API web d'accès aux données

Planche de TD pour un cours dispensé à l'université de Rennes 2

1 Travail à préparer chez vous avant la séance

1.1 Compte GraphHopper

— Créez-vous un compte GraphHopper en cliquant ici (vous utiliserez l'URL `http://www.univ-rennes2.fr` pour le champ "site web")
— Créez-vous une clé d'API en vous rendant, connecté avec votre compte, à l'adresse https://graphhopper.com/dashboard/#/api-keys
— Écrivez cette clé d'API dans un fichier JSON nommé `data/credentials.json` au format suivant :

```
{
    "GraphHopper": "..."
}
```

1.2 Installation du module `graphh`

Par défaut, le module `graphh` n'est pas installé sur votre ordinateur. Pour pouvoir l'utiliser, il va donc falloir commencer par l'installer. Pour cela, vous devrez, dans PyCharm, choisir "Fichier -> Préférences", puis dans la barre de gauche "Project" puis "Project Interpreter". Vous avez alors une liste des packages qui s'affiche et, en cliquant sur le "+" en bas à gauche, vous allez pouvoir installer `graphh`.

2 Énoncé

Le fichier `mini-rando_gps.json` fournit des séries de positions GPS correspondant à des traces GPS de sorties randonnée de M. Toulemonde. On cherchera dans ce TD à écrire un programme calculant les dénivelés cumulés positif et négatif de chacune de ces randonnées. Pour cela, vous utiliserez l'API `GraphHopper` *via* le module Python `graphh`.

2.1 Présentation du module `graphh`

Ce module, dont la documentation est accessible là fournit des fonctions permettant d'interroger l'API GraphHopper.

Pour utiliser cette API, il faut :

a. Charger votre clé d'API lue dans le fichier `data/credentials.json` dans une variable `cle_api`
b. créer un client que l'on initialise en lui fournissant notre clé d'API :

```
gh_client = graphh.GraphHopper(api_key=cle_api)
```

Enfin, on appelle, pour ce client, la fonction qui nous intéresse. Par exemple, l'instruction :

```
gps_rennes = gh_client.address_to_latlong("Rennes, République")
```

permettra de stocker dans la variable `gps_rennes` une paire de coordonnées GPS correspondant à la position `"Rennes, République"`.

2.2 C'est parti !

1. Écrivez une fonction qui prenne en entrée deux chaînes de caractères décrivant des lieux et un client GraphHopper et retourne la distance en kilomètres séparant ces lieux.
2. Écrivez une fonction qui prenne en entrée une liste de positions GPS (chacune codée sous la forme d'un dictionnaire comme précisé plus bas) et un client GraphHopper et retourne une liste d'altitudes. Vous pourrez utiliser l'exemple suivant pour vos tests :

```
lst_gps = [
  {"lng": -1.426533, "lat": 48.005135},
  {"lng": -1.418127, "lat": 47.986058},
  {"lng": -1.427611, "lat": 47.989871},
  {"lng": -1.430202, "lat": 48.000354}
]
```

3. **En utilisant la fonction écrite à la question précédente**, écrivez une fonction qui prenne en entrée une liste de positions GPS et un client GraphHopper et retourne la somme des dénivelés positifs (d'une part) et négatifs (d'autre part). Par exemple, si on a une liste de coordonnées GPS pour lesquelles on a obtenu les altitudes suivantes :

```
[35.89, 65.73, 51.02, 34.62]
```

on devrait retourner la paire de valeurs :

```
(29.84, 31.11)
```

4. **En utilisant la fonction écrite à la question précédente**, écrivez une fonction qui prenne en entrée un nom de fichier JSON (contenant des informations sur diverses randonnées) et un client GraphHopper et affiche, pour chaque randonnée, son nom (attribut `"name"`) et la somme de ses dénivelés positifs (d'une part) et négatifs (d'autre part). Pour le fichier `mini-rando_gps.json`, on doit obtenir (après quelque temps, le nombre de requêtes à effectuer étant assez grand) une sortie du type :

```
TraceGPS Le long de la quincampoix - Pire-sur-Seiche
    - son dénivelé positif cumulé  16.54 m
    - son dénivelé négatif cumulé  16.54 m
TraceGPS Issued  Messac - CIRCUIT DU PORT
    - son dénivelé positif cumulé   8.97 m
    - son dénivelé négatif cumulé   8.97 m
TraceGPS Issued  Coemes-Retiers
    - son dénivelé positif cumulé  73.39 m
    - son dénivelé négatif cumulé  73.39 m
```

2.3 Pour aller plus loin (ce travail n'est pas à rendre)

5. Écrivez une fonction qui prenne en entrée un nom de fichier JSON (contenant des informations sur diverses randonnées) et un client GraphHopper et écrit dans un nouveau fichier JSON (dont le nom sera passé en paramètre de la fonction) une liste de dictionnaires contenant le nom de la randonnée et les informations de dénivelés positif et négatif), soit quelque chose du type :

```
[
    {
        "name": "TraceGPS Le long de la quincampoix - Pire-sur-Seiche",
        "D+": 16.54,
        "D-": 16.54
    },
    {
        "name": "TraceGPS Issued  Messac - CIRCUIT DU PORT",
        "D+": 8.97,
        "D-": 8.97
    },
    {
        "name": "TraceGPS Issued  Coemes-Retiers",
        "D+": 73.39,
        "D-": 73.39
    }
]
```