Selenium Interview Questions

What is the selenium version you are using?

**Currently We were using 2.53 version. Before it was 2.48 .**

Why do you use Actions.

**To perform the mouse actions or operations like right click, double click, mouse hover, click and hold etc. (user interactions on API)**

**1.click()-**

**This method is used to click at the current mouse pointer position. It is particularly useful when used with other mouse and keyboard events, generating composite actions.**

**2.clickAndHold()-**

**The clickAndHold() method is used to perform the click method without releasing the mouse button.**

**3.contextClick()-**

**This method is used to perform the right click operation(context-click) at the current mouse position.**

**4.doubleClick()-**

**As the name suggest, this method performs double click operation at a current mouse position.**

**5.dragAndDrop(WebElement fro**

**mElement, WebElement toElement)-**

**This is a utility method to perform the dragAndDrop operation directly wherein, we can pass the source element and the target element as parameter.**

**6.moveByOffset(int xOffset, int yOffset)-**

**This method is used to move the mouse pointer to a particular position based on the x and y offsets passed as parameter.**

**7.moveToElement(WebElement toElement)-**

**This method is used to move the mouse pointer to a web element passed as parameter.**

**8.release()-**

**This method releases the pressed left mouse button at the current mouse pointer position.**

**What is the use of select class.How do you use it.**

**The 'Select' class in Selenium WebDriver is used for selecting and deselecting option in a dropdown. The objects of Select type can be initialized by passing the dropdown webElement as parameter to its constructor.**

### ***Selecting options from dropdown***

***selectByIndex***

***selectByValue***

***selectByVisibleText***

### ***Different utility methods in the Select class***

* ***deselectAll() - To deselect all the selected options.***
* ***deselectByIndex(int index) - To deselect the option based on its index.***
* ***deselectByValue(String valueAttribute) - To deselect the option its 'value' attribute.***
* ***deselectByVisibleText(String text) - To deselect the option based on the text over the option.***
* ***getOptions() - To return list of all the options(List<WebElement>).***
* ***getAllSelectedOptions() - To return the list of all the selected options(List<WebElement>).***
* ***getFirstSelectedOption() - To return the selected option or the first selected option in case of dropdowns allowing multi-select.***
* ***isMultiple() - To return a boolean value, checking if the dropdown allows multiple option select or not.***

|  |  |
| --- | --- |
| **.In an ecommerce website,there is a page called general headers. In it there is a navigational bar and on it are various links are in a drop down..Use hover.**  **on the elements and go from link to sublink?** |  |
|  |  |

**How do you access RadioButtons?**

**using isSelected() or click() api we can access form elements like radio buttons, not only radio buttons but also checkbox,textbox, access form**

**How do u check whether particular element is highlighted or not.**

**you can do by executing javascriptexecutor js with Selenium.**

**What are the limitations of using Selenium WebDriver.**

**Some of the limitation of Selenium are-**

1. **Selenium does not provide desktop application automation support.**
2. **Web Services - REST or SOAP cannot be automated using selenium.**
3. **Selenium webDriver requires programming language requirement for script creation.**
4. **For performing common tasks required in automation like logging, reading-writing to external files we have to rely on external libraries.**

**What are the different locators in WebDriver.**

**By Id**

**By className**

**By tagName**

**By name**

**By linkText**

**By partialLinkText**

**By Xpath**

**By cssSelector**

**Display all drop down option values in a given select element**

**Using Select class**

**What are waits in webdriver?**

**Implicit Waits -**

**An implicit wait when used is set to the WebDriver instance and is applied to all the web elements. In implicit wait the webdriver polls the DOM to check the availability of the webElement and waits till the maximum time specified before throwing NoSuchElementException.**

**WebDriver driver = new FirefoxDriver();  
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);**

**In the above code snippet, the value 20 specified in implicit wait method is the maximum time in seconds till which webDriver will wait before throwing NoSuchElementException while locating a webElement.**

**Explicit Waits -**

**Unlike implicit waits, the explicit waits are applied to each and every webElement. In explicit wait, certain conditions are defined for which the webDriver instance waits before locating webElements or performing actions on them. Some of the most common conditions specified in explicit waits are-**

**elementToBeClickable, presenceOfElementLocated etc.**

**WebDriverWait wait = new WebDriverWait(driver, 15);  
wait.until(ExpectedConditions.presenceOfElementLocated(ElementLocator));**

**Here the webDriver instance will wait until the condition specified is met i.e. the presence Of Element located by the *ElementLocator* with the maximum wait time of 15 seconds after which if the condition is still not met than it will throw exception.**

**Fluent Wait**

**A fluent wait is a type of wait in which we can also specify polling interval(intervals after which driver will try to find the element) along with the maximum timeout value.**

**Wait wait = new FluentWait(driver)  
   
 .withTimeout(20, SECONDS)  
   
 .pollingEvery(5, SECONDS)  
   
 .ignoring(NoSuchElementException.class);  
   
 WebElement textBox = wait.until(new Function<webdriver,webElement>() {  
   
 public WebElement apply(WebDriver driver) {  
   
 return driver.findElement(By.id("textBoxId"));  
   
 }  
}  
);**

**How would u find text color yellow in selenium?**

**String color = element.getCssValue("color");**

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

**close and quit?**

***driver.close()* - Used to close the current browser having focus**

***driver.quit()* - Used to close all the browser instances**

**How do you sort the search results or find max from results**

**Using Select class**

**Difference b/w get vs navigate**

**Both *driver.get("URL")* and *driver.navigate().to("URL")* commands are used to navigate to a URL passed as parameter.**

**Get method will get a page to load or get page source or get text that's all whereas navigate**

**will guide through the history like refresh, back, forward.For example if we want to move**

**forward and do some functionality and back to the home page this can be achieved**

**through navigate() only. driver.get will wait till the whole page gets loaded and driver.navigate**

**will just redirect to that page and will not wait**

**driver.navigate.back();**

**driver.navigate.forward();**

**driver.navigate.refresh();**

**Desired capabilities in Selenium Webdriver**

**Desired capabilities are a set of key-value pairs that are used for storing or configuring browser specific properties like its version, platform etc in the browser instances.**

### **Different types of Desired Capabilities Methods**

**getBrowserName()(“chrome”,windows)**

**setBrowserName()**

**getVersion()**

**SetVersion()**

**getPlatform()**

**setPlatform()**

**some webelement methods- isSelected,isEnabled, getAttribute ,getText**

**isSelected**

**driver.findElement(By.id("India")).isSelected());**

**isEnabled**

**driver.findElement(By locator).isDisplayed();**

**isEnabled**

**driver.findElement(By locator).isEnabled();**

**getAttribute**

**driver.findElement(By.id("elementLocator")).getAttribute("value");**

**getText**

**driver.findElement("elementLocator").getText();**

**How to find xpath ?**

**Using firebug, firepath plugins helps to find the xpath**

**Will you modify xpath or copy it directly?**

**Based on nodes if the element matching on single nodes i.e id, classname,name etc, then will copy directly or if the element matching mulitple nodes then will go and modify for child, parent, ancestor, sibling, preceding, self, contains,starts with etc.**

**If there is some element whose locator changes everyday - then how do we locate it in automation code?**

**Identify using JavaScript functions like “starts-with” or “contains” in our element**

**XPath: //input[contains(@class, 'suggest')].**

**XPath: //button[starts-with(@id, 'Submit-')]**

**In what situation you go for xpath?**

**1. if css selector is complex**

**2.browser not supporting css selectors for eg older version of IE which do not support css selector**

**3. handle dynamic elements we can go for xpath**

**Disadvantages of Selenium:**

**1. Selenium needs very much expertise resources. The resource should also be very well versed in framework architecture.**

**2. Selenium only supports web based application and does not support windows based application.**

**3. It is difficult to test Image based application.**

**4. Selenium need outside support for report generation activity like dependence on TestNG or Jenkins.**

**5. Selenium does not support built in add-ins support.**

**6. Selenium user lacks online support for the problems they face.**

**7. Selenium does not provide any built in IDE for script generation and it need other IDE like Eclipse for writing scripts.**

**8. Selenium Automation Engineers are bit in scarcity these days.**

**9. Selenium script creation time is bit high.**

**10. Selenium does not support file upload facility.**

**11. Selenium partially supports for Dialog boxes.**

**Which is the best way to locate an element?**

**Finding elements by ID is usually going to be the fastest option, because at its root, it eventually calls down to document.getElementById(), which is optimized by many browsers.**

**Finding elements by XPath is useful for finding elements using very complex selectors, and is the most flexible selection strategy, but it has the potential to be very slow, particularly in IE. In IE 6, 7, or 8, finding by XPath can be an order of magnitude slower than doing the same in Firefox. IE provides no native XPath-over-HTML solution, so the project must use a JavaScript XPath implementation, and the JavaScript engine in legacy versions of IE really is that much slower.**

**If you have a need to find an element using a complex selector, I usually recommend using CSS Selectors, if possible. It's not quite as flexible as XPath, but will cover many of the same cases, without exhibiting the extreme performance penalty on IE that XPath can.**

**Thread.sleep()**

**If the application is taking time to refresh the page, then we use Thread. Sleep ( ).it is a standard wait it simply wait to the given time.**

1. **How we can retrieve the dynamically changing**
2. **Ids? When we login Facebook the login label's id**
3. **changes dynamically thus resulting in failure.**

**We have a hierarchy of locators and Facebook Is dynamic in nature,so we are not able to**

**use "id" for identification for after that we have remaining 7 locator's for that :2. xpath ()..**

**3. name..4. css.. 5. link text.. 6. partiallinktext...7.tag name. so u can use any one for**

**identifying it. Most probably u can use "xpath" or "css-locator" and if there r tag then**

**link text or partial-link text. it depend on u . But we never use id's in Ajax application**

**because it’s not possible.**

1. **What is the MOST challenging test problem in my**

**career in Automation?**

**In my career**

* **· Changing XPATHS' between testing server and production server-by keeping generic xpath**
* **· Keep separate property files for production and UAT**
* **· automating flash apps**
* **· Mobile Automation**

1. **How does u handle dynamic elements without using**

**xpath (with example?)**

* **· By using classname or css.**

1. **How to work with dynamic web table?**

**You can get the total number of <tr> tags within a <td> tag by giving the xpath of the**

**<td> element by using this function -**

**List<WebElement> ele = driver.findElements(By.xpath("Xpath of the table"));**

**Now you can use a for each loop to loop through each of the <tr> tags in the above list**

**and then read each value by using getText() method.**