Aide mémoire API Tweepy - Version 2

Documentation

Ce document présente un résumé de quelques fonctionnalités utiles de l'API Tweepy. La documentation complète est disponible à cette adresse : `https://docs.tweepy.org/en/stable/client.html`

1 Obtenir des identifiants pour l'API Twitter

Pour travailler avec l'API Twitter, vous devrez posséder un compte Twitter, et obtenir des clés d'API.
Le document `https://raw.githubusercontent.com/alemaitr/python_opendata_l2/master/DocTweepy/compteDeveloppeur.pdf` explique comment procéder.

2 Import

Le module tweepy doit être importé en début du code source python :

```
import tweepy
```

3 Authentification

Une fois ces identifiants créés, vous pouvez créer un client d'API tweepy avec les commandes suivantes, en indiquant vos propres identifiants :

```
client = tweepy.Client(consumer_key="xxx", consumer_secret="xxx", access_token="xxx",
    access_token_secret="xxx")
```

Le client vous donne accès à diverses fonctionnalités, dont quelques-unes exposées ci-dessous.

4 Gérer ses tweets

4.1 Poster un tweet

L'identifiant du tweet créé est renvoyé dans la réponse

```
response = client.create_tweet(text="Texte à tweeter")
print(response.data["id"])
```

4.2 Supprimer un tweet

Supprimer un tweet nécessite d'en connaitre son identifiant (entier).

```
client.delete_tweet(tweet_id)
```

4.3 Liker un tweet

Liker un tweet nécessite d'en connaitre son identifiant.

```
client.like(tweet_id)
```

Il est aussi possible de ne plus aimer un tweet :

```
client.unlike(tweet_id)
```

5 Accéder aux utilisateurs

5.1 Obtenir des informations utilisateurs

Il est possible d'obtenir des informations utilisateurs à partir d'un nom ou d'un identifiant.
A partir d'un username :

```
user = client.get_user(username="metropolerennes",user_auth=True)
print(f"Utilisateur : {user.data.name}, identifiant : {user.data.id}")
```

A partir d'un identifiant :

```
user = client.get_user(id="16824660",user_auth=True)
print(f"Utilisateur : {user.data.name}, identifiant : {user.data.id}")
```

5.2 Champs d'un utilisateur

Par défaut, les champs d'un User ne sont composés que d'un identifiant, et d'un nom. Pour avoir des objets plus complets, il est nécessaire de le préciser lors de la requête, en remplissant une liste de user_fields.

Par exemple, dans la requête ci-dessus, on complète en demandant également le nom d'utilisateur et le lieu :

```
user = client.get_user(id="16824660",user_auth=True,user_fields=["username","location"])
print(f"Utilisateur : {user.data.name}, username :{user.data.username}, identifiant : {user.
    data.id}, lieu : {user.data.location}")
```

Les user_fields disponibles sont les suivants : created_at, description, entities, id, location, name, pinned_tweet_id, profile_image_url, protected, public_metrics, url, username, verified, withheld

5.3 Liste des followers

Le client permet d'accéder à la liste des followers d'un utilisateur, à partir de son identifiant. Comme ci-dessus, on pourra enrichir les champs renvoyés en précisant un user_fields.

```
response = client.get_users_followers(
    id="16824660", user_fields=["profile_image_url"],user_auth=True)
for user in response.data:
    print(user.username, user.profile_image_url)
```

Par défaut, la fonction renvoie les 100 premiers followers.

5.4 Liste des utilisateurs ayant aimé un tweet

Cette fonctionnalité nécessite de connaître l'id du tweet. Comme ci-dessus, on pourra enrichir les champs renvoyés en précisant un user_fields.

```
tweet_id = 1595360922314608641
response = client.get_liking_users(tweet_id,user_auth=True)
for user in response.data:
    print(user.name)
```

5.5 Liste des utilisateurs ayant retweeté un tweet

Cette fonctionnalité nécessite de connaître l'id du tweet. Comme ci-dessus, on pourra enrichir les champs renvoyés en précisant un user_fields.

```
tweet_id = 1595360922314608641
response = client.get_retweeters(tweet_id,user_auth=True)
for user in response.data:
    print(user.name)
```

6 Consulter des tweets

6.1 Consulter des tweets récents

Exemple : consultation des tweets récents citant "Rennes". La réponse renvoyée contient une liste de tweets (dans data). Il est possible d'afficher les identifiants de chaque tweet. Par défaut, 10 résultats sont renvoyés.

```
response = client.search_recent_tweets("Rennes",user_auth=True)
for tweet in tweets.data:
    print(tweet.id)
    print(tweet.text)
```

La requête peut être plus sophistiquée qu'un simple mot. Par exemple, ci-dessous, on cherche les 10 derniers tweets, postés par EmmanuelMacron, en supprimant les retweets.

```
myquery = 'from:EmmanuelMacron -is:retweet'
tweets = client.search_recent_tweets(query=myquery, max_results=10,user_auth=True)
```

6.2 Champs des tweets

Par défaut, les champs d'un tweet ne sont composés que d'un identifiant, et d'un texte. Pour avoir des objets plus complets, il est nécessaire de le préciser lors de la requête.

Par exemple, on complète le code précédent pour demander en plus la date de création, la langue.

```
response = client.search_recent_tweets("Rennes",tweet_fields=["created_at","lang"],user_auth
    =True)
for tweet in tweets.data:
    print(f"{tweet.id} (le{tweet.created_at}, en {tweet.lang}) : {tweet.text}")
```

Les tweet_fields disponibles sont les suivants : attachments, author_id, context_annotations, conversation_id, created_at, edit_controls, edit_history_tweet_ids, entities, geo, id, in_reply_to_user_id, lang, non_public_metrics, organic_metrics, possibly_sensitive, promoted_metrics, public_metrics, referenced_tweets, reply_settings, source, text, withheld

Le champ `geo` contient, lorsque cela a été précisé, les coordonnées géographiques du tweet. Il peut s'agir d'un simple emplacement (place_id), parfois complété d'une structure contenant des coordonnées GPS du type :

```
{'place_id': '5a110d312052166f',
'coordinates': {'type': 'Point', 'coordinates': [-122.407437, 37.787994]}}
```

Le champ `public_metrics` contient les statistiques comme le nombre de retweet ou le nombre de likes, sous la forme :

```
{'retweet_count': 1, 'reply_count': 0, 'like_count': 0, 'quote_count': 0}
```

6.3 Consulter la liste des tweets d'un utilisateur

Cette fonctionnalité nécessite de connaître l'identifiant d'un utilisateur.

```
response = client.get_users_tweets(user_id,user_auth=True)
for tweet in response.data:
    print(tweet.id)
    print(tweet.text)
```

Voici quelques paramètres supplémentaires possibles :
Comme ci-dessus, il est possible d'enrichir les données des tweets en précisant des `tweet_fields`.
Le paramètre `max_results` permet de préciser le nombre de résultat à renvoyer, par défaut 10.
Il est également possible de préciser les dates de début et fin de la requête, avec les paramètres `start_time` et `end_time` (au format datetime).

6.4 Consulter les tweets mentionnant un utilisateur

Cette fonctionnalité nécessite de connaître l'identifiant d'un utilisateur.

```
user_id = 2244994945
response = client.get_users_mentions(user_id)
for tweet in response.data:
    print(tweet.id)
    print(tweet.text)
```

6.5 Consulter des tweets à partir de leurs ids

```
tweet_ids = [1460323737035677698, 1293593516040269825, 1293595870563381249]
response = client.get_tweets(tweet_ids, tweet_fields=["created_at"],user_auth=True)
for tweet in response.data:
    print(tweet.text, tweet.created_at)
```