**Selenium:** It is an open-source web application test automation tool

**Test automation:** Performing testing of the application automatically without any manual intervention is called as test automation. Simulation of any work using a system or a tool is called as automation; here tool refers to another software application.

**Why selenium?**

1.It is free: No need to purchase license

2. Open source

3.Platform independent

4.Supports all the popular browsers: Easy to perform browser compatibility testing

5.Supports 14 different coding languages

**What are the components of selenium? Or flavors of selenium?**

1.Selenium core

2.Selenium IDE

3.Selenium RC(remote control)

4.Selenium WebDriver

5.Selenium Grid

6.Appium

7.Winium

**What are the languages supported by selenium?**

Java, C#, Ruby, Python, JavaScript (Node JS), Perl, PHP, Objective-C, JavaScript, Haskell, R, Dart, TCL.

**What are all the operating systems supported by selenium?**

All

What are all the browsers supported by selenium?

1.Chrome

2.Firefox

3.IE

4.Edge

5.Opera

6.Safari

7.Android browser

8.Iphone browser

9.Ipad browser

10.Opera blink

11.Pahntom Js:

12.Html unit

Phantom Js and Html Unit are Headless browsers or transparent browsers which are used to server-client response without GUI.

**Installing Selenium:**

1.Before installing selenium, make sure the following software are already installed

* JDK 1.8 or above
* Eclipse IDE
* Required Browsers

2.Download Required “Selenium -Server-standalone” Jar file from the following url:

<https://selenium-release.storage.googleapis.com/index.html>

3.Download the required browser’s driver files from the following url:

<https://selenium.dev/downloads/>

Firefox:

<https://ftp.mozilla.org/pub/firefox/releases/47.0.1/win64/en-US/>

**Eclipse-Selenium Mapping**:

1.Open Eclipse IDE -go to “Windows->Show View->Package Explorer” if package explorer is not visible

2.Right click on the required project present in the “package explorer”-> New->Folder->Create 2 folders with names “Drivers” and “Jars” and copy paste the Driver executable file(exe) into Drivers folder and copy paste the “Selenium-Server-Standalone.jar” file into “Jars” folder.

3.Right click on the “java Project”->Properties->Libraries Tab->click on “Add External Jars...” ->Browse and select the “Selenium-Server-Standalone.jar” file from the “Jars” folder. This will create “Selenium-server-standalone.jar” file under “Referenced Libraries”.

**List of Maven Dependencies:**

1. **Selenium Dependency:**

|  |
| --- |
| **URL:-**  https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java  **Group ID:-** org.seleniumhq.selenium  **Artifact ID:-** selenium-java  **Version:-** LATEST |

**Q) What is the use of ‘*System.setProperty ()*’? When to use this?**

System.setProperty () is used to **set the path of driver executables**. We can write this statement anywhere but it should be **before opening the browser**.

**If we do not set the path of driver executables what is the exception we get?**

* IllegalStateException

**Architecture of Selenium:**

|  |
| --- |
|  |

**Description:**

* Selenium supports 14 different languages such as Java, C#, Python etc..
* In order to write automation script in java we should download the selenium software which is developed using java. This software is called as ‘**Language Binding**’ or ‘Client Binding’ software.
* During run time these Client Bindings communicates with ‘**Selenium-Server**’.
* ‘Selenium-Server’ internally has ‘**Selenium-Java-Language-Binding**’(SJLB)
* Selenium-Server performs the action on the browser with the help of browser specific ‘**Driver Executables**’ such as ChromeDriver’ for chrome browser & ‘GeckoDriver’ for Firefox browser.
* During execution browser communicates with application server. I.e. **Application under Test**.
* For communication purpose selenium uses a protocol called ‘JSON’ (Java Script Object Notation) wire protocol.
* **For ‘Selenium-Java’ we use only ‘Selenium-Server’ & ‘Driver Executables’.** We will not use ‘Client Binding’ because selenium-server internally contains ‘Selenium-Java-Language-Binding’.

**Architecture of Selenium with respect to Java Language Binding:**

|  |
| --- |
|  |

**Description:**

* In the ‘Selenium-Java’ language binding, super most interface is ‘**SearchContext’**. Which is extended by ‘**WebDriver’** interface.
* All abstract methods are implemented in ‘**RemoteWebDriver’** class.
* ‘RemoteWebDriver’ class implements multiple interfaces such as ‘**WebDriver**, **JavaScriptExecutor**, **TakesScreenshot**, etc.… (**total-13**)
* All these implemented methods are inherited by browser specific classes such as **ChromeDriver**, **FirefoxDriver**, **SafariDriver**, etc.…

**Important Methods of Selenium:**

**SearchContext (Interface):**

1. findElement()
2. findElements()

**WebDriver (Interface):**

1. get()
2. close()
3. getCurrentUrl()
4. getPageSource()
5. getTitle()
6. getWindowHandle()
7. getWindowHandles()
8. manage()
9. navigate()
10. quit()
11. swithTo()

**JavaScriptExecutor (Interface):**

1. executeAsyncScript()
2. executeScript()

**TakesScreenshot (Interface):**

1. getScreenshotAs()

**Q) What is ‘Up-casting’? Why we use up-casting in selenium?**

🡪 Converting sub-class object to super type is called as up-casting. We do up-casting to achieve the ‘**Run time Polymorphism**’. In selenium we do up-casting to execute the same method or script on many browsers.

**Q) Give an example for Up-casting? Explain in details.**

Ex:- ***WebDriver driver=new ChromeDriver ();***

In the above example ‘WebDriver’ is parent & ‘ChromeDriver’ is child.

‘**WebDriver’** 🡪 is an Interface, ‘**driver’** 🡪 is Reference variable, ‘**new’** 🡪 keyword to create object, ‘**ChromeDriver ()**’ 🡪 Constructor of ChromeDriver class. With respect to java its initialize the object & with respect to selenium it opens the browser.

**Q) Why do you do up-casting to ‘WebDriver’? Why not to other interface or class?**

🡪 It’s Standard in Selenium. In selenium after opening the browser we will enter the ‘URL’ using ‘**get ()**’ method of WebDriver interface & then we perform other actions. If we up-cast the browser object to any other interface ‘**get ()**’ method will be hidden & we cannot perform any other actions.

**Q) Write Script to print Current URL & Title of the page?**

|  |
| --- |
| WebDriver driver=new ChromeDriver();  driver.get("http://localhost:8080/login.do");  **String** title=**driver.getTitle();**  **String** URL=**driver.getCurrentUrl();**  System.out.println("Title is "+title );  System.out.println("Url is "+URL);  driver.close(); |

**Q) How do you enter URL without using ‘get ()’ method?**

🡪 **driver.navigate().to** (“https://www.google.com”);

**Q) How do you close the browser without using ‘close ()’ method?**

🡪 driver.quit ();

**Q) How do you navigate the web pages in Selenium?**

|  |
| --- |
| WebDriver driver=new ChromeDriver();  driver.get("https://www.google.com/");  **driver.navigate().to**("https://www.gmail.com/");  **driver.navigate().back();**  **driver.navigate().forward();**  **driver.navigate().refresh();**  driver.close(); |

**Q) What is the difference between ‘get ()’ & ‘navigate ()’ method?**

🡪 ‘**get ()**’ method is used to only enter the URL. Whereas ‘**navigate ()**’ method is used to enter the URL, click back, click forward & refresh the page.

**Q) What is the difference between ‘get ()’ & ‘to ()’ method?**

🡪 There is no difference because ‘to ()’ method internally calls ‘get ()’ method to enter the URL.

**Q) What is the difference between ‘close()’ & ‘quit()’?**

🡪 ‘**close()**’ method closes the current browser, whereas ‘**quit()**’ method closes the all the browsers. (Parent & child browsers)