MINISTÈRE
DE L'ÉDUCATION
NATIONALE,
DE LA JEUNESSE
ET DES SPORTS
*Liberté
Égalité
Fraternité*

VOIE GÉNÉRALE

2DE | 1RE | TLE

Numérique et sciences informatiques

ENSEIGNEMENT
SPÉCIALITÉ

MODULARITÉ ET API

1. Principes généraux

Une API (*Application Programming Interface*), ou interface de programmation, est un ensemble normalisé de briques logicielles grâce auxquelles un système informatique offre des services à d'autres systèmes. Quand nous consultons la météo sur notre smartphone, il utilise l'API du service météo en question. La plupart des applications mobiles sont conçues autour d'une ou plusieurs API.

L'objectif d'une API est de fournir des fonctionnalités, sans révéler le fonctionnement interne de l'application qui fournit les données. C'est en cela qu'on dit qu'une API est en façade : on ne voit pas l'intérieur du bâtiment.

Concrètement, une API est constituée d'une bibliothèque logicielle, d'un service web et d'une description qui spécifie comment les clients peuvent interagir avec la plateforme logicielle qui fournit les données, appelée aussi **fournisseur**.

Comment procéder ?

Généralement, pour utiliser une API, il faut effectuer une requête sur un serveur web associé au fournisseur. Les éléments nécessaires à la requête sont précisés dans la description de l'API. Afin de limiter le nombre d'accès ou d'identifier des accès malveillants au serveur, il est souvent demandé une clé d'authentification. Cette clé s'obtient sur le site web du fournisseur. Elle peut être gratuite ou payante, selon les cas.

Intérêt

Pourquoi utiliser une API pour s'interfacer à une base de données plutôt que de télécharger simplement un ou plusieurs fichier(s) CSV de l'ensemble des données ?

Quelques avantages :

- les données changent très rapidement. Ce qui est vrai aujourd'hui peut-être faux demain. Une API permet un lien permanent avec la base de données ;
- on peut filtrer les données que l'on récupère, en fonction de multiples critères, et ne pas télécharger la totalité d'une base de données à chaque fois ;
- une API permet d'automatiser les requêtes et de les intégrer dans d'autres applications.

En Python ?

De nombreuses API disposent d'un **wrapper** Python : c'est un module Python qui permet d'interroger la base de données du fournisseur de façon simplifiée, sans avoir à écrire les requêtes Web, ni à interpréter les réponses. Celles-ci sont généralement écrites au format JSON, qui est un format d'échange de données, au même titre que le XML. À partir de ces réponses, le wrapper Python génère des listes ou des dictionnaires, facilement manipulables.

Dans cette activité, nous allons utiliser ces deux API :

- Genius, un site de tendances musicales, proposant une immense collection de paroles de chanson.
- Mars Rover Photos, une des nombreuses API proposées par la NASA.

2. API Genius

Installation

La description de l'API Genius est disponible à cette adresse : https://docs.genius.com/

Premièrement, il faut créer un compte gratuit qui permet de générer une clé d'authentification (`access_token`), indispensable pour accéder à l'API : https://genius.com/api-clients

Ce jeton peut être utilisé par tous les élèves. Il n'est pas nécessaire qu'ils s'inscrivent sur cette plateforme.

Deuxièmement, il faut installer le wrapper Python `lyricsgenius`, disponible à cette adresse : https://pypi.org/project/lyricsgenius/

Pour obtenir des précisions sur l'installation d'un package Python, suivre les liens suivants :

- https://www.delftstack.com/fr/howto/python/how-to-install-a-python-package-.whl-file/
- https://openclassrooms.com/fr/courses/6951236-mettez-en-place-votre-environnement-python/7013349-decouvrez-les-paquets-python

Une fois installé, ce wrapper s'utilise comme une bibliothèque Python classique.

Prise en main

Étape 1 : pour débuter, on demande à Genius les trois chansons les plus populaires d'Angèle.

```
# importation du module lyricsgenius
import lyricsgenius
# initialisation de la connexion avec l'API et création de
l'objet genius
genius = lyricsgenius.Genius(«coller ici le jeton access_
token»)
# utilisation d'une méthode search_artist de l'objet genius
artist = genius.search_artist(«Angèle», max_songs=3, sort=»po-
pularity»)
# affichage amélioré du résultat (facultatif)
print(artist.songs)
```

Normalement, la console doit répondre :

```
Song 1 : «Balance ton quoi»
Song 2 : «Tout oublier»
Song 3 : «Ta reine»
```

On remarque la syntaxe de la programmation orientée objet. L'objet `genius` dispose de nombreuses méthodes, dont `search_artist` : cette dernière prend des valeurs en entrée. Ici on a précisé le nom de l'artiste (Angèle), le nombre de chansons exigées, et le type de tri (la popularité de la chanson).

Étape 2 : lister les attributs et les méthodes de l'objet `genius` dans la console.

Étape 3 : que renvoie l'instruction suivante ?

```
artist = genius.search_artist(«Foo Fighters», max_songs=3,
sort=»title»)
```

Étape 4 : l'instruction ci-dessous renvoie l'URL de la photo de profil de l'artiste sur le site. Afficher cette image dans le navigateur, en copiant/collant l'URL.

```
artist.image_url
```

Étape 5 : pour récupérer les paroles d'une chanson, il faut utiliser la méthode `search_song`, qui prend en entrée le nom de l'artiste, puis le nom de la chanson.

```
chanson = genius.search_song(«Jalousie», «Angèle»)
print(chanson.lyrics)
```

Étape 6 : l'objet `chanson` dispose de nombreux attributs utiles, en plus des paroles. On peut ainsi récupérer la date de sortie du morceau (`.year`), l'URL de l'image de référence (`.song_art_image_url`) ou encore des informations sur les médias audio et vidéo associés à cette chanson (`.media`)

Récupérer ces informations en lien avec la chanson de l'étape 4, et vérifier qu'elles sont identiques à celles disponibles sur le site web (https://genius.com/).

Étape 7 : pour sauvegarder les paroles d'une chanson dans un fichier texte, on utilise la méthode `save_lyrics` :

```
chanson.save_lyrics(extension='txt')
```

Vérifier la création du fichier, et son contenu. Dans quel autre format peut-on sauvegarder ces paroles ? Dans quel but ?

À vous de jouer

Écrire une fonction qui détermine les *n* chansons les plus populaires d'un ou une artiste sur Genius. Pour chaque chanson, la fonction doit récupérer le classement, la date de sortie, et l'URL du clip vidéo s'il existe. Elle doit aussi sauvegarder les paroles dans un fichier texte, un par chanson. La structure de données générée par la fonction n'est pas imposée (tableau dynamique, dictionnaire ou autre) mais doit être adaptée au problème.

Entrées de la fonction :

- nom de l'artiste : `string` ;
- nombre de chanson·s : `integer`.

Sortie de la fonction :

- une structure de données contenant les informations désirées.

Tester cette fonction sur un artiste, et noter les résultats dans le tableau ci-dessous.

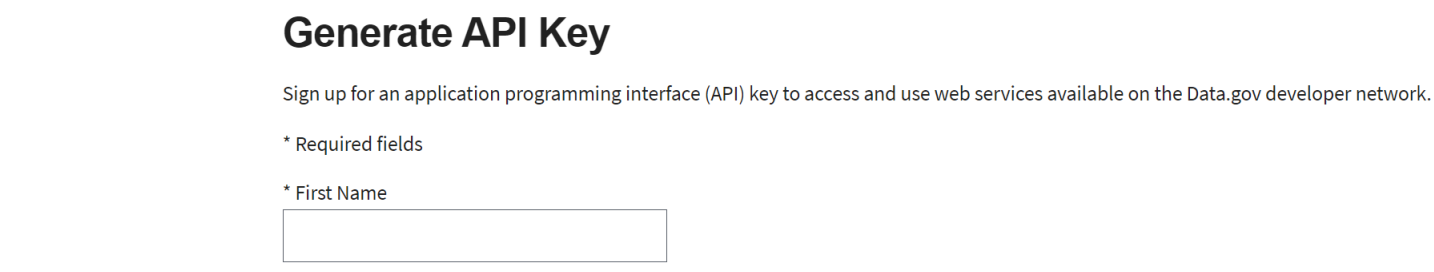
Nom de l'artiste :		
Titre de la chanson	**Date de sortie**	**Présence d'un clip vidéo ? (oui/non)**

3. API Mars Rover Photos

Installation

La description de l'API Mars Rover Photos est disponible à cette adresse : https://api.nasa.gov/.

Premièrement, il faut générer une clé d'authentification, indispensable pour accéder à l'API, à cette même adresse. Il est possible d'utiliser une clé de démo, nommé DEMO_KEY. Cette clé est plus limitée en terme de nombre de requêtes, et cela pourrait ne pas suffire à mener cette activité à son terme.



Generate API Key

Sign up for an application programming interface (API) key to access and use web services available on the Data.gov developer network.

* Required fields

* First Name

Deuxièmement, il faut installer le wrapper Python `nasaapi`, disponible à cette adresse : https://pypi.org/project/python-nasa-api/

Pour obtenir des précisions sur l'installation d'un package Python, suivre les liens suivants :

- https://www.delftstack.com/fr/howto/python/how-to-install-a-python-package-.whl-file/
- https://openclassrooms.com/fr/courses/6951236-mettez-en-place-votre-environnement-python/7013349-decouvrez-les-paquets-python

Une fois installé, ce wrapper s'utilise comme une bibliothèque Python classique.

Prise en main

Étape 1 : pour débuter, on demande l'URL de la 1ère photo que le rover Curiosity a prise le 50ème jour martien, avec sa caméra arrière nommée Rear Hazard Avoidance Camera ou RHAZ.

```
# importation du sous-module MarsRovers du module nasaapi
from nasaapi import MarsRovers
# initialisation de la connexion avec l'API et création de l'objet rovers
rovers = MarsRovers(«coller ici la clé API NASA», 50, «RHAZ»)
# récupération des données de Curiosity (dictionnaire)
cur = rovers.curiosity()
# accès à l'URL de la première photo
url_photo = cur[«photos»][0][«img_src»]
```

Normalement, la variable url_photo doit contenir cette chaîne de caractères :

http://mars.jpl.nasa.gov/msl-raw-images/proj/msl/redops/ods/surface/sol/00050/opgs/edr/rcam/RRA_401933077EDR_F0042862RHAZ00305M_.JPG

Copier/coller dans un navigateur l'URL reçue pour vérifier l'existence de cette photo.

> **Remarque**
> Le wrapper Python `nasaapi` étant moins travaillé que celui de Genius, il nous faut chercher dans le dictionnaire que l'on récupère pour trouver les données intéressantes. La maîtrise des types construits est donc indispensable.

Étape 2 : à l'aide des données reçues du rover Curiosity (dictionnaire `cur`), compléter le tableau ci-dessous.

Id de Curiosity	Date de lancement	Date d'atterrissage sur Mars	Date terrestre de la photo	Nb de photos du jour	Id de la première photo du jour

Étape 3 : Il est possible d'envoyer une URL directement à un navigateur grâce au module `webbrowser`. On peut ainsi afficher automatiquement les photos trouvées.

Exemple :

```
# importation du module webbrowser
import webbrowser
# importation du sous-module MarsRovers du module nasaapi
from nasaapi import MarsRovers
# initialisation de la connexion avec l'API et création de l'objet rovers
rovers = MarsRovers("coller ici la clé API NASA", 1000, "NAVCAM")
# récupération des données de Curiosity (dictionnaire)
cur = rovers.curiosity()
# accès à l'URL de la quatrième photo
url_photo = cur["photos"][3]["img_src"]
# envoi de l'URL dans le navigateur
webbrowser.open_new_tab(url_photo)
```

Vérifier que le code ci-dessus fonctionne et que le paysage s'affiche bien dans le navigateur.

À vous de jouer :

Écrire une fonction qui affiche dans le navigateur toutes les photos prises par le rover Curiosity un jour précis, par une de ces caméras (voir la liste exhaustive sur https://api.nasa.gov/).

Entrées de la fonction :

- numéro du jour martien : `integer`;
- abréviation de la caméra : `string`.

Tester cette fonction avec les caméras FHAZ et MARDI, au 3000ème jour martien. À quel jour terrestre cela correspond-il ?

4. Une autre API ?

De nombreux autres wrappers Python sont disponibles à cette adresse :

https://github.com/discdiver/list-of-python-api-wrappers

Ils donnent accès à plus d'un centaine d'API dans des domaines très différents.

5. Les fichiers Python associés à la ressource

Pour les télécharger, cliquer ici.