

Introducción al Web Scraping



¿Qué es web scraping



El web scraping es una técnica utilizada para extraer información de sitios web de manera automatizada. Esta técnica implica el uso de programas o scripts que acceden a páginas web, recuperan su contenido y extraen datos específicos de ellas. El web scraping se utiliza comúnmente en diversas aplicaciones, como la recopilación de datos para análisis de mercado, la monitorización de precios, la agregación de noticias, la recopilación de datos para investigaciones académicas, entre otros.

¿Cómo funciona el web scraping?

1. **Enviar una solicitud HTTP:** El scraper envía una solicitud a la URL del sitio web del que desea extraer datos.
2. **Recuperar la respuesta:** El servidor del sitio web responde con el contenido de la página, que generalmente está en formato HTML.
3. **Parsear el HTML:** El contenido de la página web es analizado para identificar y extraer los datos relevantes. Esto se hace utilizando diversas bibliotecas de programación, como BeautifulSoup en Python.
4. **Extraer los datos:** Los datos específicos son extraídos de los elementos HTML correspondientes (como etiquetas, clases, identificadores, etc.).
5. **Almacenar los datos:** Los datos extraídos se almacenan en un formato estructurado, como bases de datos, hojas de cálculo o archivos CSV, para su posterior análisis o uso.

Herramientas comunes para web scraping

- **BeautifulSoup:** Una biblioteca en Python para analizar documentos HTML y XML.
- **Scrapy:** Un framework de scraping en Python muy potente y flexible.
- **Selenium:** Una herramienta para la automatización de navegadores que también puede usarse para scraping dinámico.
- **Puppeteer:** Una biblioteca en Node.js que proporciona una API de alto nivel para controlar Chrome o Chromium.

Consideraciones legales y éticas

Es importante tener en cuenta las implicaciones legales y éticas del web scraping. Algunos sitios web prohíben explícitamente el scraping en sus términos de servicio, y el scraping sin permiso puede violar leyes de derechos de autor o de protección de datos. Además, es recomendable respetar el

archivo `robots.txt` de un sitio web, que indica qué partes del sitio están permitidas o prohibidas para los robots de scraping.

Ejemplo básico en Python usando BeautifulSoup

python

```
import requests
from bs4 import BeautifulSoup

# URL del sitio web a scrapear
url = 'http://ejemplo.com'

# Enviar solicitud HTTP a la URL
response = requests.get(url)

# Parsear el contenido HTML
soup = BeautifulSoup(response.content, 'html.parser')

# Extraer datos específicos (por ejemplo, todos los títulos <h2>)
titulos = soup.find_all('h2')
for titulo in titulos:
    print(titulo.text)
```

Este ejemplo básico muestra cómo enviar una solicitud a una página web, parsear el contenido HTML y extraer todos los títulos `<h2>` de la página.



```
Tengo el siguiente ejemplo: from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.chrome.service import Service

from selenium.webdriver.chrome.options import Options

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected_conditions as EC

from webdriver_manager.chrome import ChromeDriverManager

# Configuración de opciones para el navegador

chrome_options = Options()

chrome_options.add_argument("--start-maximized") # Abre el navegador en pantalla completa
```

```
# Configuración del controlador

service = Service(ChromeDriverManager().install())

# Inicializa el navegador

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

try:

    # Paso 1: Abre Wikipedia

    driver.get("https://es.wikipedia.org/wiki/Wikipedia:Portada")

    # Espera a que el cuadro de búsqueda esté presente

    search_box = WebDriverWait(driver, 10).until(

        EC.presence_of_element_located((By.ID, "searchInput"))

    )

    search_box.send_keys("TecNM")

    search_box.submit()

    # Espera a que aparezcan los resultados

    pachuca_link = WebDriverWait(driver, 10).until(

        EC.presence_of_element_located((By.PARTIAL_LINK_TEXT, "Instituto Tecnológico de Pachuca"))

    )

    pachuca_link.click()

    # Espera a que la página del Instituto Tecnológico de Pachuca se cargue

    oferta_academica_link = WebDriverWait(driver, 10).until(
```

```

        EC.presence_of_element_located((By.LINK_TEXT, "Oferta académica"))
    )

    oferta_academica_link.click()

finally:

    # Mantén el navegador abierto

    input("Presiona Enter para cerrar el navegador...")

    driver.quit()

```



Este código usa Selenium para automatizar el navegador web y navegar a través de Wikipedia. A continuación, te explico paso a paso qué hace el código:

1. Importación de librerías y configuración de opciones del navegador:

python

```

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

```

2. Configuración de opciones para el navegador Chrome:

python

```

chrome_options = Options()
chrome_options.add_argument("--start-maximized") # Abre el navegador en pantalla
completa

```

3. Configuración del servicio para el controlador de Chrome usando `webdriver_manager` para gestionar la instalación del controlador:

```
python
```

```
service = Service(ChromeDriverManager().install())
```

4. Inicialización del navegador:

```
python
```

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

5. Uso de un bloque `try` para gestionar la navegación y manipulación del navegador:

```
python
```

```
try:
    # Paso 1: Abre Wikipedia
    driver.get("https://es.wikipedia.org/wiki/Wikipedia:Portada")

    # Espera a que el cuadro de búsqueda esté presente
    search_box = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, "searchInput"))
    )

    # Introduce el texto "TecNM" en el cuadro de búsqueda y envía la búsqueda
    search_box.send_keys("TecNM")
    search_box.submit()

    # Espera a que aparezcan los resultados y haz clic en el enlace correspondiente
    pachuca_link = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.PARTIAL_LINK_TEXT, "Instituto Tecnológico
de Pachuca")))
    pachuca_link.click()

    # Espera a que la página del Instituto Tecnológico de Pachuca se cargue y haz
clic en "Oferta académica"
    oferta_academica_link = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.LINK_TEXT, "Oferta académica")))
    oferta_academica_link.click()

finally:
    # Mantén el navegador abierto hasta que el usuario presione Enter
    input("Presiona Enter para cerrar el navegador...")
    driver.quit()
```

Explicación de cada paso

- 1. Importaciones y configuraciones:** Importa todas las librerías necesarias y configura el navegador Chrome para abrirse en modo pantalla completa.
- 2. Inicialización del controlador y el navegador:** Usa `ChromeDriverManager` para instalar el controlador de Chrome y crea una instancia del navegador con las opciones especificadas.

3. **Abrir Wikipedia y buscar "TecNM":** Navega a la portada de Wikipedia y espera a que el cuadro de búsqueda esté presente. Una vez que está presente, introduce "TecNM" y envía la búsqueda.
4. **Navegar a la página del Instituto Tecnológico de Pachuca:** Espera a que aparezcan los resultados de búsqueda y haz clic en el enlace parcial que contiene "Instituto Tecnológico de Pachuca".
5. **Acceder a la sección "Oferta académica":** Una vez en la página del Instituto, espera a que el enlace "Oferta académica" esté presente y haz clic en él.
6. **Mantener el navegador abierto:** El navegador se mantiene abierto hasta que el usuario presiona Enter en la consola. Luego, el navegador se cierra limpiamente con `driver.quit()`.

Este ejemplo muestra cómo automatizar la navegación y la interacción con elementos de una página web utilizando Selenium.



¿me puedes indicar el prompt que generaron para que hiciera esos pasos?



Claro, el prompt que describe las acciones que deseas que el script de Selenium realice puede ser algo como lo siguiente:

1. **Configurar el navegador:**
 - Usa el navegador Chrome.
 - Abre el navegador en pantalla completa.
2. **Navegación inicial:**
 - Abre la página principal de Wikipedia en español:
`https://es.wikipedia.org/wiki/Wikipedia:Portada`.
3. **Búsqueda en Wikipedia:**
 - Espera a que aparezca el cuadro de búsqueda.
 - Introduce el término de búsqueda "TecNM" en el cuadro de búsqueda.
 - Envía la búsqueda.
4. **Interacción con los resultados de búsqueda:**
 - Espera a que aparezcan los resultados de búsqueda.
 - Encuentra y haz clic en el enlace parcial que contiene el texto "Instituto Tecnológico de Pachuca".
5. **Navegación en la página del Instituto Tecnológico de Pachuca:**
 - Espera a que la página del Instituto Tecnológico de Pachuca se cargue.
 - Encuentra y haz clic en el enlace "Oferta académica".
6. **Mantener el navegador abierto:**
 - Mantén el navegador abierto hasta que el usuario presione Enter en la consola.
 - Cierra el navegador después de que el usuario presione Enter.

Prompt para Generar el Script de Selenium:

plaintext

Configura un script en Python usando Selenium que realice las siguientes acciones:

1. Configura el navegador Chrome para que se abra en pantalla completa.
2. Abre la página principal de Wikipedia en español:
`https://es.wikipedia.org/wiki/Wikipedia:Portada`.
3. Espera a que aparezca el cuadro de búsqueda en la página.
4. Introduce el término de búsqueda "TecNM" en el cuadro de búsqueda.
5. Envía la búsqueda.
6. Espera a que aparezcan los resultados de búsqueda.
7. Encuentra y haz clic en el enlace parcial que contiene el texto "Instituto Tecnológico de Pachuca".
8. Espera a que la página del Instituto Tecnológico de Pachuca se cargue.
9. Encuentra y haz clic en el enlace "Oferta académica".
10. Mantén el navegador abierto hasta que el usuario presione Enter en la consola.
11. Cierra el navegador después de que el usuario presione Enter.

Asegúrate de manejar todas las esperas necesarias para que los elementos estén presentes antes de interactuar con ellos. Usa `WebDriverWait` y `ExpectedConditions` para las esperas.

Este prompt proporciona una descripción clara y detallada de los pasos que el script debe realizar, asegurando que cada acción necesaria se maneje adecuadamente con esperas explícitas para los elementos de la página.



¿cómo sería el prompt que debería redactar para que me devolvieras un resultado similar?



Para obtener un script de Selenium que realice las acciones descritas, podrías redactar un prompt claro y detallado que describa exactamente lo que necesitas. Aquí tienes un ejemplo de cómo podrías redactarlo:

plaintext

Necesito un script en Python que utilice Selenium para automatizar las siguientes acciones:

1. Configura el navegador Chrome con las siguientes opciones:
 - Abre el navegador en modo pantalla completa.
2. Abre la página principal de Wikipedia en español:
``https://es.wikipedia.org/wiki/Wikipedia:Portada``.
3. Espera hasta que el cuadro de búsqueda esté presente en la página.
4. Introduce el término "TecNM" en el cuadro de búsqueda y envía la búsqueda.
5. Espera a que los resultados de búsqueda aparezcan.
6. Haz clic en el enlace parcial que contiene el texto "Instituto Tecnológico de Pachuca".

7. Espera a que la página del Instituto Tecnológico de Pachuca se cargue por completo.
8. Haz clic en el enlace que dice "Oferta académica".
9. Mantén el navegador abierto hasta que el usuario presione Enter en la consola.
10. Una vez que el usuario presione Enter, cierra el navegador.

Asegúrate de utilizar `WebDriverWait` y `ExpectedConditions` para manejar las esperas de manera adecuada, asegurándote de que los elementos estén presentes antes de interactuar con ellos.

Proporcióname el código completo que realice estas acciones.

Este prompt está diseñado para ser claro y específico, proporcionando todos los detalles necesarios para que pueda generar el script exacto que necesitas.



Me podrías mostrar otro ejemplo realizando otras acciones con otra u otras páginas



¡Claro! Aquí tienes un ejemplo de un script de Selenium que automatiza la navegación y la interacción con la página de Amazon. Este script buscará un producto, hará clic en el primer resultado y añadirá el producto al carrito.

Prompt

plaintext

Necesito un script en Python que utilice Selenium para automatizar las siguientes acciones:

1. Configura el navegador Chrome con las siguientes opciones:
 - Abre el navegador en modo pantalla completa.
2. Abre la página principal de Amazon: `https://www.amazon.com`.
3. Espera hasta que el cuadro de búsqueda esté presente en la página.
4. Introduce el término "laptop" en el cuadro de búsqueda y envía la búsqueda.
5. Espera a que los resultados de búsqueda aparezcan.
6. Haz clic en el primer resultado de la lista.
7. Espera a que la página del producto se cargue por completo.
8. Añade el producto al carrito.
9. Mantén el navegador abierto hasta que el usuario presione Enter en la consola.
10. Una vez que el usuario presione Enter, cierra el navegador.

Asegúrate de utilizar `WebDriverWait` y `ExpectedConditions` para manejar las esperas de manera adecuada, asegurándote de que los elementos estén presentes antes de interactuar con ellos.

Proporcióname el código completo que realice estas acciones.

Código Completo

python

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

# Configuración de opciones para el navegador
chrome_options = Options()
chrome_options.add_argument("--start-maximized") # Abre el navegador en pantalla completa

# Configuración del controlador
service = Service(ChromeDriverManager().install())

# Inicializa el navegador
driver = webdriver.Chrome(service=service, options=chrome_options)

try:
    # Paso 1: Abre Amazon
    driver.get("https://www.amazon.com")

    # Espera a que el cuadro de búsqueda esté presente
    search_box = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, "twotabsearchtextbox"))
    )

    # Introduce el término "laptop" en el cuadro de búsqueda y envía la búsqueda
    search_box.send_keys("laptop")
    search_box.submit()

    # Espera a que aparezcan los resultados
    first_result = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.CSS_SELECTOR, ".s-main-slot .s-result-item"))
    )

    # Haz clic en el primer resultado de la lista
    first_result.click()

    # Espera a que la página del producto se cargue por completo
    add_to_cart_button = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, "add-to-cart-button"))
    )

    # Añade el producto al carrito
    add_to_cart_button.click()

finally:
    # Mantén el navegador abierto
    input("Presiona Enter para cerrar el navegador...")
    driver.quit()
```

Explicación de cada paso

1. **Importaciones y configuraciones:** Importa todas las librerías necesarias y configura el navegador Chrome para abrirse en modo pantalla completa.
2. **Inicialización del controlador y el navegador:** Usa `ChromeDriverManager` para instalar el controlador de Chrome y crea una instancia del navegador con las opciones especificadas.
3. **Abrir Amazon y buscar "laptop":** Navega a la página principal de Amazon y espera a que el cuadro de búsqueda esté presente. Una vez que está presente, introduce "laptop" y envía la búsqueda.
4. **Navegar a la página del primer resultado:** Espera a que aparezcan los resultados de búsqueda y haz clic en el primer resultado de la lista.
5. **Añadir el producto al carrito:** Espera a que la página del producto se cargue completamente y haz clic en el botón "Añadir al carrito".
6. **Mantener el navegador abierto:** El navegador se mantiene abierto hasta que el usuario presiona Enter en la consola. Luego, el navegador se cierra limpiamente con `driver.quit()`.

Este ejemplo muestra cómo automatizar la navegación y la interacción con elementos de la página de Amazon utilizando Selenium.



Podrías indicarme la solución para ese prompt



¡Claro! Aquí te dejo la solución completa para el prompt mencionado, utilizando Selenium para automatizar la navegación en Amazon y realizar las acciones especificadas:

Prompt

plaintext

Necesito un script en Python que utilice Selenium para automatizar las siguientes acciones:

1. Configura el navegador Chrome con las siguientes opciones:
 - Abre el navegador en modo pantalla completa.
2. Abre la página principal de Amazon: `https://www.amazon.com`.`
3. Espera hasta que el cuadro de búsqueda esté presente en la página.
4. Introduce el término "laptop" en el cuadro de búsqueda y envía la búsqueda.
5. Espera a que los resultados de búsqueda aparezcan.
6. Haz clic en el primer resultado de la lista.
7. Espera a que la página del producto se cargue por completo.

8. Añade el producto al carrito.
 9. Mantén el navegador abierto hasta que el usuario presione Enter en la consola.
 10. Una vez que el usuario presione Enter, cierra el navegador.
- Asegúrate de utilizar `WebDriverWait` y `ExpectedConditions` para manejar las esperas de manera adecuada, asegurándote de que los elementos estén presentes antes de interactuar con ellos.
- Proporcióname el código completo que realice estas acciones.

Código Completo

python

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

# Configuración de opciones para el navegador
chrome_options = Options()
chrome_options.add_argument("--start-maximized") # Abre el navegador en pantalla completa

# Configuración del controlador
service = Service(ChromeDriverManager().install())

# Inicializa el navegador
driver = webdriver.Chrome(service=service, options=chrome_options)

try:
    # Paso 1: Abre Amazon
    driver.get("https://www.amazon.com")

    # Espera a que el cuadro de búsqueda esté presente
    search_box = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, "twotabsearchtextbox"))
    )

    # Introduce el término "laptop" en el cuadro de búsqueda y envía la búsqueda
    search_box.send_keys("laptop")
    search_box.submit()

    # Espera a que aparezcan los resultados
    first_result = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.CSS_SELECTOR, ".s-main-slot .s-result-item"))
    )

    # Haz clic en el primer resultado de la lista
    first_result.find_element(By.TAG_NAME, "h2").click()

    # Espera a que la página del producto se cargue por completo
    add_to_cart_button = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, "add-to-cart-button"))
    )

    # Añade el producto al carrito
    add_to_cart_button.click()
```

```
finally:
    # Mantén el navegador abierto
    input("Presiona Enter para cerrar el navegador...")
    driver.quit()
```

Explicación de cada paso

- 1. Importaciones y configuraciones:** Importa todas las librerías necesarias y configura el navegador Chrome para abrirse en modo pantalla completa.
- 2. Inicialización del controlador y el navegador:** Usa `ChromeDriverManager` para instalar el controlador de Chrome y crea una instancia del navegador con las opciones especificadas.
- 3. Abrir Amazon y buscar "laptop":** Navega a la página principal de Amazon y espera a que el cuadro de búsqueda esté presente. Una vez que esté presente, introduce "laptop" y envía la búsqueda.
- 4. Navegar a la página del primer resultado:** Espera a que aparezcan los resultados de búsqueda y haz clic en el primer resultado de la lista.
- 5. Añadir el producto al carrito:** Espera a que la página del producto se cargue completamente y haz clic en el botón "Añadir al carrito".
- 6. Mantener el navegador abierto:** El navegador se mantiene abierto hasta que el usuario presiona Enter en la consola. Luego, el navegador se cierra limpiamente con `driver.quit()`.

Este ejemplo demuestra cómo automatizar la navegación y la interacción con elementos de la página de Amazon utilizando Selenium.



Me puedes mostrar otro ejemplo con una página que no tenga validación para confirmar que no soy un robot



Claro, vamos a utilizar una página que no tenga validación de CAPTCHA para el ejemplo. Vamos a hacer un script de Selenium que navegue por un sitio de noticias y recoja los titulares de la primera página.

Prompt

plaintext

Necesito un script en Python que utilice Selenium para automatizar las siguientes acciones:

1. Configura el navegador Chrome con las siguientes opciones:
 - Abre el navegador en modo pantalla completa.

2. Abre la página principal de un sitio de noticias, como `https://www.bbc.com`.
 3. Espera hasta que los titulares principales estén presentes en la página.
 4. Extrae los titulares principales de la página de inicio.
 5. Imprime los titulares en la consola.
 6. Mantén el navegador abierto hasta que el usuario presione Enter en la consola.
 7. Una vez que el usuario presione Enter, cierra el navegador.
- Asegúrate de utilizar `WebDriverWait` y `ExpectedConditions` para manejar las esperas de manera adecuada, asegurándote de que los elementos estén presentes antes de interactuar con ellos.
- Proporcióname el código completo que realice estas acciones.

Código Completo

python

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

# Configuración de opciones para el navegador
chrome_options = Options()
chrome_options.add_argument("--start-maximized") # Abre el navegador en pantalla completa

# Configuración del controlador
service = Service(ChromeDriverManager().install())

# Inicializa el navegador
driver = webdriver.Chrome(service=service, options=chrome_options)

try:
    # Paso 1: Abre BBC News
    driver.get("https://www.bbc.com")

    # Espera a que los titulares principales estén presentes
    headlines = WebDriverWait(driver, 10).until(
        EC.presence_of_all_elements_located((By.CSS_SELECTOR, ".gs-c-promo-
heading__title"))
    )

    # Extrae y muestra los titulares principales
    for headline in headlines:
        print(headline.text)

finally:
    # Mantén el navegador abierto
    input("Presiona Enter para cerrar el navegador...")
    driver.quit()
```

Explicación de cada paso

1. **Importaciones y configuraciones:** Importa las librerías necesarias y configura el navegador Chrome para abrirse en modo pantalla completa.
2. **Inicialización del controlador y el navegador:** Usa `ChromeDriverManager` para instalar el controlador de Chrome y crea una instancia del navegador con las opciones especificadas.
3. **Abrir BBC News y extraer titulares:** Navega a la página principal de BBC News y espera a que los titulares principales estén presentes. Luego, extrae los titulares usando el selector CSS para los títulos de las noticias.
4. **Imprimir los titulares:** Imprime cada titular en la consola.
5. **Mantener el navegador abierto:** El navegador se mantiene abierto hasta que el usuario presiona Enter en la consola. Luego, el navegador se cierra limpiamente con `driver.quit()`.

Este ejemplo muestra cómo automatizar la navegación y la extracción de datos de una página de noticias utilizando Selenium.



Puede ser algun otro ejemplo que no me pida instalar más librerías?



¡Claro! Aquí tienes un ejemplo de cómo usar Selenium con un sitio web que no requiere CAPTCHAs y sin necesidad de instalar librerías adicionales. Vamos a utilizar un sitio web simple como `http://quotes.toscrape.com`, que está diseñado específicamente para prácticas de web scraping.

Prompt

plaintext

Necesito un script en Python que utilice Selenium para automatizar las siguientes acciones:

1. Configura el navegador Chrome con las siguientes opciones:
 - Abre el navegador en modo pantalla completa.
2. Abre la página principal de `http://quotes.toscrape.com`.
3. Espera hasta que las citas estén presentes en la página.
4. Extrae las citas y los autores de la página de inicio.
5. Imprime las citas y los autores en la consola.
6. Mantén el navegador abierto hasta que el usuario presione Enter en la consola.
7. Una vez que el usuario presione Enter, cierra el navegador.

Asegúrate de utilizar `WebDriverWait` y `ExpectedConditions` para manejar las esperas de manera adecuada, asegurándote de que los elementos estén presentes antes de interactuar con ellos.

Proporcióname el código completo que realice estas acciones.

Código Completo

python

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

# Configuración de opciones para el navegador
chrome_options = Options()
chrome_options.add_argument("--start-maximized") # Abre el navegador en pantalla completa

# Configuración del controlador
service = Service(ChromeDriverManager().install())

# Inicializa el navegador
driver = webdriver.Chrome(service=service, options=chrome_options)

try:
    # Paso 1: Abre la página de citas
    driver.get("http://quotes.toscrape.com")

    # Espera a que las citas estén presentes en la página
    quotes = WebDriverWait(driver, 10).until(
        EC.presence_of_all_elements_located((By.CLASS_NAME, "quote"))
    )

    # Extrae y muestra las citas y los autores
    for quote in quotes:
        text = quote.find_element(By.CLASS_NAME, "text").text
        author = quote.find_element(By.CLASS_NAME, "author").text
        print(f"Cita: {text}\nAutor: {author}\n")

finally:
    # Mantén el navegador abierto
    input("Presiona Enter para cerrar el navegador...")
    driver.quit()
```

Explicación de cada paso

- 1. Importaciones y configuraciones:** Importa las librerías necesarias para utilizar Selenium y configura el navegador Chrome para abrirse en modo pantalla completa.
- 2. Inicialización del controlador y el navegador:** Usa `ChromeDriverManager` para instalar el controlador de Chrome y crea una instancia del navegador con las opciones especificadas.
- 3. Abrir el sitio de citas y extraer citas:** Navega a la página principal de `http://quotes.toscrape.com` y espera a que las citas estén presentes. Luego, extrae las citas y los autores usando los selectores de clase.
- 4. Imprimir citas y autores:** Imprime cada cita y el autor correspondiente en la consola.

5. **Mantener el navegador abierto:** El navegador se mantiene abierto hasta que el usuario presiona Enter en la consola. Luego, el navegador se cierra limpiamente con `driver.quit()`.

Este ejemplo es sencillo y utiliza un sitio web diseñado específicamente para la práctica de scraping, por lo que no requiere librerías adicionales y es ideal para experimentar con Selenium.



otro ejemplo más útil?



Claro, vamos a utilizar un ejemplo práctico para extraer datos de una página de ofertas de empleo, que puede ser útil si estás buscando empleo o investigando oportunidades laborales. Utilizaremos el sitio web de ofertas de empleo `https://www.indeed.com` para buscar ofertas de trabajo relacionadas con "software engineer".

Prompt

plaintext

Necesito un script en Python que utilice Selenium para automatizar las siguientes acciones:

1. Configura el navegador Chrome con las siguientes opciones:
 - Abre el navegador en modo pantalla completa.
2. Abre la página principal de Indeed: `https://www.indeed.com`.
3. Espera hasta que el campo de búsqueda y el botón de búsqueda estén presentes.
4. Introduce el término "software engineer" en el campo de búsqueda y haz clic en el botón de búsqueda.
5. Espera hasta que los resultados de búsqueda estén presentes en la página.
6. Extrae los títulos y las ubicaciones de las ofertas de empleo de la primera página de resultados.
7. Imprime los títulos y las ubicaciones en la consola.
8. Mantén el navegador abierto hasta que el usuario presione Enter en la consola.
9. Una vez que el usuario presione Enter, cierra el navegador.

Asegúrate de utilizar `WebDriverWait` y `ExpectedConditions` para manejar las esperas de manera adecuada, asegurándote de que los elementos estén presentes antes de interactuar con ellos.

Proporcióname el código completo que realice estas acciones.

Código Completo

python

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

# Configuración de opciones para el navegador
chrome_options = Options()
chrome_options.add_argument("--start-maximized") # Abre el navegador en pantalla completa

# Configuración del controlador
service = Service(ChromeDriverManager().install())

# Inicializa el navegador
driver = webdriver.Chrome(service=service, options=chrome_options)

try:
    # Paso 1: Abre Indeed
    driver.get("https://www.indeed.com")

    # Espera a que el campo de búsqueda y el botón de búsqueda estén presentes
    search_box = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, "text-input-what"))
    )
    location_box = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, "text-input-where"))
    )
    search_button = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.CSS_SELECTOR, "button[type='submit']"))
    )

    # Introduce el término "software engineer" y un lugar vacío (se busca en todos los
    # lugares) y haz clic en el botón de búsqueda
    search_box.send_keys("software engineer")
    location_box.clear() # Limpia el campo de ubicación
    search_button.click()

    # Espera hasta que los resultados de búsqueda estén presentes
    job_titles = WebDriverWait(driver, 10).until(
        EC.presence_of_all_elements_located((By.CSS_SELECTOR, ".jobTitle"))
    )
    job_locations = WebDriverWait(driver, 10).until(
        EC.presence_of_all_elements_located((By.CSS_SELECTOR, ".companyLocation"))
    )

    # Extrae y muestra los títulos y las ubicaciones de las ofertas de empleo
    for title, location in zip(job_titles, job_locations):
        print(f"Título: {title.text}")
        print(f"Ubicación: {location.text}")
        print("-" * 40)

finally:
    # Mantén el navegador abierto
    input("Presiona Enter para cerrar el navegador...")
    driver.quit()
```

Explicación de cada paso

1. **Importaciones y configuraciones:** Importa las librerías necesarias para usar Selenium y configura el navegador Chrome para abrirse en modo pantalla completa.
2. **Inicialización del controlador y el navegador:** Usa `ChromeDriverManager`` para instalar el controlador de Chrome y crea una instancia del navegador con las opciones especificadas.
3. **Abrir Indeed y buscar "software engineer":** Navega a la página principal de Indeed y espera a que el campo de búsqueda y el botón de búsqueda estén presentes. Luego, introduce "software engineer" en el campo de búsqueda y haz clic en el botón de búsqueda.
4. **Extraer títulos y ubicaciones:** Espera a que los resultados de búsqueda aparezcan y extrae los títulos y las ubicaciones de las ofertas de empleo. Imprime cada título y ubicación en la consola.
5. **Mantener el navegador abierto:** El navegador se mantiene abierto hasta que el usuario presiona Enter en la consola. Luego, el navegador se cierra limpiamente con `driver.quit()`.

Este ejemplo es útil si estás buscando empleo o investigando oportunidades laborales y te permite extraer información relevante de un sitio de ofertas de empleo de manera automatizada.



Hay otro ejemplo con más acciones



Por supuesto, aquí tienes un ejemplo que realiza varias acciones en un sitio web de comercio electrónico. Usaremos el sitio de ejemplo `https://www.demoshop.com``, donde realizaremos una búsqueda de productos, filtraremos los resultados, y agregaremos un producto al carrito.

Prompt

plaintext

Necesito un script en Python que utilice Selenium para automatizar las siguientes acciones:

1. Configura el navegador Chrome con las siguientes opciones:
 - Abre el navegador en modo pantalla completa.
2. Abre la página principal de la tienda en línea: `https://www.demoshop.com``.
3. Espera hasta que el campo de búsqueda y el botón de búsqueda estén presentes.
4. Introduce el término "laptop" en el campo de búsqueda y haz clic en el botón de búsqueda.
5. Espera hasta que los resultados de búsqueda estén presentes.
6. Filtra los resultados por el filtro de "Categoría: Electrónica".
7. Selecciona el primer producto de la lista de resultados.
8. Espera a que la página del producto se cargue completamente.
9. Añade el producto al carrito.

10. Navega a la página del carrito.
 11. Espera hasta que el carrito esté visible y muestra el contenido del carrito en la consola.
 12. Mantén el navegador abierto hasta que el usuario presione Enter en la consola.
 13. Una vez que el usuario presione Enter, cierra el navegador.
- Asegúrate de utilizar `WebDriverWait` y `ExpectedConditions` para manejar las esperas de manera adecuada, asegurándote de que los elementos estén presentes antes de interactuar con ellos.
- Proporcióname el código completo que realice estas acciones.

Código Completo

python

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

# Configuración de opciones para el navegador
chrome_options = Options()
chrome_options.add_argument("--start-maximized") # Abre el navegador en pantalla completa

# Configuración del controlador
service = Service(ChromeDriverManager().install())

# Inicializa el navegador
driver = webdriver.Chrome(service=service, options=chrome_options)

try:
    # Paso 1: Abre la tienda en línea
    driver.get("https://www.demoshop.com")

    # Paso 2: Espera a que el campo de búsqueda y el botón de búsqueda estén presentes
    search_box = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, "search-input"))
    )
    search_button = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, "search-button"))
    )

    # Paso 3: Introduce el término "laptop" y haz clic en el botón de búsqueda
    search_box.send_keys("laptop")
    search_button.click()

    # Paso 4: Espera a que los resultados de búsqueda estén presentes
    WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.CSS_SELECTOR, ".product-list"))
    )

    # Paso 5: Filtra los resultados por "Categoría: Electrónica"
    filter_electronics = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.CSS_SELECTOR, "#filter-electronics"))
    )
```

```

filter_electronics.click()

# Paso 6: Selecciona el primer producto de la lista
first_product = WebDriverWait(driver, 10).until(
    EC.presence_of_element_located((By.CSS_SELECTOR, ".product-item:first-child"))
)
first_product.click()

# Paso 7: Espera a que la página del producto se cargue completamente
add_to_cart_button = WebDriverWait(driver, 10).until(
    EC.presence_of_element_located((By.ID, "add-to-cart"))
)

# Paso 8: Añade el producto al carrito
add_to_cart_button.click()

# Paso 9: Navega a la página del carrito
cart_button = WebDriverWait(driver, 10).until(
    EC.presence_of_element_located((By.ID, "cart-button"))
)
cart_button.click()

# Paso 10: Espera hasta que el carrito esté visible y muestra el contenido del carrito
cart_items = WebDriverWait(driver, 10).until(
    EC.presence_of_all_elements_located((By.CSS_SELECTOR, ".cart-item"))
)

for item in cart_items:
    product_name = item.find_element(By.CSS_SELECTOR, ".product-name").text
    product_quantity = item.find_element(By.CSS_SELECTOR, ".product-quantity").text
    print(f"Producto: {product_name}, Cantidad: {product_quantity}")

finally:
    # Mantén el navegador abierto
    input("Presiona Enter para cerrar el navegador...")
    driver.quit()

```

Explicación de cada paso

- 1. Importaciones y configuraciones:** Importa las librerías necesarias y configura el navegador Chrome para abrirse en modo pantalla completa.
- 2. Inicialización del controlador y el navegador:** Usa `ChromeDriverManager` para instalar el controlador de Chrome y crea una instancia del navegador con las opciones especificadas.
- 3. Abrir la tienda en línea y buscar "laptop":** Navega a la página principal de la tienda en línea y espera a que el campo de búsqueda y el botón de búsqueda estén presentes. Introduce "laptop" en el campo de búsqueda y haz clic en el botón de búsqueda.
- 4. Filtrar resultados y seleccionar producto:** Filtra los resultados por la categoría "Electrónica" y selecciona el primer producto de la lista de resultados.
- 5. Añadir producto al carrito y mostrar contenido:** Añade el producto al carrito y navega a la página del carrito. Muestra el contenido del carrito en la consola, incluyendo el nombre y la cantidad de cada producto.
- 6. Mantener el navegador abierto:** El navegador se mantiene abierto hasta que el usuario presiona Enter en la consola. Luego, el navegador se cierra limpiamente con `driver.quit()`.

Este ejemplo cubre varias acciones típicas que puedes realizar en un sitio de comercio electrónico y puede ser adaptado a diferentes páginas de tiendas en línea.



Necesito un script en Python que utilice Selenium para automatizar las siguientes acciones:

1. Configura el navegador Chrome con las siguientes opciones:
 - Abre el navegador en modo pantalla completa.
2. Abre la página principal del Instituto Tecnológico de Pachuca: `https://itp.itpachuca.edu.mx/``.
- 3.- Posiciona el cursor en el menú principal, y coloca el cursor en la opción que dice "Ingenierías" y desliza el cursor en la lista desplegable que muestra las distintas opciones y ejecuta
- 4.- Despliega el menú de Ingenierías e Ingresa por medio de un clic a la opción de Sistemas Computacionales y para que se muestre la información de Sistemas Computacionales
- 5.- Deslízate por la página con la barra desplazable hasta llegar al texto que dice Retícula 2021 y selecciona el hipervínculo para poder visualizar la información dando doble clic y abre en una nueva página la información que contiene el hipervínculo seleccionado y ejecuta.



Aquí tienes un script en Python utilizando Selenium que automatiza las acciones descritas en tu solicitud:

python

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

# Configuración de opciones para el navegador
chrome_options = Options()
chrome_options.add_argument("--start-maximized") # Abre el navegador en pantalla completa

# Configuración del controlador
service = Service(ChromeDriverManager().install())

# Inicializa el navegador
driver = webdriver.Chrome(service=service, options=chrome_options)

try:
    # Paso 1: Abre la página principal del Instituto Tecnológico de Pachuca
    driver.get("https://itp.itpachuca.edu.mx/")

    # Paso 2: Espera a que el menú principal esté presente
    ingenierias_menu = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.LINK_TEXT, "Ingenierías"))
    )
```

```
# Paso 3: Despliega el menú de Ingenierías y selecciona la opción "Sistemas
Computacionales"
action = ActionChains(driver)
action.move_to_element(ingenierias_menu).perform() # Coloca el cursor sobre el menú
de "Ingenierías"

sistemas_computacionales_option = WebDriverWait(driver, 10).until(
    EC.presence_of_element_located((By.LINK_TEXT, "Sistemas Computacionales"))
)

action.move_to_element(sistemas_computacionales_option).click().perform() #
Selecciona "Sistemas Computacionales"

# Paso 4: Deslízate hasta el texto que dice "Retícula 2021"
reticula_link = WebDriverWait(driver, 10).until(
    EC.presence_of_element_located((By.PARTIAL_LINK_TEXT, "Retícula 2021"))
)

driver.execute_script("arguments[0].scrollIntoView(true);", reticula_link) # Desplaza
la página hasta el enlace

# Paso 5: Doble clic en el enlace de "Retícula 2021" y abre en una nueva pestaña
action.double_click(reticula_link).perform()

# Cambia a la nueva pestaña
driver.switch_to.window(driver.window_handles[-1])

finally:
    # Mantén el navegador abierto
    input("Presiona Enter para cerrar el navegador...")
    driver.quit()
```

Explicación de cada paso:

- 1. Configuración e inicialización del navegador:** El script utiliza `Options` para abrir Chrome en modo pantalla completa y `Service` para manejar el controlador de Chrome. El navegador se inicializa y se dirige a la página principal del Instituto Tecnológico de Pachuca.
- 2. Interacción con el menú:** Se espera a que el menú de "Ingenierías" esté presente y luego se utiliza `ActionChains` para simular el movimiento del cursor sobre este menú. A continuación, selecciona la opción "Sistemas Computacionales" en el submenú.
- 3. Desplazamiento a la sección deseada:** El script busca el hipervínculo que contiene el texto "Retícula 2021" y desplaza la página hasta que el enlace sea visible.
- 4. Doble clic y nueva pestaña:** Realiza un doble clic sobre el enlace para abrirlo y cambia a la nueva pestaña donde se muestra la información seleccionada.

Este script automatiza completamente el proceso de navegación en el sitio web del Instituto Tecnológico de Pachuca para llegar a la información específica de "Retícula 2021" de la carrera de "Sistemas Computacionales".

ChatGPT puede cometer errores. Comprueba la información importante.