Introduzione a Playwright

Cosa imparerai

- Cos'é *PlayWright*
- Concetti di base
- Installazione e configurazione
- Concetti fondamentali
- *Playwright* waiting
- Registrazione di script
- Oggetti principali e funzioni

Introduzione a Playwright

Cos'é Playwright

Playwright è un framework di automazione browser

- open-source
- sviluppato da Microsoft.

Permette di automatizzare browser

- Chromium
- Firefox
- WebKit

È ideale per test end-to-end, scraping e automazione di task web.

Introduzione a Playwright

Installazione

Per installare Playwright con Python, segui questi passaggi:

1. **Assicurati di avere Python installato:** Playwright supporta Python 3.7+.
2. **Installa Playwright tramite pip:**

```
pip install playwright
```

3. **Installa i browser:** Playwright necessita dei browser per funzionare. Puoi installarli con il seguente comando:

```
playwright install
```

Introduzione a Playwright

Concetti Fondamentali

Di seguito gli oggetti principali utilizzati da *playwright*

- **`Browser`:** Rappresenta un'istanza del browser (Chromium, Firefox, WebKit).
- **`BrowserContext`:** Fornisce un ambiente di navigazione isolato all'interno di un browser. Utile per gestire sessioni multiple o simulare diversi utenti.
- **`Page`:** Rappresenta una singola tab o finestra nel browser.
- **`Playwright`**: L'oggetto di partenza principale. Si ottiene tramite `playwright = sync_playwright().start()`
- **`Locator`**: Rappresenta un modo per trovare un elemento sulla pagina. Si può ottenere tramite `page.locator()`, `page.get_by_role()`, `page.get_by_text()`, ecc. Permette auto-waiting e retry automatici.

Introduzione a Playwright

Concetti Fondamentali

Utilizzo di Xpath e Selector

I *Locator* sfruttano XPath e i selector per individuare elementi all'interno del *DOM* (Document Object Model) di una pagina web.

XPath

XPath (XML Path Language) è un linguaggio di query utilizzato per navigare attraverso gli elementi e gl attributi in un documento XML. Poiché HTML è un dialetto di XML, XPath può essere utilizzato anche pe navigare e selezionare elementi in pagine web.

I CSS selector

I CSS selector sono pattern utilizzati per selezionare elementi HTML in base al loro tipo, attributo o posizione nel documento. Sono comunemente usati per applicare stili CSS, ma sono altrettanto efficaci per individuare elementi in test automatizzati.

Introduzione a Playwright

Concetti Fondamentali

Utilizzo di Xpath e Selector

Di seguito alcuni esempi di **XPath**:

- `/html/body/div/p`: Seleziona tutti gli elementi `<p>` che sono figli di `<div>`, che a loro volta sono figli di `<body>` e `<html>`.
- `//p`: Seleziona tutti gli elementi `<p>` nel documento, indipendentemente dalla loro posizione.
- `//div[@id='contenitore']`: Seleziona l'elemento `<div>` con l'attributo `id` uguale a "contenitore".
- `//a[contains(@href, 'esempio.com')]`: Seleziona tutti gli elementi `<a>` il cui attributo `href` contiene la stringa "esempio.com".
- `//button[text()='Clicca qui']`: Seleziona tutti gli elementi `<button>` il cui testo è "Clicca qui".

Introduzione a Playwright

Concetti Fondamentali

Utilizzo di Xpath e Selector

Di seguito alcuni esempi di **CSS selector**:

- `p`: Seleziona tutti gli elementi `<p>`.
- `.classe`: Seleziona tutti gli elementi con la classe "classe".
- `#id`: Seleziona l'elemento con l'ID "id".
- `div > p`: Seleziona tutti gli elementi `<p>` che sono figli diretti di `<div>`.
- `a[href='http://esempio.com']`: Seleziona tutti gli elementi `<a>` con l'attributo `href` uguale a "http://esempio.com".
- `input[type='text']`: Seleziona tutti gli elementi `<input>` con l'attributo `type` uguale a "text".

Introduzione a Playwright

Concetti Fondamentali

Altri modi per trovare elementi

- **`locator.filter(options)`**: Filtra un locator esistente basandosi su opzioni come `has_text`, `has_class`, ecc.
- **`locator.getByRole(role, options)`**: Individua elementi in base al loro ruolo ARIA (Accessible Rich Internet Applications), utile per test di accessibilità.
- **`locator.getByText(text, options)`**: Individua elementi in base al testo esatto che contengono.
- **`locator.getByLabel(label, options)`**: Individua elementi di input associati a una determinata etichetta (label).

Introduzione a Playwright

Playwright waiting

Una parte importante nell'automazione della navigazione delle pagine web è determinare se le pagine e gli elementi sono stati caricati oppure no prima di procedere con le azioni su di essi. Per farlo si possono sfruttare per esempio le funzioni di *wait* :

- **`page.wait_for_load_state(state=None, timeout=None)`**: Attende che la pagina raggiunga uno specifico stato di caricamento (load state). Gli stati possibili includono `'load'`, `'domcontentloaded'` e `'networkidle'`. Se non specificato, attende lo stato `'load'`.
- **`page.wait_for_selector(selector, state=None, timeout=None)`**: Attende che un elemento corrispondente al selettore specificato appaia (o scompaia) nella pagina. Lo stato può essere `'attached'`, `'detached'`, `'visible'` o `'hidden'`.
- **`page.wait_for_function(pageFunction, arg=None, timeout=None)`**: Attende che una funzione JavaScript (pageFunction) restituisca un valore truthy. La funzione viene eseguita ripetutamente nel contesto della pagina fino a quando non soddisfa la condizione.

Introduzione a Playwright

Esempio di Base

Ecco un esempio di script Playwright che apre una pagina web, fa uno screenshot e chiude il browser:

```
from playwright.sync_api import sync_playwright

def main():
    with sync_playwright() as p:
        browser = p.chromium.launch()  # Puoi usare anche p.firefox o
p.webkit
        page = browser.new_page()
        page.goto("https://www.alfasoft.it")
        page.wait_for_load_state("networkidle", timeout=5000)
        page.screenshot(path="alfasoft_home.png")
        browser.close()

if __name__ == "__main__":
    main()
```

Introduzione a Playwright

Esempio di Base

Spiegazione

1. **Importa `sync_playwright`:** Importa la libreria Playwright. L'API `sync` è usata in questo esempio pe semplicità (esecuzione sincrona).
2. **Avvia Playwright:** `with sync_playwright() as p:` crea un'istanza di Playwright.
3. **Lancia il browser:** `browser = p.chromium.launch()` lancia un'istanza del browser Chromium. Puoi scegliere tra `chromium`, `firefox` e `webkit`.
4. **Crea una pagina:** `page = browser.new_page()` crea una nuova tab/pagina nel browser.
5. **Naviga alla pagina:** `page.goto("https://www.example.com")` naviga all'URL specificato.
6. **Fai uno screenshot:** `page.screenshot(path="example.png")` salva uno screenshot della pagina corrente nel file "example.png".
7. **Chiudi il browser:** `browser.close()` chiude il browser.

Introduzione a Playwright

Metodi Utili

- `page.click(selector)`: Clicca su un elemento.
- `page.fill(selector, text)`: Inserisce del testo in un input.
- `page.locator(selector).is_visible()`: Verifica se un elemento è visibile.
- `page.locator(selector).inner_text()`: Ottiene il testo interno di un elemento.
- `page.locator(selector).get_attribute(name)`: Ottiene il valore di un attributo di un elemento.
- `page.wait_for_selector(selector)`: Attende che un elemento diventi disponibile nella pagina.

Introduzione a Playwright

Registrare gli Script con Playwright

E' possibile registrare direttamente gli script python attravero lo strumento *codegen*. Per farlo si può lanciare il comando `playwright codegen` seguito dall'URL della pagina web che vuoi automatizzare. A esempio:

```
playwright codegen https://www.example.com
```

Questo comando:

- **Un browser**: aprirà un'istanza del browser (Chromium, Firefox o WebKit, a seconda della tua configurazione predefinita).
- **Playwright Inspector**: aprirà una finestra del Playwright Inspector, che mostrerà il codice Playwright generato in tempo reale mentre interagisci con il browser.

Introduzione a Playwright

Approfondimenti

Consultare la documentazione Ufficiale:

- Intro
- Api

Creare delle schede informative per i seguenti argomenti:

- Browser
- Page
- Locator

> ***NOTA:*** La visione delle api ti servirà durante l'esecuzione degli esercizi.

Introduzione a Playwright

 Esercizi

ese01: Autenticazione e Verifica di Stato

- Sito web: `https://the-internet.herokuapp.com/login`
- Obiettivo: Inserire username e password ed effettuare il login e verificare il successo del login.

ese02: Manipolazione di Elementi Dinamici

- Sito web: `https://the-internet.herokuapp.com/add_remove_elements/`
- Obiettivo: Aggiungere 3 elementi e verificare la loro presenza.

Introduzione a Playwright

ese03: Interazione con Checkbox

- Sito web: `https://the-internet.herokuapp.com/checkboxes`
- Obiettivo: Selezionare tutte le checkbox non selezionate.

ese04: Caricamento di File

- Sito web: `https://the-internet.herokuapp.com/upload`
- Obiettivo: fare l'upload di un file di esempio e verificare l'esito positivo.

ese05: Estrazione di Dati da una Tabella

- Sito web: `https://the-internet.herokuapp.com/tables`
- Obiettivo: Estrarre i dati della tabella *Example 1*.

ese06: Inserire le ore del giorno su alfapeople

- Sito web: `https://dipendenti.alfasoft.it/`
- Obiettivo: Dalla Home fare rendicontazione istantanea.