Modelos de Calidad de Software

Tema Nº9:Introducción al Selenium WebDriver – Parte 1

Indicador de logro Nº9:Realiza la configuración del entorno de trabajo del IDE Selenium Web Driver para automatización de pruebas funcionales.

**TEMA 01 Teoría de los**

Imagen que contiene Icono

Descripción generada automáticamente

**TEMA Nº9:**

Introducción al Selenium WebDriver – Parte 1

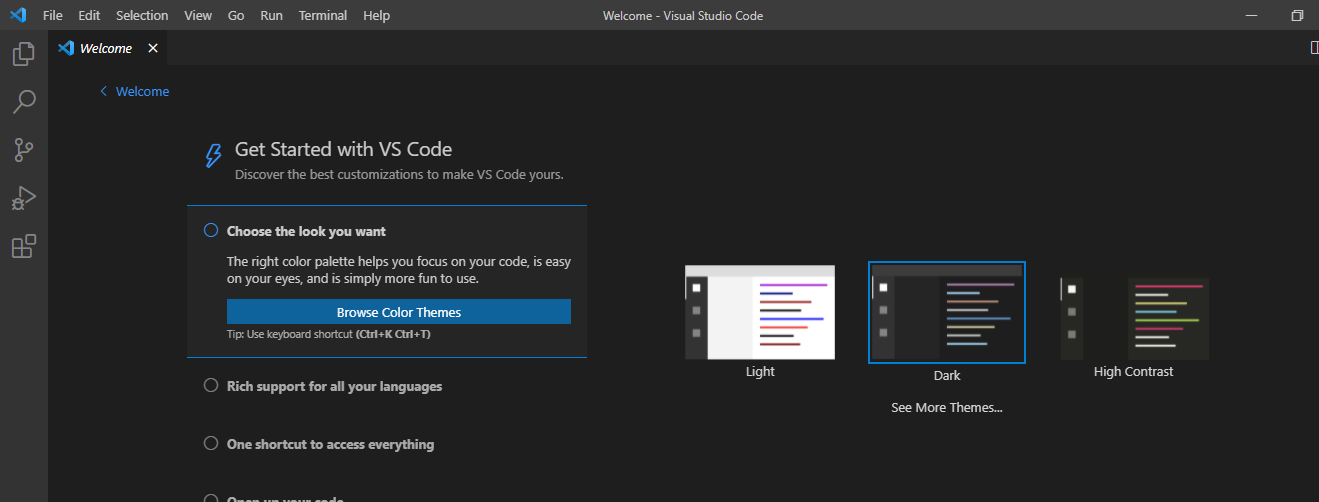
**Subtema 9.1:**

Conceptos

Selenium es un entorno de pruebas de software que se divide en IDE y el Selenium Web Driver donde:

* IDE: Versión StandAlone que podemos usar en el browser para grabar acciones (No es muy recomendable)
* Web Driver: Conjunto de API’s open source que podemos usar para hacer acciones sobre aplicación Web, sólo web (No Windows, no mac), soporta navegadores más importantes como Firefox, Chrome, IE, Edge, Opera, Safari

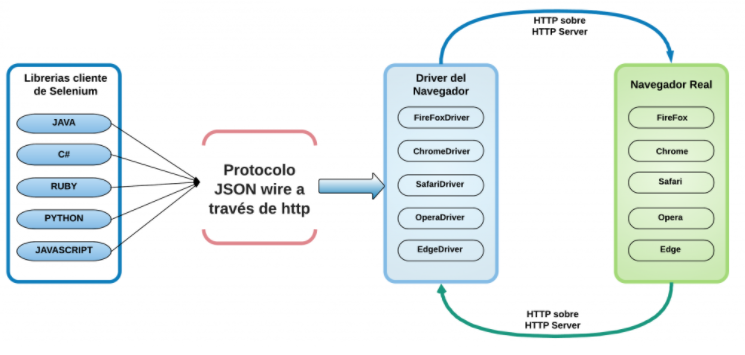
Como IDE (Ambiente integrado de desarrollo) que es donde escribes texto, código y ejecutas puedes seleccionar entre: Visual Studio Enterprise, Eclipse, VSC (Visual Studio Code), entre otros.



**Subtema 9.2:**

Arquitectura de Selenium

Siendo compatible con varios lenguajes de programación así como navegadores , a continuación graficamos la arquitectura.



Hay cuatro componentes básicos de la arquitectura de Selenium WebDriver:

* Librerías cliente de Selenium: Admite Java, Python, etc
* Protocolo JSON Wire : estándar abierto para el intercambio de datos en la web
* Drivers  de los navegadores : Usa controladores específicos para cada navegador para establecer una conexión segura con el navegador sin revelar la lógica interna de la funcionalidad del navegador
* Navegadores reales

**Subtema 9.3:**

Tipos de Selenium

Teniendo un conjunto de herramientas permite abordar distinto retos de automatización de pruebas:

* Selenium IDE: Permite grabar, editar y depurar todos los eventos del navegador, amigable para producir pruebas de manera fácil y rápida.
* Selenium Core : No permite grabar ni editar script, fácil integración con Selenium Remote Control
* Selenium Remote Control : permite escribir pruebas de UI de aplicaciones web automatizadas con lenguajes como Java, Python, C#
* Selenium Web Driver: Conjunto de API’s open source que podemos usar para hacer acciones sobre aplicación Web, sólo web (No Windows, no mac), soporta navegadores más importantes como Firefox, Chrome, IE, Edge, Opera, Safari

**Subtema 9.4:**

Drivers selenium según explorador

Selenium soporta los navegadores más usados en el mercado, emulando las interacciones de usuarios reales.

Se tienen Drivers para:

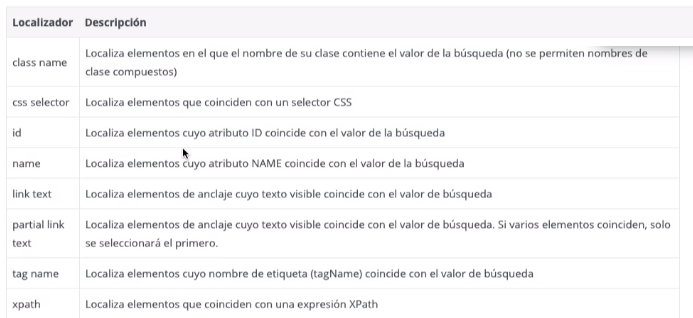


**Subtema 9.5:**

Localizadores

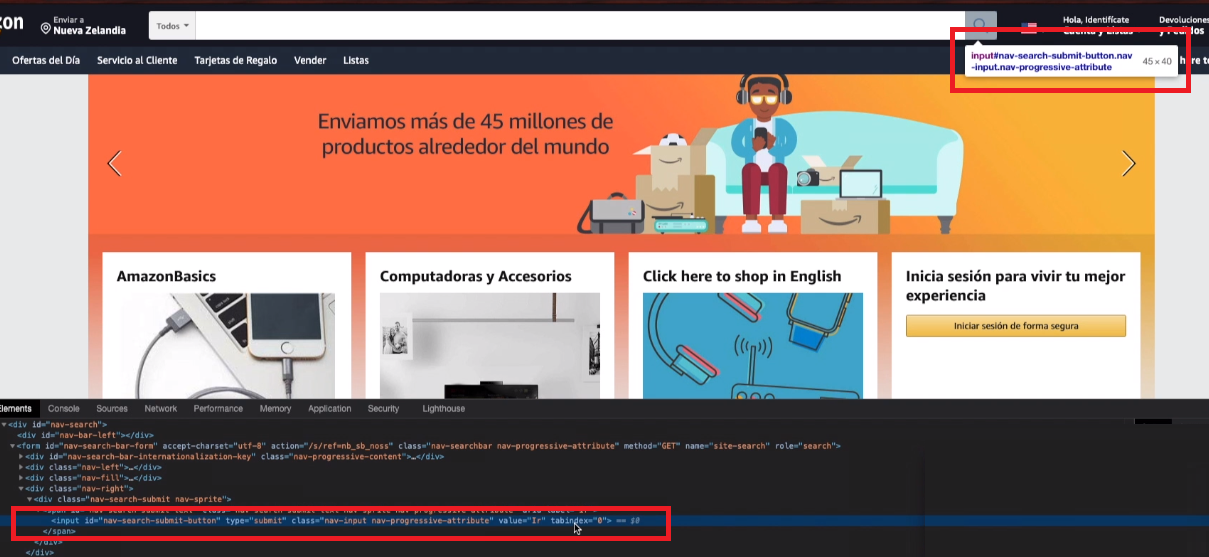
Locators son la dirección o el domicilio para poder llegar a los Web Elements con los cuales necesitamos interactuar (botón, dropdown, etc), al mismo tiempo estos We Elements están dentro del DOM (Html donde están todos los elementos)

Siendo un conjunto de parámetros con los cuales podemos identificar un elemento en una página, los tipos de locators que tenemos:



Hay varios tipos de locators pero se trata de trabajar con los Id’s, en teoría el ID es un locator único, pero en la practica el ID es dinámico.

Ver los Locators con Clic derecho e inspeccionar.



**Subtema 9.6:**

Principales Comandos

Hay 3 tipos de comandos:

1. Acciones (Actions): Modifican el estado de la aplicación, por ejemplo: pulsar un botón/ enlace.
2. Accesos (Accessors): Comprueban el estado de la aplicación y almacenan el resultado en variables. Por ejemplo: «Store Title»—> Devuelve el título de la página actual.
3. Afirmaciones (Assertions): Son como los anteriores, pero estos comprueban el estado de la aplicación con lo que se esperaba. Por ejemplo: Comprobar que el título de una página es «Selenium: Comandos Selenese».

Principales comandos:

|  |  |
| --- | --- |
| Commands | Descriptions |
| assignId(”Locator”,”String”) | Temporarily sets the “id” attribute of the specified element |
| capture Screenshot (”File name”) | Captures a PNG screenshot to thespecified file. |
| Check(”Locator”) | Check a toggle-button(checkbox/radio) |
| click(”Locator”) | Clicks on a link, button, checkboxor radio button. |
| clickAt(”Locator”,”Coordinate String”) | Clicks on a link, button, checkboxor radio button. |
| close() | Simulates the user clicking the”close” button in the title bar of a popup window or tab. |
| doubleClick(”Locator”) | Double clicks on a link, button,checkbox or radio button. |
| doubleClickAt(”Locator”,”Coordinate String”) | Double clicks on a link, button,checkbox or radio button. |

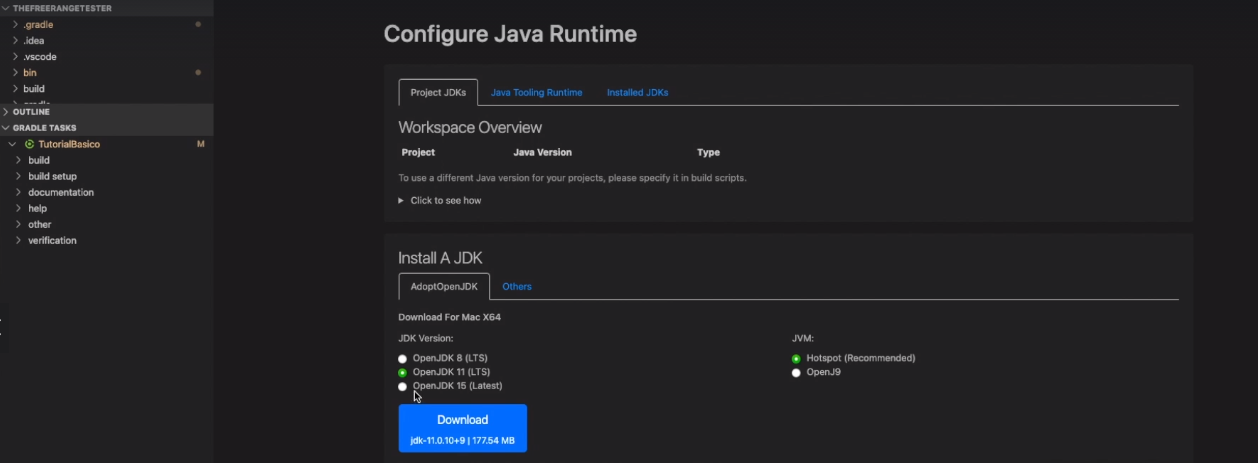
**Subtema 9.7:**

Configuración del entorno de pruebas

Descargar e Instalar:

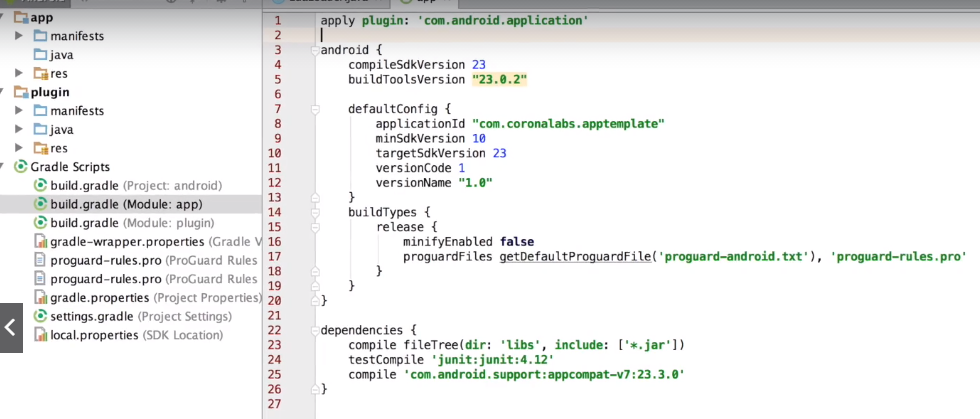
* Instalar VSC (Visual Studio Code) para Windows o Mac
* Descargar ChromeDriver (No se instala, se ejecuta)
* Instalar JDK
* Instalar las extensiones Gradle Language Support y Grade Task en VSC

Al abrir por primer vez confirmar la versión de Java con CTRL+SHIFT+P en Windows, Java Configure Runtime



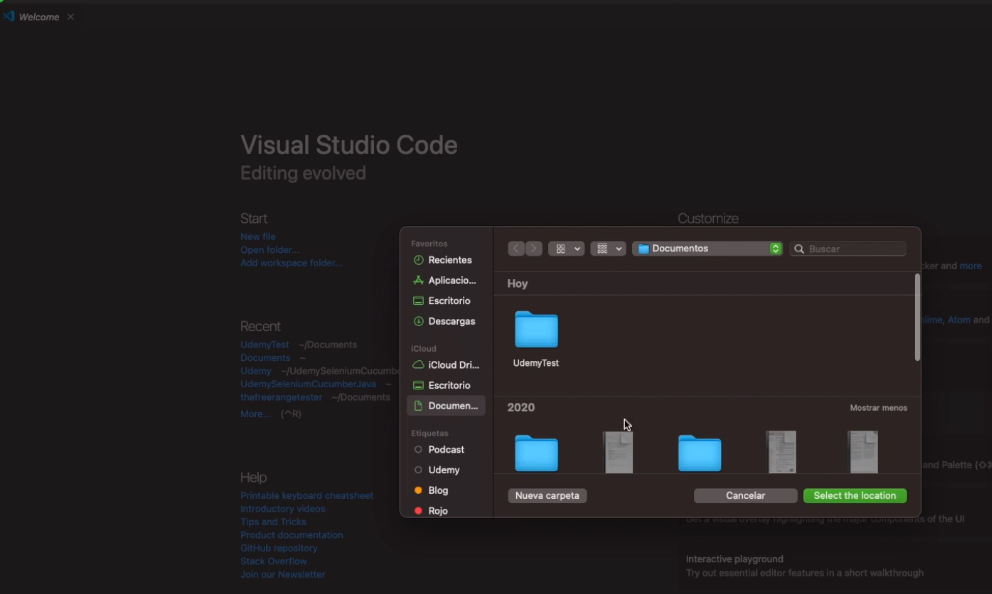
Ahora vemos nuestra Building Tool – Gradle (o Maven)

Gradel nos resuelve las dependencias, todo lo que necesitas como librerías Gradel lo trae e instala y lo deja andando (Selenium es una dependencia)



Una dependencia o librería ya está hecho para usarse, ejemplo librería para usar selenium, esto permite ganar tiempo y eficiencia.

Listo para iniciar la creación.



Actividad:

Revisa y analiza el enlace sugerido\* en la plataforma virtual y luego realiza la actividad propuesta:

* Realice la configuración del entorno de pruebas usando gradle, java y selenium web driver según lo revisado en clase.

\*Cada docente deberá proporcionar el enlace que considere apropiado para la ejecución de la actividad.