SELENIUM CONCEPTS

**1. What are limitations in selenium webdriver**

* It does not support and non web-based applications, it only supports web based applications.
* Its an open source tool so in case of any technical issues you need to rely on the selenium community forums to get your issue resolved.
* You need to know at least one of the supported language very well in order to automate your application successfully.
* No inbuilt reporting capability so you need plugins like JUnit and TestNG for generating test reports.
* Captcha and Bar code cannot be tested using selenium.

**2**. **Installing/configure selenium**

* At first, go to <http://www.seleniumhq.org> and then click on downloads -> download java version (3.3.0 latest ) under  **Selenium Client & WebDriver Language Bindings** and extract the files.
* Adding jar files to Selenium:

Go to eclipse and select project then right click on it and click on Properties and click on Java Build Path - > In that click on Libraries - > click on Add External JARs -> Then go to the specific location where the selenium jars have downloaded - > select “**client-combined-3.3.0-nodeps”** and open it -> Now, click on Apply and then ok -> In the same way import the remaining jar files which are in lib folder. Finally, It got installed into eclipse.

**3. What are different ways of locating elements in selenium**

Attribute based locators - They rely on attributes of the page

1. ID
2. Name
3. ClassName
4. Tag Name
5. Link Text and Partial LinkText

Structure based locators - they rely on structure of the page for finding the elements

1. CSS Selector  
    - Tag and ID  
    - Tag and class  
    - Tag and attribute  
    - Tag, class, and attribute  
    - Inner text
2. DOM (Document Object Model)  
    - getElementById  
    - getElementsByName  
    - dom:name - dom: index
3. XPath

**4. Which is fastest way to identify elements in web page?**

The safest and fastest way to identify elements in web page is through **“id”** because it is unique.

**5. What is absolute path and relative path in xpath**

Absolute path:

* Absolute XPath starts with the root node or a forward slash (/).
* The advantage of using absolute is, it identifies the element very fast.
* Disadvantage here is, if anything goes wrong or some other tag added in between, then this path will no longer works.

Relative path:

* A relative xpath is one where the path starts from the node of your choice - it doesn't need to start from the root node. It starts with Double forward slash(//).
* Advantage of using relative xpath is, you don't need to mention the long xpath, you can start from the middle or in between.
* Disadvantage here is, it will take more time in identifying the element as we specify the partial path not (exact path). If there are multiple elements for the same path, it will select the first element that is identified.

**6. Write code on how to use xpath functions**

**Code:**

**package** PracticalExamples;

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

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

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

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

**public** **class** XpathFunctions {

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

System.*setProperty*("webdriver.gecko.driver", "C:\\Users\\sowmy\\geckodriver.exe");

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

driver.get("https://www.facebook.com/");// pass the url to the driver

driver.manage().window().maximize();// maximizes the window

System.***out***.println(driver.getTitle()); // to know the title of the page

System.***out***.println(driver.getPageSource()); // to know the page source

System.***out***.println(driver.getCurrentUrl()); // to verify with the url

driver.findElement(By.*id*("u\_0\_1")).sendKeys("Sruthi");

driver.findElement(By.*id*("u\_0\_3")).sendKeys("Reddy");

driver.findElement(By.*id*("u\_0\_6")).sendKeys("sowmya2160@gmail.com");

driver.findElement(By.*id*("u\_0\_9")).sendKeys("sowmya2160@gmail.com");

driver.findElement(By.*id*("u\_0\_d")).sendKeys("Wrongpassword@14");

Select sel1 = **new** Select(driver.findElement(By.*xpath*(".//\*[@id='month']")));

sel1.selectByVisibleText("Feb");

Select sel2 = **new** Select(driver.findElement(By.*xpath*(".//\*[@id='day']")));

sel2.selectByValue("28");

Select sel3 = **new** Select(driver.findElement(By.*xpath*(".//\*[@id='year']")));

sel3.selectByIndex(24);

driver.findElement(By.*id*("u\_0\_g")).click();

driver.findElement(By.*linkText*("Create Account")).click();

Select drp = **new** Select(driver.findElement(By.*xpath*(".//\*[@id='userNavigationLabel']]")));

drp.selectByVisibleText("Log Out");

driver.close();

}

}

**7. Different types of waits or synchronization in selenium webdriver**  
 **Implicit wait:**

Implicit waits are used to provide a default waiting time (say 30 seconds) between each consecutive test step/command across the entire test script. Thus, subsequent test step would only execute when the 30 seconds have elapsed after executing the previous test step/command.

**Explicit wait:**

Explicit waits are used to halt the execution till the time a particular condition is met or the maximum time has elapsed. Unlike Implicit waits, explicit waits are applied for a particular instance only.

**Fluent wait:**

Each FluentWait instance defines the maximum amount of time to wait for a condition, as well as the frequency with which to check the condition. Furthermore, the user may configure the wait to ignore specific types of exceptions whilst waiting, such as NoSuchElementExceptions when searching for an element on the page.

**8. How to handle multiple windows in selenium webdriver**

**Code:**

**package** PracticalExamples;

**import** java.util.Iterator;

**import** java.util.Set;

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

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

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

**public** **class** MultipleWindowHandles {

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

System.*setProperty*("webdriver.gecko.driver", "C:\\Users\\sowmy\\geckodriver.exe");

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

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

driver.findElement(By.*cssSelector*("#terms-link")).click();

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

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

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

String parentid = it.next();

String childid = it.next();

driver.switchTo().window(childid);

}

}

**9. How to launch web page using chrome driver**

**Code:**

**package** PracticalExamples;

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

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

**public** **class** ChromePage {

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

System.*setProperty*("webdriver.chrome.driver", "C:\\Users\\sowmy\\chromedriver.exe");

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

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

}

}

**10. Write code to verify any application login page is working or not**

**package** PracticalExamples;

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

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

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

**public** **class** LoginPage {

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

System.*setProperty*("webdriver.gecko.driver", "C:\\Users\\sowmy\\geckodriver.exe");

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

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

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

driver.navigate().back();

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

driver.findElement(By.*xpath*("//\*[@name='email']")).sendKeys("sowmya2160@gmail.com");

driver.findElement(By.*xpath*(".//input[@type='password']")).sendKeys("Wrongpassword@14");

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

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

driver.close();

}

}

**11. How to select items from dropdown/select box**

**Code:**

**package** PracticalExamples;

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

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

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

**public** **class** DropDownAndSelect {

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

System.*setProperty*("webdriver.gecko.driver", "C:\\Users\\sowmy\\geckodriver.exe");

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

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

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

driver.findElement(By.*xpath*(".//input[@id='trip-type-round-trip']")).click();

driver.findElement(By.*xpath*(".//input[@id='air-city-departure']")).sendKeys("MCI");

driver.findElement(By.*xpath*(".//input[@id='air-city-arrival']")).sendKeys("dallas");

driver.findElement(By.*id*("air-date-departure")).click();

driver.findElement(By.*xpath*(".//\*[@data-date='4/4/2017']")).click();

driver.findElement(By.*cssSelector*("#air-date-return")).click();

driver.findElement(By.*cssSelector*("#calendar-may-9")).click();

driver.findElement(By.*id*("price-type-points")).click();

driver.findElement(By.*name*("adultPassengerCount")).click();

**int** i=0;

**while**(i<4){

driver.findElement(By.*xpath*(".//\*[@id='jb-number-selector-more']")).click();

i++;

}

driver.findElement(By.*cssSelector*("#jb-booking-form-submit-button")).click();

}

}

**12. Difference between assert and verify?**

Assert:

When an “**assert**” command fails, the test execution will be aborted. So when the Assertion fails, all the test steps after that line of code are skipped. The solution to overcoming this issue is to use a try-catch block. We use the Assertion in the try catch block. Mostly, the assert command is used when the end result of the check value should pass to continue to the next step. Simply, if the assert condition is true then the program control will execute the next test step but if the condition is false, the execution will stop and further test step will not be executed.

Verify:

When a “**verify**” command fails, the test will continue executing and logging the failure. Mostly, the Verify command is used to check non-critical things. In such cases where we move forward even though the end result of the check value is failed. Simply, there won’t be any halt in the test execution even though the verify condition is true or false.

**13. Difference between driver.close and driver.quit methods?**

These two methods are for closing the browser session in Selenium WebDriver

**driver.close:**

This command closes the Browser window which is in focus. If there are more than one Browser window opened by the Selenium Automation, then the close( ) command will only close the browser window which is having focus at that time. It won’t close the remaining Browser windows.

**driver.quit:**  
This command is generally used to shut down the WebDrivers instance. Hence it closes all the Browser windows that are opened by the Selenium Automation.

Note- They work similar way when Selenium Automation opens only single Browser window but they differ in their functionality when there are more than one Browser windows opened by the Selenium Automation.

**14. Write code for right click in selenium**

Code:

**package** PracticalExamples;

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

**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.firefox.FirefoxDriver;

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

**public** **class** RightClick {

**public** **static** **void** main(String[] args) { System.*setProperty*("webdriver.chrome.driver","C:\\sowmya\\chromedriver.exe");

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

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

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

driver.manage().timeouts().implicitlyWait(5, TimeUnit.***SECONDS***);

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

WebElement element = driver.findElement(By.*xpath*("//ul[@class='Pos(r) Miw(1000px) Pstart(9px) Lh(1.7) Reader-open\_Op(0) mini-header\_Op(0)']/li[2]"));

action.contextClick(element).sendKeys(Keys.***ARROW\_DOWN***).sendKeys(Keys.***ENTER***);

action.perform();

}

}

**15. How to launch different browsers (IE, safari, firefox)**

**Internet Explorer:**

At first, we need to set the system property by giving key and value.

key is **“webdriver.ie.driver”** and value is an executable file that we have to download and paste the path of it. Below is the code

**System.setProperty(“webdriver.ie.driver”,”exe path”);**

Then, for invoking the browser we have to create an object for the driver class and make class object reference to WebDriver interface (which has methods but doesn’t implement itso it make use of this driver class). So, it is given below

WebDriver d = new InternetExplorerDriver();

**Google Chrome:**

At first, we need to set the system property by giving key and value.

key is **“webdriver.chrome.driver”** and value is an executable file that we have to download and paste the path of it. Below is the code

**System.setProperty(“webdriver.chrome.driver”,”exe path”);**

Then, for invoking the browser we have to create an object for the driver class and make class object reference to WebDriver interface (which has methods but doesn’t implement itso it make use of this driver class). So, it is given below

WebDriver d = new ChromeDriver();

**Firefox:**

At first, we need to set the system property by giving key and value.

key is **“webdriver.gecko.driver”** and value is an executable file that we have to download and paste the path of it. Below is the code

**System.setProperty(“webdriver.gecko.driver”,”exe path”);**

Then, for invoking the browser we have to create an object for the driver class and make class object reference to WebDriver interface (which has methods but doesn’t implement itso it make use of this driver class). So, it is given below

WebDriver d = new FirefoxDriver();

**Safari:**

First, we need to download an extension file - “**safaridriver.safariextz”** and install it which acts like a proxy between safari browser and test cases. Then for invoking the browser we have to create an object for the driver class and make class object reference to WebDriver interface (which has methods but doesn’t implement itso it make use of this driver class). So, it like

WebDriver d = new SafariDriver();