

Automatización de Pruebas con Serenity BDD, Cucumber y POM

Una guía completa sobre el framework open source líder para automatización de pruebas basadas en comportamiento



¿Qué es Serenity BDD y por qué usarlo?

Serenity BDD es un **framework open source** potente y **flexible** diseñado específicamente para automatización de pruebas en Java. Se integra perfectamente con herramientas como Selenium WebDriver para el control completo de navegadores y Cucumber para implementar desarrollo guiado por comportamiento (BDD).

Una de sus características más destacadas es la generación automática de reportes detallados y documentación viva que captura cada paso de las pruebas con capturas de pantalla y logs completos.



Integración completa

Selenium WebDriver y Cucumber trabajando en armonía



Reportes excepcionales

Documentación automática con evidencia visual de cada ejecución

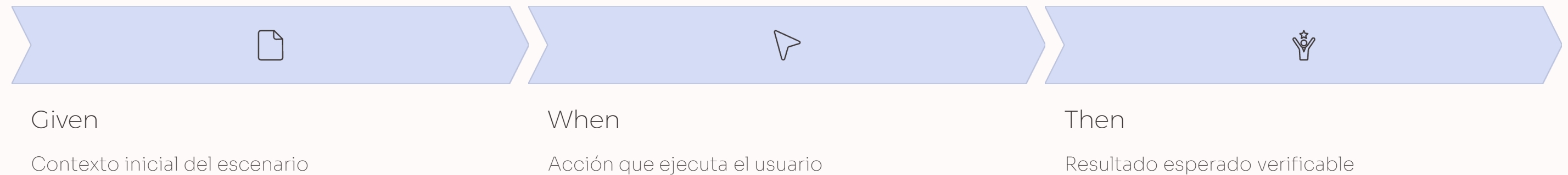


Colaboración efectiva

Facilita comunicación entre equipos técnicos y de negocio

Cucumber: Lenguaje común para pruebas BDD

Cucumber revoluciona la forma en que escribimos y entendemos las pruebas al utilizar **Gherkin**, un lenguaje natural y legible que sigue la estructura Given-When-Then. Este enfoque permite que stakeholders no técnicos, analistas de negocio y product owners puedan **entender, validar y contribuir** directamente a los requisitos del sistema.



Ejemplo práctico

```
Feature: Login de usuario
Scenario: Usuario válido inicia sesión
Given el usuario está en la página de login
When ingresa credenciales válidas
Then debe ver el dashboard principal
```

Serenity ejecuta estos escenarios escritos en Gherkin y los conecta automáticamente con el código Java mediante **step definitions**, creando un puente perfecto entre requisitos de negocio y implementación técnica.

¿Qué es el patrón Page Object Model (POM)?

El patrón **Page Object Model (POM)** es una práctica fundamental en automatización de pruebas que representa cada página web como una clase Java independiente. Este diseño arquitectónico centraliza todos los elementos y acciones de una página en un solo lugar.

01

Encapsulación

Cada página es una clase con sus propios elementos

02

Reutilización

Métodos compartidos entre múltiples pruebas

03

Mantenibilidad

Cambios en UI requieren actualizar solo una clase

Serenity y POM: integración natural

La verdadera magia ocurre cuando combinamos Serenity BDD con el patrón POM. Las clases Page Object extienden de la clase base `PageObject` de Serenity, heredando automáticamente toda su funcionalidad y poder.



Inyección automática

Serenity inyecta WebDriver automáticamente, eliminando configuración manual y boilerplate



Métodos simplificados

API fluida con métodos como `typeInto()`, `clickOn()`, `selectFromDropdown()`



Esperas inteligentes

Manejo automático de sincronización y esperas, reduciendo pruebas flaky



Ejemplo de clase POM con Serenity

```
public class LoginPage extends PageObject {
    @FindBy(id = "username")
    private WebElement campoUsuario;

    @FindBy(id = "password")
    private WebElement campoPassword;

    public void ingresarCredenciales(String user, String pass) {
        typeInto(campoUsuario, user);
        typeInto(campoPassword, pass);
        clickOn("#btnLogin");
    }
}
```

Esta integración crea un ecosistema cohesivo donde el código es limpio, expresivo y fácil de mantener, permitiendo que tu equipo se enfoque en probar funcionalidad en lugar de luchar con infraestructura técnica.