Playwright 簡介

Playwright 是一款強大的自動化測試工具，可用於瀏覽器自動化、網頁截圖、測試網頁功能等。它支援 Chrome、Firefox、Safari、Chromium 等主流瀏覽器，提供完善的 API，讓您輕鬆進行網頁自動化。

by Nancy CHEN

什麼是 Playwright？

Playwright 是一個開源自動化測試工具，是一個E2E 測試框架，專門用來進行端到端 (End-to-End, E2E) 測試。
由 Microsoft 開發，2020年1月發布，自成立以來，因一直被積極維護，並在 Web 測試社群中得到了快速成長和採用

什麼是 E2E 測試框架？

End-to-End Testing Framework
用以模擬實際用戶從應用程式前端到後端的完整交互過程。E2E 測試的目的是確保應用的所有組件、功能和流程從用戶操作的角度能夠正確協作，模擬實際操作環境來驗證應用是否能按預期運作。

在 E2E 測試中，測試框架會模擬用戶在應用中的操作，如點擊按鈕、輸入數據、提交表單，並檢查應用是否正確地響應這些操作。它通常會涵蓋以下幾個方面：

1. UI 測試
2. 業務流程測試
3. 數據處理與後端驗證

優點

- 全面性
- 高可靠性

挑戰

- 耗時
- 維護難度高

為什麼選擇Playwright？

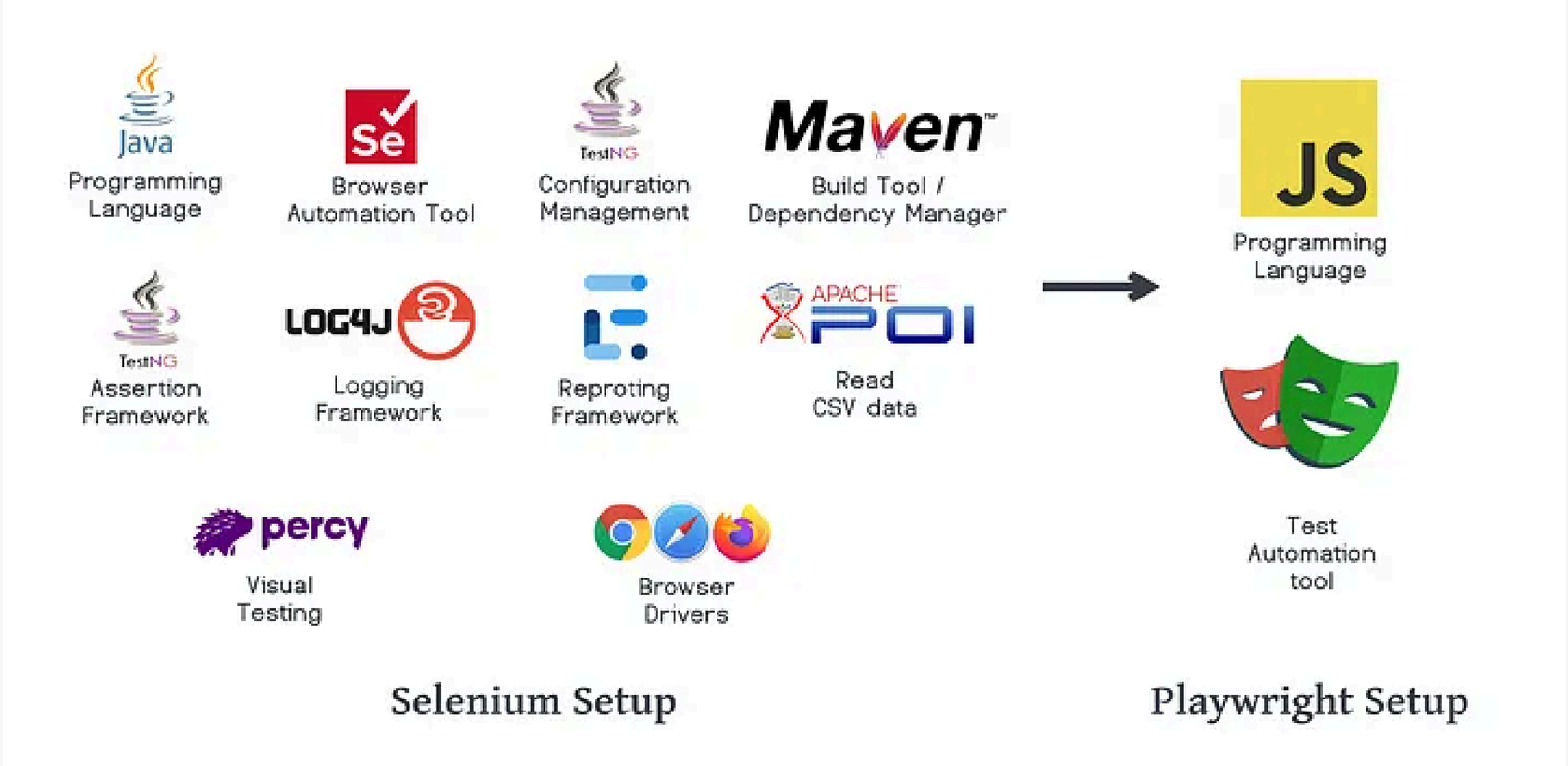
常見的 E2E 測試框架

Cypress：一個現代的 E2E 測試框架，具有簡單易用的 API 和實時回饋功能，深受前端開發者喜愛。它可以直接在瀏覽器中運行測試，並提供強大的調試工具。

Playwright：由 Microsoft 開發，支援多瀏覽器、自動化測試。提供了類似 Cypress 的功能，但支援多語言（如 Python、Java、C#）和更多瀏覽器，適合需要跨平台、跨瀏覽器測試的應用。

Selenium：一個老牌的 E2E 測試框架，支援多種瀏覽器和語言，提供了強大的自動化測試功能，但相較於 Cypress 和 Playwright，配置和運行可能更複雜。

為什麼選擇Playwright？



Playwright 的特點

Any browser • Any platform • One API

Cross-browser. Playwright supports all modern rendering engines including Chromium, WebKit, and Firefox.

Cross-platform. Test on Windows, Linux, and macOS, locally or on CI, headless or headed.

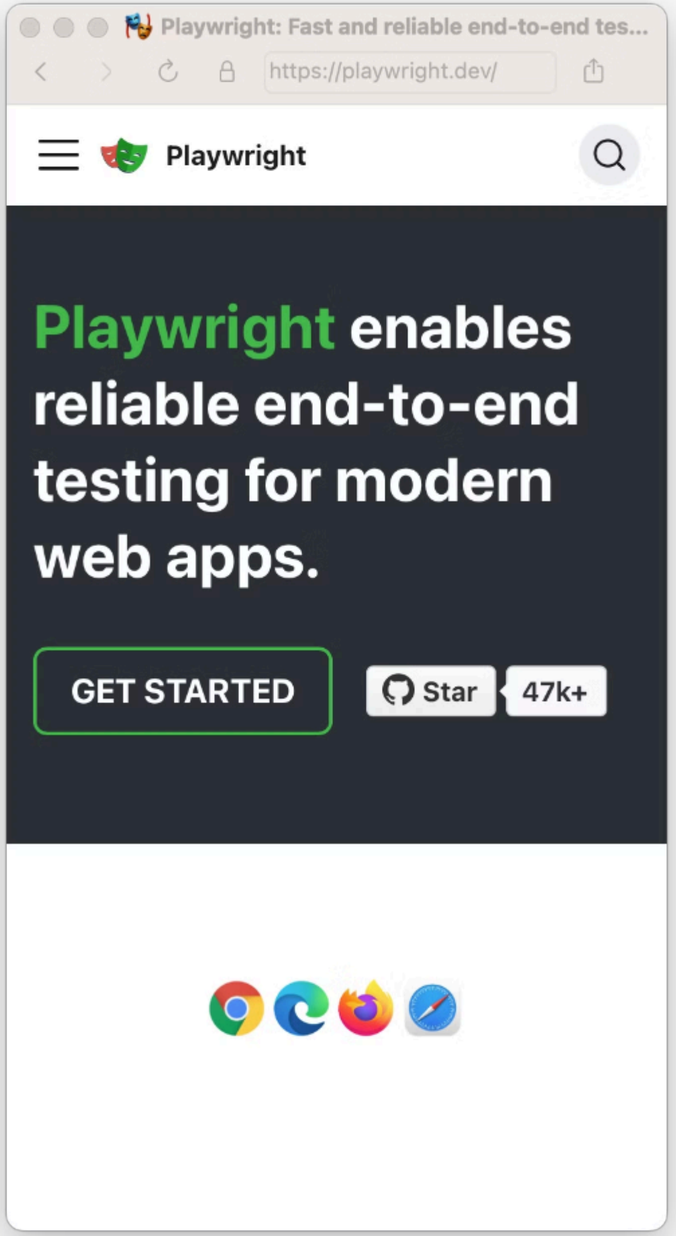
Cross-language. Use the Playwright API in TypeScript, JavaScript, Python, .NET, Java.

Test Mobile Web. Native mobile emulation of Google Chrome for Android and Mobile Safari. The same rendering engine works on your Desktop and in the Cloud.

playwright.config.ts

```ts
import { defineConfig, devices } from '@playwright/test'; // import devices

export default defineConfig({
  projects: [
    {
      name: 'chromium',
      use: {
        ...devices['Desktop Chrome'],
      },
    },
    {
      name: 'Mobile Safari',
      use: {
        ...devices['iPhone 13'],
      },
    },
  ],
});
```



Playwright 的特點

Resilient • No flaky tests

Auto-wait. Playwright waits for elements to be actionable prior to performing actions. It also has a rich set of introspection events. The combination of the two eliminates the need for artificial timeouts - the primary cause of flaky tests.

Web-first assertions. Playwright assertions are created specifically for the dynamic web. Checks are automatically retried until the necessary conditions are met.

Tracing. Configure test retry strategy, capture execution trace, videos, screenshots to eliminate flakes.

Playwright 的特點

No trade-offs • No limits

Browsers run web content belonging to different origins in different processes. Playwright is aligned with the modern browsers architecture and runs tests out-of-process. This makes Playwright free of the typical in-process test runner limitations.

Multiple everything. Test scenarios that span multiple **tabs**, multiple **origins** and multiple **users**. Create scenarios with different contexts for different users and run them against your server, all in one test.

Trusted events. Hover elements, interact with dynamic controls, produce trusted events. Playwright uses real browser input pipeline indistinguishable from the real user.

Test frames, pierce Shadow DOM. Playwright selectors pierce shadow DOM and allow entering frames seamlessly.

Playwright 的特點

Full isolation • Fast execution

Browser contexts. Playwright creates a browser context for each test. Browser context is equivalent to a brand new browser profile. This delivers full test isolation with zero overhead. Creating a new browser context only takes a handful of milliseconds.

Log in once. Save the authentication state of the context and reuse it in all the tests. This bypasses repetitive log-in operations in each test, yet delivers full isolation of independent tests.

1. 在一個測試方法中，先登入並保存狀態：

```javascript
const context = await browser.newContext();
// 執行登入
await page.goto('https://example.com/login');
await page.fill('#username', 'user');
await page.fill('#password', 'password');
await page.click('#loginButton');
// 保存登入狀態
await context.storageState({ path: 'auth.json' });
```

2. 在其他測試中重複使用這個已保存的狀態：

```javascript
const context = await browser.newContext({ storageState: 'auth.json' });
const page = await context.newPage();
await page.goto('https://example.com/dashboard');
```

這樣，你只需在一個測試方法中登入一次，後續測試都可以使用這個登入狀態，節省時間。

Playwright 的特點

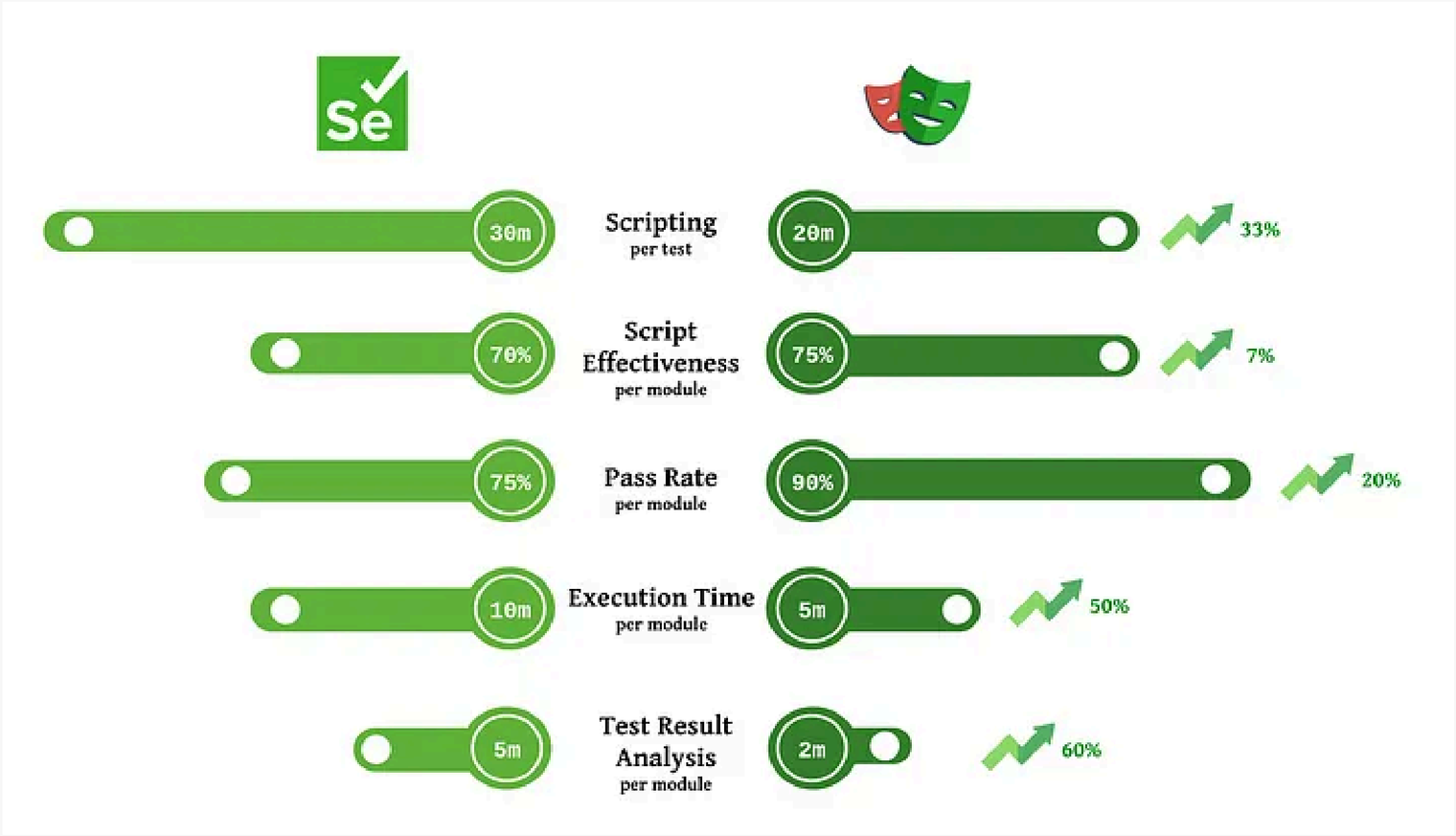
Powerful Tooling

Codegen. Generate tests by recording your actions. Save them into any language.

Playwright inspector. Inspect page, generate selectors, step through the test execution, see click points, explore execution logs.

Trace Viewer. Capture all the information to investigate the test failure. Playwright trace contains test execution screencast, live DOM snapshots, action explorer, test source, and many more.

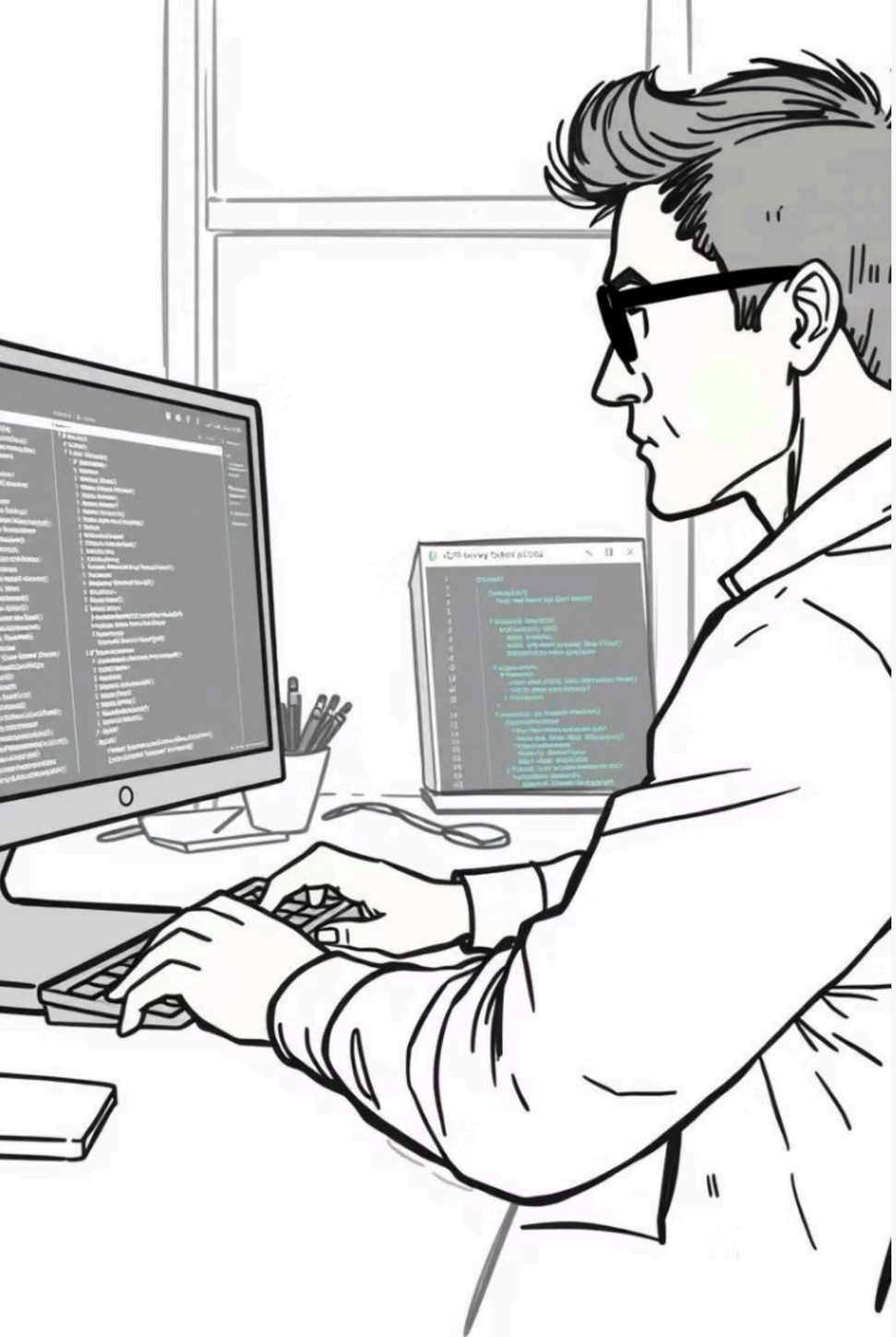
為什麼選擇Playwright？



	PLAYWRIGHT	SELENIUM	CYPRESS
Language Support	JavaScript Java, C#, Python, Ruby	JavaScript Java, C#, Python, Ruby	JavaScript/TypeScript
Browser Support	Chrome, Edge, Firefox, Safari	Chrome, Edge, Firefox, Safari	Chrome, Edge, Firefox, Safari
Framework Support	Jest/Jasmine, AVA, Mocha, and Vitest	Mocha, Jest/Jasmine,, TestNG, JUnit, Cucumber and NUnit	Supports Mocha, Jest/Jasmine, Cucumber
Continuous Integration	Can be easily integrated with continuous integration tools like Jenkins	Can be easily integrated with continuous integration tools like Jenkins	Can be easily integrated with continuous integration tools like Jenkins
Ease of use	Playwright has a user-friendly interface and requires minimal setup	Selenium requires more setup and has a steeper learning curve	Cypress has a user-friendly interface and requires minimal setup
Test Writing Experience	Intuitive	Moderate	Intuitive
DOM manipulation	Easy	Moderate	Easy
Community Support	Growing community	Large and active community with good documentation and support resources	Active community with good documentation and support resources
Support for headless mode	YES	YES	YES

查看比較表:
https://www.lambdatest.com/blog/playwright-vs-selenium-vs-cypress/

playwright 結合電商的實際應用



Playwright 在電商自動化測試的應用

Playwright 可應用於電商平台的自動化測試，例如測試商品搜尋、購物車功能、結帳流程等。

1

測試商品搜尋

驗證商品搜尋結果的準確性、排序邏輯、搜尋功能的可用性。

2

測試購物車功能

驗證商品加入購物車、數量修改、商品移除、結帳流程等功能。

3

測試結帳流程

驗證使用者資訊填寫、付款方式選擇、訂單確認、訂單狀態更新等流程。

Playwright 在同業比價的應用

Playwright 可自動訪問競爭對手的網站，收集商品價格資訊，進行同業比價，幫助電商制定更有效的競爭策略。

自動收集價格資訊

自動訪問競爭對手的網站，收集商品價格、庫存狀態、配送方式等資訊。

價格分析

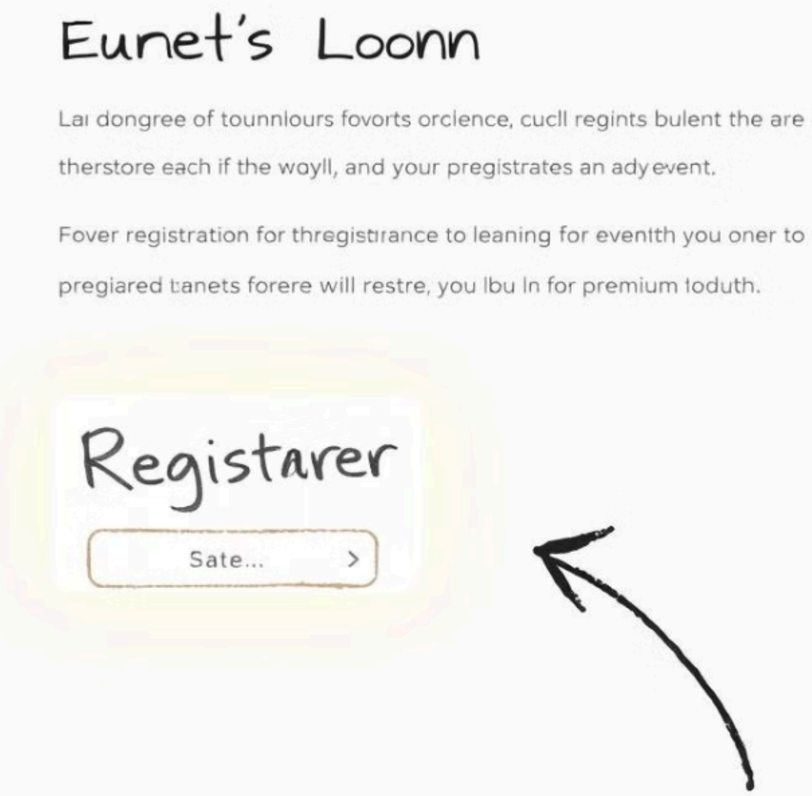
將收集到的價格資訊進行分析，找出同業的價格優勢和劣勢，以便制定更有效的競爭策略。

自動更新價格

根據同業的價格動態，自動調整自身商品價格，保持競爭力。

Playwright 在電商活動自動簽到的應用

Playwright 可自動完成電商活動的簽到流程，例如填寫報名資訊、點擊簽到按鈕等，節省人力成本，提高活動效率。



1

自動登入

自動填寫姓名、電話、電子郵件等報名資訊。

2

自動點擊簽到按鈕

自動點擊簽到按鈕，完成簽到流程。

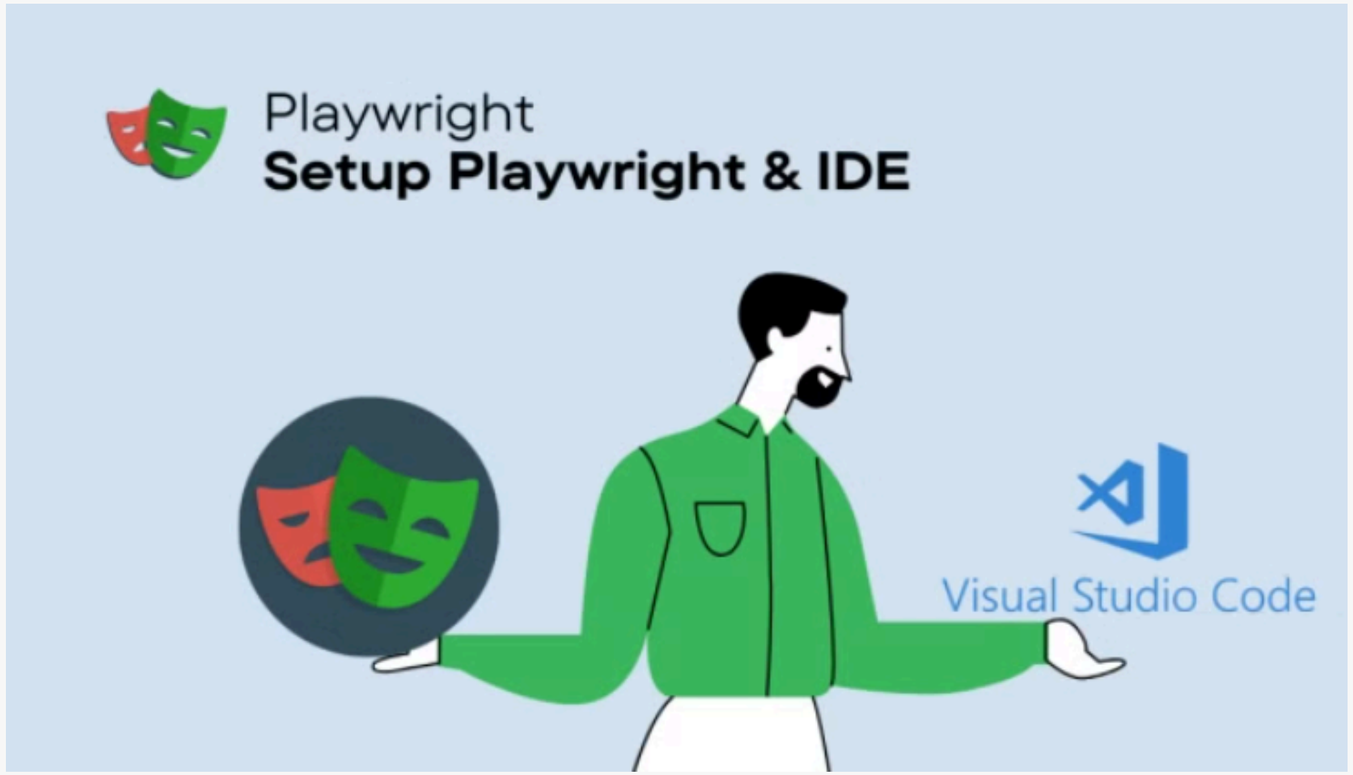
3

自動驗證簽到成功

驗證簽到成功，確保活動參與者的資訊完整性。

如果遇到驗證碼怎麼辦?

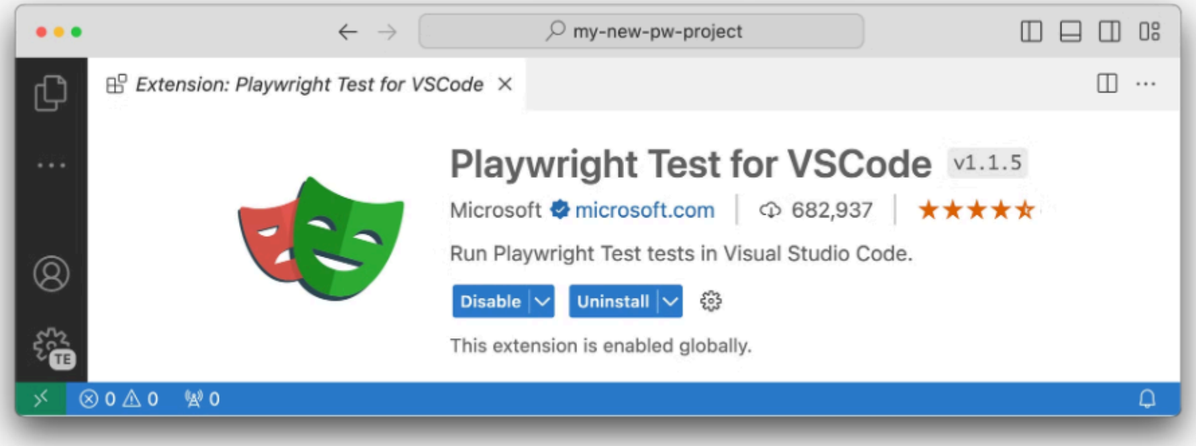
- 測試環境停用驗證碼
- 特殊權限帳號



用VS Code 開始吧!

Playwright 的安裝與設定

透過VSCode來安裝Playwright：

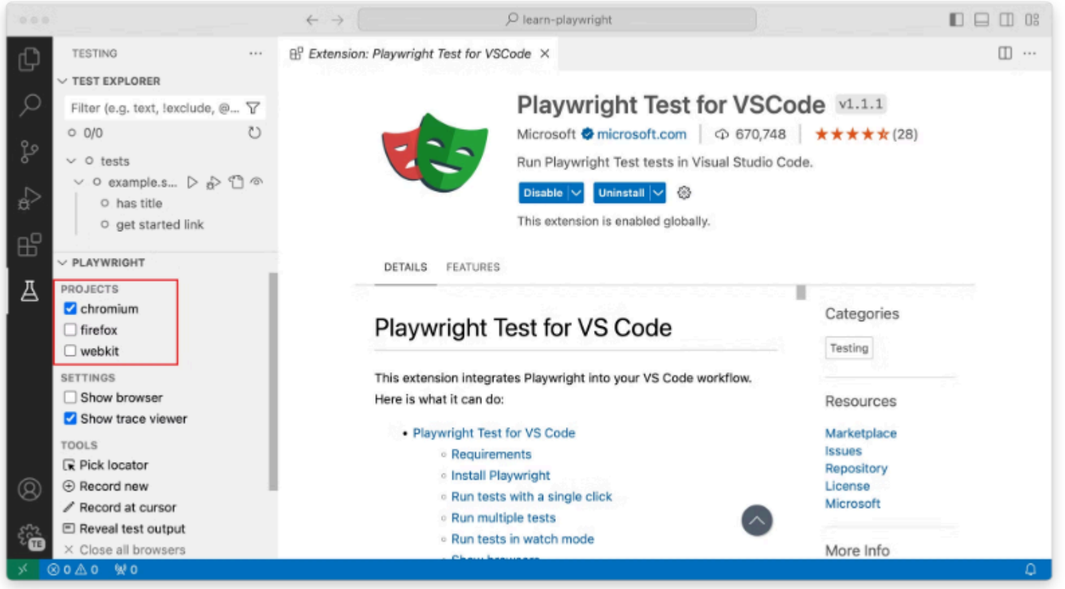


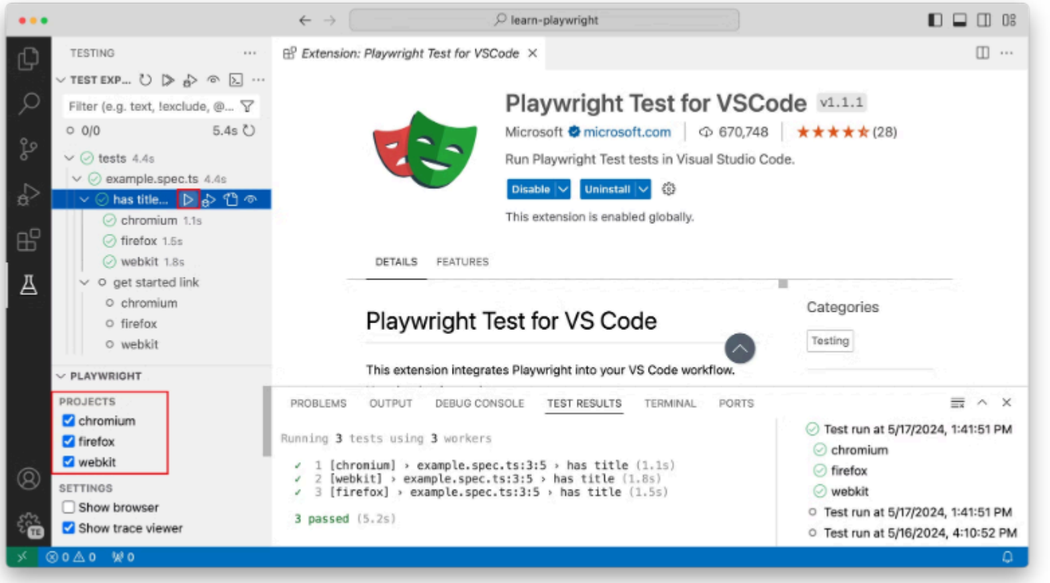

```
> install playwright
```

如果無法執行npm指令，請先下載node.js

選擇模擬運行的瀏覽器

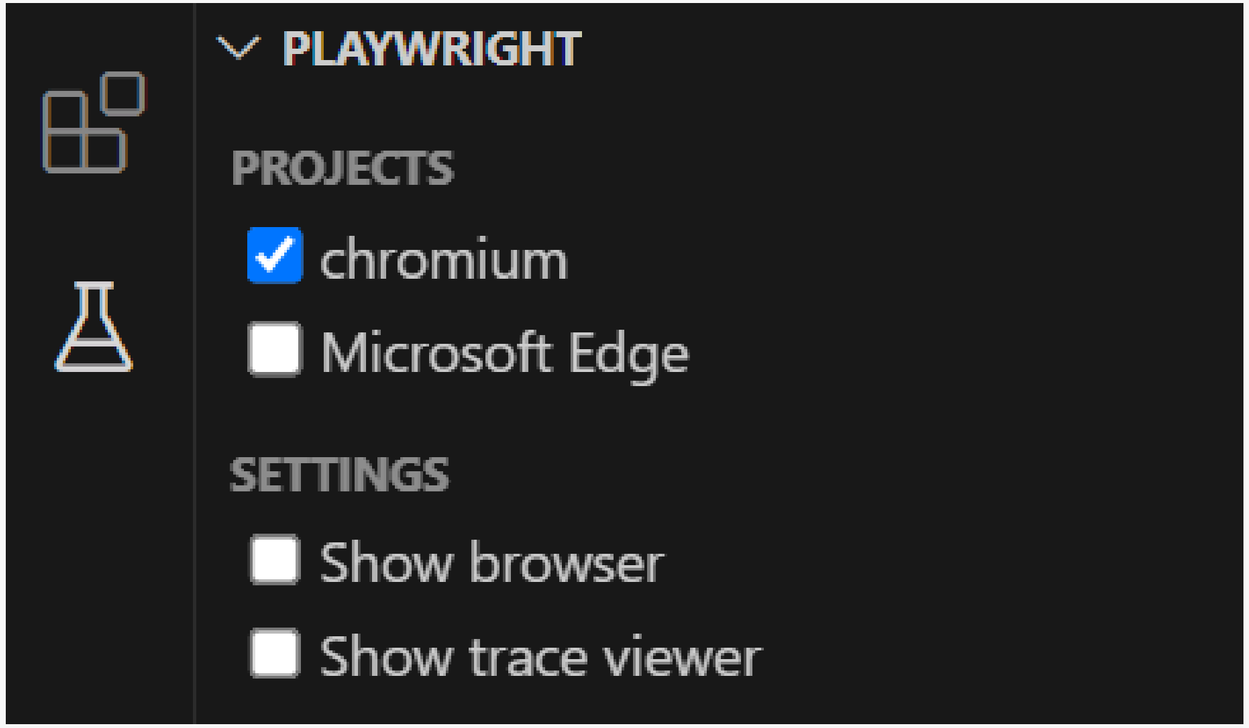
1.透過UI使用預設瀏覽器





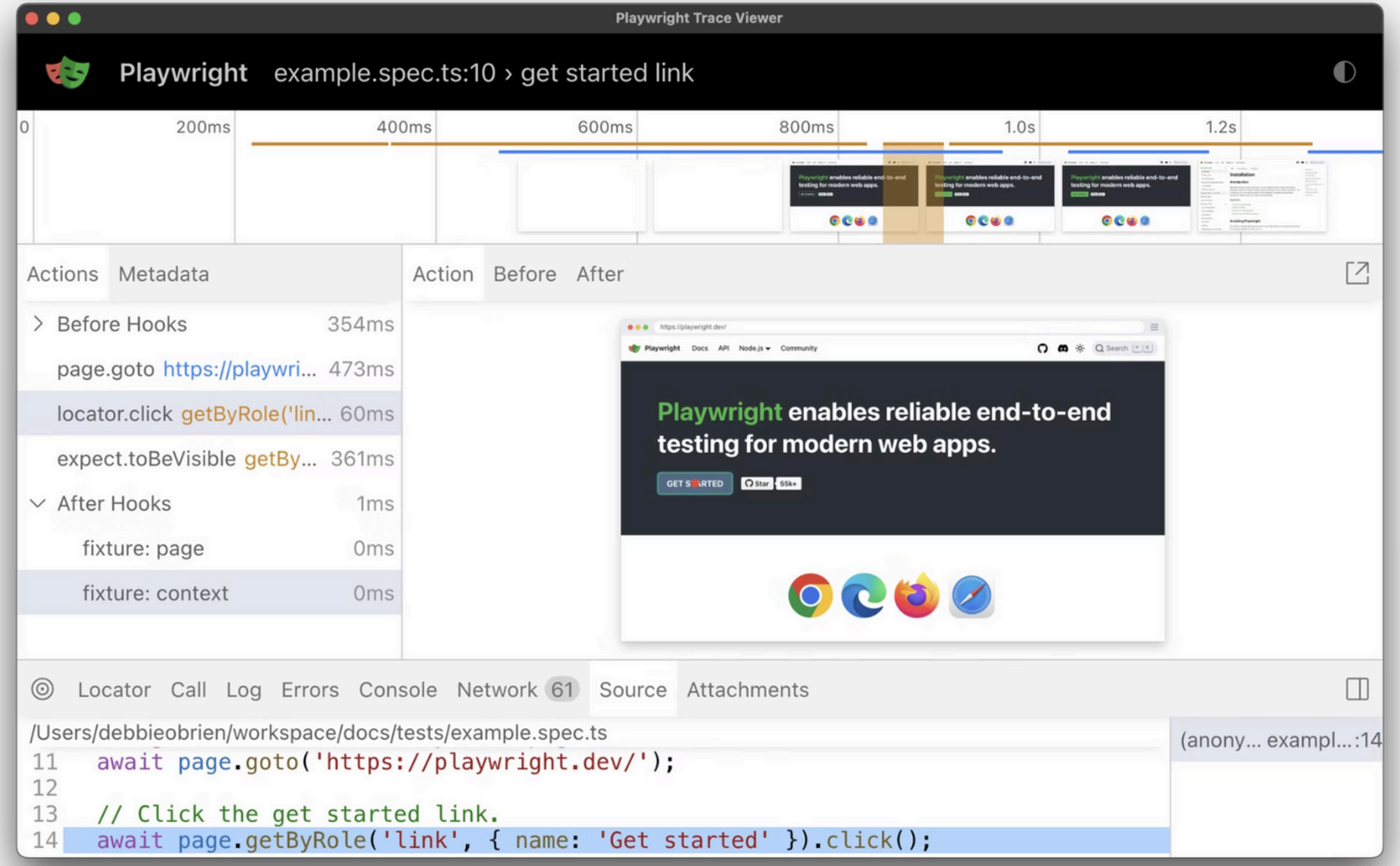
2.透過playwright.config.ts檔案

test UI and function



- 一次可執行測試多個browsers
- show browser/show trace viewer只能擇一

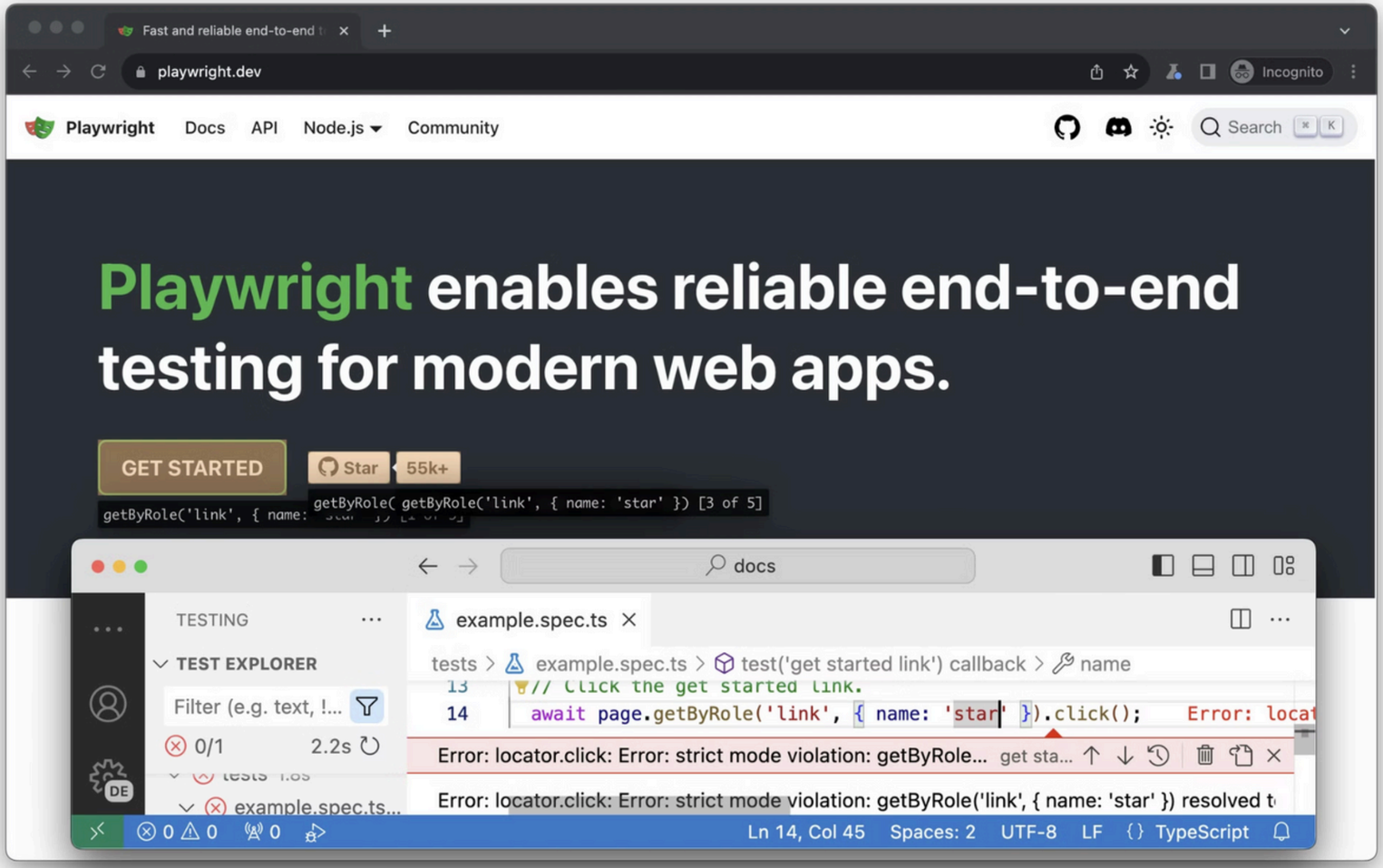
trace viewer



Debugging tests

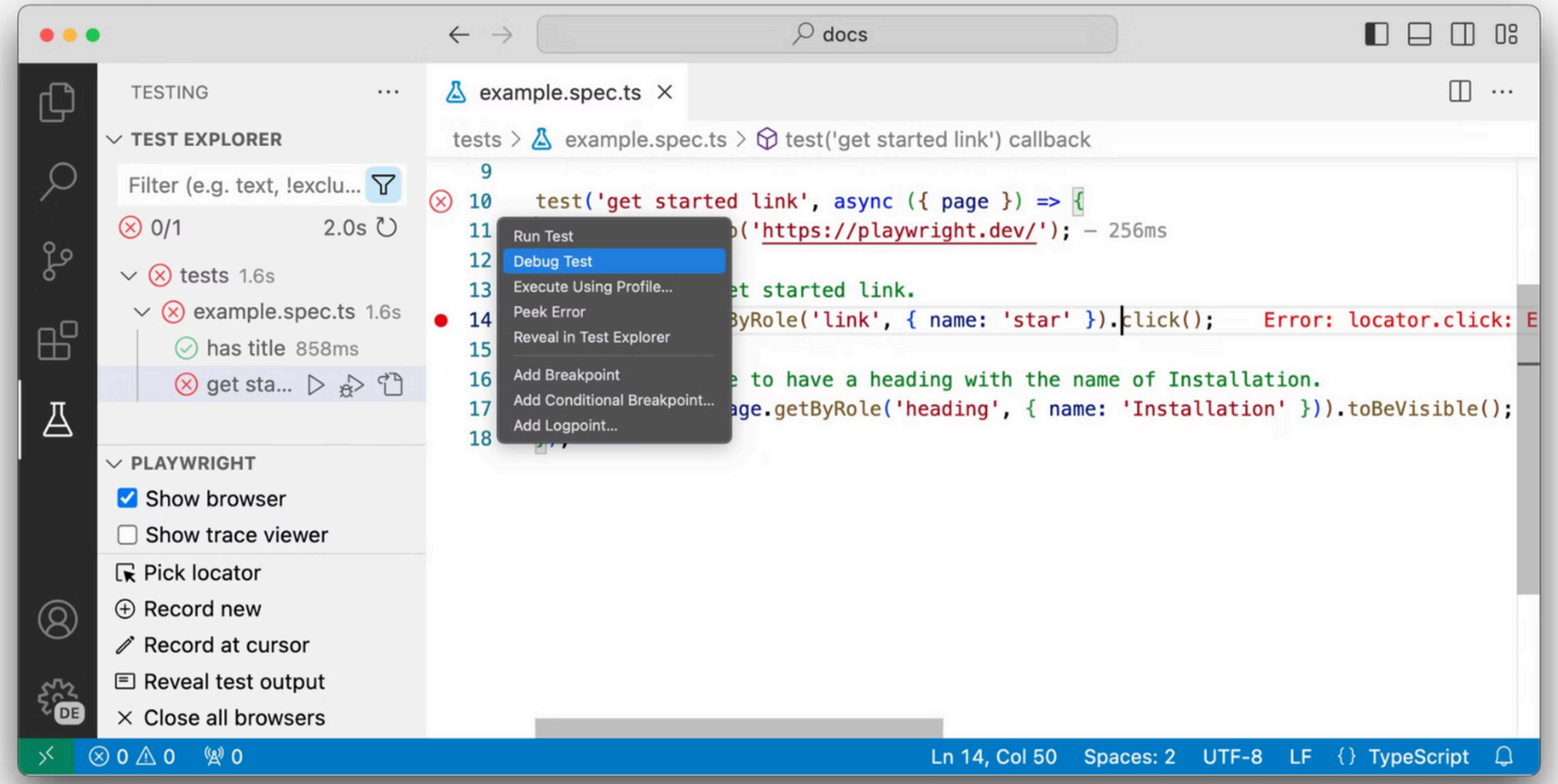
Live debugging

You can debug your test live in VS Code. After running a test with the Show Browser option checked, click on any of the locators in VS Code and it will be highlighted in the Browser window. Playwright will highlight it if it exists and show you if there is more than one result.
You can also edit the locators in VS Code and Playwright will show you the changes live in the browser window.



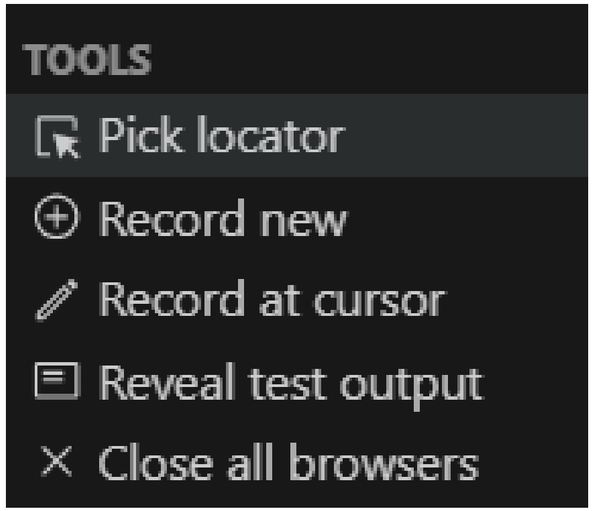
Debug mode

To set a breakpoint click next to the line number where you want the breakpoint to be until a red dot appears. Run the tests in debug mode by right clicking on the line next to the test you want to run.



Generating tests

CodeGen will auto generate your tests for you as you perform actions in the browser and is a great way to quickly get started.



- Pick locator: 允許您在網頁上選擇元素，以生成用於定位該元素的選擇器。
- Record new: 開始新的錄製會話，記錄您在瀏覽器中的操作，以生成測試腳本。
- Record at cursor: 在當前光標位置開始錄製，將新的操作添加到現有的測試腳本中。
- Reveal test output: 顯示測試執行的輸出結果，幫助您查看測試的詳細信息。
- Close all browsers: 關閉所有由Playwright打開的瀏覽器實例。

- 紅色圓圈按鈕：開始/停止錄製操作並生成相應代碼。
- 眼睛圖標：檢查元素是否可見。
- AB圖標：檢查元素的文本內容是否符合預期。
- 目字圖標：檢查輸入框等元素的值是否符合預期。

```
test('test', async ({ page }) => {
  await page.goto('https://www.google.com/'); – 1091ms
  //按了眼睛:assert visible
  await expect(page.getByRole('button', { name: 'Google 搜尋' })).toBeVisible(); – 55ms
  //按了AB:assert text
  await expect(page.getByRole('search')).toContainText('好手氣'); – 17ms
  //按了目:assert value
  await expect(page.getByLabel('搜尋', { exact: true })).toBeEmpty(); – 9ms
});
```

來製作一個自動打卡工具吧

結論與Q&A

Playwright 是一款功能強大、易於使用的網頁自動化測試工具，可以幫助您提高測試效率、提升測試品質，並節省開發時間和人力成本。

Resource

- Getting started - VS Code | Playwright
- Write tests using web first assertions, page fixtures and locators
- microsoft/playwright
- **5 Reasons Why Playwright is the Future of Web Automation Testing: A Side-by-Side Comparison of Selenium vs. Playwright**