## 🌐 Selenium Features

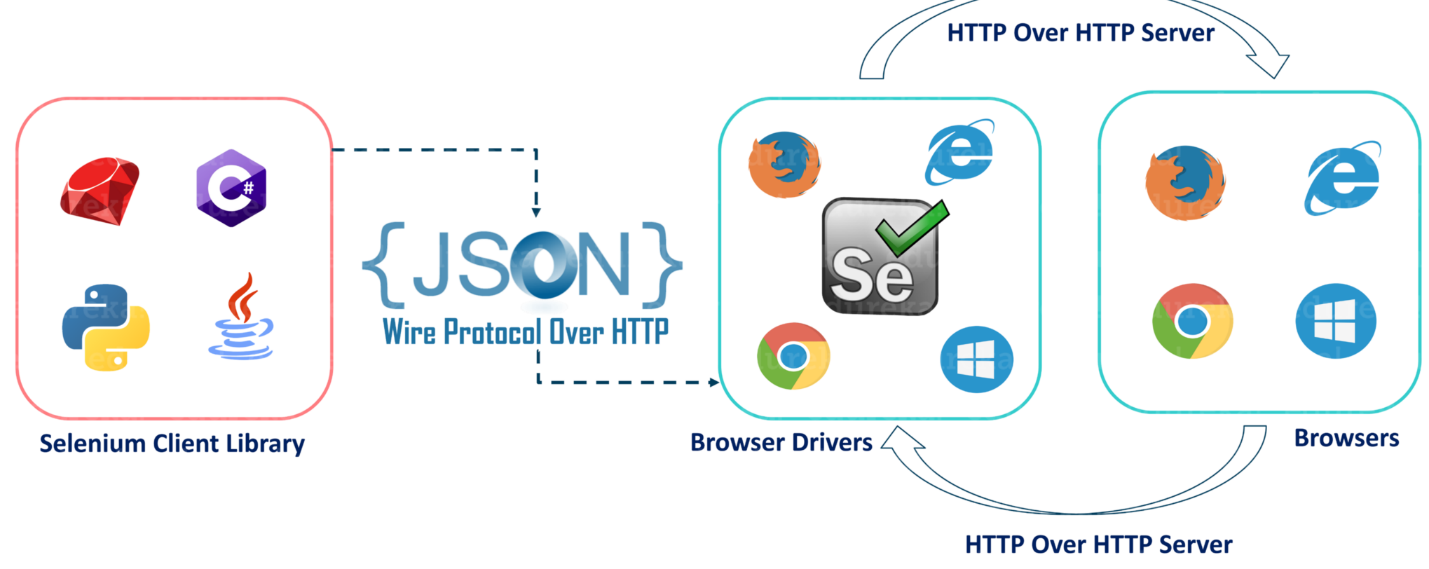
* **✅ Open Source Automation Testing Tool**
* **🌍 Exclusively for Web-Based Applications**
* **🧭 Supports Multiple Browsers: Chrome, Firefox, Internet Explorer, Safari**
* **💻 Works Across Platforms: Windows, macOS, Linux**
* **🧑‍💻 Supports Multiple Languages: Java, C#, Python, JavaScript, PHP, Ruby**

## 🔄 Selenium vs WebDriver

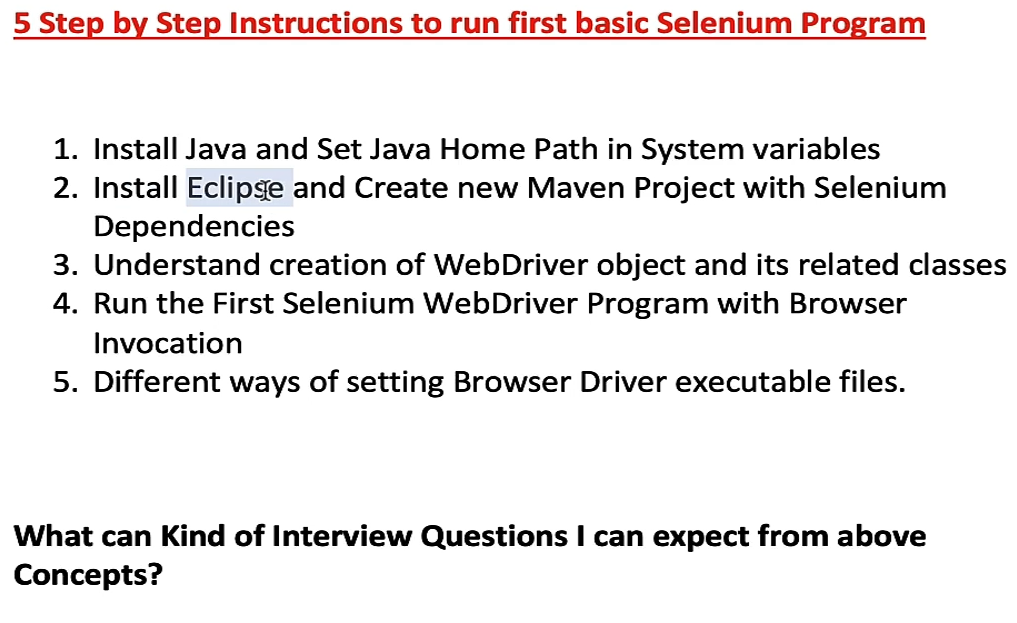
### Selenium WebDriver Architecture (Simplified)

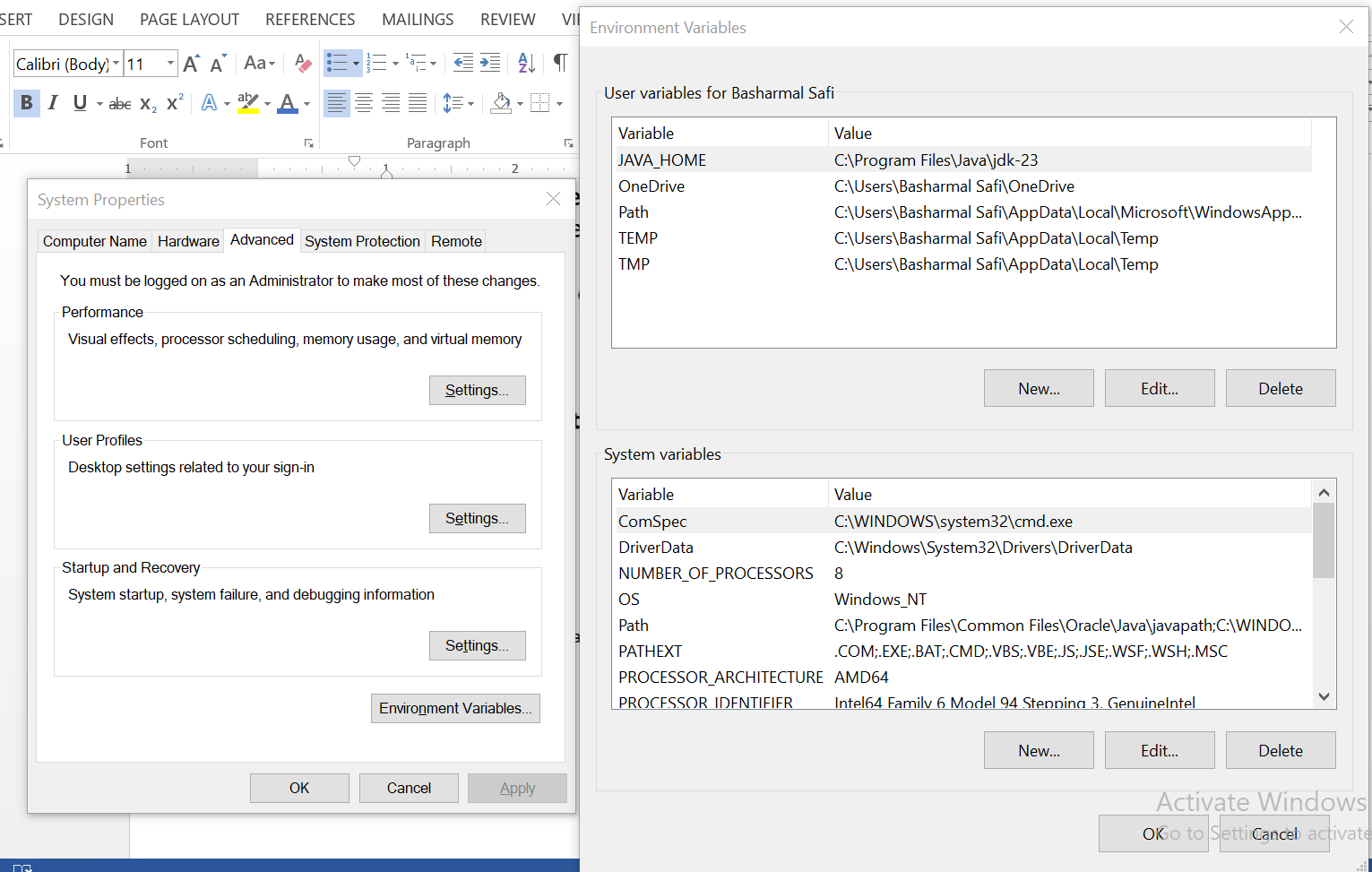
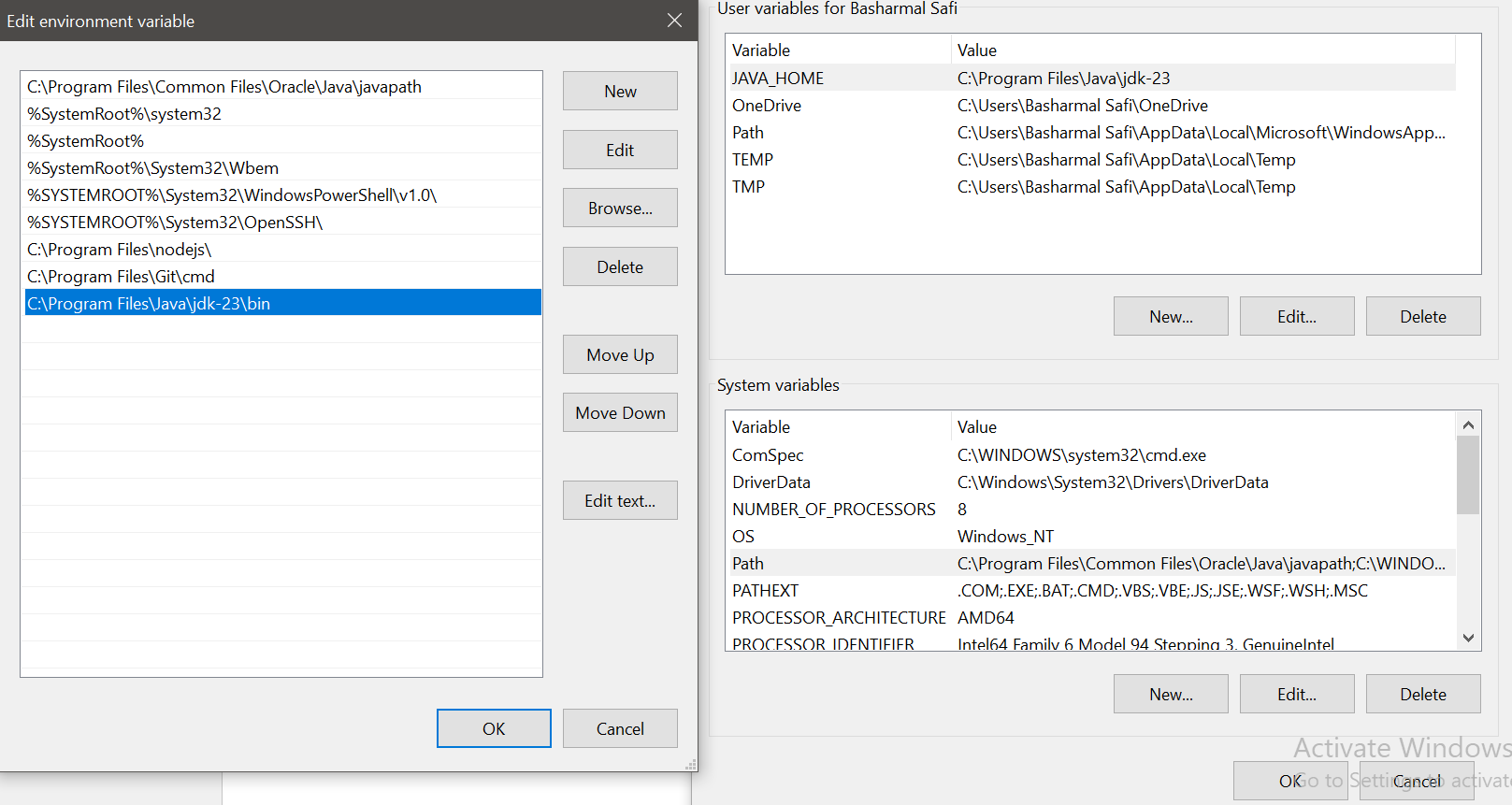
1. Selenium code (Client) is converted to **JSON format**.
2. JSON is sent to the **Browser Driver (Server)** via **HTTP protocol**.
3. Each browser has its own driver (e.g., ChromeDriver, GeckoDriver).
4. Browser Driver interprets JSON and executes commands in the browser.
5. Browser sends response back to the Driver, which returns JSON to the Client.

Selenium WebDriver Architectue Simplified:



1. After you trigger the Test, complete Selenium code (Client) which we have written will be converted to Json format
2. Generated Json is sent to Browser Driver (Server) through http Protocol
3. Note: Each browser contains a separate browser driver
4. . Browser drivers communicate with its respective browser and executes the commands by interpreting Json which It received on the browser.
5. Browser Driver receives responses back from the browser and it sends Json response back to Client.



* Installing the JDK
* Make a environmental variable
  + Got to the pat search bar and type **edit environmental system variable** and add the link below which we copied from **C:\Program Files\Java\jdk-23 see the picture below**
  + and then we set the path too ben directory first we have to copy the link **C:\Program Files\Java\jdk-23\bin** ben

### 📦 Maven Repository & Selenium Setup in Eclipse

**Maven Repository** is a central storage location for **Java libraries**, **plugins**, and **dependencies** used in **Apache Maven** projects. Developers use it to **manage and retrieve dependencies automatically**, rather than manually downloading and configuring them.

The most well-known repository is **Maven Central**, which hosts a vast collection of **open-source libraries**. If you're working with Maven, you can search for dependencies and add them to your project using this repository.

### 🛠️ Setting Up Selenium in Eclipse Using Maven

**Note:** When you create a new project in **Eclipse**, the project will not automatically recognize that it’s a **Selenium** project.

To properly configure it:

1. **Download the Selenium library** from Maven Repository.
2. Create a new **Java project** in Eclipse.
3. Go to the project’s **configuration settings**.
4. **Convert the project to a Maven project**:
   1. Right-click the project → *Configure* → *Convert to Maven Project*.
5. Open the pom.xml file and **add the highlighted dependency code** for Selenium.

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>Introduction</groupId>

<artifactId>Introduction</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>  
 <dependency>  
 <groupId>org.seleniumhq.selenium</groupId>  
 <artifactId>selenium-java</artifactId>  
 <version>4.33.0</version>  
 </dependency>  
 <dependency>  
 <groupId>io.github.bonigarcia</groupId>  
 <artifactId>webdrivermanager</artifactId>  
 <version>5.9.2</version> <!-- use latest stable -->  
 </dependency>  
 <!-- Simple logger -->  
 <dependency>  
 <groupId>org.slf4j</groupId>  
 <artifactId>slf4j-simple</artifactId>  
 <version>2.0.13</version>  
 </dependency>  
 <dependency>  
 <groupId>org.testng</groupId>  
 <artifactId>testng</artifactId>  
 <version>7.11.0</version>  
 </dependency>  
</dependencies>

<build>

<sourceDirectory>src</sourceDirectory>

<plugins>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.13.0</version>

<configuration>

<release>21</release>

</configuration>

</plugin>

</plugins>

</build>

</project>

### 📘 Interfaces in Java & Selenium

* **Interface**: A group of related methods with an **empty body**. A **class** is responsible for **implementing** the declared methods of an interface.

**WebDriver**: An **interface** that provides a set of **browser automation methods** with empty bodies *abstract method*.

### 🖥️ Declaring Browser Objects (Example: Chrome)

There are **two ways** to declare browser objects:

1. ChromeDriver driver = new ChromeDriver();
   * The driver object will have access to **all methods** of **ChromeDriver**.
2. WebDriver driver = new ChromeDriver();
   * The driver object will only have access to the methods defined in the **WebDriver interface**.

### ⚖️ Difference Between the Two Approaches

* **First way (ChromeDriver driver)**
  + Access to **all methods** of ChromeDriver.
  + Not recommended in professional environments.
* **Second way (WebDriver driver)**
  + Access only to methods defined in the **WebDriver interface**.
  + Preferred because it ensures **cross-browser compatibility**.

### 🚫 Why We Don’t Use the First Way

1. We don’t want to use **all methods** of ChromeDriver or any other browser driver.
2. Some methods of **ChromeDriver** may not work in **FirefoxDriver** or other drivers, leading to **incompatibility issues**.

### ✅ Why Use the Second Way

* By using the **WebDriver interface**, we ensure that our code is **browser-independent**.
* This allows automation scripts to run on **multiple browsers** without modification.

### **WebDriverManager**

A library by Boni García. It automatically downloads and manages the right driver version for your browser.

### 🔑 Why Professionals Prefer WebDriverManager

* **No manual driver downloads**.
* **Auto‑matches driver version** with your browser.
* **Cross‑platform** (Windows, Mac, Linux).
* **Cleaner code** (no hard‑coded paths).

### 📦 Importing Selenium Dependencies

To use Selenium drivers, we must import the required dependencies:

java

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

public class intro {

public static void main(String[] args) {

// Creating a FirefoxDriver instance

WebDriver driver = new FirefoxDriver();

// Launching a website

driver.get("https://rahulshettyacademy.com");

// Printing the page title

System.out.println(driver.getTitle());

/\* Difference between close() and quit():

- close(): Closes the current browser tab.

- quit(): Closes all browser windows opened by the driver. \*/

driver.close();

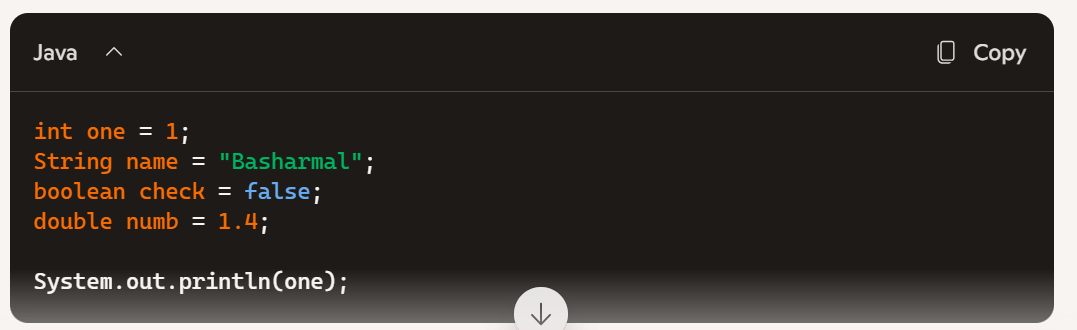
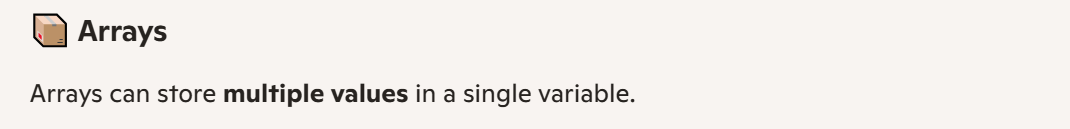
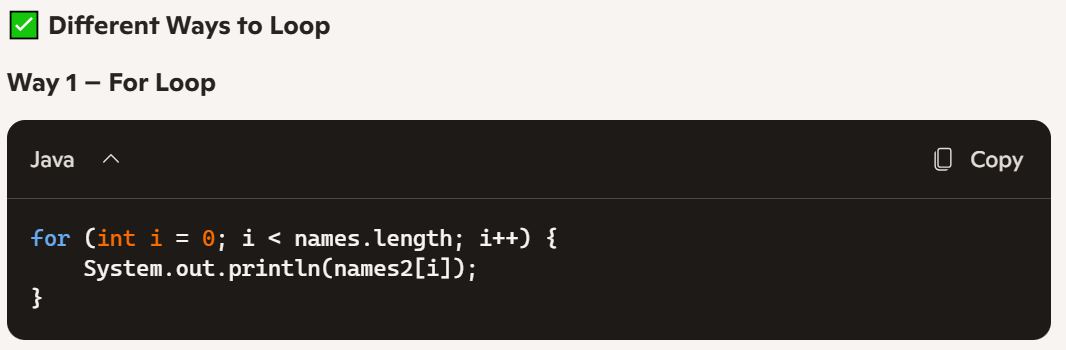
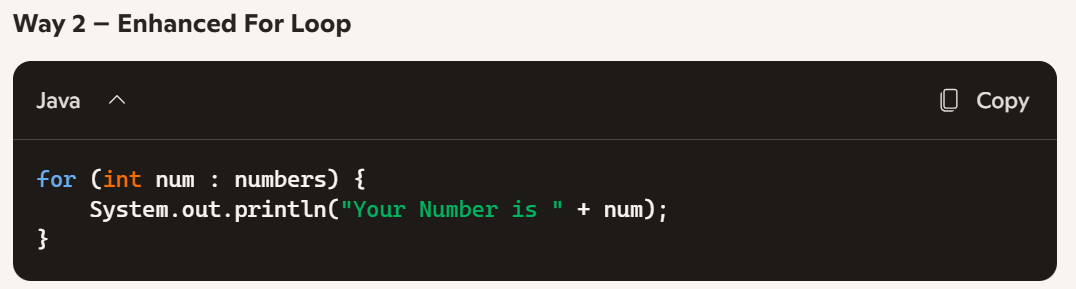
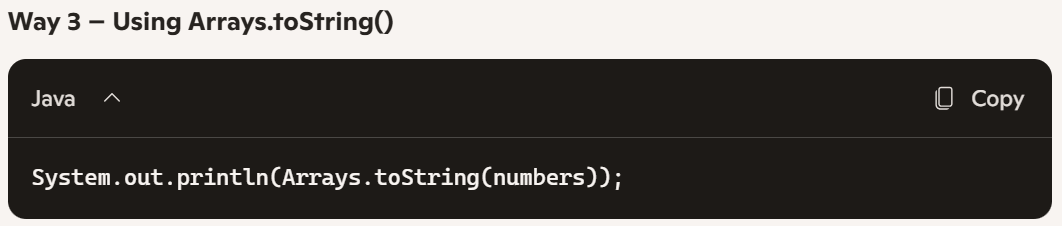
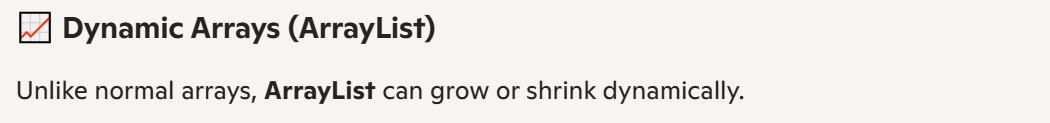
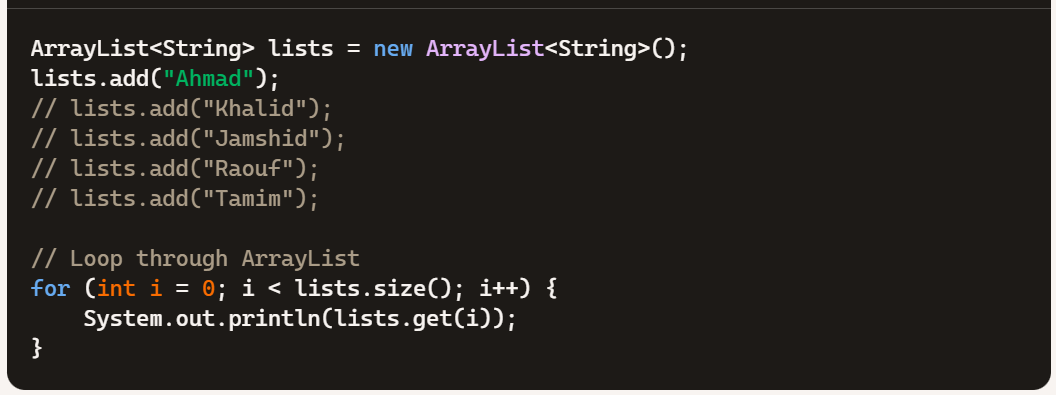
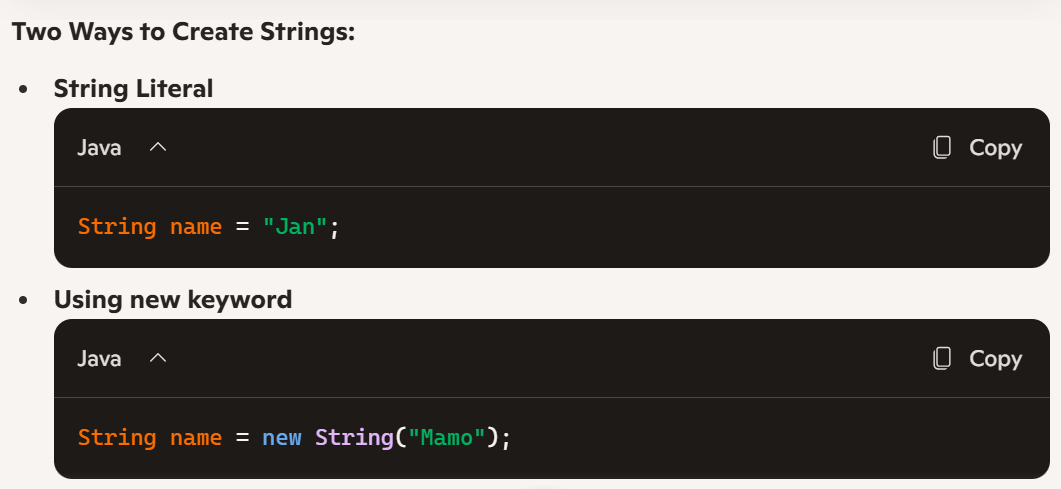
driver.quit();

}

}

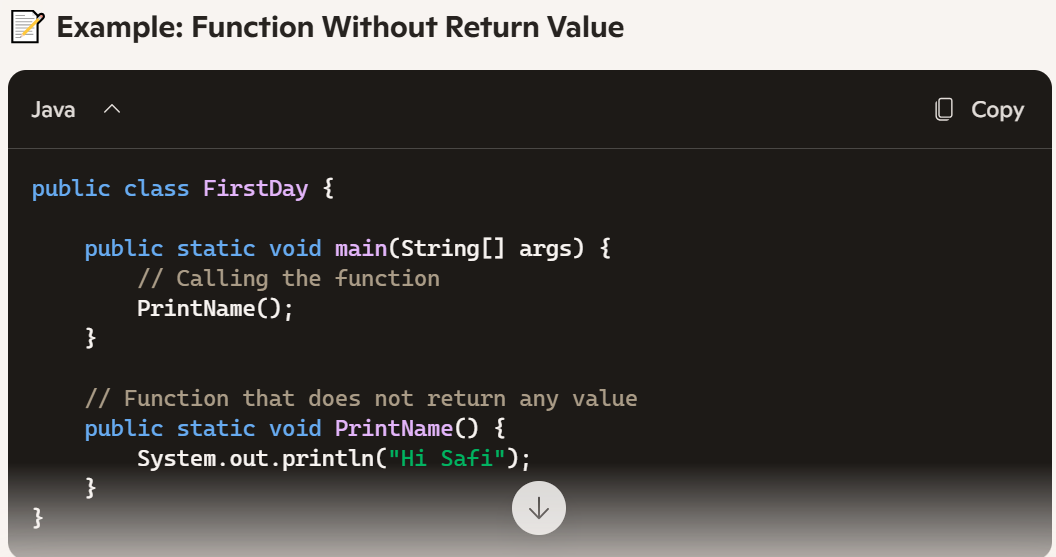
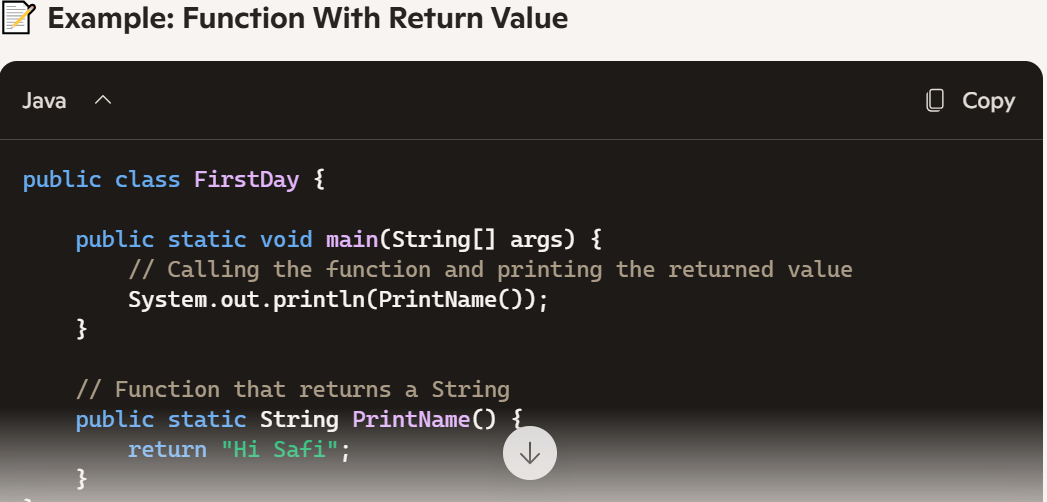
# 🔢 Variables and Data Types in Java

### 📌 Basic Variables

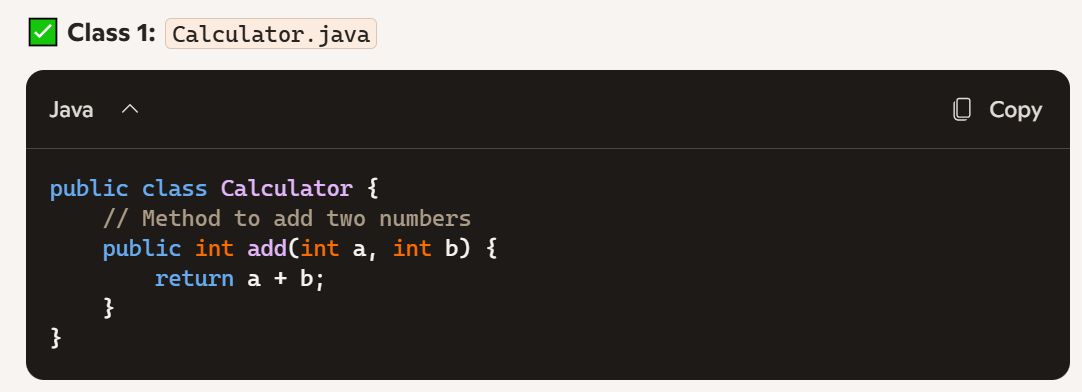
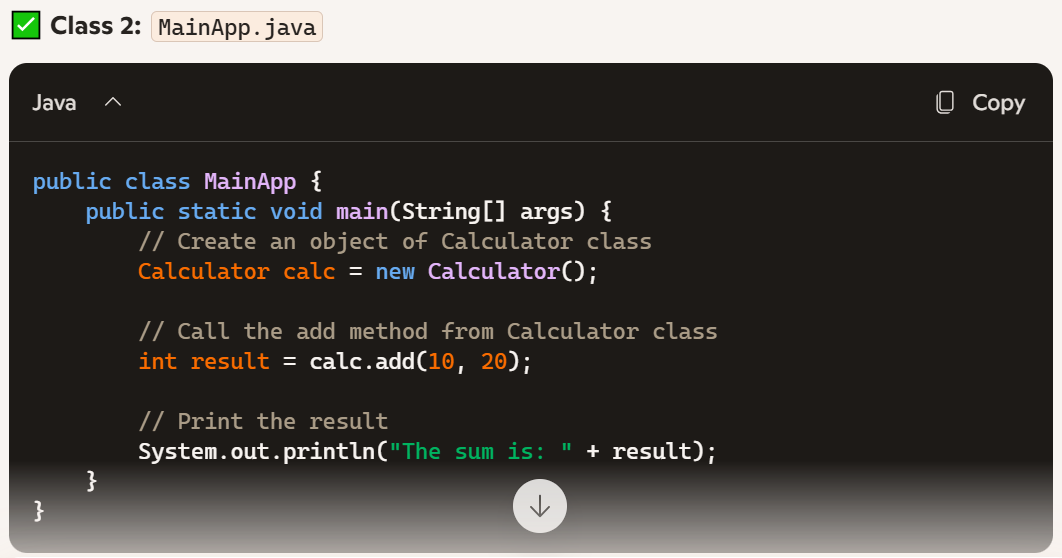
          

# ⚙️ Functions in Java

* A **Function** (or **Method** in Java) is a **block of code** designed to perform a specific task.
* Functions can be **reused** and **executed in different places** within a program, which makes code more **modular**, **readable**, and **maintainable**.

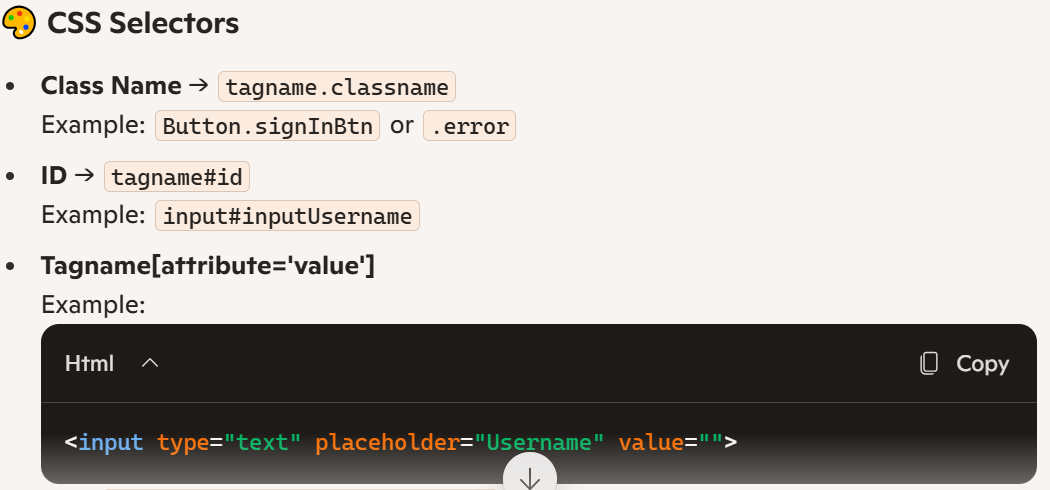
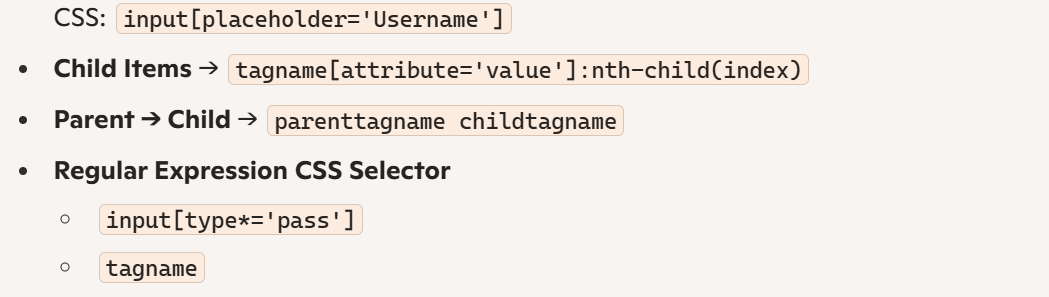
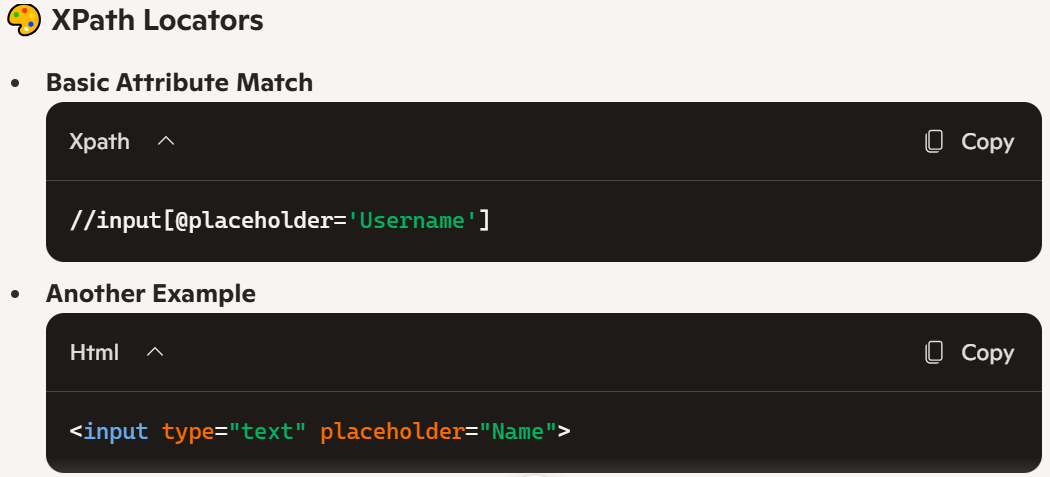
Accessing the Method from one class into another class

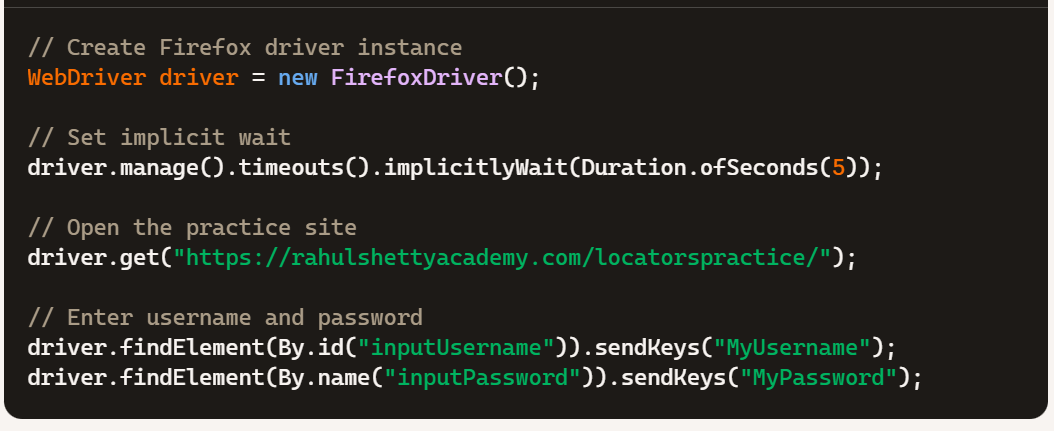
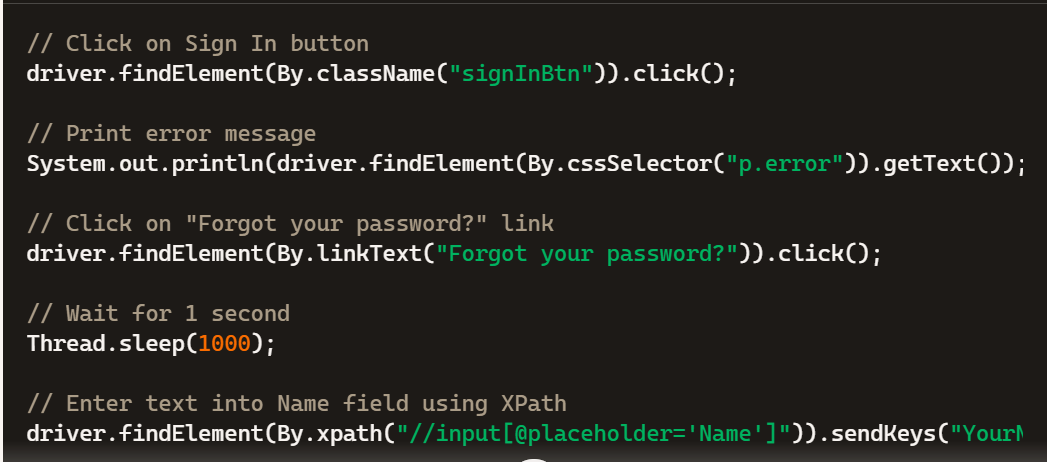
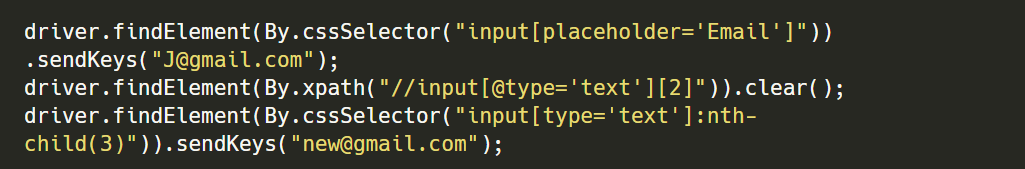
 

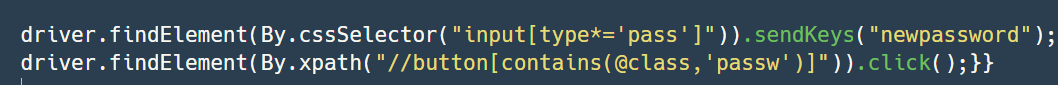
# 🕹️ Selenium WebDriver Locators

### 🔑 Types of Locators in Selenium

* **ID**
* **CSS Selector**
* **Name**
* **Class Name**
* **XPath**
* **Tag Name**
* **Link Text**
* **Partial Link Text**



Note in the above in xpant indexing is like below



And in Css indexing is like below

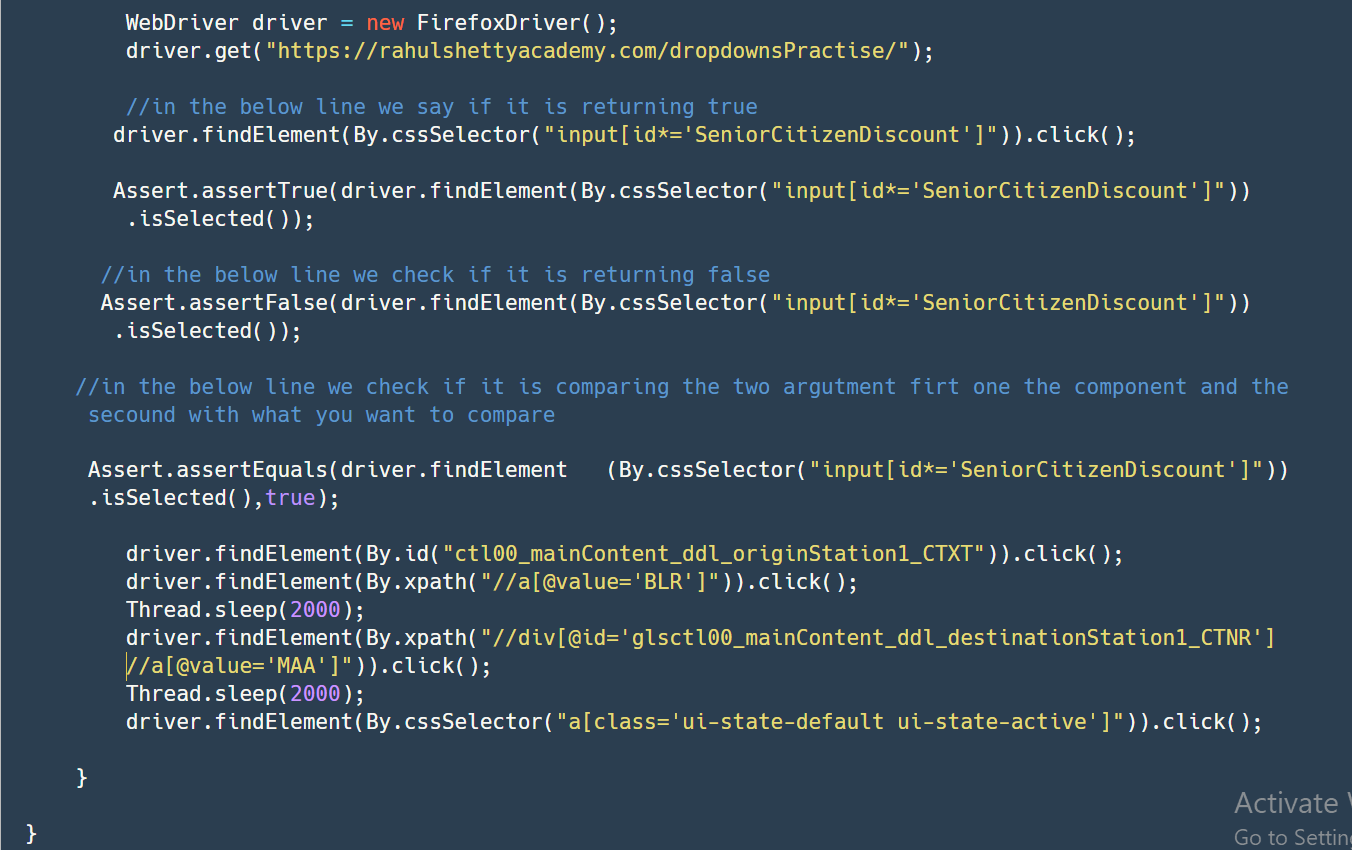


Buy prefer way is doing like below



### Dynamic dropdown is a

WebDriverManager.firefoxDriver().setup();



# Auto Sugested dropdown is a dropdown like search box



# Assert is used to check the condation if it is true or false

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.testng.Assert;

**public** **class** Practice {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

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

driver.get("https://rahulshettyacademy.com/dropdownsPractise/");

//in the below line we say if it is returning true

driver.findElement(By.*cssSelector*("input[id\*='SeniorCitizenDiscount']")).click();

Assert.*assertTrue*(driver.findElement(By.*cssSelector*("input[id\*='SeniorCitizenDiscount']")).isSelected());

//in the below line we check if it is returning false

Assert.*assertFalse*(driver.findElement(By.*cssSelector*("input[id\*='SeniorCitizenDiscount']")).isSelected());

//in the below line we check if it is comparing the two argutment firt one the component and the secound with what you want to compare

Assert.*assertEquals*(driver.findElement(By.*cssSelector*("input[id\*='SeniorCitizenDiscount']")).isSelected(),**true**);

}

}

### UI element Disabled or enabled

public class UIElementDisabledOrEnabled {  
 public static void main(String[] args) throws InterruptedException {  
 WebDriverManager.*firefoxdriver*().setup();  
 WebDriver driver = new FirefoxDriver();  
 driver.get("https://rahulshettyacademy.com/dropdownsPractise/");  
 driver.findElement(By.*id*("ctl00\_mainContent\_rbtnl\_Trip\_0")).click();  
 if(driver.findElement(By.*id*("Div1")).getAttribute("style").contains("1")){  
 Assert.*assertTrue*(true);  
  
 }else {  
 Assert.*assertTrue*(false);  
 }  
  
  
 }  
}

Full practice of the code **import** java.util.List;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.openqa.selenium.support.ui.Select;

**public** **class** FullPractiePageOne {

**public** **static** **void** main(String[] args) **throws** InterruptedException {

// **TODO** Auto-generated method stub

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

driver.get("https://rahulshettyacademy.com/dropdownsPractise/");

driver.findElement(By.*id*("autosuggest")).sendKeys("ind");

Thread.*sleep*(2000);

List<WebElement> options = driver.findElements(By.*cssSelector*("li[class='ui-menu-item'] a"));

**for**(WebElement option:options) {

**if**(option.getText().equalsIgnoreCase("india")) {

option.click();

**break**;

}

}

Thread.*sleep*(5000);

driver.findElement(By.*id*("ctl00\_mainContent\_ddl\_originStation1\_CTXT")).click();

Thread.*sleep*(2000);

driver.findElement(By.*cssSelector*("a[value=\'BLR\']")).click();

driver.findElement(By.*cssSelector*("div[id=\'glsctl00\_mainContent\_ddl\_destinationStation1\_CTNR\'] a[value=\'MAA\']")).click();

driver.findElement(By.*cssSelector*("a[class='ui-state-default ui-state-active']")).click();

Thread.*sleep*(2000);

driver.findElement(By.*xpath*("//input[@id='ctl00\_mainContent\_chk\_SeniorCitizenDiscount']")).click();

driver.findElement(By.*id*("divpaxinfo")).click();

**for**(**int** i=0;i<5;i++) {

driver.findElement(By.*id*("hrefIncAdt")).click();

}

driver.findElement(By.*id*("btnclosepaxoption")).click();

WebElement wblm = driver.findElement(By.*id*("ctl00\_mainContent\_DropDownListCurrency"));

Select dropdown = **new** Select(wblm);

dropdown.selectByIndex(3);

dropdown.getFirstSelectedOption();

Thread.*sleep*(1000);

dropdown.selectByVisibleText("AED");

dropdown.getFirstSelectedOption();

driver.findElement(By.*id*("ctl00\_mainContent\_btn\_FindFlights")).click();

}

}

# Alerts in selenium

import io.github.bonigarcia.wdm.WebDriverManager;  
import org.openqa.selenium.Alert;  
import org.openqa.selenium.By;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.firefox.FirefoxDriver;  
  
public class AlertPractice {  
 public static void main(String[] args) {  
 // Setup Firefox driver  
 WebDriverManager.*firefoxdriver*().setup();  
 WebDriver driver = new FirefoxDriver();  
  
 try {  
 // Navigate to the practice page  
 driver.get("https://rahulshettyacademy.com/AutomationPractice/");  
  
 // Trigger the alert  
 driver.findElement(By.*id*("alertbtn")).click();  
  
 // Switch to alert and handle it  
 Alert alert = driver.switchTo().alert();  
 System.*out*.println("Alert Text: " + alert.getText());  
 alert.accept(); // Accept the alert  
  
 } catch (Exception e) {  
 System.*out*.println("Exception occurred: " + e.getMessage());  
 } finally {  
 // Clean up and close the browser  
 driver.quit();  
 }  
 }  
}

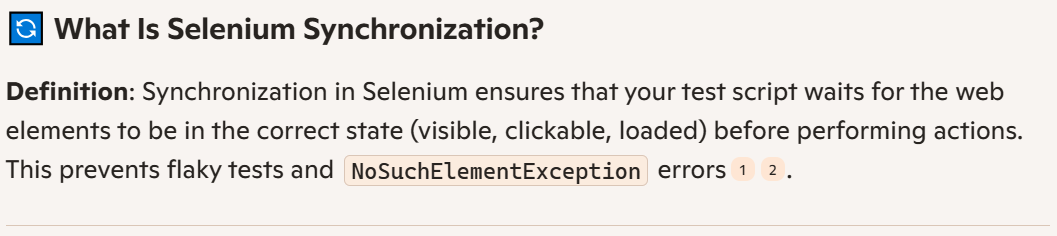
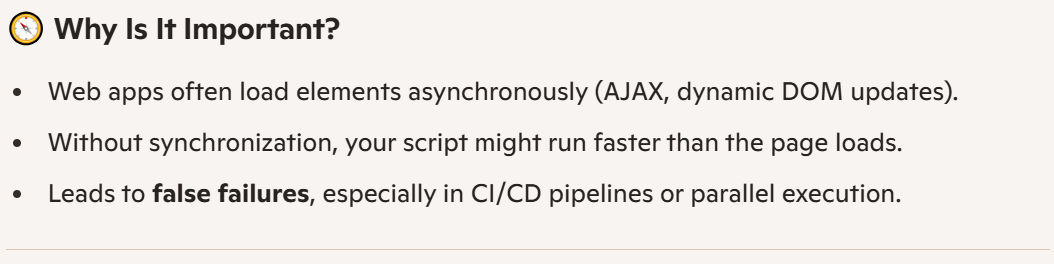
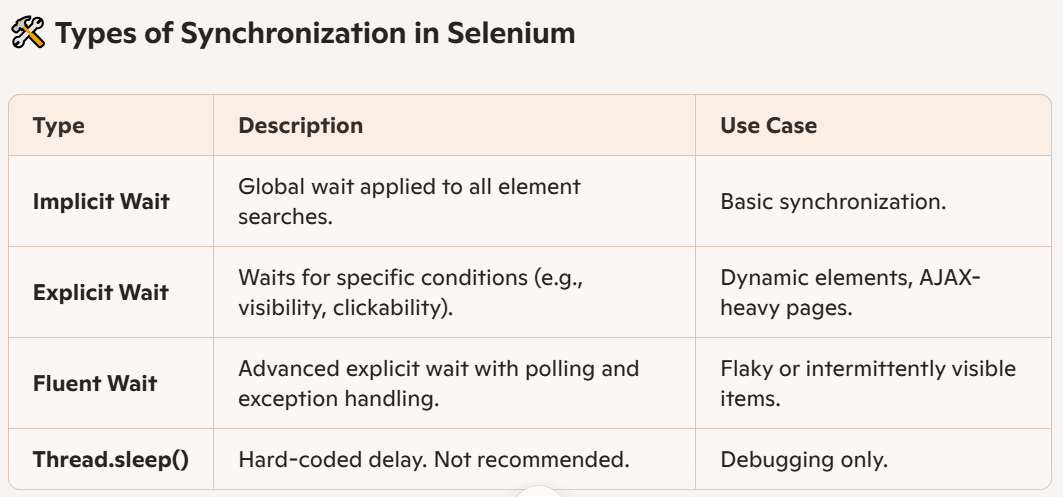
### **Complete Selenium Script: Add Specific Items to Cart**

*import* java.time.Duration;  
*import* java.util.ArrayList;  
*import* java.util.Arrays;  
*import* java.util.*List*;  
*import* org.openqa.selenium.By;  
*import* org.openqa.selenium.*WebDriver*;  
*import* org.openqa.selenium.*WebElement*;  
*import* org.openqa.selenium.firefox.FirefoxDriver;  
*import* io.github.bonigarcia.wdm.WebDriverManager;  
  
*public class* AddItemsToCart {  
 *public static void* main(String[] args) *throws* InterruptedException {  
 *// Setup Firefox driver* WebDriverManager.*firefoxdriver*().setup();  
 *WebDriver* driver = *new* FirefoxDriver();  
 driver.manage().timeouts().implicitlyWait(Duration.*ofSeconds*(5));  
 driver.get("https://rahulshettyacademy.com/seleniumPractise");  
 *List*<String> itemsAddToCart = *new* ArrayList<>(Arrays.*asList*("Cauliflower","Cucumber","Beetroot"));  
 *List*<*WebElement*>products=driver.findElements(By.*cssSelector*("h4.product-name"));  
 *List*<*WebElement*>addToCartButton=driver.findElements(By.*cssSelector*(".product-action button"));  
 *for* (*int* i=0;i<products.size();i++) {  
 String fullName = products.get(i).getText();  
 String itemName = fullName.split(" ")[0].trim();  
 *int* addToCartCount=0;  
 *if* (itemsAddToCart.contains(itemName)) {  
 addToCartButton.get(i).click();  
 addToCartCount++;  
 *if*(addToCartCount==itemsAddToCart.size()) {  
 *break*;  
 }  
 }  
  
 }  
 }  
}

Note do not relay on text like below you be facing problem

driver.findElements(By.*xpath*("//div[@class='product-action']/button[text()='ADD TO CART']")).get(i).click();

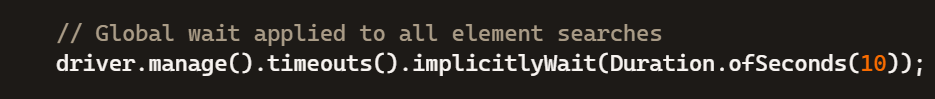
  

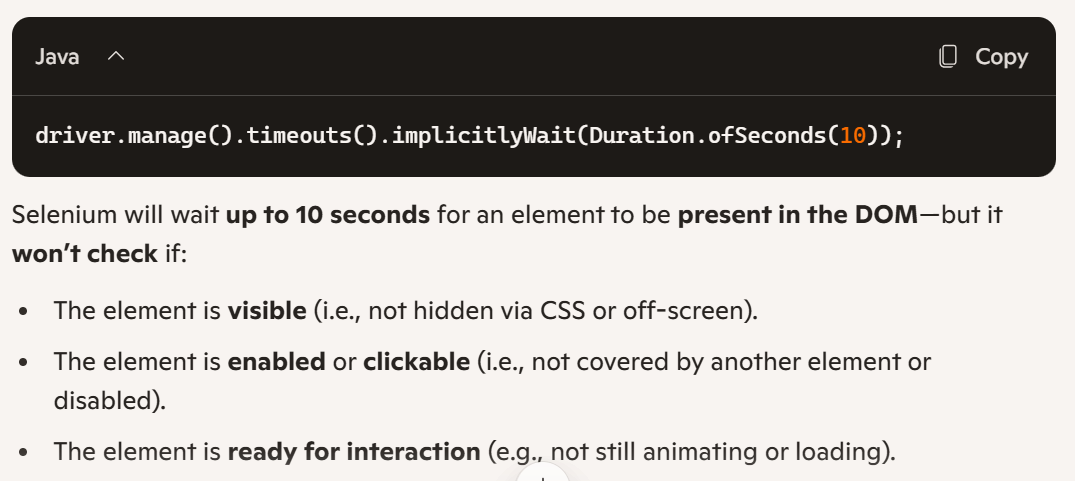
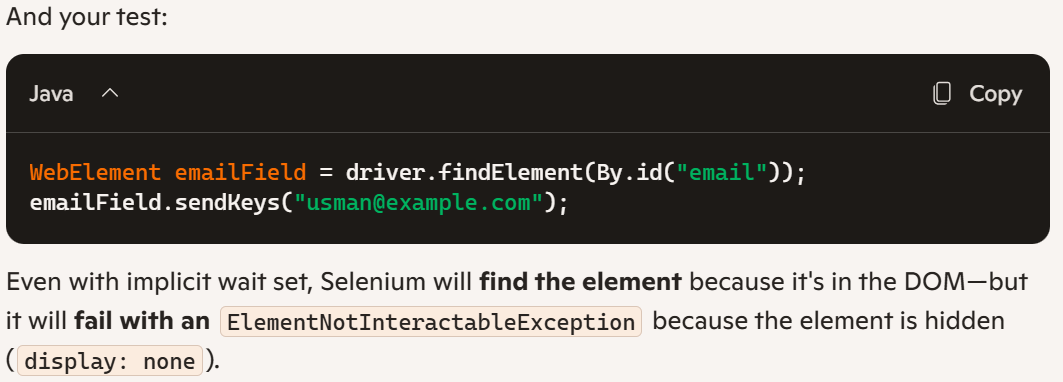
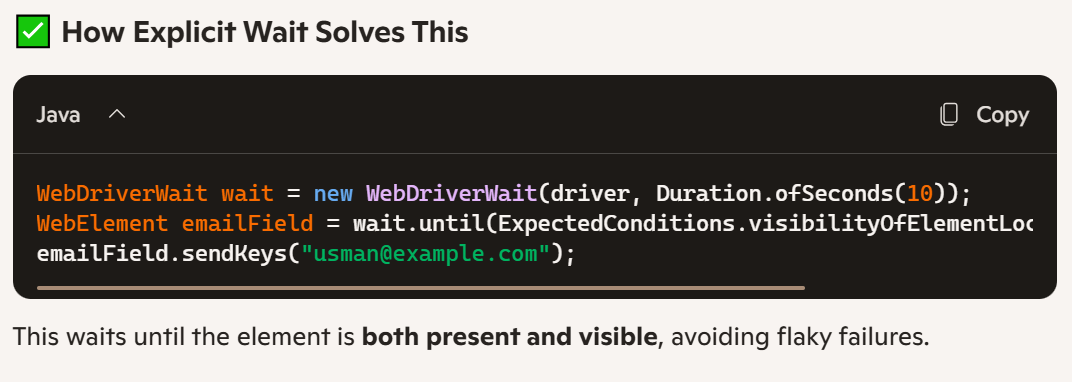
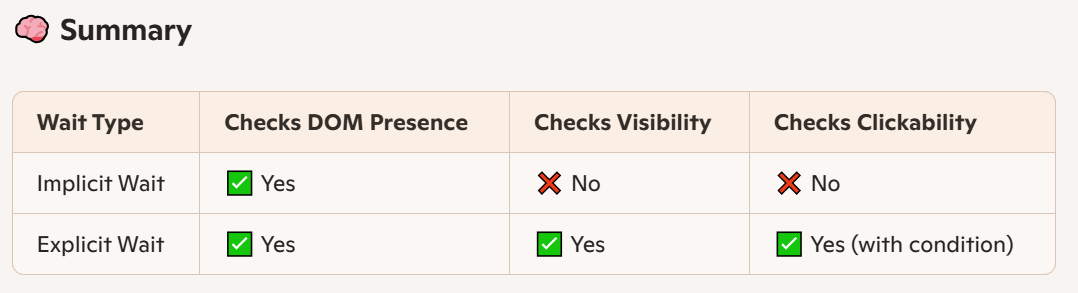
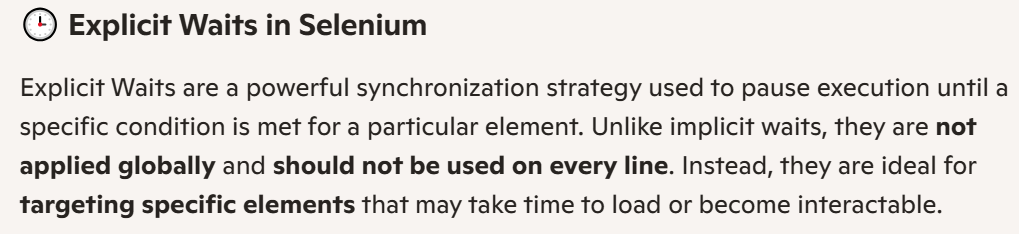
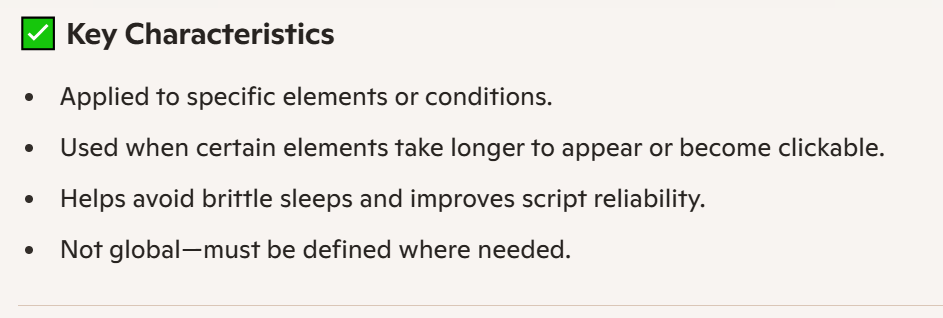
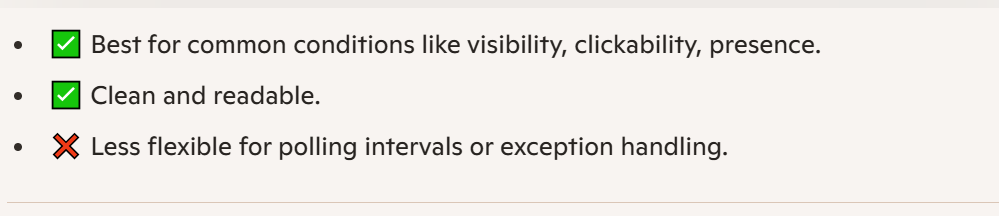
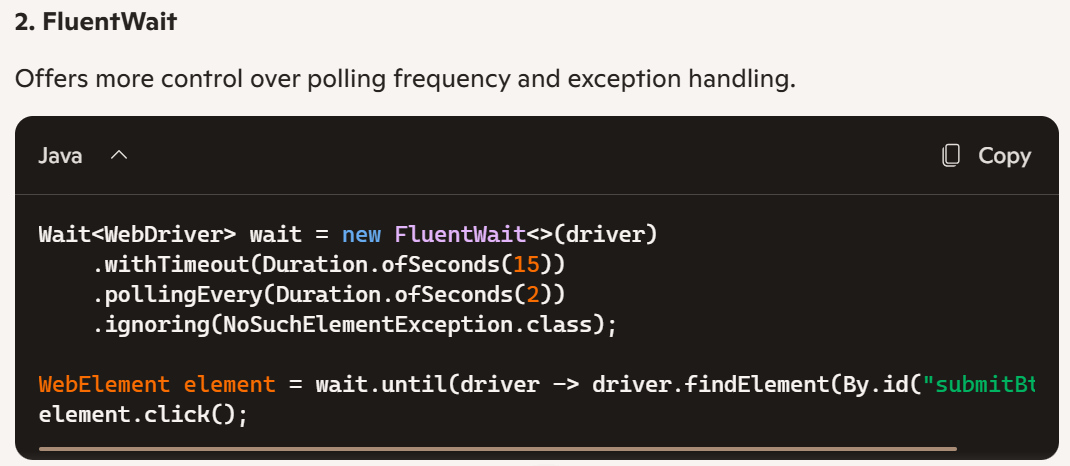
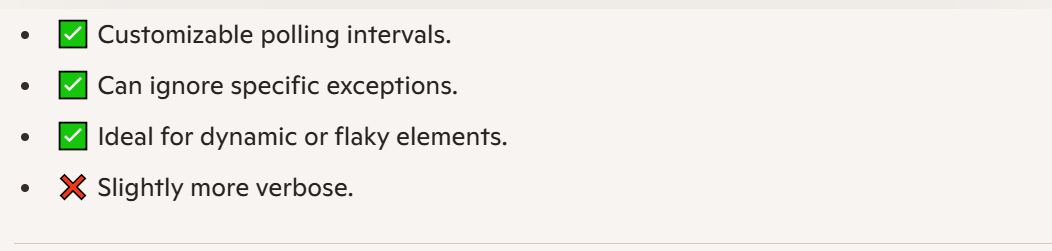
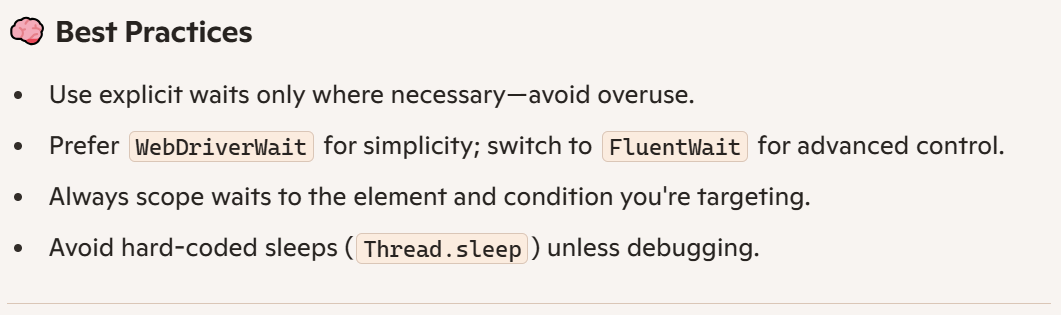
### **Implicit wait**

 **Scope**: since this works globally and applied for each and every line of elements.

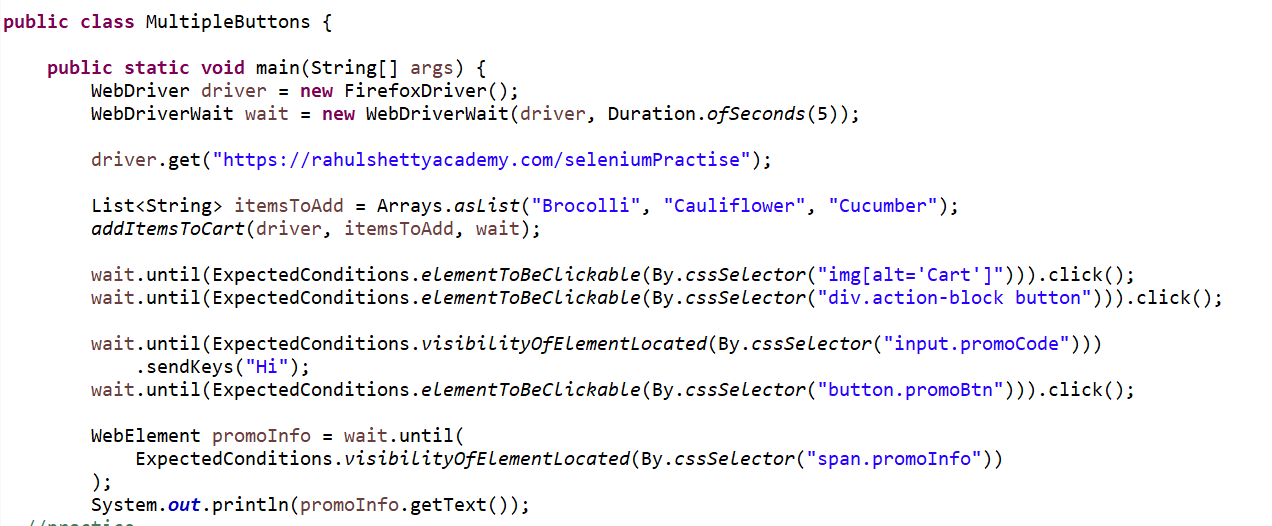
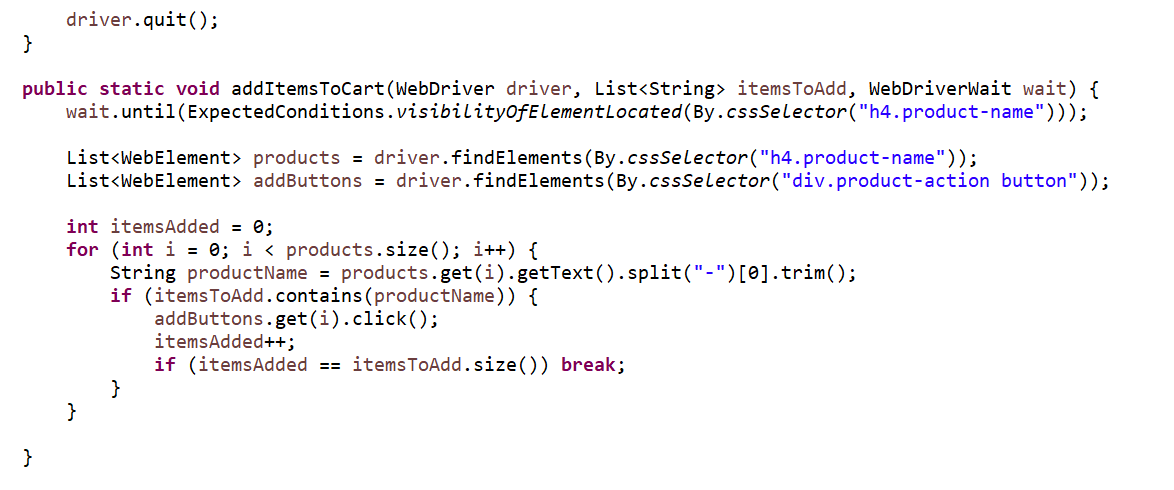
 **Behavior**: If an element is not immediately found, Selenium keeps polling the DOM until the timeout expires.

 **Limitation**: Doesn’t wait for conditions like visibility or click ability—just presence in the DOM.



# WebDriver Wait

**import** java.time.Duration;

**import** java.util.Arrays;

**import** java.util.List;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.openqa.selenium.support.ui.ExpectedConditions;

**import** org.openqa.selenium.support.ui.WebDriverWait;

**public** **class** MultipleButtons {

**public** **static** **void** main(String[] args) {

/\*

\* Task to be done

\* apply wait to necessery elements

\* get the all items text and buttons

\* add some Items to card

\*

\* test the added items in card

\* click on the image card icon

\* send text to textbox

\* click the apply button

\* and check the copon text

\*

\*/

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

WebDriverWait wait = **new** WebDriverWait(driver,Duration.*ofSeconds*(5));

driver.get("https://rahulshettyacademy.com/seleniumPractise");

List<String>itemsAddToCart =Arrays.*asList*("Brocolli", "Cauliflower", "Cucumber");

*addItemsToCart*(driver,wait,itemsAddToCart);

wait.until(ExpectedConditions.*elementToBeClickable*(By.*cssSelector*("img[alt='Cart']"))).click();

wait.until(ExpectedConditions.*elementToBeClickable*(By.*cssSelector*("div.action-block button"))).click();

wait.until(ExpectedConditions.*visibilityOfElementLocated*(By.*cssSelector*("input.promoCode"))).sendKeys("HI");

wait.until(ExpectedConditions.*elementToBeClickable*(By.*cssSelector*("button.promoBtn"))).click();

String promo = wait.until(ExpectedConditions.*visibilityOfElementLocated*(By.*cssSelector*("span.promoInfo"))).getText();

System.***out***.println(promo);

driver.quit();

}

**public** **static** **void** addItemsToCart(WebDriver driver,WebDriverWait wait,List<String>itemsAddToCart) {

wait.until(ExpectedConditions.*visibilityOfElementLocated*(By.*cssSelector*("h4.product-name")));

List<WebElement> products = driver.findElements(By.*cssSelector*("h4.product-name"));

List<WebElement> addToCartButton = driver.findElements(By.*cssSelector*("div.product-action button"));

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

String name=products.get(i).getText().split("-")[0].trim();

**int** itemsAdded=0;

**if**(itemsAddToCart.contains(name)) {

addToCartButton.get(i).click();

itemsAdded++;

**if**(itemsAdded==itemsAddToCart.size()) {

**break**;

}

}

}

}

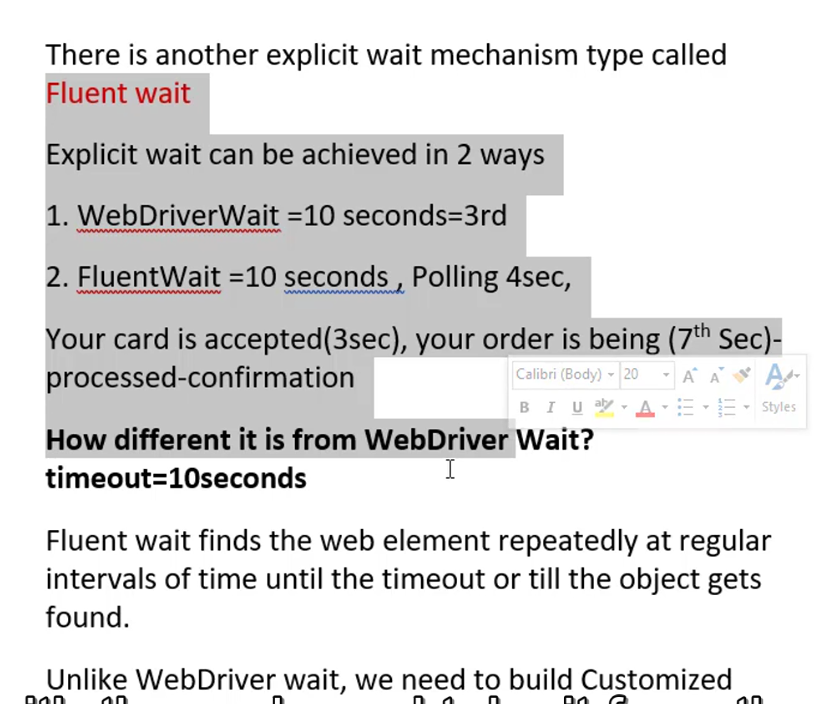
}

# Fluent Wait

## Is a kind of Explicit wait ,

## Fluent wait finds the web elements repeatedly at regular intervals of time until the timeout or till the object gets found

Both WebDriverWait and Fluentwait classes implement Wait interface



**import** java.time.Duration;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.NoSuchElementException;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.openqa.selenium.support.ui.FluentWait;

**import** org.openqa.selenium.support.ui.Wait;

**public** **class** FluentWaitExample {

**public** **static** **void** main(String[] args) {

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

driver.get("https://the-internet.herokuapp.com/dynamic\_loading/1");

driver.findElement(By.*cssSelector*("#start button")).click();

Wait<WebDriver>wait= **new** FluentWait<WebDriver>(driver)

.withTimeout(Duration.*ofSeconds*(30))

.pollingEvery(Duration.*ofSeconds*(3))

.ignoring(NoSuchElementException.**class**);

String finalText = wait.until(wdriver->{

WebElement element= wdriver.findElement(By.*cssSelector*("#finish h4"));

**return** element.isDisplayed()?element:**null**;

}).getText();

System.***out***.println(finalText);

}

}

# Action

## Throw action we navigate from one browser to another from child to parent window

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.Keys;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.openqa.selenium.interactions.Actions;

**public** **class** ActionExample {

**public** **static** **void** main(String[] args) **throws** InterruptedException {

// **TODO** Auto-generated method stub

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

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

Actions a= **new** Actions(driver);

driver.findElement(By.*className*("a-button-text")).click();

// Thread.sleep(3000);

a.moveToElement(driver.findElement(By.*id*("nav-link-accountList"))).build().perform();

a.moveToElement(driver.findElement(By.*id*("twotabsearchtextbox"))).click().keyDown(Keys.***SHIFT***).sendKeys("jan").build().perform();

}

}

Another example of Handle window

import java.util.Iterator;

import java.util.Set;

import java.time.Duration;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.support.ui.WebDriverWait;

import org.openqa.selenium.support.ui.ExpectedConditions;

public class HandleWindowsExample {

public static void main(String[] args) throws InterruptedException {

WebDriver driver = new FirefoxDriver();

driver.get("https://rahulshettyacademy.com/loginpagePractise");

// Click on the link that opens a child window

driver.findElement(By.className("blinkingText")).click();

Set<String>windowHandles=driver.getWindowHandles();

Iterator<String>iterat= windowHandles.iterator();

String parentId=iterat.hasNext()?iterat.next():null;

String childId=iterat.hasNext()?iterat.next():null;

driver.switchTo().window(childId);

WebDriverWait wait = new WebDriverWait(driver,Duration.ofSeconds(3));

String getEmailFromChildWindow = wait.until(ExpectedConditions.presenceOfElementLocated(By.cssSelector(".im-para.red a"))).getText();

driver.switchTo().window(parentId);

driver.findElement(By.id("username")).sendKeys(getEmailFromChildWindow);

// System.out.println(element);

}

}

# Iframe

A **frame** in web development is an HTML element that embeds another HTML document within the current one. In Selenium, if a target element is inside a frame (or an iframe), you must **switch** to it before interacting with elements inside.

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.openqa.selenium.interactions.Actions;

**public** **class** FrameExample {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

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

driver.get("https://jqueryui.com/droppable");

WebElement elm = driver.findElement(By.*className*("demo-frame"));

driver.switchTo().frame(elm);

Actions a = **new** Actions(driver);

WebElement source = driver.findElement(By.*id*("draggable"));

WebElement target = driver.findElement(By.*id*("droppable"));

a.dragAndDrop(source, target).build().perform();

//this will bring you to the main window or take you out of the frame

driver.switchTo().defaultContent();

}

}

# Link Count

**import** java.util.List;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**public** **class** linkCount {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

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

driver.get("https://qaclickacademy.com/practice.php");

driver.findElements(By.*cssSelector*(".gf-t a")).size();

**int** countLinks = driver.findElements(By.*tagName*("a")).size();

System.***out***.println(driver.findElements(By.*cssSelector*(".gf-t a")).size());

}

}

Cliking on Multiple links

//link count in all page

driver.findElements(By.*cssSelector*(".gf-t a")).size();

//link count in footer

// int countLinks = driver.findElements(By.tagName("a")).size();

// driver.findElements(By.cssSelector(".gf-t td")).size();

// System.out.println(driver.findElements(By.cssSelector(".gf-t td")).size());/

//link count in first column

**import** java.lang.reflect.Array;

**import** java.util.Arrays;

**import** java.util.List;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.Keys;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**public** **class** linkCount {

**public** **static** **void** main(String[] args) **throws** InterruptedException {

// **TODO** Auto-generated method stub

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

driver.get("https://qaclickacademy.com/practice.php");

List<WebElement>links=driver.findElements(By.*cssSelector*("table tbody tr td:first-child a"));

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

List<WebElement> link2 =driver.findElements(By.*cssSelector*("table tbody tr td:first-child a"));

String linksOpenInNewTab = Keys.*chord*(Keys.***CONTROL***,Keys.***ENTER***);

link2.get(i).sendKeys(linksOpenInNewTab);

Thread.*sleep*(3000);

driver.navigate().back();

}

}

}

The above and the bottom are same

**public** **static** **void** main(String[] args) **throws** InterruptedException {

// **TODO** Auto-generated method stub

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

driver.get("https://qaclickacademy.com/practice.php");

List<WebElement>links = driver.findElements(By.*cssSelector*("table tbody tr td:first-child a"));

String opentLinksInNewTab= Keys.*chord*(Keys.***CONTROL***,Keys.***ENTER***);

**for**(WebElement link:links) {

link.sendKeys(opentLinksInNewTab);

}

}

Get Tile of the page

# Slectiong Calendar

**import** java.util.List;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.testng.Assert;

**public** **class** CalanderExample {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

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

driver.get("https://rahulshettyacademy.com/seleniumPractise/#/offers");

String monthNumber ="6";

String date="15";

String year="2027";

String[] expectedList = {monthNumber,date,year};

// Open the calendar

driver.findElement(By.*cssSelector*(".react-date-picker\_\_inputGroup")).click();

// Click on year

driver.findElement(By.*cssSelector*(".react-calendar\_\_navigation\_\_label")).click();

//Click on year

driver.findElement(By.*cssSelector*(".react-calendar\_\_navigation\_\_label")).click();

//Select year

driver.findElement(By.*xpath*("//button[text()='"+year+"']")).click();

//Select Month

driver.findElements(By.*cssSelector*(".react-calendar\_\_year-view\_\_months\_\_month")).get(Integer.*parseInt*(monthNumber)-1).click();

//select day

driver.findElement(By.*xpath*("//abbr[text()='"+date+"']")).click();

// get the all date elements andloop throw

List<WebElement> actualList = driver.findElements(By.*cssSelector*(".react-date-picker\_\_inputGroup\_\_input"));

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

String actualValue = actualList.get(i).getAttribute("value");

System.***out***.println(actualValue);

//apply assertion

Assert.*assertEquals*(actualValue, expectedList[i]);

}

}

}

# How to Perform Scrolling with in table and Window level using JavaScriptExecutor

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** java.util.List;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.JavascriptExecutor;

**public** **class** ScrollingExample {

**public** **static** **void** main(String[] args) **throws** InterruptedException {

// **TODO** Auto-generated method stub

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

driver.get("https://rahulshettyacademy.com/AutomationPractice");

JavascriptExecutor js = (JavascriptExecutor)driver;

js.executeScript("window.scrollBy(0,500)");

Thread.*sleep*(3000);

js.executeScript("document.querySelector('.tableFixHead').scrollTop=5000");

List<WebElement> tableContext = driver.findElements(By.*cssSelector*("#product td:nth-child(4)"));

**int** sum=0;

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

sum =sum+Integer.*parseInt*(tableContext.get(i).getText());

System.***out***.println(sum);

}

}

}

# 104 - Handling HTTPS certifications in Automated browsers

FirefoxOptions option = **new** FirefoxOptions();

option.setAcceptInsecureCerts(**true**);

WebDriver driver= **new** FirefoxDriver(option);

driver.get("https://rahulshettyacademy.com/AutomationPractice");

# Stream

ArrayList<String>nameArray=**new** ArrayList<String>();

nameArray.add("Ahmad");

nameArray.add("Mehmood");

nameArray.add("Asad");

nameArray.add("Jan");

nameArray.add("Asif");

nameArray.add("Mustafa");

List<String>nameList = Arrays.*asList*("Ahamd","Mehmood","Jan","Khan","Asad","Mustafa");

Long counts =nameList.stream().filter(s->s.startsWith("A")).count();

System.***out***.println(counts);

// get me the names which length is less than 4

nameList.stream().filter(s->s.length()<4).forEach(s->System.***out***.println(s));

//and then convert it to upper case

nameList.stream().filter(s->s.length()<5).map(m->m.toUpperCase()).forEach(s->System.***out***.println(s));

//and to lowercase

nameList.stream().map(s->s.toLowerCase()).forEach(s->System.***out***.println(s));

//get me the only one form the list

nameList.stream().filter(s->s.length()>5).limit(1).forEach(s->System.***out***.println(s));

//sorting

nameList.stream().sorted().forEach(s->System.***out***.println(s));

//Streamof will create an array and then apply the other functions

Stream.*of*("Wali","Tawoos","Muhif","Arif").sorted().forEach(s->System.***out***.println(s));

//combine or merging two arrays to gather

Stream.*concat*(nameList.stream(), nameArray.stream()).forEach(s->System.***out***.println(s));

//or

Stream<String>combinedStream=Stream.*concat*(nameList.stream(), nameArray.stream());

// combinedStream.forEach(s->System.out.println(s));

//give me the true or false from the stream

**boolean** results = combinedStream.anyMatch(s->s.equalsIgnoreCase("Jakn"));

System.***out***.println(results);

Assert.*assertTrue*(results);

# Compairing the table columns and check it is sorted or no

**import** java.util.Arrays;

**import** java.util.List;

**import** java.util.stream.Collectors;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.testng.Assert;

**public** **class** TableSortingExample {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

/\*Our goal is to compare the column of table either it is sorted of no\*/

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

driver.get("https://rahulshettyacademy.com/seleniumPractise/#/offers");

driver.findElement(By.*cssSelector*(".sort-icon")).click();

List<WebElement>rawItems=driver.findElements(By.*cssSelector*("tr td:nth-child(1"));

List<String>originalList =rawItems.stream().map(m->m.getText()).collect(Collectors.*toList*());

List<String>sortedList = rawItems.stream().map(m->m.getText()).sorted().collect(Collectors.*toList*());

System.***out***.println(originalList);

System.***out***.println(sortedList);

Assert.*assertEquals*(originalList, sortedList);

//

}

}

# Filteration in

**import** java.util.List;

**import** java.util.stream.Collectors;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.testng.Assert;

**public** **class** FilterExample {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

//Filtration in

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

driver.get("https://rahulshettyacademy.com/seleniumPractise/#/offers");

driver.findElement(By.*id*("search-field")).sendKeys("Chocolate");

List<WebElement>originalData=driver.findElements(By.*xpath*("//tr/td[1]"));

List<WebElement>filteredData=originalData.stream().filter(s->s.getText().contains("Chocolate")).collect(Collectors.*toList*());

Assert.*assertEquals*(originalData,filteredData);

}

}

