TEST AUTOMATION ENGINEER – FORMACIÓN INTEGRAL

{desafío} latam_

CURSO:

TEST AUTOMATION ENGINEER – FORMACIÓN INTEGRAL

- Módulo 1: FUNDAMENTOS DEL TESTING DE SOFTWARE
- Módulo 2: CONTROL DE VERSIONES Y ENTORNOS DE DESARROLLO
- Módulo 3: FUNDAMENTOS DE PROGRAMACIÓN APLICADOS AL TESTING (JAVASCRIPT)
- Módulo 4: CARACTERISTICAS DE CYPRESS – AUTOMATIZACIÓN WEB I
- Módulo 5: CARACTERISTICAS DE CYPRESS – AUTOMATIZACIÓN WEB II
- Módulo 6: PRUEBAS DE APIS CON POSTMAN Y SUPERTEST

{desafío} latam_

CURSO:

TEST AUTOMATION ENGINEER – FORMACIÓN INTEGRAL

- **Módulo 7: HERRAMIENTAS DE PLAYWRIGHT Y PRUEBAS CON MÚLTIPLES NAVEGADORES**

- Módulo 8: DESARROLLO GUIADO EN EL COMPORTAMIENTO (BDD) CON CUCUMBER.JS
- Módulo 9: HERRAMIENTAS DE AUTOMATIZACIÓN MÓVIL CON APPIUM
- Módulo 10: HERRAMIENTAS DE INTEGRACIÓN DE PRUEBAS EN CI/CD
- Módulo 11: HERRAMIENTAS DE DOCKER, ENTORNOS VIRTUALIZADOS Y PRUEBAS EN LA NUBE
- HERRAMIENTAS DE AUTOMATIZACIÓN DE UN FLUJO COMPLETO WEB + API + CI/CD

{desafío} latam_

Módulo 7:
Herramientas de playwright y pruebas con múltiples navegadores.

{desafío} latam_

OBJETIVO ESPECÍFICO DEL MÓDULO

- DETERMINAR HERRAMIENTAS DE PLAYWRIGHT Y PRUEBAS CON MÚLTIPLES NAVEGADORES, DE ACUERDO A LAS APLICACIONES WEB, MÓVILES Y APIS.

{desafío} latam_

¿Qué características cree que debería tener una herramienta de automatización de pruebas para adaptarse a múltiples navegadores y entornos, y cómo imagina que Playwright responde a esa necesidad?

¿QUÉ ES PLAYWRIGHT?

- Herramienta open-source de automatización de pruebas desarrollada por Microsoft.
- Permite controlar navegadores modernos (Chromium, Firefox, WebKit).
- Soporta pruebas E2E, visuales, mobile emulation y testing con múltiples contextos.
- Diseñado para entornos modernos: asincronía, paralelismo, y cobertura multibrowser real.
- **Ventaja clave:** soporte nativo de múltiples navegadores con una sola API.

INSTALACIÓN Y PRIMEROS PASOS

- **Instalación rápida:**

```
npm init playwright@latest
```

  - Incluye estructura del proyecto, tests de ejemplo y configuración

- **Sintaxis básica:**

```
const { test, expect } = require('@playwright/t

test('mi primer test', async ({ page }) => {
  await page.goto('https://ejemplo.com');
  await expect(page).toHaveTitle(/Ejemplo/);
});
```

{desafío} latam_

- **Arquitectura:**
- Usa test runner propio
- Maneja múltiples contextos de navegador
- Facilita paralelismo, fixtures, y test parametrizados

PLAYWRIGHT VS CYPRESS

- **Playwright:**
- Soporta múltiples navegadores reales (Chromium, Firefox, WebKit)
- Permite ejecutar tests en paralelo de forma nativa
- Mayor control asincrónico y simulación avanzada (geolocalización, dispositivos, permisos)
- Testing de apps SPA y SSR sin esperar manualmente elementos

- **Cypress:**
- Corre solo en Chromium (aunque puede usar Electron)
- Arquitectura más simple para usuarios iniciales
- Flujo síncrono y esperas automáticas integradas
- UI amigable, ideal para principiantes

- Ambos son excelentes. Cypress es ideal para empezar rápido. Playwright ofrece más potencia en entornos complejos y multibrowser.

PRUEBAS EN MÚLTIPLES NAVEGADORES REALES

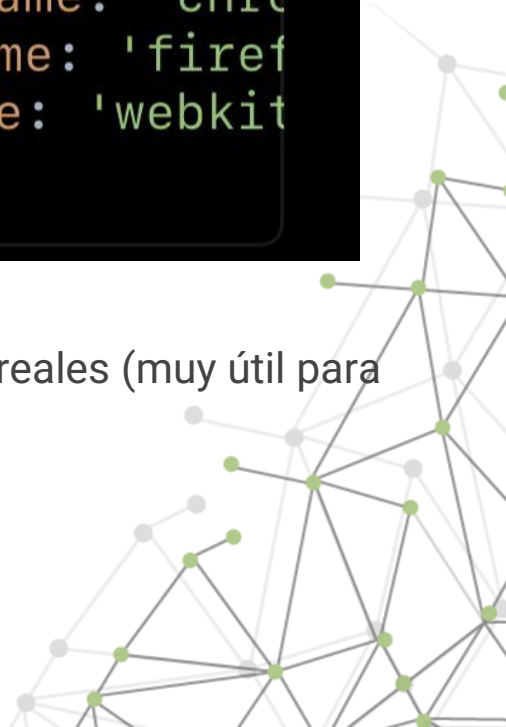
- **Playwright permite correr el mismo test en diferentes motores:**

```
npx playwright test --project=chromium
npx playwright test --project=firefox
npx playwright test --project=webkit
```

- **Configuración desde archivo playwright.config.ts:**

```
projects: [
  { name: 'Chromium', use: { browserName: 'chro
  { name: 'Firefox', use: { browserName: 'firef
  { name: 'WebKit', use: { browserName: 'webkit
]
```

- **Ventaja:** Se puede validar comportamiento en múltiples entornos reales (muy útil para QA cross-browser)



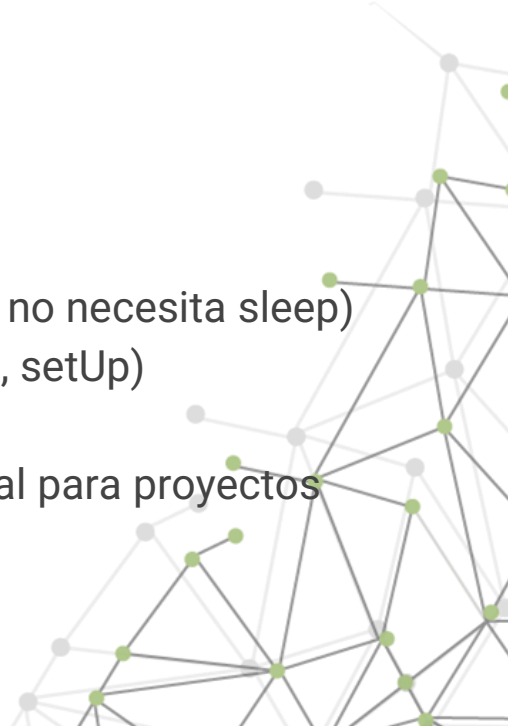
{desafío} latam_

PARALELISMO Y ASINCRONÍA EN PLAYWRIGHT

- **Pruebas paralelas:**
- Se ejecutan por defecto en múltiples workers (hilos)
- Mejora tiempos de ejecución significativamente
- Puede configurarse el número de workers por entorno o CI

- **Manejo de asincronía:**
- Toda interacción usa async/await
- Playwright maneja espera inteligente (await page.locator(...).click() no necesita sleep)
- Fixtures asincrónicos permiten inyección de lógica común (ej: login, setUp)

- El paralelismo y la asincronía nativos hacen que Playwright sea ideal para proyectos grandes o integraciones en pipelines CI/CD.



El Módulo tiene 2 ejercicios prácticos que debe desarrollar…

{desafío} latam_

¿Qué ventajas y limitaciones encontró en el uso de Playwright al compararlo con otras herramientas de testing, y cómo podría aplicar sus capacidades (como asincronía y ejecución en múltiples navegadores) en un proyecto real?

{desafío} latam_

Éxito en la evaluación parcial y en la Prueba Final...

{desafío}
latam_

Academia de
talentos digitales