Aide-mémoire pour les modules `tweepy` et `googlemaps`

Romain Tavenard

Dans la suite, vous trouverez des éléments pour vous aider à utiliser les modules Python `tweepy` (version 3.5.0) et `googlemaps` (v. 1.0.2). Ce document n'est absolument pas exhaustif, mais il a vocation à centraliser des éléments utiles en pratique parfois insuffisamment documentés.

1 Avant-propos : les clés d'API

Pour utiliser ces modules, la première chose à faire sera de préciser votre clé d'API. Attention, il ne faut jamais écrire dans vos scripts Python ces clés "en dur", mais plutôt les lire dans un fichier annexe. Dans le cadre du projet `Cluedo`, par exemple, cela vous permettra de rendre votre code Python sans avoir à divulguer au correcteur vos clés d'API (qui doivent rester secrètes, au même titre qu'un mot de passe). Dans les exemples qui suivent, on supposera donc que ces clés d'API sont stockées dans des variables (dont le contenu a été défini en lisant un fichier de clés, typiquement un fichier JSON).

2 Le module `googlemaps`

Ce module, dont la documentation est accessible à l'adresse http://py-googlemaps.sourceforge.net permet d'accéder à plusieurs API Google Maps. Les principales fonctionnalités qu'il offre sont le *geocoding* (récupérer une adresse postale à partir d'un couple longitude/latitude et *vice versa*) et le calcul d'itinéraire.

2.1 Obtenir une clé d'API Google Maps

Pour obtenir une clé d'API Google Maps, vous devrez avoir un compte GMail puis, étant connecté à ce compte, accéder à la page de gestion de vos clés d'API. Sur cette page, dans l'onglet "Identifiants", vous pourrez créer une nouvelle clé d'API.

Une fois la clé créée, vous pourrez vous identifier dans le module `googlemaps` avec les commandes suivantes :

```
api = googlemaps.Client(api_key)
```

2.2 Exemple d'utilisation

La documentation du module `googlemaps` (dont l'adresse est donnée plus haut) fournit des exemples assez explicites et simples. Par exemple, pour ce qui est du calcul d'itinéraire, l'exemple suivant est fourni :

```
directions = api.directions(address, destination)
```

où `address` est l'origine du trajet et `destination` le point d'arrivée.

Pour mieux comprendre les paramètres de cette fonction `directions`, vous pouvez descendre un peu plus bas dans l'aide.

Sachez que les paramètres `address` et `destination` peuvent être soit des chaînes de caractères (ex : `"Rennes"`), soit des dictionnaires fournissant la longitude et la latitude du point à considérer (ex : `{"lng": -1.764416, "lat": 48.137123}`). De plus, pour obtenir plus d'informations sur les paramètres facultatifs de cette fonction, vous allez devoir vous rendre sur la page d'aide Google de l'API HTTP (tous les paramètres qui peuvent être passés à l'API HTTP peuvent aussi l'être à la fonction `directions` du module `googlemaps`).

3 Tweepy

3.1 Obtenir des identifiants pour l'API Twitter

Pour obtenir travailler avec l'API Twitter, vous devrez posséder un compte Twitter, puis, étant connecté à ce compte, visiter la page https://apps.twitter.com et y créer une "Application". Une fois cette application créée, vous devrez, dans l'onglet "Keys and Access tokens", créer :

— une clé d'API (aussi appelée "Consumer Key") et sa clé secrète ;
— un "Access Token" et son pendant secret.

Une fois ces identifiants créés, vous pourrez vous identifier dans `tweepy` avec les commandes suivantes :

```
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(key, secret)
api = tweepy.API(auth)
```

3.2 Exemple d'utilisation

Les principales actions que vous pourrez effectuer depuis `tweepy` seront :

- Obtenir la `timeline` (liste de tweets) pour un utilisateur à partir de son identifiant [lien] ;
  - retourne une liste d'objets de type `Status` (voir plus bas)
- Poster un tweet [lien] ;
  - retourne un objet `Status` représentant le tweet envoyé
- Supprimer un tweet à partir de son identifiant [lien]

Un objet de type `Status` représente un tweet et il possède un nombre important d'attributs (voir ici pour une liste exhaustive).

Par exemple, pour accéder à l'attribut `created_at` de la variable `s` de type `Status`, on écrira :

```
s.created_at
```