Assertions у Playwright

Playwright має вбудовані асерти у вигляді функції expect. Викличте expect(value) і виберіть необхідний вам матчер. Є багато вбудованих матчерів, таких як toBe, toEqual, toContain, toBeTruthy та інші.

Ці асерти дозволяють перевіряти різноманітні умови в автоматизованих тестах: від простих порівнянь значень до складних перевірок стану елементів інтерфейсу. Playwright розширює стандартні можливості бібліотеки Jest, додаючи специфічні для веб-тестування матчери.

Універсальність

Підтримка різних типів перевірок для всіх сценаріїв тестування

Читабельність

Зрозумілий синтаксис для швидкого написання тестів

Інтеграція

Вбудована функціональність без додаткових залежностей

Базовий синтаксис асертів

Приклад використання

```
expect(1).toBeTruthy();
```

Це найпростіший приклад використання асертів у Playwright.

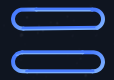
Як це працює

Функція expect приймає значення для перевірки, а метод toBeTruthy() перевіряє, чи є це значення істинним у контексті JavaScript. Якщо умова не виконується, тест завершиться з помилкою.

Такі базові асерти є синхронними і виконуються миттєво. Вони ідеально підходять для перевірки змінних, результатів обчислень та інших простих значень.

toBe

Строга перевірка рівності значень, використовує оператор ===



toEqual

Глибоке порівняння об'єктів та масивів за вмістом

toContain

Перевірка наявності елемента в масиві або підрядка в тексті

toBeTruthy

Перевірка, чи значення є істинним у логічному контексті

Асинхронні асерти

Playwright також має специфічні асерти які є асинхронними і будуть чекати поки задана умова не справдиться. Це одна з найпотужніших можливостей фреймворку, яка відрізняє його від традиційних інструментів тестування.

Асинхронні асерти автоматично повторюють перевірку протягом встановленого таймауту, що робить тести більш стабільними та надійними. Вам не потрібно вручну додавати затримки або цикли очікування.

```
await expect(page.getByTestId('status')).toHaveText('Submitted');
```

У цьому прикладі Playwright буде перевіряти текст елемента кожні кілька мілісекунд, доки він не стане 'Submitted' або не вичерпається таймаут. Це особливо корисно для динамічних веб-додатків, де стан інтерфейсу може змінюватися асинхронно.

01

Ініціація перевірки

Playwright починає виконання асерта та локалізує елемент на сторінці

02

Автоматичне очікування

Якщо умова не виконана, фреймворк чекає та повторює перевірку

03

Успішне завершення

Коли умова виконується, тест продовжує виконання наступних кроків

Асинхронні матчери для елементів

Ось деякі асинхронні асерти що автоматично повторюються, доки перевірка не виконається або не буде вичерпано таймаут. Зауважте, що перевірки виконуються асинхронно, тому потрібно очікувати їх результату await.

toBeAttached()

Елемент прикріплений до DOM-дерева документа

toBeChecked()

Чекбокс або радіокнопка знаходиться в відміченому стані

toBeDisabled()

Елемент вимкнений і недоступний для взаємодії

toBeEnabled()

Елемент увімкнений і готовий до взаємодії користувача

toBeFocused()

Елемент знаходиться в фокусі і приймає введення

toBeHidden()

Елемент прихований і не відображається користувачу

toBeInViewport()

Елемент перетинає видиму область екрану

toBeVisible()

Елемент видимий і має ненульові розміри

toContainText()

Елемент містить вказаний текст або його частину

toHaveAttribute()

Елемент має вказаний DOM-атрибут зі значенням

toHaveClass()

Елемент має вказаний CSS-клас в атрибуті class

toHaveTitle()

Веб-сторінка має вказаний заголовок в тезі title

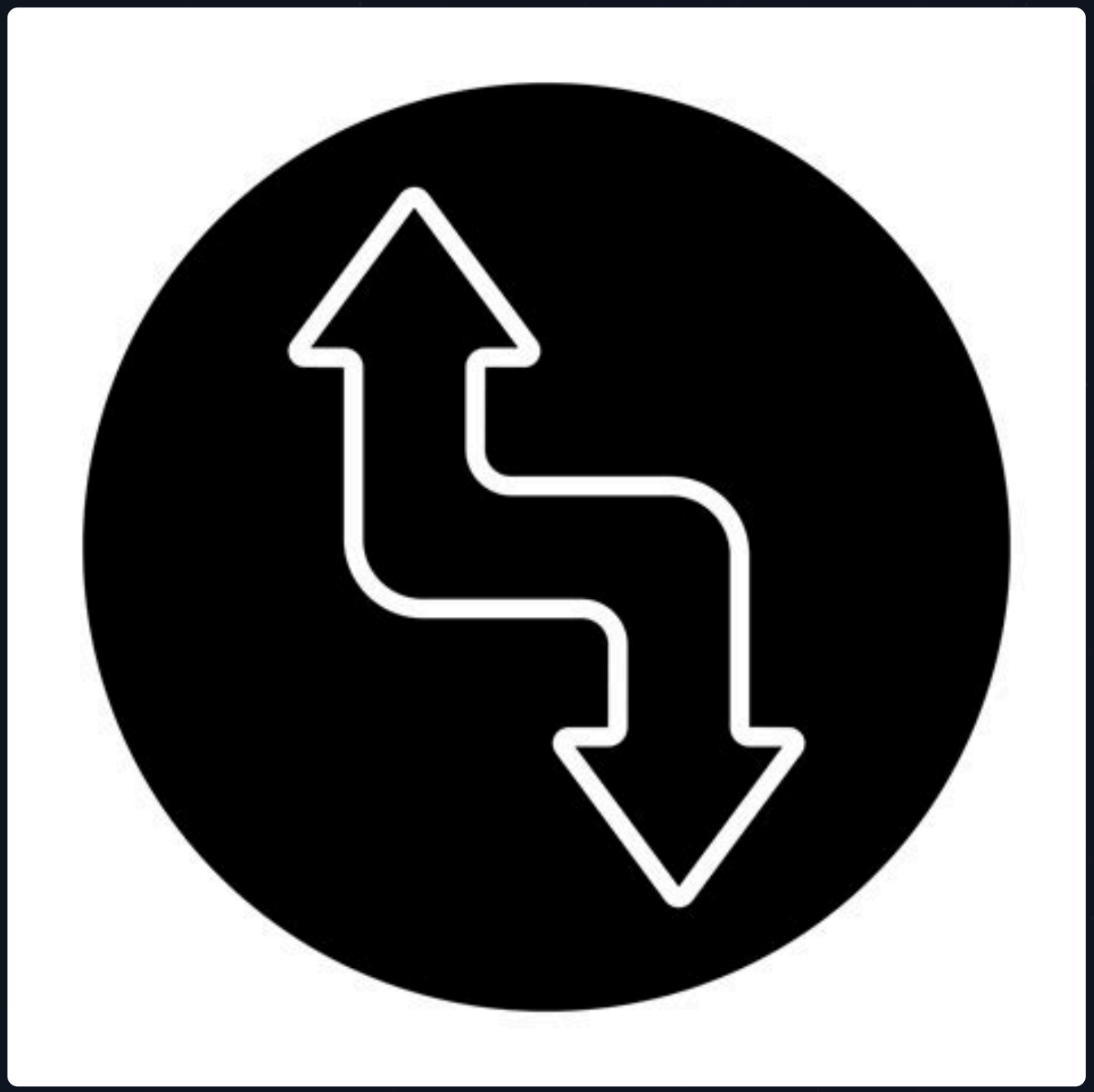
toHaveURL()

Поточна URL-адреса сторінки відповідає очікуваній

toBeOK()

HTTP-відповідь має статус в діапазоні 200-299

Негативні перевірки



Перевірка протилежних умов

Також ми можемо очікувати протилежне, додавши .not до початку матчерів. Це дозволяє перевіряти, що певна умова НЕ виконується, що часто необхідно для повного покриття тестовими сценаріями.

```
await
expect(locator).not.toContainText('
some text');
```

У цьому прикладі ми чекаємо, що локатор не містить тексту 'some text'. Це корисно для перевірки, що елементи інтерфейсу не показують помилкові повідомлення або що певний вміст був успішно видалений.

Позитивні асерти

Перевіряють наявність очікуваного стану або значення

Негативні асерти

Перевіряють відсутність небажаного стану через модифікатор .not

Порада: Використовуйте негативні асерти для перевірки відсутності помилок, прихованих елементів або некоректних станів інтерфейсу. Це робить тести більш всеохоплюючими.

М'які перевірки (Soft Assertions)

За замовчуванням, невдала перевірка припинить виконання тесту. Playwright також підтримує м'які перевірки (soft assertions): невдалі м'які перевірки не припиняють виконання тесту, але позначають його як невдалий.

Це особливо корисно, коли ви хочете зібрати всі помилки за один прохід тесту, замість того щоб виправляти їх по одній. М'які асерти дозволяють виконати всі перевірки і отримати повний звіт про всі невідповідності одночасно.

Використовуйте метод expect.soft() замість звичайного expect() для створення м'якої перевірки. Всі невдалі м'які асерти будуть зібрані і показані в кінцевому звіті про тестування.

```
// Проводимо кілька перевірок, які не зупинять тест при невдачі...
await expect.soft(page.getByTestId('status')).toHaveText('Success');
await expect.soft(page.getByTestId('eta')).toHaveText('1 day');

// ... і продовжуємо тест, щоб перевірити більше речей.
await page.getByRole('link', { name: 'next page' }).click();
await expect(page.getByRole('heading', { name: 'Make another order' })).toBeVisible();
```

Збір помилок

Всі невдалі перевірки фіксуються для аналізу

Продовження виконання

Тест не зупиняється після першої помилки

Повний звіт

Результати всіх перевірок у фінальному репорті