Info - 2A - TP3

Communication Client / Serveur

Objectif du TP : Mise en oeuvre d'une communication client / serveur

Si vous avez des problèmes de performances avec votre terminal, vous pouvez changer le moteur de rendu, allez sur Files > Preferences > Settings et entre les accolades (dans la fenêtre de droite), ajoutez la ligne suivante :

```
"terminal.integrated.rendererType": "dom"
```

1. Côté Client - Appeler une api

Récupérez la première partie du squelette sur moodle (*squelette-tp3-**client**.zip*) et ouvrez le projet qu'il contient sous VS Code.

Dans cette première partie, nous allons utiliser les apis du site open-data de Rennes : https://data.rennesmetropole.fr/explore/

Avant de lancer l'exemple, ouvrez le fichier *exemple.py*

Dans cet exemple nous récupérons les informations depuis une url (https://data.rennesmetropole.fr/…..)

Pour consulter ce que renvoie cette url, nous allons utiliser **Insomnia**

Lancez Insomnia depuis le menu démarrer (Menu démarrer / Informatique / Insomnia)

Allez dans le menu application/préférence pour configurer le proxy. Cliquez sur la checkbox *Enable Proxy* et saisissez dans les champs *HTTP Proxy* et *HTTPS Proxy* : **http://pxcache-02.ensai.fr:3128**

Dans insomnia, cliquez sur le bouton + à gauche > New Requests. Dans la popin qui s'ouvre, mettez comme nom *Prêts de DVD* puis appuyez sur **Create** (vous avez un menu déroulant avec GET sélectionné, laissez cette méthode).

En haut, dans le champ texte à côté de GET, collez l'url (https://data.rennesmetropole.fr/…..) qui est présente dans l'exemple puis appuyez sur **SEND**.

Vous obtenez dans la fenêtre de droite le résultat de la requête.

Maintenant nous allons lancer la programme, il se base sur 2 librairies :
- requests (http://docs.python-requests.org/) qui permet d'appeler l'API
- tabulate (https://pypi.org/project/tabulate/) qui facilite l'affichage des tableaux

Installez ces dépendances (présentes dans le fichiers requirements.txt)

```
pip install -r requirements.txt --user --proxy http://pxcache-02.ensai.fr:3128
```

Et lancez l'exemple.

Vous devriez voir s'afficher la liste des films star wars qui ont été emprunté.
De façon similaire, vous allez maintenant appeler une autre url pour afficher les arrêts de bus de Bruz accessibles aux personnes à mobilité réduite

Avec un navigateur, allez sur le site https://data.rennesmetropole.fr/page/home/

Puis dans l'onglet *Mes Données*, allez sur *Données géographiques du réseau STAR : arrêts physiques* et enfin sur l'onglet *API*

En bas de cette page, vous avez l'url pour récupérer les résultats de l'appel. Elle est construite en fonction des paramètres que vous choisissez sur la page.

Cliquez dans le menu à gauche sur *Bruz* (sous Commune (nom)), vous devez constatez que dans l'url un nouveau paramètre a été ajouté : *refine.nomcommune=Bruz*
Ainsi nous ne récupérons que les arrêts de bus situés à Bruz

Vous avez un champs texte sur la page appelé *rows*, c'est le nombre de résultats de la requête. Par défaut il vaut 10. Saisissez **100** pour récupérer l'ensemble des résultats et cliquez sur le bouton **Envoyez** en bas.

Vous devez constater que dans l'url un nouveau paramètre a été ajouté : *rows=100*

Maintenant cliquez sur le lien, un nouvel onglet va s'ouvrir. Récupérez l'url et saisissez là dans Insomnia comme vous l'avez fait précédemment (Cliquez sur +, sur *New Request* …)

Consultez les données qui sont renvoyées par l'url.

Nous voulons récupérer les *coordonnées* (latitude et longitude) des arrêts de bus accessibles aux personnes à mobilité réduite (*pmr*).

Affichez un tableau listant les coordonnées des arrêts de bus accessibles aux personnes à mobilité réduite avec pour colonnes latitude et longitude

Nous nous contentons d'afficher les coordonnées dans le cadre de ce TP mais nous pourrions les afficher sur une carte avec OpenStreetMap par exemple.

2. Côté Serveur - Construire son API

Ouvrez une nouvelle fenêtre VS Code (Menu File > New Window)
Récupérez la seconde partie du squelette sur moodle (*squelette-tp3-**serveur**.zip*) et ouvrez le projet qu'il contient dans cette nouvelle fenêtre.

Installez les dépendances :

```
pip install -r requirements.txt --user --proxy http://pxcache-02.ensai.fr:3128
```

Nous installons **Flask** qui permet de créer des apis (http://flask.pocoo.org)

Lancez l'exemple. Il contient une serveur qui répond sur 2 urls : **/films** et **/film**

Sur Insomnia allez dans le menu *Application* > *Preferences* et décochez la case *Enable Proxy*

Ajoutez une nouvelle requête GET (New Request) et interrogez l'url http://localhost:5000/films

Vous devez voir s'afficher la liste des 3 films que l'on retrouve dans *exemple.py*

Si vous regardez le code, vous avez une méthode *movieList* qui supporte la méthode *GET* et qui retourne la liste des films.

Nous allons maintenant ajouter un film.

Sur Insomnia, ajoutez une nouvelle requête (New Request) mais dans la popin au lieu de laisser GET, remplacer par **POST**, une nouvelle popin va s'afficher avec écrit *No Body.* Remplacez par **JSON**.

L'url que l'on va utiliser est http://localhost:5000/film

Dans la fenêtre de gauche (sous JSON), écrivez :

{"name": "Mon film préféré"}

Vous pouvez remplacer *Mon film préféré* par le film que vous souhaitez.

Une fois que vous avez exécuté cette appel, appelez à nouveau l'url http://localhost:5000/films en GET et constatez que votre film a été ajouté à la liste

Si vous regardez le code vous verrez une méthode addMovie qui ajoute le film reçu à la liste

Maintenant que vous avez compris comment fonctionnait le serveur, vous allez ajouter votre propre méthode pour gérer une liste de joueurs qui auront un pseudonyme et un score.

Créez les méthodes GET (/joueurs) et POST (/joueur) et testez les appels avec Insomnia

3. Communication Client/Serveur

Dans votre première fenêtre VS Code (celle utilisée pour la première partie), créez un programme qui ajoute une liste de joueurs (en appelant la méthode POST de votre serveur) et qui ensuite affiche les 3 premiers (en utilisant la méthode GET pour récupérer la liste)

Note : vous pouvez modifier votre méthode POST pour qu'elle prenne un tableau comme paramètre

4. Un peu de webscraping

Nous allons récupérer directement du contenu disponible sur le web. Choisissez une page wikipedia et affichez le contenu de la balise h1 (le titre de la page)

Le contenu que vous allez récupérer sera du html, pour extraire le contenu de la balise h1, vous pouvez utiliser la libraire lxml (comme dans le TP précédent pour le XML)