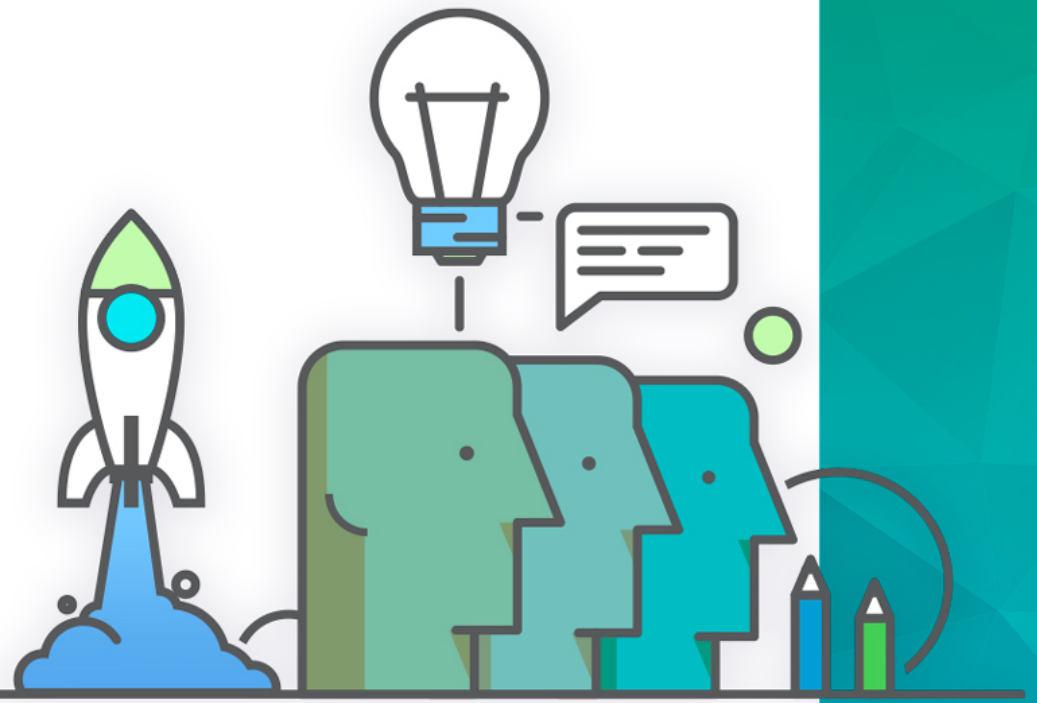
campus
Periferia IT

Periferia IT
Group



Automatización

Módulo 5: Introducción a Selenium

Un curso del Área DevOps
Periferia IT Group

Periferia IT
Corp

CONTACT&
BUSINESSIT

COOLEXPRESS

Contenido

- Introducción a Selenium
- Los cuatro componentes de Selenium
- ¿Para qué sirve Selenium?
- Selenium Webdriver

INTRODUCCIÓN A SELENIUM

- Selenium automatiza los navegadores.
- Es para la automatización de aplicaciones web con fines de pruebas
- Utiliza para comprobar si el software que se está desarrollando funciona correctamente.
- Ayuda mucho en las pruebas de regresión porque consigue pruebas automatizadas que luego se pueden reutilizar cuando se necesite.

CARACTERÍSTICAS DE SELENIUM

- Las acciones serán ejecutadas punto a punto, si así se considera.
- A la hora de escribir el código tiene la opción de autocompletar.
- Se puede referenciar a objetos DOM: nombre, ID o con XPath.
- Ejecutar test complejos que ahorran muchas horas de trabajo.
- Gran depuración y puntos de verificación
- Almacenamiento en varios formatos los test realizados

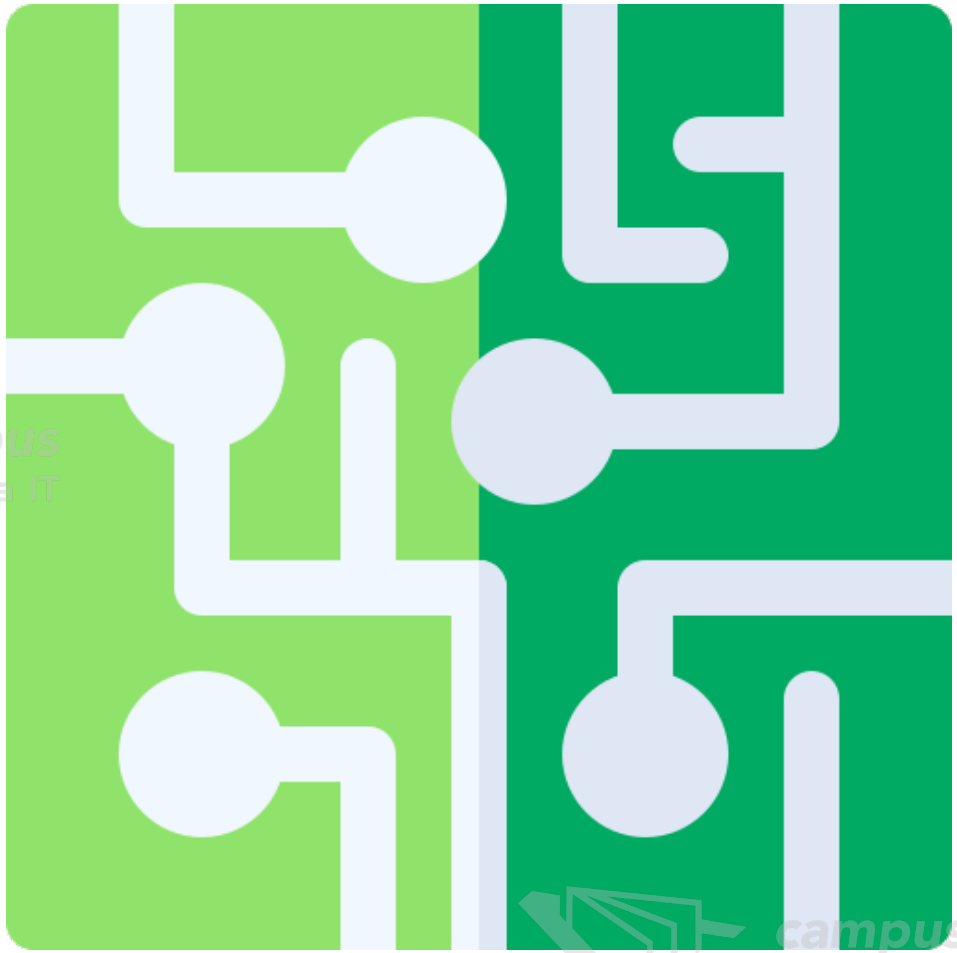
LOS CUATRO COMPONENTES DE SELNIUM

1. **Selenium IDE**
   - Es un entorno de desarrollo integrado para scripts de Selenium
   - Este IDE incluye todo el Selenium Core, que permite grabar y reproducir de forma fácil las pruebas

2. **Selenium RC (Remote Control)**
   - Es una herramienta para automatizar pruebas de interfaz de usuario (UI) de aplicaciones web.
   - Consta de dos componentes: a) un servidor que actúa como proxy para controlar e interactuar con un navegador web. b) bibliotecas para crear programas para el servidor usando una amplia gama de lenguajes de programación.

3. **Selenium WebDriver**
   - También es una herramienta para automatizar pruebas UI de aplicaciones web pero implementa un enfoque más moderno y estable que Selenium RC.
   - Controla el navegador comunicándose directamente con él interactuando con los elementos de la página de manera más realista.

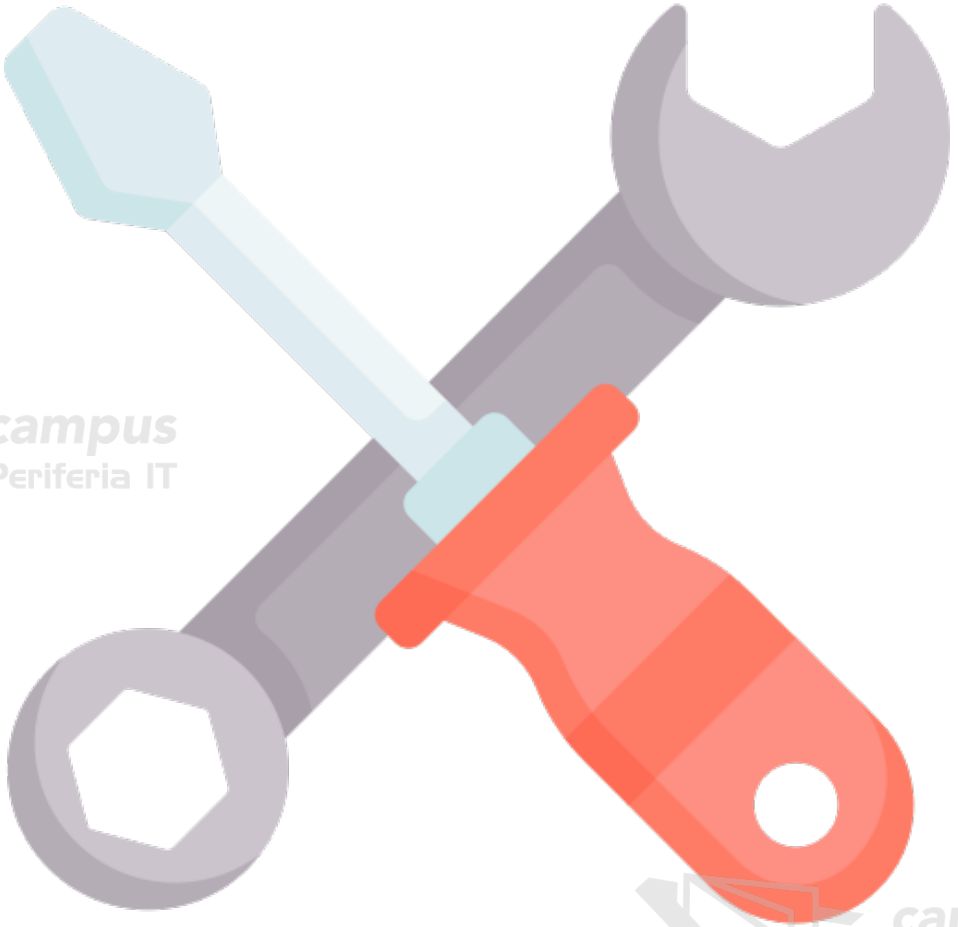
4. **Selenium Grid**
   - Se especializan en ejecutar múltiples pruebas a través de diferentes navegadores, sistemas operativo y máquinas.
   - Puede conectarse con Selenium Remote especificando el navegador, la versión del navegador y el sistema operativo que desee.



campus Periferia IT

¿PARA QUÉ SIRVE SELENIUM?

- Su objetivo principal es comprobar que un software funcione correctamente.
- La mayoría de usuarios que trabajan con Selenium son programadores QA o Engineer QA.
- Compreba que todo funcione correctamente.
- Evita problemas futuros, ya que ahí es dónde reside la mejor característica de Selenium para algunos Ingenieros QA.



campus Periferia IT

SELENIUM WEBDRIVER

- WebDriver está diseñado como una interfaz de programación simple y más concisa.
- WebDriver es una API compacta orientada a objetos.
- Conduce el navegador de manera efectiva

Selenium admite la automatización de todos los principales navegadores del mercado mediante el uso de WebDriver . WebDriver es una API y un protocolo que define una interfaz independiente del idioma para controlar el comportamiento de los navegadores web.

Cada navegador está respaldado por una implementación específica de WebDriver, llamada controlador . El controlador es el componente responsable de delegar en el navegador y maneja la comunicación hacia y desde Selenium y el navegador.

El marco de Selenium une todas estas piezas a través de una interfaz orientada al usuario que permite que los diferentes backends del navegador se utilicen de forma transparente, lo que permite la automatización entre navegadores y plataformas.

INSTALAR BIBLIOTECAS

La instalación de las bibliotecas de Selenium para Java se realiza mediante una herramienta de compilación. Puede ver todas las versiones disponibles en Maven Repository

```
<dependency>
  <groupId>org.seleniumhq.selenium</groupId>
  <artifactId>selenium-java</artifactId>
  <version>4.0.0</version>
</dependency>
```

INSTALAR CONTROLADORES DE NAVEGADOR

A través de WebDriver, Selenium admite todos los principales navegadores del mercado, como Chrome/Chromium, Firefox, Internet Explorer, Edge, Opera y Safari. Siempre que sea posible, WebDriver controla el navegador mediante el soporte integrado del navegador para la automatización

Navegador	SO compatible	Mantenido por
cromo/cromo	Windows/mac OS/Linux	Google
Firefox	Windows/mac OS/Linux	Mozilla
Borde	Windows/mac OS	microsoft
explorador de Internet	ventanas	Proyecto Selenio
Safari	macOS High Sierra y más reciente	manzana

SELENIUM WEBDRIVER

FORMAS DE USAR CONTROLADORES

1. Importar administrador de controladores web

```
import io.github.bonigarcia.wdm.WebDriverManager;
```

2. Llamar `setup()` coloca automáticamente el controlador de navegador correcto donde el código lo verá:

```
WebDriverManager.chromedriver().setup();
```

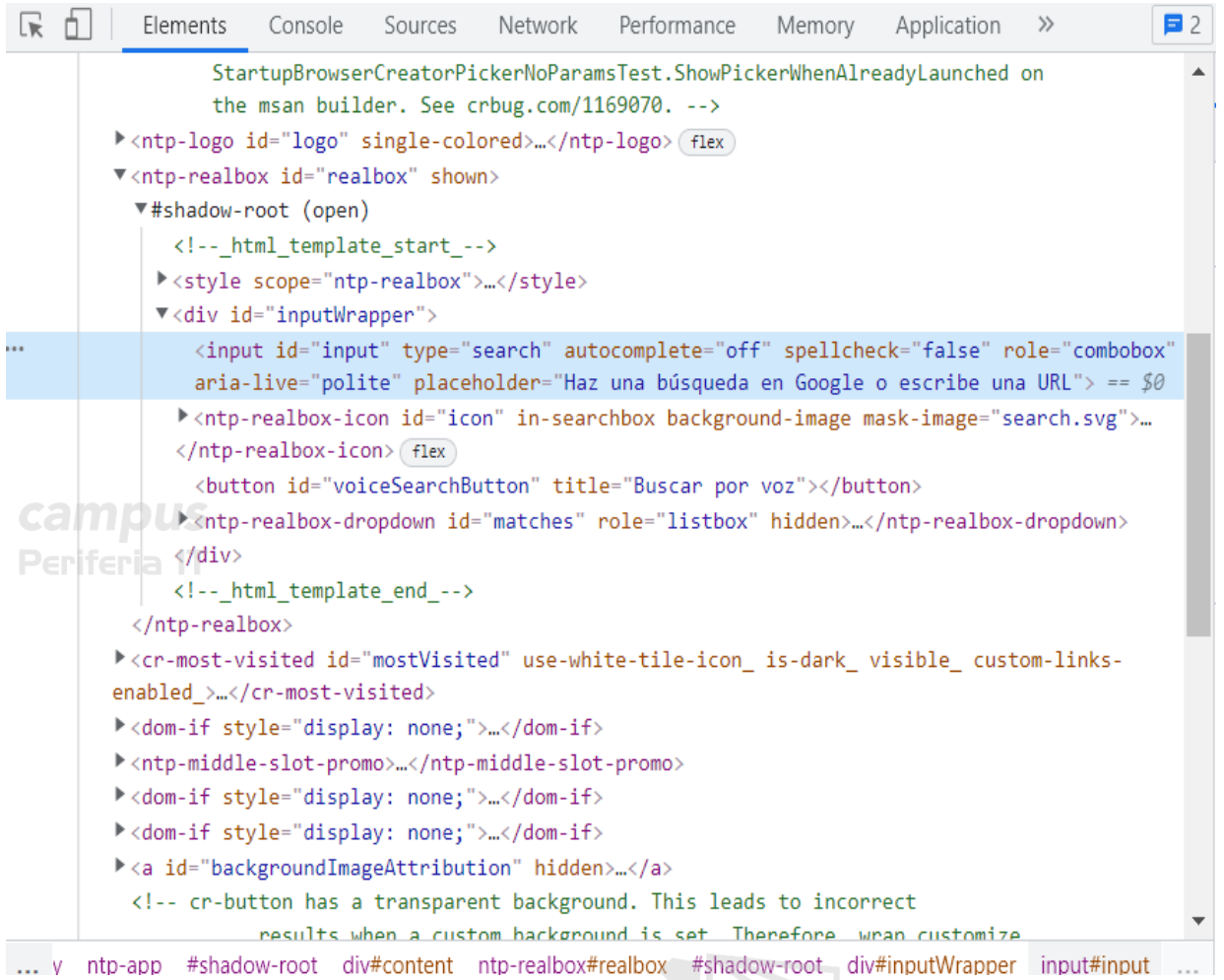
3. Simplemente inicialice el controlador como lo haría normalmente:

```
ChromeDriver driver = new ChromeDriver();
```

```
System.setProperty("webdriver.chrome.driver","/path/to/chromedriver");
ChromeDriver driver = new ChromeDriver();
```

Una página web consta de numerosos WebElements , como cuadros de texto, botones, listas, etc. Podemos realizar una variedad de acciones en estos WebElements utilizando comandos de Selenium como buscar elementos, asociar eventos con elementos web, etc.

Para realizar estas acciones primero necesitamos interactuar con una página web para que podamos usar los comandos/acciones de WebElement.



SELENIUM WEBDRIVER

¿CÓMO ENCONTRAR ELEMENTOS USANDO SELENIUM WEBDRIVER?

Para interactuar con WebElements, primero debemos encontrar o ubicar estos elementos en la página web. Podemos encontrar elementos en una página web especificando los atributos

Locador	*Descripción*
identificación	encuentra elementos por atributo ID. El valor de búsqueda dado debe coincidir con el atributo ID.
nombre	Encuentra o ubica elementos basados en el atributo NOMBRE. El atributo de nombre se utiliza para hacer coincidir el valor de búsqueda.
nombre de la clase	Busca elementos que coincidan con el nombre de clase especificado. Tenga en cuenta que las clases compuestas no están permitidas como nombres de estrategia.
nombre de la etiqueta	Encuentra o localiza elementos que tienen nombres de etiqueta que coinciden con el valor de búsqueda.
Selector de CSS	Coincide con el selector de CSS para encontrar el elemento.
XPath	Hace coincidir la expresión XPath con el valor de búsqueda y, en función de eso, se ubica el elemento.
Texto del enlace	Aquí el texto visible cuyos elementos de anclaje se van a encontrar se compara con el valor de búsqueda.
texto de enlace parcial	Aquí también hacemos coincidir el texto visible con el valor de búsqueda y encontramos el valor de anclaje. Si estamos haciendo coincidir varios elementos, solo se seleccionará la primera entrada.

¿CÓMO ENCONTRAR ELEMENTOS EN SELENIUM?

Como se discutió, Selenium WebDriver proporciona dos métodos mediante los cuales podemos encontrar un elemento o una lista de elementos en una página web. Estos son:

findElement() : este método encuentra de forma única un elemento web en la página web.

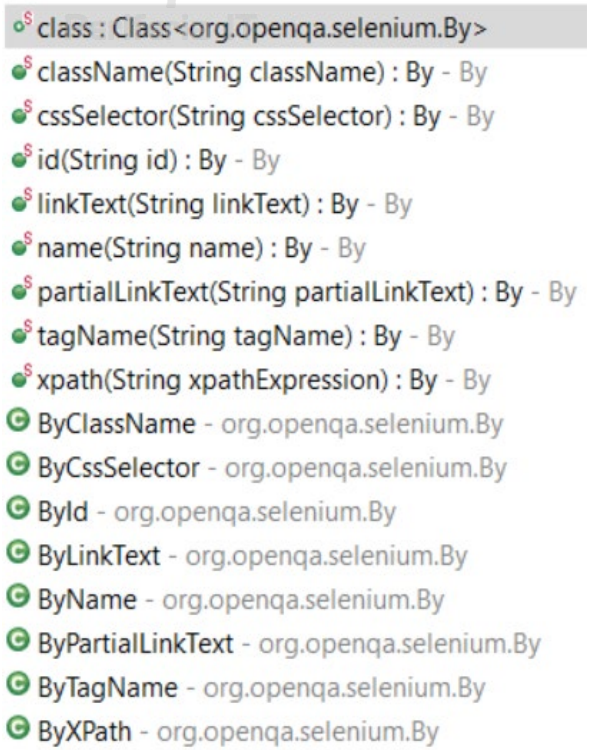
```
WebElement elementName = driver.findElement
                    (By.LocatorStrategy("LocatorValue"));
```

Este comando acepta el objeto “By" como argumento y devuelve un objeto WebElement .

El “By" es un localizador o un objeto de consulta y acepta el especificador de localizador. Entonces, si escribimos la línea "driver.findElement(By.)" , Eclipse proporcionará las siguientes estrategias de localización que podemos asociar con el objeto By.

```
driver.findElement(By.className("className"));

driver.findElement(By.cssSelector(".className"));

driver.findElement(By.id("elementId"));

driver.findElement(By.linkText("linkText"));

driver.findElement(By.name("elementName"));

driver.findElement(By.partialLinkText("partialText"));

driver.findElement(By.tagName("elementTagName"));

driver.findElement(By.xpath("xPath"));
```

¿CÓMO ENCONTRAR ELEMENTOS EN SELENIUM?



findElements() : este método encuentra una lista de elementos web en la página web. Devuelve una lista de elementos web que coinciden con los criterios especificados, a diferencia de findElement() que devuelve un elemento único. Si no hay elementos coincidentes, se devuelve una lista vacía.

```
List<WebElement> elementName = driver.findElements(By.LocatorStrategy("LocatorValue"));
```

ESTRATEGIA DE CARGA DE PÁGINA

Al navegar a una nueva página a través de la URL, de forma predeterminada, Selenium esperará hasta que el estado Listo del documento esté "completo".

Si una página tarda mucho en cargarse como resultado de la descarga de activos (p. ej., imágenes, css, js) que no son importantes para la automatización, puede cambiar el parámetro predeterminado NORMAL, EAGER, NONE,

Estrategia	Estado listo	notas
normal	completo	Usado por defecto, espera a que se descarguen todos los recursos
entusiasta	interactivo	El acceso DOM está listo, pero es posible que aún se estén cargando otros recursos, como imágenes
ninguna	Ningún	No bloquea WebDriver en absoluto

- NORMAL

```
ChromeOptions chromeOptions = new ChromeOptions();
chromeOptions.setPageLoadStrategy(PageLoadStrategy.NORMAL);
```

- EAGER

```
ChromeOptions chromeOptions = new ChromeOptions();
chromeOptions.setPageLoadStrategy(PageLoadStrategy.EAGER);
```

- NONE

```
ChromeOptions chromeOptions = new ChromeOptions();
chromeOptions.setPageLoadStrategy(PageLoadStrategy.NONE);
```

campus
Periferia IT

G R A C I A S

Periferia IT Group

Periferia IT Corp

CONTACT& BUSINESSIT

COOLEXPRESS