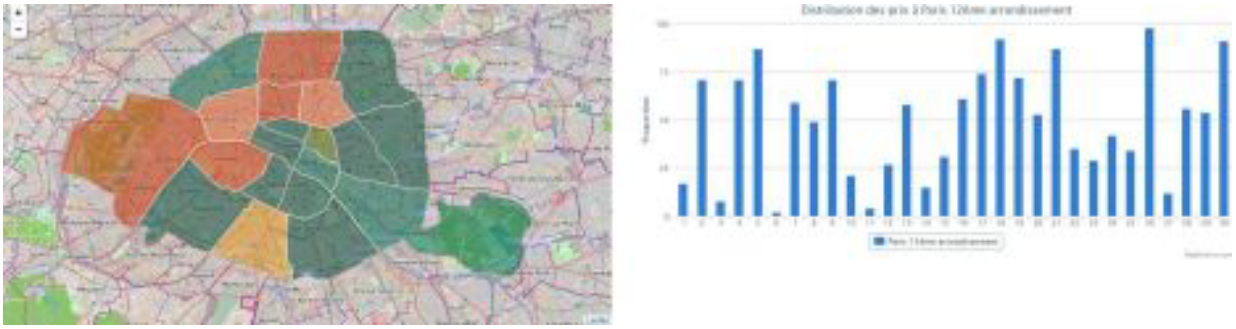
Carte des prix à Paris

L'objectif est de réaliser une carte des prix moyens (€/m²) de mise en vente par arrondissement à Paris, accompagnée de statistiques, à partir des annonces diffusées par les agences immobilières sur le site web de meilleursagents.com.

Pour ce faire, l'exercice sera constitué en deux parties :

1. Collecter, nettoyer et organiser l'information depuis le site de MeilleursAgents (p. 2)
2. Mettre à disposition de la donnée au travers d'une API Flask (p. 5)



Exemple de ce qu'on veut obtenir (les données étant ici aléatoires)

Gardez en tête qu'il y a plusieurs bonnes manières de faire chacune des questions. L'objectif est que vous fassiez un rendu globalement cohérent, fonctionnel, et pour lequel vous êtes satisfait.

Il y a un readme dans le répertoire qui récapitule quelques aspects techniques de l'exercice, n'hésitez pas à y jeter un œil et à le compléter.
`/home/meilleursagents/pikachu/readme`

1. Collecter l'information

Sur le site web de MeilleursAgents.com, dans la rubrique "Acheter", on trouve des annonces de biens immobiliers actuellement proposés à la vente, par les agences partenaires de MeilleursAgents. On souhaite donc collecter cette information.

1.1. Scénario de navigation

La première étape consiste à parcourir l'ensemble des annonces disponibles sur notre site web. Celles-ci sont accessibles via une adresse de recherche, sous la forme suivante :
https://www.meilleursagents.com/annonces/achat/search/?item_types=ITEM_TYPE.APARTMENT,ITEM_TYPE.HOUSE&place_ids=2259,2260&page=1

- `item_types` est la liste des types de bien sélectionnés (appartement et maison)
- `place_ids` est la liste des ids internes des départements, villes, quartiers etc… propres à Meilleurs Agents
- `page` est le numéro de la page (en partant de la page 1)

1.1.1. Filtre par type de bien

S'agissant d'une carte à Paris, où les maisons sont suffisamment rares pour être ignorées, nous nous limiterons aux seuls appartements. L'argument `item_types` sera donc toujours fixé à : `ITEM_TYPE.APARTMENT`

1.1.2. Filtre par localisation

Au sein de Paris, nous souhaitons sectoriser les annonces par arrondissement. L'argument `place_ids` prendra donc pour valeur, les identifiants des arrondissements de Paris, tels qu'indiqués dans le tableau suivant :

Arrondissement	id	Arrondissement	id
Paris 1er	32682	Paris 11ème	32692
Paris 2ème	32683	Paris 12ème	32693
Paris 3ème	32684	Paris 13ème	32694
Paris 4ème	32685	Paris 14ème	32695
Paris 5ème	32686	Paris 15ème	32696
Paris 6ème	32687	Paris 16ème	32697
Paris 7ème	32688	Paris 17ème	32698
Paris 8ème	32689	Paris 18ème	32699
Paris 9ème	32690	Paris 19ème	32700
Paris 10ème	32691	Paris 20ème	32701

Ces identifiants sont également disponibles en base de données, dans une table `geo_place`. Les arrondissements de Paris sont donc les entités géographiques ayant un cog (Code Officiel Géographique) proche de Paris `75056`.

1.1.3. Pagination

12 annonces sont affichées par page. Il faut donc parcourir toutes les pages de résultats afin de consulter l'ensemble des annonces.

Le nombre de pages de résultats est différent pour chaque arrondissement : de zéro à plusieurs dizaines de pages. Il faudra donc imaginer un mécanisme s'adaptant au nombre de pages à parcourir.

1.2. Extraction des caractéristiques des annonces

Pour chaque annonce, on est intéressé par les caractéristiques suivantes :

- `listing_id` : identifiant de l'annonce pour Meilleurs Agents
- `place_id` : identifiant de l'arrondissement (celui passé en paramètre de la recherche)
- `price` : prix de mise en vente, en valeur entière d'euros
- `area` : superficie du bien, en valeur entière de mètres carrés
- `room_count` : nombre de pièces du bien, en valeur entière également

L'extraction de ces caractéristiques nécessite l'utilisation d'outils de traitement du texte, tels que :

- un parseur HTML (module lxml.html en Python)
- un langage de parcours de DOM : xpath
- un langage de recherche de motifs : regexp (module re en Python)

La superficie (`area`) et le nombre de pièces (`room_count`) ne sont pas des informations structurées. Elles doivent être extraites à partir du titre de chaque annonce : « *2 pièces de 46 m²* ».

Attention : Les appartements de 1 pièce sont notés "Studio"**.**

1.3. Structure les informations en base de données

L'information extraite du site web doit ensuite être stockée en base de données, dans une ou plusieurs tables qu'il faudra définir au préalable.

En plus de leurs caractéristiques, on veut aussi modéliser l'évolution des annonces dans le temps. Concrètement, on veut connaître :

- La **date de mise en ligne** (ou au moins la date à laquelle on l'a vue pour la première fois)
- La **date de retrait du site** (ou au moins la dernière date à laquelle on l'a vue)
- L' **historique complet des prix**

Les informations requises pour se connecter au serveur de base de données sont dans un fichier de settings situé sur la machine mise à votre disposition : `/home/meilleursagents/pikachu/settings.py`

2. Restituer l'information

La carte et l'histogramme présentés en introduction de ce document sont servis par une application web écrite en Python à l'aide du micro-framework Flask.

L'application est déjà fonctionnelle, il reste à implémenter la manipulation la donnée précédemment stockée.

2.1. Cartographier les prix par arrondissement

Au chargement de la page web, le code JavaScript permettant la génération de la carte interroge l'application web afin d'obtenir la liste des entités géographiques à afficher. L'application web fournit en retour une structure de données au format GeoJSON contenant la liste des arrondissements à afficher, leur forme géométrique ainsi qu'un prix moyen.

La couleur de la forme géométrique dépend du prix de l'arrondissement qu'elle représente, selon la même échelle de couleurs que celle actuellement utilisée pour la carte de Paris sur le site web de Meilleurs Agents.

Il ne reste donc plus qu'à calculer, pour chaque arrondissement, **le prix moyen par mètre carré réel** et à intégrer ce résultat dans la réponse de l'application web au code JavaScript chargé de la génération de la carte.

2.2. Afficher des statistiques par arrondissement

Lorsque l'on clique sur un arrondissement, un histogramme apparaît en regard de la carte : cet histogramme représente la distribution du volume d'annonces, par gammes de prix, dans cet arrondissement.

De la même manière que précédemment, le code JavaScript chargé de la génération de cet histogramme interroge l'application web avant chaque affichage, en passant le code de l'arrondissement en paramètre.

L'application web fournit en retour une structure de données au format JSON contenant, entre autres, les valeurs de chacune des barres de l'histogramme. L'axe des ordonnées est alors mis à l'échelle automatiquement en fonction des valeurs fournies.

Il ne reste donc plus qu'à calculer, pour l'arrondissement ciblé, **la distribution des annonces par gammes de prix** et à l'intégrer à la réponse de l'application web au code JavaScript chargé de la génération de l'histogramme.

2.3. Afficher le prix moyen de l'arrondissement (bonus)

Entre la carte et l'histogramme, nous n'affichons nulle part le prix par mètre carré moyen de l'arrondissement de manière numérique.

Que faut-il faire pour l**'afficher sur la page web lorsqu'on clique** sur un arrondissement de la carte ?