

라즈베리파이 IOT키트 V2.0

사용자 설명서



목차

라즈베리파이 IOT키트 V2.0 사용자 설명서	1
가) 예제코드 설명 및 다운로드	3
나) 4	
다) 라즈베리파이 제어	5
1. GPIO제어 하기	5
① 라즈베리파이에 LED, 스위치 연결	5
② GPIO 출력 제어하기(LED)	8
③ 스위치 입력 테스트	9
2. 릴레이 보드 제어하기	10
① 릴레이 보드 연결하기	10
② 릴레이 동작 테스트하기	12
3. PMS7003 먼지센서 사용하기	13
① 먼지센서 연결하기	13
② 먼지센서 동작시켜 데이터 받아보기	14
라) Blynk로 원격 제어하기	17
1. blynk 소개	17
2. blynk 설치	17
① 스마트폰에 blynk 설치	17
② 라즈베리파이에 blynk python버전 설치	19
3. blynk로 GPIO제어하기	20
① 하드웨어 구성	20
② blynk 위젯 설정 및 배치	20
③ 예제코드 실행	22
4. blynk로 먼지센서 데이터 받아오기	24
① 하드웨어 구성	24
② blynk 위젯 설정 및 배치	24
③ 예제코드 실행	25
마) 통합하여 IOT 환경 구축하기	28
1. 하드웨어 및 blynk위젯 구성	28
① GPIO 구성	28
② blynk 위젯 구성	28
2. 예제 실행	29

가) 예제코드 설명 및 다운로드

라즈베리파이 IOT키트 V2.0를 사용하기 위해서는 라즈비안이 설치된 라즈베리파이와 라즈베리파이에 접속할 수 있는 환경이 갖춰져야 합니다. (모니터/VNC/SSH 등)

라즈베리파이 IOT키트 V2.0의 예제코드는 파이썬3를 이용해 작성되었습니다.
소스 코드는 엘레파츠 GitHub에서 다운받으실 수 있습니다.

<https://github.com/eleparts/iotkit>

아래에서는 라즈베리파이의 터미널 환경에서 예제코드를 다운로드 받아 실행하는 과정 진행하도록 하겠습니다.

라즈베리파이의 부팅이 완료되면 터미널 창을 열고 위의 예제코드를 다운받는 명령어를 실행해 줍니다.

github의 저장소에 업로드 되어있는 프로그램을 다운로드(복사)해 주는 명령어는
git clone https://github.com/eleparts/iotkit 입니다.

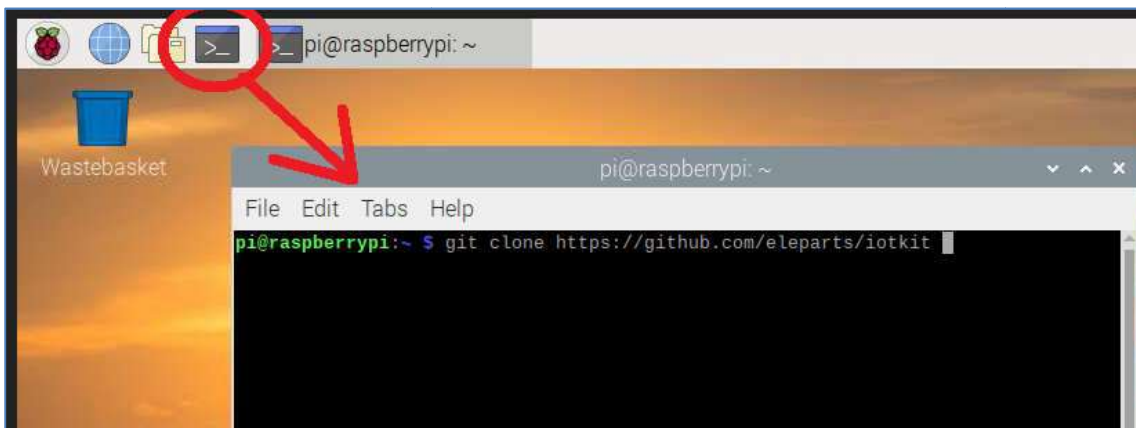


그림 1 GitHub 저장소 복사

기본 예제파일은 iotkit/example 경로에 있으나 추가 예제코드의 경우 start.sh 스크립트를 이용하여 다운로드 해 주어야 합니다.

먼저 **cd iotkit** 를 입력해 다운로드 한 디렉터리로 이동해 줍니다.

※ 디렉터리나 파일명 입력 시 tab 키를 눌러 자동완성을 이용하면 편리합니다.

그리고 **chmod +x start.sh** 명령어로 실행 권한을 추가해 준 뒤 **./start.sh** 명령어로 스크립트 파일을 실행해 줍니다.

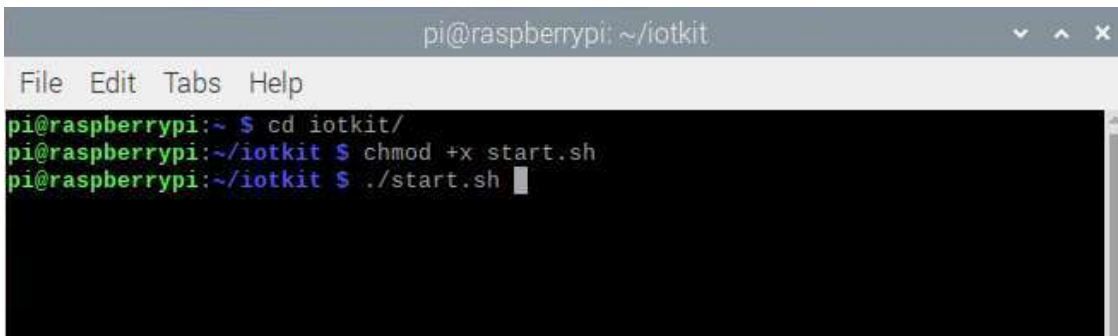


그림 2 다운로드 스크립트 실행

스크립트 프로그램이 실행되면 나머지 예제 코드도 자동으로 다운로드 됩니다.

다운로드가 완료되면 키트 구성 부품을 라즈베리파이에 연결하고 예제 프로그램을 테스트 해 볼 수 있습니다.

예제 코드의 동작 설명은 GitHub에서 직접, 혹은 라즈베리파이에서 예제코드 파일을 열어 확인 가능합니다.

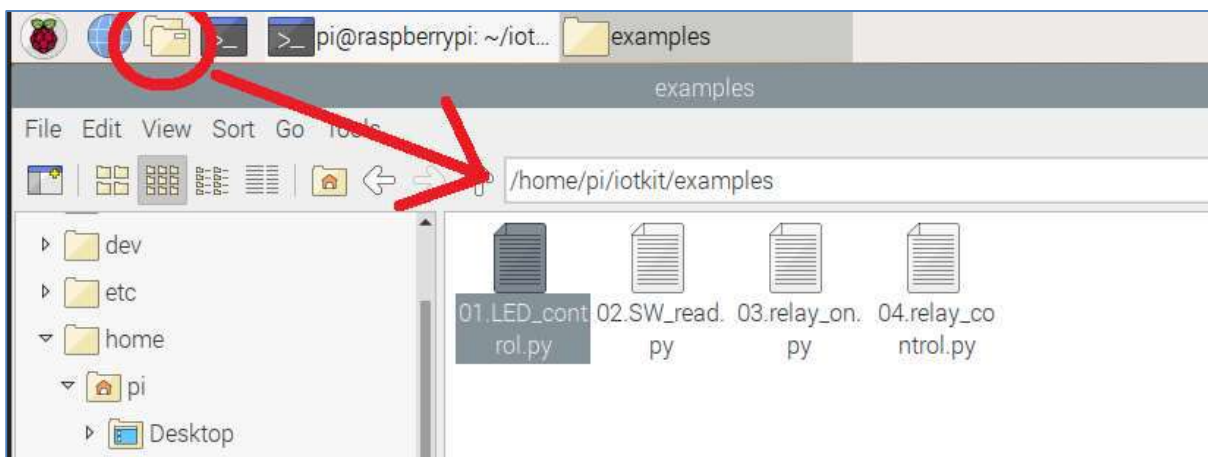


그림 3 GUI 환경에서 파일 확인

나)

위 경로에서 .py 파일을 열 경우 예제 코드를 직접 동작 가능한 파이썬용 Thonny IDE로 파일이 열리게 되어, 예제 코드를 바로 실행해 볼 수 있습니다.

※ 한글이 정상적으로 표시되지 않는 경우 터미널 창에 **sudo apt install fonts-unfonts-core**를 입력 해 한글 폰트 설치 후 재 부팅해 주면 됩니다.

다) 라즈베리파이 제어

1. GPIO제어 하기

① 라즈베리파이에 LED, 스위치 연결

H/W에서 가장 기본적인 테스트는 LED 점멸(깜빡이기) 테스트입니다.

라즈베리파이에 회로를 연결하기 위해 브레드보드에 LED 및 스위치 등을 꽂아 회로를 구성해 주도록 하겠습니다.

브레드보드는 아래의 검은 선을 따라 5개의 홀이 서로 연결되어 있어 각 부품의 핀을 서로 연결시켜 줍니다.

또한, 세로줄인 빨간색, 파란색 선은 길게 이어져 있어 전원을 연결해 전원 공급용 선으로 사용하면 매우 편리합니다.

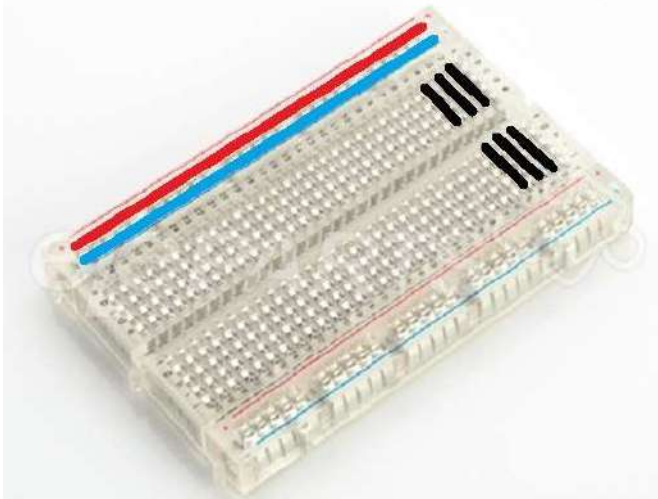


그림 4 브레드보드

점퍼케이블로 핀을 연결하실 때에는 색을 정해(+VCC는 빨간색, -/GND는 검은색, GPIO/데이터통신 핀은 초록, 노랑, 흰색 등등) 연결해 주면 추후 회로 확인에 큰 도움이 됩니다.

※ 주의: 핀 연결 시 절대 + (VCC)와 - (GND)가 직접 연결되어서는 안됩니다. (쇼트/합선)

브레드보드에 꽂는 소자 중에서 **스위치 및 LED는 방향이 있는 소자**이므로 방향에 주의해 사용해 주셔야 합니다.

LED는 다리가 휘어진/긴 다리가 + 극성이며, 짧은 쪽이 -극 입니다.

(다리를 절단한 경우 LED 원통 부분 하단의 깎인 방향(-), 내부 금속 모양 등으로도 구별이 가능합니다.)

포함된 **스위치**는 하단에 막대가 그려져 있으며, 해당 **하단 막대 그림을 기준으로 같은 방향의 핀**은 항상 연결되어 있습니다. (스위치를 누르면 전부 연결됩니다.)

라즈베리파이에 부품 연결은 아래 사진을 따라 똑같이 해 주시면 됩니다.

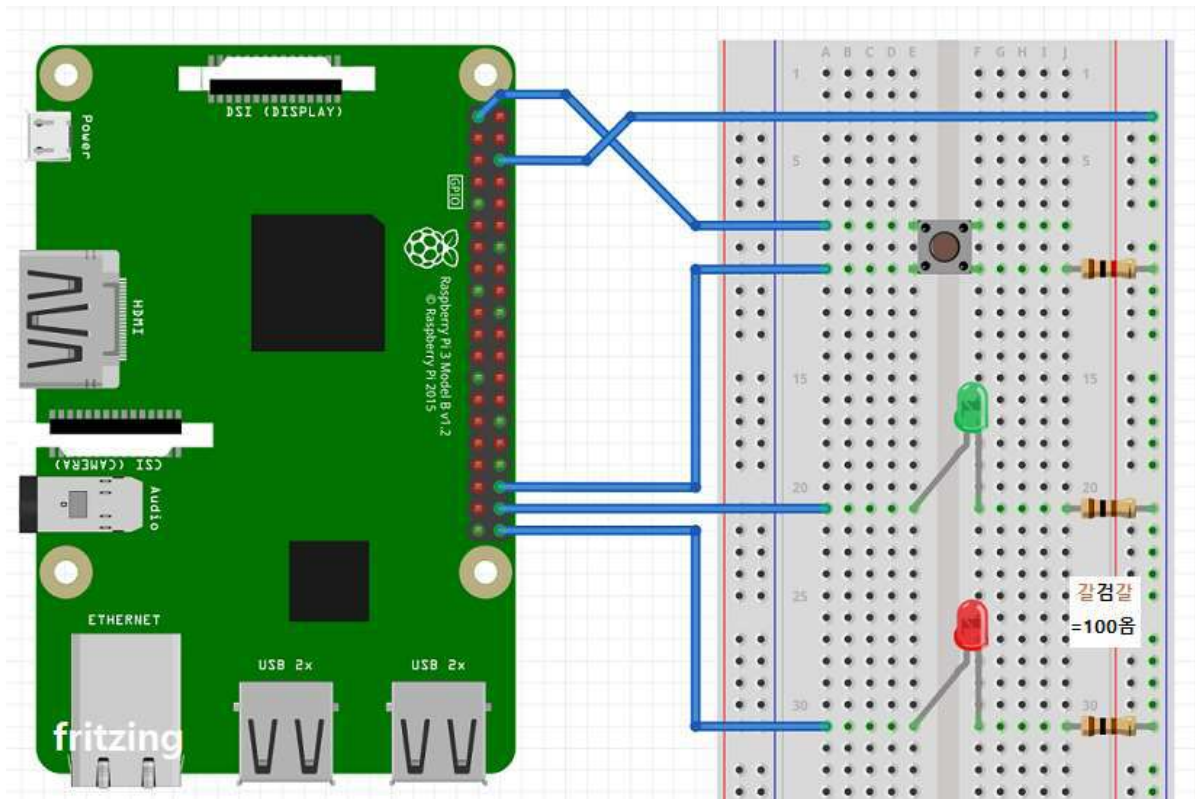


그림 5 라즈베리파이에 LED및 스위치 연결

저항은 방향이 없으며, 저항 값은 띠의 색상으로 구분 가능합니다.

LED에는 갈색/검정/갈색/(금색)띠의 저항(100옴)을 LED옆에 연결해 줍니다.

스위치의 우측에 연결된 저항은 1k~10k옴의 **풀다운 저항**으로, 스위치가 눌리지 않은 상태에서 0V쪽에 연결되어 GPIO의 핀의 **전압을 안정시켜**(풀다운 = GND/ 풀업 = VCC)주는 역할을 해 줍니다.

라즈베리파이는 프로그램에서 자체 내부 풀업/ 풀다운을 지원하기 때문에 꼭 연결하지 않아도 되지만 처음 공부하시는 경우 연결해 주시는 것을 추천합니다.

라즈베리파이의 핀 중에서 맨 위의 좌측이 3.3V핀이며, 위에서 세 번째 줄 우측에 연결된 핀은 GND 핀입니다.

핀 번호는 아래 라즈베리파이 GPIO핀 배치도를 참고해 주시면 됩니다.

※ 참고: 라즈베리파이 GPIO 핀 배치도

Raspberry Pi 3 GPIO Header				
Pin#	NAME		NAME	Pin#
01	3.3v DC Power		DC Power 5v	02
03	GPIO02 (SDA1 , I ² C)		DC Power 5v	04
05	GPIO03 (SCL1 , I ² C)		Ground	06
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14	08
09	Ground		(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)		Ground	14
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23	16
17	3.3v DC Power		(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)		Ground	20
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08	24
25	Ground		(SPI_CE1_N) GPIO07	26
27	ID_SD (I ² C ID EEPROM)		(I ² C ID EEPROM) ID_SC	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12	32
33	GPIO13		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40

Rev. 2
29/02/2016

www.element14.com/RaspberryPi

그림 6 라즈베리파이 핀 배치도

라즈베리파이의 끝부분이 1,2번 핀 방향이며 안쪽이 1번, 바깥쪽이 2번 핀입니다.

핀 번호는 물리적인 핀 순서이며 프로그램 내부에서는 GPIOxx 번호(BCM 번호)를 사용합니다.

② GPIO 출력 제어하기(LED)

라즈베리파이의 GPIO의 출력은 3.3V로 HIGH / LOW (3.3V / 0V)로 제어됩니다.

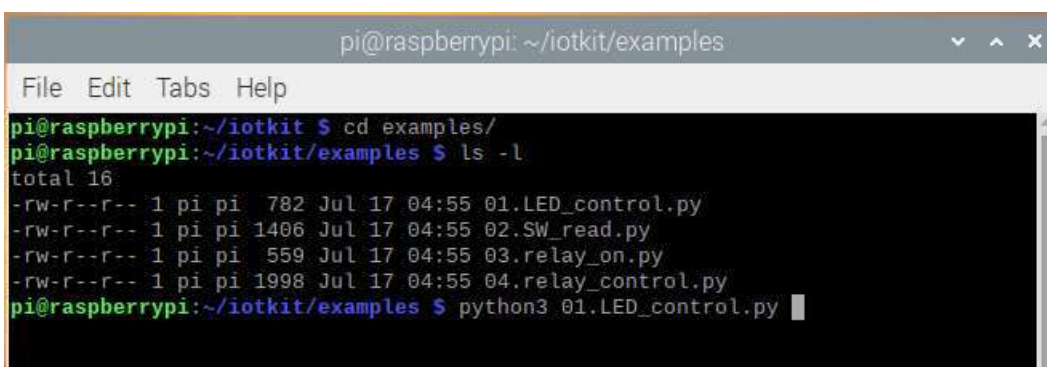
다만, 전압은 눈으로 바로 확인이 불가능하기 때문에 LED를 연결하여 동작 확인이 가능합니다.

위에서 다운받은 예제 코드를 이용하여 연결된 LED를 제어해 보도록 하겠습니다.

cd example 명령어로 예제코드가 있는 디렉터리로 이동합니다.

ls 명령어로 현재 디렉터리에 있는 파일 목록을 확인 가능합니다. (-l은 상세보기 옵션)

그리고 **python3 01.LED_control.py** 명령어로 실행해 주면 LED가 동작하게 됩니다.



```

pi@raspberrypi: ~/iotkit/examples
File Edit Tabs Help
pi@raspberrypi:~/iotkit $ cd examples/
pi@raspberrypi:~/iotkit/examples $ ls -l
total 16
-rw-r--r-- 1 pi pi 782 Jul 17 04:55 01.LED_control.py
-rw-r--r-- 1 pi pi 1406 Jul 17 04:55 02.SW_read.py
-rw-r--r-- 1 pi pi 559 Jul 17 04:55 03.relay_on.py
-rw-r--r-- 1 pi pi 1998 Jul 17 04:55 04.relay_control.py
pi@raspberrypi:~/iotkit/examples $ python3 01.LED_control.py
  
```

그림 7 예제 01.LED_control.py

예제를 실행시켜 주면 아래와 같이 LED가 번갈아 가며 켜지게 됩니다. (5회 반복 후 종료)

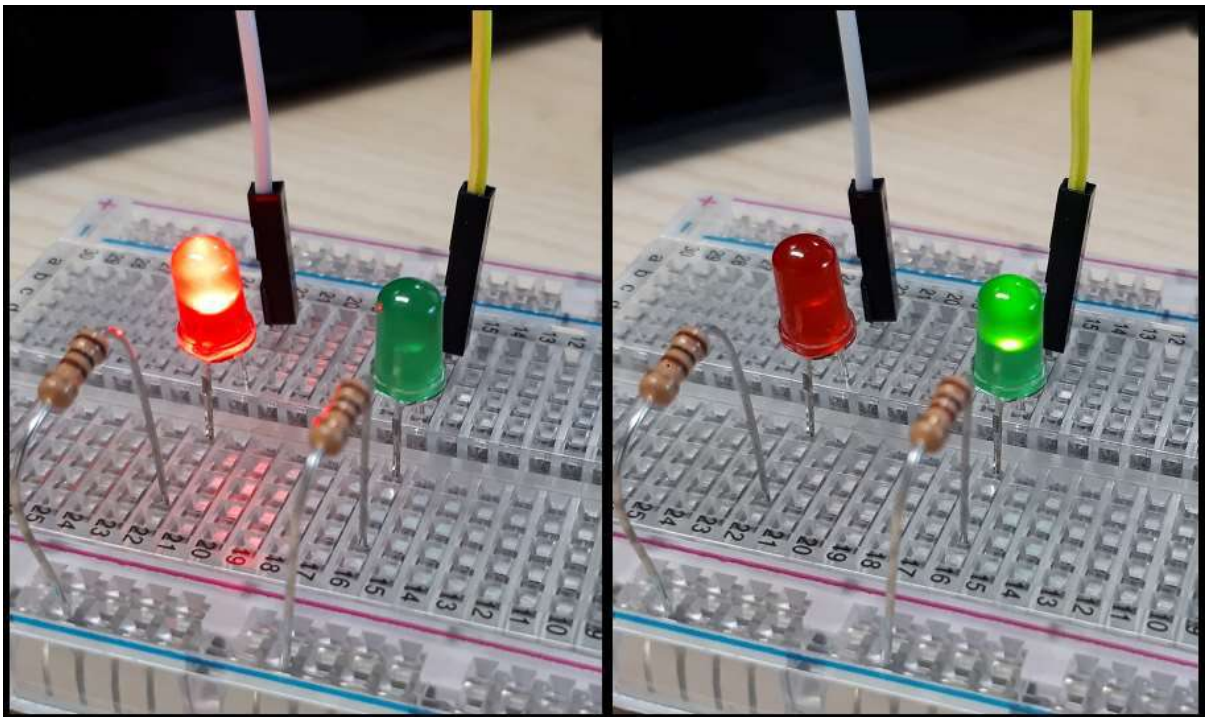
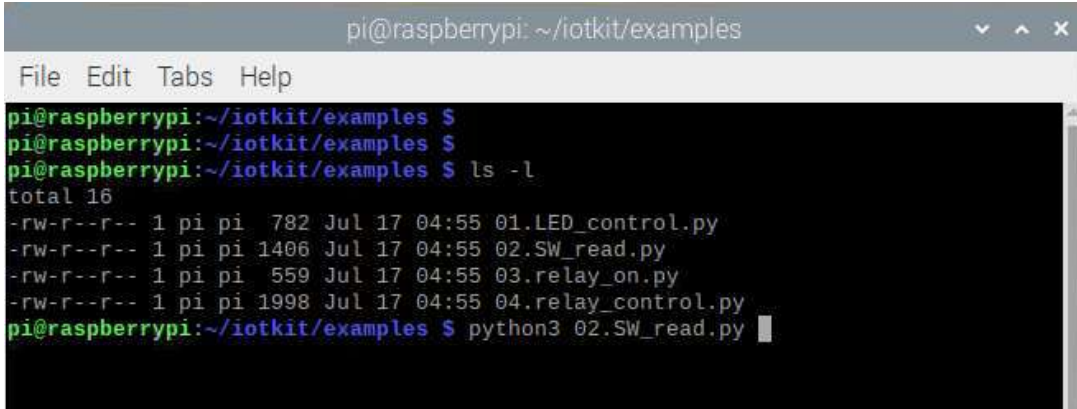


그림 8 01.LED_control.py 실행결과

③ 스위치 입력 테스트

스위치 입력 예제도 동일 디렉터리에서 **python3 02.SW_read.py** 명령어를 입력해 실행해 주시면 됩니다



```

pi@raspberrypi: ~/iotkit/examples
File Edit Tabs Help
pi@raspberrypi:~/iotkit/examples $
pi@raspberrypi:~/iotkit/examples $
pi@raspberrypi:~/iotkit/examples $ ls -l
total 16
-rw-r--r-- 1 pi pi 782 Jul 17 04:55 01.LED_control.py
-rw-r--r-- 1 pi pi 1406 Jul 17 04:55 02.SW_read.py
-rw-r--r-- 1 pi pi 559 Jul 17 04:55 03.relay_on.py
-rw-r--r-- 1 pi pi 1998 Jul 17 04:55 04.relay_control.py
pi@raspberrypi:~/iotkit/examples $ python3 02.SW_read.py
  
```

그림 9 예제 02.SW_read.py

예제 실행 시 빨간색 LED가 켜지며, 스위치를 누르면 3초간 녹색 LED와 빨간색 LED가 같이 켜졌다가 꺼지며 종료됩니다.

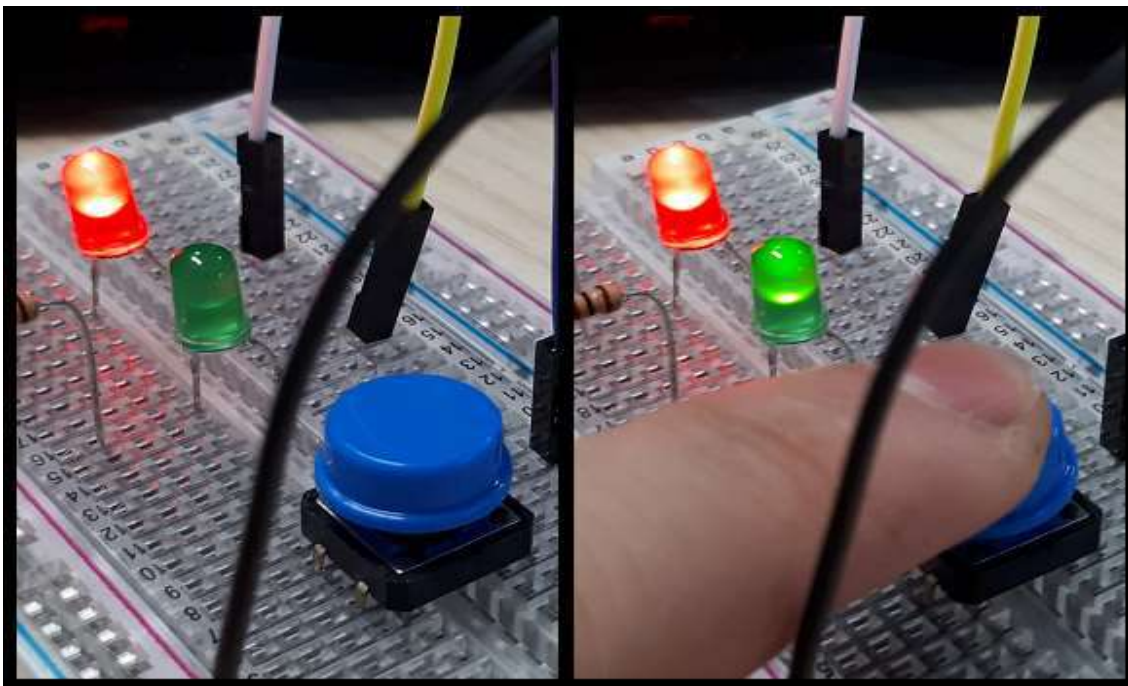


그림 10 02.SW_read.py 실행결과

2. 릴레이 보드 제어하기

① 릴레이 보드 연결하기

릴레이 보드는 브레드보드에 연결하지 않고 점퍼 케이블로 직접 연결이 가능합니다.
또한, 서로 다른 핀을 사용하도록 하여 스위치와 LED를 제거하지 않고 같이 연결이 가능합니다.

보드의 사양은 제어신호 전압 3.3V, 전원 3~5V 사양이므로 전원은 5V에 연결해 주고 컨트롤 신호는 라즈베리파이의 GPIO(3.3V)에 연결해 주시면 됩니다.

릴레이 보드의 GPIO는 18번 핀을 사용합니다.

아래 회로도와 동일하게 연결해 주시면 됩니다.

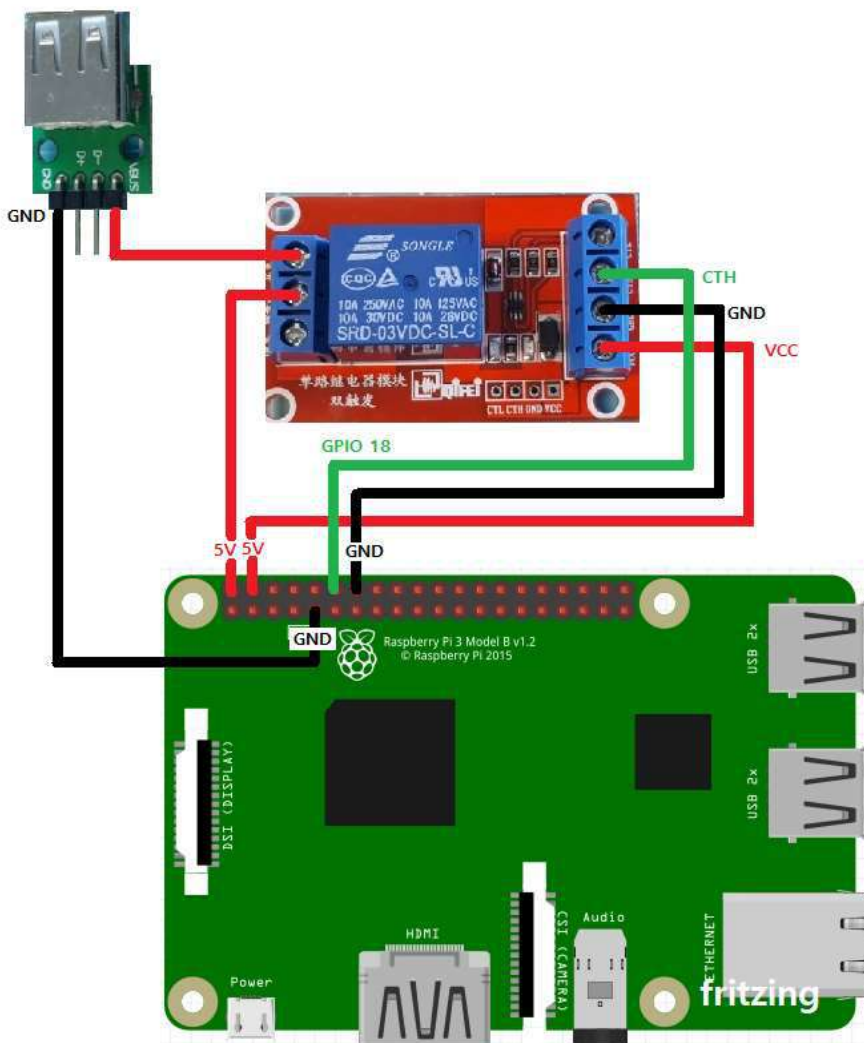


그림 11 릴레이보드 연결

릴레이 보드에 점퍼 케이블을 연결할 때는 터미널블록의 나사를 적당히 풀어 준 뒤 점퍼 케이블의 핀 부분을 터미널 블록에 꽂아 나사를 돌려서 고정시켜 주시면 됩니다.

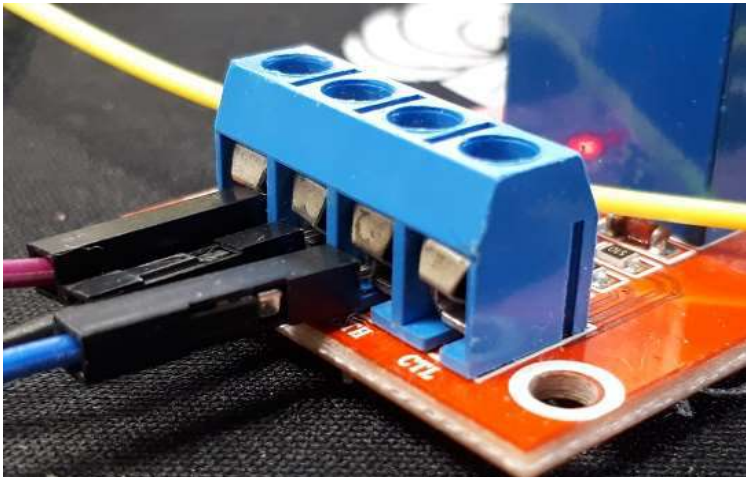


그림 12 릴레이 보드에 점퍼케이블 연결

GPIO가 입력되는 부분의 CTL / CTH는 각각 LOW일 때/HIGH일 때 동작(컨트롤)되는 것을 뜻하며, USB가 연결되는 쪽의 핀은 가운데를 기준으로 좌 우가 반대되는 동작을 합니다.

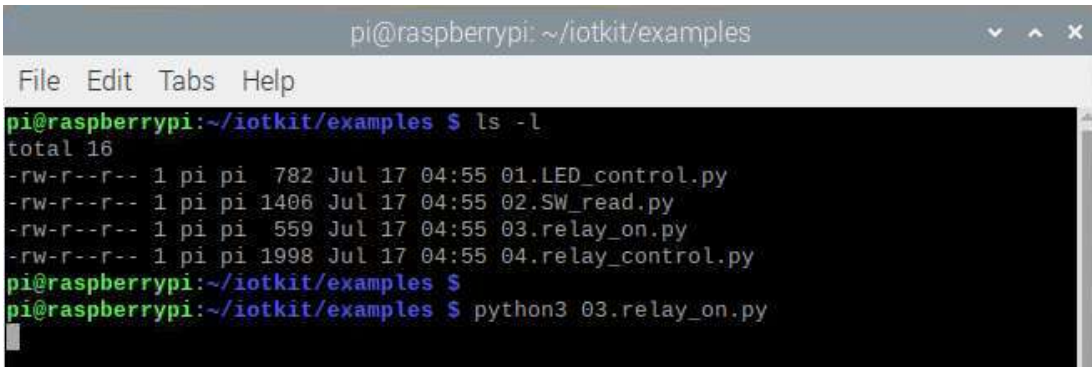
USB에는 구성품인 미니 선풍기를 연결해 릴레이의 동작을 확인할 수 있습니다.
(릴레이 동작은 소리로도 확인 가능하며 동작 시 '딸깍'하는 소리가 나게 됩니다.)

※ 라즈베리파이의 5V 전원을 이용하므로 5W (5V 1A)이상의 제품을 연결할 경우 라즈베리파이의 전원이 부족해 질 수 있습니다.

② 릴레이 동작 테스트하기

릴레이를 사용하는 예제는 **03.relay_on.py** 및 **04.relay_control.py**이며, **04.relay_control.py**는 LED 및 스위치가 연결되어 있어야 합니다.

첫 번째 릴레이 예제는 **python3 03.relay_on.py**를 입력해 실행시켜 주면 됩니다.



```

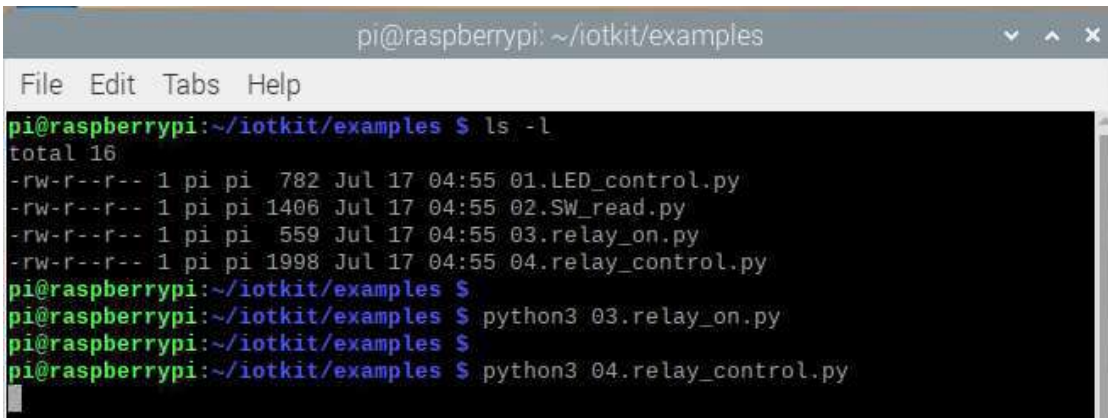
pi@raspberrypi: ~/iotkit/examples
File Edit Tabs Help
pi@raspberrypi:~/iotkit/examples $ ls -l
total 16
-rw-r--r-- 1 pi pi 782 Jul 17 04:55 01.LED_control.py
-rw-r--r-- 1 pi pi 1406 Jul 17 04:55 02.SW_read.py
-rw-r--r-- 1 pi pi 559 Jul 17 04:55 03.relay_on.py
-rw-r--r-- 1 pi pi 1998 Jul 17 04:55 04.relay_control.py
pi@raspberrypi:~/iotkit/examples $
pi@raspberrypi:~/iotkit/examples $ python3 03.relay_on.py

```

그림 13 예제 03.relay_on.py

03.relay_on.py 예제를 실행시켜 주면 5초 동안 릴레이가 작동 후 꺼진 뒤 프로그램이 종료됩니다.

두 번째 릴레이 예제는 **python3 04.relay_control.py**를 입력해 실행시켜 주면 됩니다.



```

pi@raspberrypi: ~/iotkit/examples
File Edit Tabs Help
pi@raspberrypi:~/iotkit/examples $ ls -l
total 16
-rw-r--r-- 1 pi pi 782 Jul 17 04:55 01.LED_control.py
-rw-r--r-- 1 pi pi 1406 Jul 17 04:55 02.SW_read.py
-rw-r--r-- 1 pi pi 559 Jul 17 04:55 03.relay_on.py
-rw-r--r-- 1 pi pi 1998 Jul 17 04:55 04.relay_control.py
pi@raspberrypi:~/iotkit/examples $
pi@raspberrypi:~/iotkit/examples $ python3 03.relay_on.py
pi@raspberrypi:~/iotkit/examples $
pi@raspberrypi:~/iotkit/examples $ python3 04.relay_control.py

```

그림 14 예제 04.relay_control.py

04.relay_control.py 예제는 실행시키면 빨간 LED가 켜지고 대기상태가 되며, 스위치를 누르면 빨간 LED가 꺼지고 녹색 LED가 켜지면서 릴레이가 동작합니다.

그리고 2초 후 다시 스위치를 누르면 녹색 LED가 꺼지고 빨간 LED가 켜지면서 3초 대기 후 프로그램이 종료됩니다.

3. PMS7003 먼지센서 사용하기

① 먼지센서 연결하기

PMS7003먼지센서는 USB to Serial(UART) 케이블과 인터페이스 보드를 이용해 라즈베리파이에 연결해 줄 수 있습니다.

UART 통신을 이용하기 때문에 전원(+, -)외의 통신단자 Tx(송신), Rx(수신)을 연결해 주어야 합니다.

USB to Serial(UART)케이블에서는 **녹색이 Tx, 흰색이 Rx** 이며 인터페이스 보드에 적힌 Rx, Tx와 엇갈리도록 연결해 주시면 됩니다.

데이터 방향 (Tx -> Rx)	UART 케이블 (라즈베리파이)	인터페이스보드 (먼지센서)
라즈베리파이 -> 먼지센서	녹색(Tx)	Rx 핀
라즈베리파이 <- 먼지센서	흰색(Rx)	Tx 핀

표 1 UART 케이블 연결방법 및 데이터 흐름

아래 사진을 참고해서 연결합니다.



그림 15 먼지센서 연결

그리고 인터페이스 보드에 먼지센서를 연결, USB는 라즈베리파이에 연결해 주시면 됩니다.

이때, 라즈베리파이에 연결한 USB는 라즈베리파이에서 /dev 경로에 추가됩니다.

/dev 경로에서 ls를 입력해 확인하는 것도 가능하지만 **ls /dev | grep ttyUSB** 명령어로 연결된 USB리스트만 확인하는 것도 가능합니다.

위 명령어는 /dev 경로에 있는 ttyUSB~로 시작하는 파일/디렉터리의 목록(ls)을 전부 표시해 줍니다.

USB를 연결하기 전/후로 **ls /dev | grep ttyUSB** 명령어를 입력해 확인해 보도록 합니다.



```

pi@raspberrypi: ~/iotkit
File Edit Tabs Help
pi@raspberrypi:~/iotkit $ ls /dev | grep ttyUSB
pi@raspberrypi:~/iotkit $
pi@raspberrypi:~/iotkit $ ls /dev | grep ttyUSB
ttyUSB0
pi@raspberrypi:~/iotkit $
  
```

그림 16 USB 연결확인

/dev/ttyUSB0로 먼지센서가 연결된 USB 케이블이 연결된 것을 확인할 수 있습니다.

② 먼지센서 동작시켜 데이터 받아보기

먼지센서 예제 코드는 start.sh 스크립트를 실행해서 추가로 다운로드 해야 합니다.

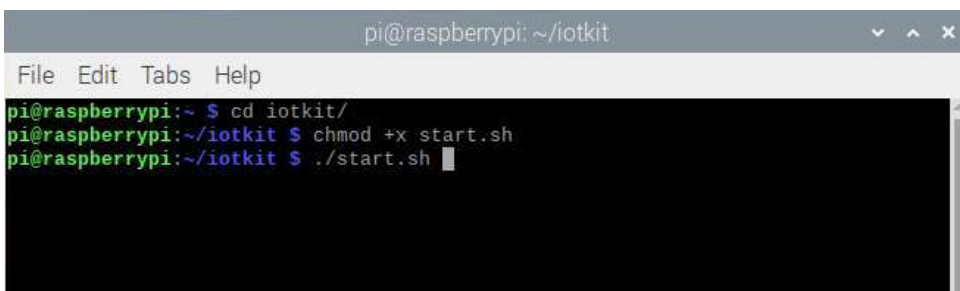
start.sh 스크립트를 실행시켜 준 경우 examples 디렉터리에 05.PMS7003 디렉터리가 추가됩니다.

만일 실행시켜 주지 않은 경우 아래와 같이 실행해 주시면 됩니다.

iotkit 경로로 이동

chmod +x start.sh

./start.sh



```

pi@raspberrypi: ~/iotkit
File Edit Tabs Help
pi@raspberrypi:~ $ cd iotkit/
pi@raspberrypi:~/iotkit $ chmod +x start.sh
pi@raspberrypi:~/iotkit $ ./start.sh
  
```

그림 17 start.sh 스크립트 실행

위 스크립트를 실행하면 아래 경로의 PMS7003 예제를 다운로드 해 줍니다.

<https://github.com/eleparts/PMS7003>

examples 경로의 05.PMS7003으로 다운로드 되므로 해당 경로로 이동해 줍니다.

PMS7003.py는 라이브러리 및 먼지센서 테스트용 프로그램이 내장되어 있습니다.

dust_chk.py는 PMS7003.py를 import하여 사용하는 예제 프로그램 입니다.

```

pi@raspberrypi: ~/iotkit/examples/05.PMS7003
File Edit Tabs Help
pi@raspberrypi:~/iotkit $ cd examples/05.PMS7003/
pi@raspberrypi:~/iotkit/examples/05.PMS7003 $ ls -l
total 20
drwxr-xr-x 2 pi pi 4096 Jul 18 01:57 'data sheet'
-rw-r--r-- 1 pi pi 2388 Jul 18 01:57 dust_chk.py
-rw-r--r-- 1 pi pi 5907 Jul 18 01:57 PMS7003.py
-rw-r--r-- 1 pi pi 1799 Jul 18 01:57 README.md
pi@raspberrypi:~/iotkit/examples/05.PMS7003 $

```

그림 18 PMS7003 라이브러리

05.PMS7003디렉터리로 이동 후 프로그램 실행에 앞서 연결되어 있는 포트의 설정을 해 주어야 합니다. GUI환경에서 해당 파일을 열거나 leafpad(윈도우의 메모장과 비슷한 텍스트 편집기) 등을 이용해 내용을 수정해 주셔야 합니다.

(ssh환경인 경우 nano혹은 vi 에디터 등을 사용해 주시면 됩니다.)

leafpad PMS7003.py를 입력해 파일을 열어준 뒤 스크롤을 아래로 쪽 내려 #USE PORT 부분을 찾아 **USB0**로 변경해 주시면 됩니다.

```

pi@raspberrypi: ~/iotkit/examples/05.PMS7003
File Edit Tabs Help
pi@raspberrypi:~/iotkit/examples/05.PMS7003 $ ls -l
total 20
drwxr-xr-x 2 pi pi 4096 Jul 18 01:57 'data sheet'
-rw-r--r-- 1 pi pi 2388 Jul 18 01:57 dust_chk.py
-rw-r--r-- 1 pi pi 5907 Jul 18 01:57 PMS7003.py
-rw-r--r-- 1 pi pi 1799 Jul 18 01:57 README.md
pi@raspberrypi:~/iotkit/examples/05.PMS7003 $ leafpad PMS7003.py
*PMS7003.py
File Edit Search Options Help
# UART / USB Serial : 'dmesg | grep ttyUSB'
USB0 = '/dev/ttyUSB0'
UART = '/dev/ttyAMA0'
# USE PORT
SERIAL_PORT = UART
# Baud Rate
Speed = 9600
# example
if name == '__main__':

```

그림 19 PORT 변경

빨간 사각형 부분의 UART를 USB0로 변경해 주시면 되며, 이는 바로 위 주황색 원 부분에 정의되어 있습니다.

만약 위에서 /dev/ttyUSB0가 아닌 /dev/ttyUSB1 등에 연결된 경우 주황색 원 뒤의 '/dev/ttyUSB0' 부분을 동일하게 변경해 주어야 합니다.

[컨트롤 + S]로 저장 후 닫아주신 뒤 해당 프로그램을 실행할 수 있습니다.

프로그램 실행은 **python3 PMS7003.py**로 실행시켜 주시면 됩니다.
실행에 문제가 발생하는 경우 앞에 **sudo**를 붙여 실행해 볼 수 있습니다.

```

pi@raspberrypi: ~/iotkit/examples/05.PMS7003
File Edit Tabs Help
0.5um in 0.1L of air : 270
1.0um in 0.1L of air : 60
2.5um in 0.1L of air : 17
5.0um in 0.1L of air : 3
10.0um in 0.1L of air : 0
Reserved F : 151 | Reserved B : 0
CHKSUM : 664 | read CHKSUM : 664 | CHKSUM result : True
=====
DATA read success
=====
Header : B M | Frame length : 28
PM 1.0 (CF=1) : 5 | PM 1.0 : 5
PM 2.5 (CF=1) : 14 | PM 2.5 : 14
PM 10.0 (CF=1) : 16 | PM 10.0 : 16
0.3um in 0.1L of air : 972
0.5um in 0.1L of air : 282
1.0um in 0.1L of air : 64
2.5um in 0.1L of air : 18
5.0um in 0.1L of air : 3
10.0um in 0.1L of air : 0
Reserved F : 151 | Reserved B : 0
CHKSUM : 711 | read CHKSUM : 711 | CHKSUM result : True
=====

```

그림 20 PMS7003 동작 테스트

PMS7003.py 예제 코드를 동작시키면 위와 같이 1초 간격으로 먼지센서의 데이터를 다운로드 하게 됩니다.

dust_chk.py 프로그램은 위의 PMS7003.py 라이브러리를 이용해 먼지센서를 동작하는 예제 코드로 PMS7003.py와 동일하게 USB0로 변경 후 실행시켜 주시면 데이터를 1회 출력합니다.

라) Blynk로 원격 제어하기

1. blynk 소개

Blynk는 <https://blynk.io/> 에서 제작한 IOT 플랫폼입니다

Android 및 IOS 환경에서 앱을 다운로드 하여 아두이노, ESP시리즈, 라즈베리파이 등의 하드웨어를 원격 제어하거나 데이터를 수신하는 동작 등을 매우 손쉽게 구성할 수 있습니다.

앱의 UI구성도 매우 간단하여 필요한 블록을 드래그&드롭으로 배치하고 각 블록의 번호를 설정하는 것으로 연결된 하드웨어에 연결할 수 있습니다.

다만, 각 블록은 할당된 에너지라는 화폐를 사용하며 기본 2000이 제공됩니다. (블록에 따라 200~500가량 사용됨)

에너지는 블록을 추가하면 사용되고, 블록을 삭제하면 다시 반환됩니다.

에너지 최대량을 늘리기 위해서는 추가 결제를 해 주셔야 합니다.

여기서는 blynk앱을 다운로드 하고 라즈베리파이에 blynk Python버전을 설치하여 두 기기를 연동하고 실시간으로 제어하는 동작을 진행하도록 하겠습니다.

2. blynk 설치

blynk를 이용하기 위해서는 스마트폰과 라즈베리파이에 각각 blynk를 다운로드 해 주어야 합니다.

① 스마트폰에 blynk 설치

스마트폰에서는 Android/ IOS에 따라 각각의 스토어를 통해 간단하게 다운로드 하실 수 있습니다.

Android는 플레이 스토어

<https://play.google.com/store/apps/details?id=cc.blynk>

IOS에서는 앱 스토어에서 다운로드가 가능합니다.

<https://apps.apple.com/us/app/blynk-iot-for-arduino-esp32/id808760481>

각 스토어에서 blynk를 검색하면 blynk 앱 페이지를 확인할 수 있습니다.



그림 21 플레이스토어 blynk 앱 페이지

설치를 해 주신 뒤 실행시키면 가입&로그인 화면이 열립니다.

가입 시 꼭 사용 가능한 이메일 주소를 입력해 주시고 진행하시면 됩니다.

가입 후 New Project 로 프로젝트를 생성해 주시면 됩니다.

> 디바이스는 **라즈베리파이 3B** 선택

생성이 완료되면 아래 이미지와 같이 이메일 주소로 라즈베리파이에 적어야 하는 토큰 주소가 발송됩니다.

토큰 번호는 삭제하지 말고 따로 적어 두시면 됩니다.
(분실 시 언제든지 재발급이 가능합니다)

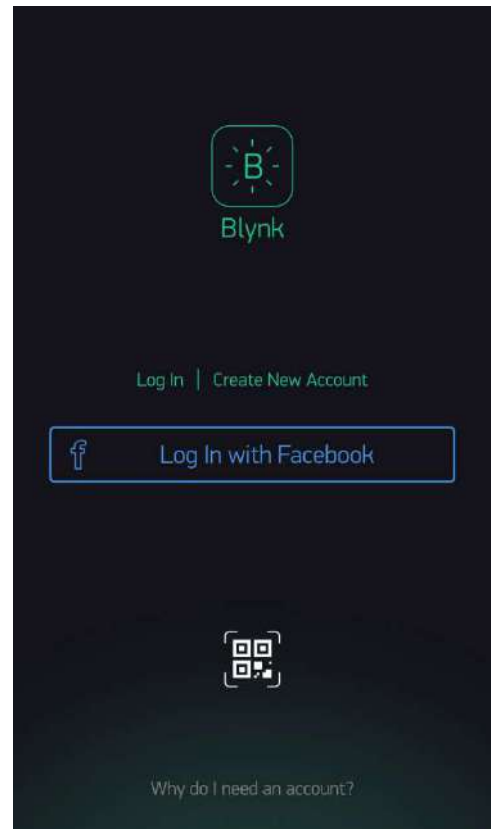


그림 22 blynk 로그인&가입 페이지



그림 23 토큰 주소 메일

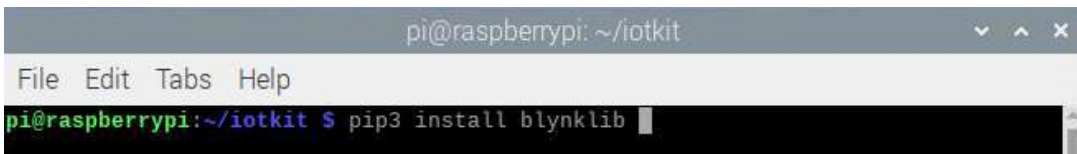
② 라즈베리파이에 blynk python버전 설치

blynk를 사용하기 위해서는 blynk 라이브러리를 pip를 이용해 설치해 주어야 합니다.

파이썬2 버전을 사용하시는 경우 pip, 파이썬3 버전을 사용하면 pip3로 설치합니다.
여기서는 python3를 사용하기 때문에 pip3로 설치를 해 주시면 됩니다.

pip3 install blynklib 명령어로 라이브러리를 설치해 주시면 됩니다.

※ 만일 권한 관련 오류가 발생하는 경우 명령어 앞에 sudo를 입력해 주시면 됩니다.



```
pi@raspberrypi: ~/iotkit
File Edit Tabs Help
pi@raspberrypi:~/iotkit $ pip3 install blynklib
```

그림 24 blynk 라이브러리 설치

blynk 라이브러리는 위 명령어로 간단히 다운로드 할 수 있습니다.

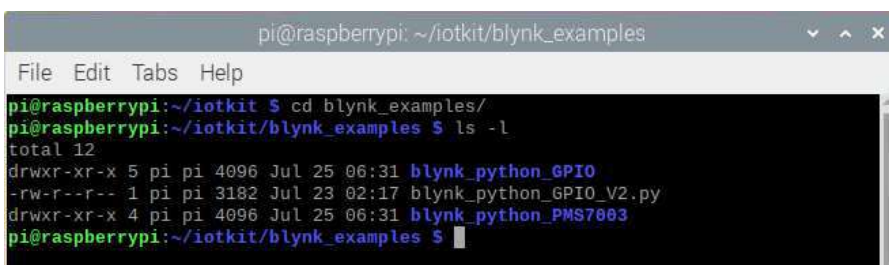
이 외에 추가로 타이머 및 PMS7003사용을 위해 라이브러리를 다운로드 해 주어야 합니다.

타이머: <https://github.com/blynkkk/lib-python/blob/master/blynktimer.py>

PMS7003: <https://github.com/eleparts/PMS7003/blob/master/PMS7003.py>

다만, **스크립트 파일(start.sh - 그림 2 다운로드 스크립트 실행)을 실행 해 주신 경우 자동으로 다운로드가 진행**되어 별도의 추가 다운로드 명령어 입력 없이 바로 예제 실행이 가능합니다.

blynk 관련 예제 코드는 blynk_examples 디렉터리에 있습니다.



```
pi@raspberrypi: ~/iotkit/blynk_examples
File Edit Tabs Help
pi@raspberrypi:~/iotkit $ cd blynk_examples/
pi@raspberrypi:~/iotkit/blynk_examples $ ls -l
total 12
drwxr-xr-x 5 pi pi 4096 Jul 25 06:31 blynk_python_GPIO
-rw-r--r-- 1 pi pi 3182 Jul 23 02:17 blynk_python_GPIO_V2.py
drwxr-xr-x 4 pi pi 4096 Jul 25 06:31 blynk_python_PMS7003
pi@raspberrypi:~/iotkit/blynk_examples $
```

그림 25 blynk 예제코드

위 2개 디렉터리가 blynk에 대한 예제 코드이며, 그림 25에 보이는 blynk_python_GPIO_V2 예제는 blynk_python_GPIO 예제 디렉터리에 복사/이동되어 해당 디렉터리에서 실행 가능합니다.

3. blynk로 GPIO제어하기

① 하드웨어 구성

스위치 및 릴레이 회로 구성은 위 [나)라즈베리파이 제어]와 동일하게 해 주시면 됩니다.

※ 참고 1 - 그림 5 라즈베리파이에 LED및 스위치 연결

※ 참고 2 - 그림 11 릴레이보드 연결

② blynk 위젯 설정 및 배치

blynk 위젯은 프로젝트를 생성해 준 뒤 빈 화면을 누르거나 상단의 + 모양을 눌러 추가해 줄 수 있습니다.

위젯을 추가하는 화면은 아래와 같으며, 여기서 Button 3개, Display 1개, LED 1개를 추가해 주시면 됩니다.

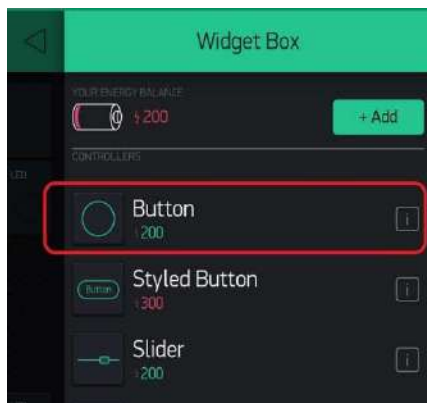


그림 27 위젯 추가 1

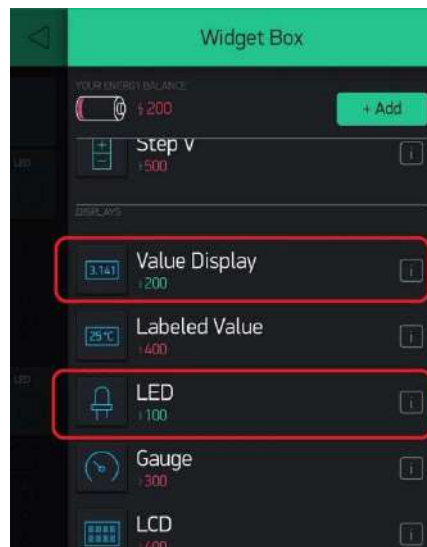


그림 26 위젯 추가 2

각 위젯을 추가해 준 뒤 위젯의 핀 설정을 바꾸어 주어야 합니다.

GPIO 핀 직접 제어 기능은 현재 파이썬 0.2.4 버전에서는 지원이 되지 않아 Virtual Pins 기능을 이용하여 명령 프로토콜을 전송/수신하도록 합니다.

아래 이미지를 참고해서 설정해 주시면 됩니다.

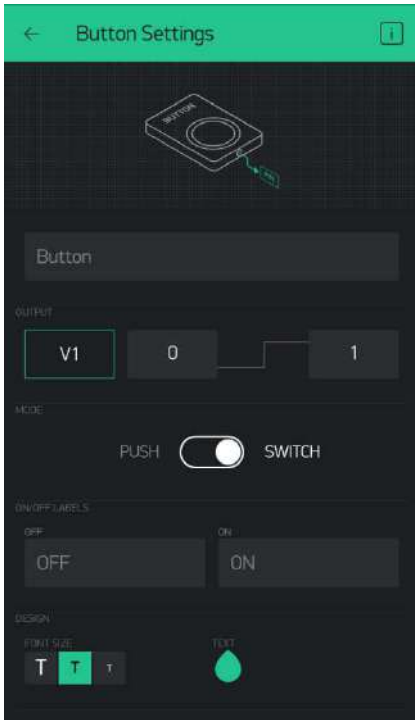


그림 30 Button



그림 29 Display

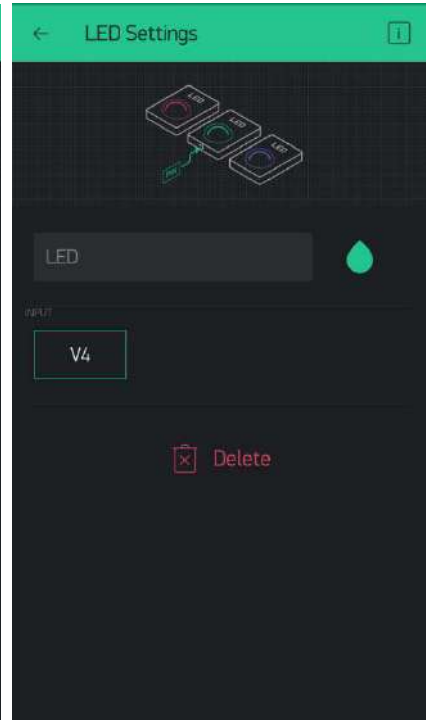


그림 28 LED

Button은 각각 **V1, V2, V5**로 설정하고 **MODE**를 **SWITCH**로 변경해 줍니다.
Display는 **V3**로 설정한 뒤 **READING RATE**를 **1sec**로 변경해 줍니다.
LED는 **V4**핀으로 연결해 주시면 됩니다.

위 내용대로 설정 후 크기 등을 조절해 적당해 배치해 주면 됩니다.



그림 31 GPIO 위젯 배치

그리고 우측 위 삼각형 모양 재생버튼을 눌러 통신을 시작해 주면 됩니다.

③ 예제코드 실행

blynk 앱을 동작시켜 주었다면 라즈베리파이의 동작을 제어해 줄 예제 코드를 실행해 주어야 합니다. 먼저 **blynk_example/blynk_python_GPIO** 디렉터리로 이동해 줍니다.

아래 디렉터리에서 **blynk_python_GPIO.py** 및 **blynk_python_GPIO_V2.py**가 예제 코드 파일입니다.

```

pi@raspberrypi: ~/iotkit/blynk_examples/blynk_python_GPIO
File Edit Tabs Help
pi@raspberrypi:~/iotkit/blynk_examples/blynk_python_GPIO $ cd ..
pi@raspberrypi:~/iotkit/blynk_examples $ cd blynk_python_GPIO/
pi@raspberrypi:~/iotkit/blynk_examples/blynk_python_GPIO $ ls -l
total 212
drwxr-xr-x 2 pi pi 4096 Jul 23 09:59 소스 코드
-rw-r--r-- 1 pi pi 4716 Jul 23 09:59 blynk_GPIO.fzz
-rw-r--r-- 1 pi pi 180188 Jul 23 09:59 blynk_GPIO.png
-rw-r--r-- 1 pi pi 3016 Jul 23 09:59 blynk_python_GPIO.py
-rw-r--r-- 1 pi pi 3182 Jul 25 06:31 blynk_python_GPIO_V2.py
-rw-r--r-- 1 pi pi 3935 Jul 25 06:31 blynktimer.py
drwxr-xr-x 3 pi pi 4096 Jul 23 09:59 old_version
-rw-r--r-- 1 pi pi 2826 Jul 23 09:59 README.md
-rwxr-xr-x 1 pi pi 238 Jul 23 09:59 start.sh
pi@raspberrypi:~/iotkit/blynk_examples/blynk_python_GPIO $
  
```

그림 32 blynk GPIO 예제

blynk_python_GPIO.py는 릴레이 동작이 없는 예제이며, V2 버전이 릴레이 기능이 추가된 예제 코드입니다.

해당 예제 코드를 동작하기 전에 위에서 이메일로 받은 토큰(그림 23 토큰 주소 메일)을 입력해 주어야 합니다.

편집기를 이용해 예제코드를 열어 수정해 줍니다.

nano를 이용하면 **nano blynk_python_GPIO.py** 명령어로 열고 토큰을 넣어준 뒤 **컨트롤+O**, **컨트롤+X**로 저장 후 종료해 주시면 됩니다.

```

pi@raspberrypi: ~/iotkit/blynk_examples/blynk_python_GPIO
File Edit Tabs Help
GNU nano 3.2 blynk_python_GPIO.py
기반 코드 및 필수 라이브러리 - blynk / python (Blynk Python Library V0.2.4)
https://github.com/blynkkk/lib-python
"""
import blynklib
import blynktimer
import RPi.GPIO as GPIO

#이메일로 받은 토큰을 여기에 추가
BLYNK_AUTH = 'YourAuthToken'

# Initialize Blynk
blynk = blynklib.Blynk(BLYNK_AUTH)

import blynktimer
import RPi.GPIO as GPIO

#이메일로 받은 토큰을 여기에 추가
BLYNK_AUTH = '40cbc...'

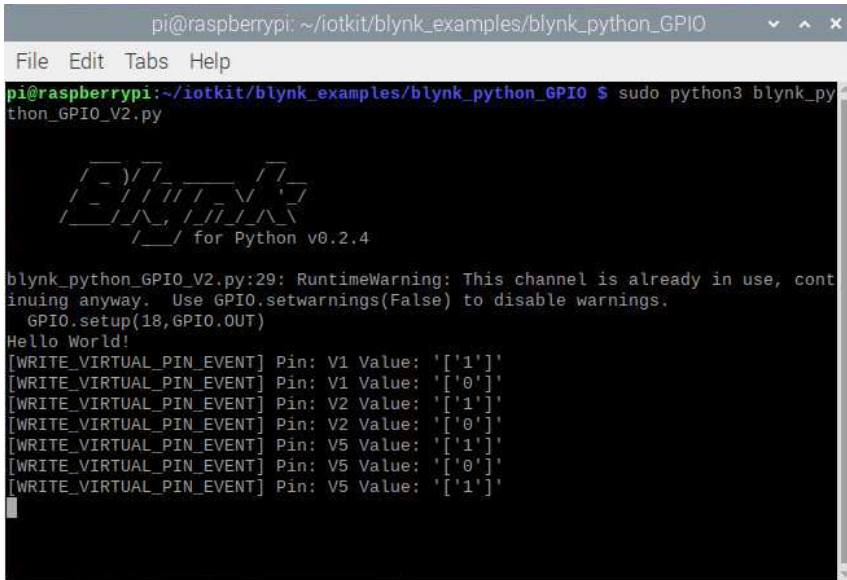
# Initialize Blynk
blynk = blynklib.Blynk(BLYNK_AUTH)
  
```

그림 33 토큰 입력

마찬가지로 blynk_python_GPIO_V2.py 파일도 동일하게 수정해 주어야 합니다.

그리고 **sudo python3 blynk_python_GPIO.py** 명령어로 프로그램을 실행시켜 줍니다.

릴레이를 연결하였다면 위 프로그램 대신 **sudo python3 blynk_python_GPIO_V2.py**로 V2 프로그램을 실행시켜 주시면 V5버튼을 사용할 수 있습니다.



```

pi@raspberrypi: ~/iotkit/blynk_examples/blynk_python_GPIO
File Edit Tabs Help
pi@raspberrypi:~/iotkit/blynk_examples/blynk_python_GPIO $ sudo python3 blynk_py
thon_GPIO_V2.py

  _ _ _ _ _
 / _ _ _ _ \
(  _ _ _ _ )
/_ _ _ _ _ \
 \ _ _ _ _ /
  _ _ _ _ _
  for Python v0.2.4

blynk_python_GPIO_V2.py:29: RuntimeWarning: This channel is already in use, cont
inuing anyway. Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(18,GPIO.OUT)
Hello World!
[WRITE_VIRTUAL_PIN_EVENT] Pin: V1 Value: '['1']'
[WRITE_VIRTUAL_PIN_EVENT] Pin: V1 Value: '['0']'
[WRITE_VIRTUAL_PIN_EVENT] Pin: V2 Value: '['1']'
[WRITE_VIRTUAL_PIN_EVENT] Pin: V2 Value: '['0']'
[WRITE_VIRTUAL_PIN_EVENT] Pin: V5 Value: '['1']'
[WRITE_VIRTUAL_PIN_EVENT] Pin: V5 Value: '['0']'
[WRITE_VIRTUAL_PIN_EVENT] Pin: V5 Value: '['1']'

```

그림 34 blynk GPIO 동작

프로그램을 실행시켜 준 뒤 스마트폰 blynk앱의 버튼을 누르거나 라즈베리파이에 연결된 스위치를 눌러 보시면 이벤트 로그가 표시되면서 각 버튼에 연동되어 있는 LED나 릴레이가 구동되는 것을 확인하실 수 있습니다.

4. blynk로 먼지센서 데이터 받아오기

① 하드웨어 구성

위와 동일하게 PMS7003 먼지센서를 인터페이스 보드와 USB to UART 케이블에 연결하여 라즈베리파이의 USB에 연결해 주시면 됩니다.



그림 35 PMS7003 먼지센서 연결

② blynk 위젯 설정 및 배치

PMS7003 위젯 배치는 센서 값을 출력할 LCD 3개 (PM 1, PM 2.5, PM 10)와 데이터 수신 에러 LED 1개로 구성됩니다.

사용 가능한 LCD의 종류는 두 가지가 있습니다.

위 **Value Display**는 글자 출력 기능만 제공합니다.

아래 **Labeled Value**는 값 설명 문구 같은 기본 문자를 수신 값과 별도로 설정해 주는 추가 기능이 있습니다.

※ 기존 GPIO와 핀 배치 등이 중복되지 않아 따로 삭제하지 않고 추가해 주어도 됩니다.

다만, 기본 제공 에너지에 제한이 있어 GPIO 위젯을 삭제하지 않고 Labeled Value 위젯을 사용하려면 에너지를 추가(결제 필요)해 주어야 합니다.

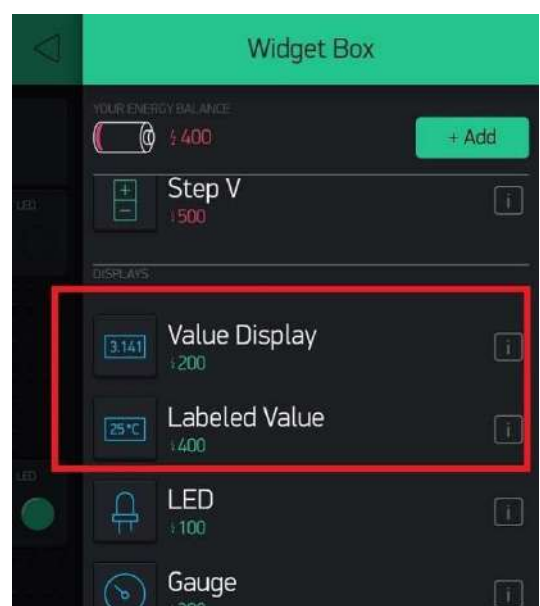


그림 36 LCD 위젯

③ 예제코드 실행

blynk 예제는 blynk_example/blynk_python_PMS7003에 다운로드 되어 있습니다.

해당 경로로 이동하여 예제 코드에 토큰을 입력해 주어야 합니다.

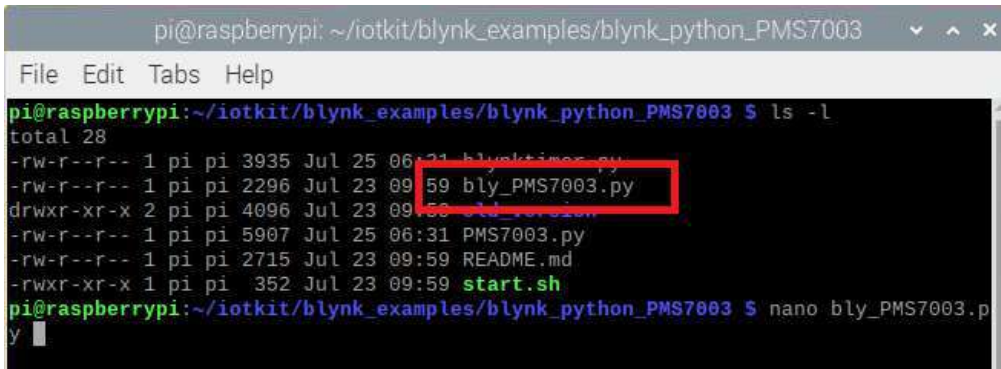


그림 37 blynk PMS7003 예제

nano 혹은 기타 에디터를 이용해 **bly_PMS7003.py** 파일을 열어 줍니다.

그리고 **BLYNK_AUTH = 'YourAuthToken'** 항목에 토큰을 입력해 줍니다.

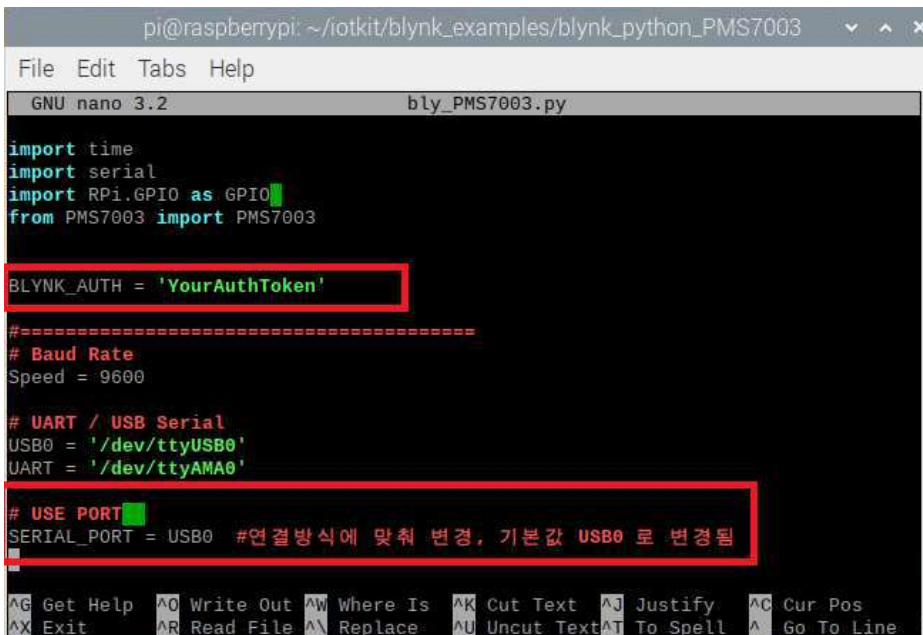


그림 38 blynk PMS7003 예제 수정

그리고 연결 방식에 맞춰 위 사진의 아래 USE PORT를 확인해 주시면 됩니다.

※ 기본값 **USB0**, 상단 [먼지센서 연결하기] 항목 참고.

그리고 추가로 LCD에 따라 예제 코드의 주석처리를 변경해 주시면 됩니다.

```

if(dust.protocol_chk(buffer)):
    data = dust.unpack_data(buffer)

    # Labeled Value (Display)
    blynk.virtual_write(7, data[dust.DUST_PM1_0_ATM])
    blynk.virtual_write(8, data[dust.DUST_PM2_5_ATM])
    blynk.virtual_write(9, data[dust.DUST_PM10_0_ATM])
    # Value Display
    #blynk.virtual_write(7, ("PM1.0 : " + str(data[dust.DUST_PM1_0_ATM])))
    #blynk.virtual_write(8, ("PM2.5 : " + str(data[dust.DUST_PM2_5_ATM])))
    #blynk.virtual_write(9, ("PM10 : " + str(data[dust.DUST_PM10_0_ATM])))

    blynk.virtual_write(6, '0')

print("send - PM1.0: %d | PM2.5: %d | PM10: %d" %(data[dust.DUST_PM1_0_ATM],data[dust.DUST_PM2_5_ATM],data[dust.DUST_PM10_0_ATM]))
# Labeled Value (Display)
#blynk.virtual_write(7, data[dust.DUST_PM1_0_ATM])
#blynk.virtual_write(8, data[dust.DUST_PM2_5_ATM])
#blynk.virtual_write(9, data[dust.DUST_PM10_0_ATM])
# Value Display
blynk.virtual_write(7, ("PM1.0 : " + str(data[dust.DUST_PM1_0_ATM])))
blynk.virtual_write(8, ("PM2.5 : " + str(data[dust.DUST_PM2_5_ATM])))
blynk.virtual_write(9, ("PM10 : " + str(data[dust.DUST_PM10_0_ATM])))

blynk.virtual_write(6, '0')

```

그림 39 LCD에 따른 송신 데이터 설정

Labeled Value를 사용 시 위의 3줄을, Value Display를 사용 시 아래 3줄을 사용하도록 합니다.

위의 3줄은 숫자 값만 전송하는 예제로 Labeled Value를 사용하여 blynk앱의 위젯에 'PM2.5 :' 같은 안내 문구가 이미 있는 경우 사용합니다.

아래 3줄은 Value Display를 사용하여 값 안내문구가 없는 경우 값 안내 문구를 문자열로 처리하여 같이 전송해 주는 방법입니다.

사용하는 위젯에 맞게 선택해 주시면 출력 화면이 더 깔끔해 집니다.

※ 사용하지 않는 부분에 주석(앞에 # 기호)처리를 해 주시면 됩니다. – 기본값 Labeled Value
> 수정하지 않아도 동작에는 문제가 없습니다.

설정이 완료되면 저장 후 프로그램을 실행해 주시면 됩니다.

명령어는 **sudo python3 bly_PMS7003.py** 입니다.

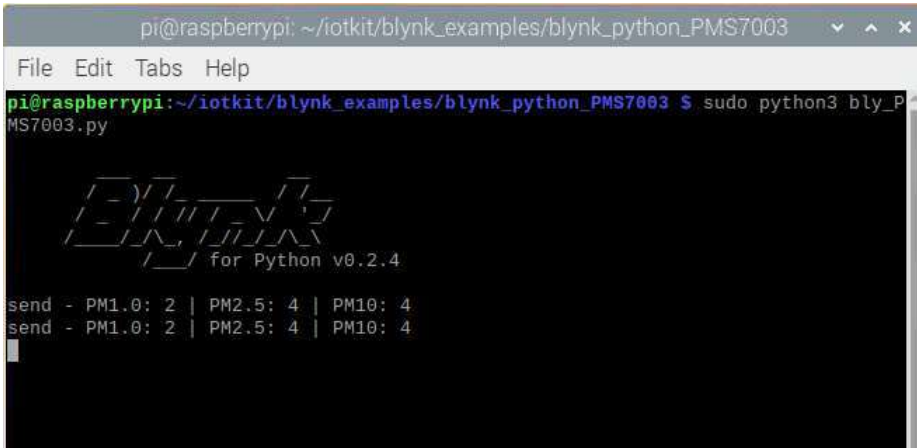


그림 40 blynk 먼지센서 예제 실행.

라즈베리파이의 터미널 창에서는 위와 같이 값을 전송하게 되며, blynk 앱에서는 아래와 같이 데이터를

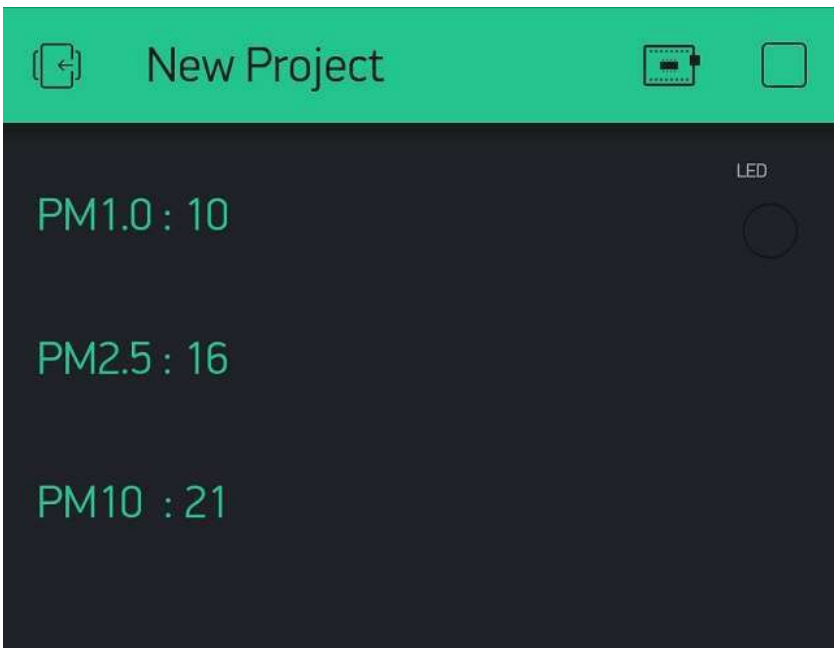


그림 41 blynk 먼지 데이터 수신

마) 통합하여 IOT 환경 구축하기

1. 하드웨어 및 blynk위젯 구성

① GPIO 구성

GPIO연결은 위에서 한 것과 동일하게 구성해 주시면 됩니다.
간단히 정리하면 아래와 같습니다.

GPIO 번호	연결 항목
20	LED
21	LED
18	릴레이
16	스위치

표 2 GPIO 구성

② blynk 위젯 구성

Blynk 위젯 구성 또한 위에서 설정했던 것과 동일하게 배치해 주시면 됩니다.
다만, Labeled Value 위젯 사용 시 에너지가 부족할 수 있으며 필요 시 앱 내 결제를 이용해 에너지 총량을 증가시킬 수 있습니다.

위젯 구성은 아래와 같습니다.

핀 설정	위젯 타입	위젯 설정
V1	Button	MODE : SWITCH
V2	Button	MODE : SWITCH
V3	Value Display	READING RATE : 1sec
V4	LED	-
V5	Button	MODE : SWITCH
V6	LED	-
V7	Value Display / Labeled Value	READING RATE : PUSH
V8	Value Display / Labeled Value	READING RATE : PUSH
V9	Value Display / Labeled Value	READING RATE : PUSH

표 3 위젯 구성

위와 같이 구성을 해 주시고 적당히 위치에 맞게 배치해 주시면 됩니다.

구성 화면입니다.



그림 42 위젯 구성

위젯의 위치는 자유롭게 해 주시면 됩니다.

2. 예제 실행

예제 코드는 마찬가지로 토큰을 넣어주고 실행해 주어야 합니다.

iotkit.py 예제 코드 파일을 열어 수정해 주어야 합니다.,

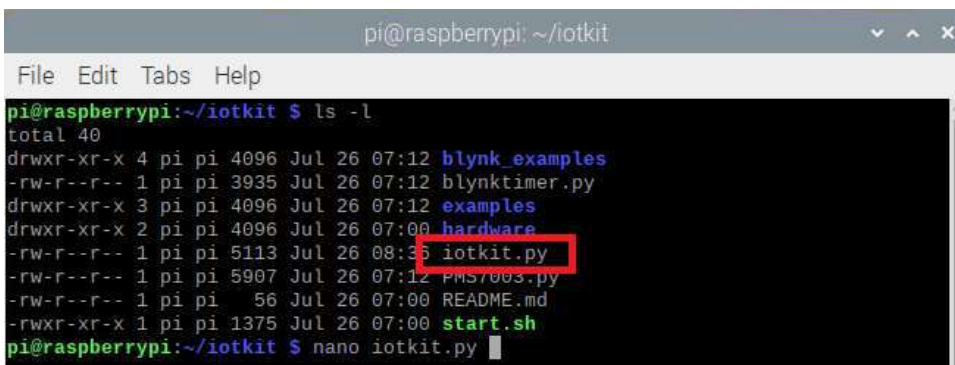


그림 43 iotkit.py

여기서는 nano 편집기를 사용합니다.

예제 파일을 열고 토큰을 넣어 줍니다.

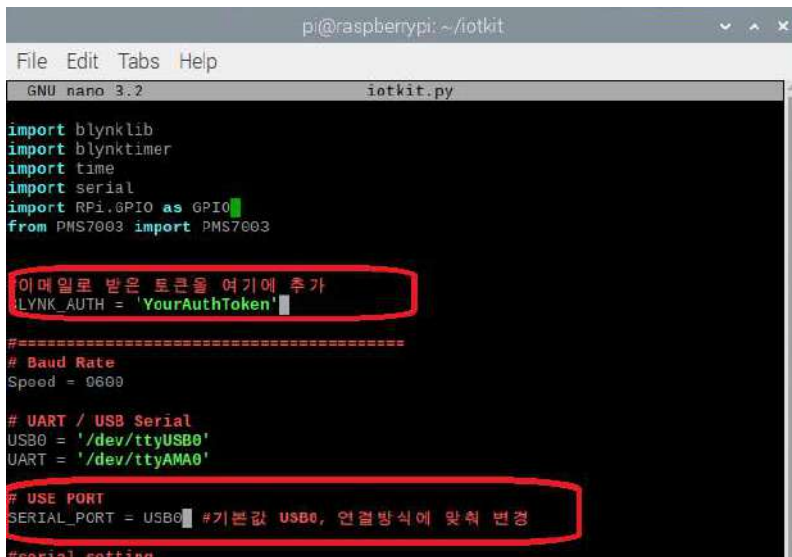


그림 44 토큰입력 및 포트 확인

먼지센서를 사용하기 위해 USB포트도 확인을 해준 뒤 저장 후 종료합니다.

예제 코드 실행은 **sudo python3 iotkit.py** 입니다.

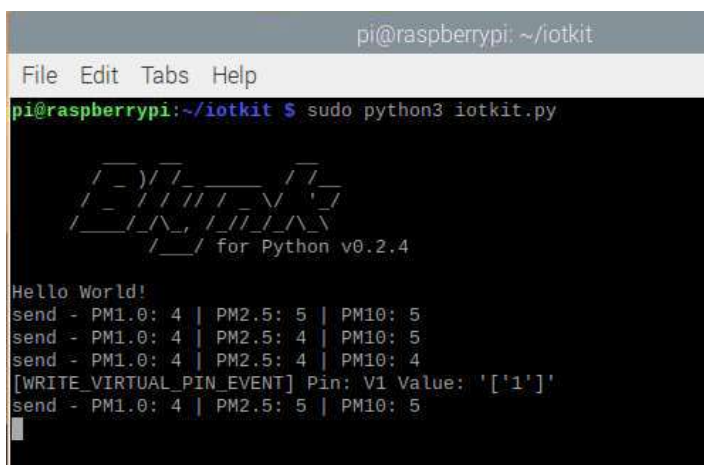


그림 46 iotkit.py 실행

예제를 실행하고 앱의 버튼을 누르면 [그림 45 blynk 실행], [그림 46 iotkit.py 실행]과 같이 입력 데이터가 전달되고 연결된 회로가 동작하게 됩니다.



그림 45 blynk 실행