

WEB AUTOMATION

¿Qué es Selenium?

Selenium es una herramienta de pruebas automatizadas para aplicaciones web. Permite a los desarrolladores escribir pruebas en varios lenguajes de programación como Java, C# y Python, entre otros, para evaluar el comportamiento de las aplicaciones web en diferentes navegadores y plataformas.

Selenium se destaca por su capacidad para automatizar acciones que un usuario real realizaría, como hacer clic en botones, completar formularios y navegar entre páginas, sin la necesidad de intervención humana.

¿Qué es Selenium WebDriver?

Selenium WebDriver es un componente de Selenium que permite la interacción directa con el navegador desde el código, sin un servidor intermedio. Proporciona una API para ejecutar instrucciones en el navegador y realizar pruebas más realistas.

Cada navegador requiere un driver específico para su correcta ejecución con Selenium WebDriver. Algunos de los más comunes son:

- **ChromeDriver** para Google Chrome.
- **GeckoDriver** para Mozilla Firefox.
- **EdgeDriver** para Microsoft Edge.
- **SafariDriver** para Safari.

Para garantizar el correcto funcionamiento de Selenium, es necesario descargar el driver correspondiente al navegador que utilizarás.

¿Por qué se deben usar URL completas con **http://** o **https://** en Selenium?

En Selenium WebDriver, es fundamental proporcionar la URL completa, incluyendo el protocolo (**http://** o **https://**), ya que el navegador necesita esta información para determinar cómo acceder al sitio web. Si omites el protocolo, Selenium no

podrá interpretar correctamente la dirección, lo que generará errores al intentar cargar la página.

Tiempo de carga de las páginas y rendimiento

Al ejecutar scripts de Selenium, el tiempo de carga de las páginas web puede variar según la velocidad de la conexión a Internet y el rendimiento del equipo. Algunas acciones, como la navegación entre páginas o la interacción con elementos, pueden tardar algunos segundos en completarse.

Es recomendable agregar **esperas explícitas o implícitas** para garantizar que los elementos estén disponibles antes de interactuar con ellos y evitar errores en la ejecución de los scripts.

¿Cómo instalar Selenium en una aplicación de Maven?

Para agregar Selenium a tu proyecto Maven, debes incluir la dependencia de Selenium WebDriver en el archivo `pom.xml`.

- Busca la versión más reciente de Selenium WebDriver en la documentación oficial de Maven: [Repositorio Maven - Selenium Java](#)
- Copia las líneas de la dependencia
- Abre el archivo `pom.xml` de tu proyecto Maven.
- Dentro del elemento `<dependencies>`, agrega la dependencia de Selenium WebDriver. Ejemplo:

```
Unset
<dependency>
  <groupId>org.seleniumhq.selenium</groupId>
  <artifactId>selenium-java</artifactId>
  <version>4.28.0</version>
</dependency>
```

*Si es necesario, ajusta la versión según tus requerimientos.

- Guarda los cambios en `pom.xml`. Maven descargará automáticamente la dependencia y la integrará en tu proyecto en la próxima compilación..

¿Qué es un script de Selenium?

Un **script de Selenium** es un conjunto de instrucciones escritas en un lenguaje de programación (como Java) que utiliza Selenium WebDriver para automatizar acciones en un navegador web.

Selenium proporciona múltiples funciones para interactuar con elementos de la página, gestionar cookies, ejecutar scripts de JavaScript y mucho más. Algunos de los métodos clave son:

Navegación y control de ventana

- `get(URL)`: Navega a una URL especificada.
- `navigate().to(URL)`: Similar a `get()`, permite navegar a una URL.
- `navigate().back()`: Retrocede en el historial de navegación.
- `navigate().forward()`: Avanza en el historial de navegación.
- `navigate().refresh()`: Recarga la página actual.
- `manage().window().maximize()`: Maximiza la ventana del navegador.
- `manage().window().minimize()`: Minimiza la ventana del navegador.
- `manage().window().setSize(new Dimension(ancho, alto))`: Ajusta el tamaño de la ventana.
- `manage().window().setPosition(new Point(x, y))`: Cambia la posición de la ventana en la pantalla.

Interacción con elementos

- `findElement(By by)`: Encuentra el primer elemento dentro de la página actual según el criterio especificado.
- `click()`: Hace clic en un elemento.
- `sendKeys(CharSequence... keysToSend)`: Envía una secuencia de teclas a un elemento.

Cierre del navegador

- `close()`: Cierra la ventana del navegador actual.
- `quit()`: Cierra todas las ventanas del navegador y finaliza la sesión de WebDriver.

¿Cómo definir un script de Selenium?

Crear un script de Selenium implica escribir código que le indique al WebDriver qué acciones realizar en el navegador.

Para que Selenium pueda controlar el navegador, es necesario configurar el driver correspondiente.

Configuración del driver del navegador

Para que Selenium pueda interactuar con un navegador, es necesario configurar el driver correspondiente. Existen dos maneras de hacerlo:

1. Definir manualmente la ubicación del driver


- Consiste en descargar el driver específico del navegador (por ejemplo, ChromeDriver para Chrome) y establecer su ruta en el código con

```
System.setProperty("webdriver.chrome.driver", "ruta/al/driver").
```

- Esta configuración brinda un mayor control sobre qué versión del driver se está utilizando, lo que es útil en entornos donde se requiere estabilidad y compatibilidad con versiones específicas del navegador.
- Aprender esta configuración es clave para comprender cómo funciona Selenium a nivel de comunicación con los navegadores y para resolver problemas de compatibilidad en proyectos reales.

2. Incorporar la dependencia WebDriverManager

- WebDriverManager automatiza la descarga e instalación del driver adecuado, eliminando la necesidad de especificar su ubicación manualmente.
- Es una opción recomendada para agilizar la configuración y evitar errores relacionados con versiones incompatibles de los drivers.
- Sin embargo, en algunos entornos restringidos o con configuraciones específicas, puede ser preferible utilizar una versión fija del driver en lugar de una administrada automáticamente.

 **En este módulo, utilizas la configuración manual del driver** porque te permite comprender mejor su funcionamiento y enfrentar escenarios donde la administración automática no es viable. Una vez que domines esta configuración, podrás optar por herramientas como WebDriverManager para simplificar el proceso en proyectos futuros.

Una vez configurado el driver, puedes crear una instancia de `WebDriver` y utilizar sus métodos para abrir un navegador, navegar a una URL y realizar acciones en la página.

Por ejemplo, el siguiente script automatiza la navegación entre diferentes sitios web, manipula la ventana del navegador y finalmente la cierra correctamente. Para más información, consulta la documentación oficial.

```
package com.egg;

import org.openqa.selenium.Dimension;
import org.openqa.selenium.Point;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class Application {

    public static void main(String[] args) {
        // Crea una instancia de ChromeDriver
        System.setProperty("webdriver.chrome.driver", "ruta/a/chromedriver.exe");
        // Crear una instancia de ChromeDriver
        WebDriver driver = new ChromeDriver();
        // Abrir Google
        driver.get("https://www.google.com");
        // Maximizar la ventana
        driver.manage().window().maximize();
        // Navegar a OpenAI
        driver.navigate().to("https://openai.com");
        // Reducir la ventana a la mitad y centrarla
        driver.manage().window().setSize(new Dimension(800, 600));
        driver.manage().window().setPosition(new Point(400, 200));
        // Navegar a Wikipedia
        driver.navigate().to("https://www.wikipedia.org");
        // Restaurar el tamaño original de la ventana
        driver.manage().window().maximize();
        // Cerrar el navegador
        driver.quit();
    }
}
```

Con estos conocimientos, ya estás listo para comenzar a crear tus propios scripts de automatización con Selenium y explorar más a fondo las capacidades de esta herramienta. Para más información, consulta la [documentación oficial](#).