**Comprendre le Processus de Web Scraping**

Avant de plonger dans le code, il est important de comprendre les étapes générales du web scraping :

1. **Identifier la cible:** Déterminez précisément les données que vous souhaitez extraire (prix de l'or, date, source, etc.) et le site web où elles se trouvent.
2. **Analyser la structure HTML:** Utilisez les outils de votre navigateur (Inspect Element) pour comprendre comment les données sont organisées dans le code HTML de la page.
3. **Choisir les outils:** Python est un excellent choix pour le web scraping grâce à des bibliothèques comme Beautiful Soup et Requests.
4. **Écrire le script:** Créez un script Python qui :
   * Fait une requête HTTP pour récupérer la page web.
   * Analyse le contenu HTML à l'aide de Beautiful Soup pour trouver les éléments contenant les données souhaitées.
   * Extrait les données et les formate comme vous le souhaitez.
   * Envoie les données par email.

**Script Python**

Il suppose que le prix de l'or est contenu dans une balise <span> avec une classe spécifique. **Vous devrez adapter ce script au site web que vous ciblez en modifiant les sélecteurs CSS.**

Python

import requests

from bs4 import BeautifulSoup

import smtplib

from email.mime.text import MIMEText

def send\_email(price):

# Remplacez par vos informations d'authentification SMTP

sender\_email = "votre\_email@exemple.com"

receiver\_email = "votre\_email@exemple.com"

password = "votre\_mot\_de\_passe"

message = MIMEText(f"Le prix de l'or est actuellement : {price}")

message['From'] = sender\_email

message['To'] = receiver\_email

message['Subject'] = "Prix de l'or"

with smtplib.SMTP('smtp.gmail.com', 587) as smtp:

smtp.starttls()

smtp.login(sender\_email, password)

smtp.sendmail(sender\_email, receiver\_email, message.as\_string())

def scrape\_gold\_price():

url = "https://www.votresiteweb.com/prix-or" # Remplacez par l'URL du site

response = requests.get(url)

soup = BeautifulSoup(response.content, 'html.parser')

# Adaptez ce sélecteur CSS à la structure de la page

price\_element = soup.find('span', class\_='prix-or')

if price\_element:

price = price\_element.text.strip()

send\_email(price)

else:

print("Impossible de trouver le prix de l'or.")

if \_\_name\_\_ == "\_\_main\_\_":

scrape\_gold\_price()

**Points clés à retenir:**

* **Sélecteurs CSS:** Apprenez à utiliser les sélecteurs CSS pour cibler les éléments HTML spécifiques que vous souhaitez extraire.
* **Respect des robots.txt:** Assurez-vous de respecter le fichier robots.txt du site pour éviter d'être bloqué.
* **Éthique:** Utilisez le web scraping de manière responsable et évitez de surcharger les serveurs.

**Conseils supplémentaires:**

* **Utilisez un environnement virtuel:** Cela vous permettra d'isoler les dépendances de votre projet.
* **Gérez les exceptions:** Implémentez des blocs try-except pour gérer les erreurs potentielles.
* **Enregistrez les données:** Enregistrez les données extraites dans un fichier CSV ou une base de données pour une analyse ultérieure.
* **Planifiez l'exécution:** Utilisez des outils comme cron (sur Linux) ou Task Scheduler (sur Windows) pour exécuter votre script régulièrement.