Chapters:

Puzzled out Selenium as Automation tool

Aced Selenium in Real Time

OOPs High on Java / OOPs with JAVA

Being an Architect

Communicating with APIS

Insider of DB Testing - SQL

Tested as Tester

Chapter 1

Puzzled out Selenium as Automation tool

1) What is Selenium?

Selenium is a open source automation testing tool used to validate web applications across different browsers and platforms. It supports multiple programming languages like Java, C#, Python etc. to create Selenium Test Scripts.

2) What are advantages of Selenium?

* Selenium is an Open-Source Tool, so it is free to use for automating our web applications
* It supports multiple programming languages and capable to operate on almost every operating system and range of browsers.
* It is independent of the language that the web Application is using.
* It has robust Element Locators.
* Selenium support integration of Open-source frameworks like TestNg, Junit with Maven and Jenkins.
* Selenium supports Web and Mobile Applications.

3) What are Limitations of Selenium?

* Selenium supports only web-based Applications.
* It has dependency on third party tools for complete benefit.
* Selenium has no official technical support team.
* Selenium has limited Support for Image based Testing, Captcha, and bar code readers.
* Selenium has no Built-in reporting Facility.
* Also, it has challenges with IE browser.

4) What are the different Components of Selenium framework?

* Selenium IDE – It is record and play tool which will record all the user actions and play them back. Its main disadvantage is that it can’t handle the dynamic web elements and difficult to maintain and pass multiple test data
* Selenium RC – Selenium Remote Control came before selenium web driver and it allows us to automate web application UI tests with the help of multiple programming languages like java, c#, python etc. Client libraries communicate with the selenium RC Server passing each selenium command for execution, then RC Server passes the selenium command to the browser using selenium-core javascript commands and browser executes them using its javascript interpreter
* Selenium Web driver - It is a collection of Native APIs that directly interacts with Browser, and they give more control and faster than RC APIs.
* Selenium Grid -It is the tool which can distribute tests across multiple browsers or different machines.

5) What is selense?

It is a set of commands in selenium used for running a test.

* Actions- used for performing interactions and operations with target elements.
* Accessors- used for storing values in a variable.
* Assertions- used as checkpoint.

6) What are the different types of Webdriver APIs supported in selenium?

|  |  |  |
| --- | --- | --- |
| GecoDriver | Firefox driver | Firefox |
| Microsoft Webdriver | Edge, internetExplorerDriver | IE |
| Google Chrome Driver | ChromeDriver | chrome |
| HTML Unit Driver | WebClient | Chrome, FF, IE |
| Open ChromeDriver | ChromeDriver | Opera |
| Safari Driver | SafariDriver | Safari |
| Android Driver | Android Driver | Android Browser |
| IOS Driver | IOS Driver | IOS Browser |
| EventFiring WebDriver | EventFiring WebDriver | All |

7) Explain selenium webdriver

* SearchContext is super most interface in selenium, which is extended by another interface webdriver.
* All abstract methods of search context and webdriver interfaces are implemented in remote webdriver class.
* All browser related classes such as firefoxdriver etc extends remotewebdriver class.
* If we use webdriver driver = new remote webdriver( new url desiredCapabilities Firefox());) we need to mention where selenium server is located and which web browser you want to use.
* For Selenium grid we must use remotewebdriver.

A whiteboard with writing on it

Description automatically generated with medium confidence

8) What is the use of creating reference variable 'driver' of type webdriver?

* If we create a reference variable driver of type webdriver then we could use the same driver variable to work with any browser of our choice such as IEdriver, SafariDriver etc.
* WebElement in selenium represents an HTML element. It basically represents a DOM element in a html element.

9) How to Launch different browsers using selenium webdriver?

* Webdriver is an interface, we need to create object of the required driver class such as firefoxdriver, chromedriver etc.
* example: WebDriver driver= new FirefoxDriver();

10) What are different types of Locators?

* By.id()
* By.name()
* By.xpath()
* By.cssSelector()
* By.tagname()
* By.className()
* By.linkText()
* By.PartialLinkText()

11) Which locator is best to use for identifying the webelement?

* ID is the best to use as id will be unique for any webelement in the DOM and to removes the duplicate element finding issue.
* The order of using locators most likely - ID>NAME>CSS>XPATH

12) Why is CSS selector better than xpath?

* xpath engines are different un each browser , hence makes them inconsistent. IE doesn't have a native engine, therefore selenium injects its own xpath engine for compatibility of its APIs.
* Xpath tend to become complex and hence make hard to read in my opinion.

13) What is the syntax of xpath and CSS?

* driver.findElement(By.xpath("//tag[@attribute='value']"))
* driver.findElement(By.css("tag[attribute='value']"))

14) What are relative and Absolute xpaths?

* Absolute Xpath is the direct way to find element. The disadvantage of absolute xpath is that for any change made in the path of the element then xpath will fail. It starts with root node or a single forward slash(/).
* Relative Xpath is a url that contains a portion of full path and it is alternatively referred as partial path/ non- absolute path.
* A Relative path is used to specify the location of a directory relative to another directory. A relative path starting from element you want to refer to and go from there.
* Absolute Xpath always starts with double forward slash, it directly interacts with current node of element. It is the shortest way to navigate to an element on a webpage.
* Absolute Xpath: /html/body/div/span
* Relative Xpath: //div[@class='value']//h4[1]

15) How to handle dynamic webelements using xpath?

* Using xpath with contains - //[contains(@type,'sub')] , //div[contains(tect(),'value')] OR/AND - //div[@type='submit' OR/AND @name='namevalue'] Starts with - //label[starts-with(@id,'message')] following - //[@type='text']//following::input[1]
* By using Ancestor keyword in the xpath - //[text()='value']//ancestor::div Child - //[@id='value']//child::li
* By using preceding keyword in the xpath - //[@type='value']//preceding::input following-sibling - //[@type='value']//following-sibling::input
* By using parent keyword in the xpath- //[@id='test']//parent::div Self - //[@type='value']//self::input
* By using Descendant keyword in the xpath - //\*[@id='value']//descendant::a

16) What is difference between xpath and CSS selector?

|  |  |
| --- | --- |
| Xpath | CSS |
| Easy to write and remember | Bit Complex |
| Various ways to find xpath | Less than xpath |
| Xpath engine is different in each browser | Same in all browsers |
| Can traverse backward and forward direction | Traverse only forward |
| In IE browser, xpath may not work. | Same for all browsers |
| Traversing the DOM in older browsers like IEs doesn't work | Doesn't support older versions |
| Success rate of finding elements using xpath is high | Low success rate |
| If element is one of its child, it is defined in xpath using '//' (//div//a//i) | In CSS white space is used( div a i) |
| ID is found using @ in xpath (//input[@id='email']) | # is used to find id in CSS(input[#id='email'] |
| Class is defined as [@class='test'] | Class is defined as '.' |

17) What are ImplicitWait and ExplicitWait?

* ImplicitWait tells the webdriver to wait before throwing exception NoSuchElement/elementnotVisible, wait until the stated time.
* Example - driver.Manage().Timeouts().ImplicitWait(10,Timeunit.seconds)
* In ExplicitWait before throwing exception it tells webdriver to wait till the specified condition is met or maximum time elapses.
* It is applied for specified test step in the test script, we must first create an instance for webdriver wait class to use
* Example - webdriverWait wait= new Webdriverwait (driver,timespan.fromseconds(30)); wait.until(expectedconditions.visibilityofallelementslocatedBy(By.Id(elementID));
* Explicit wait will make the webdriver wait for a specific webelement for the specified time where as implicit wait will make the webdriver wait for all webelements for the same specified time.

18) What is exception?

* Exceptions are events due to which Java programs ends abruptly without giving expected output. Java provides a framework where a user can handle exception.
* The process of handling exception is called exception handling. The class hierarchy of exception and error:

A picture containing text, whiteboard

Description automatically generated

Checked Exception - It is handled during compile time and it gives the compilation error if it is not caught and handled during compile time. Ex: filenotfoundexception, IOexception etc.

Unchecked Exception - Incase of unchecked exception, a compiler does not mandate to handle. The compiler ignores during compile time. Ex: ArrayIndexOutofBoundException

19) What is Error?

* When a scenario is fatal and the program can't recover then JVM throws an error. Errors can't be handled by try catch block.
* Even if the user tries to handle the error by using try catch block, it can't recover from error. Example: assertion error, Out of memory error.

20) What is difference between error and exception?

* Exceptions can be handled at the run time where as errors cannot be handled at the run time.

21) How to Handle Exception?

* Try and Catch block: It is generally used to handle exceptions.
* Throws Exception: Throws keyword is used to throw an exception rather than handling it . All checked exceptions can be thrown by methods.
* Finally Block: Finally block executes irrespective of execution of try-catch block and it execute immediately after try\catch block completes.

22) What are different types of exceptions?

* ElementNotVisibleException: when we try to locate a particular element on webpage that is not currently visible even through it is present in DOM. Also if we try to find element using xpath which associates with 2 or more elements.
* StaleElementReferenceException: Occurs when element has been deleted entirely, DOM has been refreshed, Navigation to another page, a frame or window switch , element is no longer attached to DOM. We face this when element we are interacting is destroyed and then recreated again. When this happens the reference of element in DOM becomes stale, hence we are not able to get reference to element. To avoid this we use dynamic xpath.
* WebDriver Exception: Webdriver is acting immediately after closing the browser.(comes when code is unable to initialize webdriver or we try to perform any action on the non-existing driver )
* IllegalStaleException: When we didn't specify the path of driver with system property.
* TimeOut Exception: This exception occurs when a commmand completion takes more than the wait time. If webdriver tries to find an element in webpage before page completely loads, then elementNotVisibleException is thrown. To avoid this exception wait commands are added. However if the componenets don't load even after wait, this exception occurs.
* To avoid this we can add explicit wait using Javascript executor untill page is loaded.
* webdriverwait wait= new webdriverwait(driver,Timespan.fromseconds(30)); wait.until((javascriptexecutor)webdriver).executescript("reurn document.readystate").equals("complete"));
* NosuchsessionExecption: This occurs when the browser is quit usiisng webdriver.quit. This can also happen due to web browser issues like crashes and webdriver can't excute any command using driver instance. To avoid this always choose stable version of browser to run selenium webdriver test cases.
* NosuchelementException: when webdriver is unable to find and locate elements. Happens when the incorrect element locator is used. To avoid this try giving wait command. Comes under not found exception class.
* NoSuchWindowException: It comes under not found exception class, this is thrown when web driver tries to switch to an invalid window. We should use window handles to get the set of active windows and then perform actions on the same to avoid this exception.
* NoSuchFrameException: Comes under Not found exception class. When webdriver is trying to switch to an invalid frame. to avoid this try to give wait command.
* NoalertPresentException: It is thrown when webdriver tries to switch to an alert which is not available. To avoid always use explicit or fluent wait for a particular time in all cases where an alert is expected.
* InvalidSelectorException: Subclass of nosuchelementexception. It occurs when a selector is incorrectly or syntactically invalid, commonly occurs when xpath locator is used. To avoid this, we should check the locator used before the locator is likely incorrect or syntax is wrong.
* ElementNotVisibleException: Subclass of elementnot interactable exception, this is thrown when webdriver tries to perform an action on an invisible web element, which can't be interacted with. To avoid this, we can use wait for element to get completely.
* ElementNotSelectableException: Comes under InvalidelementStateexception class, this indicates that the web element is present in web page but can't be selected. To avoid this we can add a wait command to wait until the element becomes clickable.

23) What are different methods to move back, forward and refresh browser in selenium?

* driver.navigate().forward();
* driver.navigate().back();
* driver.navigate().refresh();
* driver.navigate().to("url");

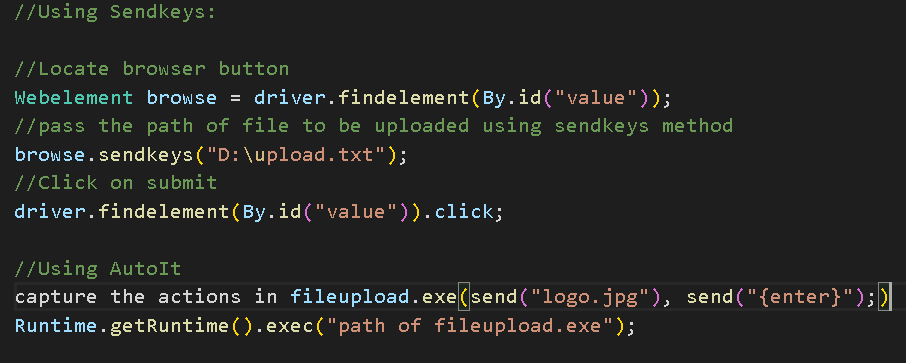
24) What is difference between driver.get() and driver.navigate().to?

* driver.get() refreshes the page also and waits for the page to load. hostory is not maintained in the case of driver.get() because of refresh.
* driver.navigate().to(url) maintains browser history or cookies to navigate back and forward and allows you moving back and forward in browser history. It doesn't wait till the page loads.

25) What is difference between driver.findelement() and driver.findelements()?

* Findelement() returns a single webelement where as findelements() returns a list of webelements.
* When no element is found findelement() throws nosuch element exception where as findelements returns a list of 0 elements.

26) How to handle file upload in selenium?



27) How to handle alert?

Alert interface provides methods to using in selenium for handling JavaScript alerts.

Text

Description automatically generated

28) How to switch to window in selenium?

Graphical user interface, text, application

Description automatically generated with medium confidence

29) What is difference between getwindowhandles() and getwindowhandle()?

* Getwindowhandles() is used to get the address of all open browsers and it returns datatype is set<string>
* Getwindowhandle is used to get address of current browser when the control is and its returntype is string data type.

30) How to select or switch to parent window?

Graphical user interface, text, application

Description automatically generated

31) How to handle frame in webdriver?

* An iframe (inline frame) is used to insert another document within the current HTML document.

Text

Description automatically generated

32) How to verify if the element is present in the webpage using selenium?

* iselementPresent(String locator)
* It takes locator as parameter and returns Boolean value.

33) How to check if the element is present and displayed in the web page using selenium?

* iselementDisplayed(String locator)
* It takes locator as parameter and returns Boolean value.

34) How to verify if the Button is enabled using selenium?

* isEnabled() can be used for verifying if the element is enabled or disabled

35) How to find the element which is active in selenium?

* driver.switchto().activeelement();

36) How to verify if the dropdown value is selected?

* Using isselected();

37) How to use JavaScript executor in selenium?

* It is a interface which provides a mechanism to execute Javascript through the selenium webdriver.
* It provides "executescript" and "executeAsyncScript" methods to run Javascript in context of currently selected frame or window.

Text

Description automatically generated with medium confidence

38) How to handle keyboard and mouse actions?

* We have advanced user interactions API for handling and it has action and actions classes.
* actions class has below methods
* clickandhold() - clicks (without releasing) the current mouse location.
* DragandDrop()- Performs click and hold at the location of the source element, moves
* Source,target- moves to the location of target element then releases the mouse.
* Perform() is used to execute the actions.

Graphical user interface, text, application

Description automatically generated

39) What are keyboard interface in selenium?

* SendKeys(keys to send) : sends a series of keystrokes onto the element.
* KeyDown(the key): sends a key press without release it.
* KeyUp(the key): performs a key release

40) What are Mouse interfaces in selenium?

* Click() - clicks on element
* Doubleclick()- Double clicks on the element
* ContextClick- performs a context click(right click) on the element
* ClickandHold- clicks and hold (release has to be used)
* Dragand DropBy(source,xoffset, yoffset)
* MoveByoffSet(x0ffset,yoffset)
* MovetoElement(toelement)

41) What are the methods available to implement the robot class in selenium?

* KeyPress(): Use this method when you want to press any key
* ex: robot.keyPress(keyEvent.VK\_UP)- will press up key on keyboard
* KeyRelease(): Use this method to release the pressed key on keyboard
* ex: robot.KeyRelease(keyEvent.VK\_CAPSLOACK)- will release the pressed caps lock key on key board.
* MouseMove()- robot.mouseMove(coordinates,get.x(),coordinates.get.y());
* MousePress()- robot.mousepress(Inputevent.button1\_MASK);
* MouseRelease()- robot.mouseRelease(InputEvent.button3\_DOWN\_MASK);

42) What are the methods available in selenium?

* Navigate Command: Navigate().to("url"), get("url"), navigate().back(), navigate().forward(), navigate().refresh()
* Resizing windows: set the size of window, the unit used is pixels.
* dimension d = new dimension(640,640);
* driver.manage().window(0.setsize(d);
* to maximize window - driver.manage().window().maximize();
* Delete Cookies: driver.manage().deleteallcookies();
* closing browser: close() - closes active window
* quit() - closes all windows
* Get methods : getcurrenturl(), getpagesource(), gettitle(), getwindowhandle(), getwindowhandles()
* Searching webelements - findelement(), findelements()
* Mouse Operations - actions act = new Actions(driver);
* act.movetoelement(driver.findelement("").build.perform();

43) How to handle a dropdown in selenium?

Text

Description automatically generated

44) What are verification points available in selenium?

* Verify and assert

45) What is difference between assert and verify?

* Assert- If the condition is true the the program control will execute the next step but if condition is false , execution will stop and further test step will not be executed.
* Verify- There willn't be any halt in test execution even though verify condition is true or false.
* Soft Assert - It collects errors during @Test and doesn't throw any exception when an assert fails and would continue with next step after assert statement.
* Hard assert- Throws assert exception immediately when an assert statement fails and test suite continues with next @Test.

46) How to take screenshot for the failed scenarios?

Text

Description automatically generated

47) How to check all the checkboxes in a page?

Text

Description automatically generated

48) What is Selenium Grid?

* It is the tool which can distribute tests across multiple browsers or different machines.
* It enables parallel execution of test cases, using this we can configure to run thousands of test cases concurrently on separate devices or browsers.

49) How to implement selenium grid?

* modify the driver() to removewebdriver (host , portno , platform , browser)
* Set the desired capabilities
* In the grid console, register hub with node and display available machines.

50) What is difference between driver.quit() and driver.close() commands?

Driver.quit() is used for closing all the open windows where as other one is used to close the active/current window

51) What are the selenium 4 changes?

* From selenium 4 , selenium server works not only in standalone, hub and node mode but also in distributed mode to make it easy to deploy any.
* Also new Selenium server supports OpenTelemetry and exposes a graph QL end point so that tracking down got easier.
* Selenium Webdriver APIs adopt W3C standardization.
* Finding elements in the web page are made easier by relative Locators which uses terms that made sense to humans like near, above, below, to\_left\_of, to\_right\_of.
* Provides authentication to websites with basic or digest authentication.
* Can intercept network Traffic.
* Easier way to open new Browser windows/Tabs (driver.switchTo().newWindow(WindowType.WINDOW/TAB))
* Capturing screenshot of particular web element (File srcfile = ((Takescreenshot)element).getscreenshotas(outputType.file)
* Deprecation of desired capabilities as Capabilities objects are now replaced with Options and we need to create an options object to use the driver class.(ChromeOptions options = new ChromeOptions(); options.setAcceptInsecureCerts("true"); options.setCapability("browserversion",latest) )
* Fluent Wait changes - withTimeOut, PollingEvery takes only one parameter ( Wait<webdriver> fluentwait = new FluentWait<Webdriver>(driver).withTimeOut(Duration.ofSeconds(120)).pollingEvery(Duration.ofMillis(2000).ignoring(NoSuchElementException.class);