Synchronization in Selenium

Implicit wait

Explicit wait

Thread.Sleep()

Fluent wait

Implicit wait – Generally it waits for given seconds and then throws exception if page is not loaded and i.e. by getting clicked next statement(element) (click or something selenium doing). For example if some page is loaded within 2 seconds then t executes next statement(element) it doesn’t wait for 5 seconds. If you give implicit wait for driver it is applicable to all the other statements that is using that driver

Disadvantage – Say for example in that program one page is getting loaded for more than 5 seconds say huge data getting populated then this fails since implicit wait is 5 seconds.

Pros – Readable code

Cons- Performance cause issues are not right

Explicit wait – to avoid above disadvantage you can apply explicit wait only on a particular statement(element)

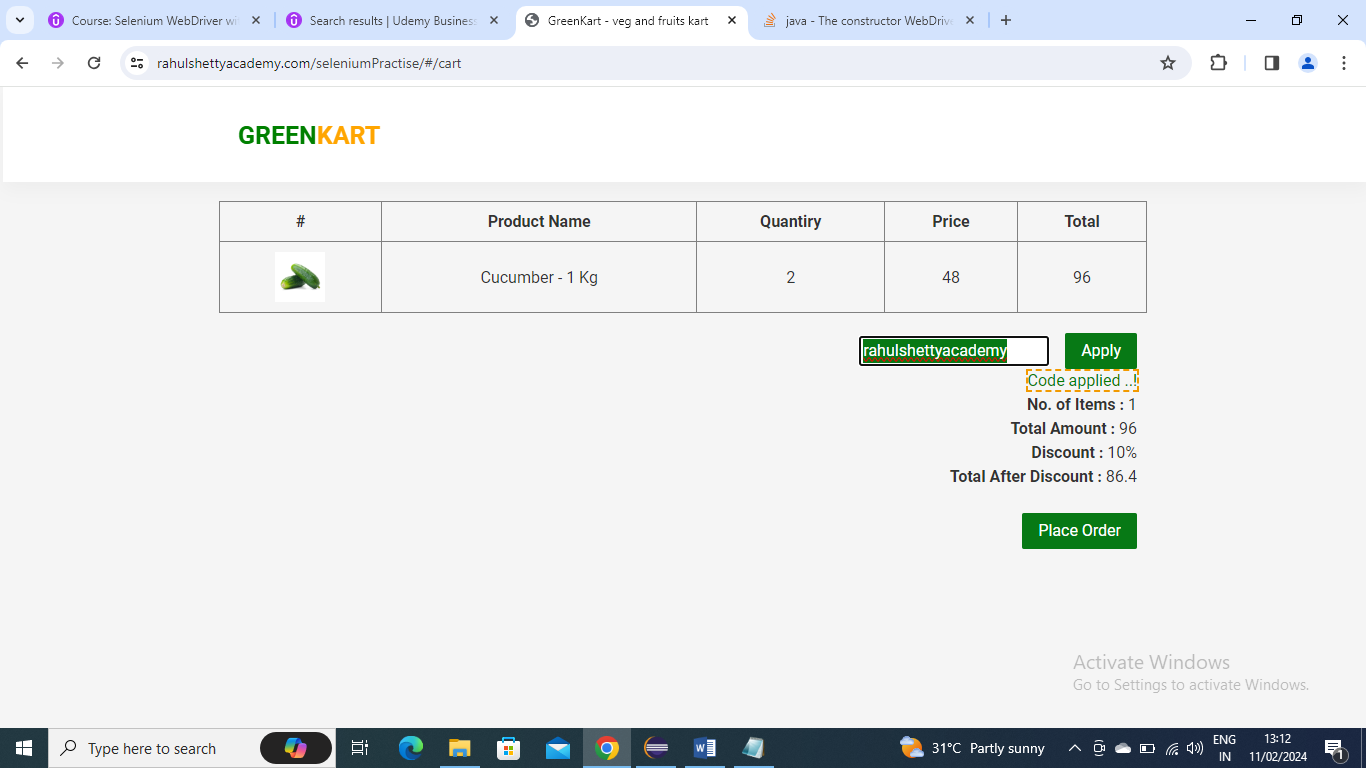
Pros – Wait is applied at target elements. So no performance issue

Cons – More code

Thread.Sleep() – it generally pauses the script for given number of seconds. It won’t execute any statement for 5 seconds even if the webpage is open as well.

Implicit wait Example –

In the below example every statement waits for 2 seconds to execute of page is not loaded within. As mentioned the disadvantage is the promo button applying takes more than 10 seconds here the statement(element) fails since check button gets applied when page is not loaded after 2 seconds.



**import** java.util.Arrays;

**import** java.util.List;

**import** java.util.concurrent.TimeUnit;

**import** javax.swing.text.Style;

**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** org.testng.Assert;

**public** **class** Selenium2 {

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

// Below is the driver extension that is required to invoke the browser

System.*setProperty*("webdriver.chrome.driver", "E:\\Selenium\\chromedriver.exe");

// Invoking Browser

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

driver.manage().timeouts().~~implicitlyWait~~(2, TimeUnit.***SECONDS***);

String[] itemsNeeded = { "Cucumber", "Brocolli", "Beetroot" };

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

Thread.*sleep*(3000);

*addItems*(driver, itemsNeeded);

driver.findElement(By.*cssSelector*("img[alt='Cart']")).click();

driver.findElement(By.*xpath*("//button[contains(text(),'PROCEED TO CHECKOUT')]")).click();

driver.findElement(By.*className*("promoCode")).sendKeys("rahulshettyacademy");

//here the promo click applying takes more than 10 seconds which will fail because implicit wait is just 2 seconds

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

driver.findElement(By.*xpath*("//button[contains(.,'Place Order')]")).click();

}

**public** **static** **void** addItems(WebDriver driver, String[] itemsNeeded)

{

**int** j = 0;

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

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

{

// Brocolli - 1 Kg

// Brocolli, 1 kg

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

String formattedName = name[0].trim();

// format it to get actual vegetable name

// convert array into array list for easy search

// check whether name you extracted is present in arrayList or not-

List itemsNeededList = Arrays.*asList*(itemsNeeded);

**if** (itemsNeededList.contains(formattedName))

{

j++;

// click on Add to cart

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

**if** (j == itemsNeeded.length)

{

**break**;

}

}

}

}

}

Explicit example –

To avoid above disadvantage where the promo code applying taking time of more than 5 seconds here we can apply explicit wait where the wait will be only for that statement or eleement (applying code) where it waits for given number of seconds until its loaded o available. If it loads early it won’t wait for given number of seconds that means it monitors every millisecond

**import** java.time.Duration;

**import** java.util.Arrays;

**import** java.util.List;

**import** java.util.concurrent.TimeUnit;

**import** javax.swing.text.Style;

**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.ExpectedCondition;

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

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

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

**import** org.testng.Assert;

**public** **class** Selenium2 {

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

// Below is the driver extension that is required to invoke the browser

System.*setProperty*("webdriver.chrome.driver", "E:\\Selenium\\chromedriver.exe");

// Invoking Browser

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

driver.manage().timeouts().~~implicitlyWait~~(2, TimeUnit.***SECONDS***);

String[] itemsNeeded = { "Cucumber", "Brocolli", "Beetroot" };

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

Thread.*sleep*(3000);

*addItems*(driver, itemsNeeded);

driver.findElement(By.*cssSelector*("img[alt='Cart']")).click();

driver.findElement(By.*xpath*("//button[contains(text(),'PROCEED TO CHECKOUT')]")).click();

driver.findElement(By.*className*("promoCode")).sendKeys("rahulshettyacademy");

//here the promo click applying takes more than 10 seconds which will fail because implicit wait is just 2 seconds

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

//explicit wait

WebDriverWait w= **new** WebDriverWait(driver,Duration.*ofSeconds*(10));

w.until(ExpectedConditions.*visibilityOfAllElementsLocatedBy*(By.*cssSelector*("span.promoInfo")));

driver.findElement(By.*xpath*("//button[contains(.,'Place Order')]")).click();

}

**public** **static** **void** addItems(WebDriver driver, String[] itemsNeeded)

{

**int** j = 0;

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

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

{

// Brocolli - 1 Kg

// Brocolli, 1 kg

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

String formattedName = name[0].trim();

// format it to get actual vegetable name

// convert array into array list for easy search

// check whether name you extracted is present in arrayList or not-

List itemsNeededList = Arrays.*asList*(itemsNeeded);

**if** (itemsNeededList.contains(formattedName))

{

j++;

// click on Add to cart

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

**if** (j == itemsNeeded.length)

{

**break**;

}

}

}

}

}

Fluent wait

Explicit wait can be achieved in 2 ways

WebDriverWait

FluentWait

How different it is from WebDriver Wait?

Fluent wait finds the web element repeatedly at regular intervals of time until the timeout or till the objects get found. For example Explicit wait monitors(poll) every millisecond while Fluent monitors(poll) like every 2 seconds.

Unlike WebDriver Wait we need to build customized wait method based on condition

Both WebDriverWait and FluentWait classes implement Wait Interface

Below is an example

**import** java.time.Duration;

**import** java.util.NoSuchElementException;

**import** java.util.concurrent.TimeUnit;

**import** java.util.function.Function;

**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.FluentWait;

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

**public** **class** Selenium3 {

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

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

driver.findElement(By.*cssSelector*("[id='start'] button")).click();

Wait<WebDriver> wait =**new** FluentWait<WebDriver>(driver).withTimeout(Duration.*ofSeconds*(30)).

pollingEvery(Duration.*ofSeconds*(3)).ignoring(NoSuchElementException.**class**);

WebElement foo =wait.until(**new** Function<WebDriver, WebElement>() {

**public** WebElement apply(WebDriver driver) {

**if**(driver.findElement(By.*cssSelector*("[id='finish']h4")).isDisplayed()) {

**return** driver.findElement(By.*cssSelector*("[id='finish']h4"));

}

**else**

**return** **null**;

}

});

System.***out***.println(driver.findElement(By.*cssSelector*("[id='finish']h4")).getText());

}

}

Actions:

How to MouseOver on object with selenium?

Performing Mouse and Keyboard interaction with selenium

Context click on element

Double click on element

Drag and dropping the element

Frames:

What are Frames?

How to identify Frames in application?

How to handle frames?

Best Practices when working with Frames application

Below is an example of action on how to Mouse hover in a webpage

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

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

**public** **class** Selenium4 {

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

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

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

a.moveToElement(driver.findElement(By.*cssSelector*("span[class='nav-line-2 ']"))).build().perform();

}

}

Above code can be written more in a better way where the xpath info is written in a webelement variable so that code looks neat

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.interactions.Actions;

public class Selenium4 {

public static void main(String[] args) {

// TODO Auto-generated method stub

System.setProperty("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

WebDriver driver = new ChromeDriver();

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

Actions a=new Actions(driver);

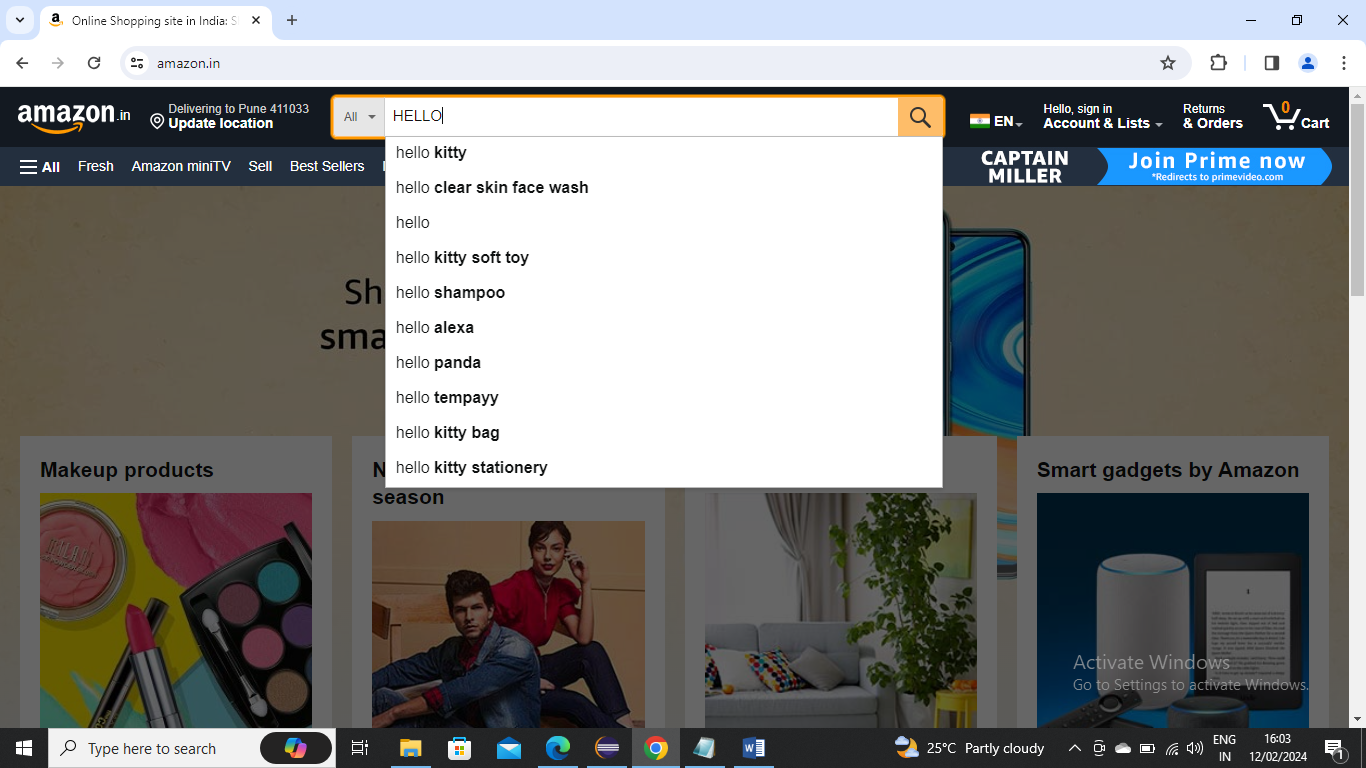
WebElement ele = driver.findElement(By.cssSelector("span[class='nav-line-2 ']"));

a.moveToElement(ele).build().perform();

}

}

Below is an example where you can move to text and write data in capital letters using selenium



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

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

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

**public** **class** Selenium4 {

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

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

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

WebElement ele1 = driver.findElement(By.*id*("twotabsearchtextbox"));

a.moveToElement(ele1).click().keyDown(Keys.***SHIFT***).sendKeys("hello");

}

}

Below is an example on how the control needs to switch to other window (i.e. child window) you can do vice versa multiple times as well

**import** java.util.Iterator;

**import** java.util.Set;

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

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

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

**public** **class** Selenium4 {

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

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

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

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

Iterator<String> it=windows.iterator();

String parentId =it.next();

String childId =it.next();

driver.switchTo().window(childId);

System.***out***.println(driver.findElement(By.*cssSelector*(".im-para.red")).getText());

}

}

Below is an example where dividing the received text to get the mail id from that text (may be to pass somewhere)

import java.util.Iterator;

import java.util.Set;

import org.openqa.selenium.By;

import org.openqa.selenium.Keys;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.interactions.Actions;

public class Selenium4 {

public static void main(String[] args) {

// TODO Auto-generated method stub

System.setProperty("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

WebDriver driver = new ChromeDriver();

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

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

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

Iterator<String> it=windows.iterator();

String parentId =it.next();

String childId =it.next();

driver.switchTo().window(childId);

System.out.println(driver.findElement(By.cssSelector(".im-para.red")).getText());

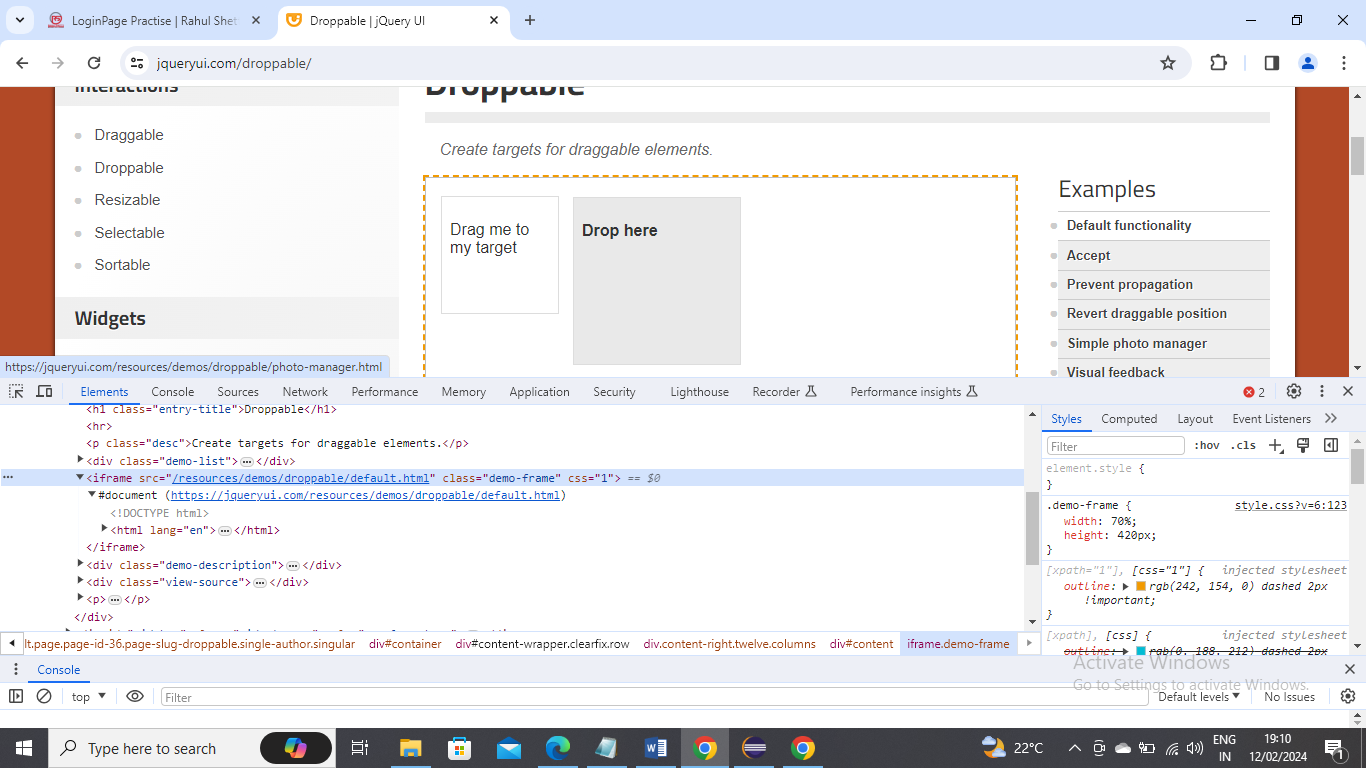
String emailId = driver.findElement(By.cssSelector(".im-para.red")).getText().split("at")[1].trim().split(" ")[0];

System.out.println(emailId);

}

}

Below is an example of frames, where how can we switch to frames and do a drag and drop from one place to another



import java.util.Iterator;

import java.util.Set;

import org.openqa.selenium.By;

import org.openqa.selenium.Keys;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.interactions.Actions;

public class Selenium4 {

public static void main(String[] args) {

// TODO Auto-generated method stub

System.setProperty("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

WebDriver driver = new ChromeDriver();

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

driver.switchTo().frame(driver.findElement(By.cssSelector("iframe[class='demo-frame']")));

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();

//to come out of frame

driver.switchTo().defaultContent();

}

}

Below is an example on how to gets count of links and also in footer, how to gets links of one column side in footer section, opening the links in new tabs, getting the title from all new tabs that are open

import java.util.Iterator;

import java.util.Set;

import org.openqa.selenium.By;

import org.openqa.selenium.Keys;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.interactions.Actions;

public class Selenium4 {

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

// TODO Auto-generated method stub

System.setProperty("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

WebDriver driver = new ChromeDriver();

// 1. Give me the count of links on the page.

// 2. Count of footer section-

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

System.out.println(driver.findElements(By.tagName("a")).size());

WebElement footerdriver = driver.findElement(By.id("gf-BIG"));// Limiting webdriver scope

System.out.println(footerdriver.findElements(By.tagName("a")).size());

// 3- How to get the first column links links from footer section

WebElement coloumndriver = footerdriver.findElement(By.xpath("//table/tbody/tr/td[1]/ul"));

System.out.println(coloumndriver.findElements(By.tagName("a")).size());

// 4- click on each link in the coloumn and check if the pages are opening in another tab

for (int i = 1; i < coloumndriver.findElements(By.tagName("a")).size(); i++) {

String clickonlinkTab = Keys.chord(Keys.CONTROL, Keys.ENTER);

coloumndriver.findElements(By.tagName("a")).get(i).sendKeys(clickonlinkTab);

Thread.sleep(5000);

}

// go to all the tabs and get the tile

Set<String> abc = driver.getWindowHandles();// 4

Iterator<String> it = abc.iterator();

while (it.hasNext()) {

driver.switchTo().window(it.next());

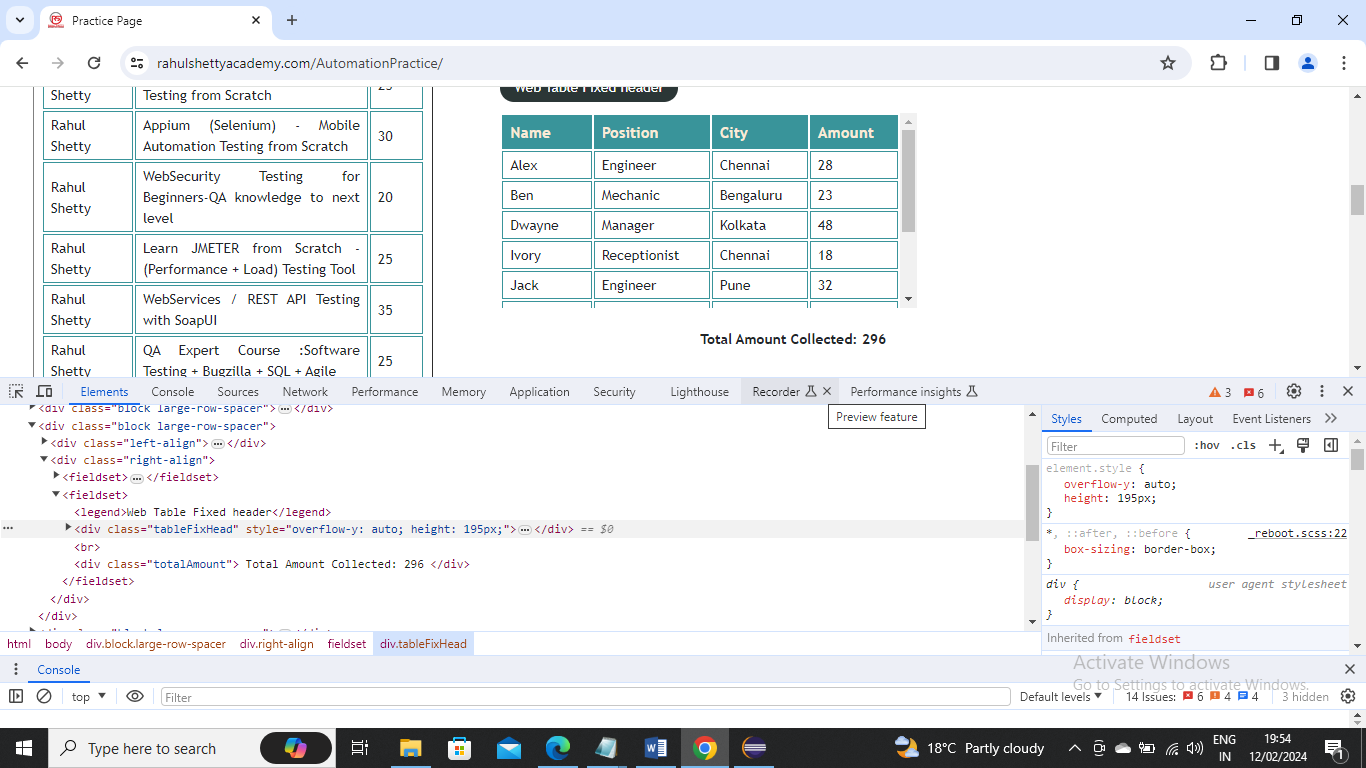
System.out.println(driver.getTitle());

}

}

}

Below is an example on how to scroll the window and also how to scroll inside a window as well



**import** java.util.Iterator;

**import** java.util.List;

**import** java.util.Set;

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

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

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

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

**import** org.testng.Assert;

**public** **class** Selenium4 {

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

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

JavascriptExecutor js= (JavascriptExecutor) driver;

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

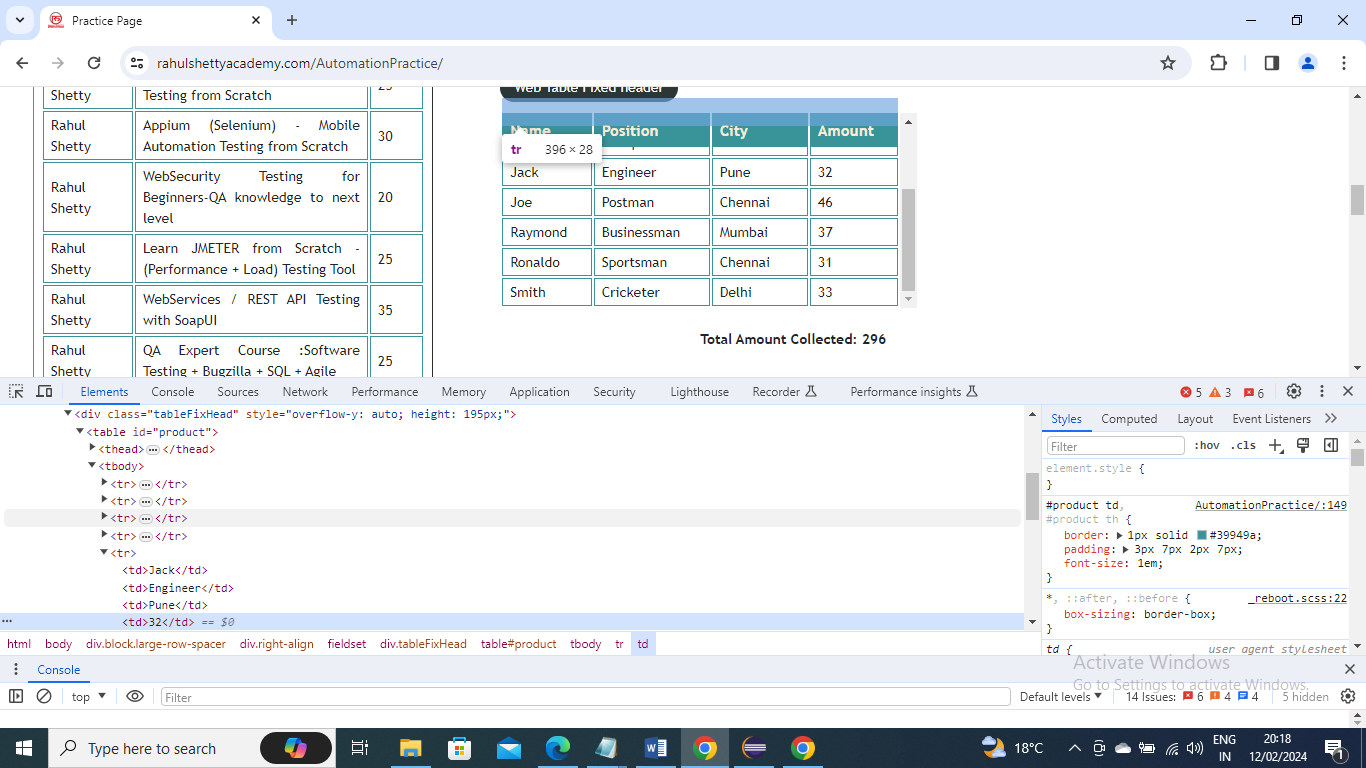
Thread.*sleep*(3000);

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

}

}

Below is an example where we count the values in 4th column and also compare with given total if it is matching the tally or not



**import** java.util.Iterator;

**import** java.util.List;

**import** java.util.Set;

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

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

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

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

**import** org.testng.Assert;

**public** **class** Selenium4 {

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

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

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

**int** sum =0;

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

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

}

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

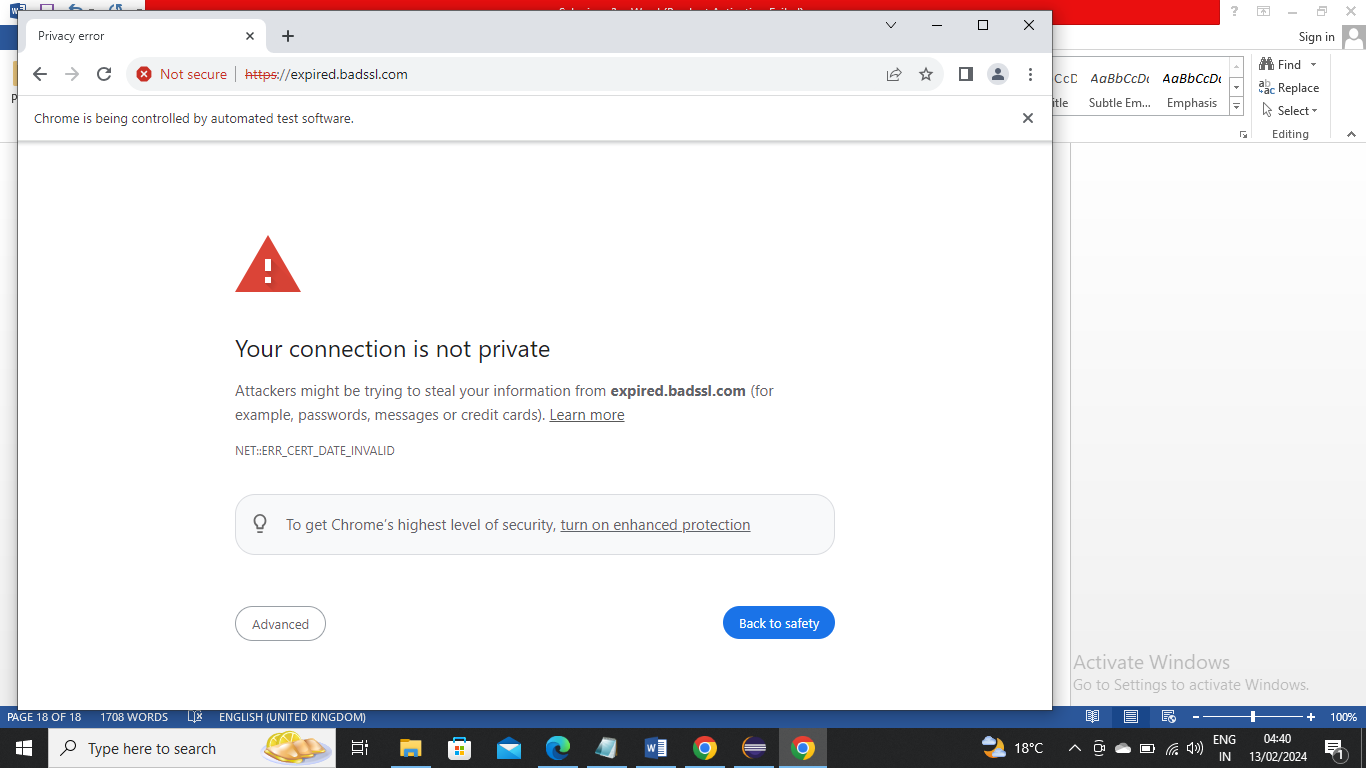
**int** total =Integer.*parseInt*(driver.findElement(By.*cssSelector*(".totalAmount")).getText().split(":")[1].trim());

Assert.*assertEquals*(sum, total);

}

}

Below is an example where certain websites get connection not private



If you write below code directly then also as usually it will give connection not private just like when you open site normally

Below is normal code

**import** java.util.Iterator;

**import** java.util.List;

**import** java.util.Set;

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

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

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

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

**import** org.testng.Assert;

**public** **class** Selenium4 {

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

driver.get("https://expired.badssl.com/");

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

}

}

Below is code where you can accept security certification (HTTP security certification) and go inside it

**import** java.util.Iterator;

**import** java.util.List;

**import** java.util.Set;

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

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

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.openqa.selenium.chrome.ChromeOptions;

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

**import** org.testng.Assert;

**public** **class** Selenium4 {

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

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

ChromeOptions options =**new** ChromeOptions();

options.setAcceptInsecureCerts(**true**);

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

driver.get("https://expired.badssl.com/");

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

}

}

Below is an example on how to delete cookies. In below “sessionkey” is an imaginary cookie name for example for demonstration

**import** java.util.Iterator;

**import** java.util.List;

**import** java.util.Set;

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

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

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.openqa.selenium.chrome.ChromeOptions;

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

**import** org.testng.Assert;

**public** **class** Selenium4 {

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

//delete all cookies

driver.manage().deleteAllCookies();

//delete specific cookie based on that name. Below is example say sessionkey is a cookie name

driver.manage().deleteCookieNamed("sessionkey");

}

}

Below is an example on how to get screenshot using selenium. In below I have took google page screenshot for example

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.Iterator;

**import** java.util.List;

**import** java.util.Set;

**import** org.apache.commons.io.FileUtils;

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

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

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

**import** org.openqa.selenium.OutputType;

**import** org.openqa.selenium.TakesScreenshot;

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.openqa.selenium.chrome.ChromeOptions;

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

**import** org.testng.Assert;

**public** **class** Selenium4 {

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

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

File src = ((TakesScreenshot)driver).getScreenshotAs(OutputType.***FILE***);

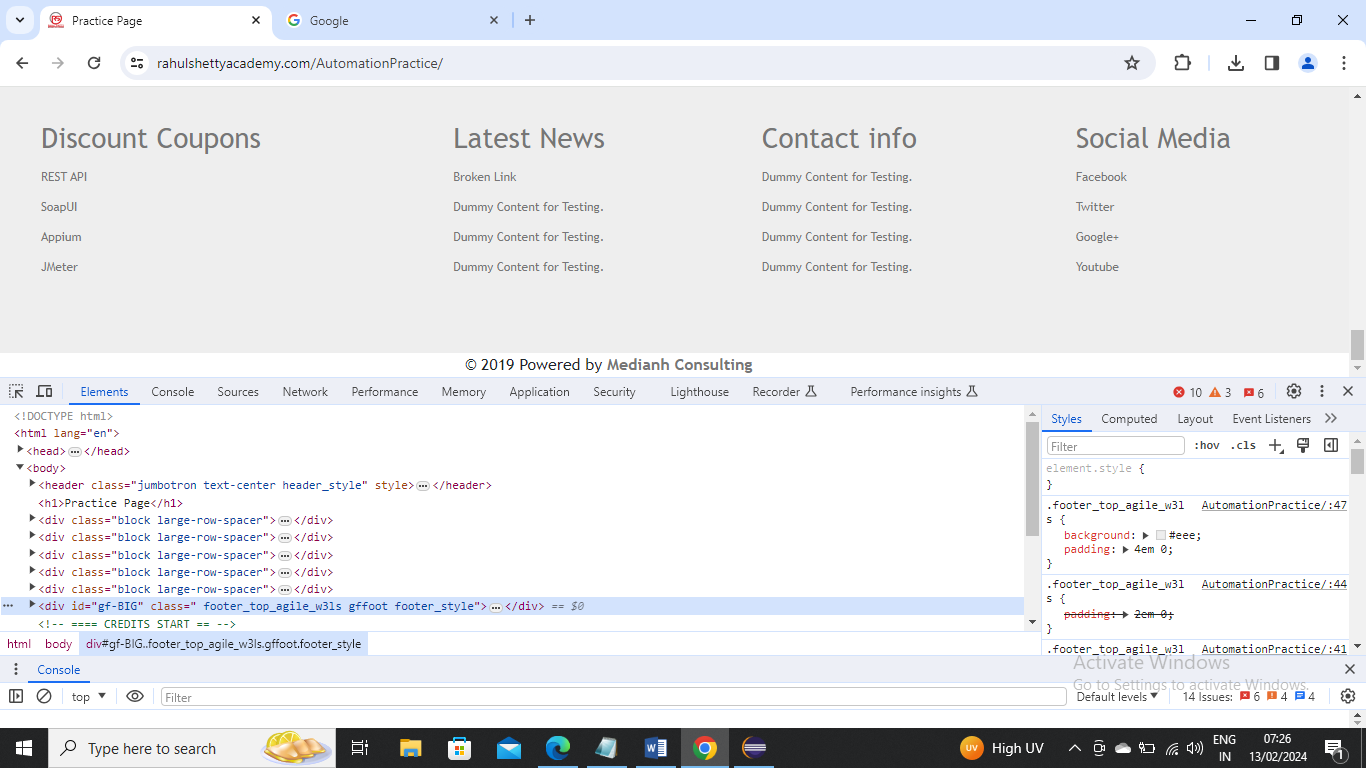
FileUtils.*copyFile*(src, **new** File("E://screenshot.png"));

}

}

In below screenshot we need to get what are all the broken links.

Broken link means that page is not loading or page not found



Any connection value more than 400 means it is broken link and below 400 means that link is not broken

In below example we will just try to get values for couple of links by hard coding it’s css to get the connection value for example

**import** java.io.File;

**import** java.io.IOException;

**import** java.net.HttpURLConnection;

**import** java.net.MalformedURLException;

**import** java.net.URL;

**import** java.util.Iterator;

**import** java.util.List;

**import** java.util.Set;

**import** org.apache.commons.io.FileUtils;

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

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

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

**import** org.openqa.selenium.OutputType;

**import** org.openqa.selenium.TakesScreenshot;

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.openqa.selenium.chrome.ChromeOptions;

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

**import** org.testng.Assert;

**public** **class** Selenium4 {

**public** **static** **void** main(String[] args) **throws** MalformedURLException, IOException {

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

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

String url1= driver.findElement(By.*cssSelector*("a[href\*=soapui]")).getAttribute("href");

//getting SOAPUI link connection value

HttpURLConnection conn1 =(HttpURLConnection)**new** URL(url1).openConnection();

conn1.setRequestMethod("HEAD");

conn1.connect();

**int** respCode1 =conn1.getResponseCode();

System.***out***.println(respCode1);

//getting broken link connection value

String url2= driver.findElement(By.*cssSelector*("a[href\*=brokenlink]")).getAttribute("href");

HttpURLConnection conn2 =(HttpURLConnection)**new** URL(url2).openConnection();

conn2.setRequestMethod("HEAD");

conn2.connect();

**int** respCode2 =conn2.getResponseCode();

System.***out***.println(respCode2);

}

}

Below is the example where we are not hardcoding to see specific link but we are searching all links in generic way and see what all links are broken by running through loop and checking with more than 400 response code condition

**import** java.io.IOException;

**import** java.net.HttpURLConnection;

**import** java.net.MalformedURLException;

**import** java.net.URL;

**import** java.util.List;

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.testng.Assert;

**public** **class** Selenium4 {

**public** **static** **void** main(String[] args) **throws** MalformedURLException, IOException {

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

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

List<WebElement> links= driver.findElements(By.*cssSelector*("li[class='gf-li'] a"));

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

String url = link.getAttribute("href");

HttpURLConnection conn =(HttpURLConnection)**new** URL(url).openConnection();

conn.setRequestMethod("HEAD");

conn.connect();

**int** respCode =conn.getResponseCode();

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

**if**(respCode > 400) {

System.***out***.println("the link broken is " +link.getText()+ "with response code" +respCode);

Assert.*assertTrue*(**false**);

}

}

}

}

Till now we have seen normal Assertion (i.e. hard assertion). Using TestNG there is soft assertion where you can first get all the response codes and after that you can get what condition is not satisfied

SoftAssert s1=**new** SoftAssert();

s1.assertTrue(respCode<400, "the link broken is " +link.getText()+ "with response code" +respCode);

In above we are giving the condition and what message needs to be given if false

S1.assertAll();

The above will now run for all provide all the details and at the end will provide the failed one’s

Below is the example

**import** java.io.IOException;

**import** java.net.HttpURLConnection;

**import** java.net.MalformedURLException;

**import** java.net.URL;

**import** java.util.List;

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.testng.Assert;

**import** org.testng.asserts.SoftAssert;

**public** **class** Selenium4 {

**public** **static** **void** main(String[] args) **throws** MalformedURLException, IOException {

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

SoftAssert s1=**new** SoftAssert();

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

List<WebElement> links= driver.findElements(By.*cssSelector*("li[class='gf-li'] a"));

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

String url = link.getAttribute("href");

HttpURLConnection conn =(HttpURLConnection)**new** URL(url).openConnection();

conn.setRequestMethod("HEAD");

conn.connect();

**int** respCode =conn.getResponseCode();

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

s1.assertTrue(respCode<400, "the link broken is " +link.getText()+ "with response code" +respCode);

}

s1.assertAll();

}

}

Selenium 4.0 features:

Selenium Relative locators --- Friendly locators

above() : Element located above with respect to specified element

below() : Element located above with respect to specified element

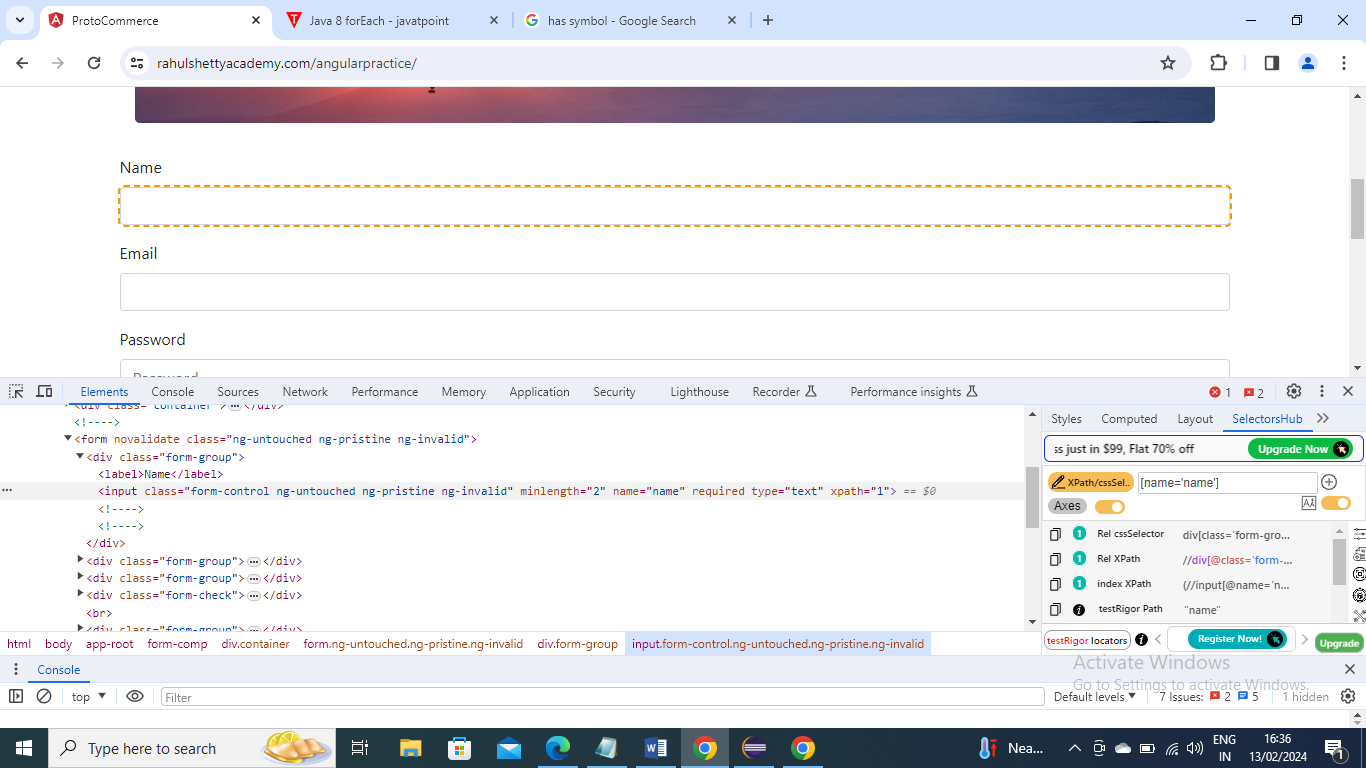
toLeftOf() : Element located above with respect to specified element

toRightOf() : Element located above with respect to specified element

syntax: findElement(withTagName(“XX”).above(WebElement));

Using selenium 4 feature above() method you can get element above the given elemenet information

In the below screenshot we take details of highlighted text box and get the above label (name) information



**import** java.util.List;

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** **static** org.openqa.selenium.support.locators.RelativeLocator.\*;

**public** **class** Selenium4 {

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

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

WebElement nameEditBox = driver.findElement(By.*cssSelector*("[name='name']"));

System.***out***.println(driver.findElement(*with*(By.*tagName*("label")).above(nameEditBox)).getText());

}

}

Below is example using other functions

**import** java.util.List;

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** **static** org.openqa.selenium.support.locators.RelativeLocator.\*;

**public** **class** Selenium4 {

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

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

//above

WebElement nameEditBox = driver.findElement(By.*cssSelector*("[name='name']"));

System.***out***.println(driver.findElement(*with*(By.*tagName*("label")).above(nameEditBox)).getText());

//below

WebElement dateofBirth = driver.findElement(By.*cssSelector*("[for='dateofBirth']"));

driver.findElement(*with*(By.*tagName*("input")).below(dateofBirth)).click();

//leftof

WebElement iceCreamLabel =driver.findElement(By.*xpath*("//label[text()='Check me out if you Love IceCreams!']"));

driver.findElement(*with*(By.*tagName*("input")).toLeftOf(iceCreamLabel)).click();

//rightoff

WebElement rdb = driver.findElement(By.*id*("inlineRadio1"));

System.***out***.println(driver.findElement(*with*(By.*tagName*("label")).toRightOf(rdb)).getText());

}

}

Below is an example to get height and width of an WebElement

**import** java.util.List;

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

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

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

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** **static** org.openqa.selenium.support.locators.RelativeLocator.\*;

**public** **class** Selenium4 {

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

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

System.*setProperty*("weddriver.chrome.driver", "E:\\\\Selenium\\\\chromedriver.exe");

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

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

//above

WebElement nameEditBox = driver.findElement(By.*cssSelector*("[name='name']"));

System.***out***.println(nameEditBox.getRect().getDimension().getWidth());

System.***out***.println(nameEditBox.getRect().getDimension().getHeight());

}

}