🚀 Playwright: Un Framework de Automatización Web

¿Qué es Playwright?

Playwright es un *framework* de automatización de pruebas de extremo a extremo (End-to-End, E2E) de código abierto, desarrollado inicialmente por Microsoft.

En esencia, es una **librería de Node.js** que proporciona una API para controlar de forma programática navegadores web modernos. Permite a los desarrolladores y profesionales de QA escribir código para simular las acciones de un usuario real en una aplicación web, como hacer clic en botones, rellenar formularios y navegar entre páginas.

Sus Características Distintivas

1. **Compatibilidad Multi-Navegador:** Soporta todos los motores de renderizado modernos en una sola API:
   - **Chromium** (para Chrome y Edge)
   - **WebKit** (para Safari)
   - **Firefox**
2. **Multi-Lenguaje:** Aunque nos enfocamos en **JavaScript/TypeScript**, Playwright también tiene soporte para Python, Java y .NET.
3. **Arquitectura "Web-First":** Está diseñado para ser **confiable** y **resistente a fallos (no-flaky)**:
   - **Espera Automática (Auto-Wait):** Playwright espera automáticamente a que los elementos sean **visibles**, **habilitados** y

accionables antes de interactuar con ellos, eliminando la necesidad de añadir pausas o esperas manuales.
    - **Eventos de Usuario Reales:** Simula eventos de entrada del usuario de manera que el navegador no pueda distinguirlos de las acciones manuales.

🎯 ¿Para Qué Sirve Playwright?

Playwright se utiliza principalmente para garantizar la **calidad y la funcionalidad** de las aplicaciones web a través de la automatización.

1. Pruebas de Extremo a Extremo (End-to-End Testing)

Este es su caso de uso principal. Permite crear escenarios de prueba completos que replican el recorrido de un usuario a través de una aplicación, desde el *login* hasta la realización de una compra o el envío de un formulario.

- **Validación de Flujos:** Asegurar que flujos críticos como el registro, la compra o la búsqueda funcionan correctamente después de cada cambio en el código.

2. Pruebas de Compatibilidad entre Navegadores

Debido a su soporte nativo para los tres motores de renderizado principales, permite ejecutar el mismo conjunto de pruebas en todos ellos, garantizando que la aplicación se vea y funcione igual para todos los usuarios.

3. Automatización de Tareas Repetitivas y Web Scraping

Aunque está optimizado para pruebas, su capacidad para controlar el navegador lo hace útil para:

- **Taras de Regresión:** Ejecutar miles de pruebas de forma rápida y repetida.
- **Web Scraping (extracción de datos):** Navegar por sitios web para recolectar información de forma estructurada.
- **Generación de Artefactos:** Capturar **videos** de la ejecución, **capturas de pantalla** y **rastros (traces)** detallados para el análisis de fallos.

💡 Conexión con JavaScript

Como hemos visto en los archivos adjuntos, el código de Playwright es inherentemente **asíncrono**. Por eso, el dominio de async y await en JavaScript es fundamental:

JavaScript

```
// El código de Playwright requiere JavaScript asíncrono
await page.goto('https://tu-aplicacion.com');
await page.locator('#username').fill('miUsuario');
await page.locator('text=Iniciar Sesión').click();
```

Resumen Rápido

Concepto	Descripción
¿Qué es?	Un framework de Node.js para automatizar navegadores web.
¿Para qué sirve?	Principalmente para realizar **pruebas de extremo a extremo (E2E)** de forma rápida y confiable.
Ventaja Clave	Soporte para **Chromium, Firefox y WebKit** con una sola API y **esperas automáticas** que reducen la inestabilidad de las pruebas.