

Playwright: Огляд та встановлення

Сучасний фреймворк для автоматизованого тестування веб-застосунків

The screenshot shows a software development interface with multiple windows open. On the left, there's a file tree and a code editor displaying a file named 'Testing'. The code in the editor is a snippet of a programming language, likely Python or JavaScript, containing logic related to file operations and command-line arguments. In the center, there's a 'Path' window showing a file named 'Lernowki.ses' with a size of 79.0M. To the right, several circular status indicators labeled 'PASS' are visible, each representing a different test or component. At the bottom, a large white text 't: Огляд та встановлення' is overlaid on the screen, and below it, another line of text 'ля автоматизованого тестування веб-застосунків'.

Можливості Playwright

Кросбраузерність

Підтримка Chromium, WebKit та Firefox. Тестування на будь-якій платформі.

Кросплатформеність

Робота на Windows, Linux та macOS локально та на CI.

Кросмовність

API доступне для TypeScript, JavaScript, Python, .NET, Java.

Мобільне тестування

Емуляція Google Chrome для Android та Mobile Safari.

Надійність та стабільність



Автоматичне очікування

Playwright автоматично чекає на готовність елементів перед виконанням дій, що усуває проблеми з таймінгами.

Вбудовані перевірки

Спеціальні асерти оптимізовані для динамічних веб-застосунків.

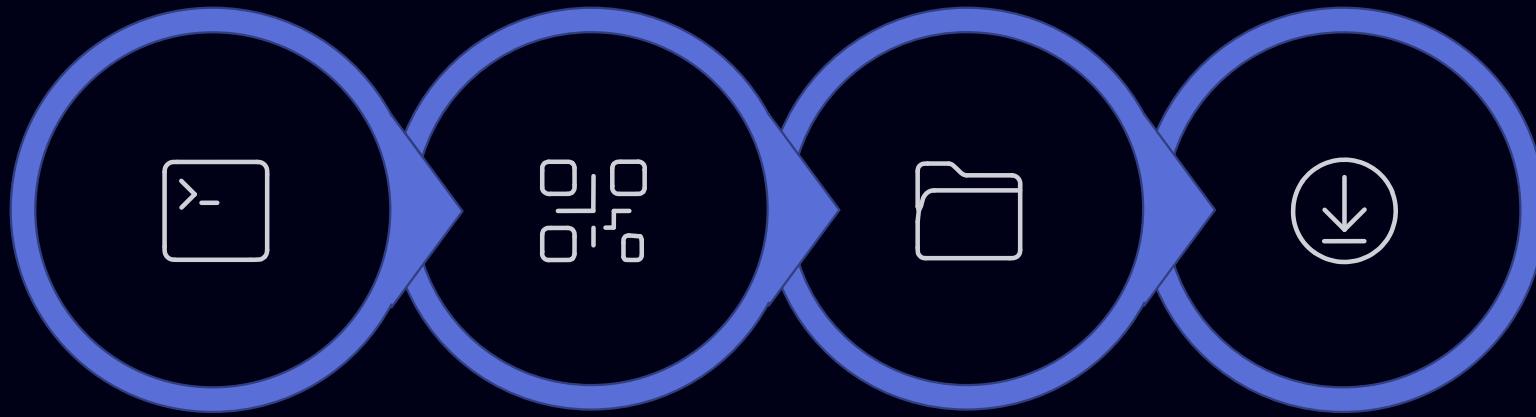
Трасування виконання

Відстеження кожного кроку тесту з можливістю захоплення відео та скріншотів для швидкого виявлення помилок.

Додаткові переваги

- Ізоляція тестів
- Спрощена автентифікація
- Потужні інструменти розробки

Встановлення Playwright



npm init

Choose TS/JS

Configure
tests

Install
browsers

Виконайте команду для встановлення останньої версії:

```
npm init playwright@latest
```

01

Оберіть мову

TypeScript (рекомендовано) або JavaScript

02

Вкажіть папку

За замовчуванням tests або e2e

03

GitHub Actions

Додайте для запуску на CI

04

Встановіть браузери

Автоматичне завантаження

Встановлення браузерів

Важлива інформація

Playwright використовує власні пропатчені версії браузерів для виконання тестів. Це забезпечує стабільність та передбачуваність результатів.

Встановлення всіх браузерів

```
npx playwright install
```

Налаштування проксі

Якщо компанія використовує внутрішній проксі-сервер, Playwright можна налаштувати для завантаження браузерів через нього.

Встановлення конкретного браузера

```
npx playwright install webkit
```

```
npx playwright install chromium
```

```
npx playwright install firefox
```

```
npx playwright install msedge
```

Запуск тестів

Після встановлення Playwright додає приклади тестів для швидкого старту.

Базова команда запуску

```
npx playwright test
```

За замовчуванням

- Виконання у трьох браузерах
- Три робочі потоки (workers)
- Headless режим
- Результати у терміналі

Перегляд репорту

Після завершення тестів можна переглянути детальний HTML-репорт:

```
npx playwright show-report
```

Конфігурація налаштовується у файлі `playwright.config`.

Структура тестів

Базовий синтаксис

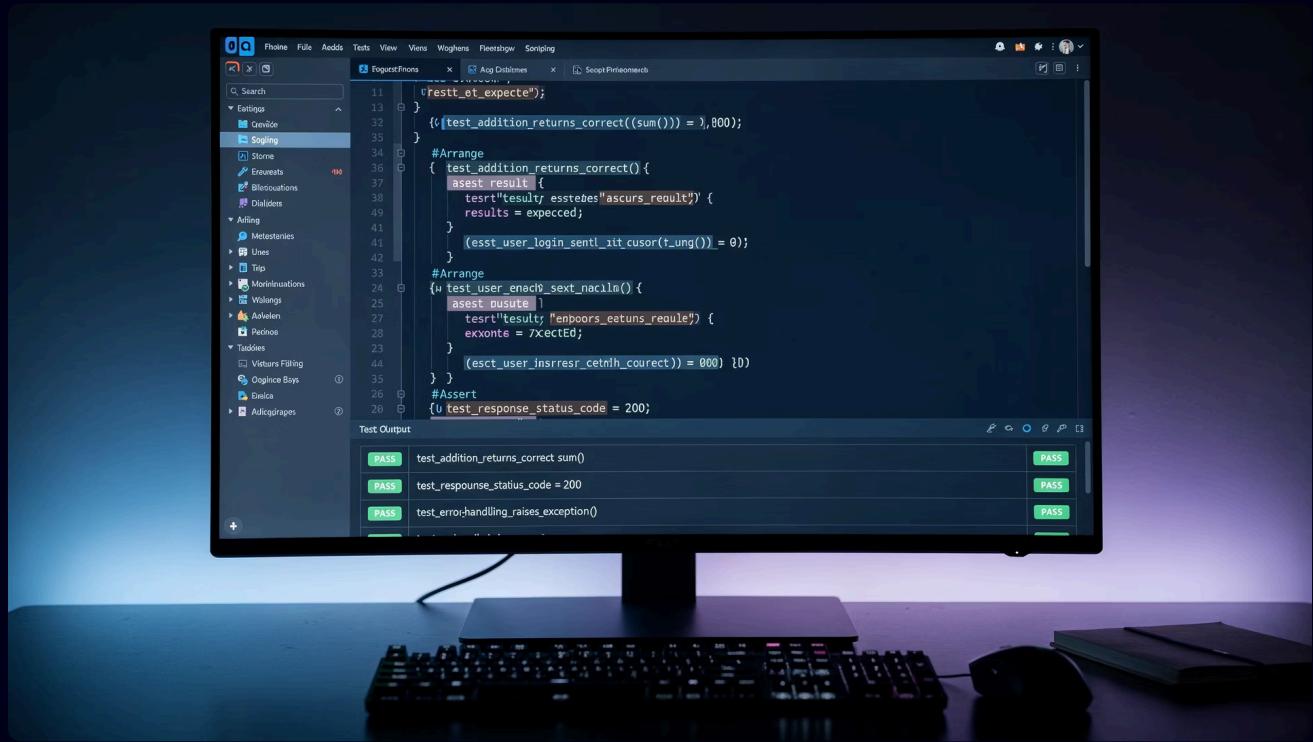
Функція `test` оголошує тест, а `expect` виконує перевірки.

```
import { test, expect } from
'@playwright/test';

test('basic test', async ({ page }) => {
  await
  page.goto('https://playwright.dev/')
;

  const name = await
  page.innerText('.navbar_title');
  expect(name).toBe('Playwright');

});
}
```



Кроки тестів

Функція `test.step` дозволяє розбити тест на логічні кроки для кращої читабельності репортів.

```
import { test, expect } from '@playwright/test';

test('test', async ({ page }) => {
    await test.step('Увійти в систему', async () => {
        // логіка входу
    });

    await test.step('Створити новий запис', async () => {
        // логіка створення
    });
});
```

1

Увійти в систему

2

Створити запис

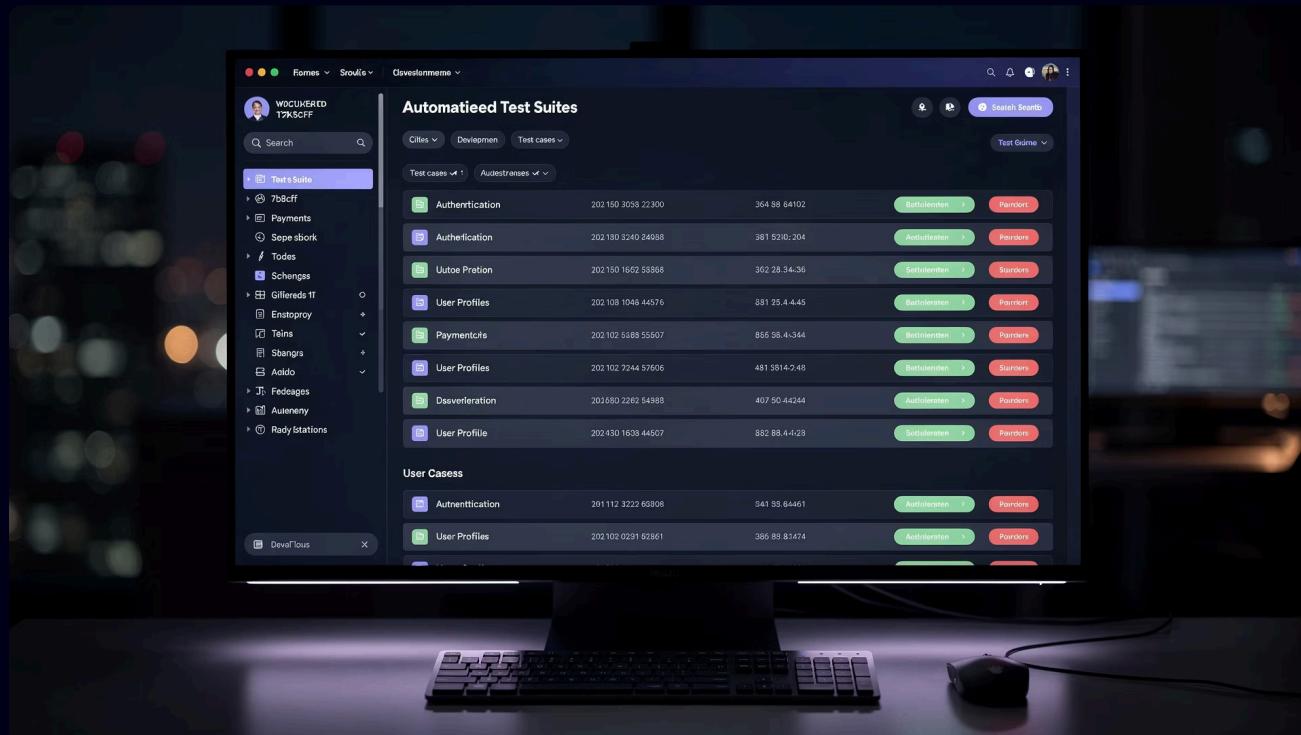
3

Перевірити результат

Групування тестів

test.describe

Функція для створення групи пов'язаних тестів.



```
test.describe('two tests', () => {
  test('one', async ({ page }) => {
    // тест 1
  });

  test('two', async ({ page }) => {
    // тест 2
  });
});
```

Важливо: Кожен тест повинен мати унікальну назву в межах describe блоку або файлу.

Хуки життєвого циклу



Порядок виконання: beforeAll → beforeEach → test → afterEach → afterAll