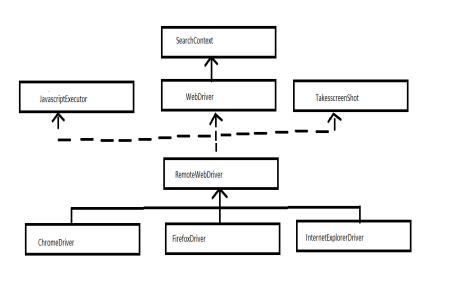
***SELENIUM JAVA ADVANCED IN DETAILS:***

**Java selenium Architecture:**

****

**Interface:**

1. SearchContext

2. JavaScriptExecutor

3. WebDriver

**Classes:**

**1.** RemoteWebDriver

2. ChromeDriver

3. InternetExplorerDriver

∙ SearchContext is the super most interface which contains 2 methods,

1. findElement()

2. findElements()

∙ SearchContext interface is inherited by another interface called WebDriver which  contains 13 methods including findELement() and findElements()

∙ There are other 2 interfaces such as

1. JavascriptExeutor:- which is used to execute javascript statements

2. TakesscreenShot:- which is used to take screenshots of webpage

∙ All these interfaces are implemented in a class called RemoteWebdriver(Super  most class in selenium)

∙ All the methods of RemoteWebdriver class are overridden in respective browser  classes such as, ChromeDriver, FirefoxDriver , InternetExplorerDriver.

**WebDriver methods:**

1 get()

2 getTitle()

3 getCurrentUrl()

4 getPageSource()

5 findElement()

6 findElements()

7 getWindowHandle()

8 getWindowHandles()

9 switchTo()

10 manage()

11 navigate()

12 close()

13 quit()

**JavascriptExecutor methods:**

1 executeScript()

2 executeAsyncScript()

**TakesscreenShot methods:**

1. getScreenShotAs()

**Note:**

1. ChromeDriver class is used to work with chrome browser.

2. FirefoxDriver class is used to work with firefox browser.

3. InternetExplorerDriver class is used to work with ie browser.

***BROWSER LAUNCHING***

1. Open the browser

Firefox browser:

**public class** FacebookAccount {

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

// to configure driver

System.*setProperty*("webdriver.gecko.driver",

"C:/Users/siva/workspace/Selenium/driver/geckodriver.exe");

// create the firefox driver

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

// url mention

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

driver.close();

}

}

**Q1: How does selenium performs the action?**

Ans: By calling native methods of the browsers.

**Q2: How do you call native methods of the browsers?**

Ans: By using driver executable files.

**Close();**

⮚ It is used to close the application.

⮚ It will close the current browser.

**Quit():**

∙ Destroy the object.

∙ It will close all browser windows opened by selenium webdriver

**Chrome Launching**:

**public class** GmailAccount {

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

System.*setProperty*("webdriver.chrome.driver",

"C:/Users/siva/workspace/Selenium/driver/chromedriver.exe");

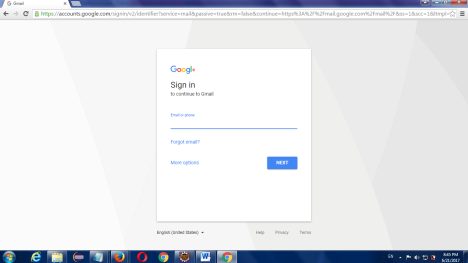
WebDriver driver = **new** ChromeDriver();

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

driver.manage().window().maximize();

}}

Output of the above program:



**IE Launching:**

**public class** IntenetExplore {

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

System.*setProperty*("webdriver.ie.driver",

"C:/Users/siva/workspace/Selenium/driver/IEDriverServer.exe");

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

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

driver.manage().window().maximize();

}

}

Output of the program



**Write a Script to open and close the browser based on user input**

**public class** Demo

{

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

{

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter brower Name:");

String browser = sc.nextLine();

WebDriver driver = **null**;

**if**(browser.equals("Firefox"))

{

System.*setProperty*("webdriver.gecko.driver",

"./drivers/geckodriver.exe");

driver = **new** FirefoxDriver();

}

**else**

**if**(browser.equals("Chrome"))

{

System.*setProperty*("webdriver.chrome.driver",

"./drivers/chromedriver.exe");

driver = **new** ChromeDriver();

}

**else**

{

System.***out***.println("Invalid browser");

}

Thread.*sleep*(2000);

driver.close();

}

}

Note: The above script is an example for Run Time Ploymorphism.

∙ To run same script on multiple browsers, we are converting sub class object into interface  type(upcasting).

WebDriver driver = new ChromeDriver();

WebDriver driver = new FirefoxDriver();

***WEB DRIVER METHODS:***

**Methods of WebDriver Interface:**

|  |  |  |
| --- | --- | --- |
| 1 | get() | To enter the url |
| 2 | getTitle() | To get the title of current web page |
| 3 | getCurrentUrl() | To get the url of current web page |
| 4 | getPageSource() | To get the page source of current web page |
| 5 | findElement() | To get single webElements |
| 6 | findElements() | To get multiple webElements |
| 7 | getWindowHandle() | To get the id of parent window |
| 8 | getWindowHandles() | To get the id of All windows |
| 9 | switchTo() | Used to switch one window to other window |
| 10 | manage() | 1. Window  2. Cookies |
| 11 | navigate() | 1. Enter the URL  2. Navigate to previous page  3. Navigate to next page  4. Refresh current web page |
| 12 | close() | To close the current/parent browser |
| 13 | quit() | To close all the browsers opened by selenium |

**public class** Demo

{

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

//open the browser

System.*setProperty*("webdriver.chrome.driver",

"./drivers/chromedriver.exe");

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

//enter the url

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

//To get the title of current web page

String title = driver.getTitle();

System.***out***.println("Title: "+title);

//To get the url of current web page

String url = driver.getCurrentUrl();

System.***out***.println("URL: "+url);

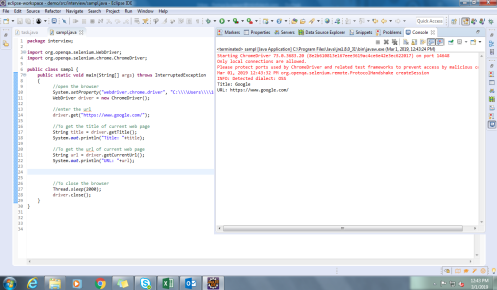
//To close the browser

Thread.*sleep*(2000);

driver.close();

}

}



***NAVIGATION COMMANDS***

**1.** Navigate().to()

2. Refresh()

3. Back()

4. Forword()

**Note it:**

//To delete cookies

driver.manage().deleteAllCookies();

**Difference Between get() and navigate():**

|  |  |
| --- | --- |
| get | navigate |
| It will just enter the URL | 1. It will enter the URL  2. It will navigate to previous page 3. It will navigate to next page  4. It will refresh the current web page |
| After entering the URL it will  not allow any statements to  execute untill the page loads  completely | After entering the URL it will not wait  untill the page loads completely |

**Inspect the element:**

∙ Fetching the source code of an element is called as inspect the element.

∙ To inspect the elements, Right click on element🡪 click on Inspect element, which will  give source code of that element

∙ In some applications, for security purpose right click option will be disabled. In such cases ⮚ Press F12 which will open developers tool

⮚ Select inspect button(mouse icon available on top left corner)

⮚ click on the element

**Methods of WebEleent Interface:**

|  |  |  |
| --- | --- | --- |
| 1 | sendKeys() | 1.To enter the value in textbox  2. To Handle some keyboard action |
| 2 | clear() | To clear the textbox value |
| 3 | Click() | To click the particular webElement(Button) |
| 4 | getCssValue() | To get the color/size/font of the particular  webelement |
| 5 | findElement() | To get single webElements |
| 6 | findElements() | To get multiple webElements |
| 7 | getText() | To get the text of the particular webelement |
| 8 | getAttribute() | To get the text of the particular attribute(id,  name, value…etc) |
| 9 | getTagName() | To get the tagname of particular webelement |
| 10 | getLocation() | To get the X axis and Y axis location of  particular webelement |
| 11 | getSize() | To get the size of particular  webelement(textbox, text…etc) |
| 12 | isDisplayed() | To check whether the particular webelement  is displaying or not( logo, textbox, text…etc) |
| 13 | isEnabled() | To check whether the textbox is enabled to  enter the text or not |
| 14 | isSelected() | To check whether the radiobutton/dropdown  is selected or not |
| 15 | submit() | To click on an element only if the type of  the element is submit  Ex: <input type="submit" id="s"  value="Submit"> |

***LOCATORS:***

∙ Static methods which are used to identify the elements which are present the webpage. ∙ All these locators are present in a class called **By** which is an Abstract class. ∙ There are 8 types of locators and all the locators takes argument of type string. They are, 1. Id(String)

2. name(String)

3. className(String)

4. tagName(String)

5. linkText(String)

6. partialLinkText(String)

7. cssSelector(String)

8. xpath(String)

**Note**:

∙ Id, name, className are available as attributes of an element.

∙ In order to Handle the single element we use findElement().

∙ Return type of findElement() is WebElement.

∙ In findElement(), if the specified locator is not matching with any element it will throw  NoSuchElementException

∙ In findElement(), if the specified locator is matching with multiple element it will return  the address of 1st matching element

∙ If the specified element is link, then we can identify that element by using linkText ∙ If the specified element is link and if it partially dynamic, then we can identify that  element by using partialLinkText

**cssSelector**

∙ If we can not identify the elements by using any of the above locators, then we can identify  that element by using cssSelector.

∙ Syntax:

∙ tagnName[attributeName=’attributeValue’]

ex: input[type='password']

∙ In order to verify cssSelector expression in firefox browser, click on TX🡪select  queryselectorAll option🡪 specify the expression in expression field and click on enter

**Note**:

∙ In cssSelector,

o Id can be represented by using #,

Ex:- input#email

o Class can be represented by using .

Ex:- tagName.classValue

Ex,

**Adactin hotel login using id/name:**

**public class** Login {

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

System.*setProperty*("webdriver.chrome.driver",

"C:\\Users\\10657527\\Downloads\\chromedriver\_win32

(1)\\chromedriver.exe");

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

driver.get("https://adactin.com/HotelApp/index.php");

driver.manage().window().maximize();

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

x.sendKeys("vengatram");

WebElement x1 = driver.findElement(By.*name*("password"));

x1.sendKeys("vengat@123445");

WebElement x2 = driver.findElement(By.*id*("login"));

x2.click();

}

}

**LinkText:**

<a href="ForgotPassword.php">Forgot Password?</a>

Text: >Forgot Password?<

linkText: Text which is available in **a** tag called linktext.

**public class** Login {

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

System.*setProperty*("webdriver.chrome.driver",

"C:\\Users\\10657527\\Downloads\\chromedriver\_win32

(1)\\chromedriver.exe");

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

driver.get("https://adactin.com/HotelApp/index.php");

driver.manage().window().maximize();

WebElement x2 = driver.findElement(By.*linkText*("Forgot Password?")); x2.click();

}

}

**PartialLinkText:**

**package** com.lnt.test;

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

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

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

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

**public class** Login {

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

System.*setProperty*("webdriver.chrome.driver",

"C:\\Users\\10657527\\Downloads\\chromedriver\_win32

(1)\\chromedriver.exe");

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

driver.get("https://adactin.com/HotelApp/index.php");

driver.manage().window().maximize();

WebElement x2 = driver.findElement(By.*partialLinkText*("Forgot")); x2.click();

}

}

***Xpath:***

∙ Path of an element present in the webpage.

o Absolute

o Relative

**Absolute**:

∙ Complete path of an element from root of the webpage(html)

∙ Represented by using /--> immediate child

**Ex:**

/html/body/div[1]/input[2]

**Relative xpath:**

∙ Path of any element which is present on web page.

∙ It is represented by using // which means any child/element

Syntax:

1. //tagName🡪 all the matching elements

2. //tagName[1]🡪 all the 1st matching elements

3. //tagName[last()]🡪 all the last matching elements

4. //\*🡪 all the elements

5. //\*[@attribute=’value’]

6. //tagName[@attribute=’value’]

**Ex:**

//div[1]/input[2]

//div[1]/input

//div[2]/input

//input[1]

//div[1]/input[2]| //div[2]/input

//input

**Xpath by attribute:**

∙ To identify the specified elements, if we use index it may not work properly when we use  the index values because whenever the position of an element changes its index value will  also changes.

∙ To overcome the above problem in place of index we can include attributes which is called  as xpath by attributes.

∙ It is applicable for both Absolute and Relative xpath.

∙ **Syntax**:

o tagName[@attributeName=’attributeValue’]

∙ **Example**:

o Absolute🡪 /html/body/div/input[@value='B']

o Relative🡪 //input[@value='B']

∙ In an xpath we can pass multiple attributes by using or operator.

∙ **Example**:

o //input[@value='B' or @value='C']

***getAttribute() and getText():***

It is a method, used to print the value whatever you gave in the text box ***Example program:***

**public class** Ex5 {

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

System.*setProperty*("webdriver.gecko.driver","C:/Users/siva/workspace/Selenium/ driver/geckodriver.exe");

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

driver.get("http://www.adactin.com/HotelApp/index.php");

driver.findElement(By.*id*("username")).sendKeys("vengat16");

driver.findElement(By.*id*("password")).sendKeys("Karthick");

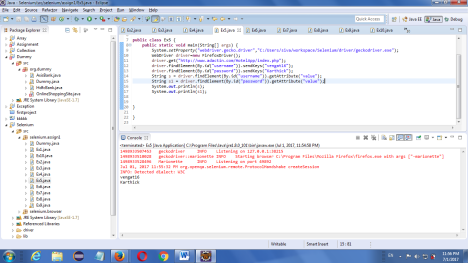
String s = driver.findElement(By.*id*("username")).getAttribute("value"); String s1 = driver.findElement(By.*id*("password")).getAttribute("value"); System.***out***.println(s);

System.***out***.println(s1);

}

}

Output:



***NoSuchElementException:***

⮚ It is throws when particular id/xpath/class or whatever is not available in DOM  structure

***isDisplayed():***

It is a method, used to check the particular id/value is available or not

***Example program:***

**public class** Ex6 {

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

System.*setProperty*("webdriver.gecko.driver","C:/Users/siva/workspace/Selenium/ driver/geckodriver.exe");

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

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

**boolean**

logo=driver.findElement(By.*xpath*("//\*[@id='blueBarDOMInspector']/div/div/div/div[1]/h 1/a/i")).isDisplayed();

**if**(logo==**true**)

{

System.***out***.println("logo is available");

}

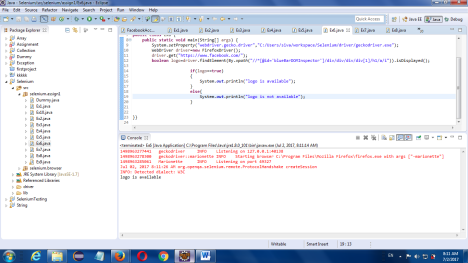
**else**{

System.***out***.println("logo is not available");

}

}}

Output:



***isEnabled:***

⮚ It is a method, is used to check particular text box is enable to print or not ***Example program:***

**public class** Dummy {

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

System.*setProperty*("webdriver.gecko.driver",

"C:/Users/siva/workspace/Selenium/driver/geckodriver.exe");

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

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

**boolean** logo = driver.findElement(By.*xpath*("//\*[@id='email']"))

.isEnabled();

**if** (logo == **true**) {

System.***out***.println("Text box is enable to print");

} **else** {

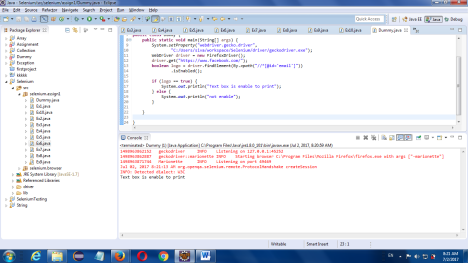
System.***out***.println("not enable");

}

}

}

Output:



***isSelected:***

⮚ It is a method, used to check the particular radio button is selected or not

***Example program:***

**public class** Dummy {

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

System.*setProperty*("webdriver.gecko.driver",

"C:/Users/siva/workspace/Selenium/driver/geckodriver.exe");

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

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

driver.findElement(By.*xpath*("//\*[@id='u\_0\_g']"))

.click();

**boolean** logo = driver.findElement(By.*xpath*("//\*[@id='u\_0\_g']"))

.isSelected();

**if** (logo == **true**) {

System.***out***.println("button is selected");

} **else** {

System.***out***.println("not selected");

}

}}

**Xpath by text():**

∙ If the specified element does not contain any attributes and if it contains text then we can  identify that element by using xpath by text()

∙ It is applicable for both absolute and relative xpath

∙ Syntax:

o tagName[text()=’textValue’]

∙ Example:

o //td[text()='Java']

∙ text() can be represented by using dot(.)

∙ Example:

o //td[.='Java']

∙ Attribute values and the text values are case and space sensitive.

∙ Example:

o //div[text()='Login ']

**Xpath by contains():**

∙ It is used to handle the partial dynamic elements

∙ It is applicable for both absolute and relative xpath.

∙ **Syntax 1**: if text value is partially dynamic,

o **tagName[contains(text(),'textValue')]**

∙ **Example:**

o //nobr[contains(text(),'actiTIME')]

∙ **Syntax 2:** if attribute value is partially dynamic,

o **tagName[contains(@attributeName,'attributeValue')]**

∙ **Example:**

o **//img[contains(@src,'/img/default/login/timer.png?hash')]**

***Radio button:***

⮚ We can able to select only one at a time

***Example program:***

**public class** Dummy {

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

System.*setProperty*("webdriver.gecko.driver",

"C:/Users/siva/workspace/Selenium/driver/geckodriver.exe");

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

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

driver.findElement(By.*xpath*("//\*[@id='u\_0\_g']"))

.click();

}}

**getCssValue():**

It is used to get the css property (font, color, size) of a web element.

**public class** Login {

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

System.*setProperty*("webdriver.chrome.driver",

"C:\\Users\\10657527\\Downloads\\chromedriver\_win32

(1)\\chromedriver.exe");

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

driver.get("https://adactin.com/HotelApp/index.php");

driver.manage().window().maximize();

WebElement x =

driver.findElement(By.*xpath*("//td[@class='build\_title']"));

String x1 = x.getCssValue("font-size");

System.***out***.println(x1);

String x2 = x.getCssValue("color");

System.***out***.println(x2);

String x3 = x.getCssValue("font-weight");

System.***out***.println(x3);

String x4 = x.getCssValue("font-family");

System.***out***.println(x4);

String x5 = x.getCssValue("background");

System.***out***.println(x5);

}

}

**CHECK BOX:**

⮚ In check box, we can able to select more than one value at a time.

**Select one value:**

***Eample program:***

**public class** Dummy {

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

System.*setProperty*("webdriver.gecko.driver","C:/Users/siva/workspace/Selenium/ driver/geckodriver.exe");

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

driver.get("http://demoqa.com/registration/");

driver.findElement(By.*xpath*(".//input[@value='dance']")).click();

}

}

**Difference between findElement() & findElements():**

|  |  |
| --- | --- |
| **findElement()** | **findElements()** |
| To handle single element | To handle multiple elements |
| Return type is WebElement | Return type is List<WebElement> |
| if the specified locator is  matching  multiple elements then it returns  address of 1st matching element | if the specified locator is  matching  multiple elements then it returns  address of all the matching  element |
| if the specified locator is not  matching then it returns  NoSuchElementException | if the specified locator is not  matching then it returns empty  list(0) |

**DROP DOWN:**

**1.Single value**

**2.Multiple value**

∙ If the list box is developed by using select tag then we can handle it by using Select class. ∙ Select class should be imported from the package org.openqa.selenium.support.ui ∙ Select class contains one constructor which takes an argument of type WebElement where  in we have to pass address of the list box.

∙ Select class contains some methods. They are,

|  |  |  |
| --- | --- | --- |
| 1 | selectByIndex(int) | Select the options |
| 2 | selectByValue(String) |
| 3 | selectByVisibleText(String) |
| 4 | deselectByIndex(int) | Deselect the options |
| 5 | deselectByValue(String) |
| 6 | deselectByVisibleText(String) |
| 7 | deselectAll() |
| 8 | getAllSelectedOptions() | To get all the selected options |
| 9 | getFirstSelectedOption() | To get first selected options |
| 10 | getOptions() | To get all the options |
| 11 | isMultiple() | To check whether list box is single or multi select |

***1.SINGLE VALUE***

**To print all the options:**

***Eample program:***

**public class** Dummy {

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

System.*setProperty*("webdriver.gecko.driver",

"C:/Users/siva/workspace/Selenium/driver/geckodriver.exe");

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

driver.get("http://ironspider.ca/forms/dropdowns.htm");

WebElement w = driver.findElement(By.*name*("coffee"));

Select s=**new** Select(w);

List<WebElement> o = s.getOptions();

**for** (WebElement x:o) {

System.***out***.println(x.getAttribute("value"));

}

}

}

**SELECT:**

**We can perform select by 3 ways**

**1. SelectByIndex**

**2. SelectByValue**

**3. SelectByVisibletext**

**1.selectByIndex:**

***Eample program:***

**public class** Dummy {

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

System.*setProperty*("webdriver.gecko.driver",

"C:/Users/siva/workspace/Selenium/driver/geckodriver.exe"); WebDriver driver = **new** FirefoxDriver();

driver.get("http://ironspider.ca/forms/dropdowns.htm"); WebElement w = driver.findElement(By.*name*("coffee")); Select s=**new** Select(w);

s.selectByIndex(3);

}

}

**2.selectByValue:**

***Eample program:***

**public class** Dummy {

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

System.*setProperty*("webdriver.gecko.driver",

"C:/Users/siva/workspace/Selenium/driver/geckodriver.exe");

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

driver.get("http://ironspider.ca/forms/dropdowns.htm");

WebElement w = driver.findElement(By.*name*("coffee"));

Select s=**new** Select(w);

s.selectByValue("regular");

}

}

**3.selectByVisibletext:**

***Eample program:***

**public class** Dummy {

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

System.*setProperty*("webdriver.gecko.driver",

"C:/Users/siva/workspace/Selenium/driver/geckodriver.exe"); WebDriver driver = **new** FirefoxDriver();

driver.get("http://ironspider.ca/forms/dropdowns.htm"); WebElement w = driver.findElement(By.*name*("coffee")); Select s=**new** Select(w);

s.selectByVisibleText("With cream & sugar");

}

}

***2.MULTIPLE VALUE***

***isMultiple():***

⮚ It is a method, used to check we can able to select multiple values or not ***Eample program:***

**public class** Dummy {

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

System.*setProperty*("webdriver.gecko.driver",

"C:/Users/siva/workspace/Selenium/driver/geckodriver.exe");

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

driver.get("http://ironspider.ca/forms/dropdowns.htm");

WebElement w = driver.findElement(By.*name*("coffee2"));

Select s=**new** Select(w);

**boolean** b = s.isMultiple();

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

}

}