Selenium with Java

Learn Tekie's – 2 Day

Mentor – Nivedha

Today's Agenda

- Selenium Introduction
- Launch Browser
- Locators
- Select Class
- Absolute Xpath
- Basic Xpath
- Advance Xpath

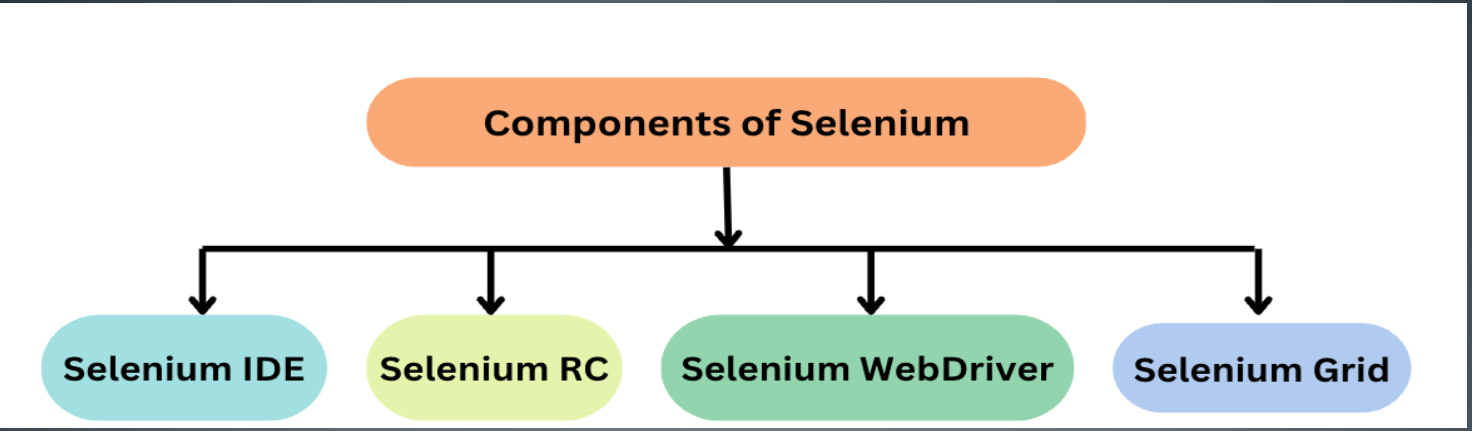
What is Selenium Web driver?

- Selenium Web Driver is a browser automation framework that accepts commands and sends them to a browser. It is implemented through a browser-specific driver.
- It directly communicates with the browser and controls it.
- Selenium Web Driver supports various programming languages like Java, C#, PHP, Python, Perl, Ruby. and Java script.
- Selenium Web driver supports most of the browser driver APIs like Chrome, Firefox, Internet Explorer, Safari.
- Selenium can Run on Window, Linux, Mac OS Platform

Tool – Scripting Languages

Selenium	Playwright	Cypress	Puppeteer
Core Java, Python, C#, JavaScript, Ruby, Perl	Core Java, Python, C#, JavaScript	JavaScript	JavaScript

Components Of Selenium



- Integrated Development Environment, It is implemented as a Firefox Add-On and as a Chrome Extension.
- Remote Control is a server, Out of date
- Selenium Web driver - This is implemented through a browser-specific browser driver, which sends commands to a browser and retrieves results. Selenium Web Driver does not need a special server to execute tests. Instead, the Web Driver directly starts a browser instance and controls it.
- Selenium Grid - Grid is used to run single script with multiple machine

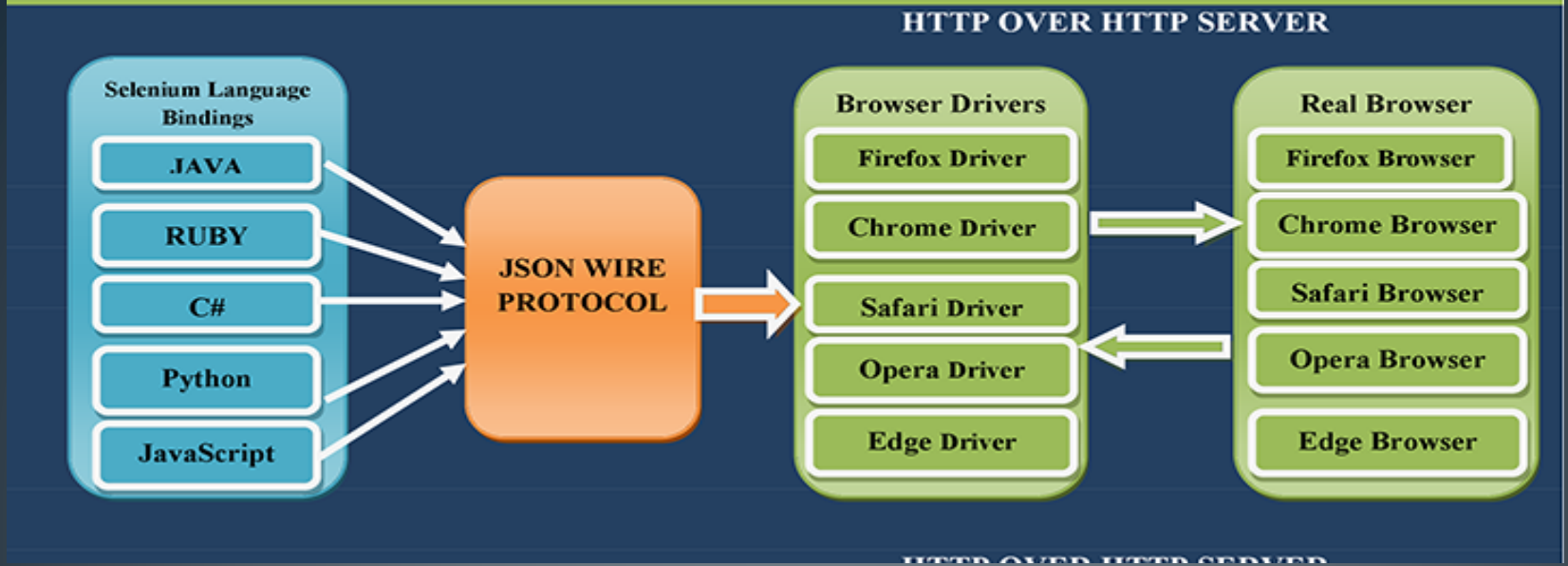
Advantage of Selenium

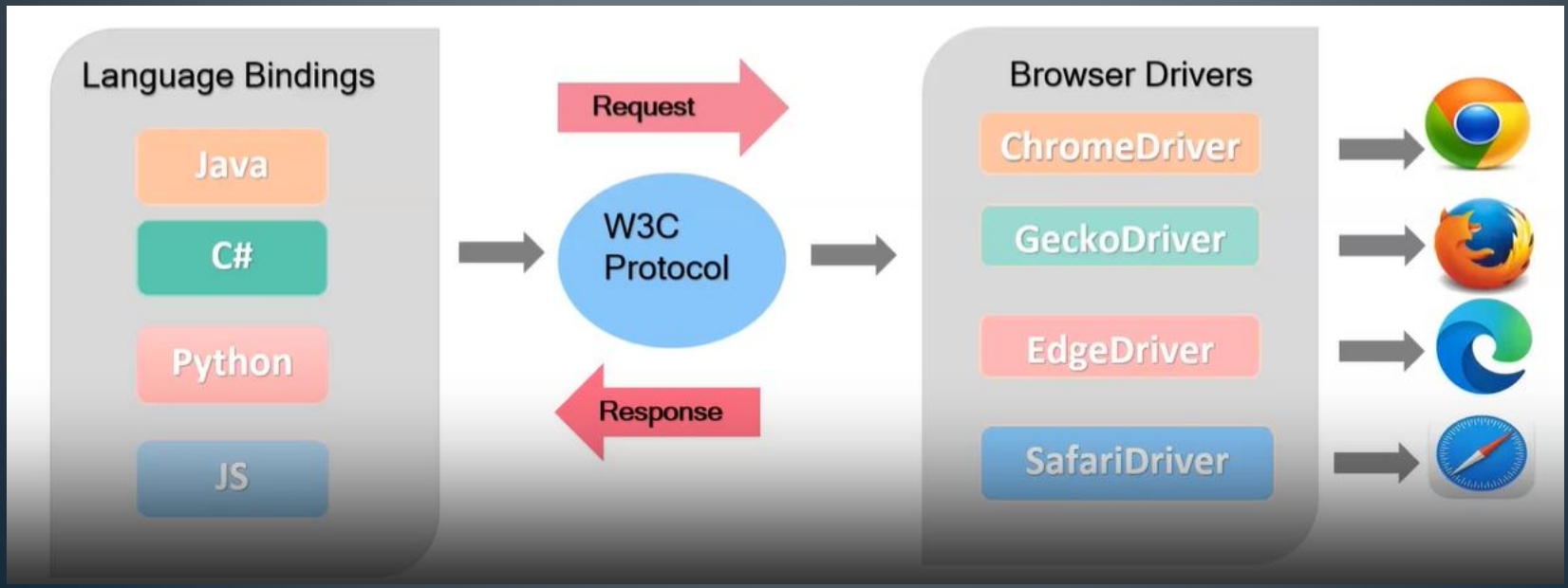
- It's supports variety of languages that include Java, Perl, Python, C#, Ruby, Java Script, and VB Script. etc.
- supports many operating systems like Windows, Macintosh, Linux, Unix etc.
- Selenium supports many browsers like Internet explorer, Chrome, Firefox, Opera, Safari etc.
- Selenium can be integrated with ANT or Maven kind of framework for source code compilation.
- Selenium can be integrated with TestNG testing framework for testing our applications and generating reports.
- Selenium can be integrated with Jenkins or Hudson for continuous integration.
- Selenium can be integrated with other open source tools for supporting other features(Cucumber).
- Selenium supports very less CPU and RAM consumption for script execution.

Disadvantages of Selenium

- Selenium only supports web based application and does not support windows based application.
- It is difficult to test Image based application, QR code.
- Selenium need outside support for report generation activity like dependence on TestNG or Jenkins.
- Not an All-in-One Solution – Requires 3rd Party Tool Bindings
- Difficult to use, and you need programming skills to create tests.

Selenium Web Driver Architecture

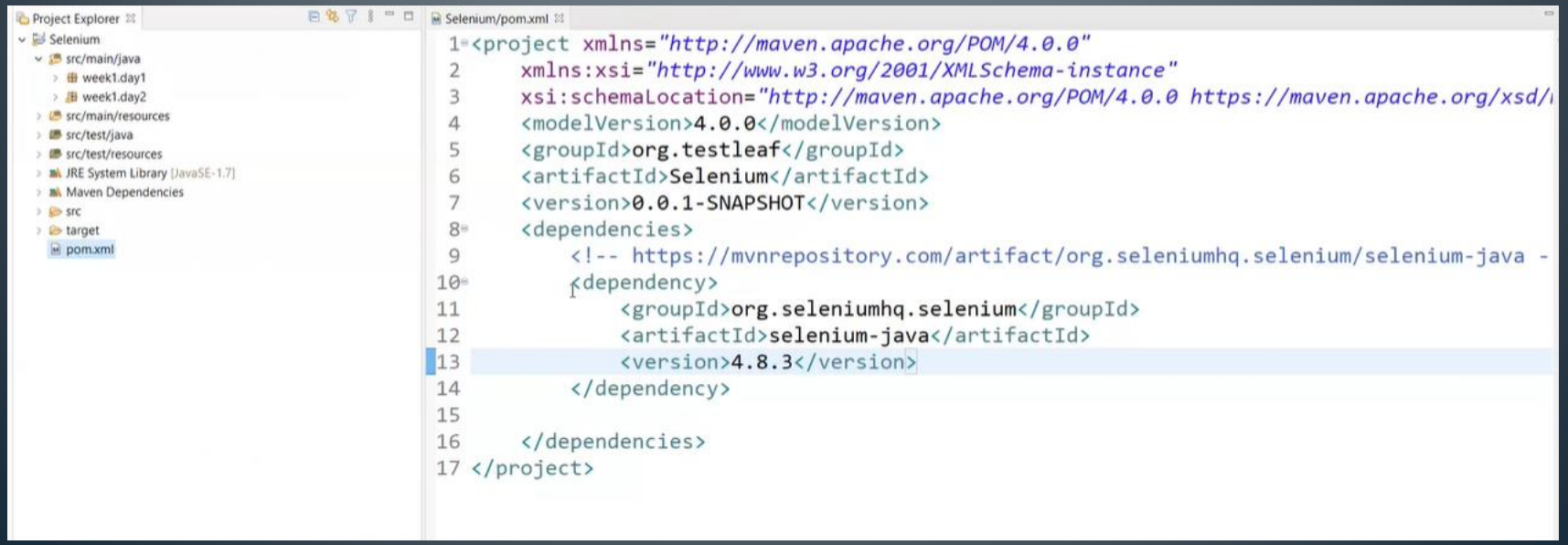




Maven Project

- In Maven Project , you can get pom.xml file, you can add dependencies by using mvnrepository.com
- First Add Selenium – Java

```
<dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>4.16.1</version>
</dependency>
```

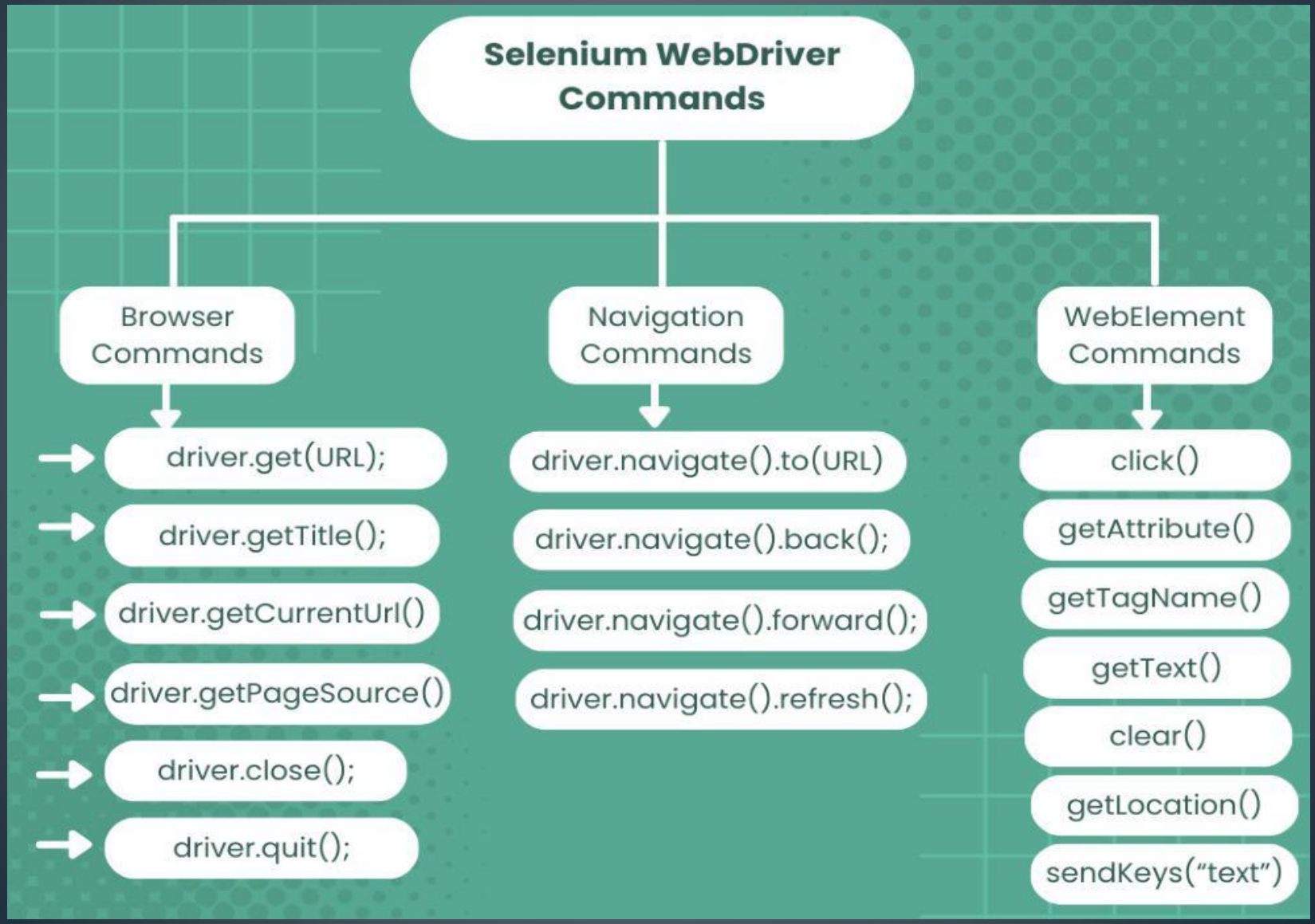


Launch Browser

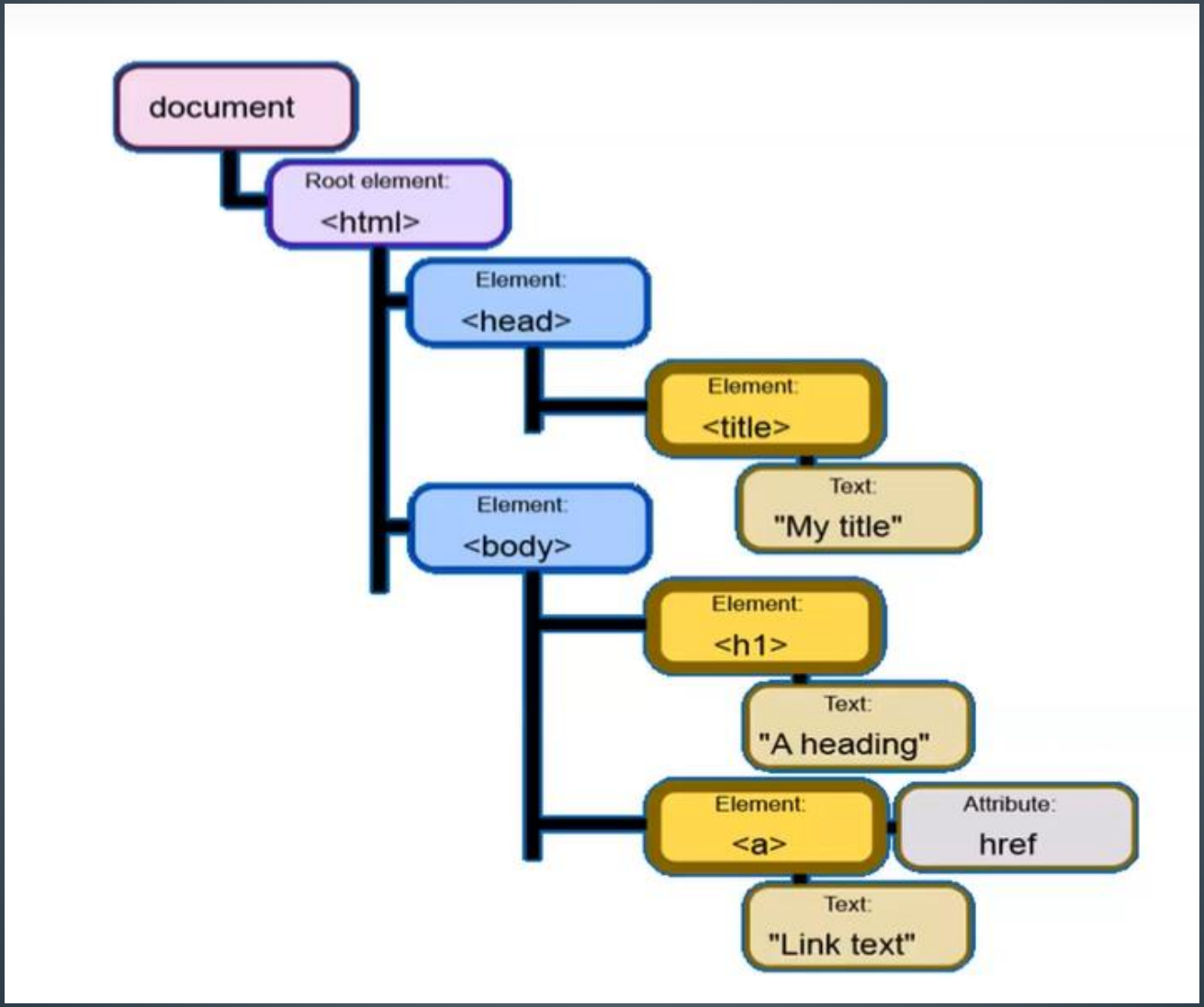
- Expand the selenium Project
- Src/main/java, Right Click here select New => create the java package Day1
- then package day1, Right click here select New => create the Class Name Click Finish
- Before 4.6.0 Webdrivermanager are used..
  WebDriverManager.chromedriver().setup();

```
1  package week2.day1;
2
3  import org.openqa.selenium.chrome.ChromeDriver;
4
5  public class LaunchBrowser {
6      public static void main(String[] args) {
7          //To launch the browser(Chrome browser)
8          //Create object for ChromeDriver
9          ChromeDriver driver = new ChromeDriver();
10     }
11
12 }
```

Selenium Commands



HTML Structure

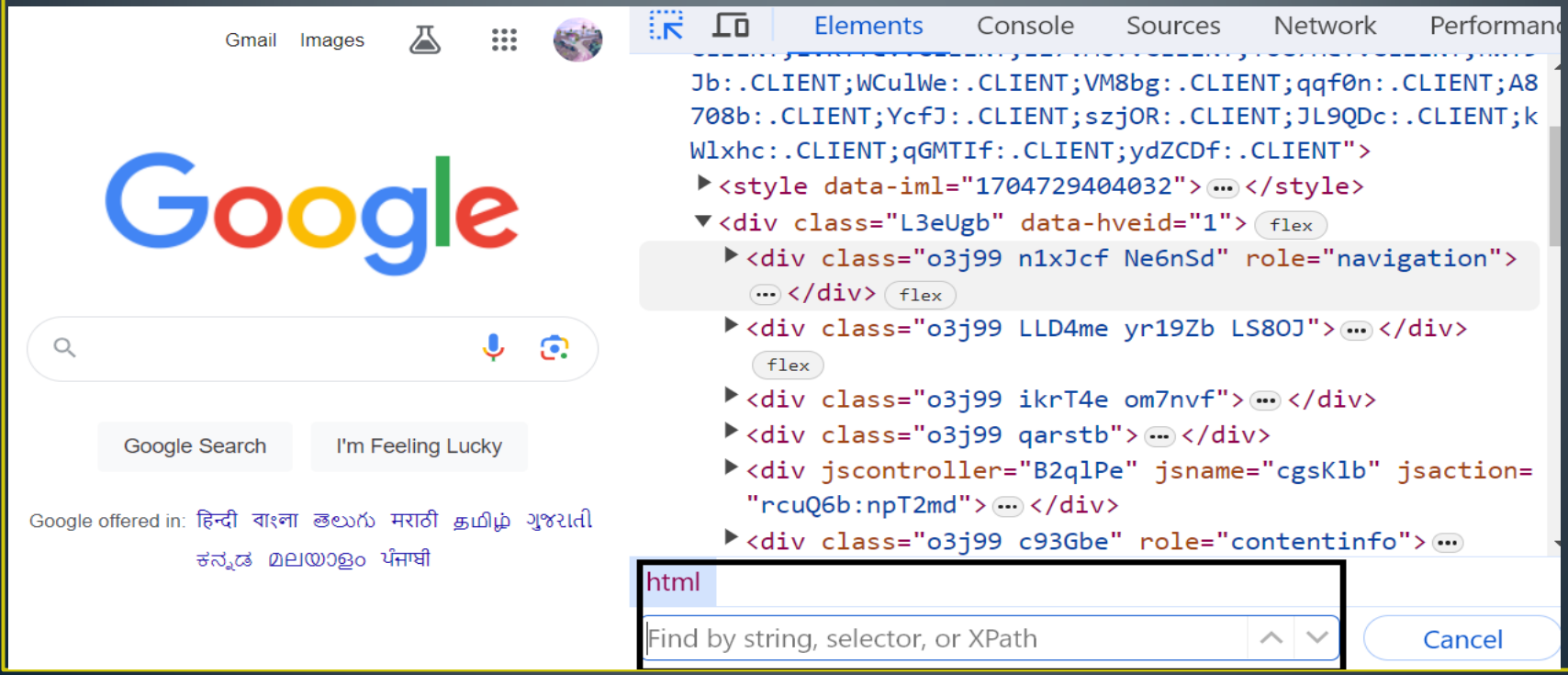


HTML Tags

HTML tags	Description
<html>	Defines the root of the webpage
<a>	Defines a hyperlink
<button>	Defines a clickable button
	Defines a section in a document
<iframe>	Defines an inline frame
<img>	Defines an image
<input>	Defines an input control
<option>	Defines options of the dropdown list
<select>	Defines a drop-down list
	Defines a section in a document
<table>	Defines a table

Find Locator

- Selenium communicate with Webpage – Through HTML Document
- DOM – Document Object Model – Representation of various component of browser, Logical structure of a web document
- Why to understand DOM - DOM helps to identify the element of webpage
- Click Ctrl + Shift + C to Open Inspector and then Click Ctrl + F



Attribute and Value

```
<form id="login" method="post" action="/opentaps/control/login">
    <p class="top">
      <label for="username">Username</label>
      <input class="inputLogin" type="text" id="username" name="USERNAME" size="50">
    </p>
     <p>
      <label for="password">Password</label>
      <input class="inputLogin" type="password" id="password" name="PASSWORD" size="50">
    </p>
    <p>
       <input class="decorativeSubmit" type="submit" value="Login">
    </p>
 </form>
```

To Locate Element in DOM

Locators	Find different elements on web page
id	To find the element by using id attribute of the element
name	To find the element by using name attribute of the element
className	To find the element by using Class attribute of the element
linkText	To find the element by text of the WebElement having <a>
partialLinkText	To find the element with link text with partial match
tagName	To locate the element using tag name of the target element
xpath	To find the dynamic element and traverse between various elements of the web page
cssSelector	CSS path also locates elements having no name, class or ID.

How to use Locator

Basic Locators

-- id → can be identified as id attribute of an element
-- name →can be identified with name attribute
-- className → can be identified as class attribute
-- linkText → can be identified with text with <a>

xpath

Xml path -> uniquely identify or address parts of an XML document
-> to navigate through a page's HTML structure and locate the element

Syntax: driver.findElement(By.*locator("value"));*

DO & Don't

Locators	When do you use?	When you cannot use?
id	Always (Most preferred)	Numbers
name	If "id" does not exist	Duplicate
className	When class is unique without white spaces	White spaces, Duplicate
linkText	If it is a link	Duplicate
tagNname	For collection of objects	Duplicate
Xpath	If none of the above, works	Xpath can change
cssSelector	This is the last option you have!	CSS can change

Browser Based Commands

- maximize browser window in Selenium WebDriver

```
driver.manage().window().maximize()
```

- delete cookies in Selenium

```
driver.manage().deleteAllCookies()
```

- Get cookies for Current Browser

```
driver.manage().getCookies()
```

Check WebElement is Displayed, Enabled & Selected Methods?

- isDisplayed()
- isEnabled()
- isSelected()

Clear the Text

Clear()

Capture the Text

getText()

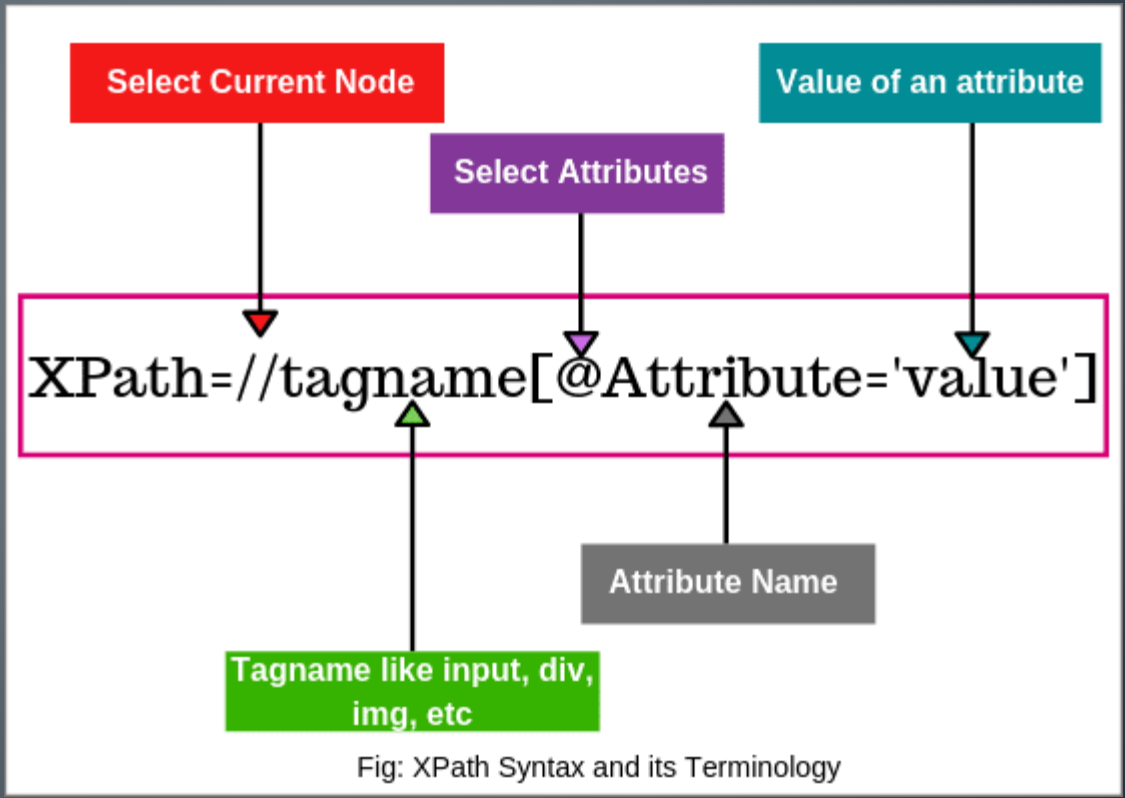
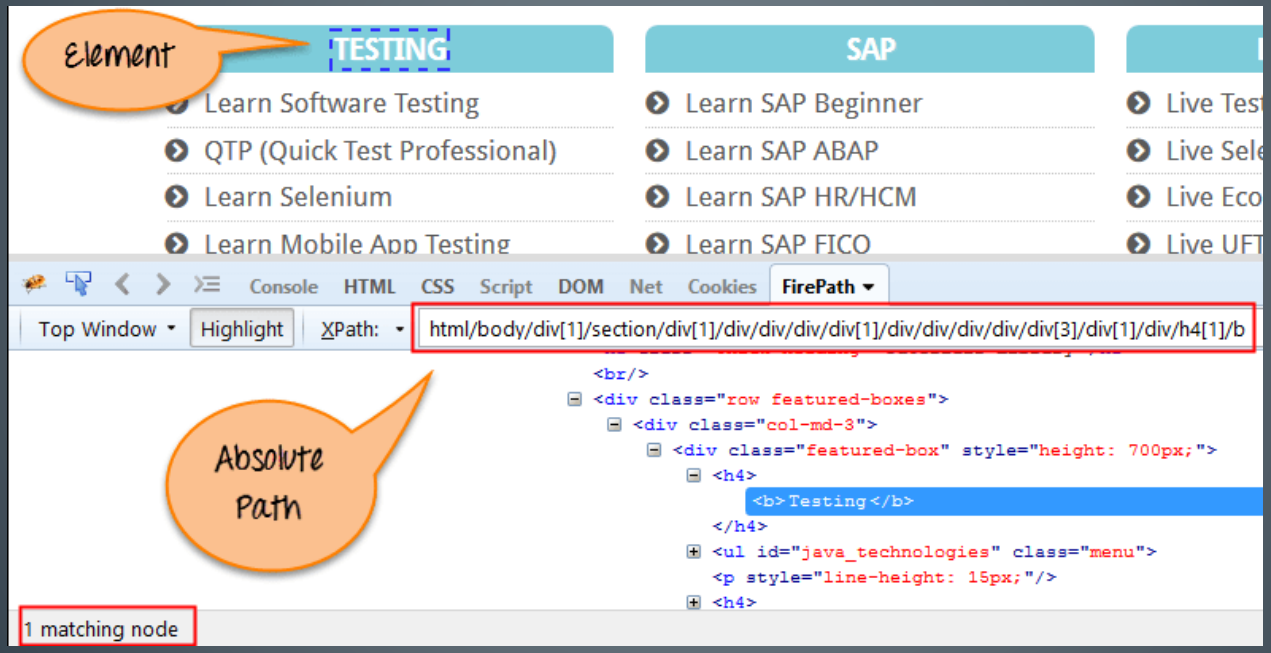


Fig: XPath Syntax and its Terminology

Xpath

- XPath in Selenium is a technique that allows you to navigate the structure of a webpage's HTML.
- **Absolute XPath:** Begins from the root of the HTML document and specifies the complete path to the element. It's not as flexible and can break if the page structure changes.



- **Relative XPath:** Starts from a specific element and navigates through the DOM hierarchy to locate the desired element. It's more flexible and resilient to changes in the page structure.

Xpath Axis

- Child to Child => //Child
- Child to parent =>//Parent
- Grand Parent to Grand child => //
- Grand Child to Grand Parent =>//Ancestor
- Elder Cousin to Younger Cousin => //Following[After the Node]
- Elder Sibling to Younger Sibling => //Following-Sibling
- Younger Cousin to Elder Cousin => //Preceding[Before the Node]
- Younger Sibling to Elder Sibling =>//Preceding-Sibling

Handle Drop Down In Selenium

Dropdown with Select Class - selectByindex(int index)

- selectByVisibleText(String text)
- selectByValue(String value)

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.Select;

import io.github.bonigarcia.wdm.WebDriverManager;

public class HandleDropDown {

    public static void main(String[] args) {

        WebDriverManager.chromedriver().setup();
        WebDriver driver=new ChromeDriver();

        driver.get("https://www.opencart.com/index.php?route=account/register");

        WebElement drpCountryEle=driver.findElement(By.id("input-country"));

        Select drpCountry=new Select(drpCountryEle);

        //drpCountry.selectByVisibleText("Denmark");

        drpCountry.selectByValue("10");  //Argentina
```

```
public void selectByList() {

      WebDriver driver = new ChromeDriver();

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

   driver.findElement(By.xpath("//select")).click();

      List<WebElement> allOptions = driver.findElements(By.cssSelector("select option"));

      String option = "Highest to lowest";

      // Iterate the list using for loop

      for (int i = 0; i < allOptions.size(); i++) {

            if (allOptions.get(i).getText().contains(option)) {

                  allOptions.get(i).click();

                  System.out.println("clicked");

                  break;

            }

      }
```

FindElement Vs FindElements

FindElement

- Returns a single WebElement
- Syntax: WebElement findElement (By.locator())

FindElements

- Does not throw any exception
- Returns an Empty List of WebElement Object
- Syntax: java.util.List<WebElement>
- findElements(By.locator())

Mouse Actions

- click()
- clickAndHold()
- contextClick()
- doubleClick()
- dragAndDrop(source, target))
- dragAndDropBy(source, xoffset,y-ffset)
- moveToElement(toElement)
- moveByOffset(x-offset, y-offset).release()

KEYBOARD ACTIONS & LINK

- sendKeys()
- keyUp()
- keyDown()
- driver.findElement(By.linkText("click here")).click();
- driver.findElement(By.partialLinkText("here")).click();

Handle Alert

- Simple Alert
- Prompt Alert
- Confirmation Alert

Commands

- driver.switchTo().alert().dismiss();//Close
- driver.switchTo().alert().accept();//Ok
- driver.switchTo().alert().getText();//Capture Alert Text
- driver.switchTo().alert().sendKeys("Text");//Send Text in Alert

Synchronization problem in Selenium

- Thread.sleep(2000)//milsec
- Implicit Wait
- Explicit Wait
- Fluent Wait
- driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

- WebDriverWait wait = new WebDriverWait(driver,30);
- wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("//div[contains(text(),'COMPOSE')]")));

CAPTURE SCREENSHOT

- //Convert web driver object to TakeScreenshot
- TakesScreenshot scrShot =((TakesScreenshot)webdriver);
- //Call getScreenshotAs method to create image file
- File SrcFile=scrShot.getScreenshotAs(OutputType.FILE);
- //Move image file to new destination
- File DestFile=new File(fileWithPath);
- //Copy file at destination
- FileUtils.copyFile(SrcFile, DestFile)

Open a Link in New Tab

- String Tab = Keys.chord(Keys.CONTROL,Keys.RETURN);
- Driver.findElement(By.id('2')).sendKeys(Tab);
- Driver.switchTo().newWindow(WindowType.TAB)

Open a Link in Multiple Tab & Windows

- Driver.switchTo().newWindow(WindowType.WINDOW);

Thank You