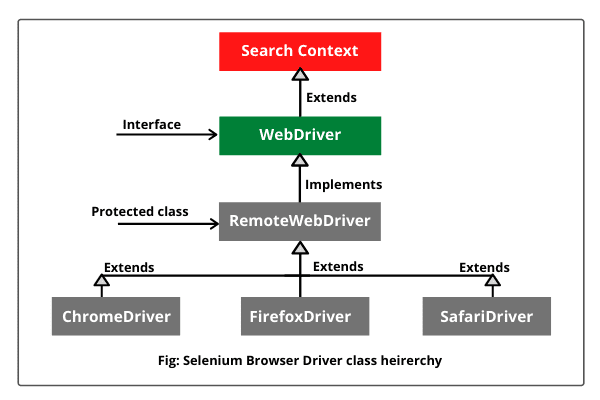
**Day - 1**

**Introduction on Selenium**

Selenium is an open-source framework for automated browser testing and web application automation, offering a suite of tools to interact with web browsers programmatically

**Hierarchy of Selenium Webdriver.**

****

**SearchContext -** is an Interface with 2 Abstract methods - FindElement() and FindElements().

**WebDriver -** is an interface which has 11+2 abstract methods.

**RemoteWebDriver -** is a protected class which is an implementation class for webdriver. It has 13 concrete methods.

Remote webdriver implements Webdriver.

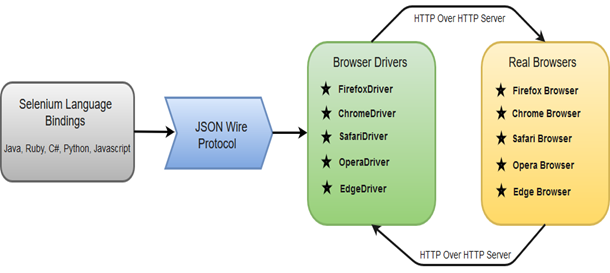
**ChromeDriver - A** [**WebDriver**](https://www.selenium.dev/selenium/docs/api/java/org/openqa/selenium/WebDriver.html) **implementation that controls a Chrome browser running on the local machine. It requires a chromedriver executable to be available in PATH.**

**ChromeDriver extends RemoteWebDriver**

**Selenium Architecture**

Selenium WebDriver API provides communication facility between languages and browsers.

The following image shows the architectural representation of Selenium WebDriver.

****

### Selenium Language Bindings / Selenium Client Libraries

Selenium developers have built language bindings/Selenium Client Libraries in order to support multiple languages. For instance, if you want to use the browser driver in java, use the java bindings.

### JSON Wire Protocol

JSON (JavaScript Object Notation) is an open standard for exchanging data on the web. It supports data structures like objects and arrays.

### Browser Drivers

Selenium uses drivers, specific to each browser in order to establish a secure connection with the browser without revealing the internal logic of browser's functionality. The browser driver is also specific to the language used for automation such as Java, C#, etc.

**Different kind of Selenium components**

Selenium comprises four main components: Selenium IDE (Integrated Development Environment) for record-and-playback, Selenium WebDriver for programmatic browser control, Selenium Grid for parallel test execution, and Selenium Standalone Server to facilitate communication between the WebDriver and browsers

**Selenium WebDriver:** Allows programmatically interacting with web browsers to automate testing

**Selenium IDE (Integrated Development Environment):** A browser extension for recording and playback of browser interactions

**Selenium RC** (Remote Control) was the predecessor to WebDriver, providing a server-based architecture for automated web testing but is now deprecated in favor of WebDriver.

**Selenium Grid:** Enables parallel test execution across multiple machines and browsers.

**Selenium Standalone Server:** Facilitates communication between the WebDriver and browsers

**Selenium Installation**

Installation of Selenium web driver, making sure all Environmental variables are set and installing selenium library in IDE.

**How to install Selenium -**

1. Downloaded Selenium Webdriver from Official Selenium Website.
2. Extracted Selenium files
3. In Eclipse, Properties of Project → Java Build path → Libraries —> Add External JARS→ Selected and opened all the Selenium files that we extracted.--> Apply and close.
4. Download Chromedriver (Make sure to download the latest version of chromedriver by checking against your chrome version / the chrome version used in your project)
5. We include the chromedriver file path in your script.

**First Selenium Script :**

public class FirstScript {

public static void main(String[] args) {

//System.setProperty("webdriver.chrome.driver", "G:\\chromedriver\_Version\\chromedriver.exe");

System.setProperty("webdriver.chrome.driver", "G:\\chromedriver\_Version\\chromedriver.exe");

**WebDriver driver = new ChromeDriver();**

driver.get("https://www.google.com/");

}

}

**System.setproperty** - It is used to enter the details of the driver(chromedriver,firefoxdriver,edgedriver) in a key value pair of Setproperty(key, value) key is the driver detail, value is the location of

the driver

**Web driver** is an **Interface**, driver is the reference variable, new is the keyword, ChromeDriver is the Constructor, together new **ChromeDriver**() object is created.

and stored in reference variable **driver**.

Here, All the driver methods are just declared in **Webdriver** Interface, and all the driver methods(get,close,quit,findelement) are implemented in

Implementation class called **ChromeDriver**();

**Locators**

**I**n Automation , before performing any action such as click, clear, pass the data (send keys) we need to find the address of the element.

We do that by inspecting the element,

To find the element, we use Locators

Locators are classified into 8 types.

1. ClassName.
2. CSS selector.
3. ID.
4. LinkText.
5. Name
6. PartialLink text.
7. TagName
8. X path

All of the Locators are static methods of By class.

By class is an abstract class.

Official info about Locators in **Selenium website** :

<https://www.selenium.dev/documentation/webdriver/elements/locators/>

**Xpath**

It is a path of the element in the HTML tree structure.

Xpath Syntax - //Tagname[@key = ‘Value’];

**Xpath example** - **//input[@id='userName']**

**Absolute Xpath - only Forward slash** - Used to navigate from parent to immediate child tag.

**Relative Xpath -**

**Double forward slash** - Used to travel directly to the specified tag.

We can also use combination of **Relative** and **Absolute** Xpaths.

**How to write Xpath with Multiple attributes.**

USing ‘**and’** operator

**Example** - //input[@class='nav-input nav-progressive-attribute' and @type="text"]

**How to use Text Function in Xpath -**

**Syntax :**

//tagname[text()=’Text Value’]

Example in Gmail SIgninPage :

//span[text()='Create account']

**How to write Xpath using Indexing -**

Syntax - **(//tag[@attribute=’Value’])[index]**

**Contains Function**

**Syntax -**

//Tag[contains(text(),’textvalue’)]

Example -

(//div[@class="\_fluid-quad-image-label-v2\_style\_fluidQuadImageLabelBody\_\_3tld0"]//img)[3]

//a[contains(text(),'Grocery ')]

//\*[contains(text(),'Forgot')]

**Combination of Contains as well as Indexing in Xpath** (//\*[contains(text(),'akshay')])[2]

**How to use By.id in selenium script?**

WebElement username = driver.findElement(By.id("userName"));

**How to use By.name in selenium script?**

driver.findElement(By.name("checkBoxOption1"));

**Note :**

Whenever we give a incorrect Locator, we get **NoSuchElement Exception.**

**13 Methods used from Webdriver Instance :**

1. close()
2. findElement()
3. findElements()
4. get()
5. getCurrentUrl()
6. getPageSource()
7. getTitle()
8. getWindowHandle()
9. getWindowHandles()
10. manage()
11. navigate()
12. quit()
13. switchTo()

**Web Element Commands: Edit Box, Button, Check box, Radio Button.**

**How to handle textbox :**

**Locate the Textbox/EditBox**

Use one of the available methods to locate the textbox element. Common methods include

**findElement(By.id())**,

**findElement(By.name())**,

**findElement(By.xpath())**

**Type into the Textbox using sendkeys() :**

textbox.sendKeys("Hello, Selenium!");

**How to click on a button.**

First we have to Locate the Element , then we store it into a variable of WebElement

Then we click on it using click().

WebElement button = driver.findelement(By.id("button\_id"));

button.click();

**How to select a checkbox**

First we have to Locate the Element , then we store it into a variable of WebElement

Then we click on it using click().

WebElement checkbox = driver.findelement(By.id("checkbox\_id"));

checkbox.click();

**How to select a RadioButton**

First we have to Locate the Element , then we store it into a variable of WebElement

Then we click on it using click().

WebElement radio = driver.findelement(By.id("radio\_id"));

radio.click();

### **Select in Selenium WebDriver**

The 'Select' class in Selenium WebDriver is used for selecting and deselecting option in a dropdown. The objects of Select type can be initialized by passing the dropdown webElement as parameter to its constructor.

WebElement DropDown = driver.findElement(By.id("testingDropdown"));

Select sel = **new** Select(DropDown);

WebDriver provides three ways to select an option from the drop-down menu.

1. **selectByIndex** - It is used to select an option based on its index, beginning with 0.

sel.selectByIndex(5);

**2 selectByValue** - It is used to select an option based on its 'value' attribute