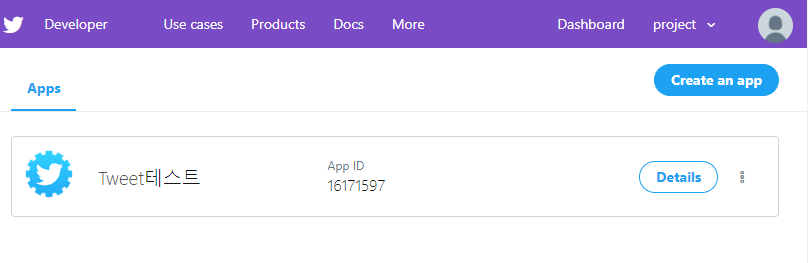
**환경**

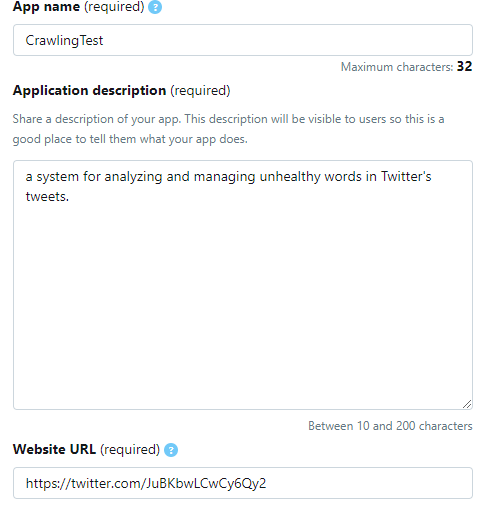
Twitter의 데이터를 가져오기 위해서 Tweepy를 이용하여 데이터를 모아올 것이다. Tweepy를 실행하기 전에 해야 할 작업들이다.

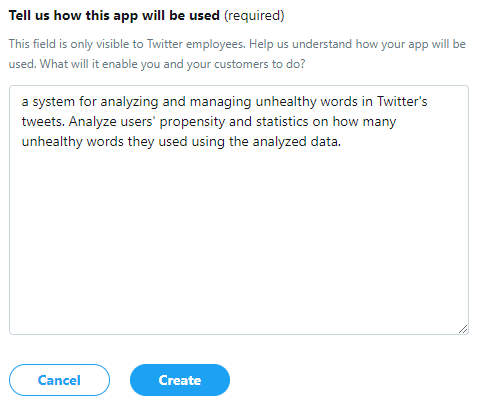
**1. API 키 받아오기**

1) <https://developer.twitter.com/> 접속하여 Apps -> Create an App 클릭.



2) details 설정





여러 항목이 있지만 필수적으로 작성해야하는(required) 항목만 작성해주면 된다.

App name – 앱의 이름

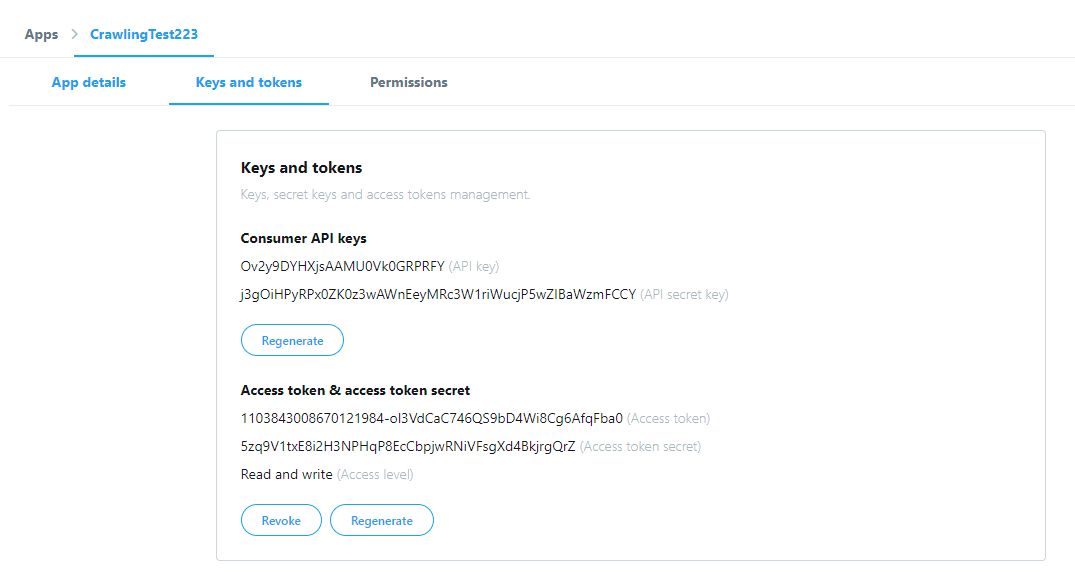
Application Description – 앱에 대한 소개

Website URL – 앱에서 보여질 URL. API 키를 받아오기만 하면 되기에 어떤 주소를 써도 상관없다.

Tell us how this app will be used – 앱이 어떻게 쓰이는가

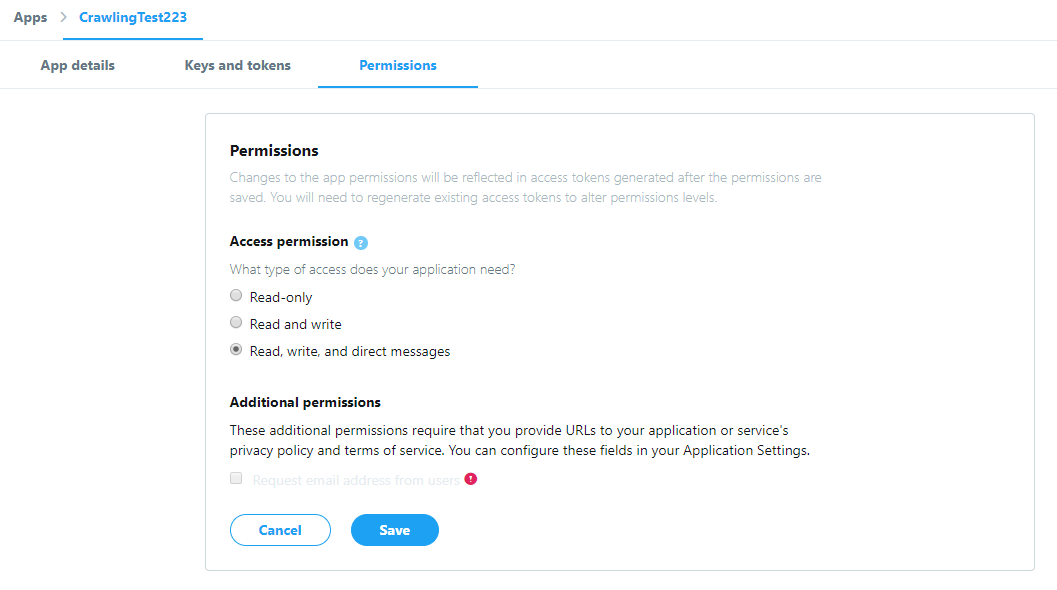
이후 Create 를 눌러서 앱 등록을 한다.

3) API Key와 Access Token 받기



앱이 등록되었다면, Keys and tokens를 눌러서 API 키와 Access token을 발급받는다.

4) Permission 설정



Permission 설정을 기본값인 Read and write 에서 Read, write and direct messages로 변경

**2. 파이선(3.6.7) 설치**

Tweepy를 사용하기 위해서 파이선을 이용할 것인데, Ubuntu의 Terminal 창에서 다음 명령어를 입력한다.

sudo apt-get install python3

설치가 제대로 되었는지 확인을 하려면 다음 명령어를 입력한다.

python3 --version



**3. tweepy(3.5.0) 설치**

트위터에 있는 단어들을 수집하기 위해서는 tweepy라는 모듈을 사용해야 한다. 이때 tweepy는 보통 Streaming과 Rest API, 2가지 종류로 나뉘게 된다.

1) Streaming API

실시간으로 데이터 크롤링. 많은 양의 트윗을 가져온다. 처리 속도가 느리고 일주일 양의 데이터밖에 가져오지 못한다.

2) REST API

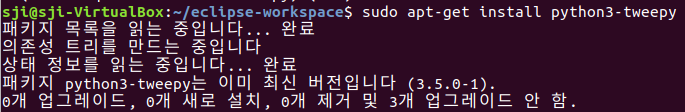
과거에 기록된 데이터들을 가져온다. 2013년부터 자료를 가져올 수 있다. 또한 15분마다 180개 제한이 있는데, 제한을 해제하려면 요금을 더 지불해야한다.

Streaming API를 사용하려면 모을 수 있는 데이터의 양이 한정적이기에 Rest API를 사용할 것이다.

Tweepy를 설치하기 위해서는 Ubuntu의 Terminal 창에서 다음 명령어를 입력한다.

sudo apt-get install python3-tweepy

제대로 설치가 되었다면 이런 문구를 확인 할 수 있다.



**Python에서 Twitter API 인증**

API 인증 시, OAuth 자격 증명을 처리하게 되는데, Tweepy에 있는 OAuthHandler를 이용하여 진행한다. 여기서 OAuth는 인터넷 사용자들이 비밀번호를 제공하지 않고 다른 웹사이트 상의 자신들의 정보에 대해 웹사이트나 애플리케이션의 접근 권한을 부여할 수 있는 공통적인 수단으로서 사용되는 접근 위임을 위한 개방형 표준이다.

**인증방식**

1) Request Token 요청 및 발급

Consumer Key와 요청 시간(timestamp), 요청자료를 암호화 한 방식(HMAC-SHA1 등), 버전(1.0) 정보, 악의적인 정보 요청을 방지하기 위한 임의 문자열과 서명값을 만든 후 해당 서버로 요청.

서명값(oauth\_signature) 앞에서 설명한 각 변수값과 HTTP 요청 방식(POST 또는 GET)을 지정한 암호화 방식으로 암호화하여 만들어 낸다.

서비스 제공자에게 Request Token을 요청하면, 서비스 제공자는 ‘access\_token’과 ‘access\_token\_secret’값을 회신한다.

2) 사용자 인증 페이지 호출 및 수락

수신한 ‘access\_token’을 가지고 서비스 제공자가 지정한 인증페이지로 인증 수락을 요청한다. 인증 페이지는 각 서비스제공자가 제공한다.

3) Access Token 요청 및 발급

‘consumer\_key’와 ‘access\_token’ 및 기타 변수들을 이용하여 ‘oauth\_signature’ 를 생성한 후 요청하면 서비스 제공자는 ‘access\_token’과 ‘access\_token\_secret’ 및 부가 정보를 회신한다.

4) 해당 API 접근

수신한 ‘access\_token’을 이용하여 서비스제공자의 API를 사용한다.

**토큰을 이용한 파이선 코드**

import tweepy # 트위피 사용하기 위함

import os # 크롤링 결과를 저장하기 위함

import pandas as pd # 크롤링한 데이터를 DataFrame 유형으로 저장

consumer\_key = "MGRK5IsX8xwxhz0FYv5Llm5ps" #API 키를 받아오기 위한 키

consumer\_secret ="JRh3fHqPqEq6VWcyoKax6MG4nE21z0zatiDjEGnvmHm99cyrLA" #API 키를 받아오기 위한 비밀키

access\_token ="1103843008670121984-qw1ooMrZLzK10AcQkuixvvq0dizVfR" # 앱에 접근이 가능하도록 하기 위한 토큰

access\_token\_secret= "dqplsyeDz5n7kkYB8kW6kIkDW7lPkoFnL3r4vpCR0brdJ" # 앱에 접근이 가능하도록 하기 위한 비밀 토큰

auth = tweepy.OAuthHandler(consumer\_key, consumer\_secret) # 핸들러 생성 및 개인정보 인증 요청

auth.set\_access\_token(access\_token, access\_token\_secret) # 앱에 대한 액세스 요청

api = tweepy.API(auth, wait\_on\_rate\_limit=True) #트위터 API 생성

results = [] # 결과값들을 저장하기 임시적으로 저장하기 위함

for tweet in tweepy.Cursor(api.search, q='버닝썬',since='2019-03-27', until='2019-03-29').items(5): #Cursor를 사용하여 search api를 사용함. 검색어는 버닝썬 날짜는 2019-03-27 부터 2019-03-26 23:59:59 까지, 10개의 트윗을 가져온다.

results.append(tweet) # search로 찾은 결과를 results에 추가한다.

print(results) # results 출력

def process\_results(results): # 가져온 트윗들을 정리한다.

id\_list = [tweet.id for tweet in results] # results에 들어간 tweet 중 id를 id\_list로 저장

data\_set = pd.DataFrame(id\_list, columns=["id"]) #id\_list 값을 ‘id’ 이름의 columns 로 저장하는 데이터프레임을 만든다.

data\_set["text"] = [tweet.text for tweet in results] # results에 들어간 tweet 중 text를 “text” 이름의 columns 로 추가 저장

data\_set["created\_at"] = [tweet.created\_at for tweet in results] # results에 들어간 tweet 중 created\_at를 “text” 이름의 created\_at 로 추가 저장

data\_set["retweet\_count"] = [tweet.retweet\_count for tweet in results] # results에 들어간 tweet 중 retweet\_count를 “retweet\_count” 이름의 columns 로 추가 저장

data\_set["favorite\_count"] = [tweet.favorite\_count for tweet in results] # results에 들어간 tweet 중 text를 “text” 이름의 columns 로 저장

data\_set["source"] = [tweet.source for tweet in results] # results에 들어간 tweet 중 source를 “source” 이름의 columns 로 추가 저장

data\_set["user\_id"] = [tweet.author.id for tweet in results] # results에 들어간 tweet 중 author.id를 “user\_id” 이름의 columns 로 추가 저장

data\_set["user\_screen\_name"] = [tweet.author.screen\_name for tweet in results] # results에 들어간 tweet 중 author.screen\_name를 “user\_screen\_name” 이름의 columns 로 추가 저장

data\_set["user\_name"] = [tweet.author.name for tweet in results] # results에 들어간 tweet 중 text를 “text” 이름의 columns 로 추가 저장

data\_set["user\_created\_at"] = [tweet.author.created\_at for tweet in results]

data\_set["user\_description"] = [tweet.author.description for tweet in results]

data\_set["user\_followers\_count"] = [tweet.author.followers\_count for tweet in results]

data\_set["user\_friends\_count"] = [tweet.author.friends\_count for tweet in results]

data\_set["user\_location"] = [tweet.author.location for tweet in results]

return data\_set

data\_set = process\_results(results) #results 결과를 data\_set에 저장

wfile = open(os.getcwd()+"/twitter.txt", mode='w', encoding='utf8') #twitter.txt 라는 파일에 읽기전용으로 저장한다.

data = {}

i = 0

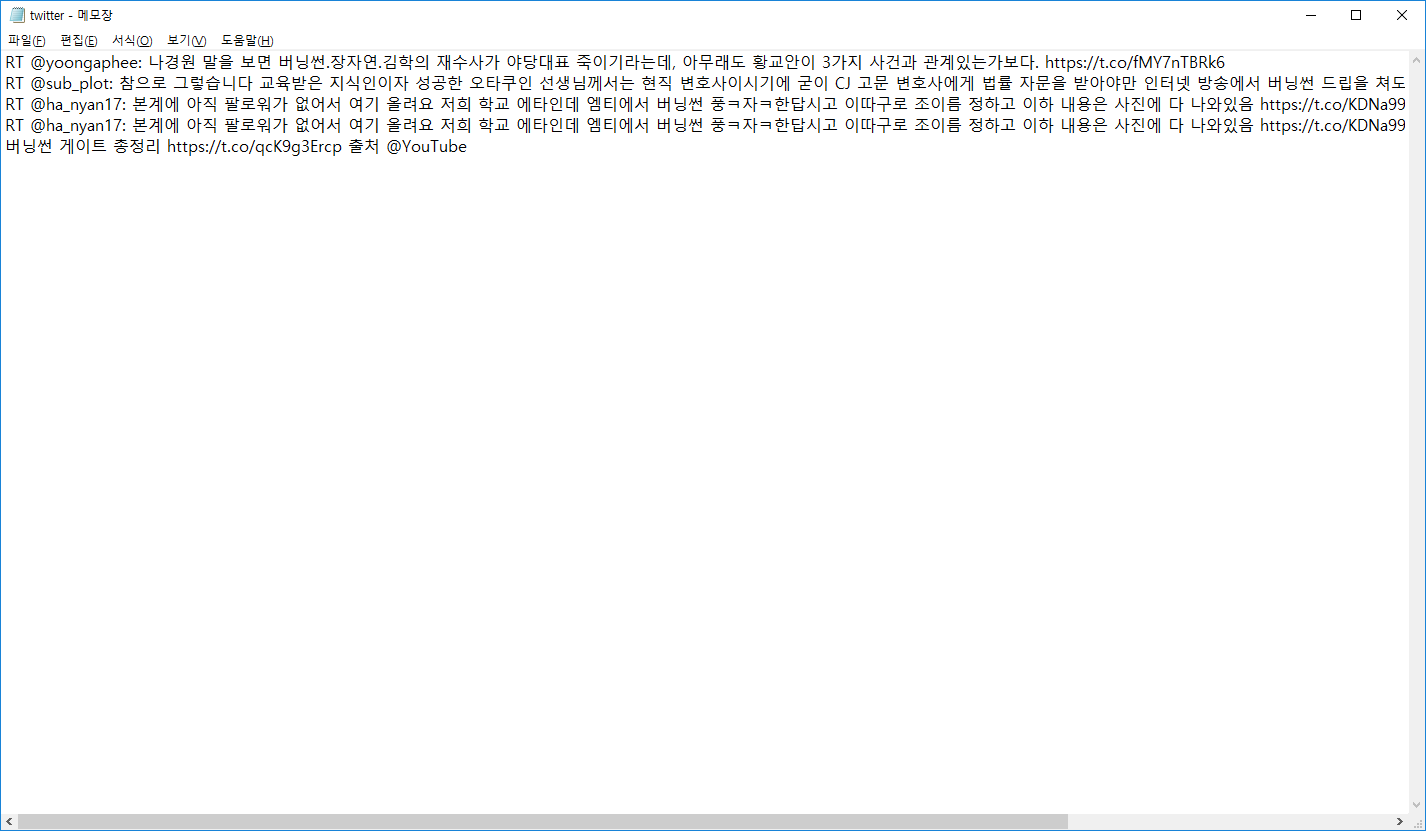
for tweet in results:

data['text'] = tweet.full\_text #추출해낸 트윗들은 제한없이 저장할 수 있다.

wfile.write(data['text']+'\n')

i += 1

**결과**



추출해낸 트윗들이 텍스트 파일로 저장되어 있음을 확인할 수 있다.

**Tweepy API**

Tweepy API를 사용하여 화면 출력 시 다음과 같은 형태에 맞추어서 출력된다.

[Status(\_api=<tweepy.api.API object at 0x0000020A98D66EB8>,

\_json={'created\_at': 'Mon Mar 25 05:00:05 +0000 2019', // 글 작성일

'id': 1110043656436613120, // 숫자로 표현된 ID

'id\_str': '1110043656436613120', string 형식으로 표현된 'id'

'full\_text': '파이선 테스트 입니다.', // 트윗 내용

'truncated': False, //140자로 내용 제한. 초과된 것들은 ...으로 표시

'display\_text\_range': [0, 12], // 트윗 시작과 끝을 식별하는 배열

'entities': {'hashtags': [], // 해시태그 내용

'symbols': [], //$ 같은 텍스트에 포함된 기호

'user\_mentions': [], // 언급된 다른 트위터 사용자 표시

'urls': []}, //트윗 텍스트에 포함된 URL 표시

'metadata': {'result\_type': 'recent', //결과 유형 표시. mixed-인기있는 결과와 실시간 결과 포함, recent-최근만 반환, popular-인기있는 결과만

'iso\_language\_code': 'ko'}, // 트위터가 지원하는 언어 코드 ko

'source': '<a href="http://twitter.com" rel="nofollow">Twitter Web Client</a>', // 트윗을 HTML 형식으로 표시

'in\_reply\_to\_status\_id': None, //트윗의 답글이면 답글의 숫자로 표현된 ID

'in\_reply\_to\_status\_id\_str': None, // 트윗의 답글이면 답글 ID의 string 형

'in\_reply\_to\_user\_id': None, //트윗 답글이면 그 트윗의 ID

'in\_reply\_to\_user\_id\_str': None, // 트윗의 답글이면 그 트윗 ID의 string 형

'in\_reply\_to\_screen\_name': None, // 트윗의 답글이면 그 트윗의 screen\_name

'user': {'id': 1103843008670121984, // 트윗 작성자의 숫자로 표현된 ID

'id\_str': '1103843008670121984', // 트윗 작성자 ID의 string 형

'name': '서재익', //작성자 이름

'screen\_name': 'JuBKbwLCwCy6Qy2', //주소 이름

'location': '', // 작성자가 트윗한 위치, 설정에서 위치 정보 포함가능

'description': '', // 작성자 소개

'url': None, // 설정한 url

'entities': {'description': {'urls': []}}, // URL에 대한 소개

'protected': False, // 비공개 여부

'followers\_count': 2, // 팔로워수

'friends\_count': 7, // 팔로우한 수

'listed\_count': 0, // 리스트(계정을 모아서 그룹으로 만듬) 갯수

'created\_at': 'Fri Mar 08 02:20:56 +0000 2019', // 계정 생성일

'favourites\_count': 1, // 좋아요 횟수

'utc\_offset': None, // 국제 사회가 사용하는 표준시간대. null로 셋팅된다.

'time\_zone': None, // 광고주들이 계정 관리자들에게 계약이나 청구를 하기위해 사용되는 시간대. null로 셋팅

'geo\_enabled': False, // 사용자의 위치기능 활성화 여부

'verified': False, // 유명인들의 계정 인증여부.

'statuses\_count': 8, // 사용자가 작성한 트윗과 리트윗 수

'lang': 'ko', //사용자가 사용하는 언어

'contributors\_enabled': False, // 참가자모드(사용자가 작성한 트윗을 다른 사람도 작성할 수 있도록함)가 적용된 계정의 여부

'is\_translator': False, // True이면 트위터의 번역을 했음 표시

'is\_translation\_enabled': False, // 번역의 사용 가능여부

'profile\_background\_color': 'F5F8FA', //사용자가 설정한 배경색을 16진수로

'profile\_background\_image\_url': None, // 사용자가 프로필에 업로드한 배경 이미지의 HTTP기반 URL

'profile\_background\_image\_url\_https': None, // 사용자가 프로필에 업로드한 배경 이미지의 HTTPS기반 URL

'profile\_background\_tile': False, //True이면 사용자의 'profile\_background\_image\_url'이 타일링(타일 단위로 나누어서 메모리 관리)

'profile\_image\_url': 'http://abs.twimg.com/sticky/default\_profile\_images/default\_profile\_normal.png', //프로필 이미지의 HTTP 기반 URL

'profile\_image\_url\_https': 'https://abs.twimg.com/sticky/default\_profile\_images/default\_profile\_normal.png', // 프로필 이미지의 HTTPS 기반 URL

'profile\_link\_color': '1DA1F2', // 트위터 UI에서 링크를 표시할 떄 사용하는 색깔

'profile\_sidebar\_border\_color': 'C0DEED', //사용자의 사이드바 테두리의 색깔

'profile\_sidebar\_fill\_color': 'DDEEF6', //사용자의 사이드바 배경 색

'profile\_text\_color': '333333', // 사용자만의 텍스트 색깔

'profile\_use\_background\_image': True, // True이면 업로드된 배경 이미지를 사용하려는 사용자들 표시

'has\_extended\_profile': False, // 내부에서만 사용되는 것.

'default\_profile': True, // True이면 사용자 프로필 테마나 배경을 변경하지 않았음을 표시

'default\_profile\_image': True, //True 이면 사용자가 기본 이미지를 사용

'following': False, //true라면 인증 사용자가 이 사용자를 팔로잉 하고 있음을 표시.

'follow\_request\_sent': False, //follow\_request\_sent 속성값 반환

'notifications': False, 인증된 사용자가 SMS를 통해 이 사용자의 트위트를 수신하도록 선택했는지 여부 확인

'translator\_type': 'none'}, //번역을 했다면 어떤 언어로 했는가

'geo': None, // 좌표를 (경도, 위도) 형태로 포맷

'coordinates': None, // (경도, 위도) 형식을 받아들임

'place': None, // 위치 기능을 활성화하면 트윗을 작성한 곳을 기준으로 주변 장소 목록 생성.

'contributors': None, // contributors 계정의 권한을 할당 받았는지 여부(공유계정으로써 메세지 트윗 팔로잉가능)

'is\_quote\_status': False, //인용된 트윗인지를 확인

'retweet\_count': 0, //리트윗 갯수

'favorite\_count': 0, //좋아요 갯수

'favorited': False, // 인증된 사용자가 좋아요를 눌렀는지

'retweeted': False, //인증된 사용자가 리트윗을 했는지

'lang': 'ko'}, //언어는 ko

created\_at=datetime.datetime(2019, 3, 25, 5, 0, 5), // 글 작성일

id=1110043656436613120, // 숫자로 표현된 ID

id\_str='1110043656436613120', // string 형식으로 표현된 ID

full\_text='파이선 테스트 입니다.', //트윗 글 내용

truncated=False, //140자로 내용 제한. 초과된 것들은 ...으로 표시

display\_text\_range=[0, 12], //트윗의 시작과 끝을 식별하는 배열

entities={'hashtags': [],

'symbols': [],

'user\_mentions': [],

'urls': []},

metadata={'result\_type': 'recent',

'iso\_language\_code': 'ko'},

source='Twitter Web Client',

source\_url='http://twitter.com',

in\_reply\_to\_status\_id=None,

in\_reply\_to\_status\_id\_str=None,

in\_reply\_to\_user\_id=None,

in\_reply\_to\_user\_id\_str=None,

in\_reply\_to\_screen\_name=None,

author=User(\_api=<tweepy.api.API object at 0x0000020A98D66EB8>, //인증된 사용자

\_json={'id': 1103843008670121984, // 숫자로 이루어진 ID

'id\_str': '1103843008670121984', // string 형식으로된 ID

'name': '서재익', // 사용자의 이름

'screen\_name': 'JuBKbwLCwCy6Qy2', // 사용자의 화면이름

'location': '',

'description': '',

'url': None,

'entities': {'description': {'urls': []}},

'protected': False,

'followers\_count': 2,

'friends\_count': 7,

'listed\_count': 0,

'created\_at': 'Fri Mar 08 02:20:56 +0000 2019',

'favourites\_count': 1,

'utc\_offset': None,

'time\_zone': None,

'geo\_enabled': False,

'verified': False,

'statuses\_count': 8,

'lang': 'ko',

'contributors\_enabled': False,

'is\_translator': False,

'is\_translation\_enabled': False

, 'profile\_background\_color': 'F5F8FA',

'profile\_background\_image\_url': None,

'profile\_background\_image\_url\_https': None

, 'profile\_background\_tile': False,

'profile\_image\_url': 'http://abs.twimg.com/sticky/default\_profile\_images/default\_profile\_normal.png',

'profile\_image\_url\_https': 'https://abs.twimg.com/sticky/default\_profile\_images/default\_profile\_normal.png',

'profile\_link\_color': '1DA1F2',

'profile\_sidebar\_border\_color': 'C0DEED',

'profile\_sidebar\_fill\_color': 'DDEEF6',

'profile\_text\_color': '333333',

'profile\_use\_background\_image': True,

'has\_extended\_profile': False,

'default\_profile': True,

'default\_profile\_image': True

, 'following': False,

'follow\_request\_sent': False,

'notifications': False,

'translator\_type': 'none'},

id=1103843008670121984,

id\_str='1103843008670121984',

name='서재익',

screen\_name='JuBKbwLCwCy6Qy2',

location='',

description='',

url=None,

entities={'description': {'urls': []}},

protected=False,

followers\_count=2,

friends\_count=7,

listed\_count=0,

created\_at=datetime.datetime(2019, 3, 8, 2, 20, 56),

favourites\_count=1,

utc\_offset=None,

time\_zone=None,

geo\_enabled=False,

verified=False,

statuses\_count=8,

lang='ko',

contributors\_enabled=False,

is\_translator=False,

is\_translation\_enabled=False,

profile\_background\_color='F5F8FA',

profile\_background\_image\_url=None,

profile\_background\_image\_url\_https=None,

profile\_background\_tile=False,

profile\_image\_url='http://abs.twimg.com/sticky/default\_profile\_images/default\_profile\_normal.png',

profile\_image\_url\_https='https://abs.twimg.com/sticky/default\_profile\_images/default\_profile\_normal.png',

profile\_link\_color='1DA1F2', profile\_sidebar\_border\_color='C0DEED', profile\_sidebar\_fill\_color='DDEEF6',

profile\_text\_color='333333', profile\_use\_background\_image=True, has\_extended\_profile=False, default\_profile=True,

default\_profile\_image=True,

following=False,

follow\_request\_sent=False,

notifications=False,

translator\_type='none'),

user=User(\_api=<tweepy.api.API object at 0x0000020A98D66EB8>,

\_json={'id': 1103843008670121984,

'id\_str': '1103843008670121984',

'name': '서재익',

'screen\_name': 'JuBKbwLCwCy6Qy2',

'location': '',

'description': '',

'url': None,

'entities': {'description': {'urls': []}},

'protected': False,

'followers\_count': 2,

'friends\_count': 7,

'listed\_count': 0,

'created\_at': 'Fri Mar 08 02:20:56 +0000 2019',

'favourites\_count': 1,

'utc\_offset': None, 'time\_zone': None,

'geo\_enabled': False,

'verified': False,

'statuses\_count': 8,

'lang': 'ko',

'contributors\_enabled': False,

'is\_translator': False,

'is\_translation\_enabled': False,

'profile\_background\_color': 'F5F8FA',

'profile\_background\_image\_url': None,

'profile\_background\_image\_url\_https': None,

'profile\_background\_tile': False,

'profile\_image\_url': 'http://abs.twimg.com/sticky/default\_profile\_images/default\_profile\_normal.png',

'profile\_image\_url\_https': 'https://abs.twimg.com/sticky/default\_profile\_images/default\_profile\_normal.png',

'profile\_link\_color': '1DA1F2',

'profile\_sidebar\_border\_color': 'C0DEED',

'profile\_sidebar\_fill\_color': 'DDEEF6',

'profile\_text\_color': '333333',

'profile\_use\_background\_image': True,

'has\_extended\_profile': False,

'default\_profile': True,

'default\_profile\_image': True,

'following': False,

'follow\_request\_sent': False,

'notifications': False,

'translator\_type': 'none'},

id=1103843008670121984,

id\_str='1103843008670121984',

name='서재익',

screen\_name='JuBKbwLCwCy6Qy2',

location='',

description='',

url=None,

entities={'description': {'urls': []}},

protected=False,

followers\_count=2,

friends\_count=7,

listed\_count=0,

created\_at=datetime.datetime(2019, 3, 8, 2, 20, 56),

favourites\_count=1,

utc\_offset=None, time\_zone=None,

geo\_enabled=False,

verified=False,

statuses\_count=8,

lang='ko',

contributors\_enabled=False,

is\_translator=False,

is\_translation\_enabled=False,

profile\_background\_color='F5F8FA',

profile\_background\_image\_url=None,

profile\_background\_image\_url\_https=None,

profile\_background\_tile=False,

profile\_image\_url='http://abs.twimg.com/sticky/default\_profile\_images/default\_profile\_normal.png',

profile\_image\_url\_https='https://abs.twimg.com/sticky/default\_profile\_images/default\_profile\_normal.png',

profile\_link\_color='1DA1F2',

profile\_sidebar\_border\_color='C0DEED',

profile\_sidebar\_fill\_color='DDEEF6',

profile\_text\_color='333333',

profile\_use\_background\_image=True, has\_extended\_profile=False,

default\_profile=True,

default\_profile\_image=True,

following=False,

follow\_request\_sent=False,

notifications=False,

translator\_type='none'),

geo=None,

coordinates=None,

place=None,

contributors=None,

is\_quote\_status=False,

retweet\_count=0,

favorite\_count=0,

favorited=False,

retweeted=False,

lang='ko')]

**Tweepy.api (트위피 기본 api 클래스)**

|  |  |
| --- | --- |
| auth\_handler | 사용할 인증 핸들러 |
| host | 일반 API 호스트 |
| search\_host | API 호스트 검색 |
| cache | 사용할 백엔드 캐시 |
| api\_root | 일반 API 경로 루트 |
| search\_root | API 경로 루트 검색 |
| retry\_count | 오류 발생 시, 재시도 횟수 |
| retry\_delay | 재시도 사이의 대기 시간(단위 : 초) |
| retry\_errors | 재시도 할 HTTP 상태 코드 |
| timeout | Twitter에서 응답을 기다리는 최대 시간 |
| parser | Twitter로부터의 응답을 해석하기 위해서 사용하는 객체 |
| compression | 요청에 GZIP 압축 사용 여부 |
| wait\_on\_rate\_limit | 요금 제한이 보충 될 때까지 자동으로 대기할지 여부 |
| wait\_on\_rate\_limit\_notify | Tweepy가 속도 제한이 보충되기를 기다리고있을 때 알림을 인쇄할지 여부 |
| proxy | HTTPS 프록시가 Twitter에 연결하는 데 사용할 전체 URL |

**Timeline Methods**

**API.home\_timeline( [since\_id ][ , max\_id ][ , count ][ , page ])**

인증 사용자 및 해당 사용자의 친구가 게시 한 리트 윗을 포함하여 가장 최근의 20 개의 상태를 반환.

|  |  |
| --- | --- |
| since\_id | 지정한 ID보다 더 최근 ID의 상태(트윗, 리트윗 등)를 반환 |
| max\_id | 지정한 ID보다 같거나 오래된 ID의 상태를 반환 |
| count | 검색할 상태 수를 지정함 |
| page | 검색할 결과 페이지 지정 |

**API.statuses\_lookup( id\_ [ , include\_entities ][ , trim\_user ][ , map\_ ])**

지정 ID의 트윗에 대한 객체 반환

|  |  |
| --- | --- |
| id | 검색할 트윗 ID 목록(최대 100) |
| **include\_entities** | 반환된 트윗에 entities(해시태그, url, 유저멘션, 캐시태그) 포함 유무 |
| **trim\_user** | 유저의 모든 정보 대신, 아이디만 제공하는지 |
| map | 표시 못하거나 값이 없는 트윗 포함 유무 |

**API.user\_timeline( [id / user\_id / screen\_name ][ , since\_id ][ , max\_id ][ , count ][ , page ])**

지정 사용자가 게시한 최근 20개 상태 반환

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |
| **user\_id** | 검색할 트윗의 숫자 ID |
| **screen\_name** | 사용자의 이름 |
| since\_id | 지정한 ID보다 더 최근 ID의 상태(트윗, 리트윗 등)를 반환 |
| max\_id | 지정한 ID보다 같거나 오래된 ID의 상태를 반환 |
| count | 검색할 상태 수를 지정함 |
| page | 검색할 결과 페이지 지정 |

**API.retweets\_of\_me( [since\_id ][ , max\_id ][ , count ][ , page ])**

다른 사용자가 리트윗한 인증 된 사용자의 최근 트윗 20 개 반환

|  |  |
| --- | --- |
| since\_id | 지정한 ID보다 더 최근 ID의 상태(트윗, 리트윗 등)를 반환 |
| max\_id | 지정한 ID보다 같거나 오래된 ID의 상태를 반환 |
| count | 검색할 상태 수를 지정함 |
| page | 검색할 결과 페이지 지정 |

**API.mentions\_timeline( [since\_id ][ , max\_id ][ , count ])**

리트 윗을 포함하여 가장 최근에 언급 된 20 개 항목을 반환

|  |  |
| --- | --- |
| since\_id | 지정한 ID보다 더 최근 ID의 상태(트윗, 리트윗 등)를 반환 |
| max\_id | 지정한 ID보다 같거나 오래된 ID의 상태를 반환 |
| count | 검색할 상태 수를 지정함 |

**Status methods**

**API.get\_status( id )**

지정 ID의 상태(트윗, 리트윗 등) 반환

|  |  |
| --- | --- |
| id | 검색할 트윗 숫자 ID |

**API.update\_status(status[, in\_reply\_to\_status\_id][, auto\_populate\_reply\_metadata][, lat][, long][, source][, place\_id])**

인증된 유저의 상태 업데이트.

|  |  |
| --- | --- |
| status | 상태 업데이트 |
| **in\_reply\_to\_status\_id** | 업데이트 하려는 아이디 |
| **auto\_populate\_reply\_metadata** | 상태 데이터에 멘션을 포함할지 |
| **lat** | 트윗이 참조하는 위치의 위도 |
| **long** | 트윗이 참조하는 위치의 경도 |
| **source** | Identi.ca에서만 지원. 업데이트 소스 |
| **place\_id** | 트위터 아이디의 위치 |

**API.update\_with\_media(filename[, status][, in\_reply\_to\_status\_id][, auto\_populate\_reply\_metadata][, lat][, long][, source][, place\_id][, file])**

API.media\_upload() 대신 사용(사용하지 않음). 인증된 사용자의 상태 업데이트.

|  |  |
| --- | --- |
| filename | 업로드하려는 이미지의 이름 |
| status | 상태 업데이트 |
| **in\_reply\_to\_status\_id** | 업데이트 하려는 아이디 |
| **auto\_populate\_reply\_metadata** | 상태 데이터에 멘션을 포함할지 |
| **lat** | 트윗이 참조하는 위치의 위도 |
| **long** | 트윗이 참조하는 위치의 경도 |
| **source** | Identi.ca에서만 지원. 업데이트 소스 |
| **place\_id** | 트위터 아이디의 위치 |
| **file** | MIME 유형이나 POST 데이터 사용시 필요함 |

**API.destroy\_status(id)**

지정된 ID의 상태 제거

|  |  |
| --- | --- |
| id | 검색할 트윗 숫자 ID |

**API.retweet(id)**

지정된 ID로 리트윗할 때 사용

|  |  |
| --- | --- |
| id | 검색할 트윗 숫자 ID |

**API.retweets(id[, count])**

트윗의 첫번쨰 리트윗부터 100개까지 반환.

|  |  |
| --- | --- |
| id | 검색할 트윗 숫자 ID |
| count | 검색할 상태 수를 지정함 |

**User methods**

**API.get\_user(id/user\_id/screen\_name)**

유저에 대한 정보 반환.

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |
| **user\_id** | 검색할 트윗의 숫자 ID |
| **screen\_name** | 사용자의 이름 |

**API.me()**

인증받은 유저의 정보 반환

**API.followers([id/screen\_name/user\_id][, cursor])**

유저의 팔로워들을 반환

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |
| **user\_id** | 검색할 트윗의 숫자 ID |
| **screen\_name** | 사용자의 이름 |
| **cursor** | -1 값을 입력하여 결과를 페이지로 나눔. |

**API.search\_users(q[, count][, page])**

사용자를 찾을 때

|  |  |
| --- | --- |
| q | 찾으려는 사용자 |
| count | 검색할 상태 수를 지정. 최대 20 |
| page | 검색할 결과 페이지 지정. |

**Direct Message Methods**

**API.direct\_messages([since\_id][, max\_id][, count][, page][, full\_text])**

인증된 사용자에게 보내진 메시지 반환

|  |  |
| --- | --- |
| since\_id | 지정한 ID보다 더 최근 ID의 상태(트윗, 리트윗 등)를 반환 |
| max\_id | 지정한 ID보다 같거나 오래된 ID의 상태를 반환 |
| count | 검색할 상태 수를 지정함 |
| page | 검색할 결과 페이지 지정 |
| full\_text | 메시지의 전체 텍스트 반환 여부를 나타냄. 기본값인 False일 경우 반환된 메시지 텍스트는 140자로 잘림. |

**API.get\_direct\_message([id][, full\_text])**

메시지 반환

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |
| full\_text | 메시지의 전체 텍스트 반환 여부를 나타냄. 기본값인 False일 경우 반환된 메시지 텍스트는 140자로 잘림. |

**API.sent\_direct\_messages([since\_id][, max\_id][, count][, page][, full\_text])**

인증된 사용자가 보낸 메시지 반환

|  |  |
| --- | --- |
| since\_id | 지정한 ID보다 더 최근 ID의 상태(트윗, 리트윗 등)를 반환 |
| max\_id | 지정한 ID보다 같거나 오래된 ID의 상태를 반환 |
| count | 검색할 상태 수를 지정함 |
| page | 검색할 결과 페이지 지정 |
| full\_text | 메시지의 전체 텍스트 반환 여부를 나타냄. 기본값인 False일 경우 반환된 메시지 텍스트는 140자로 잘림. |

**API.send\_direct\_message(user/screen\_name/user\_id, text)**

인증된 사용자로부터 지정된 사용자에게 메시지를 보냄

|  |  |
| --- | --- |
| user | 메시지를 받는 유저 ID |
| **user\_id** | 검색할 트윗의 숫자 ID |
| **screen\_name** | 사용자의 이름 |

**API.destroy\_direct\_message(id)**

메시지 삭제

|  |  |
| --- | --- |
| id | 삭제할 메시지 아이디 |

**Friendship Methods**

**API.create\_friendship(id/screen\_name/user\_id[, follow])**

지정된 사용자를 팔로우.

|  |  |
| --- | --- |
| id | 메시지를 받는 유저 ID |
| **user\_id** | 검색할 트윗의 숫자 ID |
| **screen\_name** | 사용자의 이름 |
| **follow** | 친구 추가와 대상 사용자에게 알림 설정 |

**API.destroy\_friendship(id/screen\_name/user\_id)**

지정된 사용자 언팔로우

|  |  |
| --- | --- |
| id | 메시지를 받는 유저 ID |
| **user\_id** | 검색할 트윗의 숫자 ID |
| **screen\_name** | 사용자의 이름 |

**API.show\_friendship(source\_id/source\_screen\_name, target\_id/target\_screen\_name)**

두 유저 사이의 관계 정보를 상세히 반환.

|  |  |
| --- | --- |
| source\_id | 대상 사용자의 숫자로 이루어진 ID |
| Source\_screen\_name | 대상 사용자의 화면상의 이름 |
| **Target\_id** | 다른 대상 사용자의 숫자로 이루어진 ID |
| **Target\_screen\_name** | 다른 대상 사용자의 화면상의 이름 |

**API.friends\_ids(id/screen\_name/user\_id[, cursor])**

지정된 사용자가 팔로우한 ID 반환

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |
| **user\_id** | 검색할 트윗의 숫자 ID |
| **screen\_name** | 사용자의 이름 |
| **cursor** | -1 값을 입력하여 결과를 페이지로 나눈다. |

**API.followers\_ids(id/screen\_name/user\_id)**

지정된 사용자를 팔로우한 ID 반환

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |
| **screen\_name** | 사용자의 이름 |
| **user\_id** | 검색할 트윗의 숫자 ID |
| **cursor** | -1 값을 입력하여 결과를 페이지로 나눈다. |

**Account Methods**

**API.verify\_credentials()**

사용증명이 올바른지 확인

**API.rate\_limit\_status()**

 API 제한에 도달하기 전에 요청 사용자가 사용할 수 있는 나머지 API 요청 수를 반환

**API.update\_profile\_colors([profile\_background\_color][, profile\_text\_color][, profile\_link\_color][, profile\_sidebar\_fill\_color][, profile\_sidebar\_border\_color])**

사용자의 프로필 수정. 16진수로 설정

|  |  |
| --- | --- |
| **profile\_background\_color** | 프로필 배경색 |
| **profile\_text\_color** | 프로필 글자색 |
| **profile\_link\_color** | 프로필 링크 표시색 |
| **profile\_sidebar\_fill\_color** | 프로필 사이드바 배경 표시색 |
| **profile\_sidebar\_border\_color** | 프로필 사이드바 테두리 표시색 |

**API.update\_profile\_image(filename)**

인증한 사용자의 프로필 이미지. GIF, JPG, PNG만 가능.

|  |  |
| --- | --- |
| **filename** | 업로드한 이미지 파일 경로 |

**API.update\_profile\_background\_image(filename)**

인증한 사용자의 배경 이미지. GIF, JPG, PNG만 가능.

|  |  |
| --- | --- |
| **filename** | 업로드한 이미지 파일 경로 |

**API.update\_profile([name][, url][, location][, description])**

설정 페이지에서 설정할 수 있는 값 설정.

|  |  |
| --- | --- |
| **name** | 계정 이름 |
| **url** | 계정 주소 |
| **Location** | 사용자의 위치 |
| **description** | 계정 설명하는 문자열 |

**Favorite Methods**

**API.favorites([id][, page])**

지정된 ID의 선호 상태를 반환

|  |  |
| --- | --- |
| id | 검색할 트윗 숫자 ID |
| page | 검색할 결과 페이지 지정 |

**API.create\_favorite(id)**

지정된 ID를 인증 사용자가 즐겨찾기 추가

|  |  |
| --- | --- |
| Id | 검색할 트윗 숫자 ID |

**API.destroy\_favorite(id)**

지정된 ID를 인증 사용자가 즐겨찾기 해제

|  |  |
| --- | --- |
| id | 검색할 트윗 숫자 ID |

**Block Methods**

**API.create\_block(id/screen\_name/user\_id)**

지정된 ID를 인증 사용자가 차단

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |
| **screen\_name** | 사용자의 이름 |
| **user\_id** | 검색할 트윗의 숫자 ID |

**API.destroy\_block(id/screen\_name/user\_id)**

지정된 ID를 인증 사용자가 차단 해제

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |
| **screen\_name** | 사용자의 이름 |
| **user\_id** | 검색할 트윗의 숫자 ID |

**API.blocks([page])**

인증 사용자가 차단한 사용자 반환

|  |  |
| --- | --- |
| page | 검색할 결과 페이지 지정 |

**API.blocks\_ids()**

인증 사용자가 차단한 숫자로 이루어진 사용자 아이디를 반환

**Spam Reporting Methods**

**API.report\_spam([id/user\_id/screen\_name])**

지정된 ID는 인증된 사용자가 스팸 메일로 신고

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |
| **screen\_name** | 사용자의 이름 |
| **user\_id** | 검색할 트윗의 숫자 ID |

**Saved Searches Methods**

**API.saved\_searches()**

인증된 사용자가 저장한 검색 쿼리

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |

**API.get\_saved\_search(id)**

인증된 사용자가 지정된 ID로 검색한 데이터를 찾는다.

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |

**API.create\_saved\_search(query)**

인증된 사용자에 대한 저장된 검색어 생성

|  |  |
| --- | --- |
| query | 사용자가 저장할 검색 단어 |

**API.destroy\_saved\_search(id)**

인증된 사용자에 대한 저장된 검색 삭제

|  |  |
| --- | --- |
| id | 검색할 트윗 ID |

**Help Methods**

**API.search(q[, lang][, locale][, rpp][, page][, since\_id][, geocode][, show\_user])**

|  |  |
| --- | --- |
| q | 찾으려는 단어 |
| lang | 트윗을 ISO 639-1 코드에 지정된 언어로 제한 |
| locale | 보내려는 조회의 언어를 지정하십시오(현재 ja만 유효함 |
| rpp | 페이지당 반환하는 트윗 개수. 최대 100개 |
| page | 반환할 페이지 번호. 최대 1500개 |
| since\_id | 지정한 ID보다 더 최근 ID의 상태(트윗, 리트윗 등)를 반환 |
| geocode | 지정된 위도, 경도를 지정된 반경내에 트윗 반환. (매개변수는 “latitide,longitude,radius” 로 지정 |
| Show\_user | true일 경우 트윗의 시작 부분에 "<user>"를 추가. Atom’s author field를 표시하지 않는 이용자들에게 유용. 디폴트는 false. |

**List Methods**

**API.create\_list(name[, mode][, description])**

인증된 사용자의 새로운 리스트를 만든다. 계정은 20개로 제한

|  |  |
| --- | --- |
| name | 만드는 리스트 이름 |
| mode | 리스트의 공개 여부. Public, private로 구분. 기본적으로 공개 상태. |
| description | 만드는 리스트에 대한 설명 |

**API.destroy\_list(slug)**

인증된 사용자에 의한 목록 삭제

|  |  |
| --- | --- |
| slug | 리스트의 카테고리의 이름이나, 숫자로 이뤄진 ID |

**API.update\_list(slug[, name][, mode][, description])**

리스트 목록을 업데이트한다.

|  |  |
| --- | --- |
| slug | 리스트의 카테고리의 이름이나, 숫자로 이뤄진 ID |
| name | 만드는 리스트 이름 |
| mode | 리스트의 공개 여부. Public, private로 구분. 기본적으로 공개 상태. |
| description | 만드는 리스트에 대한 설명 |

**API.lists([cursor])**

지정된 사용자의 목록을 나열. 목록에 있는 사용자와 인증된 사용자가 같으면 개인 목록도 포함.

|  |  |
| --- | --- |
| **cursor** | -1 값을 입력하여 결과를 페이지로 나눔. |

**API.lists\_memberships([cursor])**

지정된 사용자가 추가된 목록 나열.

|  |  |
| --- | --- |
| **cursor** | -1 값을 입력하여 결과를 페이지로 나눔. |

**API.lists\_subscriptions([cursor])**

지정된 사용자를 팔로우하는 리스트 나열

|  |  |
| --- | --- |
| **cursor** | -1 값을 입력하여 결과를 페이지로 나눔. |

**API.list\_timeline(owner, slug[, since\_id][, max\_id][, count][, page])**

지정된 리스트 구성원들의 트윗들을 보여준다.

|  |  |
| --- | --- |
| owner | 리스트 주인의 화면상 이름 |
| slug | 리스트의 카테고리의 이름이나, 숫자로 이뤄진 ID |
| since\_id | 지정한 ID보다 더 최근 ID의 상태(트윗, 리트윗 등)를 반환 |
| max\_id | 지정한 ID보다 같거나 오래된 ID의 상태를 반환 |
| count | 검색할 상태 수를 지정함 |
| page | 검색할 결과 페이지 지정 |

**API.add\_list\_member(slug, id)**

인증된 사용자가 리스트의 구성원을 추가한다. 최대 500명까지 가능

|  |  |
| --- | --- |
| slug | 리스트의 카테고리의 이름이나, 숫자로 이뤄진 ID |
| id | 추가할 구성원의 사용자 ID |

**API.remove\_list\_member(slug, id)**

인증된 사용자가 리스트에서 구성원을 지운다.

|  |  |
| --- | --- |
| slug | 리스트의 카테고리의 이름이나, 숫자로 이뤄진 ID |
| id | 추가할 구성원의 사용자 ID |

**API.list\_members(owner, slug, cursor)**

리스트의 구성원을 보여준다.

|  |  |
| --- | --- |
| owner | 리스트 주인의 화면상 이름 |
| slug | 리스트의 카테고리의 이름이나, 숫자로 이뤄진 ID |
| **cursor** | -1 값을 입력하여 결과를 페이지로 나눔. |

**API.is\_list\_member(owner, slug, id)**

사용자가 리스트 구성원인지 확인한다.

|  |  |
| --- | --- |
| owner | 리스트 주인의 화면상 이름 |
| slug | 리스트의 카테고리의 이름이나, 숫자로 이뤄진 ID |
| id | 확인할 사용자 ID |

**API.subscribe\_list(owner, slug)**

인증된 사용자가 리스트를 팔로우 하도록 만든다.

|  |  |
| --- | --- |
| owner | 리스트 주인의 화면상 이름 |
| slug | 리스트의 카테고리의 이름이나, 숫자로 이뤄진 ID |

**API.unsubscribe\_list(owner, slug)**

인증된 사용자가 리스트를 팔로우 취소 하도록 만든다.

|  |  |
| --- | --- |
| owner | 리스트 주인의 화면상 이름 |
| slug | 리스트의 카테고리의 이름이나, 숫자로 이뤄진 ID |

**API.list\_subscribers(owner, slug[, cursor])**

리스트의 팔로워들을 반환한다.

|  |  |
| --- | --- |
| owner | 리스트 주인의 화면상 이름 |
| slug | 리스트의 카테고리의 이름이나, 숫자로 이뤄진 ID |
| **cursor** | -1 값을 입력하여 결과를 페이지로 나눔. |

**API.is\_subscribed\_list(owner, slug, id)**

지정된 사용자가 리스트의 팔로워들인지 확인한다.

|  |  |
| --- | --- |
| owner | 리스트 주인의 화면상 이름 |
| slug | 리스트의 카테고리의 이름이나, 숫자로 이뤄진 ID |
| id | 확인할 사용자 ID |

**Trends Methods**

**API.trends\_available()**

트위터에 대한 추세 항목 정보가 있는 위치를 반환한다. 응답은 해당 지역의 WOEID와 해당 위치가 속한 국가 등 인간이 읽을 수 있는 다른 정보를 인코딩하는 "위치"의 배열이다.

WOEID란

-(Where On Earth Identifier의 약자로 야후에서 지구를 세분화하기 위해 32비트로 할당한 지역코드)

**API.trends\_place(id[, exclude])**

WOEID에 대한 상위 10개의 트렌드 주제로 반환.

|  |  |
| --- | --- |
| Id | WOEID의 위치 정보를 반환. 1을 주면 WOEID 사용 가능. |
| exclude | 해시태그와 동일하게 설정하면 트렌드 리스트에서 모든 해시태그가 제거. |

**API.trends\_closest(lat, long)**

지정한 위치에 가장 가까운 트렌드 주제를 가진 위치를 반환.

|  |  |
| --- | --- |
| Lat | long 변수와 같이 쓰면, 가장 멀리있거나 좌표 쌍에 가장 가까운 거리에 따라 정렬. |
| long | Lat 변수와 같이 쓰면, 가장 멀리 있거나 가장 가까운 거리에 따라 정렬. |

**Geo Methods**

**API.reverse\_geocode([lat][, long][, accuracy][, granularity][, max\_results])**

위도 경도를 지정하면, 위치 이름을 statue methods의 ‘update\_status()’ 로 표시하기 위해 가장 가까운 장소를 찾는다.

|  |  |
| --- | --- |
| **lat** | 트윗이 참조하는 위치의 위도 |
| **long** | 트윗이 참조하는 위치의 경도 |
| **accuracy** | 반지름 길이와 검색할 지역을 지정 |
| **granularity** | 기본적으로 ‘이웃’으로 가정되며, 도시가 될 수도 있다. |
| **Max\_results** | 반환할 최대 결과 수에 대한 근접한 위치 수.. |

**API.geo\_id(id)**

위치의 ID나 자세한 부분을 제공한다..

|  |  |
| --- | --- |
| **id** | 트위터 아이디의 위치 |

**Utility methods**

**API.configuration()**

사용자 이름, 최대 사진 해상도 및 t.co 단축 URL 길이가 아닌 slug 반환

**Media methods**

**API.media\_upload()**

트위터에 이미지를 업로드하고 ‘media\_id’(업로드한 이진 파일 컨턴츠의 ID)

|  |  |
| --- | --- |
| **media** | 반환할 최대 결과 수에 대한 근접한 위치 수.. |
| **Media\_data** | 업로드 중인 ‘base64-encoded’ 파일. Media와 같이 사용 불가. |
| **Addition\_owner** | Media\_id를 사용할 수 있는 추가 사용자(쉼표로 구분)로 설정, |

**Tweepy.error (예외 처리)**

**Exception TweepError**

기본적으로 사용하는 응답 오류에 의한 예외 처리.

**Exception RateLimitError**

트위터 속도 제한으로 인해 API 메소드가 실패 할 때 처리.