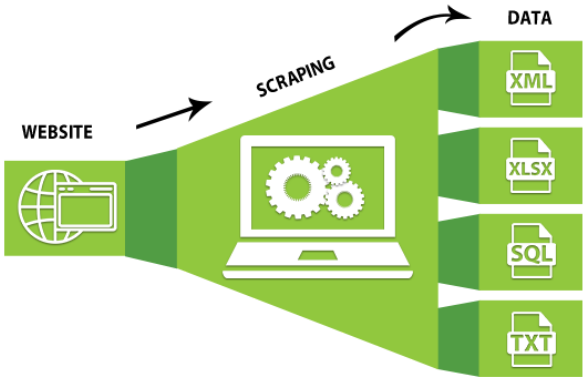


Introduction

Dans le métier de data analyst, il est important de pouvoir se constituer des jeux de données si ceux-ci n'existent pas. Une façon de faire est d'utiliser internet et de récupérer les données sur des sites intéressants.
Le Web Scraping permet d'automatiser ce genre de procédure et de créer des routines récupérant et structurant les données au format défini par le data analyst.

Objectifs

A l'issue de ce module, vous serez capable de :

- Scraper les données d'un site à l'aide de python avec les packages suivants :
  - Scrapy
  - Selenium

Pré-requis

- Programmation en Python
- Ligne de commande linux

Compétences à acquérir

- Récupérer et structurer des données venant d'un site web.

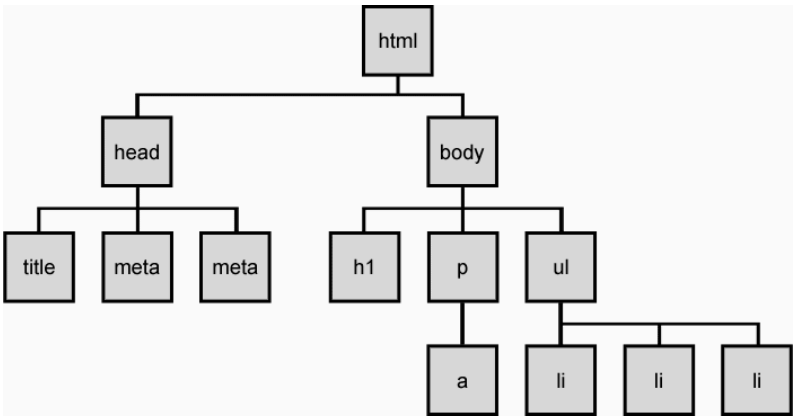
Démarche pédagogique

Avec les packages scrapy et selenium les apprenants vont accéder à des données non structurées. Dans un premier temps, ils vont récupérer et structurer des données sur les livres référencés sur un site web. Dans un second temps, ils vont collecter des données au sujet d'entreprises de l'isère dans différents secteurs d'activité avec scrapy puis avec Selenium

Une introduction rapide au HTML

Pour scraper des données il est nécessaire de comprendre les bases du langage web HTML.

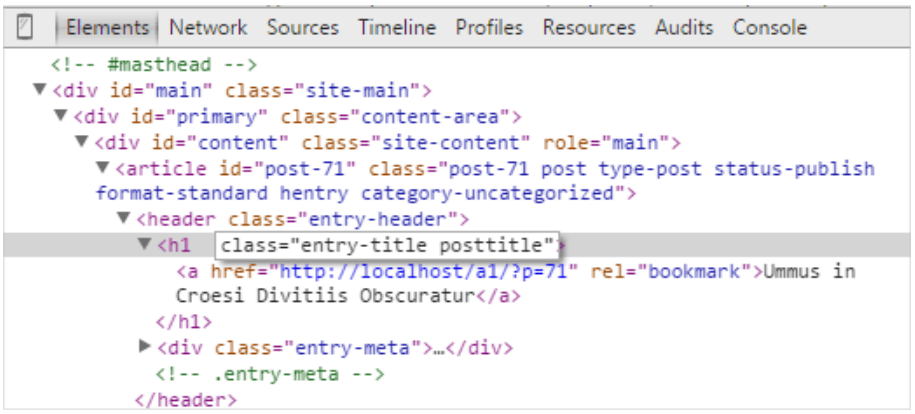
Le langage HTML décrit le contenu d'une page web, sous forme d'une arborescence, pour cela il utilise des balises qui englobent tout ou partie du contenu afin de structurer la position, le formatage et plein d'autres paramètres concernant les éléments situés à l'intérieur.



L'exemple ci-dessous correspond à une page web très basique. La page est englobée entre les balise `<html>` (balise ouvrante) et `</html>` (balise fermante). La partie contenue entre les balises `<head>` et `</head>` contient essentiellement des métadonnées, pour ce mini-projet nous nous intéresserons plus au contenu de la page contenu entre les balises `<body>` et `</body>`.

```
<html>
  <head>
    <title>Heading in HTML</title>
  </head>
  <body>
    <h1>Heading 1</h1>
    <h2>Heading 2</h2>
    <h3>Heading 3</h3>
    <h4>Heading 4</h4>
    <h5>Heading 5</h5>
    <h6>Heading 6</h6>
  </body>
</html>
```

Lors de l'inspection d'une page web avec votre navigateur web, on retrouve les balises vues précédemment ainsi que des attributs (`class`, `href`, `id`,...) qui sont généralement utilisés pour définir plus précisément le style, l'usage ou la position du contenu de la balise.



(Regardez comment utiliser l'inspecteur de code de votre navigateur, cela sera probablement utile)

Maintenant que la structure d'un site est introduite, nous allons utiliser python pour récupérer des informations.

Comment trouver les tag html ? Sur n'importe quelle page, cliquez droit et choisissez "inspect". Cela ouvre une écran avec le code html de votre page.

Partie 1: Scrapy

Books to Scrape

- https://books.toscrape.com/

Objectifs

- Créer un premier projet de scraping en se familiarisant avec scrapy
- Utiliser le scraping pour collecter et structurer les données d'un site

Scrapy utilise ce que l'on appelle des 'spider', chaque 'spider' a pour but d'aller récupérer et structurer les données d'un site.

Avant de commencer à scraper:

- Installer la version 2.11.0 de scrapy

La gestion d'un projet scrapy peut se faire assez facilement en ligne de commande, avant de créer notre premier 'spider', il est nécessaire de créer un nouveau projet scrapy que l'on nommera "bookstore" avec la commande suivante:

```
scrapy startproject bookstore
```

- Créer le projet

Cette commande génère la configuration du projet de scraping, on obtient ensuite l'arborescence suivante:

```
bookstore
|-- bookstore
|   |-- __init__.py
|   |-- items.py
|   |-- middlewares.py
|   |-- pipelines.py
|   |-- settings.py
|   `-- spiders
|       `-- __init__.py
`-- scrapy.cfg
```

Pour nous familiariser avec l'outil nous allons créer un premier 'spider' pour récupérer les noms des livres de la page 1.
La commande `scrapy genspider nom_spider nom_domaine` permet d'obtenir un template à compléter pour créer un spider.

Nous allons dans un premier temps récupérer les titres des livres présents sur le site.

- Créer un premier 'spider' nommé "`book_names`"

Avant de créer la routine pour extraire les noms de la page, il est nécessaire de compléter le fichiers `items.py` en créant une nouvelle classe qui va contenir un attribut nommé `name`. Cette structure de données va "accueillir" les données scrapées!

Pour extraire les données on utilise une notation nommée XPATH qui permet de naviguer dans des documents comme l'HTML. (pour en savoir plus voir ici)

- Compléter le code du fichier `bookstore/spiders/book_names.py`
  - Créez une fonction `start_requests` permettant de faire une requête sur la page principale
  - Complétez la fonction `parse` afin d'utiliser la réponse du serveur pour en extraire les titres des livres.
  - Stockez les données dans la classe créée dans `items.py`

(TIP: pour cette partie il peut être utile d'afficher seulement les titres des livres en sortie afin de s'assurer que l'on récupère bien les bonnes données!)

Maintenant que les données voulues sont accessibles pour une page, le 'spider' va être modifié afin de parcourir les différentes pages (lors d'un clic sur le bouton next en bas de la page).

- modifiez la fonction `start_request` or `parse` afin de parcourir toutes les pages
- stockez le résultat dans un fichier csv

Combien y a-t-il de livres dans les pages du site?

- ajoutez une colonne afin de numéroter les livres de façon unique par page
- stockez le résultat dans un autre fichier csv.

Bravo! Vous avez scrappé votre premier site!

Question 1

Maintenant créez un (ou plusieurs) nouveau(x) spider(s) pour récupérer les colonnes suivantes:

- numéro de page
- numéro unique (comme fait précédemment)
- titre
- prix
- livre en stock (booléen)

Livrables

- Les fichiers csv demandés.

Pour aller plus loin

Ajoutez au csv:

- Le genre du livre
- Le code UPC (présent dans la page concernant le livre)

Ressources

- https://docs.scrapy.org/en/latest/index.html
- https://realpython.com/web-scraping-with-scrapy-and-mongodb/#scrapy-project
- https://scrapeops.io/python-scrapy-playbook/scrapy-save-csv-files/
- https://towardsdatascience.com/a-minimalist-end-to-end-scrapy-tutorial-part-i-11e350bcdec0

Partie 2: Les entreprises en isère

Mise en contexte

Dans ce module d'extraction de données, vous allez apprendre à **collecter automatiquement des informations disponibles sur le web** à l'aide de différents outils. Pour rendre cet apprentissage concret, nous allons travailler sur un **cas réel** : la mise à jour des données d'entreprises situées en Isère, en collaboration avec **Invest in Grenoble Alpes**. (Nous reviendrons plus en détail sur cette collaboration et sur le projet dans le module BI.)

Ce qu'il faut simplement retenir pour l'instant, c'est qu'il s'agit d'une **agence qui œuvre pour le développement économique du territoire** et la **valorisation des entreprises locales**.

Les données que vous allez extraire (nom, SIREN, localisation, etc.) pourront ensuite être utilisées dans le module BI. Cependant, pas d'inquiétude : vous êtes ici pour **découvrir et apprendre**, et nous vous fournirons un **jeu de données** pour débuter ce projet. Vous verrez alors comment **représenter et analyser ces informations sous forme de tableaux de bord**, afin de mieux comprendre l'activité économique locale et de la mettre en valeur.

Pour le moment, ne vous souciez pas si certains termes comme **BI (Business Intelligence)** ou **ETL** vous semblent nouveaux : vous les aborderez progressivement dans la suite de la formation. Retenez simplement que **l'extraction de données est la première brique d'un projet data** — elle permet de **récupérer les informations nécessaires** pour ensuite **les transformer, les analyser et les partager** de façon claire et utile.

Objectif

L'objectif de cette partie est d'utiliser Scrapy pour récupérer des informations sur les entreprises des secteurs de l'informatique, des technologies de l'information et du logiciel en Isère. Les données collectées seront utilisées pour créer un dataset complet sur les entreprises de cette région.

Secteurs d'activités spécifiques à récupérer

Nous allons nous concentrer sur les secteurs dont les codes NAF/APE commencent par 62 et 63.

Les étapes suivantes sont à effectuer en utilisant scrapy.

Étape 1 : Accéder à la Liste des Secteurs

1. **URL de Base** : Commencez par accéder à la page principale qui liste les différents

secteurs d'activité en Isère.
    - Liste des secteurs en Isère

Étape 2 : Parcourir les Pages par Secteur

2. **Pages par Secteur** : Pour chaque secteur identifié, accédez à la page spécifique qui liste les entreprises de ce secteur.
    - Exemple pour le secteur de la programmation informatique :
        - Programmation Informatique en Isère

Étape 3 : Récupérer les Données

3. **Données à Extraire** : Pour chaque entreprise listée, récupérez les informations suivantes :
    - **SIREN** : Le numéro SIREN de l'entreprise.
    - **Localisation** : L'adresse ou la localisation de l'entreprise.
    - **Nom** : Le nom de l'entreprise.

Étape 4 : Vérifier le Nombre d'Entreprises par Secteur

4. **Validation des Données** : Assurez-vous de vérifier le nombre d'entreprises par secteur pour garantir que toutes les données ont été correctement récupérées.

Pour Aller Plus Loin

Pour enrichir davantage votre dataset, vous pouvez parcourir les liens "Afficher les 800 entreprises suivantes" présents sur les pages de résultats. Cela vous permettra de récupérer un plus grand nombre d'entreprises et de compléter votre dataset de manière exhaustive.

1. **Identifier les Liens de Pagination** : Repérez les liens de pagination sur les pages de résultats, tels que "Afficher les 800 entreprises suivantes".
2. **Automatiser la Navigation** : Utilisez Scrapy pour automatiser la navigation à travers ces liens et récupérer les données des entreprises supplémentaires.
3. **Ajouter les Données au Dataset** : Intégrez les données supplémentaires dans votre dataset existant pour obtenir une vue complète des entreprises des secteurs ciblés.

Livrables

- Un script Scrapy fonctionnel capable de récupérer les données requises.
- Un dataset complet contenant les informations sur les entreprises des secteurs ciblés en Isère.

Ressources

- Documentation de Scrapy : Scrapy Documentation
- Tutoriel sur le Web Scraping avec Scrapy : Scrapy Tutorial

- Bonnes Pratiques de Web Scraping : https://scrapfly.io/blog/web-scraping-with-scrapy/

Partie 3 : Utilisation de Selenium pour le Web Scraping

Objectif

L'objectif de cette partie est d'utiliser Selenium pour compléter le scraping des données sur le site annuaire-entreprises.data.gouv.fr. Ce site ne peut pas être scrappé directement avec Scrapy car les données sont chargées dynamiquement via JavaScript. Selenium permet de simuler des interactions utilisateur, comme cliquer sur des boutons pour naviguer entre les pages. Nous allons faire évoluer le dataset créé dans la partie 2 avec les données supplémentaires récupérées via Selenium.

Prérequis

Avant de commencer, assurez-vous d'avoir installé Selenium et les bibliothèques nécessaires :

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.support.ui import WebDriverWait
from selenium.common.exceptions import TimeoutException
```

Champs à Extraire

Pour chaque entreprise listée sur le site annuaire-entreprises.data.gouv.fr, nous allons extraire les informations suivantes :

1. **Nom de l'entreprise** : Le nom officiel de l'entreprise.
2. **SIREN/SIRET** : Le numéro SIREN ou SIRET de l'entreprise.
3. **Adresse** : L'adresse complète de l'entreprise.
4. **Code Postal et Ville** : Le code postal et la ville où l'entreprise est située.
5. **Secteur d'Activité** : Le secteur d'activité principal de l'entreprise.
6. **Capital** : Le capital de l'entreprise.
7. **Dirigeants** : Les noms des dirigeants de l'entreprise.
8. **Nombre de salariés** : Le nombre de salariés de l'entreprise.

Étapes à Suivre

Étape 1 : Configuration de Selenium

1. **Initialisation du WebDriver** : Configurez le WebDriver pour utiliser un navigateur

comme Firefox ou Chrome.

```
driver = webdriver.Firefox()
```

Étape 2 : Scraping sur annuaire-entreprises.data.gouv.fr

Accéder à la Page : Utilisez Selenium pour accéder à la page principale du site.

```
url = "https://annuaire-entreprises.data.gouv.fr"
driver.get(url)
```

Pagination avec Selenium : Utilisez Selenium pour naviguer entre les pages.

```
while True:
    # Extraire les éléments souhaités
    entreprises = driver.find_elements(By.CLASS_NAME, 'entreprise')
    for entreprise in entreprises:
        # Code pour extraire les informations des entreprises
        pass

    # Pagination
    try:
        next_button = WebDriverWait(driver, 10).until(
            EC.presence_of_element_located((By.XPATH, "//a[contains(text(), 'Page suivante')]"))
        )
        next_button.click()
    except TimeoutException:
        print('No more pages')
        break
```

Étape 3 : Extraction des Données

- **Extraire les éléments souhaités :**
  Pour chaque entreprise trouvée sur la page, extrayez les informations suivantes : nom, SIREN/SIRET, adresse, code postal et ville, secteur d'activité, capital, dirigeants, et nombre de salariés.

Étape 4: Enregistrement des Données

Créez un fichier csv et enregistrez les données extraites dans ce fichier. (il est possible de créer plusieurs fichiers csv si cela est plus simple)

Astuce

Pour limiter le nombre de requêtes il est possible d'utiliser les numéro SIREN récupérés dans la partie 2 et de les utiliser pour créer les urls.

Pour Aller Plus Loin

Pour enrichir davantage votre dataset, vous pouvez également scraper des données supplémentaires depuis le site société.com en utilisant les mêmes techniques de Selenium.

Livrables

- Un script Selenium fonctionnel capable de récupérer les données requises depuis annuaire-entreprises.data.gouv.fr.
- Un dataset complet et évolué contenant les informations sur les entreprises des secteurs ciblés en Isère, incluant les données supplémentaires récupérées via Selenium. (le dataset doit être stocké en json ou en csv.

Ressources

- **Documentation de Selenium** : Selenium Documentation
- **Tutoriels sur le scraping avec selenium:**
  https://builtin.com/articles/selenium-web-scraping
  https://scrape.do/blog/selenium-web-scraping/
- **Guide pour la pagination avec Selenium** : Efficiently Scraping Multiple Pages of Data