

Python  
chat bot



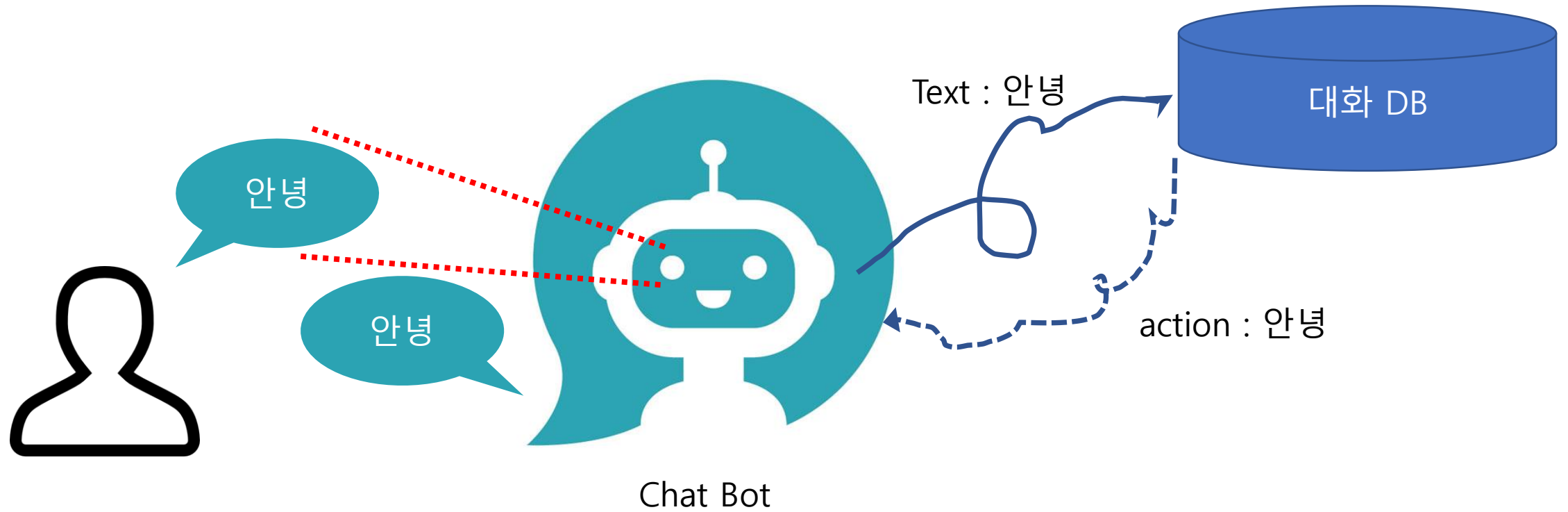
*“Life is too short, You need  
Python!”*

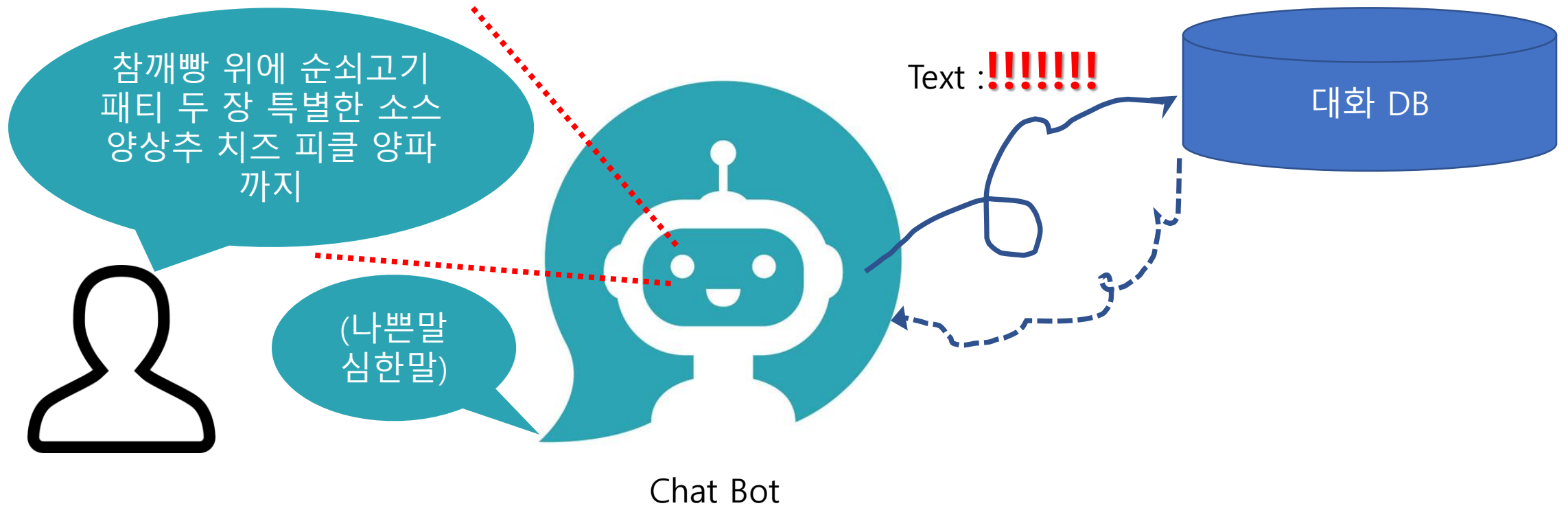
- `for (i=0; i < mylist_length; i++)  
    {do_something(mylist[i]);}`
- `i = 0  
while i < mylist_length:  
    do_something(mylist[i])  
    i += 1`
- `for i in range(mylist_length):  
    do_something(mylist[i])`
- `for element in mylist:  
    do_something(element)`

**PYTHONIC CODE**

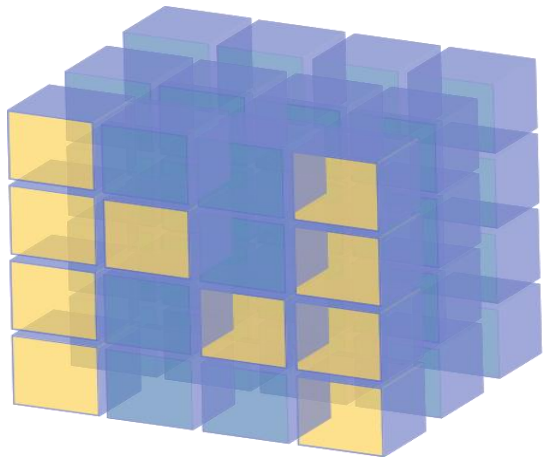
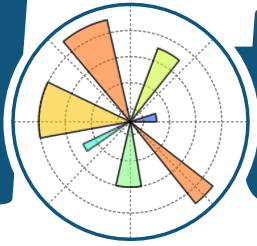
지금까지 뭐 했지?

코딩.... 음... 파이썬.... 리스트.... 딕셔너리.... API..... ??



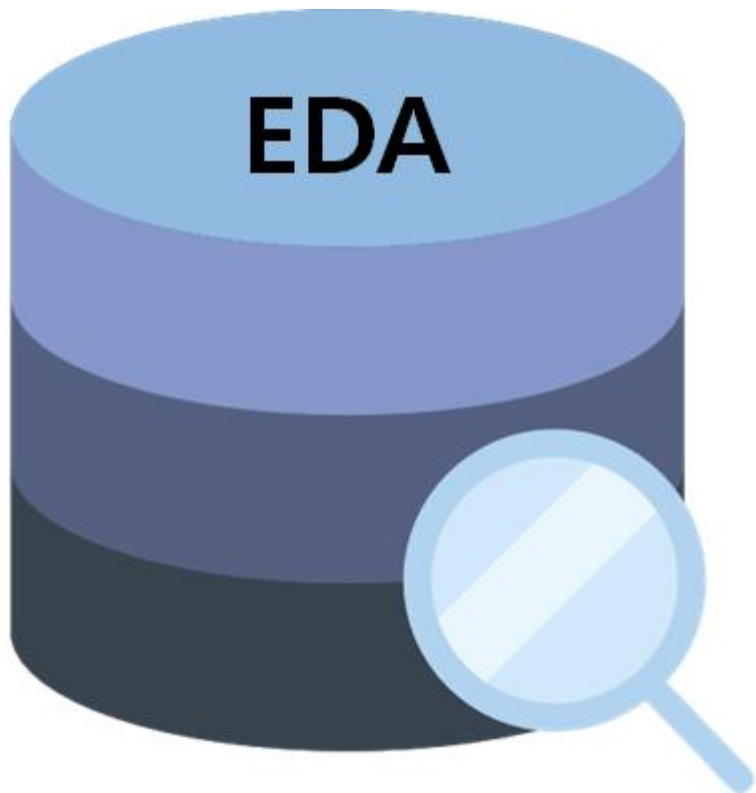


*matplotlib*



NumPy





- EDA(Exploratory Data Analysis)  
탐색적 데이터 분석
- 수집한 데이터를 다양한 각도에서 관찰하고 이해하는 과정
- 우리가 뭘 가지고 있는지
- 지금까지는 잘 정제된 데이터를 기반으로 작업  
(= 다른사람이 만든거 가져와서 사용)
- 우리가 원하는게 정제된 데이터로 있는 경우는 매우 적다  
(정제된 데이터에서 원하는 결과를 선택하는 경우를 제외)
- 예시) 데이터 크롤링 후 원하는 데이터가 잘 들어왔는지, 균일하게 들어왔는지 확인하는 방법



<https://www.apistore.co.kr/main.do> - API 스토어

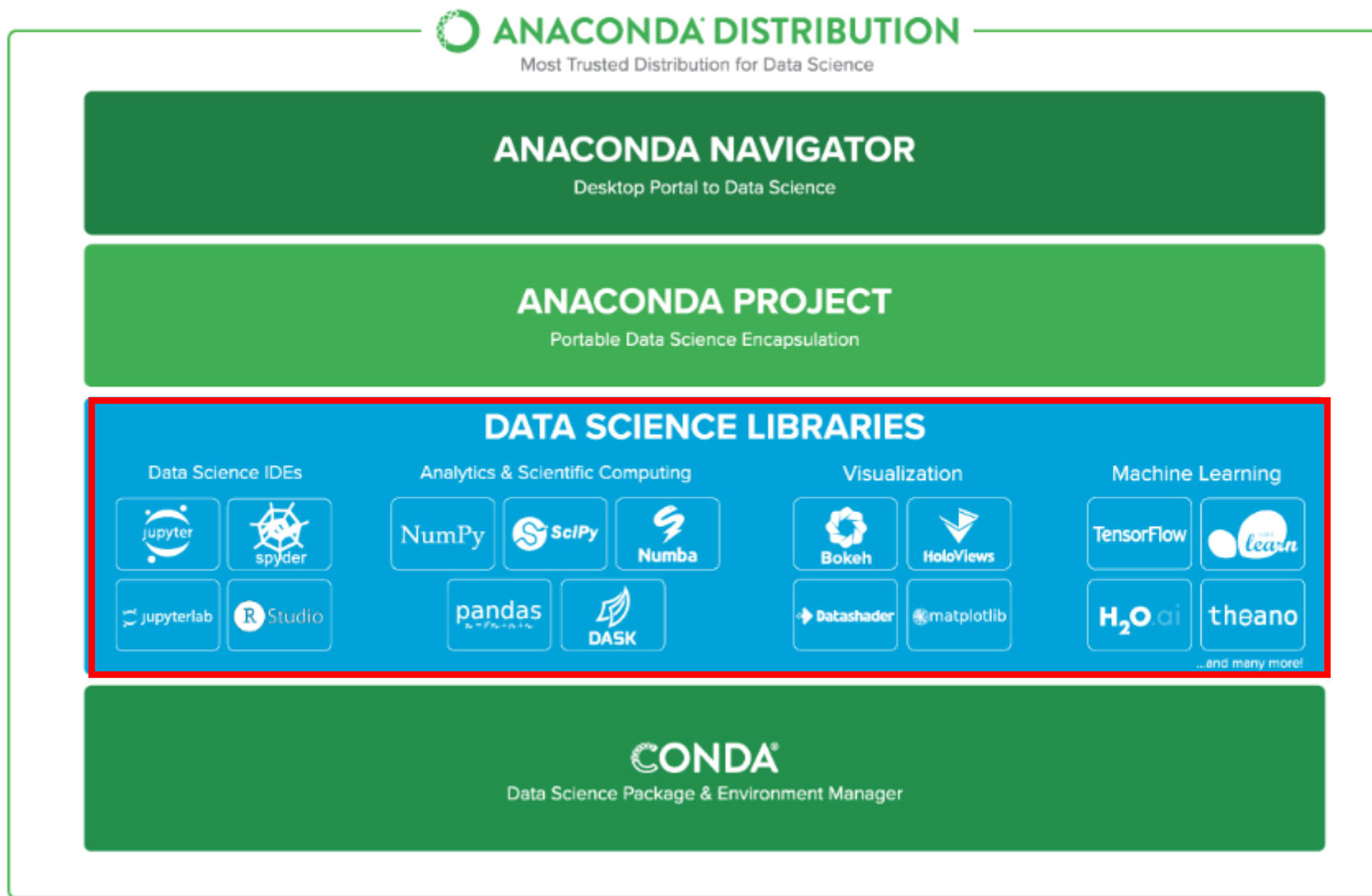
<https://developers.naver.com/main/> - 네이버 개발자 센터

<https://developers.kakao.com> - 카카오 개발자 센터

<http://api.danawa.com/main/index.html> - 다나와 API

<https://developer.auction.co.kr> - 옥션 API

<https://openweathermap.org/api> - Open weather map API





Windows



macOS



Linux

## Anaconda 2019.07 for macOS Installer

### Python 3.7 version

Download

64-Bit Graphical Installer (653 MB)  
64-Bit Command Line Installer (435 MB)

### Python 2.7 version

Download

64-Bit Graphical Installer (634 MB)  
64-Bit Command Line Installer (408 MB)

## Get Started with Anaconda Distribution

### Documentation

Installation and user  
guide for Anaconda  
Distribution 5

[Read More](#)

### Anaconda Blog

News, software  
releases, and  
developer best

practices [Read More](#)

### Community Support

Solutions and  
knowledge from the  
community [Read More](#)

### Anaconda Webinars

Industry trends and  
tutorials from  
Anaconda [Read More](#)

### Anaconda Training

Learn Python for Data  
Science with  
DataCamp

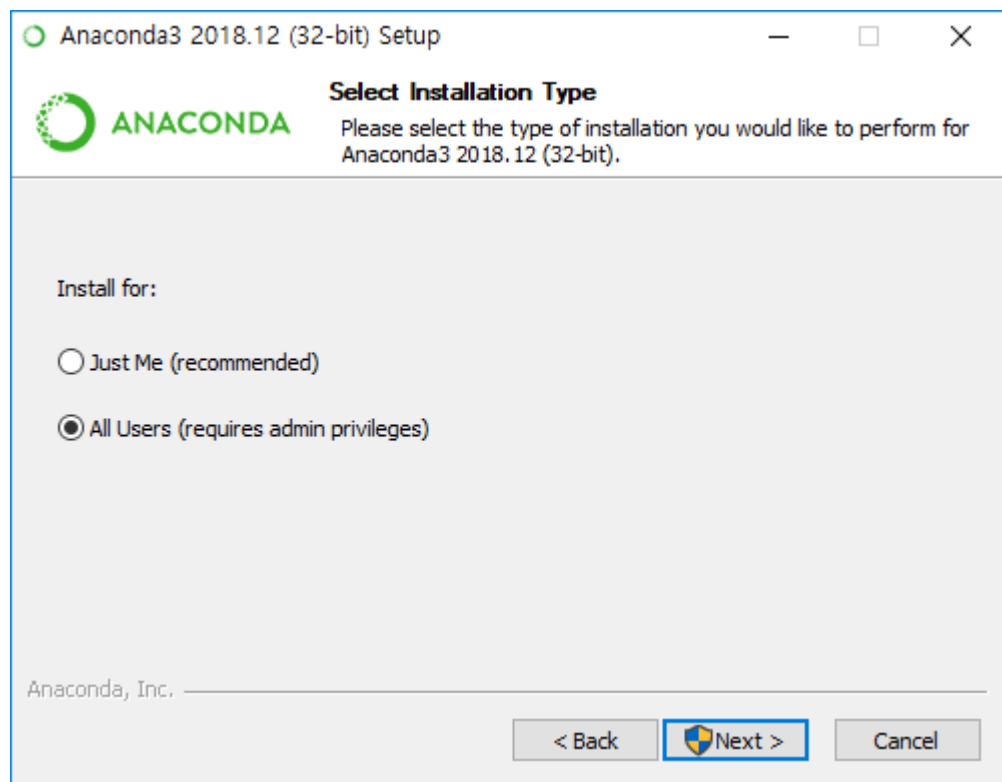
[Start Learning](#)



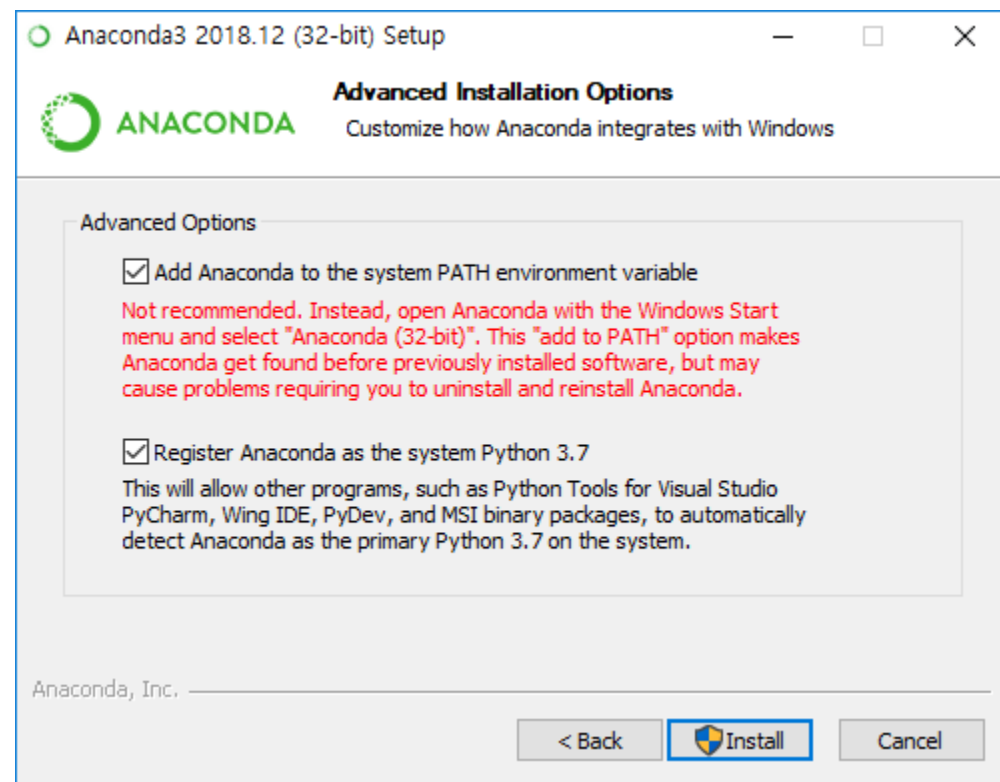
This website uses cookies to ensure you get the best experience on our website. [Privacy Policy](#)

ACCEPT

# Anaconda 설치화면



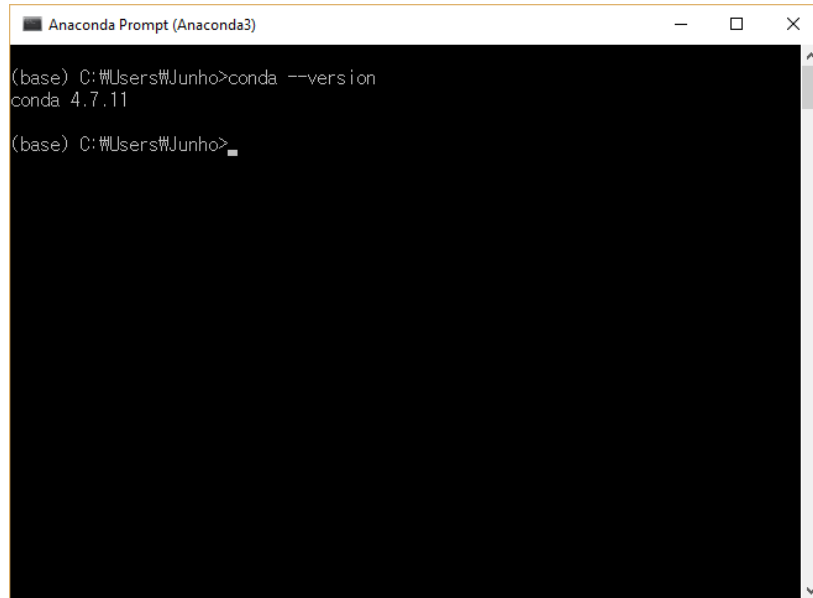
All Users



둘 다 체크

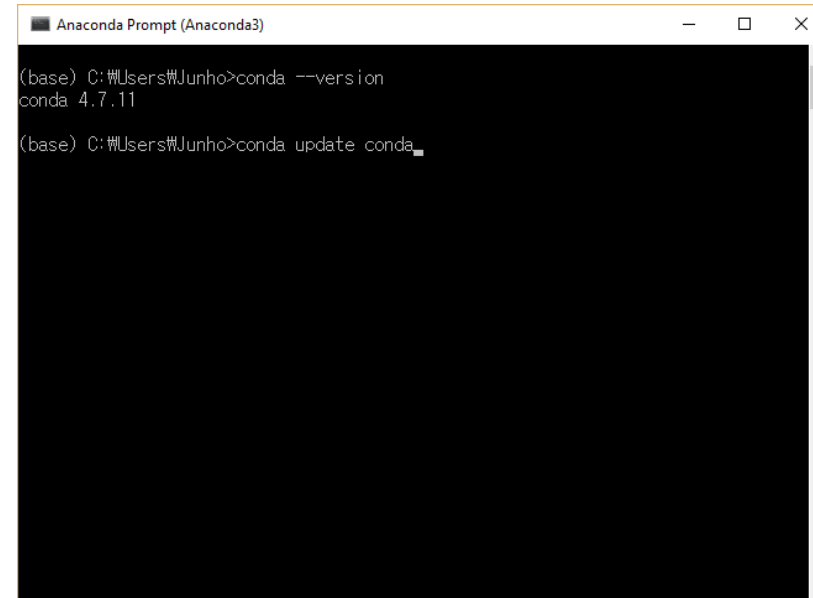
주의!!

# Anaconda prompt



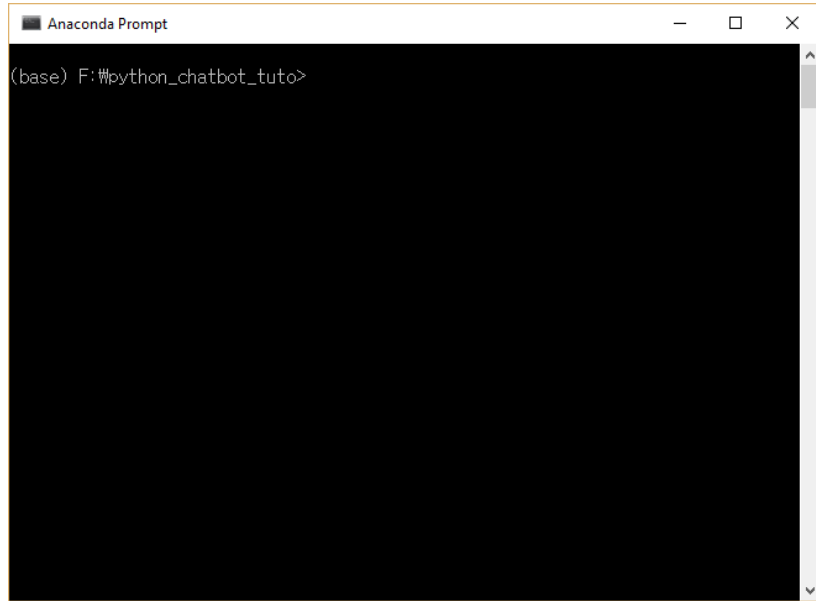
```
Anaconda Prompt (Anaconda3)
(base) C:\Users\Junho>conda --version
conda 4.7.11
(base) C:\Users\Junho>
```

➤ conda --version  
아나콘다 버전확인



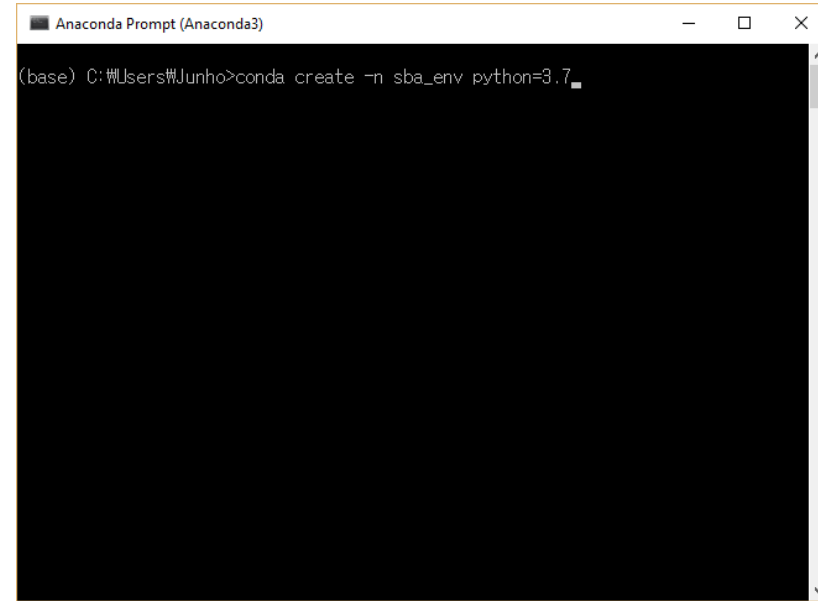
```
Anaconda Prompt (Anaconda3)
(base) C:\Users\Junho>conda --version
conda 4.7.11
(base) C:\Users\Junho>conda update conda
```

➤ conda update conda  
아나콘다 업데이트  
9월 15일 기준 4.7.11



```
Anaconda Prompt
(base) F:\python_chatbot_tuto>
```

<실행화면>



```
Anaconda Prompt (Anaconda3)
(base) C:\Users\Junho>conda create -n sba_env python=3.7
```

➤ conda create -n sba\_env python=3.7

# Anaconda 가상환경

```
Anaconda Prompt (Anaconda3)
Proceed ([y]/n)? y

Downloading and Extracting Packages
certifi-2019.6.16 | 152 KB | ##### | 100%
pip-19.2.2 | 1.7 MB | ##### | 100%
python-3.7.4 | 14.7 MB | ##### | 100%
ca-certificates-2019 | 127 KB | ##### | 100%
openssl-1.1.1d | 5.7 MB | ##### | 100%
sqlite-3.29.0 | 624 KB | ##### | 100%
vs2015_runtime-14.16 | 1.1 MB | ##### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

#
# To activate this environment, use
#
# $ conda activate sba_env
#
# To deactivate an active environment, use
#
# $ conda deactivate

(base) C:\Users\Junho>conda activate sba_env_
```

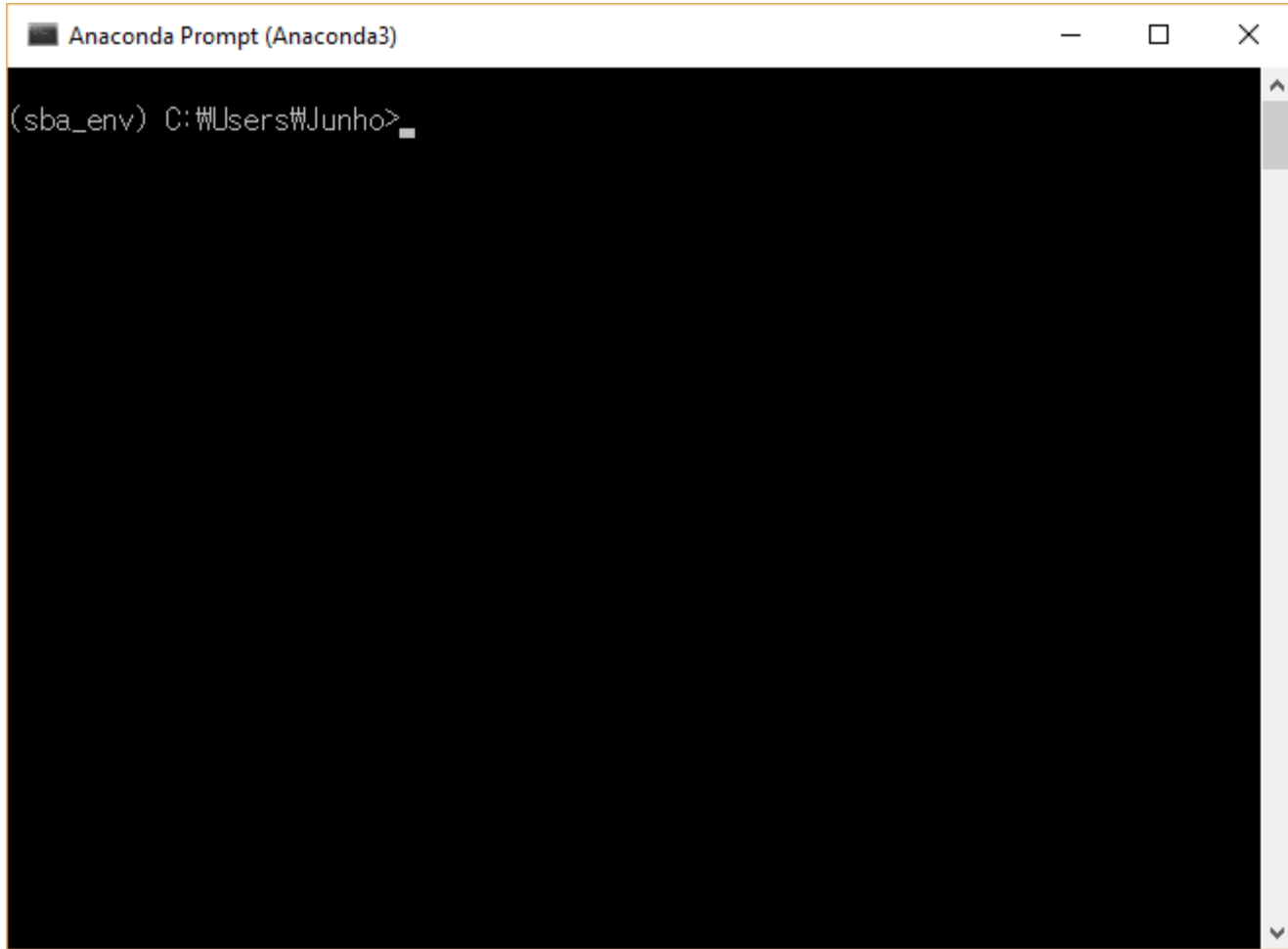
```
Anaconda Prompt (Anaconda3)

Downloading and Extracting Packages
certifi-2019.6.16 | 152 KB | ##### | 100%
pip-19.2.2 | 1.7 MB | ##### | 100%
python-3.7.4 | 14.7 MB | ##### | 100%
ca-certificates-2019 | 127 KB | ##### | 100%
openssl-1.1.1d | 5.7 MB | ##### | 100%
sqlite-3.29.0 | 624 KB | ##### | 100%
vs2015_runtime-14.16 | 1.1 MB | ##### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

#
# To activate this environment, use
#
# $ conda activate sba_env
#
# To deactivate an active environment, use
#
# $ conda deactivate

(base) C:\Users\Junho>conda activate sba_env
(sba_env) C:\Users\Junho>
```

➤ conda activate sba\_env



Anaconda Prompt (Anaconda3)

```
(sba_env) C:\Users\Junho>
```

- conda install flask
- conda install openpyxl
- conda install xlwt
- conda install requests
- pip install tensorflow==2.0.0-rc0
- conda install flask, openpyxl, xlwt, ...



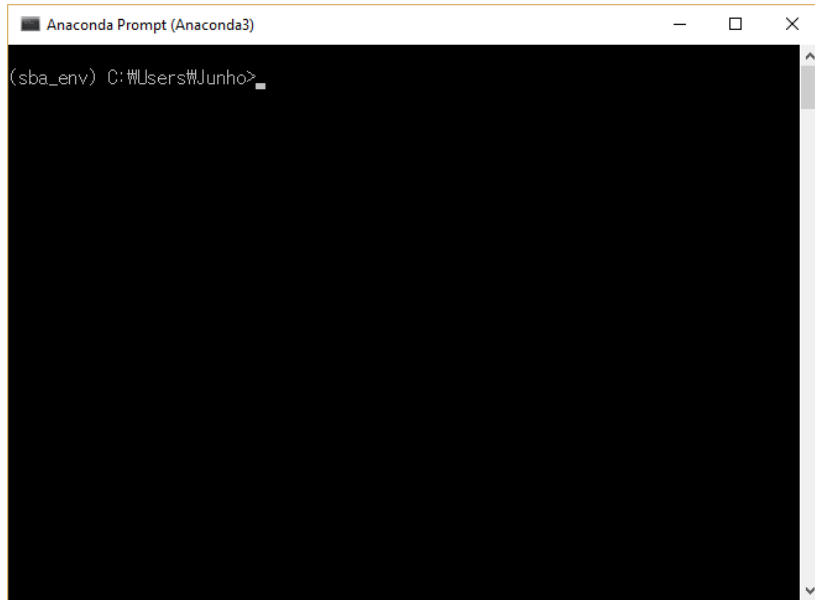
# Anaconda 가상환경

```
Anaconda Prompt (Anaconda3)
protobuf 3.9.1 pypi_0 pypi
pycparser 2.19 py37_0
pyopenssl 19.0.0 py37_0
pysocks 1.7.0 py37_0
python 3.7.4 h5263a28_0
requests 2.22.0 py37_0
setuptools 41.0.1 py37_0
six 1.12.0 py37_0
sqlite 3.29.0 he774522_0
tb-nightly 1.15.0a20190806 pypi_0 pypi
tensorflow 2.0.0rc0 pypi_0 pypi
termcolor 1.1.0 pypi_0 pypi
tf-estimator-nightly 1.14.0.dev2019080601 pypi_0 pypi
urllib3 1.24.2 py37_0
vc 14.1 h0510ff6_4
vs2015_runtime 14.16.27012 hf0eaf9b_0
werkzeug 0.15.5 py_0
wheel 0.33.4 py37_0
win_inet_pton 1.1.0 py37_0
wincertstore 0.2 py37_0
wrap 1.11.2 pypi_0 pypi
xmltodict 0.12.0 py_0

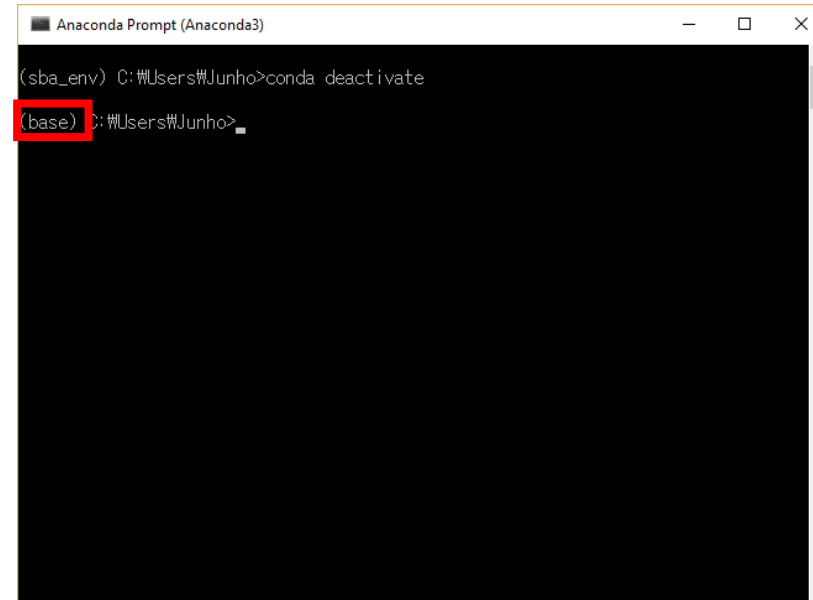
(sba_env) C:\Users\Junho>
```

➤ conda list

현재 활성화된 가상환경에 설치된 패키지 확인

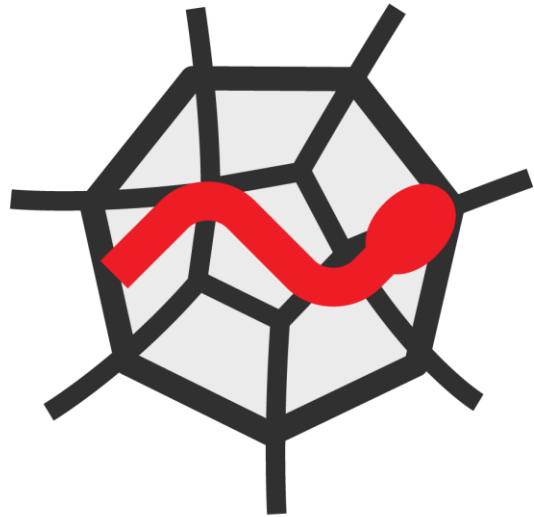


```
Anaconda Prompt (Anaconda3)
(sba_env) C:\Users\Junho>
```



```
Anaconda Prompt (Anaconda3)
(sba_env) C:\Users\Junho>conda deactivate
(base) C:\Users\Junho>
```

➤ conda deactivate



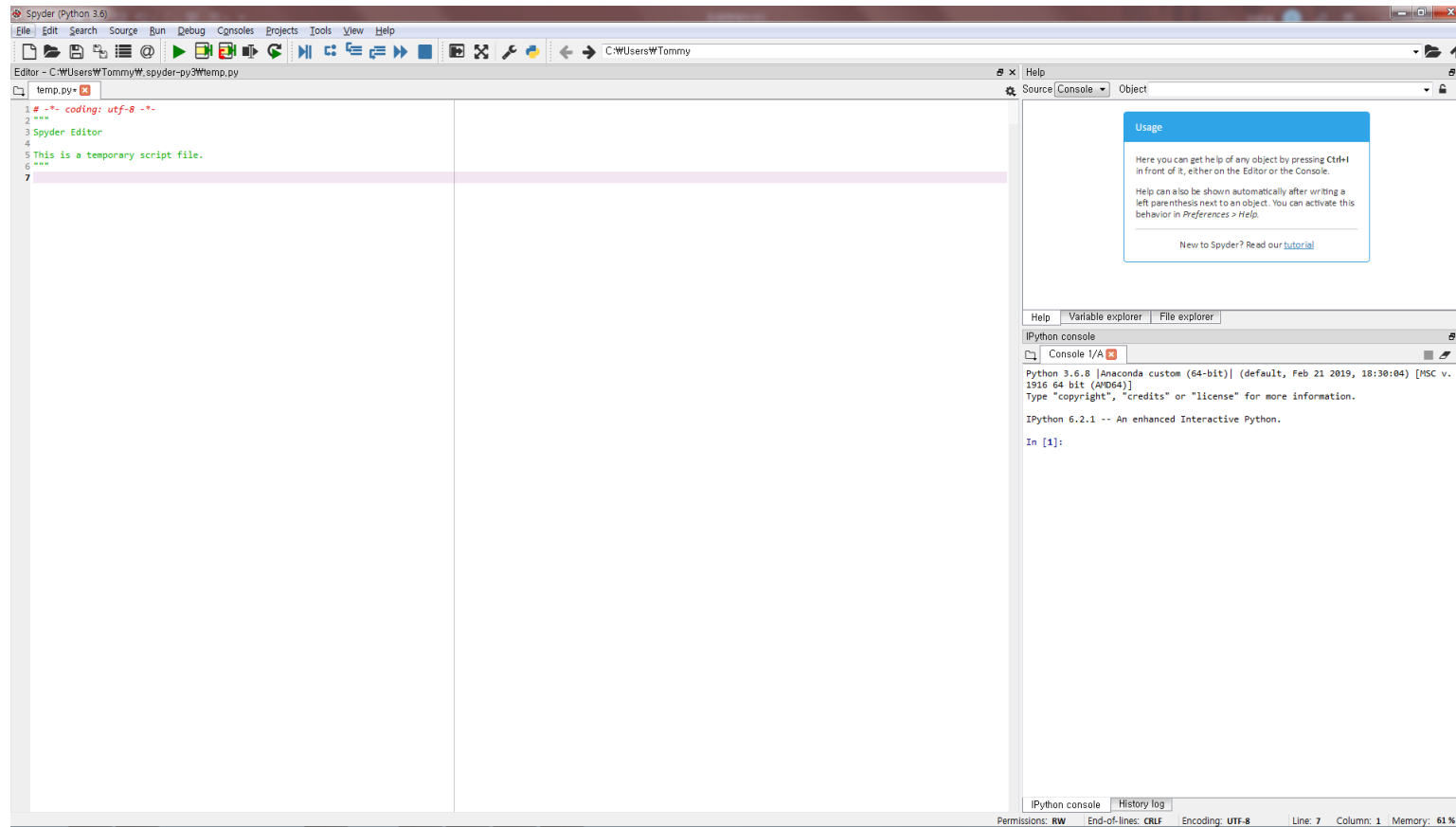
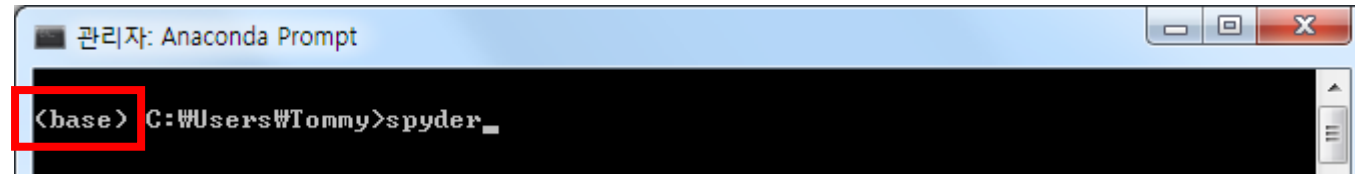
# SPYDER

---

The Scientific Python Development Environment

**Anaconda 에 포함된 무료 python IDE**  
(환경 통일을 위해 사용 권장)

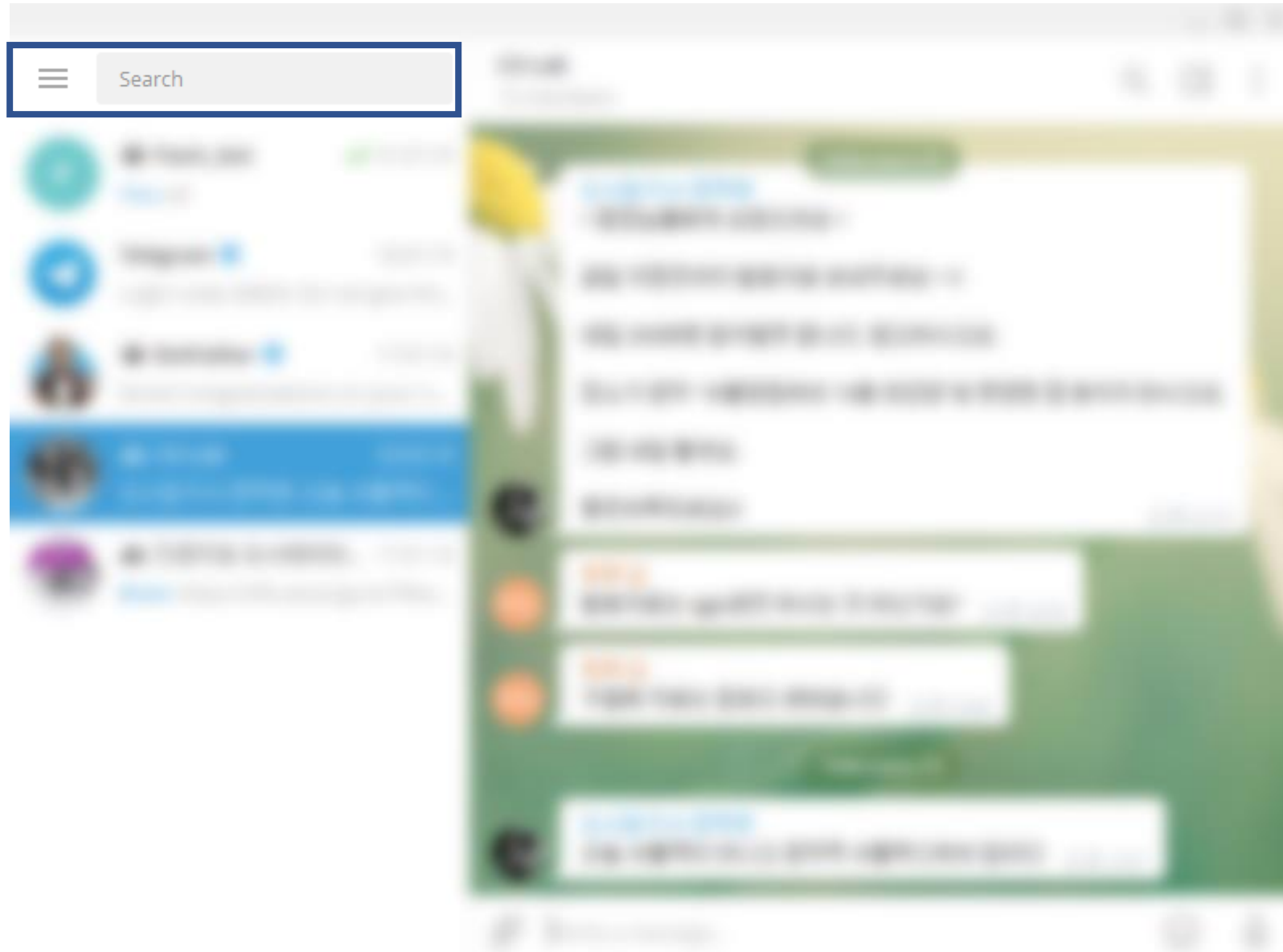
## Spyder 환경



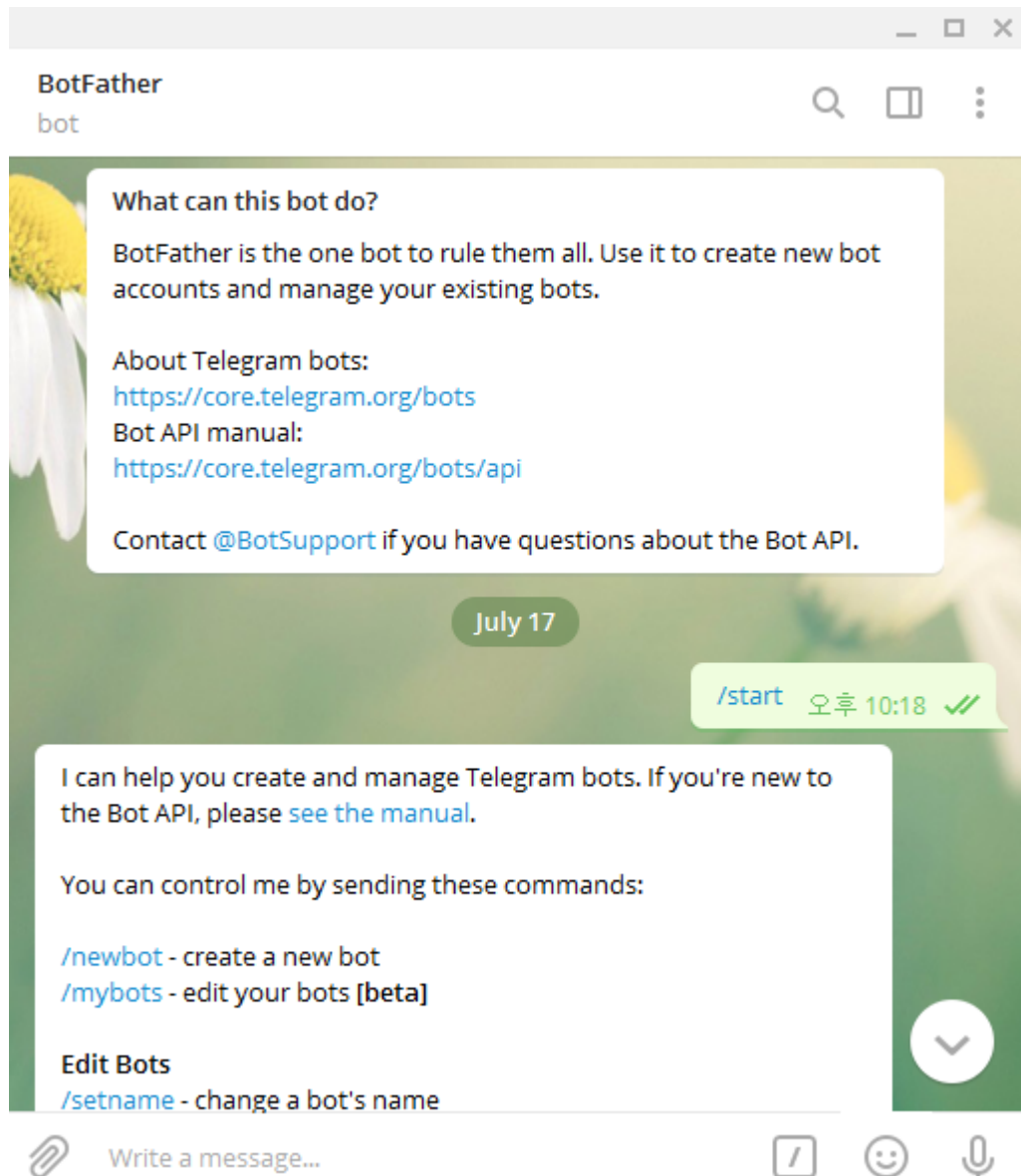


**챗봇 구현을 연습해보기 위한 플랫폼  
Telegram 메신저 회원가입 및 다운로드**

<https://telegram.org>



**BotFather** 검색



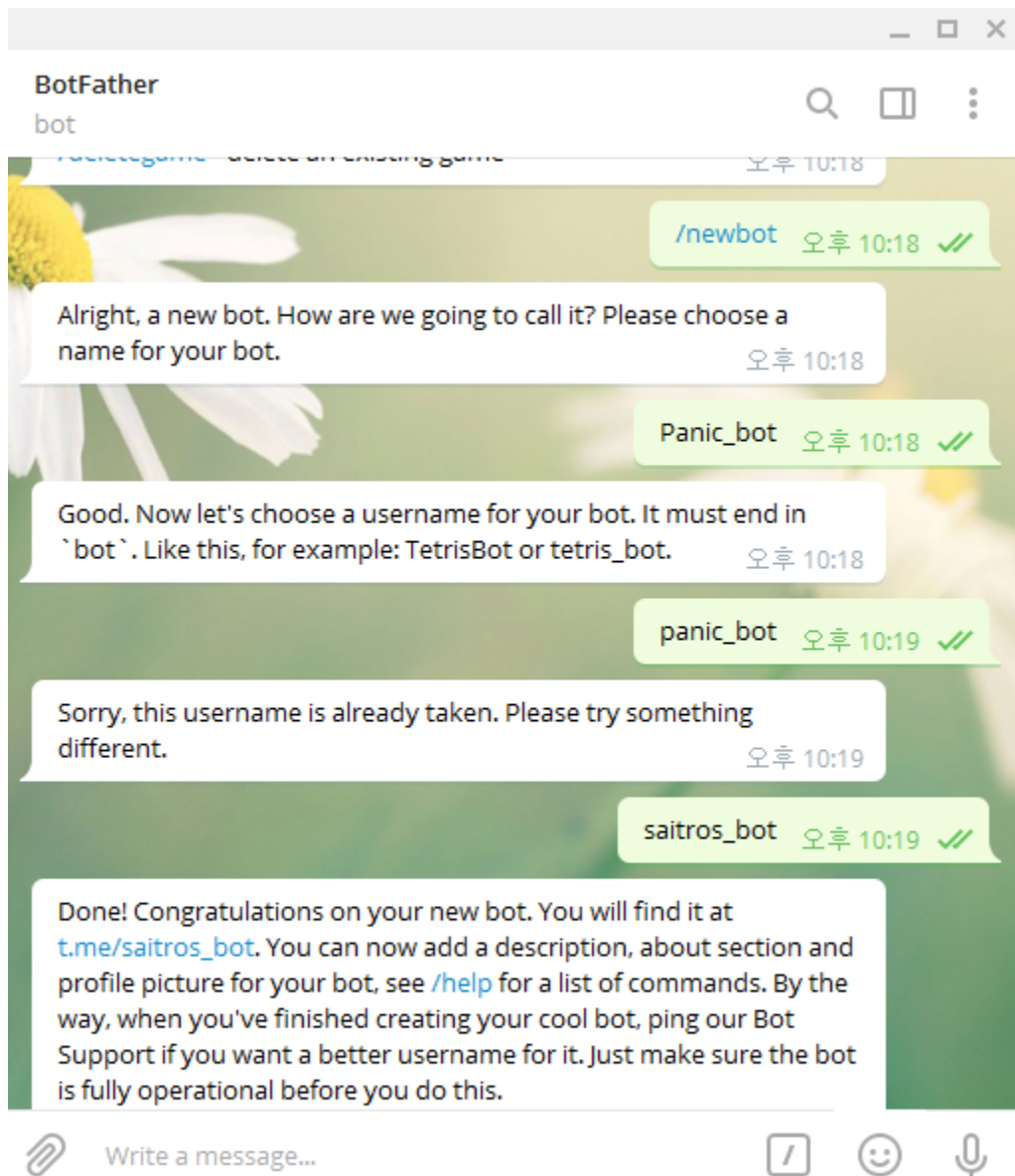
/start

채팅 입력으로 BotFather 활성화

/newbot

채팅 입력으로 봇 생성에 진입

# Telegram 설정



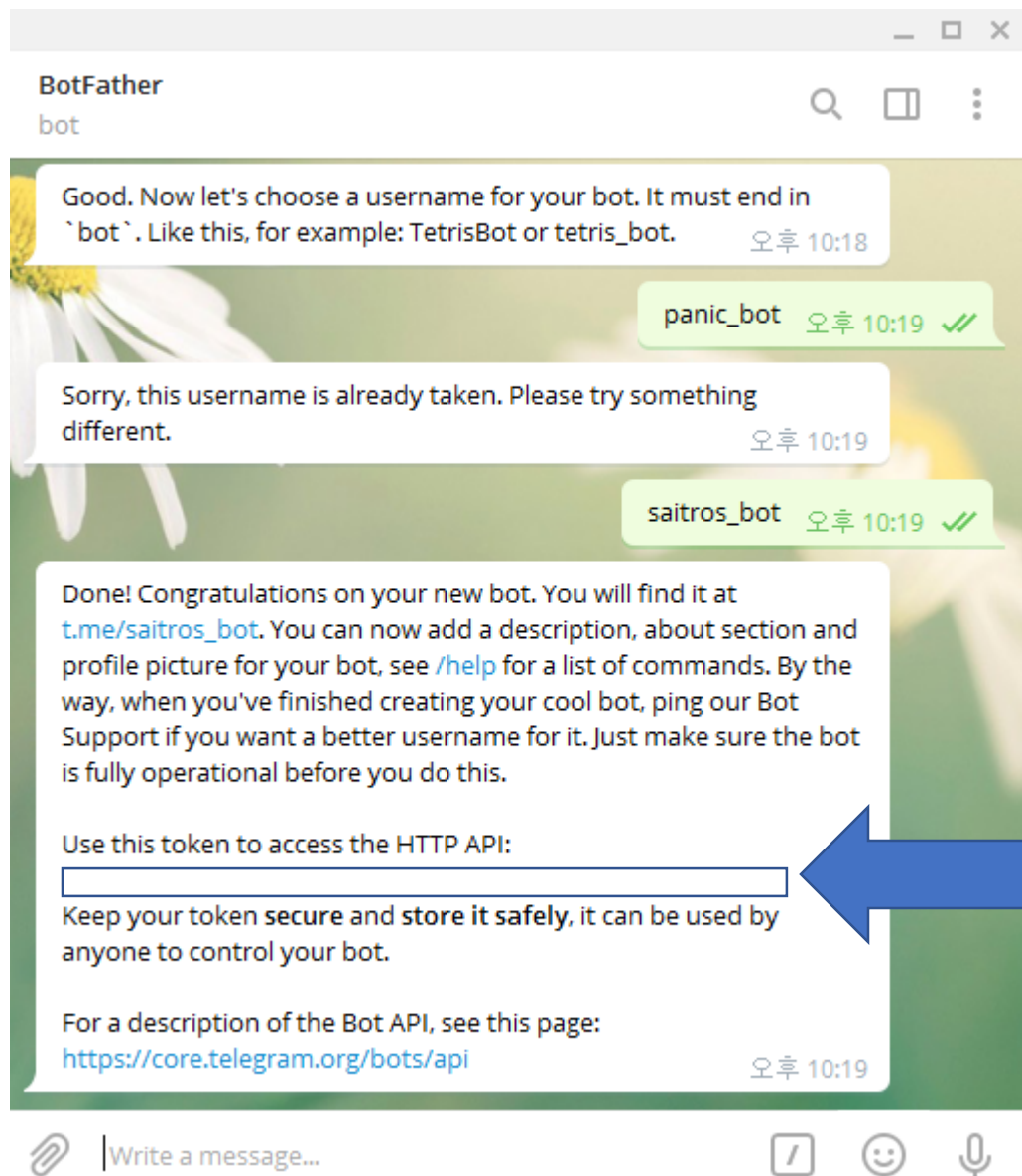
봇의 이름을 정하는 단계

다음으로 봇의 사용자 이름을 정하는 단계  
(마지막은 bot으로 끝나야 합니다. 검색용)

마지막으로  
Done! Congratulations ~~~ 나오면 완료



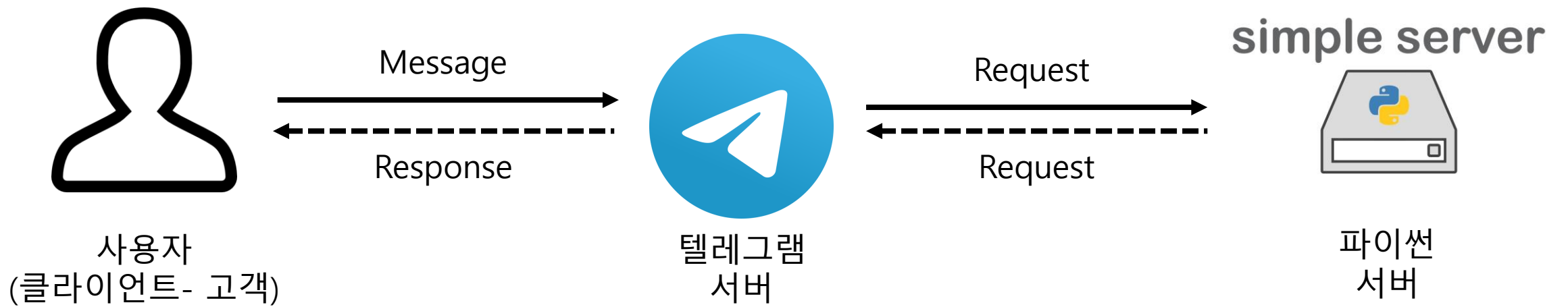
# Telegram 설정

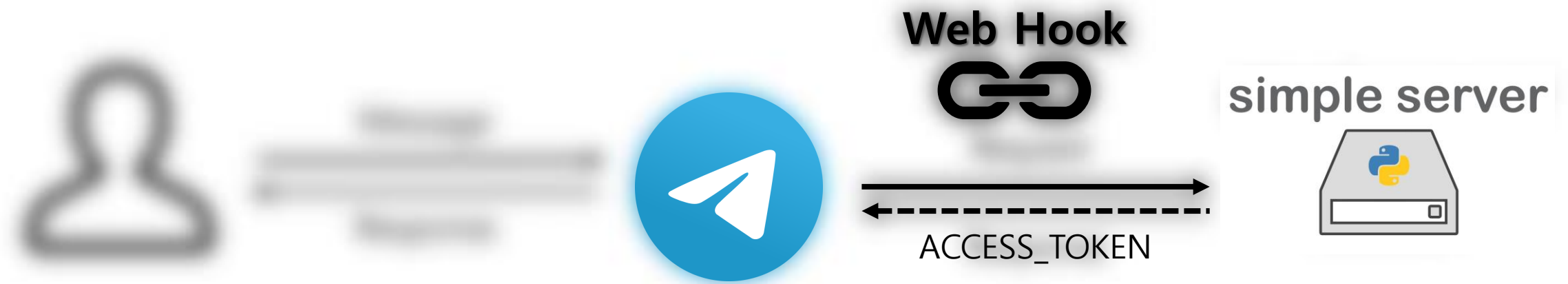


접근 가능한 HTTP API token

Bot 에 대한 제어를 위한 토큰

<ACCESS TOKEN>  
안전하게 보관







손님

내가 가고 싶을 때 간다



매장

1. 가게가 열려 있어야 손님이 온다
2. 손님은 문으로 들어온다
3. 주방에서 온 빅맥을 요청한 손님에게 전달한다.



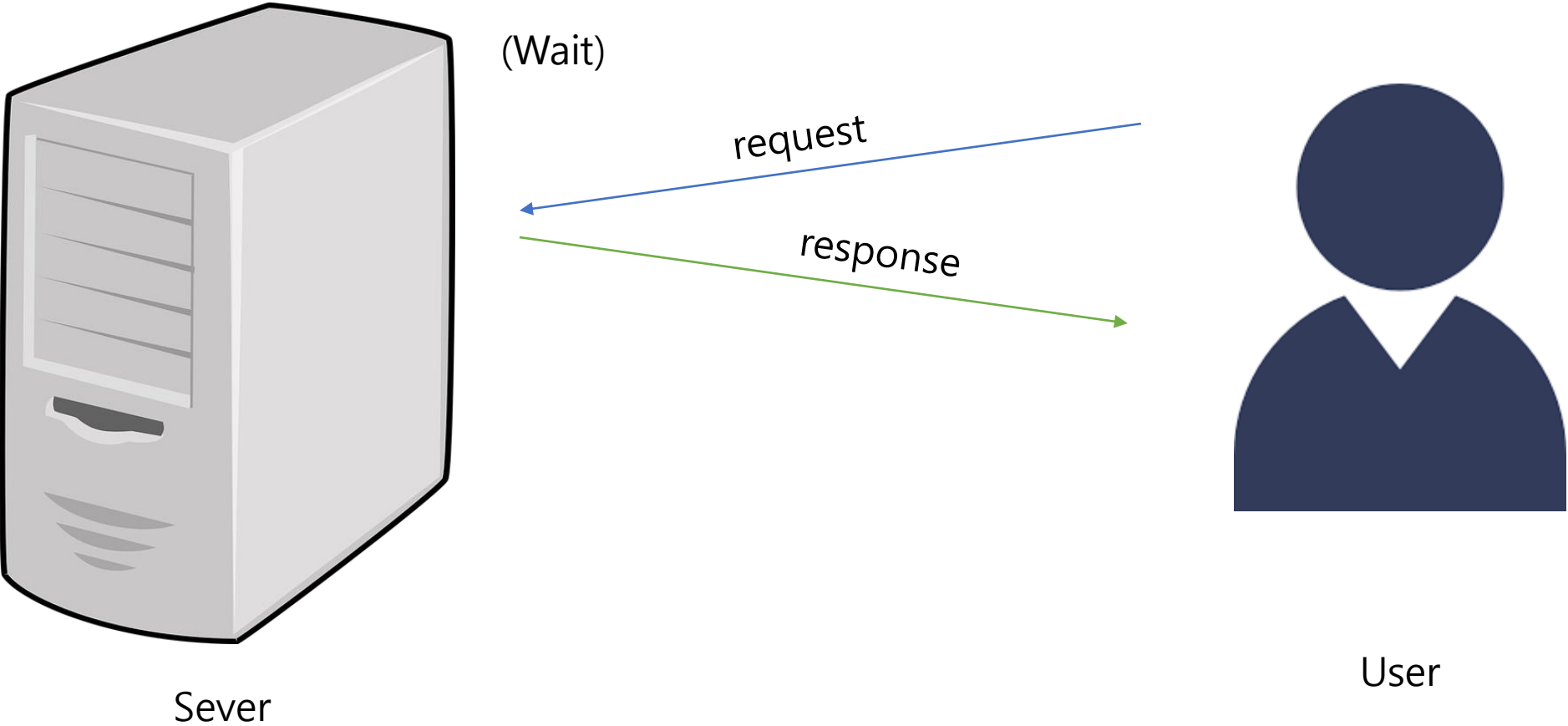
주방

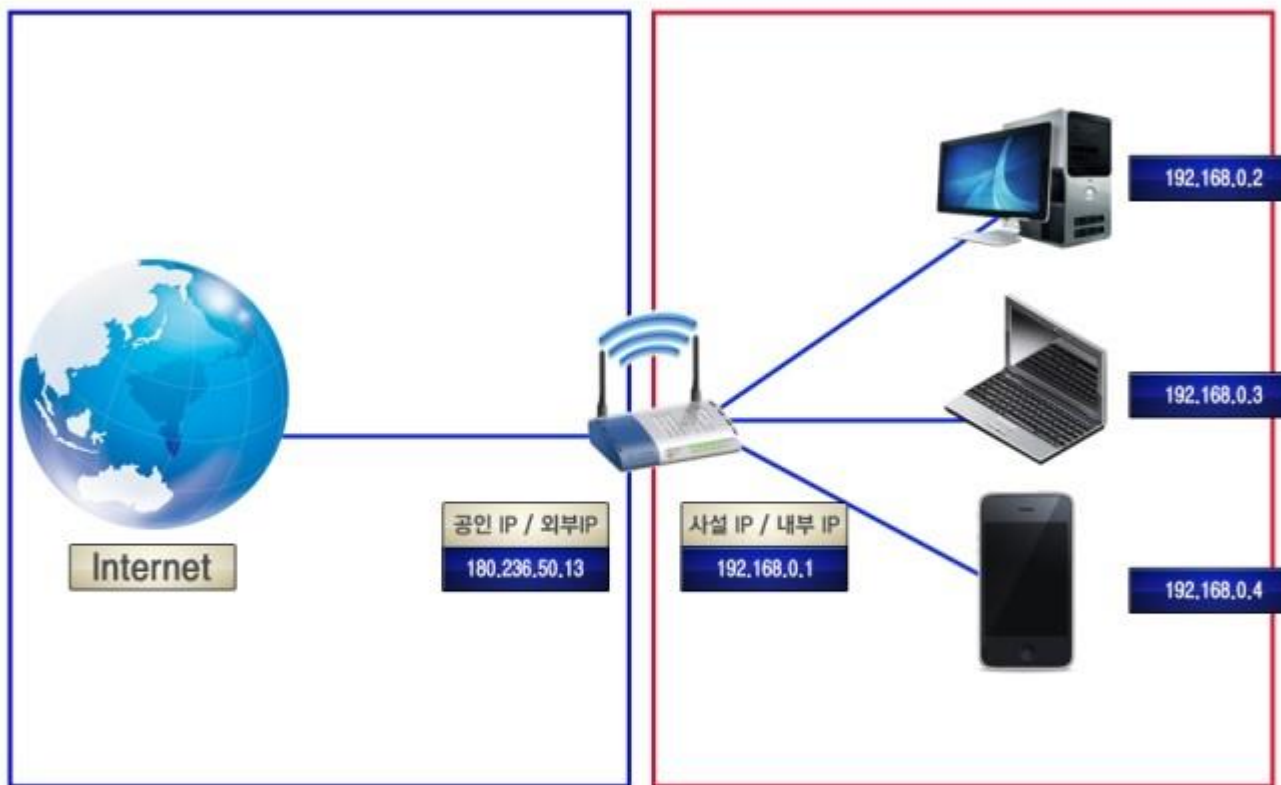
1. 빅맥 주문은 빅맥 만드는 사람에게
2. 이상한 사람이 만든 빅맥이 제공되면 안된다

클라이언트는 언제든지 요청한다

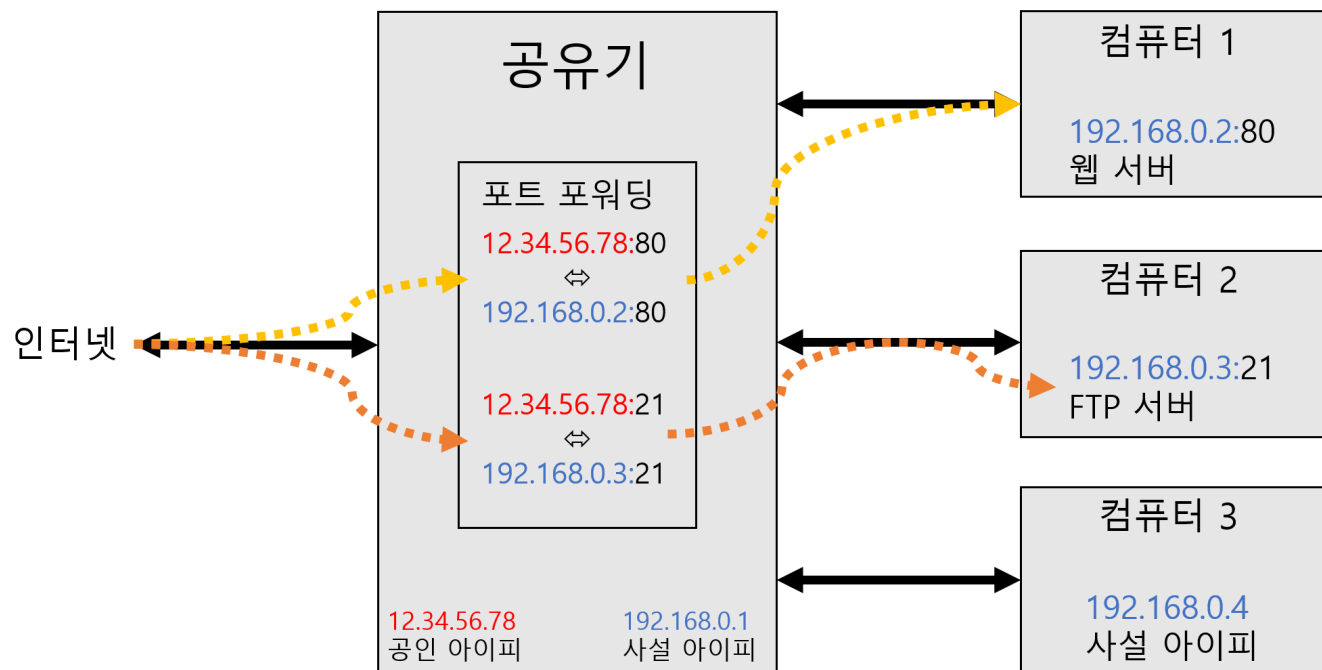
1. 서버는 클라이언트를 기다려야한다
2. 클라이언트는 URL 주소로 들어온다
3. 정확한 클라이언트에게 처리된 데이터를 전달한다.

1. 서버는 정확한 데이터 처리를 위해 경로를 설정한다 (WEBHOOK)
2. 다른 데이터가 전송되지 않게 보안 키를 확인한다 (ACCESS TOKEN)





내부 IP는 설정을 하지 않으면,  
외부에서 접속할 수 없다.



포트를 통해서 데이터 통신을 관리

80포트 → 웹 서버

21 포트 → FTP 서버

예시 )

채용 서류 → 인사과(80번 문)

회계 서류 → 총무과(21번 문)

101동 302호

→ 같은 아파트에 산다면 접근 가능

→ 그렇지 않으면, 어느 아파트?

```
Command Prompt

Windows IP Configuration

Ethernet adapter Ethernet:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::f57a:2421:9532:98d0%6
    IPv4 Address. . . . . : 192.168.35.218
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.35.1

Tunnel adapter Teredo Tunneling Pseudo-Interface:

    Connection-specific DNS Suffix  . : 
    IPv6 Address. . . . . : 2001:0:2851:782c:1cf8:af3:c587:3ad3
    Link-local IPv6 Address . . . . . : fe80::1cf8:af3:c587:3ad3%5
    Default Gateway . . . . . : ::

Tunnel adapter isatap.{EA42AB95-FFBD-479F-8758-159C76DE6CA2}:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : 

C:\Users\Junho>
```

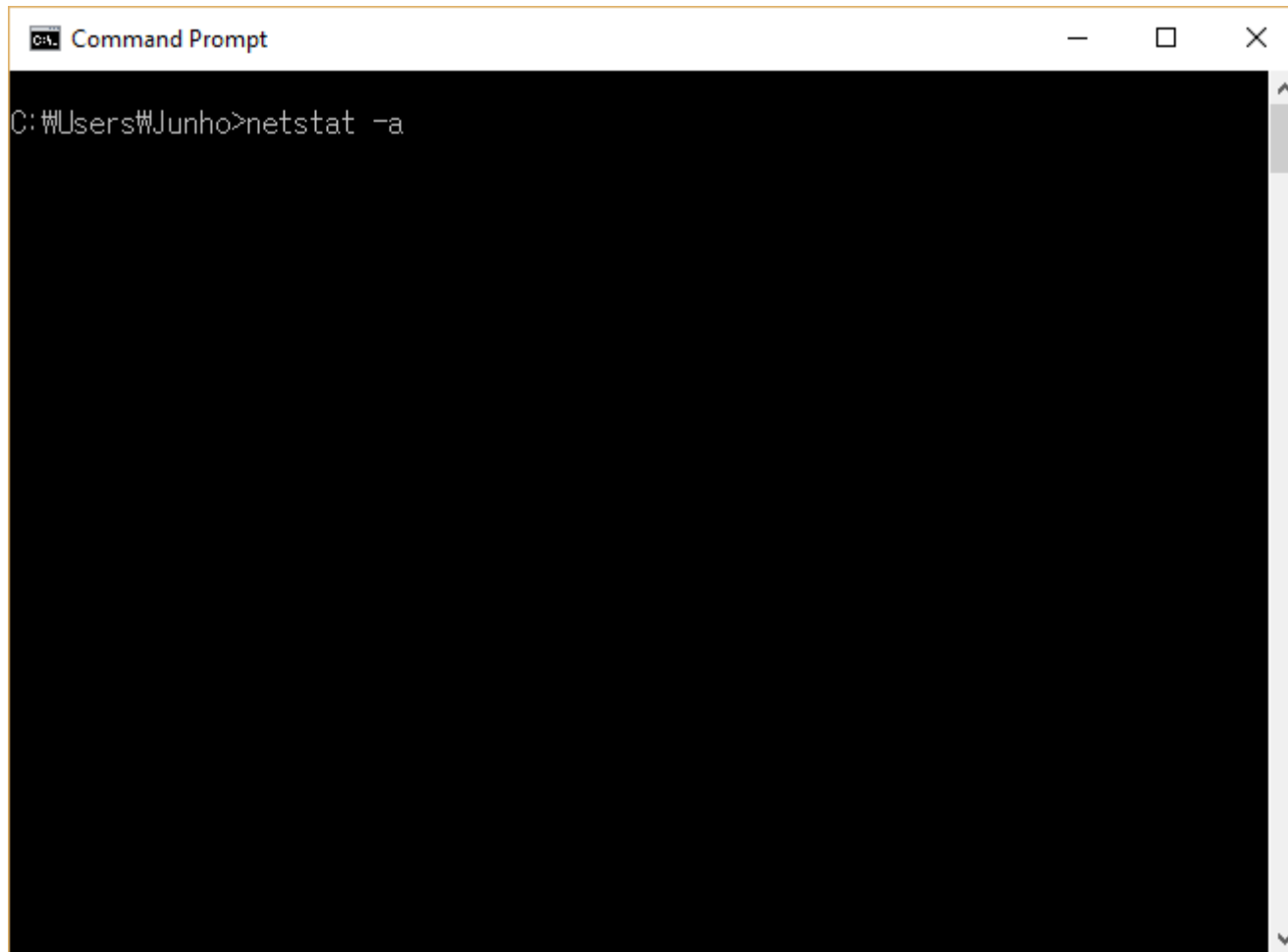
Window :

(Window Key) + R → cmd실행  
ipconfig 입력

Mac :

(터미널)  
ifconfig





```
C:\Users\Junho>netstat -a
```

Window, Mac :  
netstat -a 입력

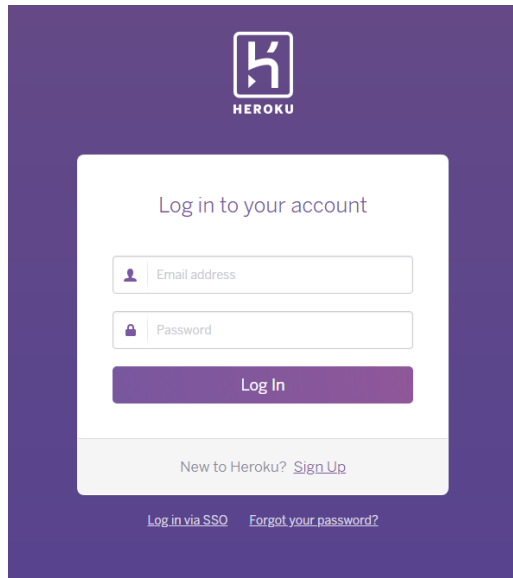
포트	TCP	UDP	설명	상태
0		UDP	예약됨; 사용하지 않음	공식
1	TCP		TCPMUX (TCP 포트 서비스 멀티플렉서)	공식
7	TCP	UDP	ECHO 프로토콜	공식
9	TCP	UDP	DISCARD 프로토콜	공식
13	TCP	UDP	DAYTIME 프로토콜	공식
17	TCP		QOTD (Quote of the Day) 프로토콜	공식
19	TCP	UDP	CHARGEN (Character Generator) 프로토콜 - 원격 오류 수정	공식
20	TCP		FTP (파일 전송 프로토콜) - 데이터 포트	공식
21	TCP		FTP - 제어 포트	공식
22	TCP		SSH (Secure Shell) - ssh scp, sftp같은 프로토콜 및 포트 포워딩	공식
23	TCP		텔넷 프로토콜 - 암호화되지 않은 텍스트 통신	공식
24	TCP		개인메일 시스템	공식
25	TCP		SMTP (Simple Mail Transfer Protocol) - 이메일 전송에 사용	공식
37	TCP	UDP	TIME 프로토콜	공식
49		UDP	TACACS 프로토콜	공식
53	TCP	UDP	DNS (Domain Name System)	공식
67		UDP	BOOTP (부트스트랩 프로토콜) 서버, DHCP로도 사용	공식
68		UDP	BOOTP (부트스트랩 프로토콜) 클라이언트, DHCP로도 사용	공식
69		UDP	TFTP	공식
70	TCP		고퍼 프로토콜	공식
79	TCP		Finger 프로토콜	공식
80	TCP	UDP	HTTP (HyperText Transfer Protocol) - 웹 페이지 전송	공식

Well Known Port Number

이런게 있다.

포트 설정을 잘못하면 복잡해짐

5000번 포트를 사용



- 원격 저장소에 파일을 업로드
- 24시간 사용
- 배포 할 때 편리
- 입출력 파일 확인이 불가능
- 사용이 어렵지 않음

<https://ngrok.com>



- 로컬 주소를 포트 포워딩
- 한번 할당에 8시간 사용
- 개발 할 때 편리
- 입출력 파일 확인이 쉬움
- 사용이 쉬움



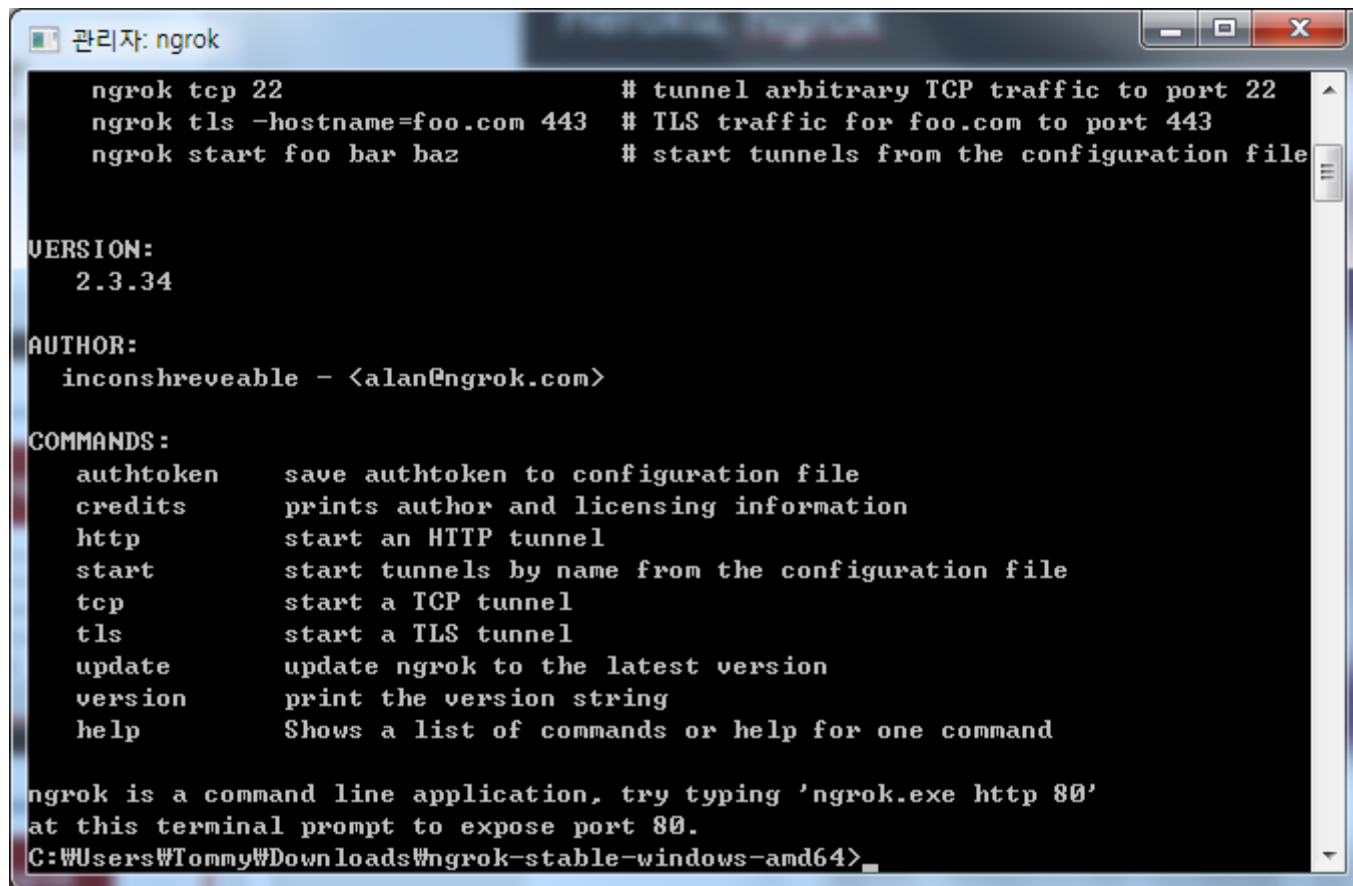
- 로컬 주소로 포트 포워딩
- 한번 할당에 8시간 사용
- 개발 할 때 편리
- 입출력 파일 확인이 쉬움
- 사용이 쉬움

<https://ngrok.com>



- 로컬 주소를 포트 포워딩
- 한번 할당에 8시간 사용
- 개발 할 때 편리
- 입출력 파일 확인이 쉬움
- 사용이 쉬움

# Ngrok for test



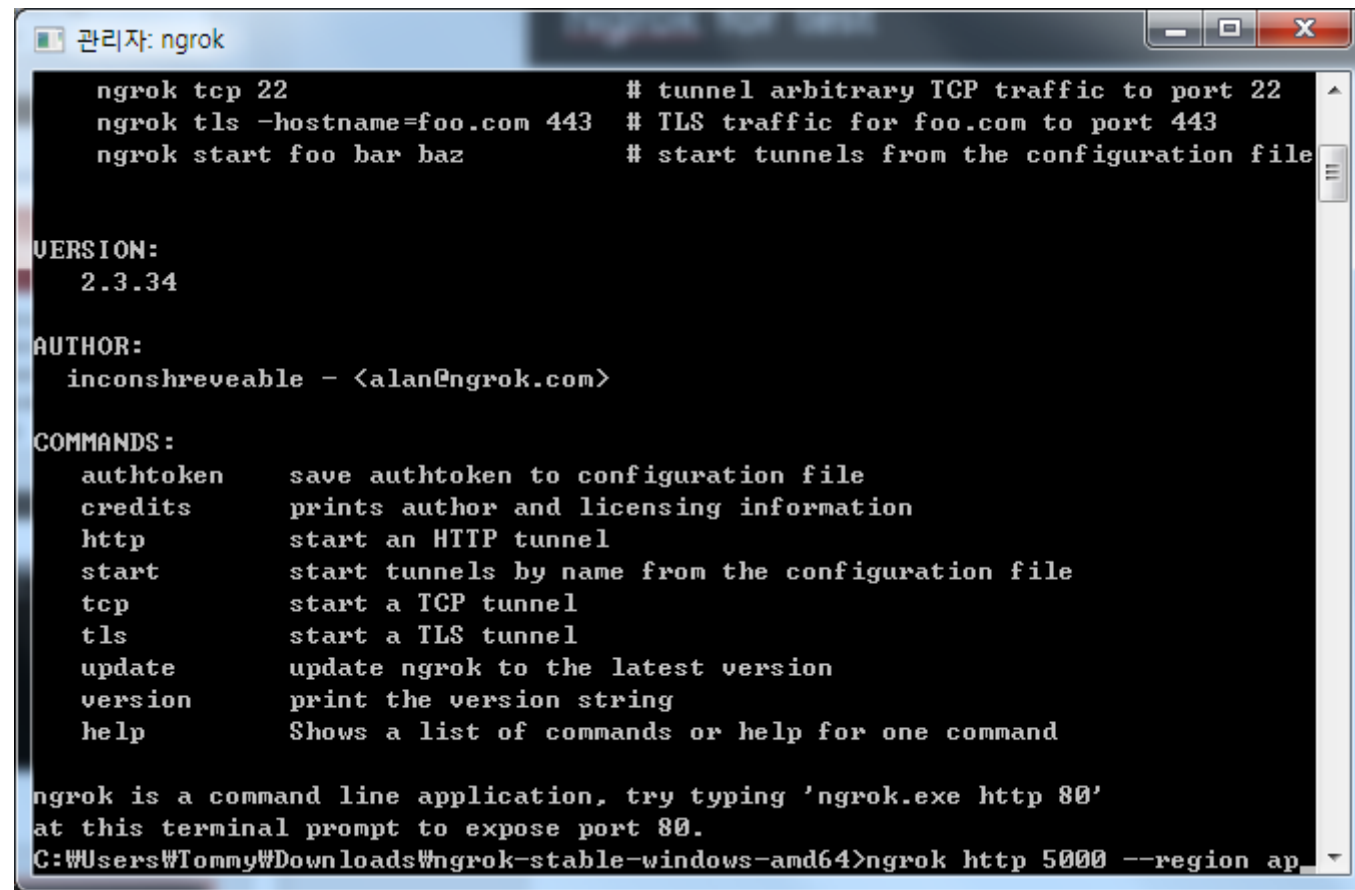
```
ngrok tcp 22 # tunnel arbitrary TCP traffic to port 22
ngrok tls -hostname=foo.com 443 # TLS traffic for foo.com to port 443
ngrok start foo bar baz # start tunnels from the configuration file

VERSION:
  2.3.34

AUTHOR:
  inconshreveable - <alan@ngrok.com>

COMMANDS:
  authtoken  save authtoken to configuration file
  credits    prints author and licensing information
  http       start an HTTP tunnel
  start      start tunnels by name from the configuration file
  tcp        start a TCP tunnel
  tls        start a TLS tunnel
  update     update ngrok to the latest version
  version    print the version string
  help       Shows a list of commands or help for one command

ngrok is a command line application, try typing 'ngrok.exe http 80'
at this terminal prompt to expose port 80.
C:\Users\Tommy\Downloads\ngrok-stable-windows-amd64>
```



```
관리자: ngrok

ngrok tcp 22                # tunnel arbitrary TCP traffic to port 22
ngrok tls -hostname=foo.com 443 # TLS traffic for foo.com to port 443
ngrok start foo bar baz      # start tunnels from the configuration file

VERSION:
  2.3.34

AUTHOR:
  inconshreveable - <alan@ngrok.com>

COMMANDS:
  authtoken  save authtoken to configuration file
  credits    prints author and licensing information
  http       start an HTTP tunnel
  start      start tunnels by name from the configuration file
  tcp        start a TCP tunnel
  tls        start a TLS tunnel
  update     update ngrok to the latest version
  version    print the version string
  help       Shows a list of commands or help for one command

ngrok is a command line application, try typing 'ngrok.exe http 80'
at this terminal prompt to expose port 80.
C:\Users\Tommy\Downloads\ngrok-stable-windows-amd64>ngrok http 5000 --region ap
```

➤ ngrok http 5000 --region ap

```
관리자: ngrok - ngrok http 5000 --region ap
ngrok by @inconshreveable (Ctrl+C to quit)

Session Status      online
Session Expires     7 hours, 59 minutes
Version             2.3.34
Region              Asia Pacific (ap)
Web Interface        http://127.0.0.1:4040
Forwarding           http://3c4754fd.ap.ngrok.io -> http://localhost:5000
Forwarding           https://3c4754fd.ap.ngrok.io -> https://localhost:5000

Connections
  ttl    opn    rt1    rt5    p50    p90
    0      0    0.00    0.00    0.00    0.00
```

외부에서 접근  
가능한 주소

WEBHOOK\_URL

```
2 """
3 목표 : 가장 기본적인 형태를 익혀봅시다.
4 """
5 from flask import Flask
6
7 app = Flask(__name__)
8
9
10 @app.route('/', methods=['GET', 'POST'])
11 def receive_message():
12     return 'Hello World!'
13
14 @app.route('/Home', methods=['GET', 'POST'])
15 def home():
16     return 'My Sweet Home!'
17
18 if __name__ == '__main__':
19     app.run()
20
```

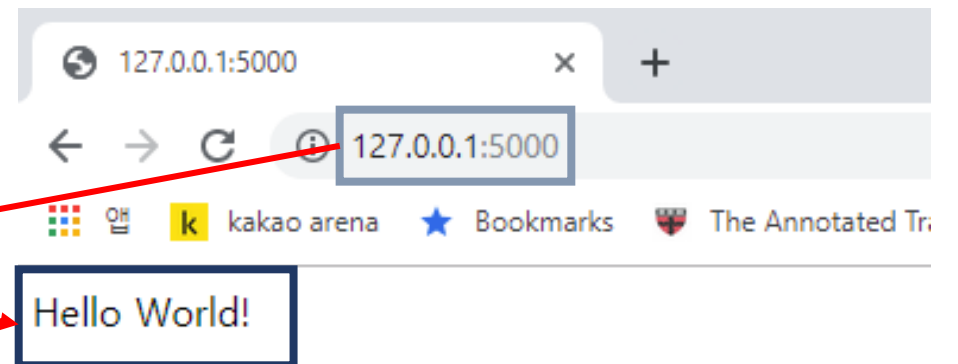
Python 3.6.4 |Anaconda, Inc.| (default, Jan 16 2018, 10:22:32) [MSC v.1900 64 bit (AMD64)]  
Type "copyright", "credits" or "license" for more information.

IPython 6.2.1 -- An enhanced Interactive Python.

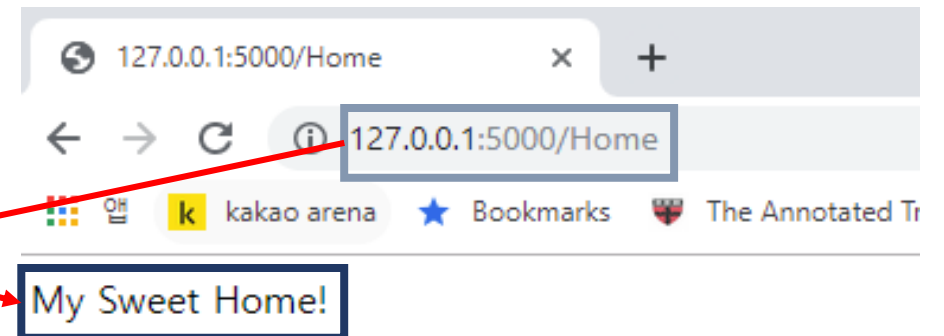
In [1]: runfile('F:/FB\_bot\_tuto/tuto00.py', wdir='F:/FB\_bot\_tuto')  
\* Serving Flask app "tuto00" (lazy loading)  
\* Environment: production  
 WARNING: Do not use the development server in a production environment.  
 Use a production WSGI server instead.  
\* Debug mode: off  
\* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)



```
2 """
3 목표 : 가장 기본적인 형태를 익혀봅시다.
4 """
5 from flask import Flask
6
7 app = Flask(__name__)
8
9
10 @app.route('/', methods=['GET', 'POST'])
11 def receive_message():
12     return 'Hello World!'
13
14 @app.route('/Home', methods=['GET', 'POST'])
15 def home():
16     return 'My Sweet Home!'
17
18 if __name__ == '__main__':
19     app.run()
20
```



```
2 """
3 목표 : 가장 기본적인 형태를 익혀봅시다.
4 """
5 from flask import Flask
6
7 app = Flask(__name__)
8
9
10 @app.route('/', methods=['GET', 'POST'])
11 def receive_message():
12     return 'Hello World!'
13
14 @app.route('/Home', methods=['GET', 'POST'])
15 def home():
16     return 'My Sweet Home!'
17
18 if __name__ == '__main__':
19     app.run()
20
```



```
1 from flask import Flask
2 from break_time import time_to_coffee
3
4 app = Flask(__name__)
5
6 print('쉬는 시간을 가질까요?')
7 print(time_to_coffee())
8
9
10 @app.route('/', methods=['GET', 'POST'])
11 def hello_world():
12     return 'Hello World!'
13
14
15 @app.route('/break')
16 def break_time():
17     return 'python 에서 정상적인 출력이 나오면 10분간 쉬도록하겠습니다'
18
19
20 if __name__ == '__main__':
21     app.run()
22
```

Basic\_flask.py

```
1 def time_to_coffee():
2     return "아주 좋소!"
3
4
5 print('이 문장은 출력되면 안됩니다')
6
7
```

break\_time.py

```
Flask (tuto00/basic_flask.py) x
* Serving Flask app "tuto00/basic_flask.py"
* Environment: development
* Debug mode: off
이 문장은 출력되면 안됩니다
쉬는 시간을 가질까요?
아주 좋소!
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Basic\_flask.py 실행화면

```
1 def time_to_coffee():
2     return "아주 좋소!"
3
4
5 print('이 문장은 출력되면 안됩니다')
6
7
```

break\_time.py

```
1 def time_to_coffee():
2     return "아주 좋소!"
3
4
5 if __name__ == '__main__':
6     print('이 문장은 출력되면 안됩니다')
7
8
```

break\_time.py (수정)

Run: Flask (tuto00/basic\_flask.py) x

```
* Environment: development
* Debug mode: off
쉬는 시간을 가질까요?
아주 좋소!
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

Process finished with exit code -1
```

break\_time.py 수정 후 basic\_flask.py 실행화면

```
API_KEY = <ACCESS_KEY>
WEBHOOK_URL = <WEBHOOK_URL>
```

```
API_KEY = '[REDACTED]'
WEBHOOK_URL = 'https:[REDACTED].ap.ngrok.io'
BOT_INFO_URL = 'https://api.telegram.org/bot{API_KEY}/getMe'.format(API_KEY=API_KEY)
BOT_UPDATE_URL = 'https://api.telegram.org/bot{API_KEY}/Update'.format(API_KEY=API_KEY)
BOT_SET_WEBHOOK_URL = 'https://api.telegram.org/bot{API_KEY}/setWebhook?url={WEBHOOK_URL}'.format(API_KEY=API_KEY, WEBHOOK_URL=WEBHOOK_URL)
BOT_DELETE_URL = 'https://core.telegram.org/bot{API_KEY}/deleteWebhook'.format(API_KEY=API_KEY)
```

```
41
42 bot_set_webhook_call()
```

결과 :

```
{"ok":true,"result":true,"description":"Webhook was set"}
```

1. 각각 발급받은 API\_KEY 를 변수에 할당
2. Ngrok에서 발급받은 URL 을 변수에 할당
3. bot\_set\_webhook\_call( ) 함수 실행(파일 하단)

## Telegram Bot API

The Bot API is an HTTP-based interface created for developers keen on building bots for Telegram. To learn how to create and set up a bot, please consult our [Introduction to Bots](#) and [Bot FAQ](#).

## Recent changes

July 29, 2019

Bot API 4.4

- Added support for **animated stickers**. New field *is\_animated* in [Sticker](#) and [StickerSet](#) objects, animated stickers can now be used in [sendSticker](#) and [InlineQueryResultCachedSticker](#).
- Added support for **default permissions** in groups. New object [ChatPermissions](#), containing actions which a member can take in a chat. New field *permissions* in the [Chat](#) object; new method [setChatPermissions](#).
- The field *all\_members\_are\_administrators* has been removed from the documentation for the [Chat](#) object. The field is still returned in the object for backward compatibility, but new bots should use the *permissions* field instead.
- Added support for more permissions for group and supergroup members: added the new field *can\_send\_polls* to [ChatMember](#) object, added *can\_change\_info*, *can\_invite\_users*, *can\_pin\_messages* in [ChatMember](#) object for restricted users (previously available only for administrators).
- The method [restrictChatMember](#) now takes the new user permissions in a single argument of the type [ChatPermissions](#). The old way of passing parameters will keep working for a while for backward compatibility.
- Added *description* support for basic groups (previously available in supergroups and channel chats). You can pass a group's chat\_id to [setChatDescription](#) and receive the group's description in the [Chat](#) object in the response to [getChat](#) method.
- Added *invite\_link* support for basic groups (previously available in supergroups and channel chats). You can pass a group's chat\_id to [exportChatInviteLink](#) and receive the group's invite link in the [Chat](#) object in the response to [getChat](#) method.
- File identifiers from the [ChatPhoto](#) object are now invalidated and can no longer be used whenever the photo is changed.
- All **webhook requests** from the Bot API are now coming from the subnets `149.154.160.0/20` and `91.108.4.0/22`. Most users won't need to do anything to continue receiving webhooks. If you control inbound access with a firewall, you may need to update your configuration. You can always find the list of actual IP addresses of servers used to send webhooks there: <https://core.telegram.org/bots/webhooks>.
- As of the **next Bot API update (version 4.5)**, nested [MessageEntity](#) objects will be allowed in message texts and captions. Please make sure that your code can correctly handle such entities.

### Recent changes

July 29, 2019  
May 31, 2019  
April 14, 2019

### Authorizing your bot

### Making requests

### Getting updates

### Available types

### Available methods

### Updating messages

### Stickers

### Inline mode

### Payments

### Telegram Passport

### Games

<https://core.telegram.org/bots/api#getupdates>

## Tuto02

```
1  # -*- coding: utf-8 -*-
2  import ...
3
4
5  API_KEY = 'API_KEY'
6
7  # Flask 객체를 생성 __name__ 을 인수로 입력
8  app = Flask(__name__)
9
10
11 # 경로 설정, URL 설정
12 @app.route('/', methods=['POST', 'GET'])
13 def index():
14     if request.method == 'POST':
15         message = request.get_json()
16
17         with open('response.json', 'w', encoding='UTF-8') as f:
18             json.dump(message, f, indent=4, ensure_ascii=False)
19
20         return Response('ok', status=200)
21     else:
22         return "Hello World!"
23
24
25 if __name__ == '__main__':
26     app.run(port=5000)
```

```
{
  "update_id": 108939676,
  "message": {
    "message_id": 174,
    "from": {
      "id": 727895200,
      "is_bot": false,
      "first_name": "이",
      "last_name": "준호",
      "language_code": "ko"
    },
    "chat": {
      "id": 727895200,
      "first_name": "이",
      "last_name": "준호",
      "type": "private"
    },
    "date": 1563941727,
    "text": "안녕하세요"
  }
}
```

```
def parse_message(message):
```

```
    """
```

telegram 에서 data 인자를 받아옴  
data 내부 구조를 이해해야 한다.

Return :

```
chat_id = 사용자 아이디 코드
```

```
msg = 사용자 대화 내용
```

```
    """
```

```
chat_id = message['message']['chat']['id']
```

```
msg = message['message']['text']
```

```
return chat_id, msg
```

```
def send_message(chat_id, text='hello'):
```

```
    """
```

chat\_id : 사용자 아이디 코드

text : 사용자 대화내용

함수에 변수를 할당할때 text='bla-bla-bla' 라고 선언

--> text에 관련된 값이 전달되지 않으면 bla-bla-bla를 default로 사용

사용자에게 메시지를 보내는 내용의 함수

```
    """
```

```
url = 'https://api.telegram.org/bot{token}/sendMessage'.format(token=API_KEY)
```

# 변수들을 딕셔너리 형식으로 묶음

# 사용자에게 보내는 text는 사용자가 보낸 text와 똑같다

# 똑같은 소리를 한다고 해서 Echo\_bot

```
params = {'chat_id':chat_id, 'text': '안녕!'}
```

# Url 에 params 를 json 형식으로 변환하여 전송

# 메시지를 전송하는 부분

```
response = requests.post(url, json=params)
```

```
print(response)
```

```
return response
```



```
def send_message(chat_id, text='hello'):
    """
    chat_id : 사용자 아이디 코드
    text : 사용자 대화내용
    함수에 변수를 할당할때 text='bla-bla-bla' 라고 선언
    --> text에 관련된 값이 전달되지 않으면 bla-bla-bla를 default로 사용

    사용자에게 메시지를 보내는 내용의 함수
    """
    url = 'https://api.telegram.org/bot{token}/sendMessage'.format(token=API_KEY)
    # 변수들을 딕셔너리 형식으로 묶음
    # 사용자에게 보내는 text는 사용자가 보낸 text와 똑같다
    # 똑같은 소리를 한다고 해서 Echo_bot
    params = {'chat_id': chat_id, 'text': text}

    # Url 에 params 를 json 형식으로 변환하여 전송
    # 메시지를 전송하는 부분
    response = requests.post(url, json=params)
    print(response)
    return response
```

```

keyboard = {                                     # Keyboard 형식
    'keyboard': [[{
        'text': '버튼1'
    },
    {'text': '버튼2'}
    ]],
    [{'text': '버튼3'},
    {'text': '버튼4'}]
    ],
    'one_time_keyboard': True
}

```

```

if text == '버튼1':                             # 사용자가 button_1 이라고 응답하면 ~

```

```

    keyboard = {
        'keyboard': [[{'text': '버튼1을 누르셨습니다.'}]],
        'one_time_keyboard': True
    }

```

```

params = {'chat_id': chat_id, 'text': text, 'reply_markup': keyboard}

```

```

requests.post(url, json=params)

```