TUTORIEL PYTHON

Géolocalisation – Adresses à Coordonnées (lat., long.)

Ce document ne constitue pas un tutoriel exhaustif concernant la géolocalisation. De plus, notez que d'autres méthodes (faciles ou non, payantes ou non) existent. Cela étant dit, voici une méthode relativement simple, rapide et gratuite pour convertir vos **adresses civiques** en **coordonnées** (latitude, longitude). Vous pouvez aussi consulter un tutoriel de référence sur YouTube à l'adresse suivante : TUTO. La Figure 1 présente ainsi les quelques lignes de code nécessaires pour convertir les données.

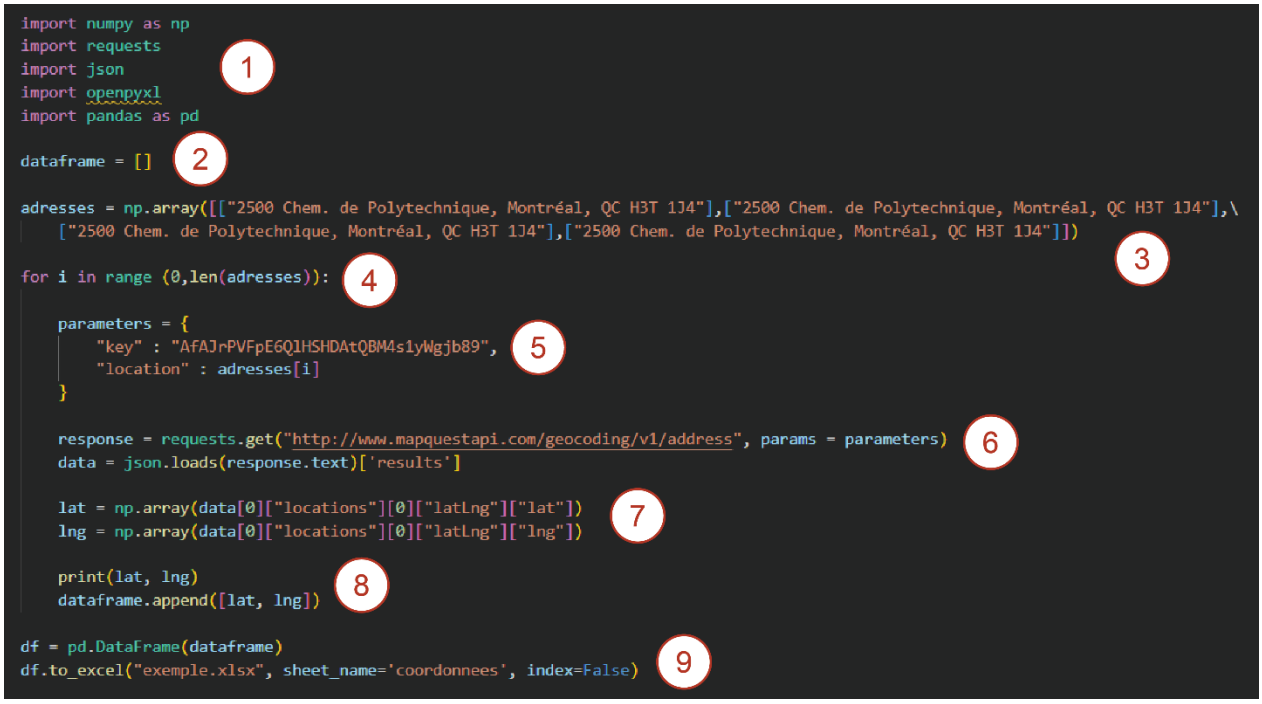

Figure 1

1. Importation des modules

La liste des modules importés dans ce code permet de manipuler les matrices/vecteurs/tables (module **numpy**), de faire un appel (« request ») directement à l'outil présent sur le site de *MapQuest* pour aller chercher l'ensemble de données géospatiales associées à l'adresse civique (module **requests**) et de lire/manipuler un dictionnaire JSON pour en retirer les valeurs désirées (module **json**). De plus, deux derniers modules permettent dans cet exemple de manipuler des « data frame » (module **pandas**) et d'écrire un fichier Excel directement à partir d'un « data frame » de Python (module **openpyxl**).

2. Initialisation de variable

Cette étape permet simplement d'**initialiser une variable** pour laisser savoir à Python que cette variable contient une **liste** qui est pour le moment vide.

3. Base de données contenant les adresses civiques

Dans l'exemple offert ici, des adresses sont entrées manuellement « from scratch » par la création d'une colonne contenant à chaque cellule une adresse. Il est **important** que le format de l'adresse soit standard, autrement, de mauvaises coordonnées pourraient être retournées. Contrairement à l'exemple ci-haut, vous aurez à utiliser le module **pandas** pour importer votre fichier de données « csv ». Il faudra donc créer une variable dans laquelle vous pourrez y mettre vos données importées. Par la suite, vous pourrez associer à une seconde variable uniquement la colonne désirée à l'aide de l'indexation. La Figure 2 suivante illustre un exemple de cette manipulation.

```
## Importer un fichier csv

import pandas as pd

data = pd.read_csv(r'C:\Users\Defaut\OneDrive - polymtl.ca\Z - AUTOMNE 2022\CIV8760 (TD)\Observation_age_genre-Grid view.csv')
print(data)

#'r' devant le path pour qu'il comprenne les caractères particuliers comme le "\"

data_age = np.array(data.age)
```

Figure 2

Ainsi, la variable « data_age » contenant la colonne de données désirées correspondrait à la variable « adresse » dans l'exemple de conversion des adresses.

4. Boucle « for » permettant de passer chaque adresse dans l'outil de conversion

La boucle « for » permet de passer chacune des adresses dans l'outil de conversion. Notez que l'intervalle de valeurs prises pour la boucle va de 0 à la longueur de la colonne en utilisant la fonction « len() ».

5. Création d'une variable contenant les paramètres pour la conversion

À chacune des conversions, la variable « parameters » se met à jour en changeant l'adresse civique. La clé (« key ») donne l'accès à l'outil de conversion. Cette dernière se retrouve sur le site de *MapQuest* sous le nom de « Cosumer Key ».

6. Conversion à l'aide de *MapQuest* et enregistrement des données dans une variable

Cette étape utilise tout simplement la variable « parameters » créée et fait appel à l'information se trouvant sur le site web de *MapQuest* en passant par l'URL. Ensuite, l'information associée à l'adresse civique est récoltée sous forme de données JSON. Cette information est lue, puis associée à la variable « data » (ne pas confondre les deux variables « data » dans chacune des figures, ce sont des exemples distincts).

7. Lecture des coordonnées à l'aide d'indexation

Cette étape permet tout simplement d'aller chercher seulement la longitude et la latitude dans toute l'information récoltée et insérée dans la variable « data ». Pour mieux comprendre pourquoi il y a autant d'indexation, tentez de faire « print(data) » pour voir ce qui se trouve uniquement dans l'information associée à une adresse civique. Notez qu'ici, plutôt que de mettre dans les variables « lng » et « lat » une seule adresse et « écraser » ce qui se trouvait dans la variable au préalable, vous pouvez créer une variable colonne qui insère chaque nouvelle coordonnée dans une nouvelle cellule. Cette manipulation est relativement simple et plusieurs exemples peuvent être trouvés sur le web en cherchant, par exemple, « Python create new row array ».

8. Affichage des résultats et mise en commun des coordonnées

Enfin, vous pouvez afficher les résultats pour voir si une erreur s'est glissée quelque part. De plus, au sein de la boucle « for », vous pouvez créer une liste dans laquelle pour chaque itération de la boucle, une ligne avec la latitude et la longitude est ajoutée.

9. Création d'un « data frame » et d'un fichier Excel

Afin d'exporter un fichier Excel, Python a besoin d'un « data frame ». Ce dernier est facilement créé à l'aide du module **pandas** et de la liste créée préalablement contenant les coordonnées. Ainsi, de ce « data frame », le fichier Excel est créé et exporté directement dans le même répertoire que votre fichier Python.