Test Automation & Advanced Selenium

Lesson 5: Testing Web Applications Using Web Driver API

Lesson Objectives

- Writing first Web Driver Test
- Locating UI Elements-Developers Tools
- Navigation API
  - get
  - navigate
- Interrogation API
  - getTitle
  - getCurrentUrl
  - getPageSource
- WebElement API
  - findElement & findElements
  - By
    - id
    - xpath
    - cssSelector
    - className
    - linkText, name, tagName, partialLinkText

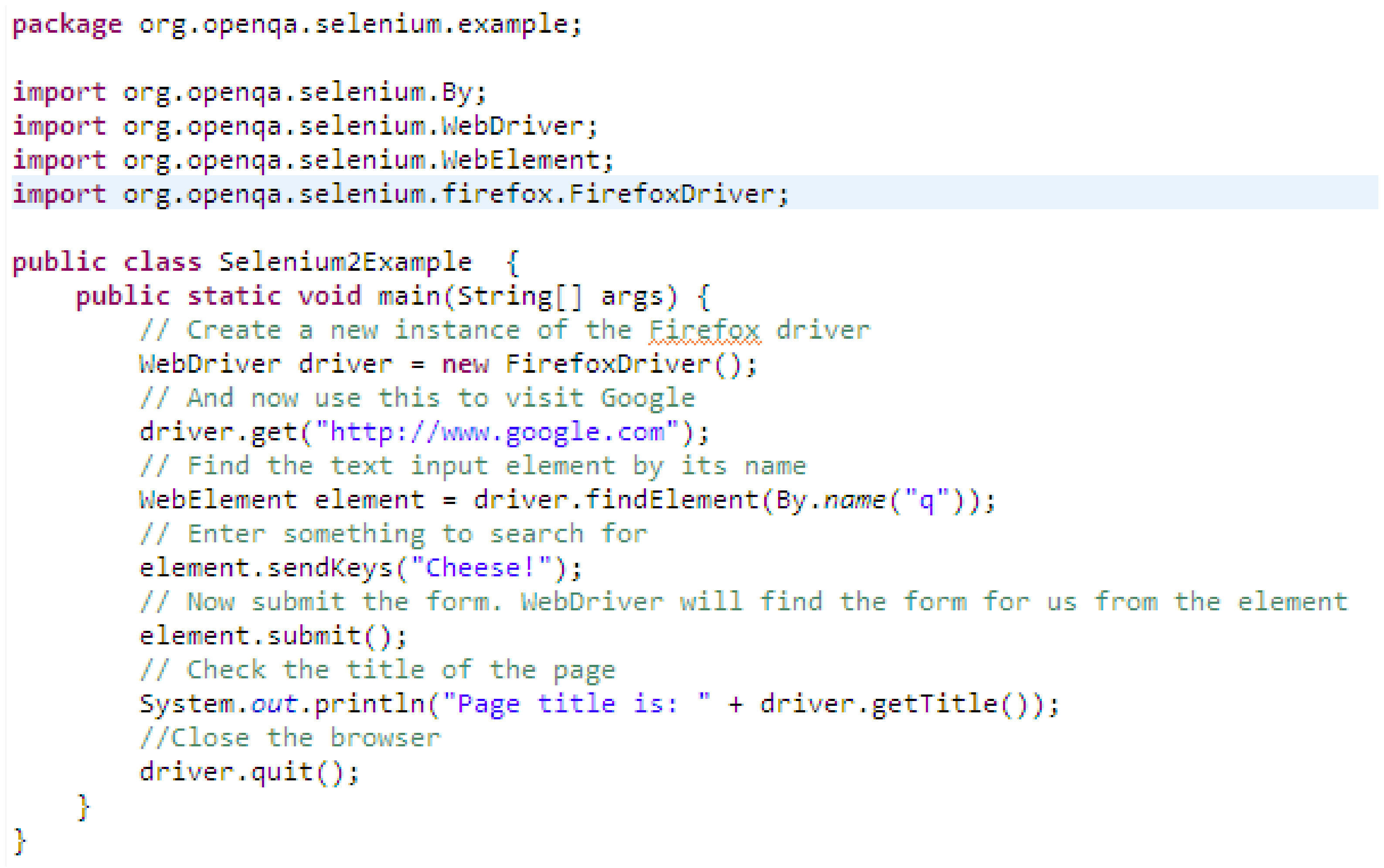
Lesson Objectives (Cont.)

- WebElement API
  - click
  - clear
  - sendKeys
  - submit
  - Select – selectByVisibleText etc.
  - getText
  - getAttribute
- Handling Popup Dialogs and Alerts
- Windows
  - getWindowHandle and getWindowHandles
  - switchTo
  - manage
- Alerts
  - switchTo
  - dismiss
  - accept
- Why synchronization is important

Lesson Objectives (Cont.)

- Using Explicit & Implicit Wait
  - Expected Condition & Expected Conditions
  - WebDriverWait
  - ImplicitlyWait
  - pageLoadTimeout
- JavaScript Executor

Writing first Web Driver Test(Java)

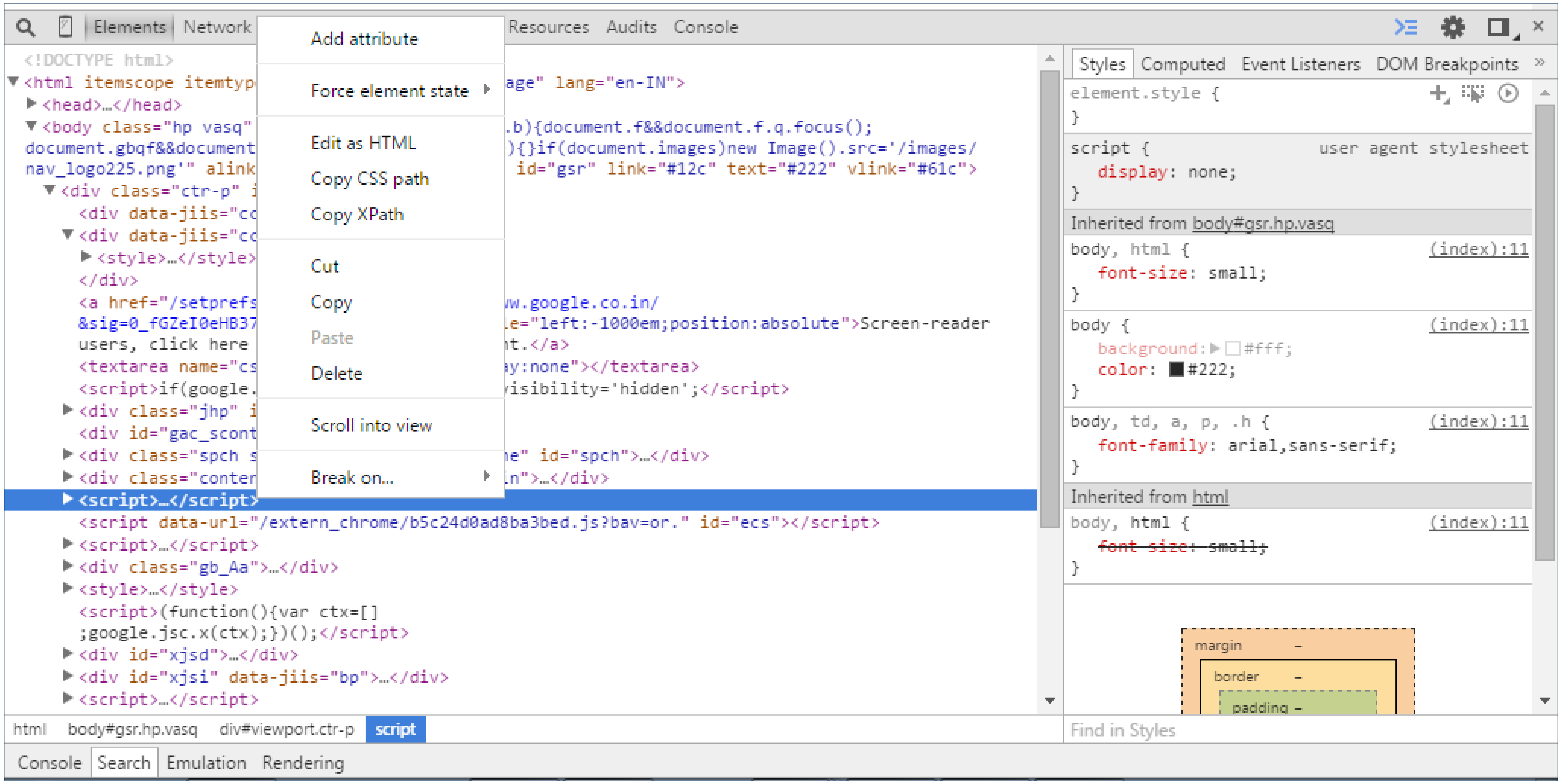


```java
package org.openqa.selenium.example;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;

public class Selenium2Example  {
    public static void main(String[] args) {
        // Create a new instance of the Firefox driver
        WebDriver driver = new FirefoxDriver();
        // And now use this to visit Google
        driver.get("http://www.google.com");
        // Find the text input element by its name
        WebElement element = driver.findElement(By.name("q"));
        // Enter something to search for
        element.sendKeys("Cheese!");
        // Now submit the form. WebDriver will find the form for us from the element
        element.submit();
        // Check the title of the page
        System.out.println("Page title is: " + driver.getTitle());
        //Close the browser
        driver.quit();
    }
}
```

Locating UI Elements-Developers Tools



```
<!DOCTYPE html>
▼<html itemscope itemtyp                age" lang="en-IN">
  ▶<head>…</head>
  ▼<body class="hp vasq"                .b){document.f&&document.f.q.focus();
  document.gbqf&&documen                ){}if(document.images)new Image().src='/images/
  nav_logo225.png'" alink                id="gsr" link="#12c" text="#222" vlink="#61c">
    ▼<div class="ctr-p" i
       <div data-jiis="cc
      ▼<div data-jiis="cc
        ▶<style>…</style>
       </div>
       <a href="/setprefs               ww.google.co.in/
       &sig=0_fGZeI0eHB37               .e="left:-1000em;position:absolute">Screen-reader
       users, click here                nt.</a>
       <textarea name="cs               ay:none"></textarea>
       <script>if(google.               visibility='hidden';</script>
      ▶<div class="jhp" i
       <div id="gac_scont
      ▶<div class="spch s               e" id="spch">…</div>
      ▶<div class="conter               n">…</div>
      ▶<script>…</script>
       <script data-url="/extern_chrome/b5c24d0ad8ba3bed.js?bav=or." id="ecs"></script>
      ▶<script>…</script>
      ▶<div class="gb_Aa">…</div>
      ▶<style>…</style>
       <script>(function(){var ctx=[]
       ;google.jsc.x(ctx);})();</script>
      ▶<div id="xjsd">…</div>
      ▶<div id="xjsi" data-jiis="bp">…</div>
      ▶<script>…</script>
```

html body#gsr.hp.vasq div#viewport.ctr-p script

```
element.style {
}
script {                          user agent stylesheet
   display: none;
}
Inherited from body#gsr.hp.vasq
body, html {                      (index):11
   font-size: small;
}
body {                            (index):11
   background: ▶□#fff;
   color: ■#222;
}
body, td, a, p, .h {              (index):11
   font-family: arial,sans-serif;
}
Inherited from html
body, html {                      (index):11
   ~~font-size: small;~~
}
```

Navigation API

- driver.get("URL")
  - Required to navigate to a page
  - E.g.: driver.get("http://www.google.com");
  - WebDriver will wait until the page has fully loaded before returning control to your test or script
  - to ensure page is fully loaded then wait commands can be used

- driver.navigate().to("URL")
  - E.g.: driver.navigate().to("http://www.google.com");
  - Other Navigate commands
  - driver.navigate().refresh();
  - driver.navigate().forward();
  - driver.navigate().back();

Interrogation API

- driver.getTitle()
  - Get the title of the current page
- driver. getCurrentUrl()
  - Get the current URL of the browser
- driver.getPageSource()
  - Get the source code of the page

- Syntax:

```
      public void testTitleReliability() {
      driver.get("https://www.google.com");
                 boolean title = driver.getTitle().contains("Google");
                 if(title)
                String currentURL = driver.getCurrentUrl();
                (If you want to verify a particular text is present or not on the page,do as below)
                boolean b = driver.getPageSource().contains("your text");
                System.out.println("Expected title is present ");
                else if(!title)
      System.out.println(" Expected title is not present"); ");
      }
```

WebElement API

- findElement:
  - Used to locate single element and return WebElement object of first occurrences element on web page
  - If element not found, throw s exception NoSuchElementException
  - Syntax: findElement(By by)

Example:

```
WebElement element = driver.findElement(By.id("Home"));
element.click();
```

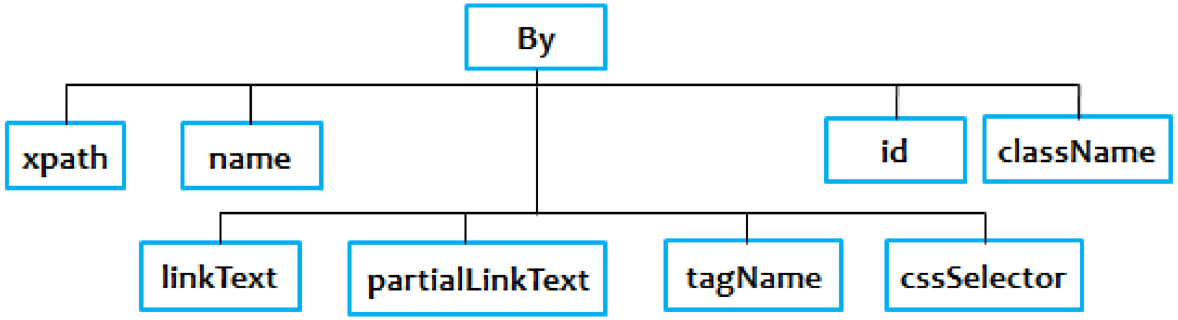
- findElements:
  - Used to find multiple element on webpage, e.g.: count total number of row in table
  - Returns List of WebElement object of all occurrences of element
  - If element not found, returns empty List of WebElement object
  - Syntax: List element = findElements(By by)

Example:

```
List {WebElement} element = driver.findElement(By.xpath("//table/tr"));
int size = element.size();
```

WebElement API (Cont.)

- By:
  - A collection of factory functions for creating webdriver.Locator instances



- By id: Locates an element by its ID
  - Syntax: driver.findElement(By.id("element id"))
- By className: Locates elements that have a specific class name
  - Syntax : driver.findElement(By.className("element class"))
- By name: Locates elements whose name attribute has the given value
  - Syntax : driver.findElement(By.name("element name"))

WebElement API (Cont.)

- By XPath: Locates elements matching a XPath selector.
  - For example, given the selector "//div", WebDriver will search from the document root regardless of whether the locator was used with a WebElement
  - Syntax: driver.findElement(By.xpath("xpath expression"))
- By linkText :Locates link elements whose visible text matches the given string
  - Syntax : driver.findElement(By.link("link text"))
- By partialLinkText: Locates link elements whose visible text contains the given substring
  - Syntax : driver.findElement(By. partialLinkText("link text"))
- By tagName: Locates elements with a given tag name.
  - The returned locator is equivalent to using the getElementsByTagName DOM function
  - Syntax : driver.findElement(By.tagName("element html tag name"))
- By CSS Selector: Locates elements with a given tag name.
  - Syntax : driver.findElement(By.cssSelector("css selector"))

WebElement API (Cont.)

- Click():
  - For Example: Login button is available on login screen
  - Syntax:

```
WebElement click = driver.findElement(By.xpath("//*[@id='btnLogOn']"));
click.click();
```

- Scenarios where Click() is used:
  - “Check / Uncheck “ a checkbox
  - Select a radio button

- Clear():
  - Function sets the value property of the element to an empty string ('')
  - Syntax:

```
driver.findElement(By.xpath("//*[@id=“textBox”]")).clear();
```

- SendKeys():
  - Method is used to simulate typing into an element, which may set its value
  - Syntax:

```
driver.findElement(By.id(“NameTextBox")).sendKeys(“Rahul”);
```

WebElement API (Cont.)

- SendKeys():
  - Scenarios where sendKeys() is used:
    - Sending special characters (Enter , F5, Ctrl, Alt etc..)
    - Key events to WebDriver
    - Uploading a file

    Syntax:

    //Sending Ctrl+A

    driver.findElement(By.xpath("//body")).sendKeys(Keys.chord(Keys.CONTROL, "a"));

    //Sending pagedown key from keyboard

    driver.findElement(By.id("name")).sendKeys(Keys.PAGE_DOWN);

    //Sending space key

    driver.findElement(By.id("name")).sendKeys(Keys.SPACE);
- Submit():
  - If form has submit < type=" " value=" ">stead of type = "button" then .submit() method will work
  - If button Is not Inside <form> tag then .submit() method will not work.

  Syntax:

  driver.findElement(By.xpath("//input[@name='Company']")).submit();

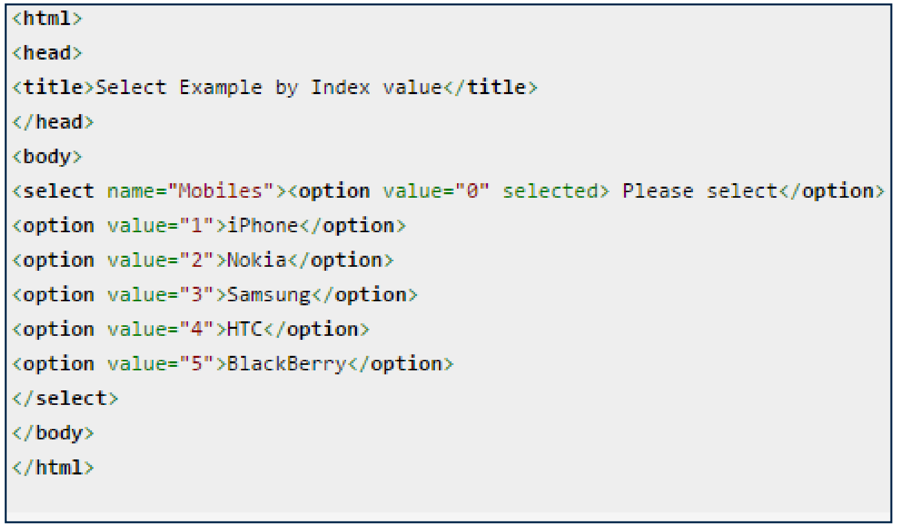
WebElement API (Cont.)

- Select:
  - WebDriver’s support classes
  - Used to work with Dropdowns

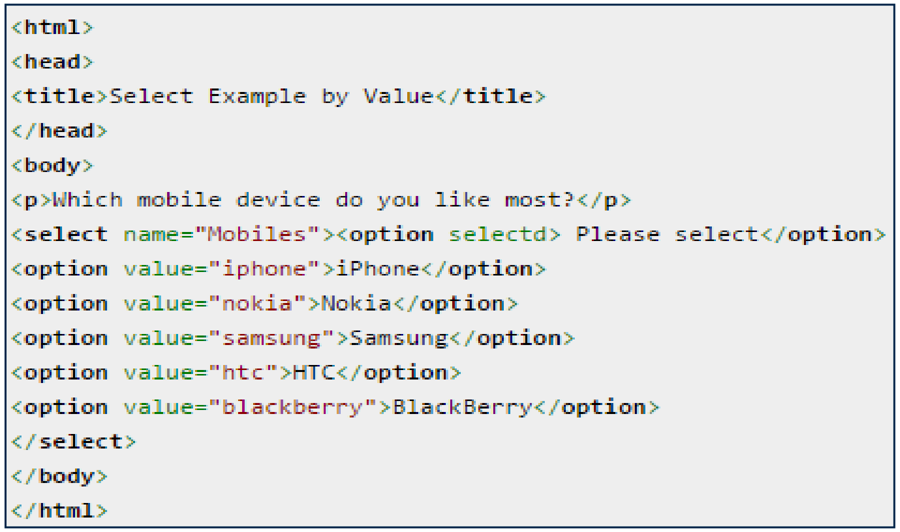
  - Method Name: selectByIndex
  - Syntax: select.selectByIndex(Index);

  - Method Name: selectByValue
  - Syntax: select.selectByValue(Value);

  - Method Name: selectByVisibleText
  - Syntax: select.selectByVisibleText(Text);



```html
<html>
<head>
<title>Select Example by Index value</title>
</head>
<body>
<select name="Mobiles"><option value="0" selected> Please select</option>
<option value="1">iPhone</option>
<option value="2">Nokia</option>
<option value="3">Samsung</option>
<option value="4">HTC</option>
<option value="5">BlackBerry</option>
</select>
</body>
</html>
```



```html
<html>
<head>
<title>Select Example by Value</title>
</head>
<body>
<p>Which mobile device do you like most?</p>
<select name="Mobiles"><option selectd> Please select</option>
<option value="iphone">iPhone</option>
<option value="nokia">Nokia</option>
<option value="samsung">Samsung</option>
<option value="htc">HTC</option>
<option value="blackberry">BlackBerry</option>
</select>
</body>
</html>
```

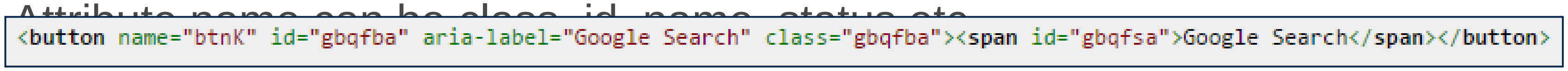
WebElement API (Cont.)

- getText():
  - Get the text content from a DOM-element found by given selector
  - Make sure the element you want to request the text from is interact able otherwise empty string is returned

  Syntax:

  WebElement TxtBoxContent = driver.findElement(By.id(“WebelementID”));

  TxtBoxContent.getText();
- getAttribute():
  - getText() will only get the inner text of an element
  - To get the value, you need to use getAttribute(“attribute name”)
  - Attribute name can be class, id, name, status etc.



```
<button name="btnK" id="gbqfba" aria-label="Google Search" class="gbqfba"><span id="gbqfsa">Google Search</span></button>
```

  Syntax:

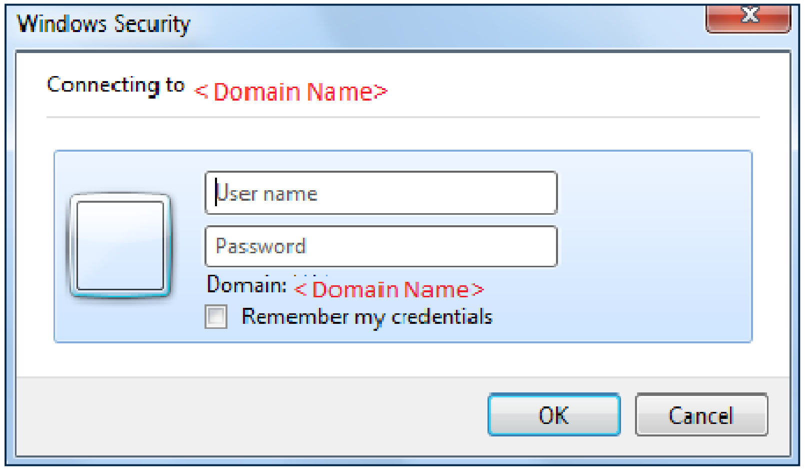
  WebElement TxtBoxContent = driver.findElement(By.id(WebelementID));

  System.out.println("Printing " + TxtBoxContent.getAttribute(“class"));

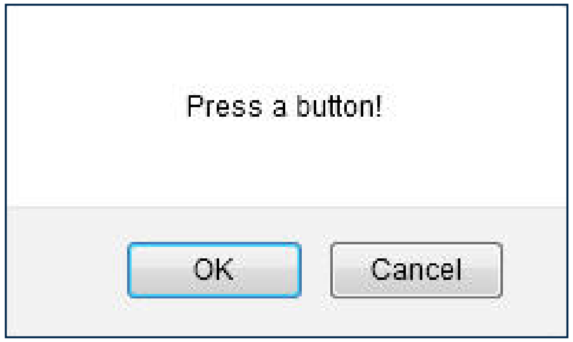
Handling Popup Dialogs and Alerts

Two types of alerts:

- Windows based alert pop ups
  - Selenium will not be able to recognize it, since it is an OS-level dialog



- Web based alert pop ups
  - Can be Alert box/ Pop up box/ confirmation Box/ Prompt/ Authentication Box
  - Alert interface gives us following methods to deal with the alert
    - accept() : To accept the alert
    - dismiss() : To dismiss the alert
    - getText() : To get the text of the alert
    - sendKeys() : To write some text to the alert



Windows

- Multiple windows are handled by switching the focus from one window to another
- **Syntax:**

```
// Opening site
  driver.findElement(By.xpath("//img[@alt='SeleniumMasterLogo']")).click();
// Storing parent window reference into a String Variable
  String Parent_Window = driver.getWindowHandle();
// Switching from parent window to child window
  for (String Child_Window : driver.getWindowHandles())
  {
        driver.switchTo().window(Child_Window);
// Performing actions on child window
        driver.findElement(By.id("dropdown_txt")).click();
        driver.findElement(By.xpath("//*[@id='anotherItemDiv']")).click();
  }
//Switching back to Parent Window
  driver.switchTo().window(Parent_Window);
//Performing some actions on Parent Window
  driver.findElement(By.className("btn_style")).click();
```

Alerts

- Present in the **org.openqa.selenium.Alert** package
- Syntax:

```
Alert simpleAlert = driver.switchTo().alert();      //switch from main window to an alert
String alertText = simpleAlert.getText();           //To get the text present on alert
System.out.println("Alert text is " + alertText);
//Simple alert
simpleAlert.accept();                               //To click on ‘Ok’/’Yes’ on Alert

                    OR
//Confirmation Alert
simpleAlert. dismiss();                         //To click on ‘Cancel’/’No’ on Alert

                    OR
//Prompt Alerts
simpleAlert. sendKeys("Accepting the alert"); //Send some text to the alert
```

Why synchronization is important

- “Mechanism which involves more than one components to work parallel with each other”
- Every time user performs an operation on the browser, one of the following happens:
  - The request goes all the way to server and entire DOM is refreshed when response comes back
  - The request hits the server and only partial DOM gets refreshed (Ajax requests or asynchronous JavaScript calls)
  - The request is processed on the client side itself by JavaScript functions
- So if we think about the overall workflow, there is a need of certain synchronization that happens between the client(aka. browser) and the server (the url)

Using Explicit & Implicit Wait

- Implicit Wait

- Element Synchronization
  - Default element existence timeout can be set
  - Below statement will set the default object synchronization timeout as 20
  - Means that selenium script will wait for maximum 20 seconds for element to exist
  - If Web element does not exist within 20 seconds, it will throw an exception

- driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

Using Explicit & Implicit Wait

- Explicit Wait
  - Specific condition synchronization
- Instruct selenium to wait until element is in expected condition
  - Syntax:

```
WebDriverWait w = new WebDriverWait(driver,20);
w.ignoring(NoSuchElementException.class);
WebElement P = null;
//below statement will wait until element becomes visible
P=w.until(ExpectedConditions.visibilityOfElementLocated(By.id("x")));
//below statement will wait until element becomes clickable.
P= w.until(ExpectedConditions.elementToBeClickable(By.id("ss")));
```

JavaScript Executor

- An interface which provides mechanism to execute Javascript through selenium driver
- Provides “executescript” & "executeAsyncScript" methods, to run JavaScript in the context of the currently selected frame or window
- Used to enhance the capabilities of the existing scripts by performing Javascript injection into our application under test
- Package
  import org.openqa.selenium.JavascriptExecutor;

  Syntax

```
    JavascriptExecutor js = (JavascriptExecutor) driver;
  js.executeScript(Script,Arguments);
```

script - The JavaScript to execute
Arguments - The arguments to the script.(Optional)

JavaScript Executor(Scenarios)

- How to generate Alert Pop window in selenium?
- Code:

  JavascriptExecutor js = (JavascriptExecutor)driver

  Js.executeScript("alert('hello world');");

- How to click a button in Selenium WebDriver using JavaScript? Code: JavascriptExecutor js = (JavascriptExecutor)driver;

js.executeScript("arguments[0].click();", element);

- How to refresh browser window using Javascript ?
- Code:

JavascriptExecutor js = (JavascriptExecutor)driver;

driver.executeScript("history.go(0)");

JavaScript Executor(Scenarios)

- How to get innertext of the entire webpage in Selenium?
  Code:
  JavascriptExecutor js = (JavascriptExecutor)driver;

string sText = js.executeScript("return document.documentElement.innerText;").toString();

- How to get the Title of our webpage ?
  Code:
  JavascriptExecutor js = (JavascriptExecutor)driver;

string sText = js.executeScript("return document.title;").toString();

JavaScript Executor(Scenarios)

- How to perform Scroll on application using Selenium?
  Code:
  JavascriptExecutor js = (JavascriptExecutor)driver; //Vertical scroll - down by 50 pixels js.executeScript("window.scrollBy(0,50)");

- Note: for scrolling till the bottom of the page we can use the code:

  js.executeScript("window.scrollBy(0,document.body.scrollHeight)");

JavaScript Executor(Scenarios)

- How to click on a Sub Menu which is only visible on mouse hover on Menu?
  Code:
  JavascriptExecutor js = (JavascriptExecutor)driver;

  //Hover on Automation Menu on the Menu Bar

  js.executeScript("$('ul.menus.menu-secondary.sf-js-enabled.sub-menu li').hover()");

- How to navigate to different page using Javascript?
  Code:
  JavascriptExecutor js = (JavascriptExecutor)driver; //Navigate to new Page js.executeScript("window.location = 'https://www.facebook.com/uftHelp'");

Summary

- In this lesson, you have learnt
  - Multiple windows are handled by switching the focus from one window to another.
  - By is a collection of factory functions for creating webdriver.Locator instances.
  - Alert contains methods for dismissing, accepting, inputting, and getting text from alert prompts.
  - Explicit synchronization points are inserted in the script using WebDriverWait class.
  - Each and every time when there is need to match speed of the application and speed of test execution we have to use thread.sleep().
  - The implicit wait will not wait for the entire time that is specified, rather it will only wait, until the entire page is loaded.

Summary

- In this lesson, you have learnt
  - An interface which provides mechanism to execute Javascript through selenium driver
  - Used to click on a Sub Menu which is only visible on mouse hover on Menu
  - Used to to get innertext of the entire webpage in Selenium
  - Used to navigate to different page using Javascript
  - Used to click a button in Selenium WebDriver using JavaScript

Review Question

- Question 1
  - Select which is NOT an Explicit Wait
    - VisibilityOfElementLocated
    - ElementToBeClickable
    - PageLoadTimeout
    - None of the above

- Question 2: True/False
  - The syntax is correct:
  - Syntax : driver.findElement(By. PartialLinkText("link text"));

- Question 3: Fill in the Blanks
  - findElements is used to find ______ element on webpage

Review Question

- Question 4: True/False
  - The syntax is correct:

  Syntax :

  JavascriptExecutor js = (JavascriptExecutor)driver;

- Question 5: Fill in the Blanks
  - An interface which provides mechanism to execute ____________ through selenium driver