
💻 Ejemplo Práctico de Playwright en JavaScript

El siguiente código utiliza el entorno de pruebas integrado de Playwright (Playwright Test), que proporciona las funciones test y expect.

1. Requisitos Clave de JavaScript Aplicados

Requisito JS	Aplicación en el Código
async/await	Es la sintaxis fundamental. Cada interacción con page (navegación, clics, etc.) es una Promesa y debe ir precedida por await.
Funciones Flecha (=>)	Se utilizan comúnmente en la definición de las pruebas (test('...', async ({ page }) => { ... });).
Módulos (ESM)	La función test y expect se importan al inicio: import { test, expect } from '@playwright/test';.

2. El Caso de Prueba (Archivo: test-titulo.spec.js)

JavaScript

```
// 1. Importar las funciones de Playwright Test
import { test, expect } from '@playwright/test';

// 2. Definir una prueba con 'test'
// La función de callback debe ser 'async' porque contiene operaciones 'await'.
test('Verificar el título de la página de Playwright', async ({ page }) => {

  // === ACCIÓN 1: Navegación Asíncrona ===
  // 'await' espera a que la promesa de navegación se resuelva
  await page.goto('https://playwright.dev/');

  // === ACCIÓN 2: Obtener el Título de la Página Asíncrona ===
  // 'await' espera a que se recupere el título.
  const titulo = await page.title();

  // === ACCIÓN 3: Verificación (Assertion) ===
  // Playwright incluye sus propias aserciones 'web-first'

  // 3.1. Usando la aserción de Playwright (Recomendado)
  // El 'toHaveTitle' automáticamente reintenta y espera a que el título coincida.
```

```
  await expect(page).toHaveTitle(/Playwright/);

  // 3.2. Verificación de JavaScript tradicional (funciona, pero menos potente)
  // expect(titulo).toContain('Playwright');

  console.log(`El título de la página es: ${titulo}`);

}); // Fin de la prueba
```

3. Explicación Paso a Paso

1. **import { test, expect } from '@playwright/test';**: Importamos las herramientas clave del *framework*. test define una prueba y expect se usa para hacer verificaciones.
2. **test('...', async ({ page }) => { ... });**:
   - Definimos la prueba y le damos un nombre (Verificar el título...).
   - El objeto { page } es un *fixture* inyectado por Playwright, que representa una pestaña de navegador limpia.
   - La función debe ser **async** para poder usar await dentro.
3. **await page.goto('...')**: Esta es la primera operación asíncrona. Playwright inicia la navegación y **la ejecución se pausa** hasta que la página haya cargado completamente (Promesa cumplida/Resolved).
4. **await page.title()**: Esto solicita el título del navegador. De nuevo, la ejecución espera el resultado.
5. **await expect(page).toHaveTitle(/Playwright/);**: Esta es una de las mayores ventajas. Playwright no solo verifica el estado, sino que **espera inteligentemente** a que el título coincida antes de fallar la prueba.

Este ejemplo ilustra cómo tu conocimiento de JavaScript (especialmente el manejo del **asincronismo**) se traduce directamente en código de automatización robusto con Playwright.