Selenium Interview Preparations Questions

SELENIUM

1. **What is Selenium and what is composed of?**

- Selenium is a suite of tools for automated web testing. It is composed of;
  - Selenium IDE(Integrated Development Environment); a Firefox plugin that works for recording and playing back.
  - Selenium RC(Remote Control) (1.0) ; is a test tool and is used to work on JS to automate the web application. (2004)
  - WebDriver (2.0); is a web automation framework and allows you to execute your tests in different browsers. (2011)
  - Selenium Grid; allows tests to run in parallel across multiple machines.

2. **What are the advantages of Selenium?**
- Selenium is open source and free to use without any licensing cost
- It supports multiple languages like Java, Ruby, Python, C#...
- It supports multi-browser testing
- It has a good amount of resources and helping community
- It supports many operating systems like Windows, Mac, Linux ...
- Interact with the web application

3. **What are the disadvantages of Selenium?**
- Selenium supports only web-based applications, does not support windows-based application
- No built-in reporting tool, it needs third party tools for report generation activity
- Cannot work with graphics, captchas, barcodes, shapes
- It does not support file upload facility.
- Hard to master, requires developer level knowledge
- Hard to write good locators
- Hard to synchronize

4. **What are the limitations of Selenium?**
- We cannot test desktop application
- We cannot test web services
- Ewe have to use external libraries and tools for performing tasks like testing framework (TestNG, JUnit), reading from external files (Apache POI for excel)
- Automating Captcha is not possible using Selenium
- It does not support file upload facility.

5. **What types of testing you automate with Selenium?**
- functional tests (positive/negative, UI)
- smoke tests
- regression tests
- integration tests
- end to end testing
- data driven

6. **What we don't do with selenium?**
- Performance, load, stress testing, manual ad hoc testing, (These tests are done by experts trained in these tools)
- Pure database testing (if we only test the DB itself),
- Unit tests..., look and feel based testing (color, shapes, etc.),
- static testing

7. **What is in the Selenium tool set?**
   - Selenium IDE → implemented as a Chrome and Firefox extension, and allows you to record, edit, and debug tests.
   - Selenium RC → to write automated web application UI tests in any programming language
   - Selenium WebDriver→ execute your tests against different browsers
   - Selenium GRID → run your tests on different machines against different browsers in parallel.

8. **What version of Selenium do you use right now?**
   - JDK (JAVA) - 1.8 → I like it because of → Lambda exp. and, Try catch error handling you may add multiple catches.
   - IntelliJ - 2018.03.04
   - Selenium - 3.141.59
   - TestNG - 6.14.3
   - Cucumber – 4.2.6
   - Maven - 3.6.0
   - GIT - 2.17.2

9. **Implicit Wait vs Explicit Wait?**
   - **Implicit wait** is a wait which waits for a specified time while locating an element before throwing "NoSuchElementException". As by default selenium tries to find elements immediately without any wait. So, it is good to use implicit wait. This wait applied to all elements of the current driver instance.
   - **Explicit wait** is a wait which is applied to a particular webelement until the ExpectedCondition specified is met.
   - Implicit wait is simply; if condition is met before the timeout, it will continue to next step, if condition is not met within timeout throw "No Such Element" exception.
   - Explicit wait sometimes we need to wait for a certain event/condition such as element is visible, clickable, enabled….

```
driver.manage().timeouts().implicitlyWait(5,TimeUnit.SECONDS);
webDriverWait wait = new WebDriverWait (driver, 5);
wait.until (ExpectedConditions.visibilityOf(element);
```

10. **What is fluentWait? (JUST EXTRA INFORMATION)**
    - Let's say you have an element which sometime appears in just 1 second and some time it takes minutes to appear. In that case it is better to use fluent wait, as this will try to find element again and again until it find it or until the final timer runs out. Example is AJAX or JQuery
    - Subtype of explicit wait but you can override the conditions

```
Wait<WebDriver>wait=new
FluentWait<Webdriver>(driver).withTimout(5,timeUnit.seconds).pollingEvery(100,timeunit.
milliseconds).ignoring(NoSuchElementException.class);
```

11. **What are various ways of locating an element in Selenium?**
    - Selenium Locators → Id & name
    - In selenium locator is a means of finding an element in the html :
    - Id, name, className, xpath, css, linkText, partialLinkText, tagName

12. **Why I cannot find element?**
    - Locator changed
    - There is an iframe
    - Waiting time:: page is loading slowly or Element is dynamic:: locator
    - Page is not fully loaded/opened
    - Page changes and that element does not exist anymore

13. **How to highlight an element?**
    - Selenium WebDriver doesn't have highlight action.
    - But we can use JavaScript to do it

```
JavascriptExecutor js = ((JavascriptExecutor) driver);
String bgcolor = element.getCssValue("backgroundColor");
for(inti=0;i< 10;i++){
   changeColor("rgb(0,200,0)",
   element,driver);//1
   changeColor(bgcolor,
   element,driver);//2
}
```

14. **What is Xpath?**
    - Xpath is used to find the location of any element on a webpage using html structure.
    - We could navigate through elements and attributes in an XML document to locate web Elements such as textbox.
    - button. checkbox, Image ext… in web Page

15. **Absolute (/) and Relative (//) Xpath?**
    - Syntax → `//tagname[@attribute=`value`]`
    - Absolute xpath starts with single slash (/), starting from root element and all the way to the element.
    - Relative xpath starts with double slash (//), starting selection matching anywhere in the document.

16. **How do you handle dynamic elements?**
    - Find the static part of the id and write a locator(xpath or css) → And then use Startswith, contains, EndsWith
    - contains( ) → //*[contains(@name=`btn`)]
    - startwith( ) → //label[startwith(@id, `message`)]
    - text( ) → //td[text() = `usedId`]
    - or & and → //input[@type = `submit` AND @name = `login`]

17. **How to test dynamic web page?**
    - There is no one size fits all solution to this problem. We have to understand the application very well
        - Use explicit waits where necessary.
        - Use custom xpaths and css locators
            - Xpath: contains, starts with, ends with, contains text.
            - By finding the element in relation to another stable element using parent, child, sibling relationships

18. **How to test dynamic table?**
    - Use custom xpaths and css locators
        - Xpath: contains, starts with, ends with, contains text.
        - By finding the element in relation to another stable element using parent, child, sibling relationships
    - I have utility methods that work with table. I have method that takes a table webelement and returns all the column names. I have a method that takes a table, number and returns all the data in that row.

19. **How can we move to parent element using xpath?**
    - Using (..) expression in xpath, we can move to parent element

20. **Difference between close() and quit() command?**
    - `driver.close()` → used to close the current browser
    - `driver.quit()` → used to close all the browser instances

21. **How can we move to nth child element using xpath?**
    - **There are two ways:**
        - using square brackets with index position
          For ex: div[2] will find the second div element
        - using position ( ) method
          For ex: div[position()=2] will find the second div element

22. **Difference between xpath and css selector?**
    - with xpath, we can search elements backward or forward…
      while css works only in forward direction
    - Xpath can work with text, css cannot work
    - Xpath has more combination and can search by index
      css cannot search by index, but css is working faster than xpath.

23. **What is framework?**
    - In test automation, framework is the blueprint of test automation.
    - It includes your folder structures, where to save you function library, test results, test data, resources.
    - It is essential because when you are working on a automation project everyone will have a guideline to follow and our script will be easier to maintain.

24. **Talking about HTML reporting during the interview?**
    - I use multiple methods of reporting in my framework, driver script writes pass/fail to the test cases excel sheet,
    - Reporter utility object writes to UFT report, also I have developed a custom HTML reporting engine.
    - It sends HTML code to the Notepad and creates a nice HTML report document that nontechnical people can easily understand and use.

25. **How to maximize a web page?**

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

26. **In some cases, maximize() will not work > so what will be the way around?**
    - Actions or change version.

```
ChromeOptions options = new ChromeOptions();
options.addArguments("startmaximized");
```

27. **What is the key class in Selenium?**
    - Gives us option for pressing keys from keyboard
    - Key.ENTER
    - MUST BE PASSED TO SendKeys() method
    - Ex; .sendkeys(“charger” + keys.ENTER)

28. **What if there is a dynamic popup that comes up randomly**
    - Use try/catch with alert

29. **What is Thread.sleep()?**
    - Slows down selenium to catch up
    - Throws exception so must handle it or throw it

30. What is Selenium Framework?

- It is a code structure that helps to make code maintenance easy, code readability and code reuse.
- There are mainly 3 type of frameworks created by Selenium WebDriver to automate test cases:

Data Driven Framework

- It is one of the most popular automation frameworks in the market
- All of our test data is generated from some external files;
    - excel
    - or scenario outline in feature file
    - or TestNG Data Provider
- Selenium WebDriver is a great tool to automate web-based applications. But it does not support read and write operations on excel files. Therefore, we use third party APIs like **Apache POI**

Keyword Driven Framework

- Keyword driven testing is a scripting technique that uses data files to contain the keywords related to the application being tested.
- Keywords are written in some external files like excel file and Java code will call this file and execute test cases.

HybridDriven Framework

- A combination of the DDF and KDF is commonly said to be HDF.
- Both the test data and test action are kept in external files.

31. How did you use overloaded Methods in Selenium?

- When asserting if two values are equal, I use → `Assert.assertEquals(actual, Expected)` from TestNG
- You can put in the parameters String, Objects, int, boolean values

32. Why we get NoSuchElementException?

- Check if locator is correct
- Check if timing is correct
- Check if element is hidden inside an iframe

33. How you handle js alerts?

- If the alert on the browser comes from JavaScript, we use the Alert class.

```
Alert alert = driver.switchTo.alert();
alert.accept();
alert.dismiss();
alert.sendKeys();
alert.getText()
```

34. How to handle multiple frames?

- If there are 4 frames, you have to go through each from consecutively to reach certain frame. Can't jump to the 3rd frame from 1st frame.

35. What is the difference between driver.get() and driver.navigateto() ?

- driver.get() → To open an URL and it will wait till the whole page gets loaded
- driver.navigateto() → To navigate to an URL and it will not wait till the whole page get loaded

36. How to handle frames in Selenium?

- Frames used to embed a html page into another
- Steps
  - Locate the iframe
  - Switch to another iframe with driver.switchTo().frame();
    .frame() → takes string, Integer, webElement, name or id directly as parameter

```
driver.switchTo().frame(webElement);
driver.switchTo().frame();
```

- Now you are in the 2nd frame, if you want to find an element outside of the 2nd frame (that you're currently on) throws NosuchElementException
- If you need to switch back to previous frame
  - driver.switchTo().parentFrame() → Goes one level up
  - driver.switchTo().defaultcontent() → Goes to the very top
- Can switch using count
  - driver.switchTo(0) → Counts anything that is not the default frame

These methods might give you different results based on what browser you are using

37. How you handle browser pop ups?

- **void dismiss()** → clicks on the "Cancel" button as soon as the pop-up window appears.
- **void accept()** → clicks on the "Ok" button as soon as the pop-up window appears.
- **String getText()** → returns the text displayed on the alert box.
- **void sendKeys(String stringToSend)** → enters the specified string pattern into the alert box.

38. How you handle windows/ OS pop ups?

- Selenium doesn't support windows-based apps, it is an automation testing tool that supports only web application testing.
- We could handle windows-based popups in Selenium using some third-party tools such as AutoIT, Robot class
- **driver.getWindowHandle();** This will handle the current window that uniquely identifies it within this driver instance.
- **driver.getWindowHandles();** To handle all opened windows

39. How to handle Headless browser

- Headless browser: browser that does not open, it runs as a background service / program.
- Example is htmlunitdriver from selenium
  - WebDriver = new htmlunitdriver()
  - Not very stable
- Phantonjsbrowser
  - More stable
  - browser = new phantomjsbrowser()

40. findElement vs findElements?

- FindElement > this method returns first WebElement !
  - gives Exception if the element not found
- FindElements > returns List <WebElement>;
  - does not give Exception if the element not found as a result list has null values

41. **How to handle multiple windows/tabs?**
    - Selenium stays on one window
    - If you open a window and then 5 tabs popped open, selenium is focused on the first window
    - If you are on a new window and you tell selenium to print an element on the default window, it will still work even that user's focus is on the new window
    - Must switch to new window
        - Use windowHandle()
        - Driver.getWindowHandle()
            - Everytime Selenium opens a browser, it's going to give an index ID for the page called Handles
            - Returns the handle/id of current page (as a string)
        - driver.switchTo().window(string handle)
        - driver.getWindowHandles() for multiple windows
            - Returns a Set of window handles
        - Switch using titles

```
for(string handle: driver.getWindowHandles()){
   driver.switchTo().Window(handle)
   if(driver.getTitle().equals(targetTitle){
   break;
   }
}
```

42. **How to find all links in the page?**

```
List<WebElement> list = driver.findElements(By.tagName("a"));
```

43. **Difference between isDisplayed(), isEnabled(). And is Selected() method in selenium WebDriver?**
    - isDispalyed() → verify the presence of a web element within the web page. If found → true, If not found → false
    - isDisplayed() → check for the presence of all kinds of web elements available
    - isEnabled() → verify if the web element is enabled or disabled within the web page.
    - isEnabled() → is primarily used with buttons
    - isSelected() → verifies if the web element is selected or not
    - isSelected() → used with radio buttons, dropdowns and checkboxes.

44. **How to Drag And Drop ?**

```
Actions action = new Actions(driver);
action.clickAndHold(driver.findElement(By.id("item")))
.moveToElement(driver.findElement(By.id("destination")))
.release().build()
.perform();
```

45. **For Scroll down:**

```
WebDriver driver = new ChromeDriver(); JavascriptExecutor jse = (JavascriptExecutor)driver;
jse.executeScript("window.scrollBy(0,250)", "");
```

    - OR, we can do as follows:

```
jse.executeScript("scroll(0, 250);");
```

46. **For Scroll up:**

```
jse.executeScript("window.scrollBy(0,-250)", ""); OR, jse.executeScript("scroll(0,-250);");
```

47. How to handle cookies?

- In Selenium Webdriver, we can query and interact with cookies with below built-in method:

```
driver.manage().getCookies();   // Return The List of all Cookies
driver.manage().getCookieNamed(arg0); //Return specific cookie according to name
driver.manage().addCookie(arg0); //Create and add the cookie
driver.manage().deleteCookie(arg0); // Delete specific cookie
driver.manage().deleteCookieNamed(arg0); // Delete specific cookie according Name
driver.manage().deleteAllCookies(); // Delete all cookies
```

48. Why do we need to handle cookies?

- Each cookie is associated with a name, value, domain, path, expiry, and the status of whether it is secure or not. In order to validate a client, a server parses all of these values in a cookie.
- When Testing a web application using selenium web driver, we may need to create, update or delete a cookie.
- For example, when automating Online Shopping Application, we many need to automate test scenarios like place order, View Cart, Payment Information, order confirmation, etc.
- If cookies are not stored, we will need to perform login action every time before we execute above listed test scenarios. This will increase your coding effort and execution time.
- The solution is to store cookies in a File. Later, retrieve the values of cookie from this file and add to it your current browser session. As a result, you can skip the login steps in every Test Case because your driver session has this information in it.
- The application server now treats your browser session as authenticated and directly takes you to your requested URL.

49. Store cookies cookie information

```
public class cookieRead extends BasePage{
    public static void main(String[] args){
        driver.get("http://demo.guru99.com/test/cookie/selenium_aut.php");

        // Input Email id and Password If you are already Register
        driver.findElement(By.name("username")).sendKeys("abc123");
        driver.findElement(By.name("password")).sendKeys("123xyz");
        driver.findElement(By.name("submit")).click();

        File file = new File("Cookies.data");    // create file to store cookies

        try {
            file.delete(); // Delete old file if exists
            file.createNewFile();
            FileWriter fileWrite = new FileWriter(file);
            BufferedWriter Bwrite = new BufferedWriter(fileWrite);

            // loop for getting the cookie information
            for(Cookie ck : driver.manage().getCookies()){
               Bwrite.write((ck.getName() +";"+ ck.getValue() +";"+ ck.getDomain() +";"+ ck.getPath()
                                          + ";" + ck.getExpiry() +";"+ ck.isSecure()));

               Bwrite.newLine();
            }

            Bwrite.flush(); Bwrite.close(); fileWrite.close();
        } catch(Exception ex) {
            ex.printStackTrace();
        }
    }
}
```

50. Use stored cookies to login information

```
public static void main(String[] args){
    try{
        File file = new File("Cookies.data");
        FileReader fileReader = new FileReader(file);
        BufferedReader Buffreader = new BufferedReader(fileReader);

        String strline;
        while((strline=Buffreader.readLine())!=null){
            StringTokenizer token = new StringTokenizer(strline,";");
            while(token.hasMoreTokens()){
                 String name = token.nextToken();
                 String value = token.nextToken();
                 String domain = token.nextToken();
                 String path = token.nextToken();
                 Date expiry = null;

                 String val;
                 if(!(val=token.nextToken()).equals("null")) {
                     expiry = new Date(val);
                 }
                 Boolean isSecure = new Boolean(token.nextToken()).booleanValue();
                 Cookie ck = new Cookie(name,value,domain,path,expiry,isSecure);
                 System.out.println(ck);
                 driver.manage().addCookie(ck); // This will add the stored cookie to current session
            }
        }
    }catch(Exception ex){
        ex.printStackTrace();
    }
    driver.get("http://demo.guru99.com/test/cookie/selenium_aut.php");

}
```

OUTPUT: You are taken directly to the login success screen without entering the input user id and password
NOTE: Use hard refresh in case you see the login page after executing the above script.

51. Do you use JavaScriptExecutor?

- This helps me write my own JavaScript. JS has way more control than selenium.
- we can send JS commands to the browser with using this class
  `JavaScriptExecutor jsExecutor=(JavaScriptExecutor)driver;`
  - executeScript(); performs the command
  - Inside the parameter is where you put JS code
- jsExecutor.executeScript("alert('WARNING: This is a useless message');" → This code will bring up a JS popup
- You can also put 2 parameters is .executeScript("js code",element);
  - Used for scrolling (selenium is not good with scrolling, you can say a challenge is when I was working on terms and condition page, where you have to read the page before clicking on continue.
  - When I tried using selenium and actions class it didn't work, so i used javaexecutor) and clicking an element;

52. How to check if element is present/visible/enable/ and to check text present?

- To check Element Present:

```
if(driver.findElements(By.xpath("value")).size() != 0){
   System.out.println("Element is Present");
}else{
   System.out.println("Element is Absent");}
```

- or

```
if(driver.findElement(By.xpath("value"))!= null){
   System.out.println("Element is Present");
}else{
   System.out.println("Element is Absent"); }
```

- To check Visible:

```
if(driver.findElement(By.cssSelector("a > font")).isDisplayed()){
   System.out.println("Element is Visible");
}else{
   System.out.println("Element is InVisible"); }
```

- To check Enable:

```
if(driver.findElement(By.cssSelector("a > font")).isEnabled()){
   System.out.println("Element is Enable");
}else{
   System.out.println("Element is Disabled"); }
```

- To check text present

```
if(driver.getPageSource().contains("Text to check")){
   System.out.println("Text is present");
}else{
   System.out.println("Text is absent"); }
```

53. How check the multiple selected values in dropdown?

- Select carsList = new Select(el)
- carList.getSelectedOptions(): //returns the the selected options a list (List<webelement>)
- for each : carList.getSelectedOptions()

54. How to use actions class?

- Actions class lets us do advanced mouse and keyboard operations:
- Control the mouse
- Class that provides methods for advanced user interactions
  - Hovering
  - Double click
  - Right click
  - Scroll
  - Drag and drop
  - mix/match operators

- Actions action=new Actions(driver)
- Action methods
  - click()
  - hold()
  - moveToElement(element)
  - perform()
  - keydown()build()
  - dragAndDrop(source,target).perform()
  - sendKeys() different from the one we usually use

    - Let's you do the sendkeys operation on different elements
    - Regular sendkeys that comes from webelement will throw an exception on something that is not input text.
    - The long way is;
      `actions.moveToElement(source).clickAndHold().moveToElement(target).release().perform();`
- Actions won't work unless perform() is used
- If you are chaining methods, you must use build() before perform()

55. What is the syntax for double click action ?

- To perform any actions against web element using actions class, we need to locate the element first:

```
WebElement el = driver.findELement
Actions actions = new Actions (driver).perform actions.doubleClick(el).perform()
actions.moveTo(el).perform actions.doubleClick.perform
actions.moveTo(el).doubleClick().build.perform()
```

56. File download and upload

- **Download**
    - Selenium itself cannot verify file downloads, can click on download link but can't go outside the browser and open the downloaded file
    - Other tools need to be used for that Robot and AutoIT
- **Upload**
    - Selenium handles the upload, but does it differently compared to actual user
    - Steps
        - Find the element that triggers the upload window
        - Find the path of the file you want to upload
            - Store into a String
                - Ex: String → file="C:\\Users\\Andy\\Desktop\\folder1\\file.key";
                - Then driver.findElement(upload button).sendkeys(file);

57. How check the selected value in dropdown?

```
Select carsList = new Select(el)
carList.getFirstSelectedOption()
assertequals("some text",carList.getFirstSelectedOption().getText() )
```

58. How to work with dropdown without the select tag?

- If the dropdown list has no select tag, we cannot use the select class
- Treat the dropdown list and its options as separate elements, locate every element separately
- To select an option:
    - 1. Find and click on the list
    - 2. Find and click on the option

59. What if there's no select tag?

- You have to select the label for the dropdown separately as a webelement.
- Then manually use click method

60. Sometimes sendKeys does not work

- Robot or AutoIT
- library==jar file==dependency

61. What is the syntax for switching frame ?

- Frame is a html document inside another html document.
- Web driver handles one page/html document at a time. To control another frame, we always need to switch
- Driver.switchTo.frame(webelement) → find the iframe and pass as a param
- Driver.switchTo.frame(string) → find the id or name of the iframe and pass as a param
- Driver.switchTo.frame(int) → find the index and pass as a param

62. What is the syntax for switching windows ?

- To handle separate tabs/windows we have to switch to that tab
- Web driver handles one page/html document at a time.
- To control another tab, we always need to switch
- To be able switch we need to get the window handle first using

```
getWindowHandles() method driver.switchTo.window(String) // → window handle
//for each loop : driverGetWIndowHandles:
Driver.switchTo.window("handle")
Ifdriver.getTitle==expectedtitle;
Break;
```

63. What is the syntax for uploading a file?

```
Public void fileUpload(Stirng path){
   WebELement upload = driver.findELement; Upload.sendKeys(path)
}
```

- We need to locate the upload button in html.
- The element will have tag input.
- Then we do sendKeys by passing the path to file which we want to upload

64. Sometimes sendKeys/path does not work

- Building a dynamic path for a file inside our project Path to the project location:

```
String projectDir= System.getProperty("user.dir") // project directory
String file= "src/test/resources/test_data/myfile.txt";
Element.sendKeys(projectDir+file);
```

65. How to input text in the text box without calling the sendKeys()?

```
   //Use                                                                    javascriptExecutor
JavascriptExecutor JS = (JavascriptExecutor)webdriver;
   //To                                    enter                                    username
JS.executeScript("document.getElementById('User').value= 'www.google.com'");
   //To enter password
JS.executeScript("document.getElementById('pass').value=' tester'");
```

66. How to press ENTER key on text box in Selenium WebDriver?

- To press Enter key using Selenium WebDriver,
- We need to use Selenium Enum keys with its constant Enter
- `Driver.findElement(By.xpath("xpath")).sendKeys(Keys.ENTER);`

67. Have you done any cross-browser testing? cross browser testing

- Always mention that you have a control file for keywords like browser type, main url, username, password, environment.

68. How you resolve certification issue?

- CHROME, IE → DesiredCapabilities capability = DesiredCapabilities.chrome();
- on Jenkins we need to insert → `.relaxedHTTPSValidation`

```
Response response=RestAssured.given()
                    .contentType(ContentType.JSON)
                            .relaxedHTTPSValidation()
                            .get("https://api.got.show/api/continents");
System.out.println(response.asString());
```

69. How would you verify the position of the Web Element on the page?

- `element.getLocation();`
- WebElement class has a get Location method with returns the top left corner of the element

70. Page Factory class?

- Page Factory class comes with Selenium.
- And it is used whenever we create page object classes.
- Its purpose is to initialize webElements that were defined in the class.

71. What tools are you using to test UX and Restful webServices?

- UX → User Experience. First ensure UX is acceptable manually.
- After that since it is UI testing, I use Selenium WebDriver to automate it.
- RESTFul API Automation > RestAssured Library, PostMan for manual tests

72. How To resize browser Window Using Selenium WebDriver?

- To resize the browser window to particular dimensions, we use 'Dimension' class to resize the browser window.
- //Create object of Dimensions class
  Dimension d = new Dimension(480,620);
- //Resize the current window to the give n dimension
  driver.manage().window().setSize(d);

73. What exceptions do you know in Selenium?

- I often have **NoSuchElementException**
- **StaleElementException**
  - The element has been deleted entirely.
  - The element is no longer attached to the DOM.
  - How we handle StaleElementException;
    - Element is not attached to DOM → 'try catch block' within 'for loop'
      Or
    - 1. Refresh the page and try again for the same element.
    - 2. Wait for the element till it gets available
- **TimeOutException**

74. ASSERT(hard assert) VS VERIFY(soft assert)

- Hard assert throws an AssertException immediately when an assert statement fails, and test suite continues with next @Test. If Assert steps fails, execution of test stops at that point! and will go to next test if present!

- (Example: just simple Assert.assertTrue(boolean);)
- Soft assert collects errors during @Test Soft Assert does not throw an exception when an assert fails and would continue with the next step after the assert statement. If Verify steps fails, it will report a fail but will continue execution!
  - Example: SoftAssert soft=new SoftAssert(); //for soft create object
  - soft.assertTrue(boolean);
  - soft.assertAll(); //put at the end it will report what is failing!

75. What the verification point available in Selenium ?

- In selenium IDE, We use Selenium Verify and Assert Commands as Verification points
- In Selenium WebDriver, There is no built-in features for verification points, it totally depends on our coding style. Some of the Verification points are
  - to check for page title
  - to check for certain text
  - to check for certain element(text box, button, drop down, etc.)

76. Verify text exists?

- VerifyTextPresent → returns TRUE if the specified text string was FOUND somewhere in the page; FALSE if otherwise.
- VerifyTextNotPresent → returns TRUE if the specified text string was NOT FOUND anywhere in the page; FALSE if it was found.

77. How do you find a text in a webpage?

- //tagname[contains(text(),'text')] contains certain test
- //tagname[.='text'] contains exact text sometimes doesn't work Selenium

78. How to get all the preceding siblings of Apple?

- Xpath: "//ul/li[contains(text(),'Apple Mobiles')]/precedingsibling::li"
- This will give "Samsung Mobiles"

79. How to get all the following siblings of Apple?

- Xpath: "//ul/li[contains(text(),'Apple Mobiles')]/followingsibling::li"
- This will give all the preceding siblings (Nokia Mobiles, HTC Mobiles, Sony Mobiles, Micromax mobiles)

80. How to handle Web Tables/grid?

- Table tag used for table data is arranged in a grid format
  - th tag for column name Example –

```
<tr>
   <th>FirstName</th> column names on the very top row
   <th>Lastname</th>
   <th>Age</th>
</tr>
```

  - </tr> tr tag used to indicate a row, applies to whole column td tag to indicate a column in a row Example

```
<tr>
   <td>Danny</td> actual_data_on_the_very_first_row
   <td>Smith</td>
   <td>29</td>
</tr>
```

- Some tables have tbody Used to indicate the data of the table, usually does not include column names (th)

81. How to use Excel?

```
FileInputStream ExcelFile = new FileInputStream(path);
excelWBook = new XSSFWorkbook(ExcelFile);
excelWSheet =  excelWBook.getSheet(sheetName);
cell = excelWSheet.getRow(rowNum).getCell(colNum);
```

82. How do you like Selenium version 3? Is Selenium 3 drastically different from Selenium 2? (*JUST EXTRA INFORMATION*)

- Selenium 3 has bug fixes from selenium 2 also it is more mobile automation focused.
- We aim for Selenium 3 to be "a tool for user-focused automation of mobile and web apps".
- Here is the summary of the change.
    - For WebDriver users, it's more of bug fixes and drop-in replacement for 2.
    - Selenium Grid bug fixes are done as well.
    - Selenium project will not actively support only the WebDriver API.
    - By a quirk of timing, Mozilla have made changes to Firefox that mean that from Firefox 48 you must use their geckodriver to use that browser, regardless of whether you're using Selenium 2 or
    - As we know Selenium 3.0 is the latest version of Selenium Jar