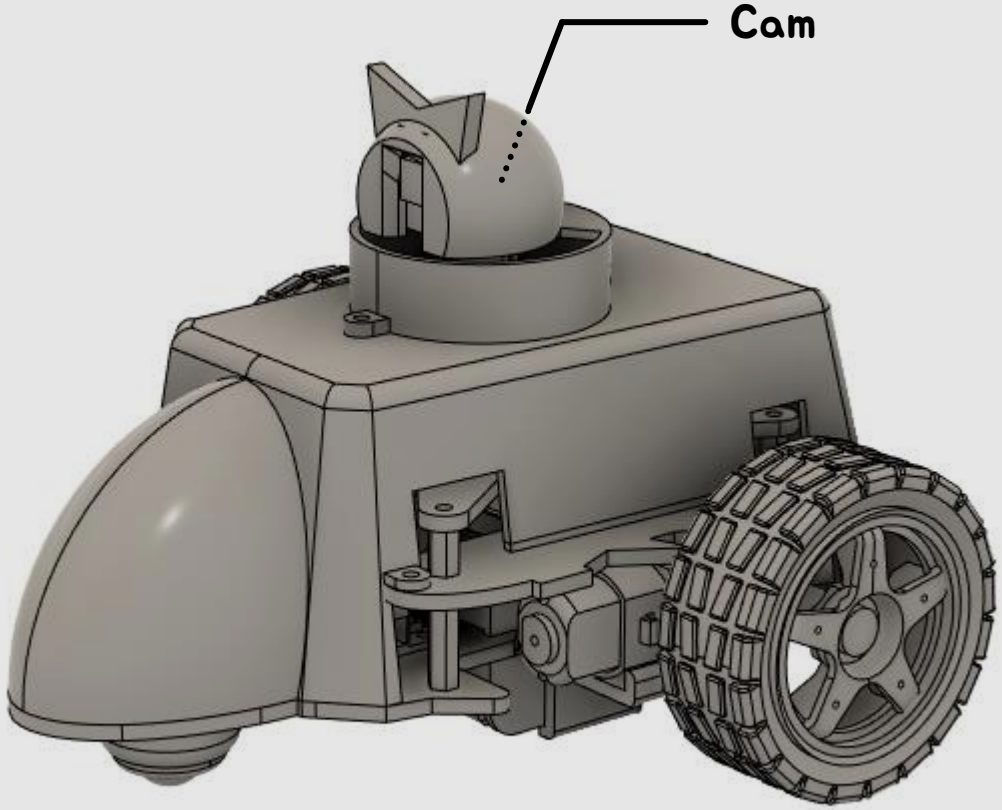
Raspberry Pi

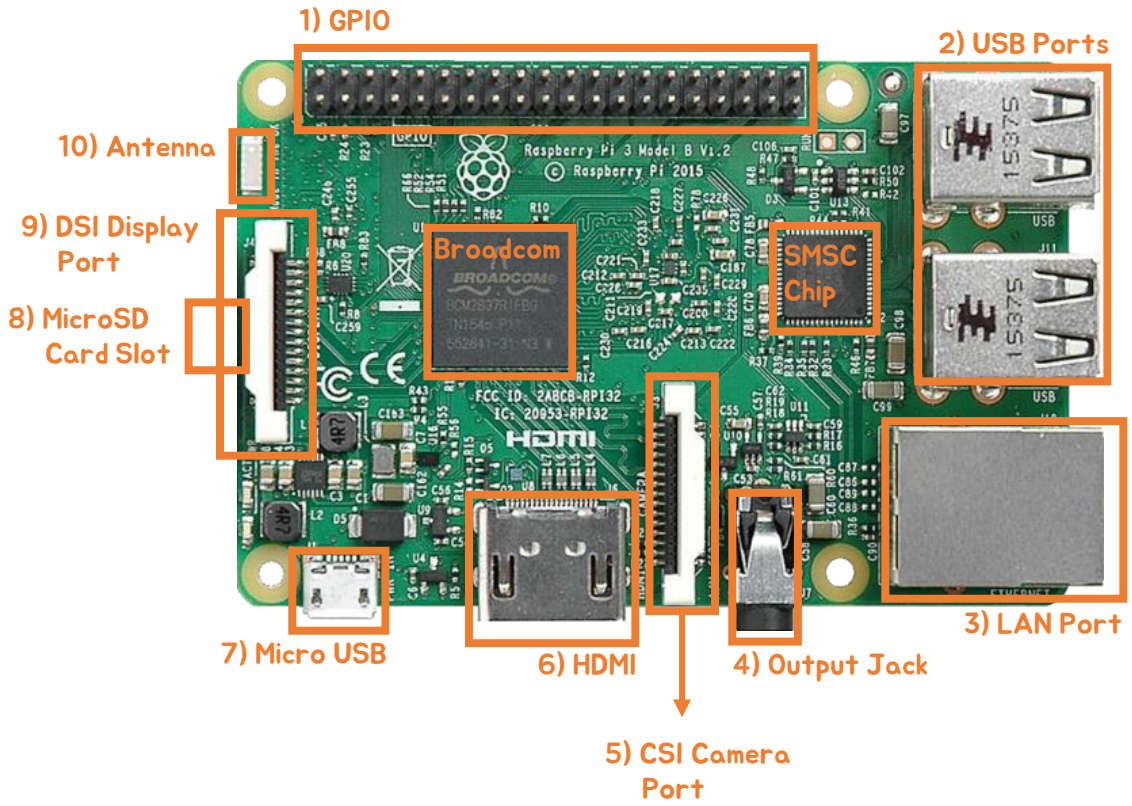
IoT CCTV RC Car



Intro

라즈베리파이란?

2012년 2월 영국의 라즈베리파이 재단에서 교육적 목적으로 만든 싱글 보드 컴퓨터(SBC)



라즈베리파이 3B Diagram

1) GPIO: 디지털 값의 입력/출력에 사용
2) USB Ports: USB 장치를 연결할 수 있는 포트
3) LAN Port: 이더넷 포트
4) Output Jack: 오디오 연결 단자
5) CSI: 카메라 연결 인터페이스
6) HDMI: HDMI 단자
7) Micro USB: 전원 단자
8) Micro SD Card Slot: 마이크로 SD카드 장착 슬롯
9) DSI: 디스플레이 장치 인터페이스
10) Antenna: 와이파이 안테나

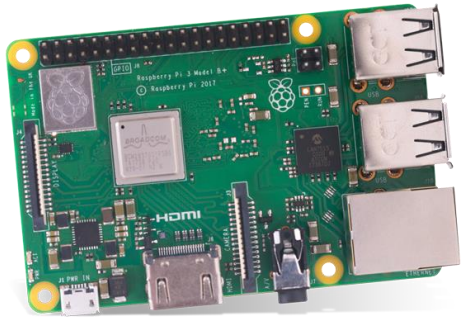
Intro

라즈베리파이 SPEC

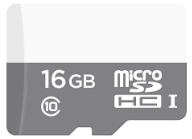
모델명	4 MODEL B	3 MODEL B+	3 MODEL B	ZERO
출시일	19.06.24	18.03.14	16.01.29	15.11.25
SOC	Broadcom BCM2711	BCM2837B0	BCM2837	BCM2835
CPU	Quad core Cortex-A72(ARM v8) 64-bit SoC @1.5GHz	Quad Cortex-A53 @1.4GHz	Quad Cortex-A53 @1.2GHz	ARM11 @1GHz
RAM	1GB, 2GB, 4GB, 8GB LPDDR4-3200 SDRAM (depending on model)	1GB SDRAM	1GB SDRAM	512MB SDRAM
Storage	Micro-SD			
Ethernet	Gigabit Ethernet	Gigabit	10/100	None
Wireless	2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0 BLE	WLAN/Bluetooth 4.2	WLAN/Bluetooth 4.0	None ZERO W의 경우 Wireless 지원
GPIO	40	40	40	40

Intro

준비물



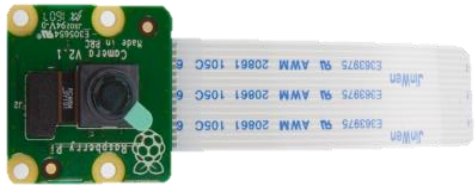
라즈베리파이 3B 이상



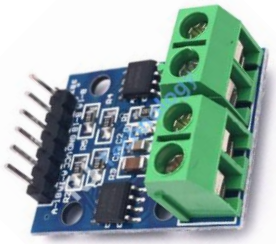
마이크로SD Card
16GB 이상

마이크로SD Card
리더기

마이크로 USB



라즈베리파이용 카메라



L9110 모터 드라이버

DC모터 x 2

서보 모터

라커 스위치

배터리 홀더

리튬 배터리

리튬 배터리 충전모듈

+ 조립용 물품

+ WIFI 공유기 관리자 ID/PWD

+ 제어할 PC (라즈베리파이를 PC로 사용시에 필요없음)

Install

Install

Raspberry OS

Install Raspberry Pi OS using Raspberry Pi Imager

Raspberry Pi Imager is the quick and easy way to install Raspberry Pi OS and other operating systems to a microSD card, ready to use with your Raspberry Pi. Watch our 45-second video to learn how to install an operating system using Raspberry Pi Imager.

Download and install Raspberry Pi Imager to a computer with an SD card reader. Put the SD card you'll use with your Raspberry Pi into the reader and run Raspberry Pi Imager.

Download for Windows

Download for macOS

Download for Ubuntu for x86

To install on **Raspberry Pi OS**, type
`sudo apt install rpi-imager`
in a Terminal window.

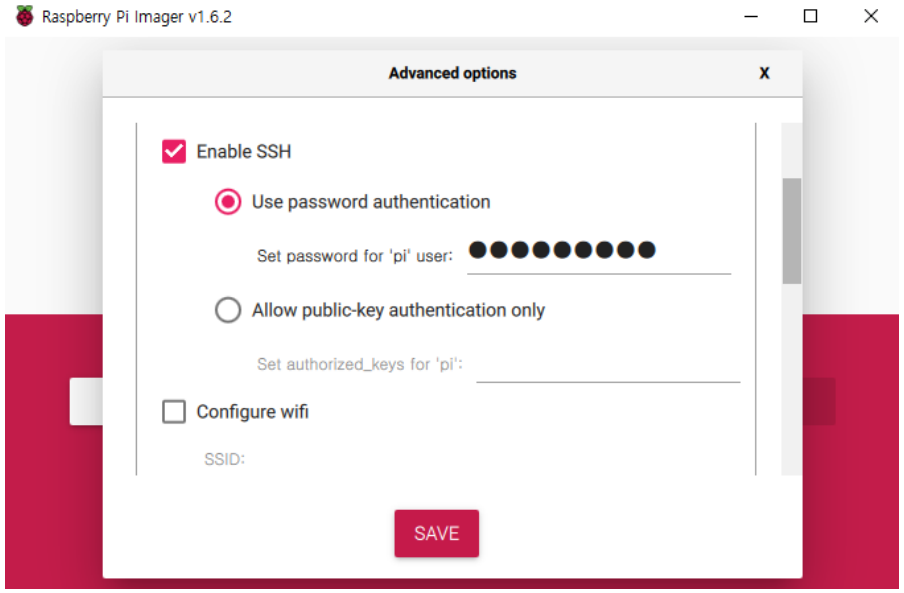
참고 링크:
https://www.youtube.com/watch?v=ntaXWS8Lk34

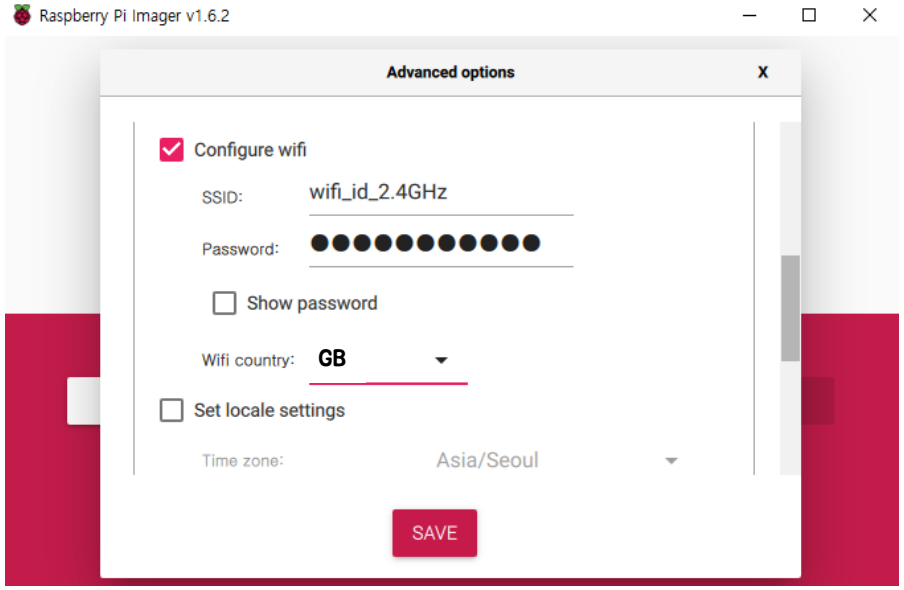
https://www.raspberrypi.com/software/

Install

Raspberry OS

Ctrl + Shift + x





CHOOSE OS → Use custom

(최신 버전에서 카메라 호환의 문제가 있는 것으로 판단)

구버전: http://downloads.raspberrypi.org/raspbian/images/raspbian-2020-02-14/

Install

Raspberry OS

Install

program

Install

Python

https://www.python.org/

1991년 귀도 반 로섬이 발표한 고급 프로그래밍 언어

언어 이름은 귀도가 좋아하는 코미디 Monty Python`s Flying Circus에서 따옴

인터프리식 언어, 객체지향적, 동적 타이핑 대화형 언어

비영리 파이썬 소프트웨어 재단이 관리하고 있으며, 사실상 표준은 C언어로 구현된 Cython

귀도 반 로섬

*인터프리식 언어: 코드를 한줄 한줄 읽어가며 명령을 처리

**객체지향적: 파이썬은 Object(객체)로 이루어져 있으며, 객체 지향 프로그래밍을 추구하고 있다.

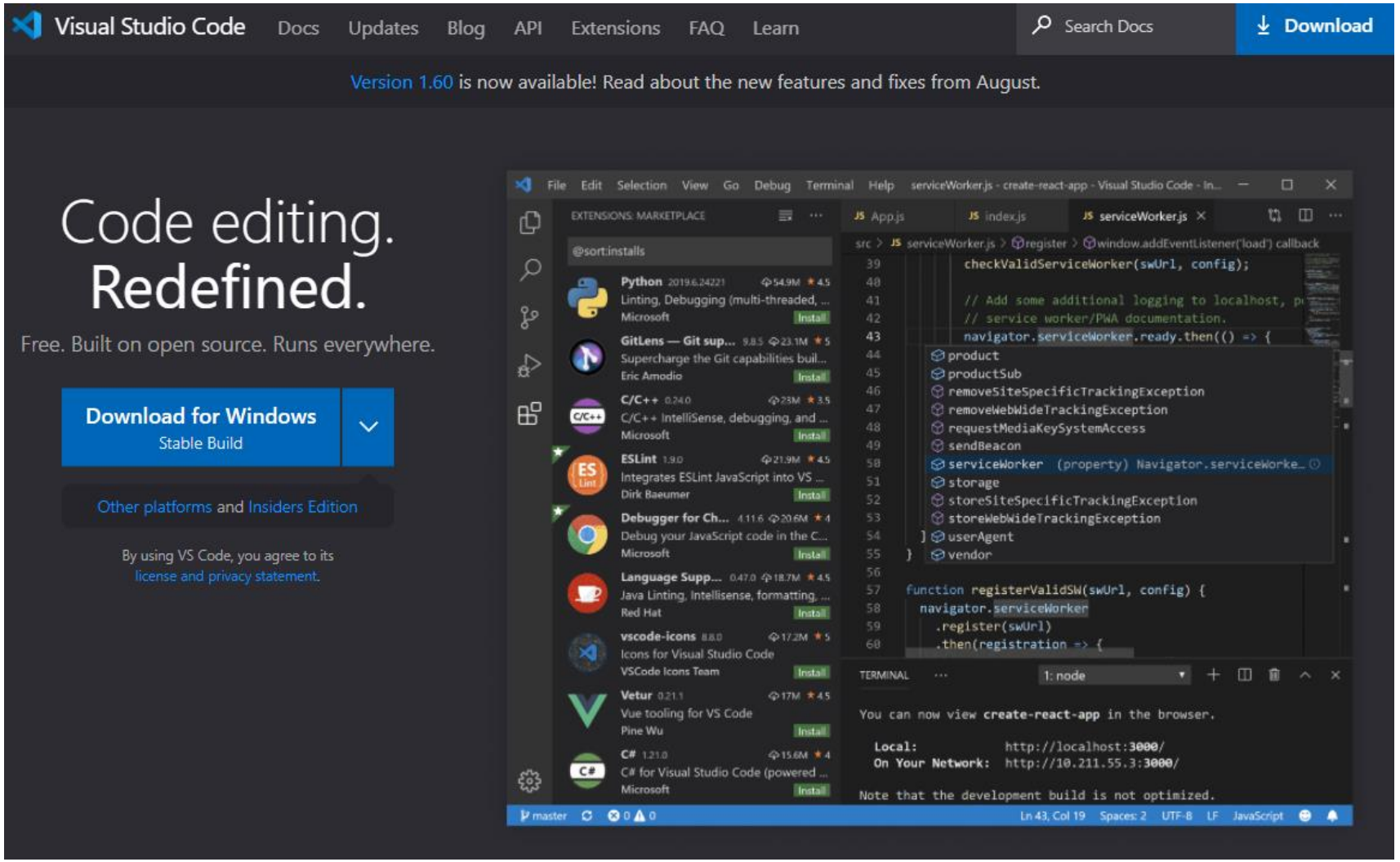
OOP(Object Oriented Programming, 객체 지향 프로그래밍): 프로그래밍에 필요한 데이터를 추상화하여 상태와 행위를 가진 객체를 만들고, 객체들 간의 유기적 상호작용을 통해 로직을 구성하는 방법

***동적 타이핑 대화형 언어: 파이썬은 변수를 만들 때 변수의 타입을 먼저 지정해주지 않습니다. 그리고, 프로그램 자체가 영어 친화적으로 대화하듯 되어 되어 있습니다.

Install

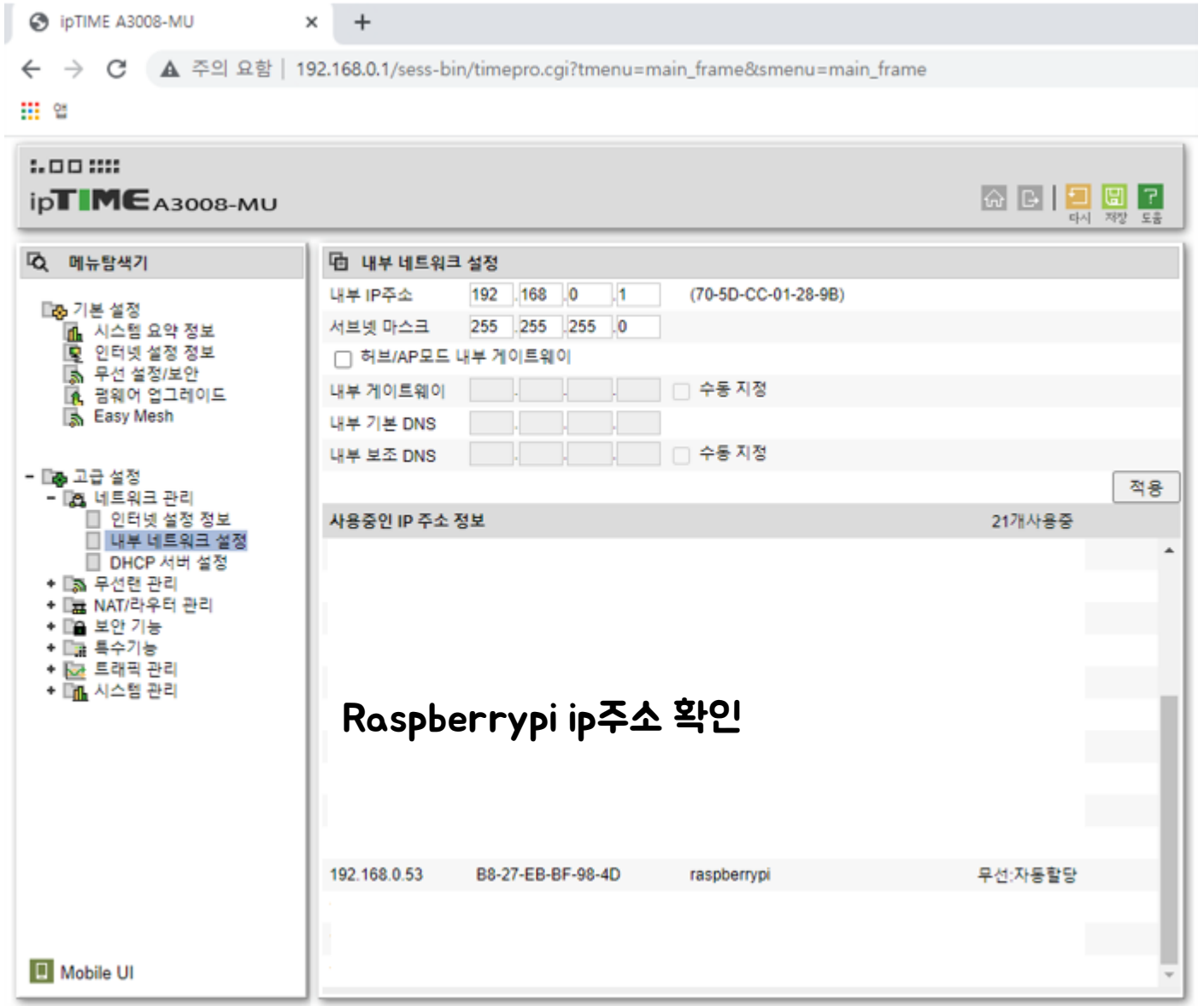
코드 편집툴: VS Code

https://code.visualstudio.com/



Install

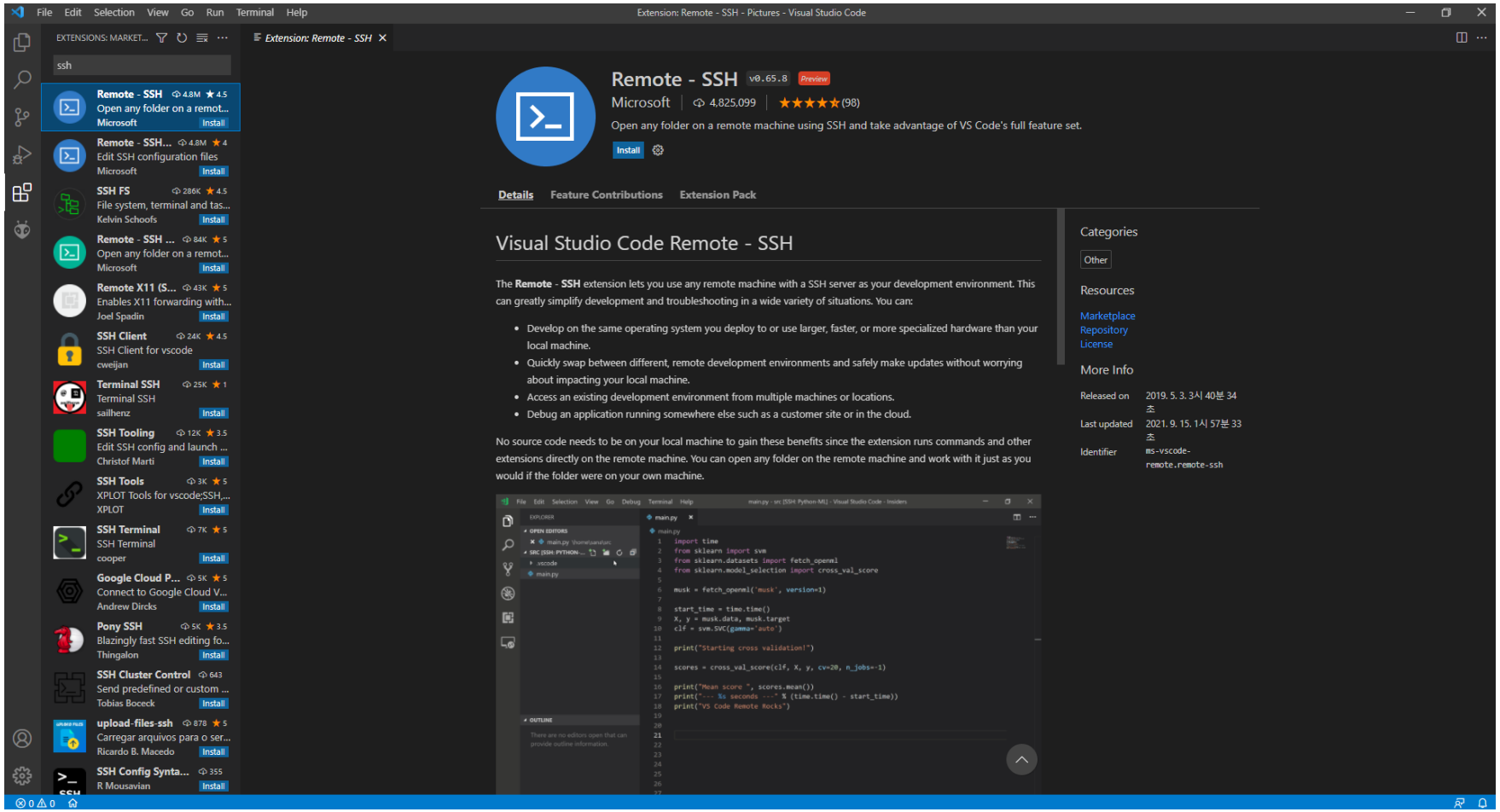
Raspberry Pi, IP 확인



Install

VS Code SSH Extension

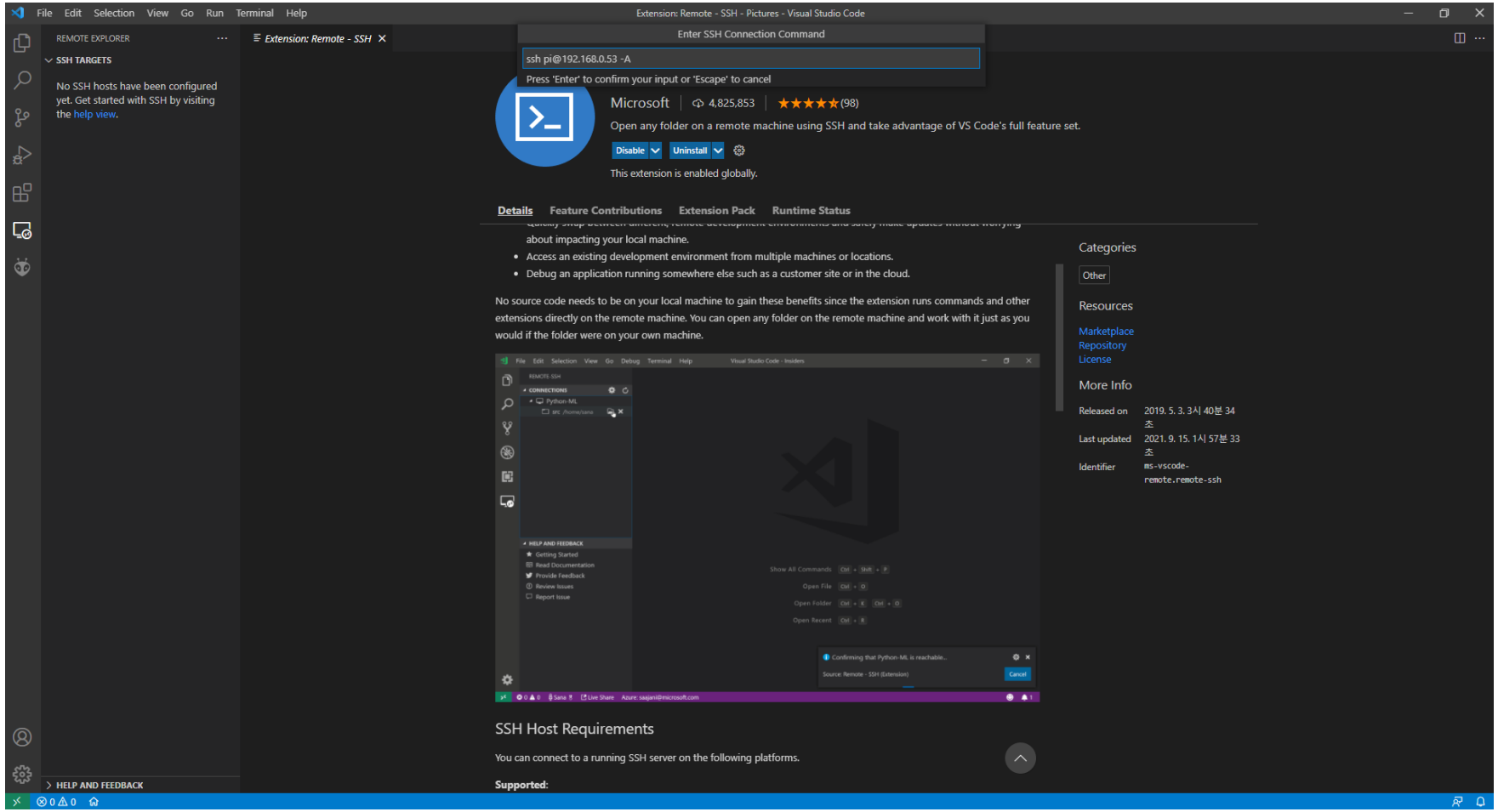
VS Code Extension에 `Remote -SSH`를 설치하여 라즈베리파이와 연결



Install

VS Code SSH Extension

ssh id@ip주소 -A

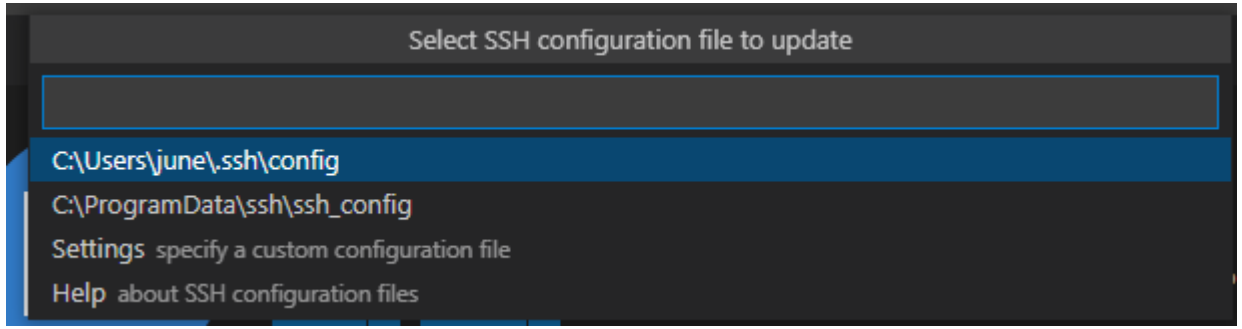


Install

VS Code SSH Extension

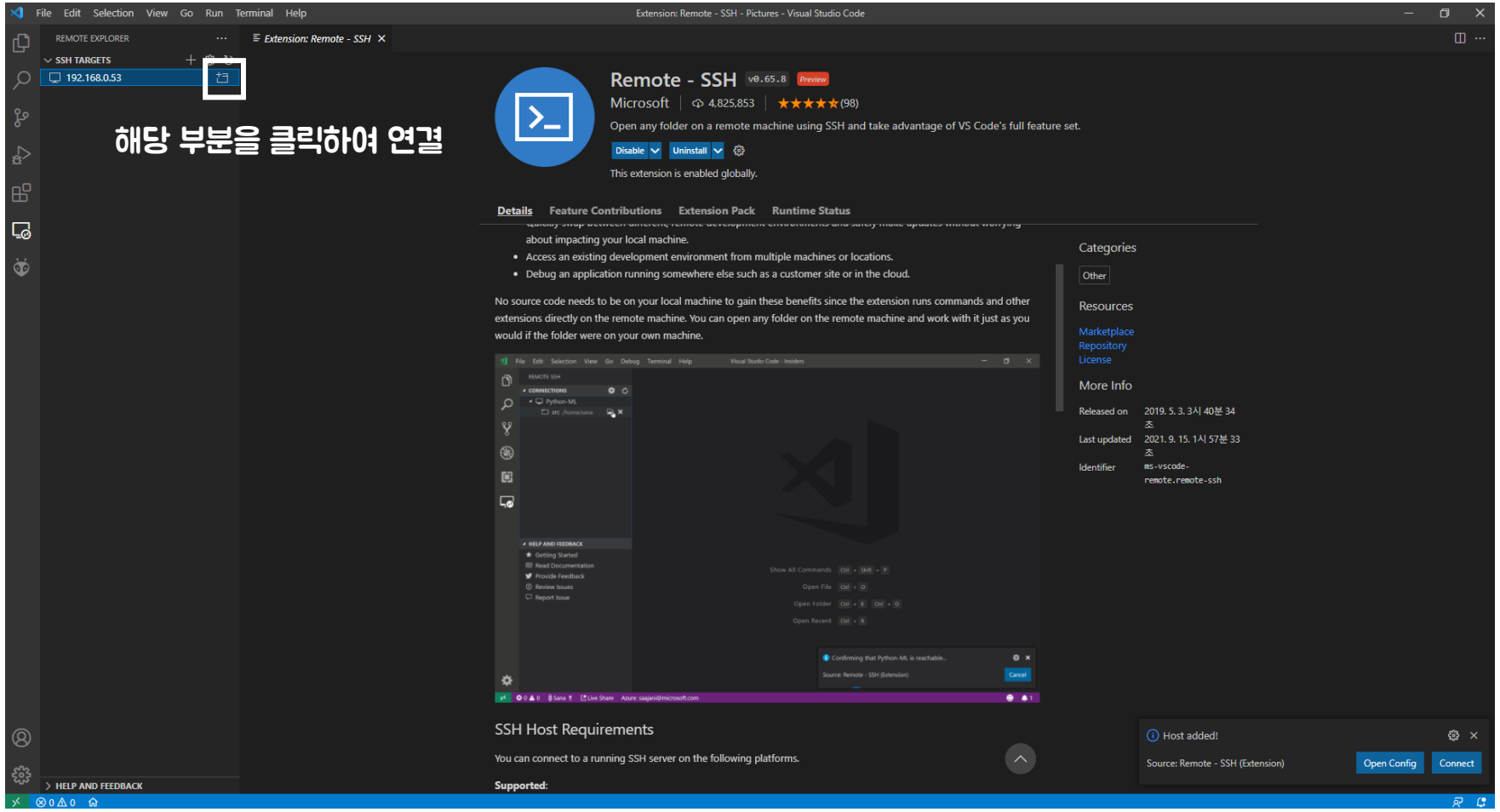
설정할 파일 위치 저장

기본 처음으로 설정해서 진행함



Install

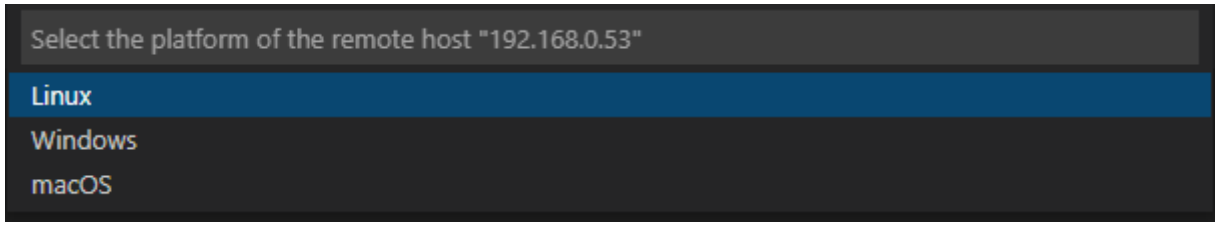
VS Code SSH Extension



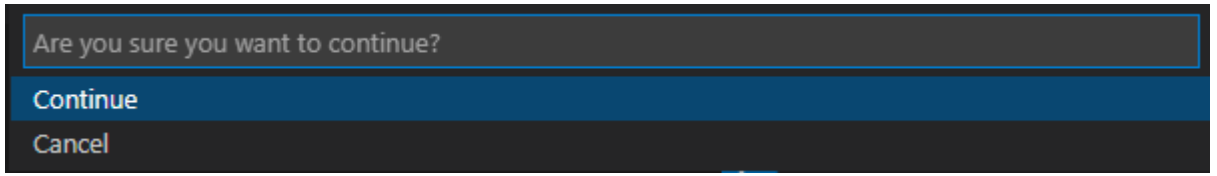
Install

VS Code SSH Extension

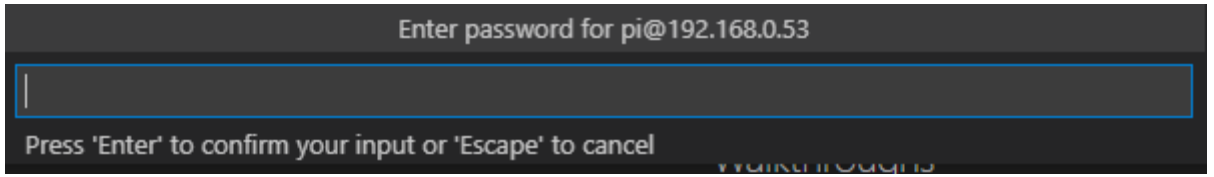
Linux로 설정



Continue



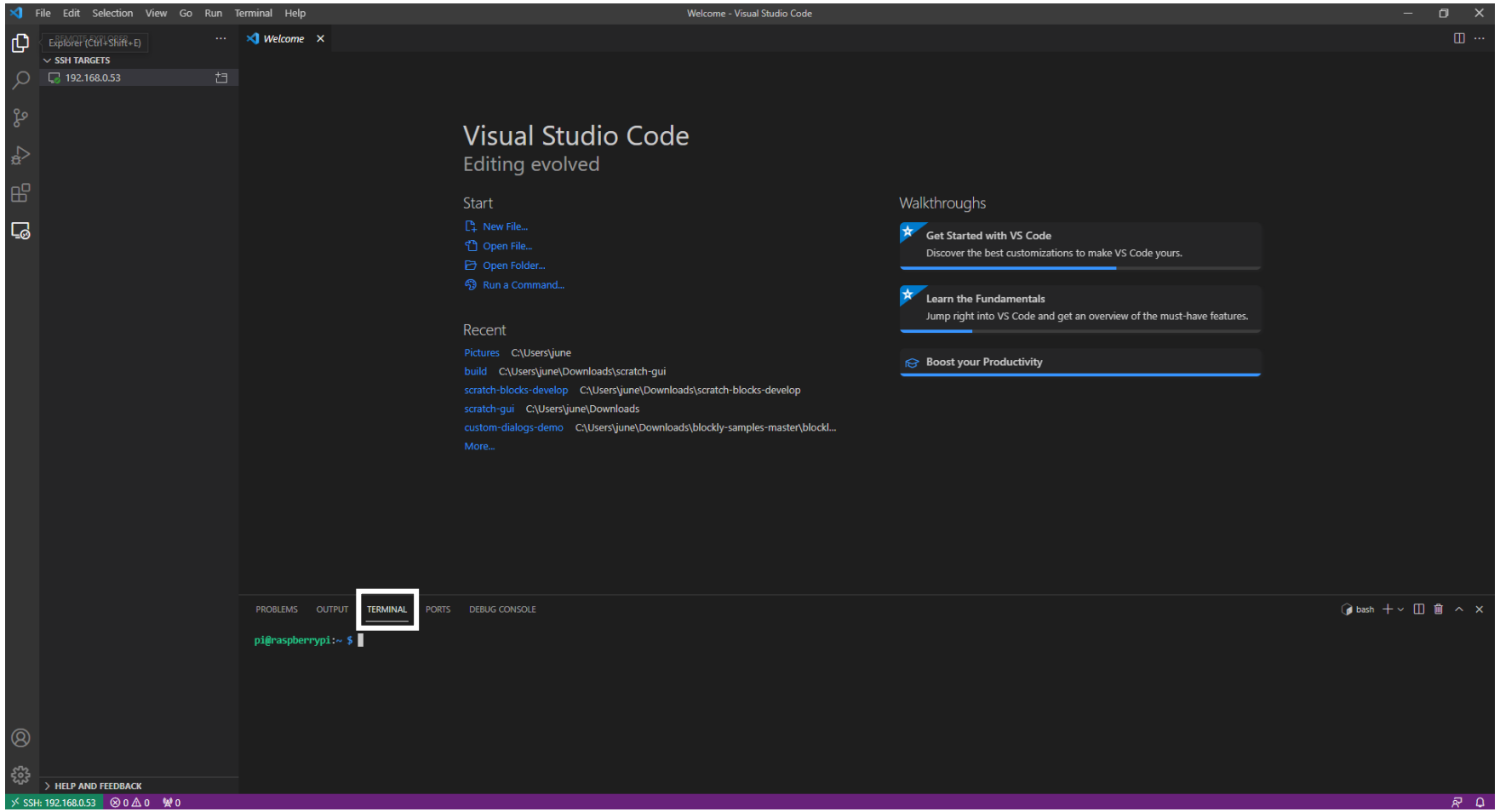
비밀번호 작성: 초기 비밀번호: raspberry



Install

VS Code SSH Extension

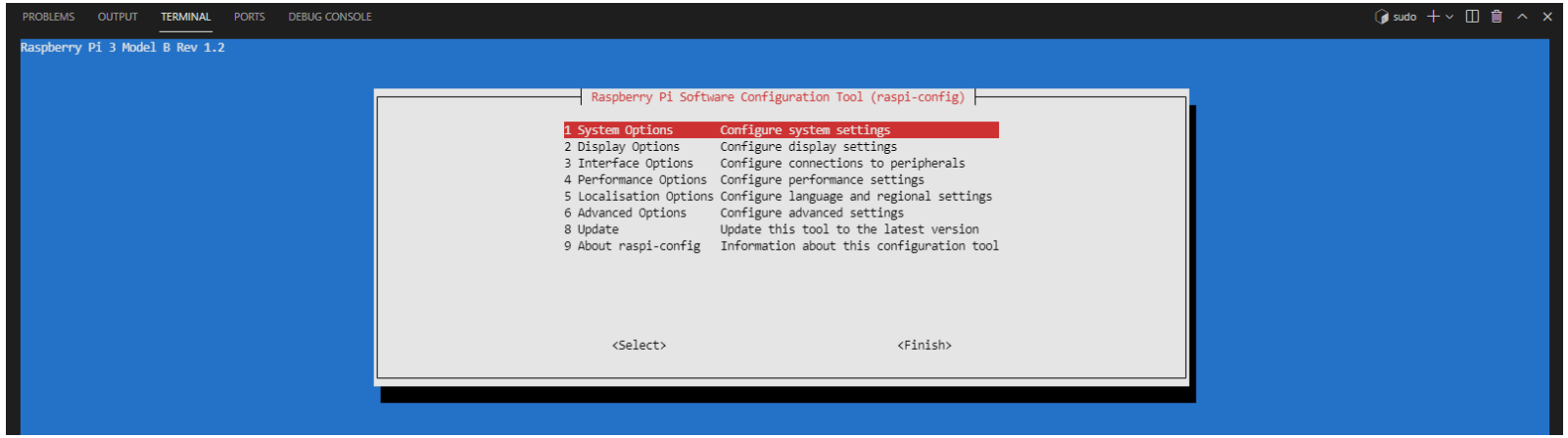
터미널을 열면 SSH로 제어 가능



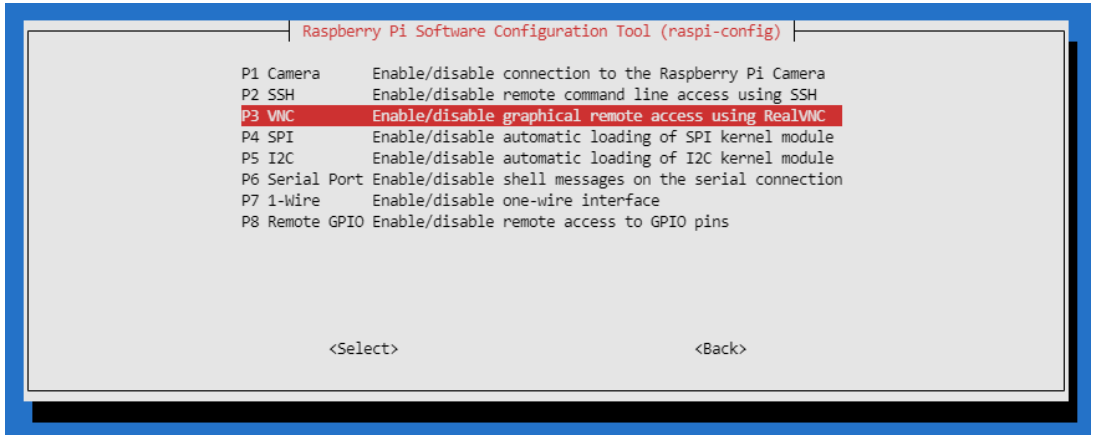
Install

Raspberry Pi VNC ON

터미널에 `sudo raspi-config` 작성시에 해당 설정이 나옴



해당 부분에서 `3 Interface Options` 클릭시에 아래와 같이 나오며

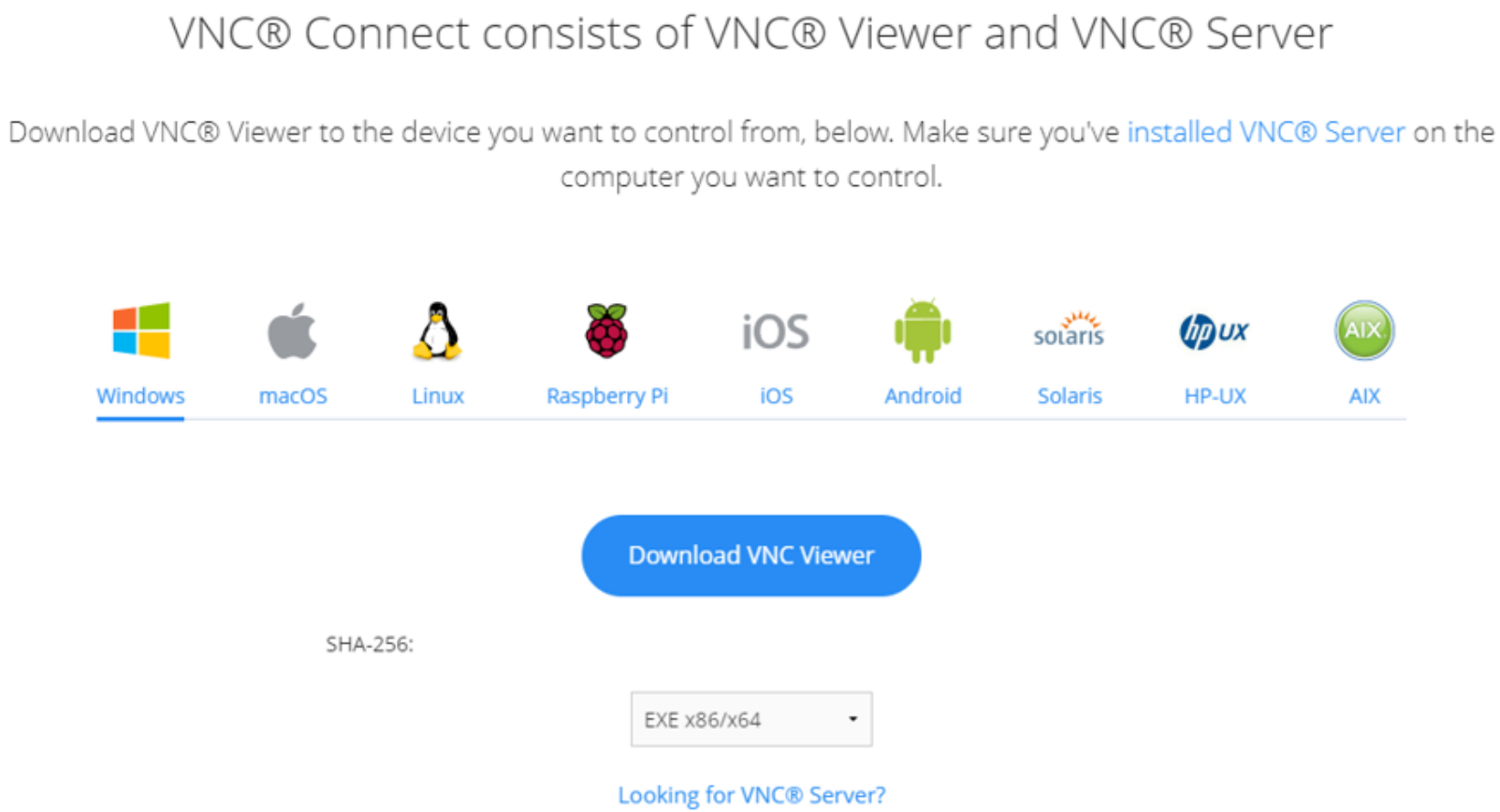


VNC를 눌러서 `YES`를 눌러서 사용 가능하게 설정 → 이와 같이 하면 VNC Viewer에서 접근 가능

Install

VNC Viewer

https://www.realvnc.com/en/connect/download/viewer/

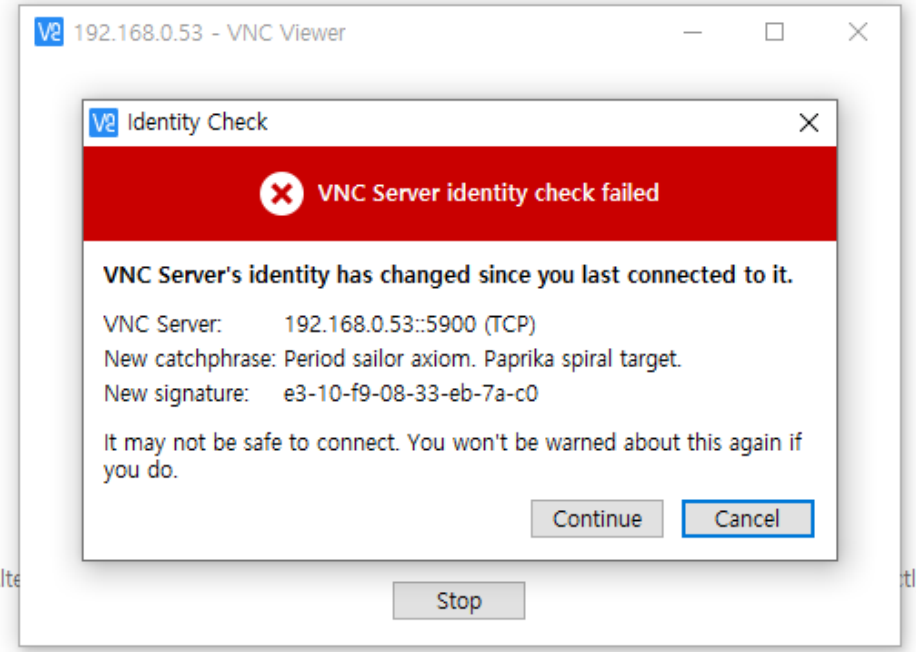


VS Code SSH로 연결하여 진행할 시에 VNC Viewer 필요없이 텍스트 상으로만 제어하실 수도 있습니다.

Install

VNC Viewer

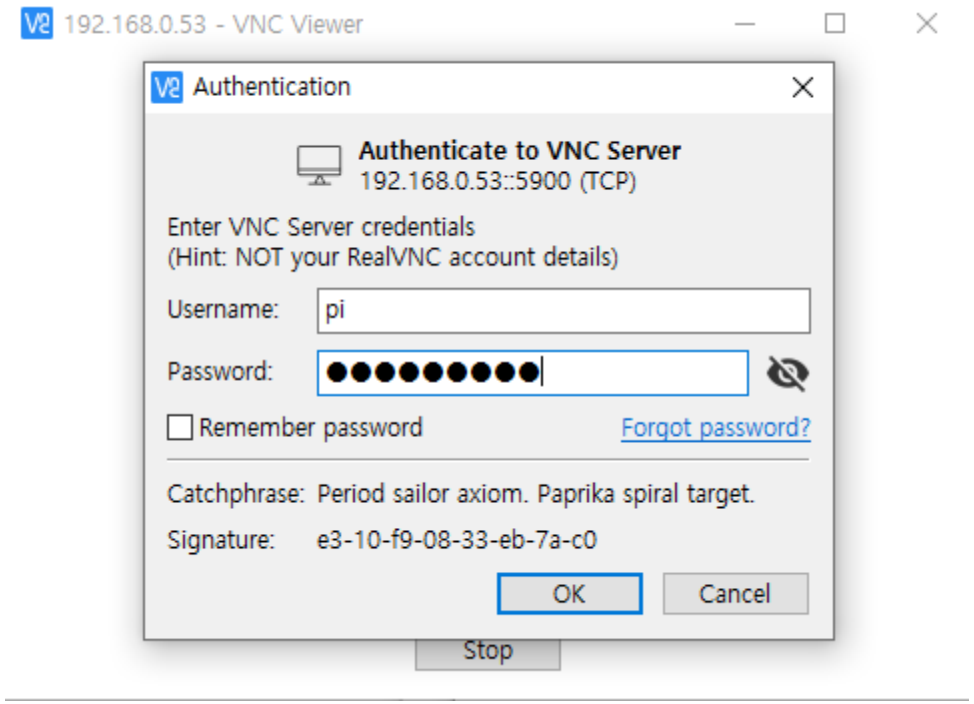
라즈베리파이 ip 주소 작성을 하고 엔터를 누르면



해당 창이 나오고, continue를 눌러서 진행하면 됩니다.

Install

VNC Viewer



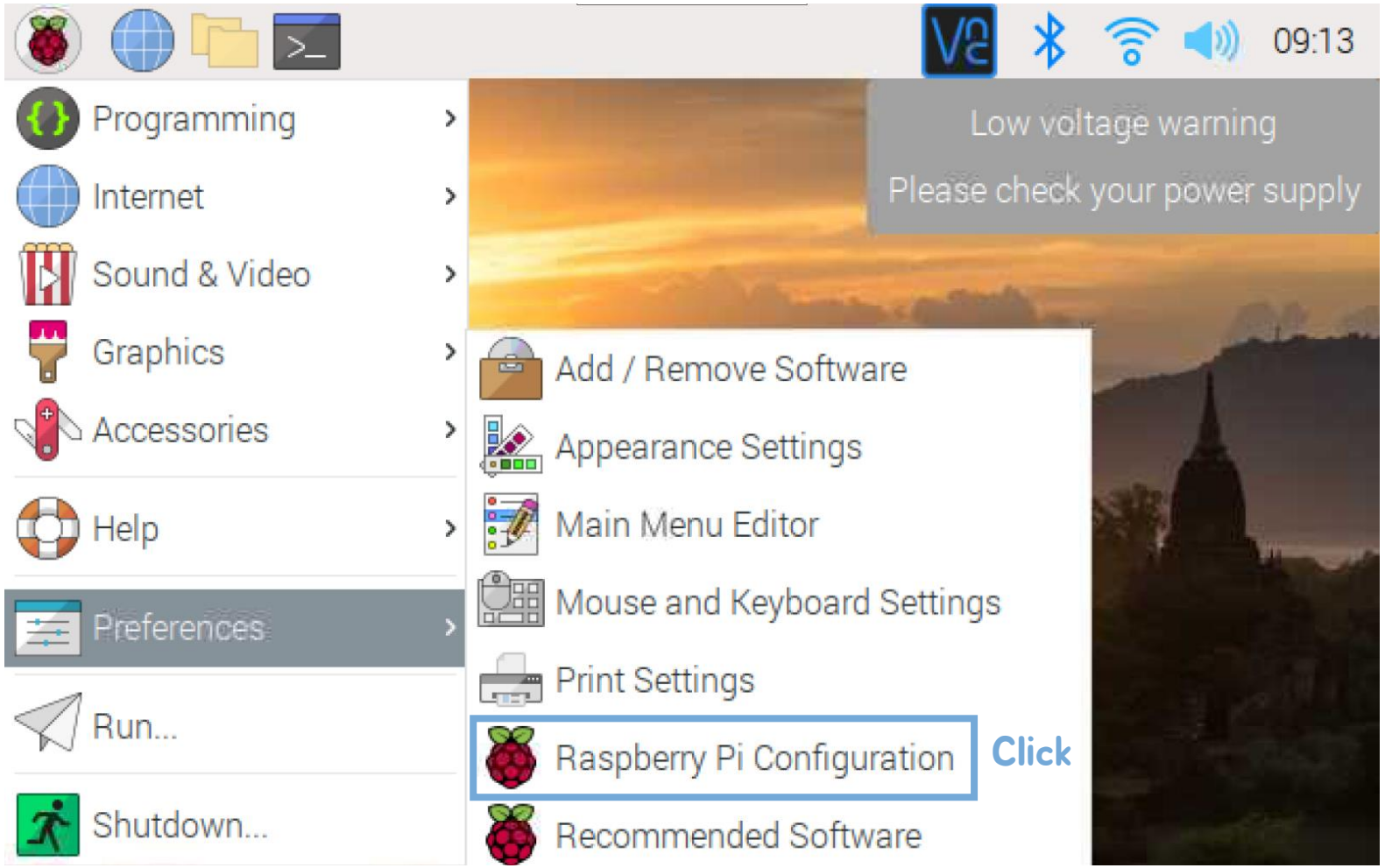
Username과 password를 작성하고 `OK`을 클릭하면 됩니다.

기본 Username: pi

기본 Password: raspberry

Install

VNC Viewer

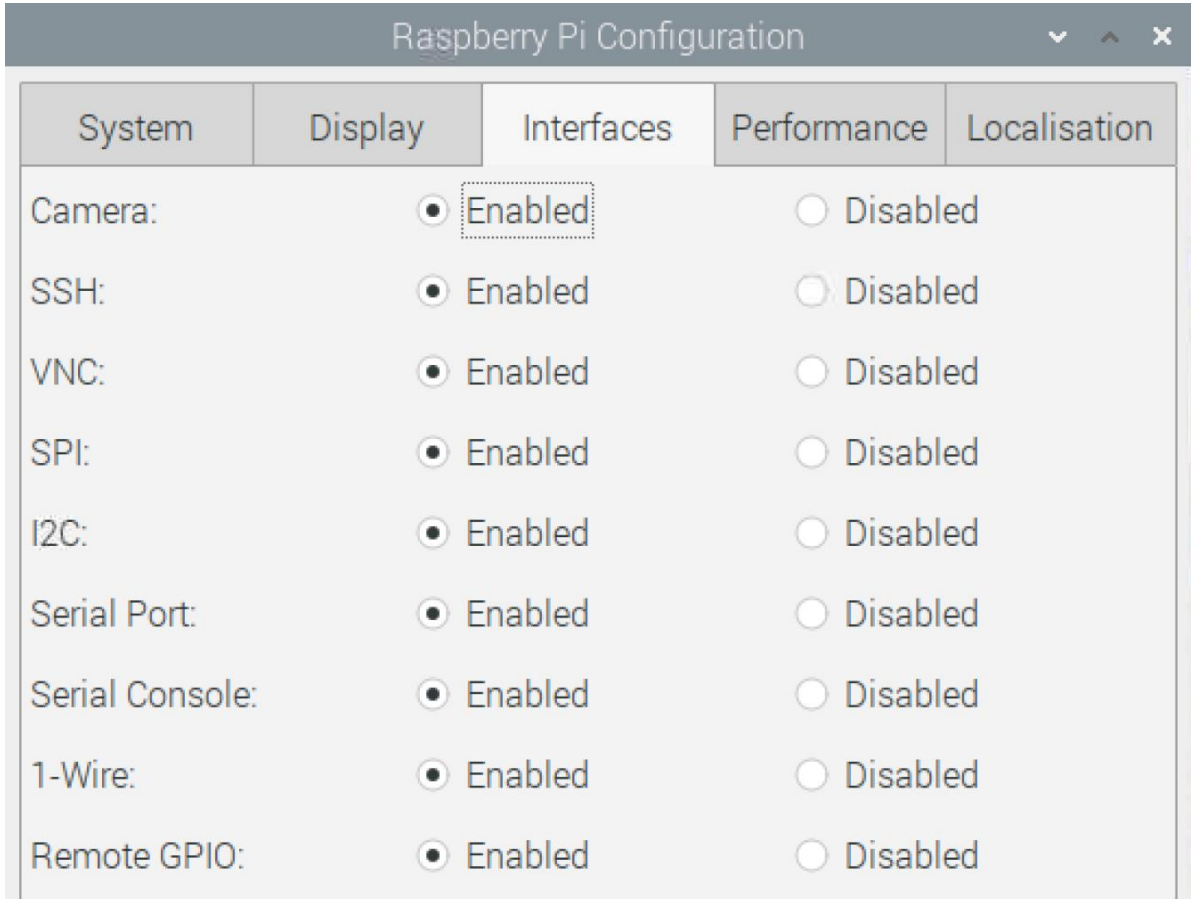


Install

VNC Viewer

Interfaces 부분에 Enabled로 다 두어서 사용 가능하게 하겠습니다.

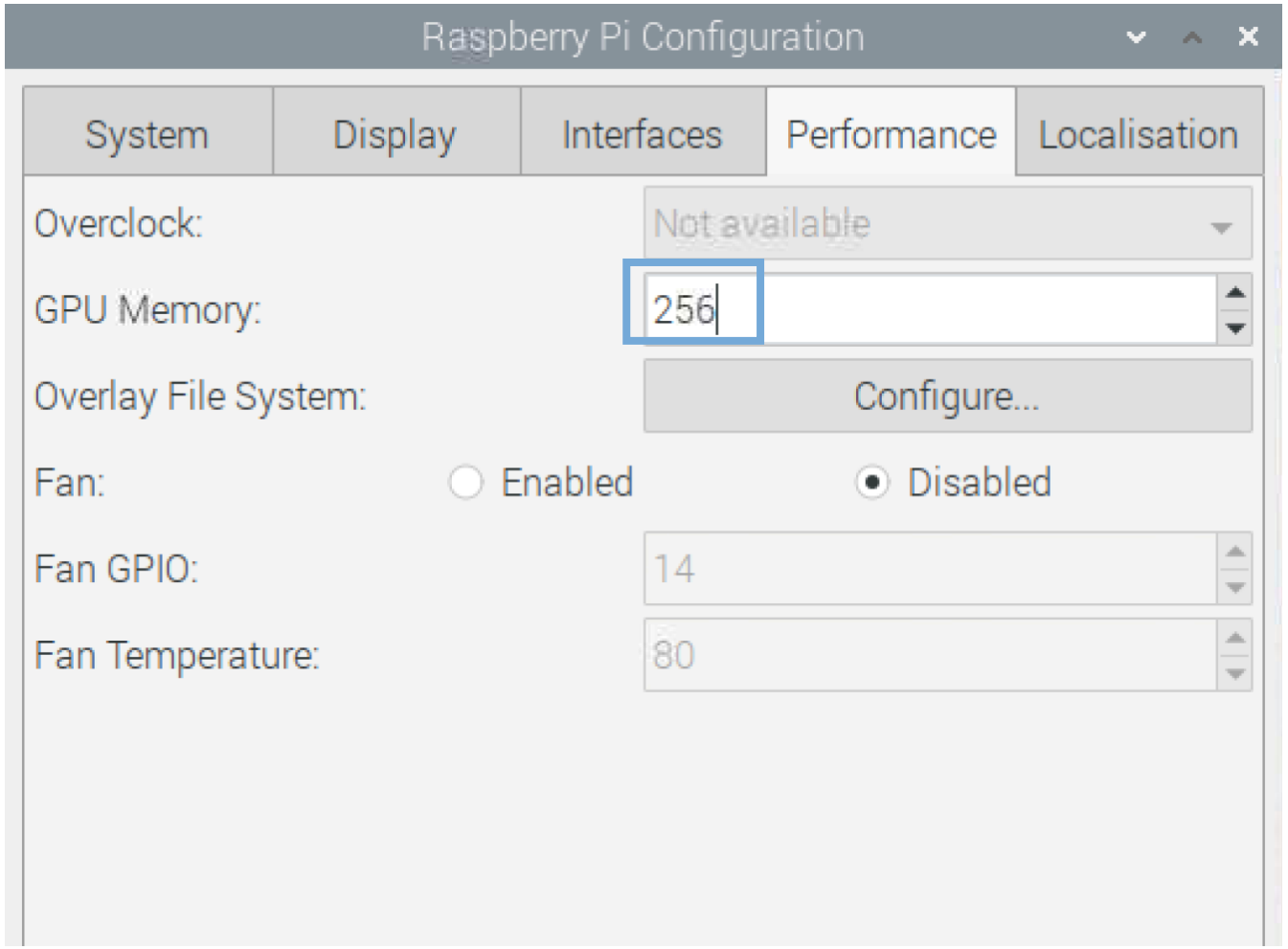
본래, 사용 목적에 따라 설정해야 하지만, 편의를 위해 다 가능하게 설정



Install

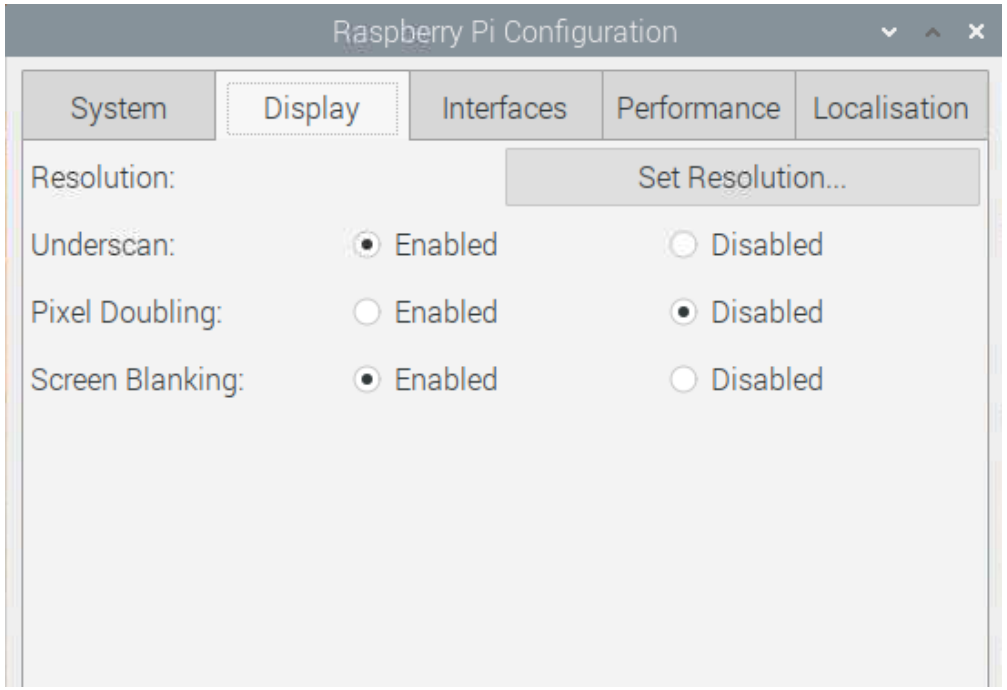
VNC Viewer

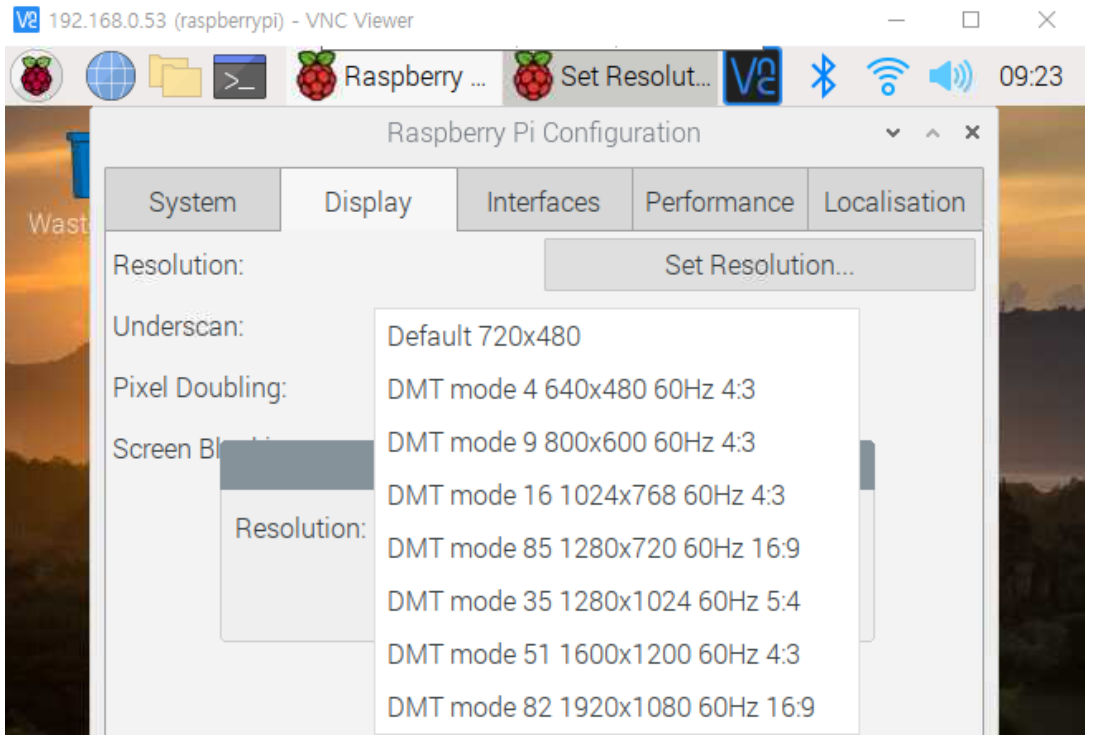
GPU Memory를 256 정도로 올려주도록 하겠습니다.



Install

VNC Viewer





Install

VNC Viewer

참고) 라즈베리파이 VNC Port: 5900

Linux

Base Command

해당 프로젝트를 진행하면서 사용할 기본적인 리눅스 명령어만 알아보겠습니다.

sudo: Super User Do

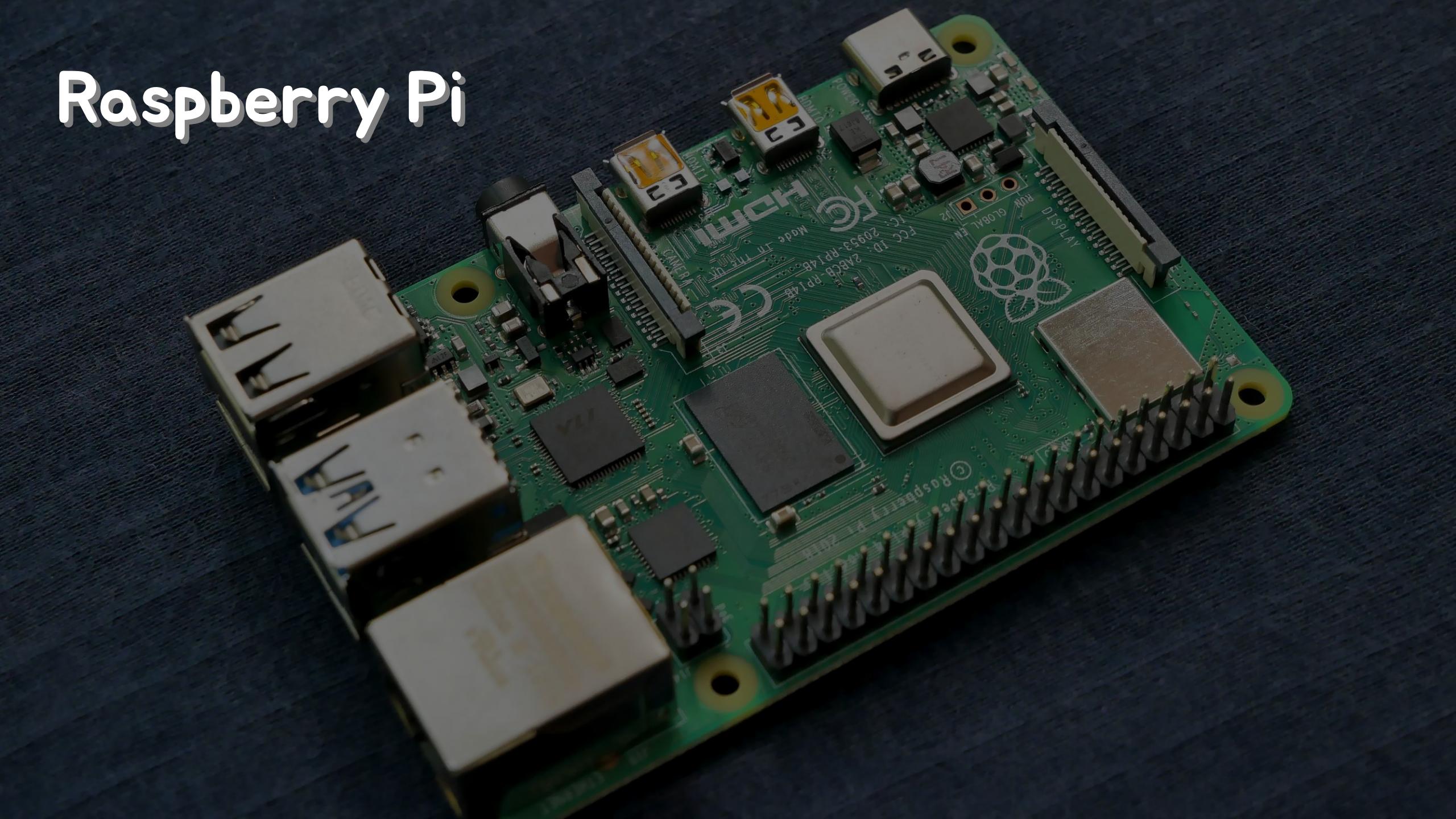
ls : List

cd : Change Directory

mkdir: Make Directory

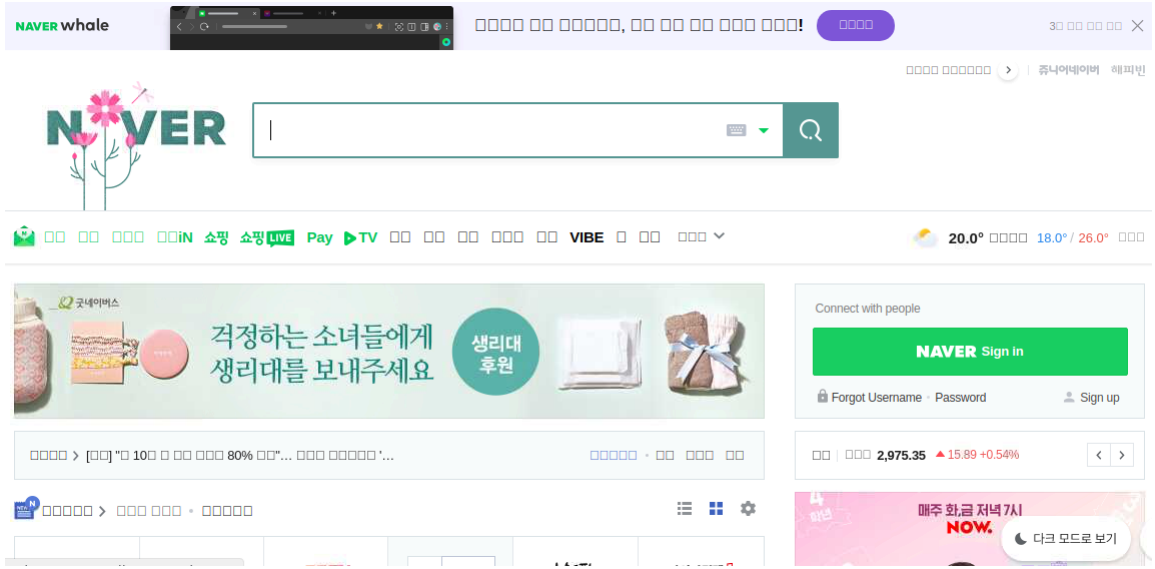
touch: Create File

Raspberry Pi



Raspberry Pi 3B +

한글화



초기에는 한글이 깨져서 나오는 것을 확인 가능

먼저, 터미널을 열고

```
sudo apt-get update
sudo apt-get upgrade
```

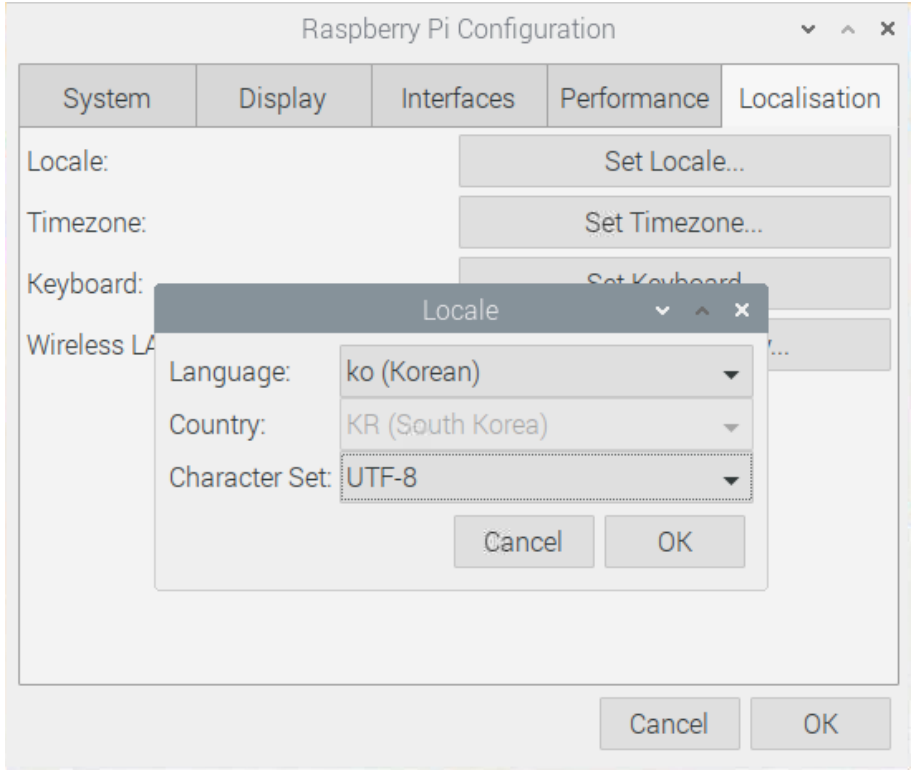
Apt 패키지를 업데이트 해줍니다.

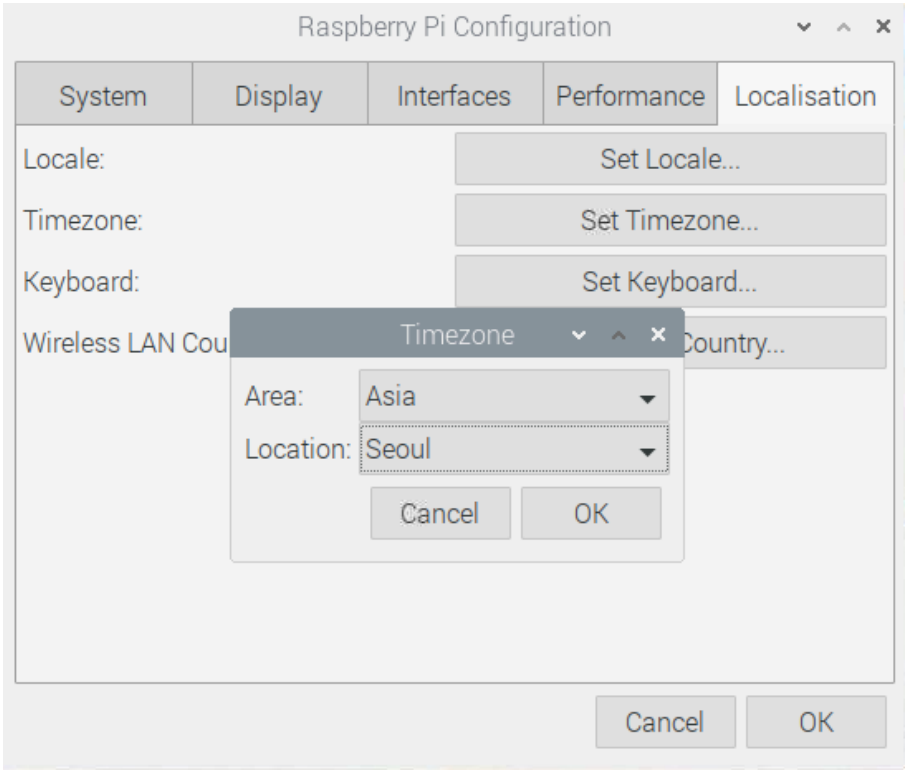
Advanced Packaging Tool(apt)은 리눅스에서 쓰이는 패키지 관리 명령어 도구입니다.

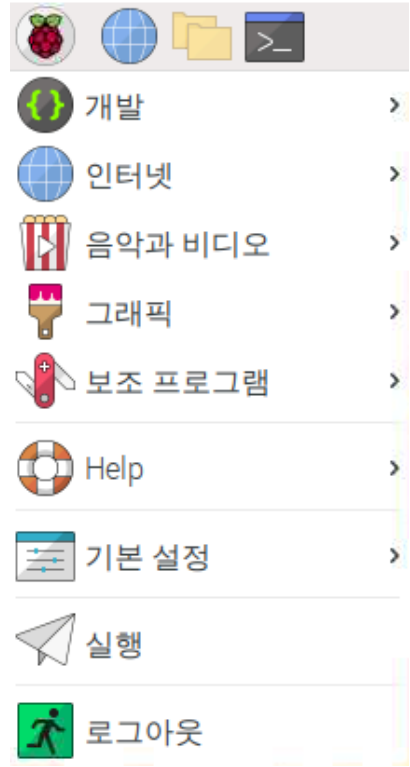
```
sudo apt install fonts-unfonts-core -y
```

Raspberry Pi 3B +

한글화







이와 같이 한글로 나오는 것을 확인할 수 있습니다.

Raspberry Pi 3B +

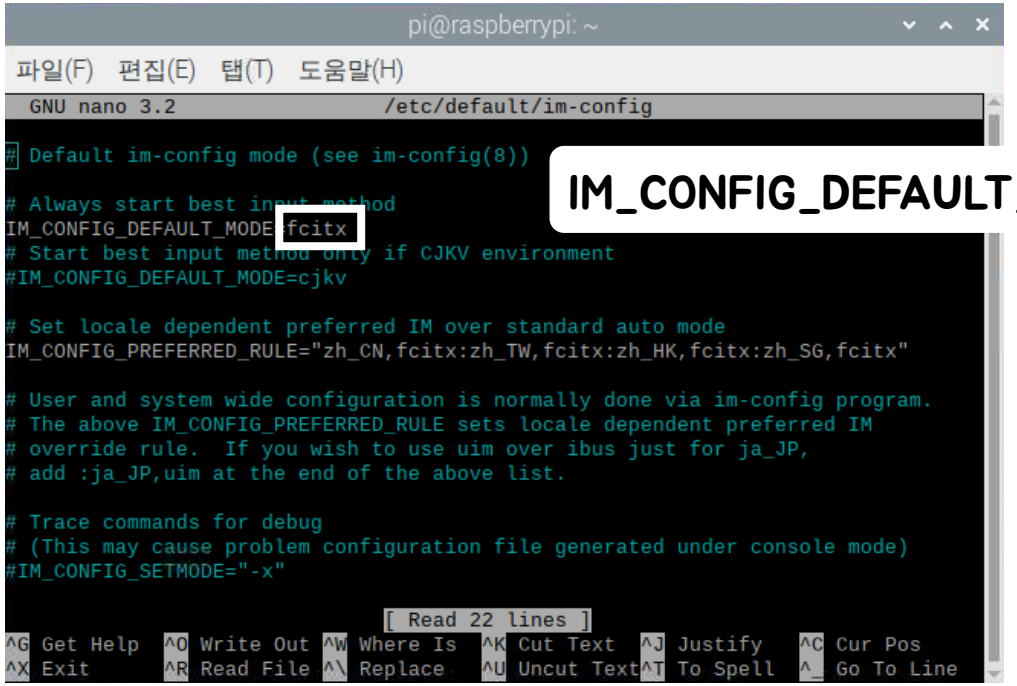
한글화

```
sudo apt remove ibus ibus-hangul
sudo apt install fcitx fcitx-hangul -y
sudo nano /etc/default/im-config
```

기존에 패키지를 삭제하고

필요한 패키지 설치

환경 설정을 위해 파일 OPEN



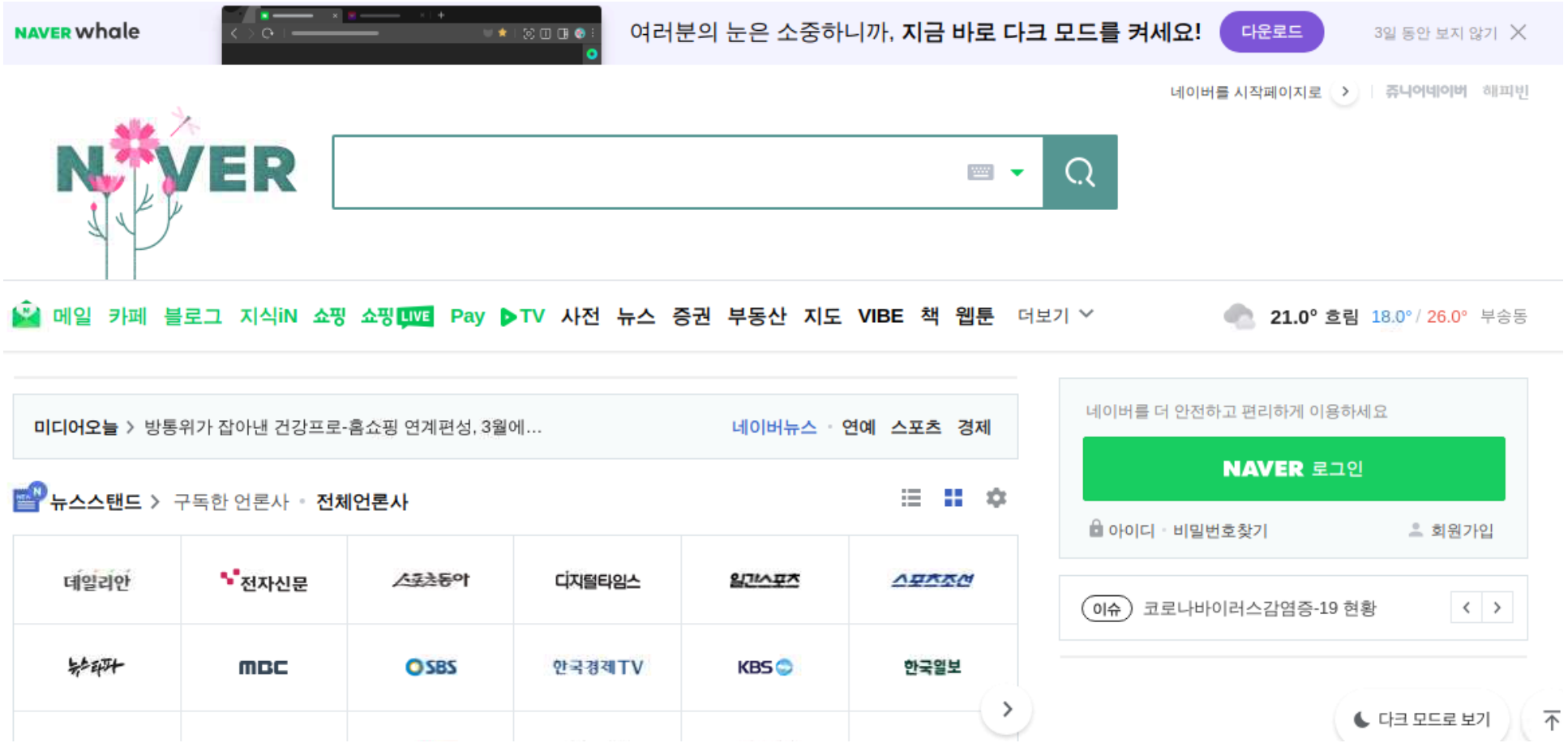
IM_CONFIG_DEFAULT_MODE=fcitx

해당 부분 수정 후, `ctrl+x` → `Y` 엔터

터미널에 sudo reboot로 재실행

Raspberry Pi 3B +

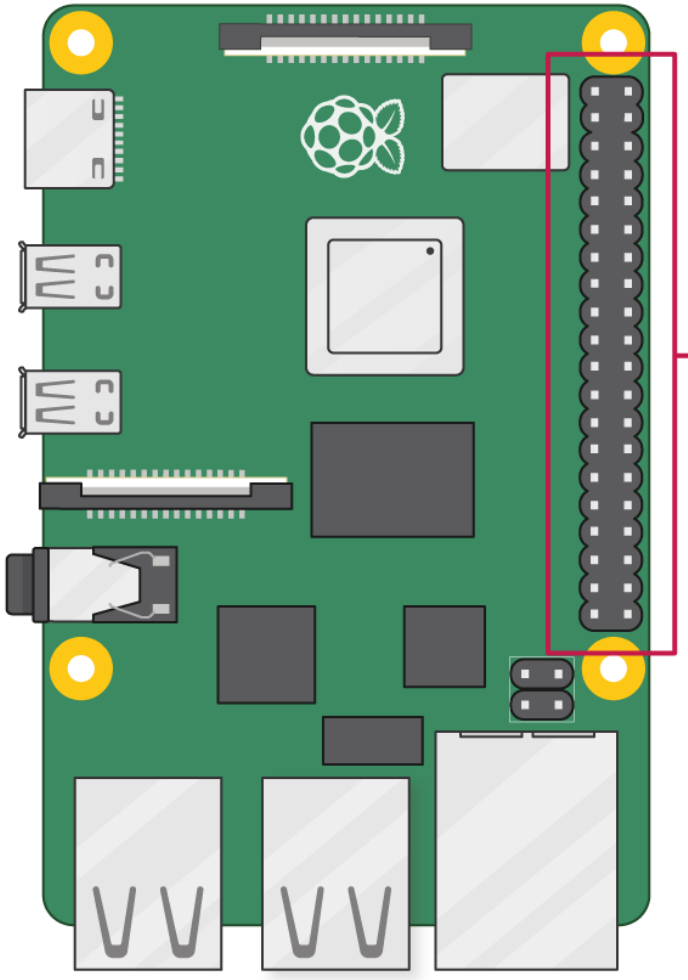
한글화

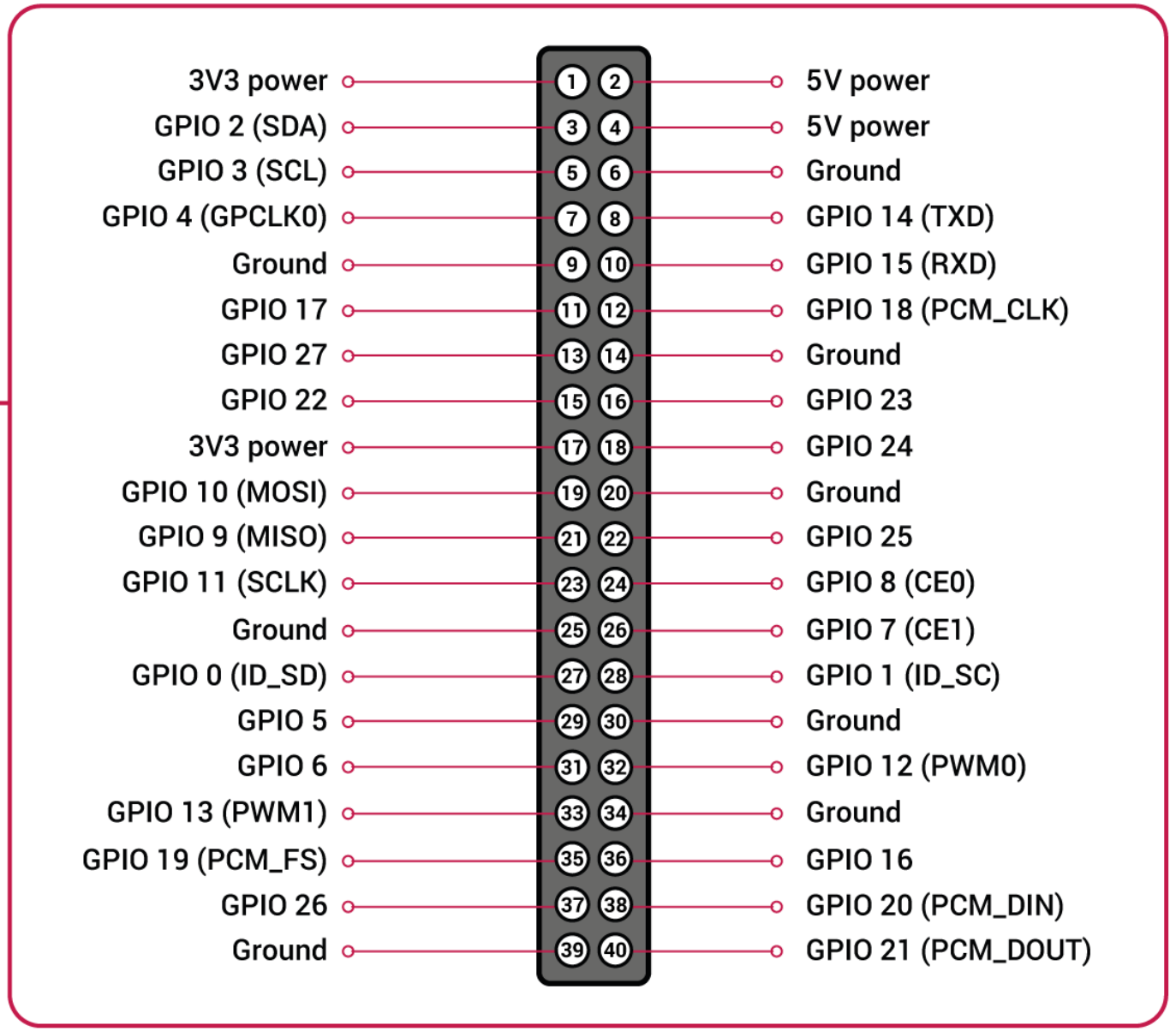


`ctrl + space`를 누르면 한/영키 변환이 가능

Raspberry Pi 3B +

GPIO(BCM) 핀맵





출처 : https://wikidocs.net/3172

GPIO 제어 with Python

LED ON / OFF

```python
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

PIN_LED = 26

GPIO.setup(PIN_LED, GPIO.OUT)

GPIO.output(PIN_LED, True)
time.sleep(5)
GPIO.output(PIN_LED, False)

GPIO.cleanup()
```

LED를 켜고 5초 후에 끄는 CODE

Import로 외부 라이브러리 사용 설정

- Rpi.GPIO 라즈베리 파이 GPIO제어 라이브러리 as → GPIO 별칭 설정
- time: 시간 관련 라이브러리

GPIO.setMode → BOARD와 BCM 설정

- BOARD: 1~40번
- BCM: GPIO 번호

LED 핀번호 GPIO 26 설정

GPIO.setup → 해당 핀 번호, 출력/입력 설정

GPIO.output → 해당 핀, 전기 + / - 설정

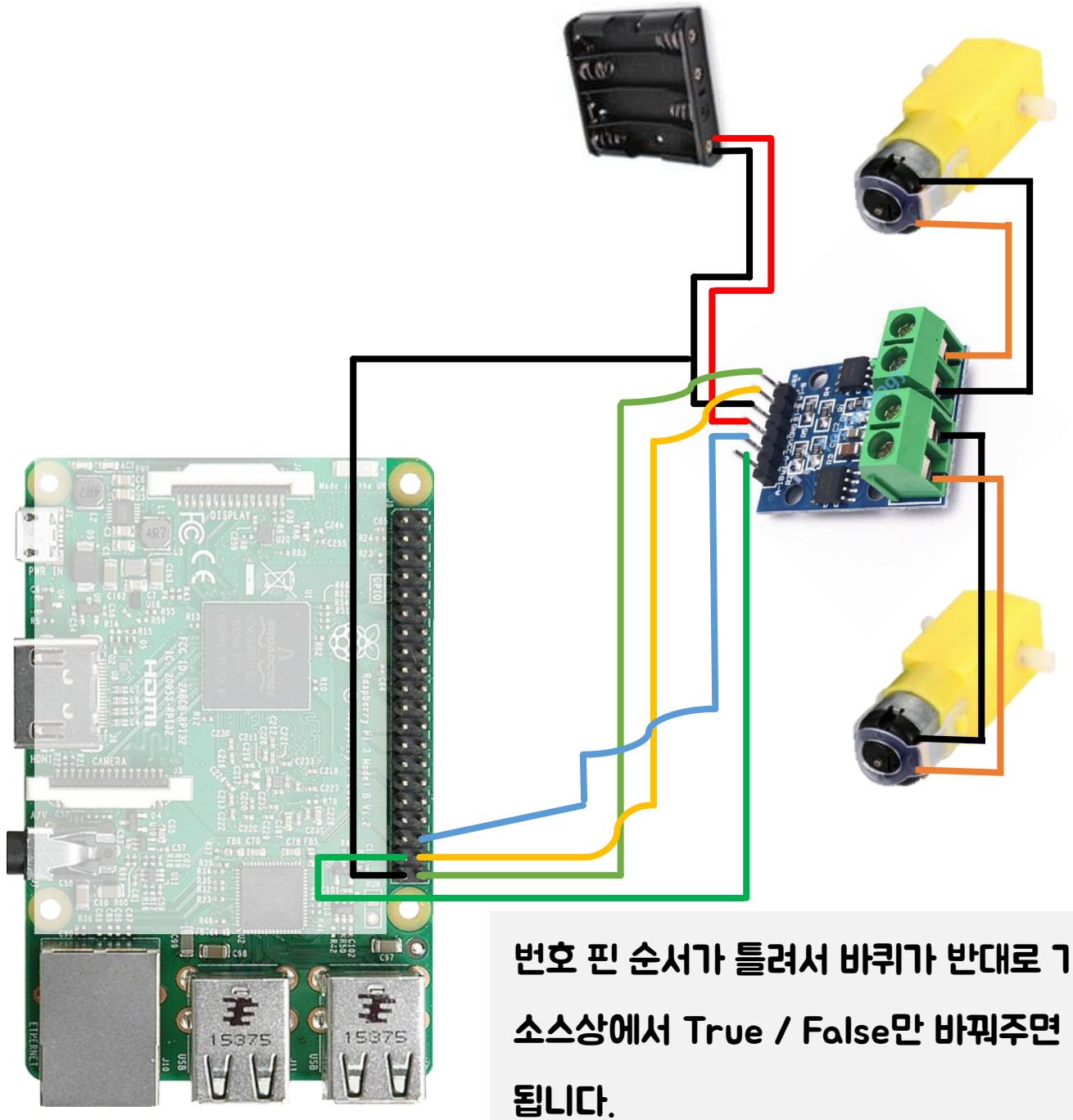
time.sleep(숫자) : 숫자 `초`만큼 멈추기

GPIO.cleanup() → GPIO 설정 핀 clean

GPIO 제어 with Python

DC Motor

```python
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

PIN_LEFT_MOTOR_FW = 20
PIN_LEFT_MOTOR_BW = 21
PIN_RIGHT_MOTOR_FW = 16
PIN_RIGHT_MOTOR_BW = 26

GPIO.setup(PIN_LEFT_MOTOR_FW, GPIO.OUT)
GPIO.setup(PIN_LEFT_MOTOR_BW, GPIO.OUT)
GPIO.setup(PIN_RIGHT_MOTOR_FW, GPIO.OUT)
GPIO.setup(PIN_RIGHT_MOTOR_BW, GPIO.OUT)

GPIO.output(PIN_LEFT_MOTOR_FW, True)
GPIO.output(PIN_LEFT_MOTOR_BW, False)
time.sleep(3)
GPIO.output(PIN_LEFT_MOTOR_FW, False)
GPIO.output(PIN_LEFT_MOTOR_BW, True)
time.sleep(3)
GPIO.output(PIN_LEFT_MOTOR_FW, False)
GPIO.output(PIN_LEFT_MOTOR_BW, False)
time.sleep(1)
GPIO.output(PIN_RIGHT_MOTOR_FW, True)
GPIO.output(PIN_RIGHT_MOTOR_BW, False)
time.sleep(3)
GPIO.output(PIN_RIGHT_MOTOR_FW, False)
GPIO.output(PIN_RIGHT_MOTOR_BW, True)
time.sleep(3)
GPIO.output(PIN_RIGHT_MOTOR_FW, False)
GPIO.output(PIN_RIGHT_MOTOR_BW, False)
time.sleep(1)

GPIO.cleanup()
```



번호 핀 순서가 틀려서 바퀴가 반대로 가면 소스상에서 True / False만 바꿔주면 됩니다.

GPIO 제어 with Python

DC Motor

```python
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

PIN_LEFT_MOTOR_FW = 5
PIN_LEFT_MOTOR_BW = 6
PIN_RIGHT_MOTOR_FW = 22
PIN_RIGHT_MOTOR_BW = 23

GPIO.setup(PIN_LEFT_MOTOR_FW, GPIO.OUT)
GPIO.setup(PIN_LEFT_MOTOR_BW, GPIO.OUT)
GPIO.setup(PIN_RIGHT_MOTOR_FW, GPIO.OUT)
GPIO.setup(PIN_RIGHT_MOTOR_BW, GPIO.OUT)

def motor_ctrl(is_left_front, is_right_front, is_left_stop, is_right_stop):
    if is_left_stop:
        GPIO.output(PIN_LEFT_MOTOR_FW, False)
        GPIO.output(PIN_LEFT_MOTOR_BW, False)
    else:
        GPIO.output(PIN_LEFT_MOTOR_FW, is_left_front)
        GPIO.output(PIN_LEFT_MOTOR_BW, not is_left_front)

    if is_right_stop:
        GPIO.output(PIN_RIGHT_MOTOR_FW, False)
        GPIO.output(PIN_RIGHT_MOTOR_BW, False)
    else:
        GPIO.output(PIN_RIGHT_MOTOR_FW, is_right_front)
        GPIO.output(PIN_RIGHT_MOTOR_BW, not is_right_front)

motor_ctrl(True, False, False, True)
time.sleep(3)
motor_ctrl(False, False, False, True)
time.sleep(3)
motor_ctrl(False, False, True, True)
time.sleep(1)
motor_ctrl(False, True, True, False)
time.sleep(3)
motor_ctrl(False, False, True, False)
time.sleep(3)
motor_ctrl(False, False, True, True)
time.sleep(1)

GPIO.cleanup()
```

Python에 함수 문법을 이용하여

제어를 편하게 사용하기 위해 변경된 CODE

Python에서 함수는 `def` 로 표현

함수 참조 링크

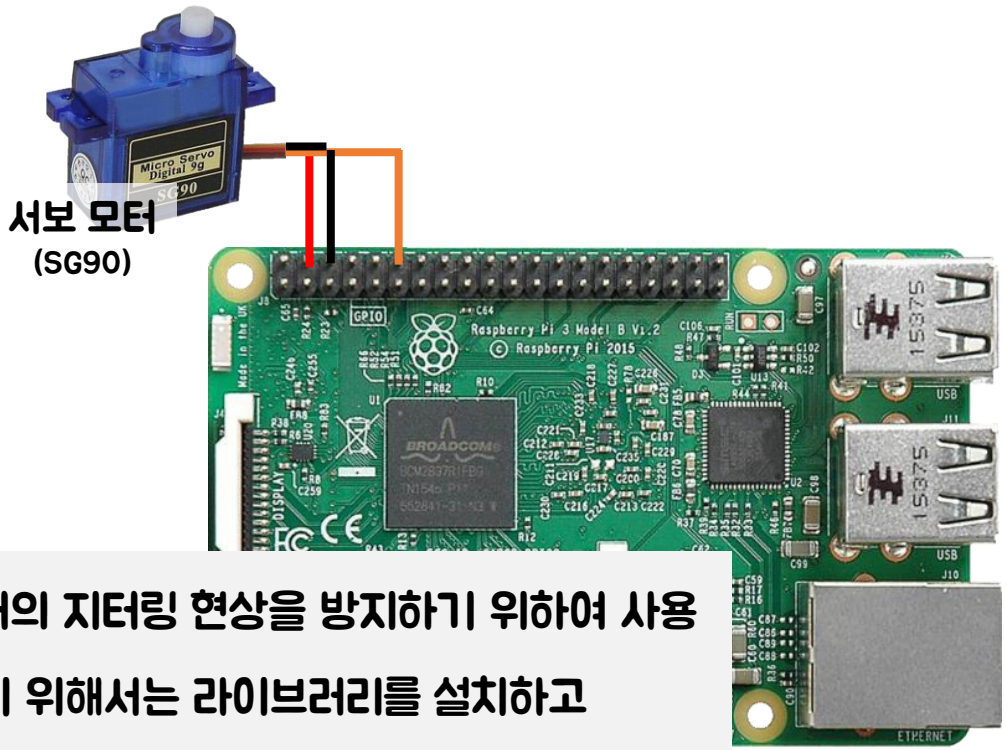
- https://github.com/EduProgramming/Python/blob/develop/15def.ipynb
- https://wikidocs.net/24

GPIO 제어 with Python

Servo

```
sudo apt-get install pigpio python-pigpio python3-pigpio

sudo pigpiod
```

```python
import pigpio
import time

PIN_SERVO = 18

servo = pigpio.pi()
servo.set_servo_pulsewidth(PIN_SERVO, 1500)
time.sleep(1)
servo.set_servo_pulsewidth(PIN_SERVO, 2400)
time.sleep(1)
servo.set_servo_pulsewidth(PIN_SERVO, 600)
time.sleep(1)
servo.set_servo_pulsewidth(PIN_SERVO, 1500)
time.sleep(1)
```



서보 모터
(SG90)

pigpiod는 서보모터의 지터링 현상을 방지하기 위하여 사용

pigpiod를 사용하기 위해서는 라이브러리를 설치하고

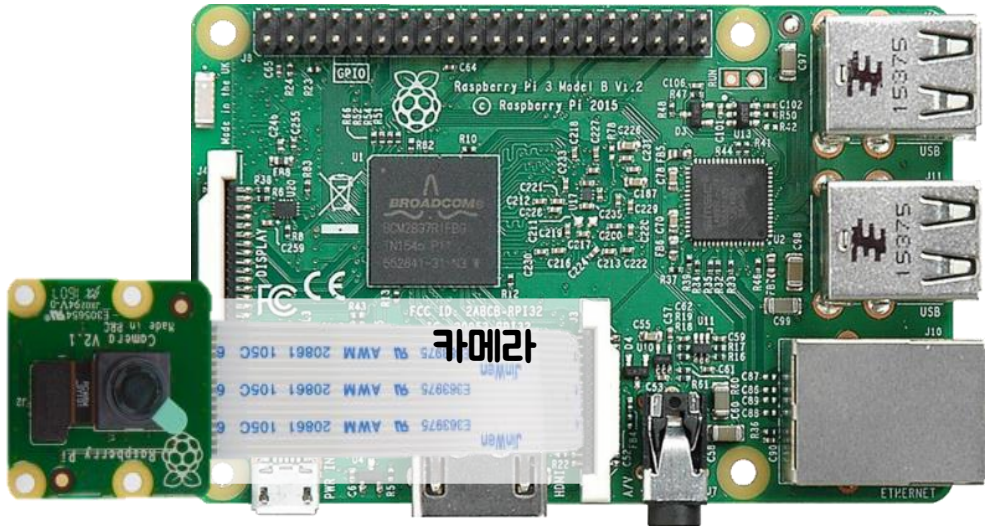
pigpiod 서버를 실행하면 됩니다.

Pulse 범위: 500 ~ 2500

참고 링크: http://abyz.me.uk/rpi/pigpio/python.html

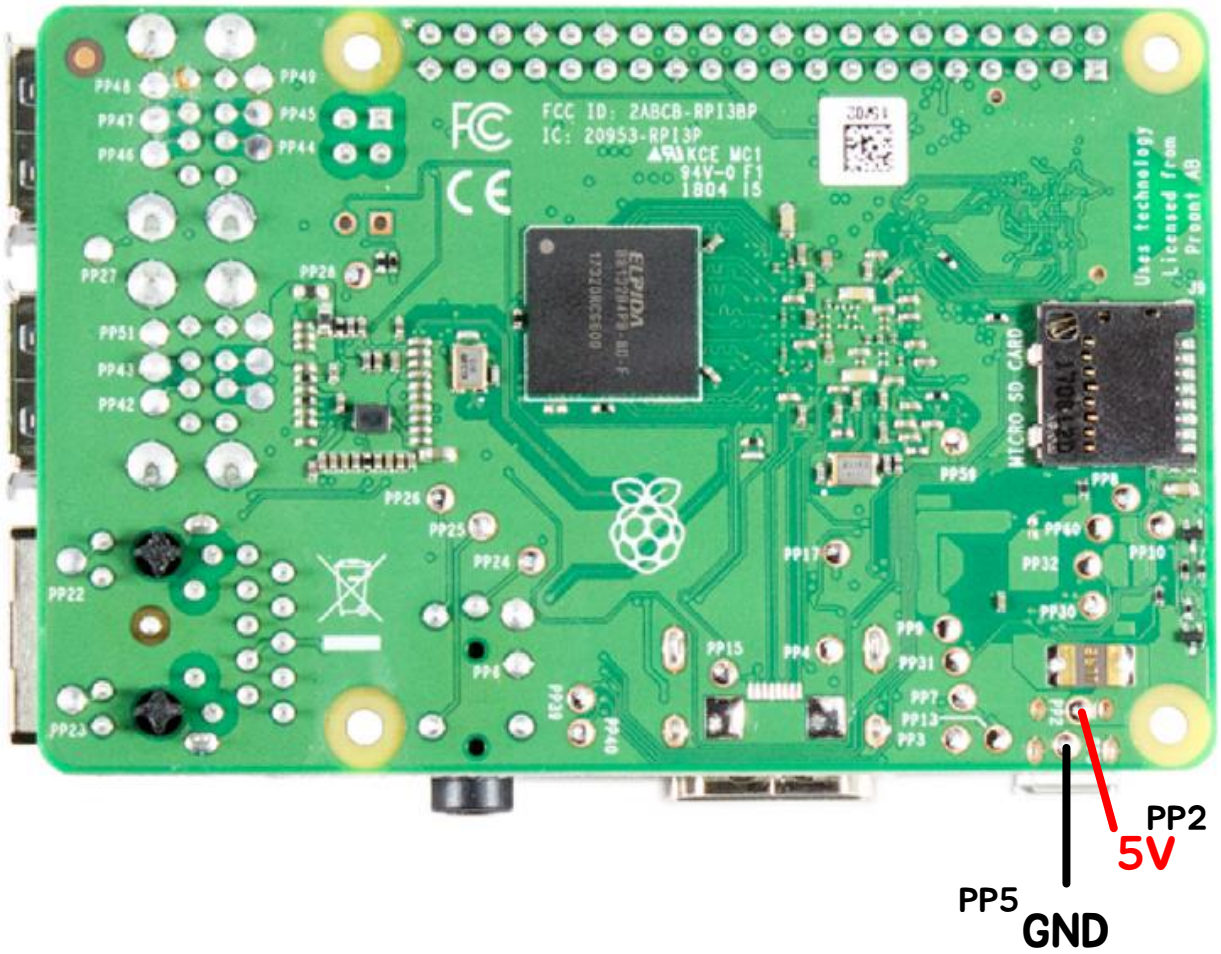
Raspberry Pi 3B +

Camera Module



외부 전력 공급 방법

PP2 / PP5

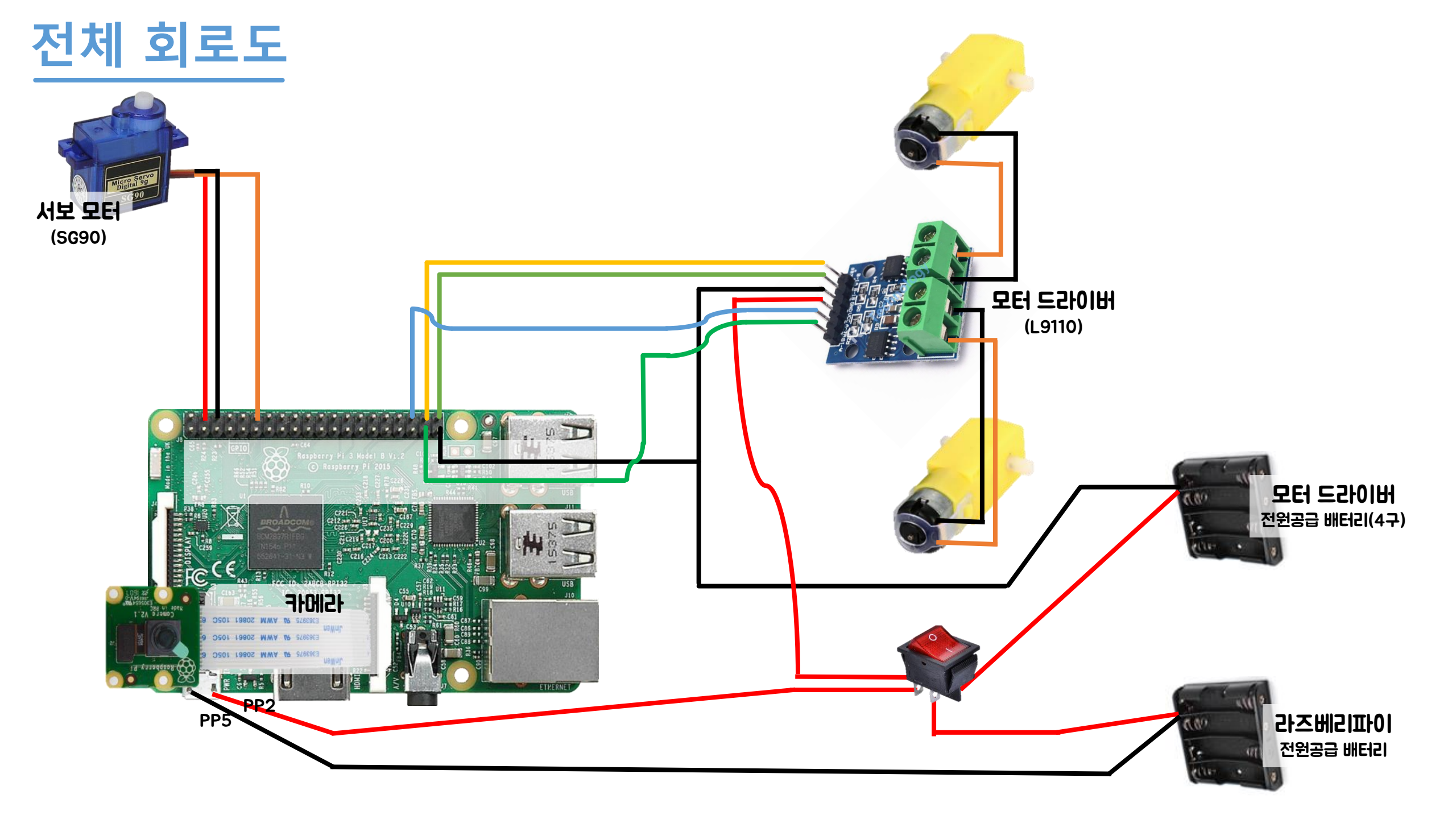


PP2에 5V 전원단

PP5에 GND

연결하여 외부 전원으로 작동 시킬 수 있습니다.

전체 회로도


서보 모터
(SG90)
모터 드라이버
(L9110)
카메라
PP5
PP2
모터 드라이버
전원공급 배터리(4구)
라즈베리파이
전원공급 배터리

CODE

Flask

파이썬 웹 프레임워크 중 하나로써, 웹서버 역할을 합니다.

프레임워크: 뼈대 역할을 하는 것(간단히, 클래스 + 라이브러리 형태)

```python
from flask import Flask

app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello, World!"

if __name__ == '__main__':
    app.run(host='0.0.0.0', threaded=True)
```

라즈베리파이IP:5000 브라우저에 주소를 치면 localhost:5000도 VS Code 진행시 가능했음

Hello, World! 이렇게 나오는 웹 페이지가 나옵니다.

참고링크: https://flask.palletsprojects.com/en/2.0.x/

Flask

flask의 render_template을 이용하여 다른 HTML을 열게 설정

```python
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')

if __name__ == '__main__':
    app.run()
```

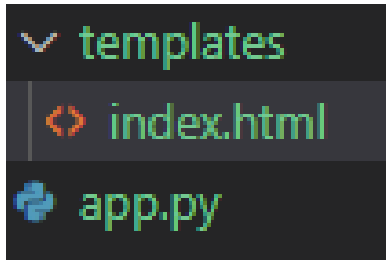
HTML CODE

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible"
content="IE=edge">
    <meta name="viewport"
content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    Hello, RaspberryPi
</body>
</html>
```

Flask

여기서 주의할 점은 HTML 파일의 위치인데 Flask는 기본적으로

`templates`폴더를 기반으로 html파일을 찾습니다.

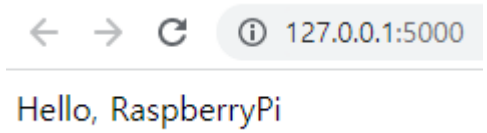


App.py 파일이 있는 곳에 `templates`폴더가 있으며,

해당 폴더 안에는 index.html 파일이 있습니다.

`sudo python app.py` 터미널에 작성시에 서버가 실행됩니다.

127.0.0.1:5000으로 이동하게 되면 아래와 같이 나오게 됩니다.



해당 주소는 서버를 킨 local에서만 사용 가능

이외의 요소에서는 접근이 불가능합니다.