

Selenium est utile car il permet d'automatiser des tâches répétitives sur un site web, sans que tu aies à faire ces actions manuellement. Par exemple, si tu veux traduire plusieurs phrases ou tester différentes fonctionnalités d'un site web, Selenium peut faire tout cela pour toi.

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time

# Phrase à traduire
phrase_fr = "je vais travailler dure "

# Configurer les options du navigateur Chrome pour le mode headless
(sans interface graphique)
chrome_options = Options()
chrome_options.add_argument("--headless") # Exécution sans interface
graphique
chrome_options.add_argument("--disable-gpu") # Désactiver
l'utilisation du GPU (recommandé en mode headless)
chrome_options.add_argument('--ignore-certificate-errors') # Ignorer
les erreurs SSL
chrome_options.add_argument('--ignore-ssl-errors')

# Spécifier le chemin de chromedriver via la classe Service
service = Service('C:/Windows/chromedriver.exe')

# Créer l'instance WebDriver avec le service et les options
driver = webdriver.Chrome(service=service, options=chrome_options)

# Aller sur le site de Google Traduction
driver.get("https://translate.google.com/")

# Attendre un peu pour que la page se charge
time.sleep(2)

# Sélectionner la zone de texte d'entrée (nouvelle méthode)
input_box = driver.find_element(By.XPATH,
'//textarea[@aria-label="Texte source"]')
input_box.clear()
input_box.send_keys(phrase_fr)
```

```

# Attendre que la traduction soit faite et que l'élément de sortie soit
visible
try:
    output_box = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.XPATH,
'//span[@jsname="W297wb"]'))
    )
    phrase_en = output_box.text
except Exception as e:
    print("Erreur : ", e)
    phrase_en = None


# Fermer le navigateur
driver.quit()

# Sauvegarder les phrases dans des fichiers texte
with open("Phrase_fr.txt", "w", encoding="utf-8") as file_fr:
    file_fr.write(phrase_fr)

if phrase_en:
    with open("Phrase_en.txt", "w", encoding="utf-8") as file_en:
        file_en.write(phrase_en)
    print("Traduction terminée. Les fichiers ont été générés.")
else:
    print("Aucune traduction disponible.")

```


python

 Copier le code

```
from selenium import webdriver
```

- **selenium** : C'est une bibliothèque qui permet d'automatiser les interactions avec des navigateurs web.
- **webdriver** : C'est l'outil principal de Selenium, qui permet de contrôler un navigateur.

python

 Copier le code

```
from selenium.webdriver.chrome.options import Options
```

- **Options** : Utilisé pour configurer des options spécifiques pour Chrome, comme l'exécution en mode "headless" (sans interface graphique).

```
from selenium.webdriver.common.by import By
```

- **By** : Utilisé pour spécifier comment Selenium doit localiser les éléments dans une page web (par exemple, via `XPATH`, `ID`, `NAME`, etc.).

```
from selenium.webdriver.support.ui import WebDriverWait  
from selenium.webdriver.support import expected_conditions as EC
```

- **WebDriverWait** : Utilisé pour attendre qu'un élément devienne disponible sur une page.
- **EC (Expected Conditions)** : Fournit des conditions pour vérifier la présence d'éléments sur la page, comme attendre qu'un élément soit visible.

```
import time
```

- **time** : Ce module permet d'utiliser des fonctions liées au temps, comme `sleep()`, pour faire des pauses dans l'exécution du script.

```
phrase_fr = "je vais travailler dure "
```

- Vous définissez la phrase en français qui sera traduite en anglais.

```
chrome_options = Options()
chrome_options.add_argument("--headless")
chrome_options.add_argument("--disable-gpu")
chrome_options.add_argument('--ignore-certificate-errors')
chrome_options.add_argument('--ignore-ssl-errors')
```

- `Options()` : Crée un objet pour stocker des paramètres du navigateur.
- `--headless` : Permet d'exécuter Chrome sans interface visuelle, pratique pour l'automatisation.
- `--disable-gpu` : Désactive l'utilisation de la carte graphique, recommandé pour l'exécution sans interface graphique.
- `--ignore-certificate-errors` et `--ignore-ssl-errors` : Ces options permettent d'ignorer les erreurs SSL (liées aux certificats de sécurité), ce qui peut être utile pour éviter des blocages sur certaines pages.

```
service = Service('C:/Windows/chromedriver.exe')
```

Ici, vous spécifiez le chemin de `chromedriver.exe`, qui permet de contrôler Chrome via Selenium.

```
driver = webdriver.Chrome(service=service, options=chrome_options)
```

- `driver` : C'est l'objet WebDriver qui contrôle le navigateur Chrome.
- `service=service` : Utilise le chemin du service spécifié plus tôt.
- `options=chrome_options` : Applique les options définies précédemment (comme l'exécution en mode headless).

```
driver.get("https://translate.google.com/")
```

- Cela charge la page de Google Traduction dans le navigateur contrôlé par Selenium.

```
time.sleep(2)
```

- Ici, vous faites une pause de 2 secondes pour permettre au contenu de la page de se charger complètement.

```
input_box = driver.find_element(By.XPATH, '//textarea[@aria-label="Texte source"]')
input_box.clear()
input_box.send_keys(phrase_fr)
```


`find_element(By.XPATH, ...)` : Localise la zone de texte sur la page en utilisant un **XPATH**, qui est une façon de décrire où se trouve un élément dans une page web.

`clear()` : Efface tout texte qui pourrait être dans la zone de texte.

`send_keys(phrase_fr)` : Tape le texte français dans la zone de texte pour que Google Traduction puisse le traduire.



```
try:
    output_box = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.XPATH, '//span[@jsname="w297wb"]'))
    )
    phrase_en = output_box.text
except Exception as e:
    print("Erreur : ", e)
    phrase_en = None
```

 Copier le code

- `WebDriverWait(driver, 10)` : Attente maximale de 10 secondes pour que l'élément contenant la traduction soit disponible.
- `presence_of_element_located` : Cette condition vérifie que l'élément (ici la traduction) est bien présent sur la page.
- `output_box.text` : Récupère le texte traduit (la phrase en anglais).
- Si une erreur se produit (par exemple, si l'élément n'est pas trouvé), elle est capturée par le `try-except`.



```
with open("Phrase_fr.txt", "w", encoding="utf-8") as file_fr:
    file_fr.write(phrase_fr)
```

`open("Phrase_fr.txt", "w", encoding="utf-8")` : Ouvre un fichier texte nommé `Phrase_fr.txt` en mode écriture. Le fichier sera encodé en UTF-8 (pour bien gérer les accents).

`file_fr.write(phrase_fr)` : Écrit la phrase française dans ce fichier.

```

if phrase_en:
    with open("Phrase_en.txt", "w", encoding="utf-8") as file_en:
        file_en.write(phrase_en)
    print("Traduction terminée. Les fichiers ont été générés.")
else:
    print("Aucune traduction disponible.")

```

- Si une traduction a été récupérée, elle est enregistrée dans un fichier `Phrase_en.txt`. Si aucune traduction n'est disponible, un message l'indique.

Ce code utilise **Selenium** pour automatiser l'interaction avec le site de **Google Traduction**. Voici comment il fonctionne :

1. **Charger Google Traduction** : Le script ouvre le site de Google Traduction dans un navigateur Chrome sans interface graphique (mode "headless").
2. **Envoyer la phrase à traduire** : La phrase en français définie dans le script est automatiquement insérée dans la zone de texte de Google Traduction.
3. **Récupérer la traduction** : Le script attend que la traduction en anglais soit générée par le site, puis extrait ce texte.
4. **Sauvegarder les phrases** : La phrase en français et sa traduction en anglais (si disponible) sont sauvegardées dans deux fichiers texte (`Phrase_fr.txt` et `Phrase_en.txt`).
5. **Afficher des messages** : Si la traduction est réussie, le script affiche un message confirmant la génération des fichiers. Sinon, il affiche un message d'erreur.

## Ce que vous devez installer

**Selenium** : Installez la bibliothèque Selenium en exécutant la commande suivante :

bash

Copier le code

```
pip install selenium
```

- 1.
2. **Chrome** : Vous devez avoir Google Chrome installé sur votre machine.
3. **ChromeDriver** : Téléchargez la version de **ChromeDriver** correspondant à la version de Chrome installée sur votre machine. Assurez-vous que le fichier `chromedriver.exe` est placé dans le bon répertoire (ex. : `C:/Windows/`), et que vous spécifiez correctement le chemin dans le code.
  - Vous pouvez télécharger ChromeDriver ici : <https://sites.google.com/chromium.org/driver/>