 Device가 직접 제어 명령을 읽어 가는 경우

Sensing Layer: 데이터 정제

Sensing Layer: 데이터 정제

- 데이터 정제(Data cleaning/cleansing/scrubbing)
 - 데이터 정제란 data set 내의 incorrect/incomplete/duplicate 인 값을 처리하는 것(= 수정 or 삭제)
 - 데이터 정제를 거치지 않은 raw data를 dirty data 라고 함
 - 센서를 통해서 관측한 값에 오류가 있으면 의사 결정 과정에서 부정확한 결과를 초래함



- 데이터 정제의 목적: 수집한 데이터의 품질 개선을 통해 일관적이고, 안정적인 정보를 전달하여 사물인터넷 서비스를 고도화 하는 것
- 센싱 데이터에 오류가 있는 경우 분석 결과를 신뢰할 수 없고, 잘못된 판단을 내린 경우 그 원인을 파악하기 어려움
 - 오류가 있는 데이터로 학습한 로직은 신뢰성이 떨어짐
 - 의사 결정에 오류가 있는 경우, 센싱 데이터 오류인지 의사결정 로직 오류인지 구분이 어려움

Sensing Layer: 데이터 정제

- 데이터 정제(Data cleaning/cleansing/scrubbing)
 - Dirty data 유형

유형	예
잘못된 데이터(wrong data)	나이: “25”가 “225”로 잘못 입력됨
없는 데이터(missing data)	이름: (미입력)
지난 데이터(obsolete data)	주소: 예전 주소가 업데이트 되지 않음
비표준 데이터(non-standard data)	출생년월일: 2000년 1월 1일, 2000-1-1, 00-01-01 등 표준화되지 않은 방식으로 입력함
미완성 데이터(incomplete data)	주소: “서울 강남구 강남대로 390” 를 “서울 강남구”로 입력함
중복 데이터(redundant/duplicate data)	학생 정보: 출석부에 동일한 학생이 두 번 등록됨
관련 없는 데이터(irrelevant data)	전공 능력 평가 결과에 선호하는 음식 정보가 포함됨

Sensing Layer: 데이터 정제

■ 데이터 정제(Data cleaning/cleansing/scrubbing)

• 데이터 정제 과정

1. Inspection and profiling

- Dataset을 조사 및 검사하여, 수정해야 할 이슈가 있는지를 확인하고, 이 과정에서 profiling을 수행함
- profiling: 데이터 요소 간 상관관계 분석, 데이터 품질 점검, 데이터 오류/불일치 등의 문제 분석을 위한 통계적 접근 등의 수행 결과를 기록하는 것

2. Cleaning

- 잘못된 데이터를 정정하고, 데이터 불일치/중복 등의 문제를 해소하는 과정

3. Verification

- Cleaning 후에 얻은 데이터를 대상으로, 데이터 셋의 품질을 다시 점검하는 과정

4. Reporting

- 결과 보고

Sensing Layer: 데이터 정제

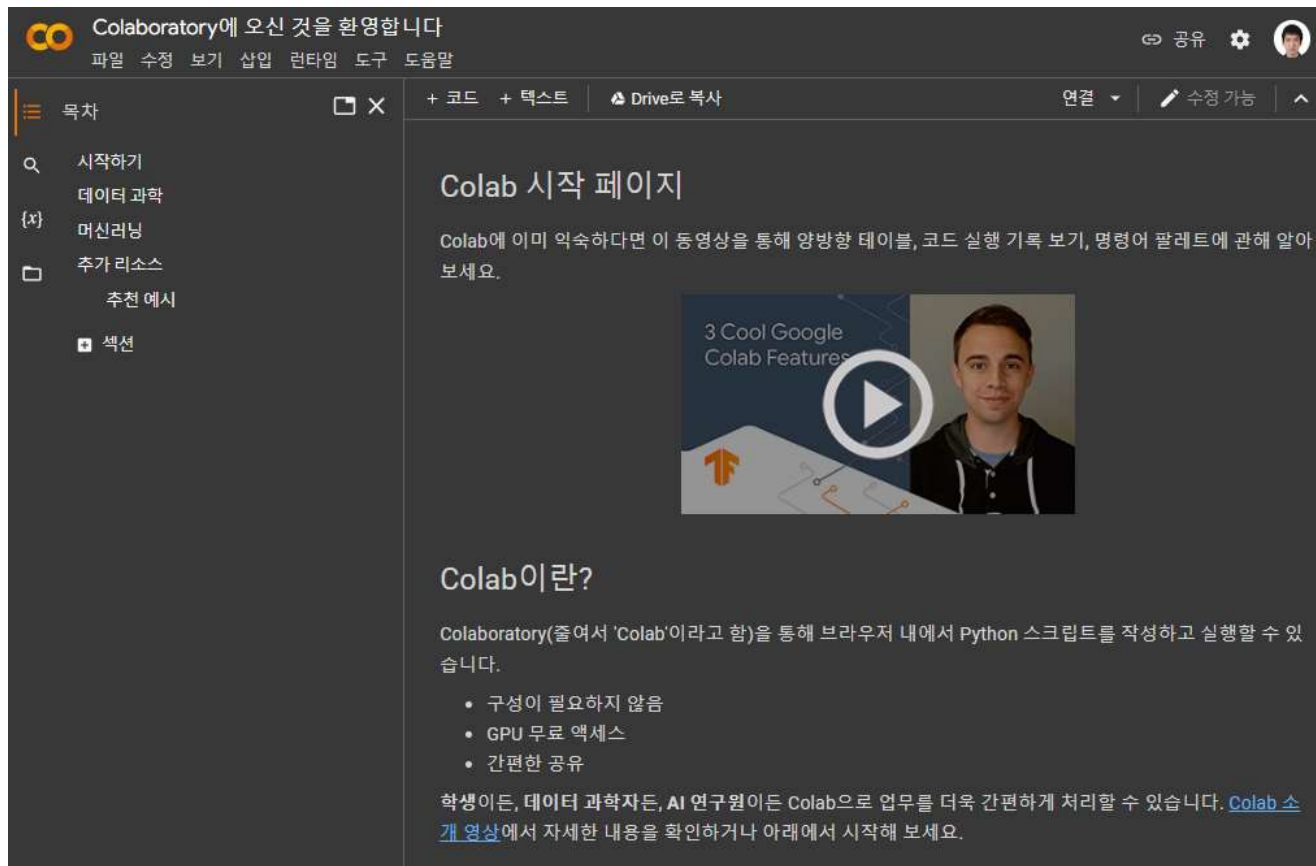
- 데이터 정제(Data cleaning/cleansing/scrubbing)
 - 데이터 정제 실습을 위한 사전 준비
 - 구글 코랩
 - pandas 라이브러리
 - 실습 환경 설정: 구글 코랩
 - 구글 코랩
 - 클라우드 기반의 주피터 노트북 개발 환경
 - 웹 브라우저에서 텍스트와 프로그램 코드를 자유롭게 작성할 수 있는 일종의 온라인 텍스트 에디터
 - 구글 드라이브와 연동하여, 개발 결과물을 클라우드에 저장할 수 있음
 - 무료로 사용 가능. 단,
 - 노트북을 최대 5개만 동시에 사용(활성화)할 수 있음
 - 노트북 하나를 12시간 이상 연결(활성화 상태로 유지)할 수 없음. 12시간 후 다시 연결 해야 함

구글 코랩을 사용하지 않고, 본인의 PC/노트북을 사용해도 됩니다.

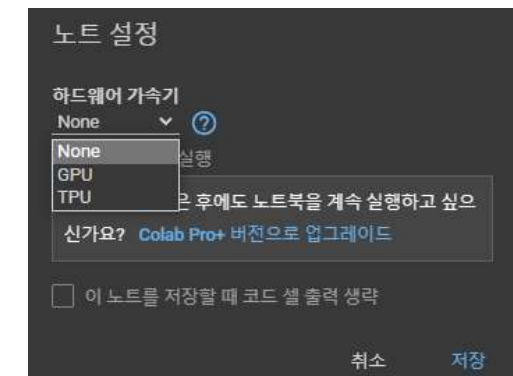


Sensing Layer: 데이터 정제

- 데이터 정제(Data cleaning/cleansing/scrubbing)
 - 데이터 정제 실습을 위한 사전 준비: 1) 구글 코랩, 2) pandas 라이브러리
 - 실습 환경 설정: 1) 구글 코랩
 - 구글 코랩
 - 구글 계정(Gmail 계정)이 필요
 - 접속 주소: <https://colab.research.google.com/>

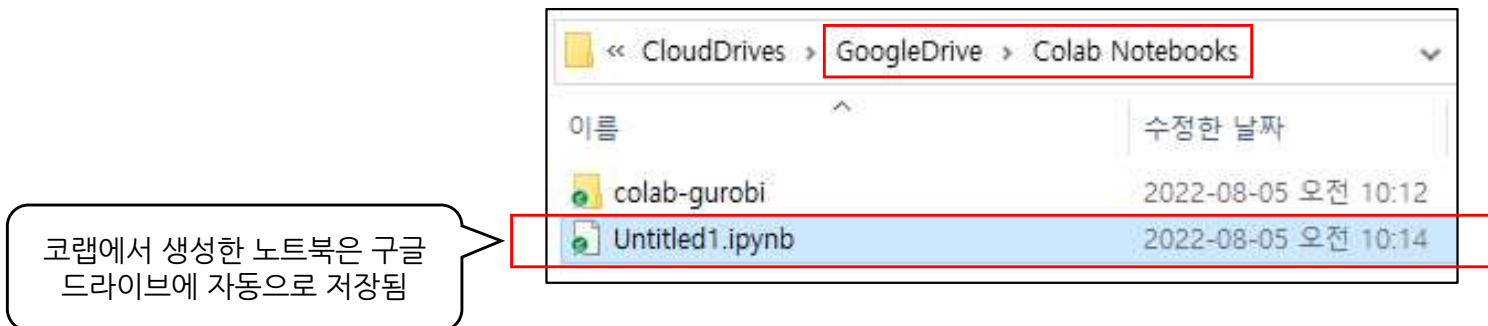
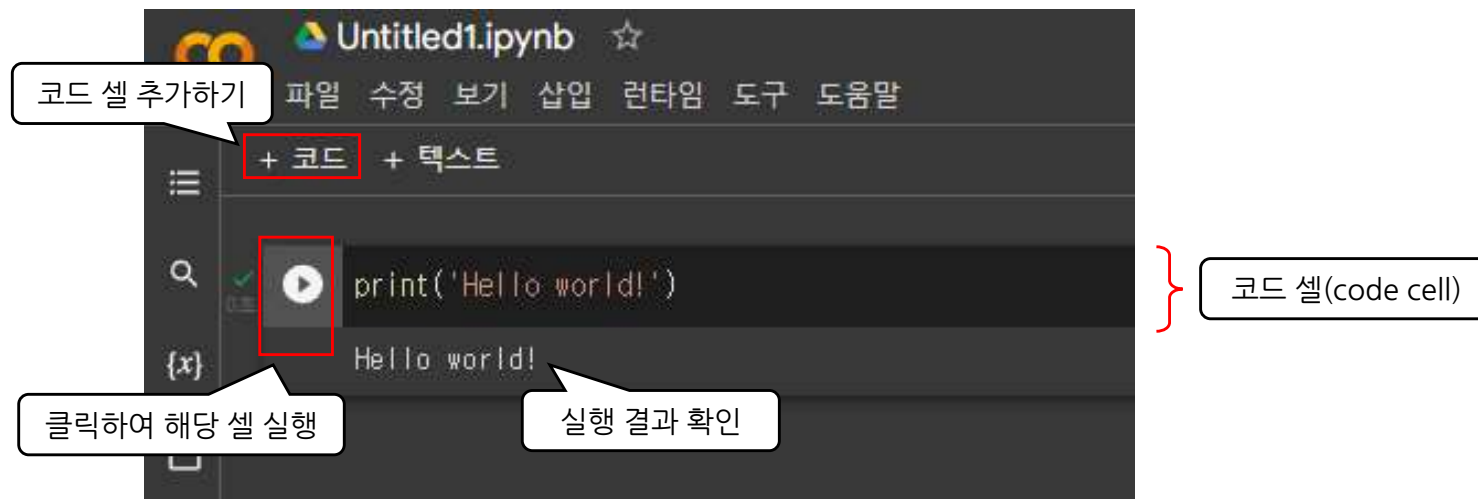


[런타임 > 런타임 유형 변경] 메뉴에서
딥러닝을 위한 하드웨어 가속기 설정도
할 수 있음



Sensing Layer: 데이터 정제

- 데이터 정제(Data cleaning/cleansing/scrubbing)
 - 데이터 정제 실습을 위한 사전 준비: 1) 구글 코랩, 2) pandas 라이브러리
 - 실습 환경 설정: 1) 구글 코랩
 - 구글 코랩
 - [파일 > 새 노트] 메뉴를 클릭하여 새 노트 생성 후, 화면 구성 살펴보기



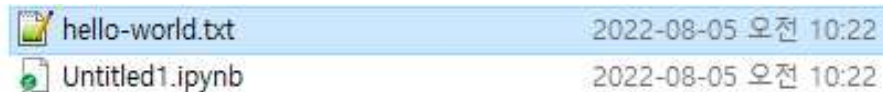
Sensing Layer: 데이터 정제

■ 데이터 정제(Data cleaning/cleansing/scrubbing)

- 데이터 정제 실습을 위한 사전 준비: 1) 구글 코랩, 2) pandas 라이브러리
- 실습 환경 설정: 1) 구글 코랩

- 구글 코랩

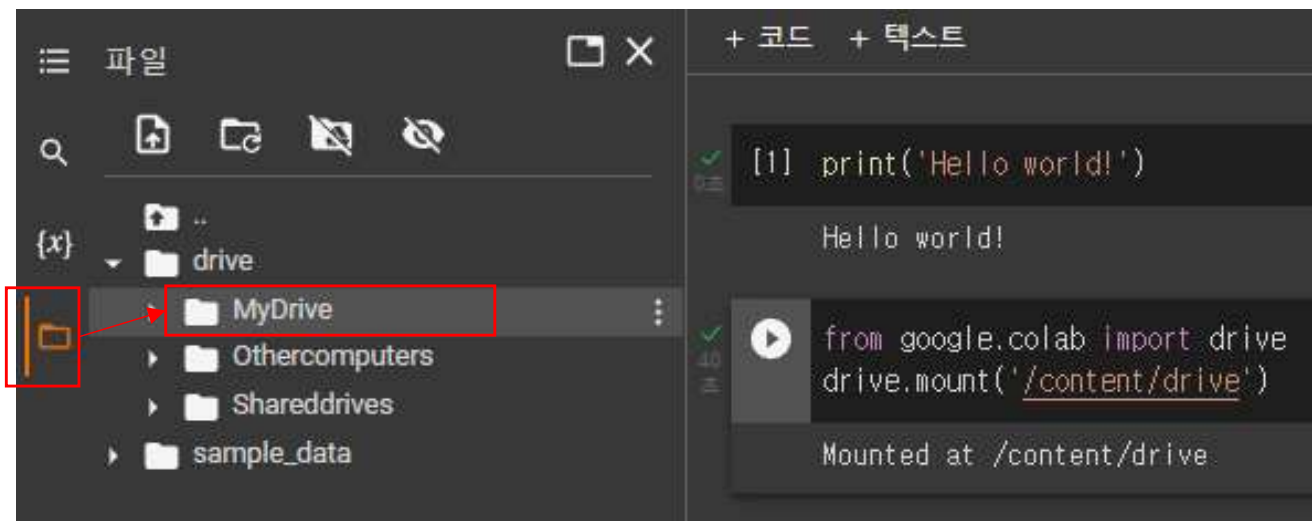
- 구글 드라이브를 코랩과 연동하기(구글 드라이브에 업로드한 파일을 코랩에서 접근하기)
 - 1) 구글 드라이브에 텍스트 파일 만들기



- 2) 코랩에서 아래의 코드 실행

```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

- 3) 코랩 왼쪽 메뉴에서 드라이브 탐색 가능 여부 확인



Sensing Layer: 데이터 정제

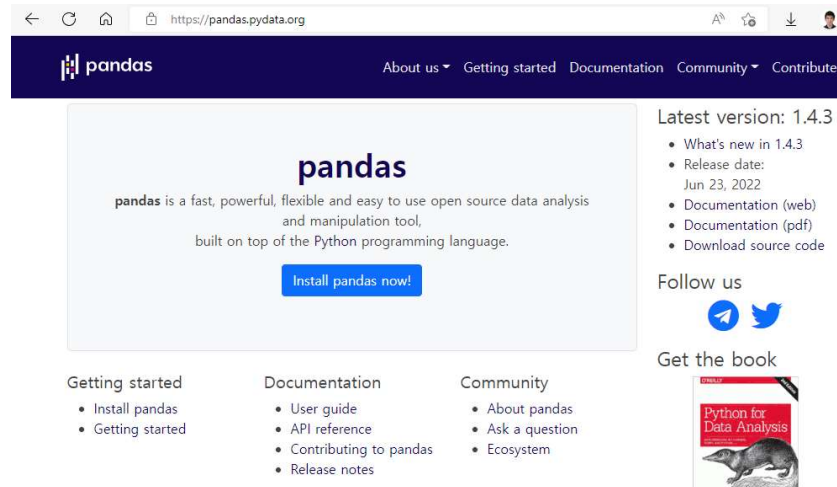
- 데이터 정제(Data cleaning/cleansing/scrubbing)
 - 데이터 정제 실습을 위한 사전 준비: 1) 구글 코랩, 2) pandas 라이브러리
 - 실습 환경 설정: 1) 구글 코랩
 - 구글 코랩
 - 구글 드라이브를 코랩과 연동하기(구글 드라이브에 업로드한 파일을 코랩에서 접근하기)
 - 1) 구글 드라이브에 텍스트 파일 만들기
 - 2) 코랩에서 아래의 코드 실행
 - 3) 코랩 왼쪽 메뉴에서 드라이브 탐색 가능 여부 확인
 - 4) 구글 드라이브에 저장된 파일의 내용을 읽어서 화면에 출력하기

```
▶ my_file_path = "/content/drive/MyDrive/Colab Notebooks/hello-world.txt"
f = open(my_file_path, 'r')
line = f.readline()
print(line)
f.close()

Hello, text file.
```

Sensing Layer: 데이터 정제

- 데이터 정제(Data cleaning/cleansing/scrubbing)
 - 데이터 정제 실습을 위한 사전 준비: 1) 구글 코랩, 2) pandas 라이브러리
 - 실습 환경 설정: 2) pandas 라이브러리
 - pandas: 데이터 분석(Data Analysis)을 위해 사용하는 파이썬 패키지 (<https://pandas.pydata.org/>)



- pandas 패키지 설치 설치 여부 확인(버전 확인)

```
+ 코드 + 텍스트

[1] import pandas
    print(pandas.__version__)

# pandas가 설치 안되어 있는 경우: !pip install pandas 명령으로 설치

1.3.5
```

감사합니다.

