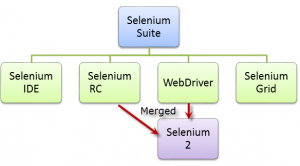
**1) What is Selenium and what is composed of?**

Selenium is a suite of tools for automated web testing.  It is composed of

* **Selenium IDE (Integrated Development Environment) :**It is a tool for recording and playing back.  It is a firefox plugin
* **WebDriver and RC:**  It provide the APIs for a variety of languages like Java, .NET, PHP, etc. With most of the browsers Webdriver and RC works.
* **Grid:**With the help of Grid you can distribute tests on multiple machines so that test can be run parallel which helps in cutting down the time required for running in browser test suites

**2) What is Selenium 2.0 ?**

Web testing tools Selenium RC and WebDriver are consolidated in single tool in Selenium 2.0

[](http://cdn.career.guru99.com/wp-content/uploads/2014/08/SeleniumSuite.png)

**3) How will you find an element using Selenium?**

In Selenium every object or control in a web page is referred as an elements, there are different ways to find an element in a web page they are

* ID
* Name
* Tag
* Attribute
* CSS
* Linktext
* PartialLink Text
* Xpath etc

**4) List out the test types that are supported by Selenium?**

For web based application testing selenium can be used

The test types can be supported are

a)      Functional

b)      Regression

For post release validation with continuous integration automation tool could be used

a)      Jenkins

b)      Hudson

c)       Quick Build

d)      CruiseCont

**5) Explain what is assertion in Selenium and what are the types of assertion?**

Assertion is used as a verification point. It verifies that the state of the application conforms to what is expected. The types of assertion are “assert”, “verify” and “waifFor”.

**6) Mention what is the use of X-path?**

X-Path is used to find the WebElement in web pages. It is also useful in identifying the dynamic elements.

**7) Explain the difference between single and double slash in X-path?**

Single slash ‘/ ’

* Single slash ( / ) start selection from the document node
* It allows you to create ‘absolute’ path expressions

Double Slash ‘// ’

* Double slash ( // ) start selection matching anywhere in the document
* It enables to create ‘relative’ path expressions

**8) List out the technical challenges with Selenium?**

Technical challenges with Selenium are

* Selenium supports only web based applications
* It does not support the Bitmap comparison
* For any reporting related capabilities have to depend on third party tools
* No vendor support for tool compared to commercial tools like HP UFT
* As there is no object repository concept in Selenium, maintainability of objects becomes difficult

**9) What is the difference between type keys and type commands ?**

TypeKeys() will trigger JavaScript event in most of the cases whereas .type() won’t. Type key populates the value attribute using JavaScript whereas .typekeys() emulates like actual user typing

**10) What is the difference between verify and assert commands?**

**Assert:**  Assert allows to check whether an element is on the page or not. The test will stop on the step failed, if the asserted element is not available. In other words, the test will terminated at the point where check fails.

**Verify:** Verify command will check whether the element is on the page, if it is not then the test will carry on executing.  In verification, all the commands are going to run guaranteed even if any of test fails.

**11) What is JUnit Annotations and what are different types of annotations which are useful ?**

In JAVA a special form of syntactic meta-data can be added to Java source code, this is known as Annotations.  Variables, parameters, packages, methods and classes are annotated some of the JUnit annotations which can be useful are

* Test
* Before
* After
* Ignore
* BeforeClass
* AfterClass
* RunWith

**12) While using click command can you use screen coordinate?**

To click on specific part of element, you would need to use clickAT command.  ClickAt command accepts element locator and x, y co-ordinates as arguments-

clickAt (locator, cordString)

**13)  What are the advantages of Selenium?**

* It supports C#, PHP, Java, Perl, Phython
* It supports different OS like Windows, Linux and Mac OS
* It has got powerful methods to locate elements (Xpath, DOM , CSS)
* It has highly developer community supported by Google

**14) Why testers should opt for Selenium and not QTP?**

Selenium is more popular than QTP as

* Selenium is an open source whereas QTP is a commercial tool
* Selenium is used specially for testing web based applications while QTP can be used for testing client server application also
* Selenium supports Firefox, IE, Opera, Safari  on operating systems like Windows, Mac, linux etc. however QTP is limited to Internet Explorer on Windows.
* Selenium supports many programming languages like Ruby, Perl, Python whereas QTP supports only VB script

**15) What are the four parameter you have to pass in Selenium?**

Four parameters that you have to pass in Selenium are

* Host
* Port Number
* Browser
* URL

**16) What is the difference between setSpeed() and sleep() methods?**

Both will delay the speed of execution.

**Selenium.setSpeed:**  
1. takes a single argument in string format  
ex: selenium.setSpeed(“2000”) – will wait for 2 seconds  
2. Runs each command in after setSpeed delay by the number of milliseconds mentioned in setSpeed.

**Thread.sleep:**  
1. takes a single argument in integer format  
ex: thread.sleep(2000) – will wait for 2 seconds  
2. Waits for only once at the command given at sleep.

If we want to set a delay time 2000 for each of these, defining setSpeed() method once will accomplish the task something like below:

selenium.setSpeed("2000");

Opeartion 1

Opeartion 2

Opeartion 3

But if we use Thread.sleep(), it will be something like below:

Thread.sleep(2000);

Opeartion 1

Thread.sleep(2000);

Opeartion 2

Thread.sleep(2000);

Opeartion 3

**17) What is same origin policy? How you can avoid same origin policy?**

The **“Same Origin Policy”** is introduced for security reason, and it ensures that content of your site will never be accessible by a script from another site.  As per the policy, any code loaded within the browser can only operate within that website’s domain.

To avoid “Same Origin Policy” proxy injection method is used, in proxy injection mode the Selenium Server acts as a client configured **HTTP proxy** , which sits between the browser and application under test and then masks the AUT under a fictional URL

**18) What is heightened privileges browsers?**

The purpose of heightened privileges is similar to Proxy Injection, allows websites to do something that are not commonly permitted.  The key difference is that the browsers are launced in a special mode called heightened privileges.  By using these browser mode, Selenium core can open the AUT directly and also read/write its content without passing the whole AUT through the Selenium RC server.

**19) How you can use “submit” a form using Selenium ?**

You can use “submit” method on element to submit form-

element.submit () ;

Alternatively you can use click method on the element which does form submission

**20) What are the features of TestNG and list some of the functionality in TestNG which makes it more effective?**

TestNG is a testing framework based on JUnit and NUnit to simplify a broad range of testing needs, from unit testing to integration testing. And the functionality which makes it efficient testing framework are

* Support for annotations
* Support for data-driven testing
* Flexible test configuration
* Ability to re-execute failed test cases

**21) Mention what is the difference between Implicit wait and Explicit wait?**

There are two [types of waits available in Web Driver](http://www.softwaretestinghelp.com/selenium-webdriver-waits-selenium-tutorial-15/):

1. Implicit Wait
2. Explicit Wait

**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.

**dr.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);**

**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.

**WebDriverWait wait = new WebDriverWait(driver, *time period*);**

**wait.until(ExpectedConditions.textToBePresentInElement(By.xpath(getTxtVerifyTitle()), *Text Title*));**

**22) Which attribute you should consider throughout the script in frame for “if no frame Id as well as no frame name”?**

You can use…..driver.findElements(By.xpath(“//iframe”))….

This will return list of frames.

You will need to  switch to each and every frame and search for locator which we want.

Then break the loop

**23) Explain what is the difference between find elements () and find element () ?**

***findElement() :***

1. Find the first element within the current page using the given "locating mechanism".
2. Returns a single WebElement.
3. Syntax: WebElement findElement(By by)

*Ex:*

[view plainprint?](http://www.mythoughts.co.in/2012/05/selenium-webdriver-findelement-vs.html)

1. driver.get("https://signup.live.com");
2. WebElement firstName=driver.findElement(By.id("iFirstName"));
3. firstName.sendKeys("Automated First Name");

***findElements() :***

1. Find all elements within the current page using the given "locating mechanism".
2. Returns List of WebElements.
3. Syntax:  java.util.List<WebElement> findElements(By by)

*Ex:*

[view plainprint?](http://www.mythoughts.co.in/2012/05/selenium-webdriver-findelement-vs.html)

1. driver.get("https://signup.live.com");
2. List <webelement> textboxes1=driver.findElements(By.xpath("//input[@type='text']"));
3. System.out.println("total textboxes "+textboxes1.size());
5. </webelement>

**24) Explain what are the JUnits annotation linked with Selenium?**

The JUnits annotation linked with Selenium are

* @Before public void method() – It will perform the method () before each test, this method can prepare the test
* @Test public void method() – Annotations @Test identifies that this method is a test method environment
* @After public void method()- To execute a method before this annotation is used, test method must start with test@Before

**25) Explain what is Datadriven framework and Keyword driven?**

**Data driven framework:**  In this framework, the test data is separated and kept outside the Test Scripts, while test case logic resides in Test Scripts.  Test data is read from the external files ( Excel Files) and are loaded into the variables inside the Test Script.  Variables are used for both for input values and for verification values.

**Keyword driven framework:** The keyword driven frameworks requires the development of data tables and keywords, independent of the test automation.  In a keyword driven test, the functionality of the application under test is documented in a table as well as step by step instructions for each test.

**26) Explain how you can login into any site if it’s showing any authentication popup for password and username?**

Pass the username and password with url

* Syntax-http://username:password@url
* ex- http://creyate:tom@www.gmail.com

**27) Explain how to assert text of webpage using selenium 2.0 ?**

WebElement el = driver.findElement(By.id(“ElementID”))

//get test from element and stored in text variable

String text = el.getText();

//assert text from expected

Assert.assertEquals(“Element Text”, text);

**28) Explain what is the difference between Borland Silk and Selenium?**

|  |  |
| --- | --- |
| **Silk Test Tool** | **Selenium Test Tool** |
| * Borland Silk test is not a free testing tool | * Selenium is completely free test automation tool |
| * Silk test supports only Internet Explorer and Firefox | * Selenium supports many browsers like Internet Explorer, Firefox, Safari, Opera and so on |
| * Silk test uses test scripting language | * Selenium suite has the flexibility to use many languages like Java, Ruby,Perl and so on |
| * Silk test can be used for client server applications | * Selenium can be used for only web application |

**29) What is Object Repository ?**

An object repository is an essential entity in any UI automations which allows a tester to store all object that will be used in the scripts in one or more centralized locations rather than scattered all over the test scripts.

**30) Explain how Selenium Grid works?**

Selenium Grid sent the tests to the hub. These tests are redirected to Selenium Webdriver, which launch the browser and run the test.  With entire test suite, it allows for running tests in parallel.

**31) Can we use Selenium grid for performance testing?**

Yes. But not as effectively as a dedicated performance testing tool like Loadrunner.

**Q #3) Why should Selenium be selected as a test tool?**

Selenium

1. is free and open source
2. have a large user base and helping communities
3. have cross Browser compatibility (Firefox, chrome, Internet Explorer, Safari etc.)
4. have great platform compatibility (Windows, Mac OS, Linux etc.)
5. supports multiple programming languages (Java, C#, Ruby, Python, Pearl etc.)
6. has fresh and regular repository developments
7. supports distributed testing

**Q #4) What is Selenium? What are the different Selenium components?**

Selenium is one of the most popular automated testing suites. Selenium is designed in a way to support and encourage automation testing of functional aspects of web based applications and a wide range of browsers and platforms. Due to its existence in the open source community, it has become one of the most accepted tools amongst the testing professionals.

Selenium is not just a single tool or a utility, rather a package of several testing tools and for the same reason it is referred to as a Suite. Each of these tools is designed to cater different testing and test environment requirements.

The suite package constitutes of the following sets of tools:

* [**Selenium Integrated Development Environment (IDE)**](http://www.softwaretestinghelp.com/selenium-ide-download-and-installation-selenium-tutorial-2/) – Selenium IDE is a record and playback tool. It is distributed as a Firefox Plugin.
* **Selenium Remote Control (RC)** – Selenium RC is a server that allows user to create test scripts in a desired programming language. It also allows executing test scripts within the large spectrum of browsers.
* [**Selenium WebDriver**](http://www.softwaretestinghelp.com/selenium-webdriver-selenium-tutorial-8/) – WebDriver is a different tool altogether that has various advantages over Selenium RC. WebDriver directly communicates with the web browser and uses its native compatibility to automate.
* [**Selenium Grid**](http://www.softwaretestinghelp.com/selenium-grid-selenium-tutorial-29/) – Selenium Grid is used to distribute your test execution on multiple platforms and environments concurrently.

**Q #5) What are the testing types that can be supported by Selenium?**

Selenium supports the following types of testing:

1. Functional Testing
2. Regression Testing

**Q #6) What are the limitations of Selenium?**

Following are the limitations of Selenium:

* Selenium supports testing of only web based applications
* Mobile applications cannot be tested using Selenium
* Captcha and Bar code readers cannot be tested using Selenium
* Reports can only be generated using third party tools like TestNG or Junit.
* As Selenium is a free tool, thus there is no ready vendor support though the user can find numerous helping communities.
* User is expected to possess prior programming language knowledge.

**Q #7)** **What is the difference between Selenium IDE, Selenium RC and WebDriver?**

| **Feature** | **Selenium IDE** | **Selenium RC** | **WebDriver** |
| --- | --- | --- | --- |
|  |  |  |  | |
| Browser Compatibility | Selenium IDE comes as a Firefox plugin, thus it supports only Firefox | Selenium RC supports a varied range of versions of Mozilla Firefox, Google Chrome, Internet Explorer and Opera | WebDriver supports a varied range of versions of Mozilla Firefox, Google Chrome, Internet Explorer and Opera. Also supports HtmlUnitDriver which is a GUI less or headless browser. | |
| Record and Playback | Selenium IDE supports record and playback feature | Selenium RC doesn't supports record and playback feature | WebDriver doesn't support record and playback feature | |
| Server Requirement | Selenium IDE doesn't require any server to be started before executing the test scripts | Selenium RC requires server to be started before executing the test scripts | WebDriver doesn't require any server to be started before executing the test scripts | |
| Architecture | Selenium IDE is a Javascript based framework | Selenium RC is a JavaScript based Framework | WebDriver uses the browser's native compatibility to automation | |
| Object Oriented | Selenium IDE is not an object oriented tool | Selenium RC is semi object oriented tool | WebDriver is a purely object oriented tool | |
| Dynamic Finders (for locating web elements on a webpage) | Selenium IDE doesn't support dynamic finders | Selenium RC doesn't support dynamic finders | WebDriver supports dynamic finders | |
| Handling Alerts, Navigations, Dropdowns | Selenium IDE doesn't explicitly provides aids to handle alerts, navigations, dropdowns | Selenium RC doesn't explicitly provides aids to handle alerts, navigations, dropdowns | WebDriver offers a wide range of utilities and classes that helps in handling alerts, navigations, and dropdowns efficiently and effectively. | |
| WAP (iPhone/Android) Testing | Selenium IDE doesn't support testing of iPhone/Andriod applications | Selenium RC doesn't support testing of iPhone/Andriod applications | WebDriver is designed in a way to efficiently support testing of iPhone/Android applications. The tool comes with a large range of drivers for WAP based testing. For example, AndroidDriver, iPhoneDriver | |
| Listener Support | Selenium IDE doesn't support listeners | Selenium RC doesn't support listeners | WebDriver supports the implementation of Listeners | |
| Speed | Selenium IDE is fast as it is plugged in with the web-browser that launches the test. Thus, the IDE and browser communicates directly | Selenium RC is slower than WebDriver as it doesn't communicates directly with the browser; rather it sends selenese commands over to Selenium Core which in turn communicates with the browser. | WebDriver communicates directly with the web browsers. Thus making it much faster. | |

**Q #8) When should I use Selenium IDE?**

Selenium IDE is the simplest and easiest of all the tools within the Selenium Package. Its record and playback feature makes it exceptionally easy to learn with minimal acquaintances to any programming language. Selenium IDE is an ideal tool for a naïve user.

**Q #9) What is Selenese?**

Selenese is the language which is used to write test scripts in Selenium IDE.

**Q #10)** **What are the different types of locators in Selenium?**

Locator can be termed as an address that identifies a web element uniquely within the webpage. Thus, to identify web elements accurately and precisely we have [different types of locators in Selenium](http://www.softwaretestinghelp.com/using-selenium-xpath-and-other-locators-selenium-tutorial-5/):

* ID
* ClassName
* Name
* TagName
* LinkText
* PartialLinkText
* Xpath
* CSS Selector
* DOM

**Q #11)** **What is difference between assert and verify commands?**

**Assert:**Assert command checks whether the given condition is true or false. Let’s say we assert whether the given element is present on the web page or not. If the condition is true then the program control will execute the next test step but if the condition is false, the execution would stop and no further test would be executed.

**Verify:**Verify command also checks whether the given condition is true or false. Irrespective of the condition being true or false, the program execution doesn’t halts i.e. any failure during verification would not stop the execution and all the test steps would be executed.

**Q #12) What is an Xpath?**

[Xpath](http://www.softwaretestinghelp.com/using-selenium-xpath-and-other-locators-selenium-tutorial-5/) is used to locate a web element based on its XML path. XML stands for Extensible Markup Language and is used to store, organize and transport arbitrary data. It stores data in a key-value pair which is very much similar to HTML tags. Both being markup languages and since they fall under the same umbrella, Xpath can be used to locate HTML elements.

The fundamental behind locating elements using Xpath is the traversing between various elements across the entire page and thus enabling a user to find an element with the reference of another element.

**Q #13) What is the difference between “/” and “//” in Xpath?**

**Single Slash “/” –**Single slash is used to create Xpath with absolute path i.e. the xpath would be created to start selection from the document node/start node.

**Double Slash “//” -** Double slash is used to create Xpath with relative path i.e. the xpath would be created to start selection from anywhere within the document.

**Q #14) What is Same origin policy and how it can be handled?**

The problem of same origin policy disallows to access the DOM of a document from an origin that is different from the origin we are trying to access the document.

Origin is a sequential combination of scheme, host and port of the URL. For example, for a URL http:// http://www.softwaretestinghelp.com/resources/, the origin is a combination of http, softwaretestinghelp.com, 80 correspondingly.

Thus the Selenium Core (JavaScript Program) cannot access the elements from an origin that is different from where it was launched. For Example, if I have launched the JavaScript Program from “http://www.softwaretestinghelp.com”, then I would be able to access the pages within the same domain such as “http://www.softwaretestinghelp.com/resources” or “http://www.softwaretestinghelp.com/istqb-free-updates/”. The other domains like google.com, seleniumhq.org would no more be accessible.

So, In order to handle same origin policy, Selenium Remote Control was introduced.

**Q #15)** **When should I use Selenium Grid?**

Selenium Grid can be used to execute same or different test scripts on multiple platforms and browsers concurrently so as to achieve distributed test execution, testing under different environments and saving execution time remarkably.

**Q #16) What do we mean by Selenium 1 and Selenium 2?**

Selenium RC and WebDriver, in a combination are popularly known as Selenium 2. Selenium RC alone is also referred as Selenium 1.

**Q #17) Which is the latest Selenium tool?**

WebDriver

**Q #18) How do I launch the browser using WebDriver?**

The following syntax can be used to launch Browser:  
*WebDriver driver =****new****FirefoxDriver();*  
*WebDriver driver =****new****ChromeDriver();*  
*WebDriver driver =****new****InternetExplorerDriver();*

**Q #19) What are the different types of Drivers available in WebDriver?**

The different drivers available in WebDriver are:

* FirefoxDriver
* InternetExplorerDriver
* ChromeDriver
* SafariDriver
* OperaDriver
* AndroidDriver
* IPhoneDriver
* HtmlUnitDriver

**Q #20) What are the different types of waits available in WebDriver?**

There are two [types of waits available in WebDriver](http://www.softwaretestinghelp.com/selenium-webdriver-waits-selenium-tutorial-15/):

1. Implicit Wait
2. Explicit Wait

**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.

**Q #21)** **How to type in a textbox using Selenium?**

User can use sendKeys(“String to be entered”) to enter the string in the textbox.

**Syntax:**  
*WebElement username = drv.findElement(By.id(“Email”));*  
*// entering username*  
*username.sendKeys(“sth”);*

**Q #22)** **How can you find if an element in displayed on the screen?**

WebDriver facilitates the user with the following methods to check the visibility of the web elements. These web elements can be buttons, drop boxes, checkboxes, radio buttons, labels etc.

1. isDisplayed()
2. isSelected()
3. isEnabled()

**Syntax:**

**isDisplayed():**  
***boolean****buttonPresence = driver.findElement(By.id(“gbqfba”)).isDisplayed();*

**isSelected():**  
***boolean****buttonSelected = driver.findElement(By.id(“gbqfba”)).isDisplayed();*

**isEnabled():**  
***boolean****searchIconEnabled = driver.findElement(By.id(“gbqfb”)).isEnabled();*

**Q #23)** **How can we get a text of a web element?**

Get command is used to retrieve the inner text of the specified web element. The command doesn’t require any parameter but returns a string value. It is also one of the extensively used commands for verification of messages, labels, errors etc displayed on the web pages.

**Syntax:**  
*String Text = driver.findElement(By.id(“Text”)).getText();*

**Q #24) How to select value in a dropdown?**

Value in the drop down can be selected using WebDriver’s Select class.

**Syntax:**

**selectByValue:**  
*Select selectByValue =****new****Select(driver.findElement(By.id(“SelectID\_One”)));*  
*selectByValue.selectByValue(“greenvalue”);*

**selectByVisibleText:**  
*Select selectByVisibleText =****new****Select (driver.findElement(By.id(“SelectID\_Two”)));*  
*selectByVisibleText.selectByVisibleText(“Lime”);*

**selectByIndex:**  
*Select selectByIndex =****new****Select(driver.findElement(By.id(“SelectID\_Three”)));*  
*selectByIndex.selectByIndex(2);*

**WebElement room\_type =dr.findElement(By.*name*("room\_type"));**

**Select room\_type\_list = new Select(room\_type);**

**room\_type\_list.selectByValue("Double");**

**Q #25) What are the different types of navigation commands?**

Following are the [navigation commands](http://www.softwaretestinghelp.com/selenium-webdriver-waits-selenium-tutorial-15/):  
**navigate().back()** – The above command requires no parameters and takes back the user to the previous webpage in the web browser’s history.

**Sample code:**  
*driver.navigate().back();*

**navigate().forward()** – This command lets the user to navigate to the next web page with reference to the browser’s history.

**Sample code:**  
*driver.navigate().forward();*

**navigate().refresh()** – This command lets the user to refresh the current web page there by reloading all the web elements.

**Sample code:**  
*driver.navigate().refresh();*

**navigate().to()** – This command lets the user to launch a new web browser window and navigate to the specified URL.

**Sample code:**  
*driver.navigate().to(“https://google.com”);*

**Q #26) How to click on a hyper link using linkText?**

*driver.findElement(By.linkText(“Google”)).click();*

The command finds the element using link text and then click on that element and thus the user would be re-directed to the corresponding page.

The above mentioned link can also be accessed by using the following command.

*driver.findElement(By.partialLinkText(“Goo”)).click();*

The above command find the element based on the substring of the link provided in the parenthesis and thus partialLinkText() finds the web element with the specified substring and then clicks on it.

**Q #27)** **How to**[**handle frame in WebDriver**](http://www.softwaretestinghelp.com/selenium-tutorial-18/)**?**

An inline frame acronym as iframe is used to insert another document with in the current HTML document or simply a web page into a web page by enabling nesting.

**Select iframe by id**  
*driver.switchTo().frame(“ID of the frame“);*

**Locating iframe using tagName**  
*driver.switchTo().frame(driver.findElements(By.tagName(“iframe”).get(0));*

**Locating iframe using index**

**frame(index)**  
*driver.switchTo().frame(0);*

**frame(Name of Frame)**  
*driver.switchTo().frame(“name of the frame”);*

**frame(WebElement element)**  
**Select Parent Window**  
*driver.switchTo().defaultContent();*

**Q #29)** **How to find more than one web element in the list?**

At times, we may come across elements of same type like multiple hyperlinks, images etc arranged in an ordered or unordered list. Thus, it makes absolute sense to deal with such elements by a single piece of code and this can be done using WebElement List.

**Sample Code**

|  |  |
| --- | --- |
| 1 | // Storing the list |
| 2 | List <WebElement> elementList = driver.findElements(By.xpath("//div[@id='example']//ul//li")); | |

|  |  |
| --- | --- |
| 3 | // Fetching the size of the list |
| 4 | int listSize = elementList.size(); | |

|  |  |  |
| --- | --- | --- |
| 5 | for (int i=0; i<listSize; i++) | |
| 6 | { |

|  |  |  |
| --- | --- | --- |
| 7 | // Clicking on each service provider link | |
| 8 | serviceProviderLinks.get(i).click(); |

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | // Navigating back to the previous page that stores link to service providers | | |
| 10 | | driver.navigate().back(); |

|  |  |
| --- | --- |
| 11 | } |

**Q #30) What is the difference between driver.close() and driver.quit command?**

**close()**: WebDriver’s close() method closes the web browser window that the user is currently working on or we can also say the window that is being currently accessed by the WebDriver. The command neither requires any parameter nor does is return any value.

**quit()**: Unlike close() method, quit() method closes down all the windows that the program has opened. Same as close() method, the command neither requires any parameter nor does is return any value.

**Q #31) Can Selenium handle windows based pop up?**

Selenium is an automation testing tool which supports only web application testing. Therefore, windows pop up cannot be handled using Selenium.

**Q #32) How can we handle web based pop up?**

WebDriver offers the users with a very efficient way to [handle these pop ups using Alert interface](http://www.softwaretestinghelp.com/handle-alerts-popups-selenium-webdriver-selenium-tutorial-16/). There are the four methods that we would be using along with the Alert interface.

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

**Syntax:**  
*// accepting javascript alert*  
*Alert alert = driver.switchTo().alert();*  
*alert.accept();*

**Q #33) How can we handle windows based pop up?**

Selenium is an automation testing tool which supports only web application testing, that means, it doesn’t support testing of windows based applications. However Selenium alone can’t help the situation but along with some third party intervention, this problem can be overcome. There are several third party tools available for handling window based pop ups along with the selenium like AutoIT, Robot class etc.

**Q #34) How to assert title of the web page?**

*//verify the title of the web page*  
*assertTrue(“The title of the window is incorrect.”,driver.getTitle().equals(“Title of the page”));*

**Q #35) How to mouse hover on a web element using WebDriver?**

WebDriver offers a wide range of interaction utilities that the user can exploit to automate mouse and keyboard events. Action Interface is one such utility which simulates the single user interactions.

Thus, In the following scenario, we have used Action Interface to mouse hover on a drop down which then opens a list of options.

**Sample Code:**

|  |  |
| --- | --- |
| 1 | // Instantiating Action Interface |
| 2 | Actions actions=new Actions(driver); | |

|  |  |
| --- | --- |
| 3 | // howering on the dropdown |
| 4 | actions.moveToElement(driver.findElement(By.id("id of the dropdown"))).perform(); | |

|  |  |
| --- | --- |
| 5 | // Clicking on one of the items in the list options |
| 6 | WebElement subLinkOption=driver.findElement(By.id("id of the sub link")); | |

|  |  |
| --- | --- |
| 7 | subLinkOption.click(); |

**Q #36) How to retrieve css properties of an element?**

The values of the css properties can be retrieved using a get() method:

**Syntax:**  
*driver.findElement(By.id(“id“)).getCssValue(“name of css attribute”);*  
*driver.findElement(By.id(“id“)).getCssValue(“font-size”);*

**Q #38) What is Junit?**

[Junit](http://www.softwaretestinghelp.com/selenium-junit-framework-selenium-tutorial-11/) is a unit testing framework introduced by Apache. Junit is based on Java.

**Q #39) What are Junit annotations?**

Following are the Junit Annotations:

* **@Test:**Annotation lets the system know that the method annotated as @Test is a test method. There can be multiple test methods in a single test script.
* **@Before:**Method annotated as @Before lets the system know that this method shall be executed every time before each of the test method.
* **@After:**Method annotated as @After lets the system know that this method shall be executed every time after each of the test method.
* **@BeforeClass:**Method annotated as @BeforeClass lets the system know that this method shall be executed once before any of the test method.
* **@AfterClass:**Method annotated as @AfterClass lets the system know that this method shall be executed once after any of the test method.
* **@Ignore:**Method annotated as @Ignore lets the system know that this method shall not be executed.

**Q #40)** **What is TestNG and how is it better than Junit?**

[TestNG](http://www.softwaretestinghelp.com/testng-framework-selenium-tutorial-12/) is an advance framework designed in a way to leverage the benefits by both the developers and testers. With the commencement of the frameworks, JUnit gained an enormous popularity across the Java applications, Java developers and Java testers with remarkably increasing the code quality. Despite being easy to use and straightforward, JUnit has its own limitations which give rise to the need of bringing TestNG into the picture. TestNG is an open source framework which is distributed under the Apache software License and is readily available for download.

TestNG with WebDriver provides an efficient and effective test result format that can in turn be shared with the stake holders to have a glimpse on the product’s/application’s health thereby eliminating the drawback of WebDriver’s incapability to generate test reports. TestNG has an inbuilt exception handling mechanism which lets the program to run without terminating unexpectedly.

There are various advantages that make TestNG superior to JUnit. Some of them are:

* Added advance and easy annotations
* Execution patterns can set
* Concurrent execution of test scripts
* Test case dependencies can be set

**Q #41)** **How to set test case priority in TestNG?**

**Setting Priority in TestNG**

**Code Snippet**

|  |  |
| --- | --- |
| 1 | package TestNG; |
| 2 | import org.testng.annotations.\*; | |

|  |  |  |
| --- | --- | --- |
| 3 | public class SettingPriority { | |
| 4 | @Test(priority=0) |

|  |  |  |
| --- | --- | --- |
| 5 | public void method1() { | |
| 6 | } |

|  |  |
| --- | --- |
| 7 | @Test(priority=1) |
| 8 | public void method2() { | |

|  |  |  |
| --- | --- | --- |
| 9 | } | |
| 10 | | @Test(priority=2) | |

|  |  |  |
| --- | --- | --- |
| 11 | public void method3() { | |
| 12 | } |

|  |  |
| --- | --- |
| 13 | } |

**Test Execution Sequence:**

1. Method1
2. Method2
3. Method3

**Q #42) What is a framework?**

Framework is a constructive blend of various guidelines, coding standards, concepts, processes, practices, project hierarchies, modularity, reporting mechanism, test data injections etc. to pillar automation testing.

**Q #43)** **What are the advantages of Automation framework?**

**Advantage of**[**Test Automation framework**](http://www.softwaretestinghelp.com/test-automation-frameworks-selenium-tutorial-20/)

* Reusability of code
* Maximum coverage
* Recovery scenario
* Low cost maintenance
* Minimal manual intervention
* Easy Reporting

**Q #44) What are the different types of frameworks?**

**Below are the different types of frameworks:**

1. **Module Based Testing Framework:** The framework divides the entire “Application Under Test” into number of logical and isolated modules. For each module, we create a separate and independent test script. Thus, when these test scripts taken together builds a larger test script representing more than one module.
2. **Library Architecture Testing Framework:** The basic fundamental behind the framework is to determine the common steps and group them into functions under a library and call those functions in the test scripts whenever required.
3. Data Driven Testing Framework: Data Driven Testing Framework helps the user segregate the test script logic and the test data from each other. It lets the user store the test data into an external database. The data is conventionally stored in “Key-Value” pairs. Thus, the key can be used to access and populate the data within the test scripts.
4. **Keyword Driven Testing Framework:** The Keyword driven testing framework is an extension to Data driven Testing Framework in a sense that it not only segregates the test data from the scripts, it also keeps the certain set of code belonging to the test script into an external data file.
5. **Hybrid Testing Framework:** Hybrid Testing Framework is a combination of more than one above mentioned frameworks. The best thing about such a setup is that it leverages the benefits of all kinds of associated frameworks.
6. **Behavior Driven Development Framework:** Behavior Driven Development framework allows automation of functional validations in easily readable and understandable format to Business Analysts, Developers, Testers, etc.

**Q #45) How can I read test data from excels?**

Test data can efficiently be read from excel using JXL or POI API.[See detailed tutorial here](http://www.softwaretestinghelp.com/selenium-framework-design-selenium-tutorial-21/).

**Q #46) What is the difference between POI and jxl jar?**

| **#** | **JXL jar** | **POI jar** |
| --- | --- | --- |
| 1 | JXL supports “.xls” format i.e. binary based format. JXL doesn’t support Excel 2007 and “.xlsx” format i.e. XML based format | POI jar supports all of these formats |
| 2 | JXL API was last updated in the year 2009 | POI is regularly updated and released |
| 3 | The JXL documentation is not as comprehensive as that of POI | POI has a well prepared and highly comprehensive documentation |
| 4 | JXL API doesn’t support rich text formatting | POI API supports rich text formatting |
| 5 | JXL API is faster than POI API | POI API is slower than JXL API |

**Q #47)** **What is the difference between Selenium and QTP?**

| **Feature** | **Selenium** | **Quick Test Professional (QTP)** |
| --- | --- | --- |
| Browser Compatibility | Selenium supports almost all the popular browsers like Firefox, Chrome, Safari, Internet Explorer, Opera etc | QTP supports Internet Explorer, Firefox and Chrome. QTP only supports Windows Operating System |
| Distribution | Selenium is distributed as an open source tool and is freely available | QTP is distributed as a licensed tool and is commercialized |
| Application under Test | Selenium supports testing of only web based applications | QTP supports testing of both the web based application and windows based application |
| Object Repository | Object Repository needs to be created as a separate entity | QTP automatically creates and maintains Object Repository |
| Language Support | Selenium supports multiple programming languages like Java, C#, Ruby, Python, Perl etc | QTP supports only VB Script |
| Vendor Support | As Selenium is a free tool, user would not get the vendor’s support in troubleshooting issues | Users can easily get the vendor’s support in case of any issue |

**Q #48) Can WebDriver test Mobile applications?**

WebDriver cannot test Mobile applications. WebDriver is a web based testing tool, therefore applications on the mobile browsers can be tested.

**Q #49) Can captcha be automated?**

No, captcha and bar code reader cannot be automated.

**Q #50) What is Object Repository? How can we create Object Repository in Selenium?**

Object Repository is a term used to refer to the collection of web elements belonging to Application Under Test (AUT) along with their locator values. Thus, whenever the element is required within the script, the locator value can be populated from the Object Repository. Object Repository is used to store locators in a centralized location instead of hard coding them within the scripts.

In Selenium, objects can be stored in an excel sheet which can be populated inside the script whenever required.

That’s all for now.

**Question 6:**Which web driver implementation is fastest?  
**Answer:** HTMLUnitDriver. Simple reason is HTMLUnitDriver does not execute tests on browser but plain http request – response which is far quick than launching a browser and executing tests. But then you may like to execute tests on a real browser than something running behind the scenes

**Question 9:** How to capture screen shot in Webdriver?

**Answer:**

File file= ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);

FileUtils.copyFile(file, new File("c:\\name.png"));

**Question 10:**How do I clear content of a text box in Selenium 2.0?

Answer:

WebElement el  =  driver.findElement(By.id("ElementID"));

el.clear();

**Question 13:** How to automate radio button in Selenium 2.0?

**Answer:**

WebElement el = driver.findElement(By.id("Radio button id"));

//to perform check operation

el.click()

//verfiy to radio button is check it return true if selected else false

el.isSelected()

**Question 15:** How to count total number of rows of a table using Selenium 2.0?

**Answer:**

List <WebElement> rows = driver.findElements(By.className("//table[@id='tableID']/tr"));

int totalRow = rows.size();

**Question 16:**  How to capture page title using Selenium 2.0?

**Answer:**

String title =  driver.getTitle()

**Question 17:** How to store page source using Selenium 2.0?

**Answer:**

String pagesource = driver.getPageSource()

**Question 18**: How to store current url using selenium 2.0?

**Answer:**

String currentURL  = driver.getCurrentUrl()

**Question 19**: How to assert text assert text of webpage using selenium 2.0?

**Answer:**

WebElement el  =  driver.findElement(By.id("ElementID"));

//get test from element and stored in text variable

String text = el.getText();

//assert text from expected

Assert.assertEquals("Element Text", text);

**Question 20:**How to get element attribute using Selenium 2.0?

**Answer:**

WebElement el  =  driver.findElement(By.id("ElementID"));

//get test from element and stored in text variable

String attributeValue = el. getAttribute("AttributeName") ;

**Question 21:** How to double click on element using selenium 2.0?

**Answer:**

WebElement el  =  driver.findElement(By.id("ElementID"));

Actions builder = new Actions(driver);

builder.doubleClick(el).build().perform();

**Question 22:**How to perform drag and drop in selenium 2.0?

**Answer:**

WebElement source  =  driver.findElement(By.id("Source ElementID"));

WebElement destination  =  driver.findElement(By.id("Taget ElementID"));

Actions builder = new Actions(driver);

builder.dragAndDrop(source, destination ).perform();

**22a) How to RightClick by Selenium WebDriver?**

Actions action= new Actions(driver);

action.contextClick(productLink).build().perform();

**Question 23:**How to maximize window using selenium 2.0?

**Answer:**

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

Road to verify PDF file text using java webdriver

In this post I will explain the procedure to verify PDF file content using java webdriver. As  some time we need to verify content of web application pdf file, opened in browser.  
Use below code in your test scripts to get pdf file content.

 //get current urlpdf file url

 URL url = new URL(driver.getCurrentUrl());

 //create buffer reader object

 BufferedInputStream fileToParse = new BufferedInputStream(url.openStream());

 PDFParser pdfParser = new PDFParser(fileToParse);

 pdfParser.parse();

 //save pdf text into strong variable

 String pdftxt = new PDFTextStripper().getText(pdfParser.getPDDocument());

 //close PDFParser object

 pdfParser.getPDDocument().close();

After applying above code, you can store all pdf file content into “pdftxt” string variable. Now you can verify string by giving  in put.  As if you want to verify “Selenium Or Webdiver” text. Use below code.

Assert.assertTrue(pdftxt.contains(“Selenium Or Webdiver”))

Hope this post will help  to verify web application PDF content.

Road to capture clip of element locator in webdriver java

In this post I am going to show you how to capture clip of page element using webdriver.  
Below I have written a “CaptureElementClip.java“java webdriver test script of a google application where I capture google menu clip and save into project.

package com.webdriver.test;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.IOException;

import java.util.concurrent.TimeUnit;

import javax.imageio.ImageIO;

import org.apache.commons.io.FileUtils;

import org.openqa.selenium.By;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.Point;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.testng.annotations.AfterSuite;

import org.testng.annotations.BeforeSuite;

import org.testng.annotations.Test;

public class CaptureElementClip {

        private WebDriver driver;

        private String baseUrl;

        @BeforeSuite

        public void setUp() throws Exception {

                    driver = new FirefoxDriver();

                    baseUrl = "http://google.com";

                    driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

         }

        @Test

        public void testGoogle() throws IOException {

                    //open application url

                    driver.get(baseUrl);

                    //take screen shot

                    File screen = ((TakesScreenshot) driver)

                                        .getScreenshotAs(OutputType.FILE);

                    //get webelement object of google menu locator

                    WebElement googleMenu = driver.findElement(By.id("gbz"));

                    Point point = googleMenu.getLocation();

                    //get element dimension

                    int width = googleMenu.getSize().getWidth();

                    int height = googleMenu.getSize().getHeight();

                    BufferedImage img = ImageIO.read(screen);

                    BufferedImage dest = img.getSubimage(point.getX(), point.getY(), width,

                                                                                 height);

                    ImageIO.write(dest, "png", screen);

                    File file = new File("Menu.png");

                    FileUtils.copyFile(screen, file);

          }

          @AfterSuite

          public void tearDown() throws Exception {

                    driver.quit();

           }

}

After executing above test a “Menu.png” file is generated in root folder of your project.

**16. How to handle drop down:** by using select class we can automate dropdown following are the some main operation which we perform on drop down list.  
By using selectByVisibleText() method of Select class we can select visible text of dropdown.

Select sel= new Select(driver.findElement(By.id("drop down ID")));

sel.selectByVisibleText("Pass your desire Text");

Select drop down by value  
Select sel= new Select(driver.findElement(By.id("drop down ID")));

sel. selectByValue( ("Pass your desire Text");

Select drop down by index  
Select sel= new Select(driver.findElement(By.id("drop down ID")));

sel. selectByIndex (0);  // 0 is first index

 Verify dropdown supported multiple options or not  
Select sel= new Select(driver.findElement(By.id("drop down ID")));

Boolean status = sel. isMultiple()

Get number of options are available in drop down  
Select sel= new Select(driver.findElement(By.id("drop down ID")));

int  totalOption  = sel. getOptions().size()

**Ques 10) What is Datadriven framework & Keyword Driven ?**

Ans- **Datadriven framework-** In this Framework , while Test case logic resides in Test Scripts, the Test Data is separated and kept outside the Test Scripts.Test Data is read from the external files (Excel File) and are loaded into the variables inside the Test Script. Variables are used both for Input values and for Verification values.

**Keyword Driven framework-** The Keyword-Driven or Table-Driven framework requires the development of data tables and keywords, independent of the test automation tool used to execute them . Tests can be designed with or without the Application. In a keyword-driven test, the functionality of the application-under-test is documented in a table as well as in step-by-step instructions for each test.

### What is the alternate way to send text in textbox of webpage with out using sendKeys() method ?

WebDriver driver = new FirefoxDriver();

JavascriptExecutor executor = (JavascriptExecutor)driver;

executor.executeScript("document.getElementById("textbox\_id").value= ”new value”);

### how to count no of links in a page by selenium webdriver java

### public class linktest { WebDriver driver = new FirefoxDriver(); private String baseUrl="https://www.flipkart.com/";   @Test   public void testNoOfLinks() { // Select all the links from webpage using selenium webdriver System.out.println("Select all links from webpage using selenium webdriver"); List<WebElement> links = driver.findElement(By.xpath("//\*")).findElements(By.tagName("a")); System.out.println("total no. links available on webpage"); System.out.println(links.size()); } }

**Ques 11) While explaining the framework, what are points which should be covered ?**

Ans-

a) What is the frame work.

b) Which frame work you are using.

c) Why This Frame work.

d) Architecture.

e) Explanation of every component of frame work.

f) Process followed in frame work.

g) How & when u execute the frame work.

h) Code (u must write code and explain).

i) Result and reporting .

j) You should be able to explain it for 20 Minutes.

# Selenium Webdriver Data-Driven Framework

Selenium Webdriver Data-Driven Framework is where test input and output values are from data files (ODBC sources, CVS files, Excel files, and DAO objects) and are loaded into variables in captured or manually coded scripts.

In this framework, variables are used for both the input values and output verification values. The framework should include navigation through the program, reading of the data files, and logging of test status and information. All the processes should be in the test scripts. The framework should have the following features:

* Well defined architectural design
* Less time to test large data
* Script execution in multiple environments
* Easier, faster, and efficient analysis of result logs
* Communication of results
* Easy debugging and scrip tmaintenance
* Robust and stable due to error and exception handling
* 100% reliability of utility scripts, online execution and report packages.

### Framework Architecture Design

Automation architecture package should include:

* Config - Keeps all the configuration files such as property files
* InputTestData - has files containing input data for application
* OutputData - Contains downloaded documentations, images, fetched data in excel
* TestReports - Contains ANT generated reports
* Util package - Should contain all generic functions & business functions such as email configuration settings and all other utilities
* TestLogs-Contain log file corresponding to tests
* DAO - Classes for accessing persistent storage, such as to a database
* Pages - Page classes for particular pages

|  |  |  |
| --- | --- | --- |
| **Keyword** | **Data** | **Hybrid** |
| Keywords controls everything | Here there is one java file for one TC | Here there are two java files are present. One is Driver script and(**Driver.java**) another is Keyword script(**Keyword.java**) and these two files are responsible to generate the keyword and everything |
|  | In development Data driven framework is preferable to use. |  |
|  | In eclipse heirarchy: One **Config** package is there where 1)**config.properties**(holds the test site name, e.g.(https://google.com) and browser name(Firefox)) , test environment SIT UAT 2) **OR.properties** contains xpath and  3) **TestCase.xlsx** files will be embedded  **Datatable** package where there is **xls** Interclass to read which help to read the excel data file   **TestBase** class and  **Test suit** are present, that hold one java file/TC  One **build.xml** file which helps us to execute the file and hold the path of .jar file and by the help of which we execute the TC  Mailing a result Report is done by **SendMail.java** |  |
|  | For every TC, there is one sheet. |  |

### Reading Font Properties In Selenium WebDriver Using .getCssValue() Method

**//Read font-size property and print It In console.**

String fontSize = text.getCssValue("font-size");

System.out.println("Font Size -> "+fontSize);

**//Read color property and print It In console.**

String fontColor = text.getCssValue("color");

System.out.println("Font Color -> "+fontColor);

**//Read font-family property and print It In console.**

String fontFamily = text.getCssValue("font-family");

System.out.println("Font Family -> "+fontFamily);

**//Read text-align property and print It In console.**

String fonttxtAlign = text.getCssValue("text-align");

System.out.println("Font Text Alignment -> "+fonttxtAlign);

**// to find the background color**

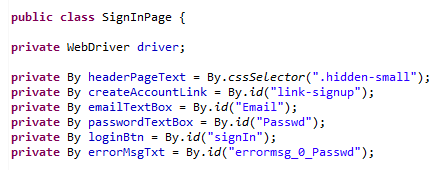
String bg\_color = link\_Home.getCssValue("background-color");

System.***out***.println("The background color is : "+bg\_color);

# *Page Object Model Framework*

Page Object Model Framework has now a days become very popular test automation framework in the industry and many companies are using it because of its easy test maintenance and reduces the duplication of code.

The main advantage of Page Object Model is that if the UI changes for any page, it don’t require us to change any tests, we just need to change only the code within the page objects (Only at one place). Many other tools which are using selenium, are following the page object model.



In the above screen shot, we have first identified the locators and defined it on the top after the class. In this way we can achieve readability of test scripts and we can easily identify locators and change them if needed at only one place.

Page Object model is writing all the functionalities / reusable components of a page that we want to automate in a separate class. Say now if we consider four pages as Home page, Login page, Create Account and Forgot password page etc.

**The Page Object model provides the following advantages.**

1. There is clean separation between test code and page specific code such as locators (or their use if you’re using a UI map) and layout.

2. There is single repository for the services or operations offered by the page rather than having these services scattered through out the tests.

**Example: Lets us take a simple login example:**

/\*\*\*

  \* Tests login functionality

  \*/

**public** **void** **loginTestCase**() {

       driver.navigate().to(URL);

       driver.findElement(By.name("signIn")).click();

       driver.findElement(By.id("username")).sendKeys("testuser");

       driver.findElement(By.id("password")).sendKeys("testpassword");

       driver.findElement(By.name("loginbtn")).click();

       WebDriverWait wait = **new** WebDriverWait(driver, 10);

       wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("profile")));

       String Expected=driver.findElement(By.id("message")).getText();

       Assert.assertEquals(Expected, "Welcome");

}

If you observe the above test, there is no separation of test and test locators. If this is the case, in future if the UI changes, it must be changed in multiple places. It will also become difficult to identify where these locators are used as the chances of locators are being used in multiple tests are more.

We will try to rewrite the above example by implementing the page object model:

/\*\*\*

\* Tests login functionality

\*/

**public** **void** **loginTestCase**() {

*// To go to home page*

homePage.gotoHomePage();

*//To click on SignIn link*

accountLoginPage = homePage.clickOnSignIn()

*//To verify if user is navigated to sign-in page*

Assert.assertTrue(accountLoginPage.verifyPage());

*//Login to the account*

accountLoginPage.userLogin(username,password);

*//To verify if user is navigated to user home page after successfull login*

Assert.assertTrue(userHomePage.verifyPage());

}

In the above test, we have not used any locators. It is completely separated by driver.findElement 's, waits, exceptions and no static values in the code etc.We will be working only with the methods which are defined in multiple pages. Based on test, we will navigate to the required page and access those page methods.

It is a mechanism which involves more than one components to work parallel with Each other.

Generally in Test Automation, we have two components  
**1. Application Under Test**  
**2. Test Automation Tool.**

Both these components will have their own speed. We should write our scripts in such a way that both the components should move with same and desired speed, so that we will not encounter "Element Not Found" errors which will consume time again in debugging.

Synchronization can be classified into two categories:

**1. Unconditional**  
**2. Conditional Synchronization**

**Unconditional :**  
In this we just specify timeout value only. We will make the tool to wait until certain amount of time and then proceed further.

Examples: Wait() and Thread.Sleep();

The main disadvantage for the above statements are, there is a chance of unnecessary waiting time even though the application is ready.

The advantages are like in a situation where we interact for third party systems like interfaces, it is not possible to write a condition or check for a condition. Here in this situations, we have to make the application to wait for certain amount of time by specifying the timeout value.

**Conditional Synchronization:**

We specify a condition along with timeout value, so that tool waits to check for the condition and then come out if nothing happens.

It is very important to set the timeout value in conditional synchronization, because the tool should proceed further instead of making the tool to wait for a particular condition to satisfy.

In Selenium we have implicit Wait and Explicit Wait conditional statements. Check here for [Examples on how to use Webdriver Waits](http://seleniumeasy.com/selenium-tutorials/webdriver-wait-examples)

**1. Implicit Wait.**

An implicit wait is to tell WebDriver to poll the DOM for a certain amount of time when trying to find an element or elements if they are not immediately available.

The default setting is 0. Once when we define the implicit wait, it will set for the life of the WebDriver object instance.

It is a mechanism which will be written once and applied for entire session automatically. It should be applied immediately once we initiate the Webdriver.

Implicit wait will not work all the commands/statements in the application. It will work only for "FindElement" and "FindElements" statements.

If we set implicit wait, find element will not throw an exception if the element is not found in first instance, instead it will poll for the element until the timeout and then proceeds further. We should always remember to add the below syntax immediately below the Webdriver statement.

Syntax: driver.manage.TimeOuts.implicitwait(6,Timeunit.SECONDS);

**Example:  
WebDriver driver = new FirefoxDriver();  
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);  
driver.get("**[**www.google.com**](http://www.google.com/)**");**

**Explicit Wait:**

We need to define a wait statement for certain condition to be satisfied until the specified timeout period. If the Webdriver finds the element within the timeout period the code will get executed.

Explicit wait is mostly used when we need to Wait for a specific content/attribute change after performing any action, like when application gives AJAX call to system and get dynamic data and render on UI.

Example: Like there are drop-downs Country and State, based on the country value selected, the values in the state drop-down will change, which will take few seconds of time to get the data based on user selection.

Example:

/\*Explicit **wait** **for** **state** dropdown field\*/

    WebDriverWait **wait** = new WebDriverWait(driver, 10);

**wait**.**until**(ExpectedConditions.visibilityOfElementLocated(By.id("statedropdown")));

**Fluent Wait:**

Using FluentWait we can define the maximum amount of time to wait for a condition, as well as the frequency with which to check for the condition.

And also the user can configure to ignore specific types of exceptions such as "NoSuchElementExceptions" when searching for an element.

Syntax:

Wait<WebDriver> wait = new FluentWait<WebDriver>(driver)

//Wait for the condition

       .withTimeout(30, TimeUnit.SECONDS)

         // which to check for the condition with interval of 5 seconds.

       .pollingEvery(5, TimeUnit.SECONDS)

     //Which will ignore the NoSuchElementException

       .ignoring(NoSuchElementException.class);

Hexaware:

What is retrospective meeting:

# Agile retrospective definition

This definition is part of our Essential Guide: [Next generation Agile: Guide to continuous development](http://searchsoftwarequality.techtarget.com/essentialguide/Next-generation-Agile-Guide-to-continuous-development)

An Agile retrospective is a meeting that's held at the end of an [iteration](http://searchsoftwarequality.techtarget.com/definition/iteration) in Agile software development ([ASD](http://searchsoftwarequality.techtarget.com/definition/agile-software-development) ). During the retrospective, the team reflects on what happened in the iteration and identifies actions for improvement going forward.

Each member of the team members answers the following questions:

* What worked well for us?
* What did not work well for us?
* What actions can we take to improve our process going forward?

The Agile retrospective can be thought of as a "lessons learned" meeting. The team reflects on how everything went and then decides what changes they want to make in the next iteration. The retrospective is team-driven, and team members should decide together how the meetings will be run and how decisions will be made about improvements.

How to handle popups in selenium web driver?

**public** **static** **void** closePopupIfPresent(){

Set<String> winIds = *driver*.getWindowHandles();

System.***out***.println("Total windows -> "+ winIds.size());

**if**(winIds.size() == 2){

Iterator<String> iter = winIds.iterator();

String mainWinID = iter.next();

String popupWinID = iter.next();

*driver*.switchTo().window(popupWinID);

*driver*.close();

*driver*.switchTo().window(mainWinID);

}

String and StringBuffer:

The **toString()** method of **StringBuffer** class can be used to convert StringBuffer content to a String. This method returns a String object that represents the contents of StringBuffer.  
Method:

public class ConvertStringBufferToString1 {

public static void main(String[] args) {

StringBuffer sb = new StringBuffer("beginnersbook");

String str = sb.toString();

System.out.println("String object: "+str);

}

}

Reference Variable:

The only way you can access an object is through a reference variable. A reference variable is declared to be of a specific type and that type can never be changed. Reference variables can be declared as static variables, instance variables, method parameters, or local variables

## **Types of reference variables**

* Static Variable
* Instance Variable
* Method Parameter
* Local Variable

Explain what is continuous integration?

# What is Jenkins?

Jenkins is an [award-winning](https://wiki.jenkins-ci.org/display/JENKINS/Awards), cross-platform, **continuous integration and continuous delivery** application that increases your productivity. Use Jenkins to **build and test your software projects continuously** making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. It also allows you to **continuously deliver** your software by providing powerful ways to define your build pipelines and integrating with a large number of testing and deployment technologies.

# Features

Jenkins offers the following major features out of the box, and many more can be added through [plugins](https://wiki.jenkins-ci.org/display/JENKINS/Plugins):

1. **Easy installation**: Just run java -jar jenkins.war, deploy it in a servlet container. No additional install, no database. Prefer an installer or native package? We have those as well.
2. **Easy configuration**: Jenkins can be configured entirely from its friendly web GUI with extensive on-the-fly error checks and inline help.
3. **Rich plugin ecosystem**: Jenkins integrates with virtually every SCM or build tool that exists. [View plugins](https://wiki.jenkins-ci.org/display/JENKINS/Plugins).
4. **Extensibility**: Most parts of Jenkins can be extended and modified, and it's easy to create new Jenkins plugins. This allows you to customize Jenkins to your needs.
5. **Distributed builds**: Jenkins can distribute build/test loads to multiple computers with different operating systems. Building software for OS X, Linux, and Windows? No problem

13. What is self join?

## Self Join Example in SQL

In order to solve this query let's first see schema and data of Employee table.

mysql> **select** \* **from** employee;  
+--------+----------+---------+--------+--------+  
| emp\_id | emp\_name | dept\_id | salary | mgr\_id |  
+--------+----------+---------+--------+--------+  
|    103 | Jack     |       2 |   1400 |    104 |  
|    104 | John     |       2 |   1450 |    104 |  
|    105 | Johnny   |       3 |   1050 |    104 |  
|    108 | Alan     |       3 |   1150 |    104 |  
|    106 | Virat    |       4 |    850 |    105 |  
|    107 | Vina     |       4 |    700 |    105 |  
|    109 | joya     |       4 |    700 |    105 |  
+--------+----------+---------+--------+--------+  
7 rows **in** **set** (0.00 sec)

In above  table all employees who are managers has there emp\_id as mgr\_id in other employees and by using SELF JOIN  i.e. join two instances of employee table and comparing, we can find all employees who are managers. Here is the [SELECT query example](http://javarevisited.blogspot.com/2011/10/selct-command-sql-query-example.html) using self-join:

mysql> **select** **distinct** e.emp\_id, e.emp\_name **from** employee e **join** employee m **on** e.emp\_id=m.mgr\_id;  
+--------+----------+  
| emp\_id | emp\_name |  
+--------+----------+  
|    104 | John     |  
|    105 | Johnny   |  
+--------+----------+  
2 rows **in** **set** (0.00 sec)

Cucumber BDD:

**Cucumber Introduction:**

Cucumber is a tool based on Behavior Driven Development (BDD) framework which is used to write acceptance tests for web application. It allows automation of functional validation in easily readable and understandable format (like plain English) to Business Analysts, Developers, Testers, etc.

Cucumber feature files can serve as a good document for all. There are many other tools like JBehave which also support BDD framework. Initially Cucumber was implemented in Ruby and then extended to Java framework. Both the tools support native JUnit.

Behavior Driven Development is extension of Test Driven Development and it is used to test the system rather than testing the particular piece of code. We will discuss more about the BDD and style of writing BDD tests.

Cucumber can be used along with Selenium, Watir, and Capybara etc. Cucumber supports many other languages like Perl, PHP, Python, .Net etc. In this tutorial we will concentrate on Cucumber with Java as a language.

**Cucumber Basics:**

In order to understand cucumber we need to know all the features of cucumber and its usage.

**#1) Feature Files:**

Feature files are essential part of cucumber which is used to write test automation steps or acceptance tests. This can be used as live document. The steps are the application specification. All the feature files ends with .feature extension.

**Sample feature file:**

**Feature**: Login Functionality Feature

In order to ensure Login Functionality works,  
I want to run the cucumber test to verify it is working

**Scenario**: Login Functionality

**Given** user navigates to SOFTWARETETINGHELP.COM  
**When** user logs in using Username as “USER” and Password “PASSWORD”  
**Then** login should be successful

**Scenario**: Login Functionality

**Given** user navigates to SOFTWARETETINGHELP.COM  
**When** user logs in using Username as “USER1” and Password “PASSWORD1”  
**Then** error message should be thrown

**#2) Feature:**

**T**his gives information about the high level business functionality (Refer to previous example) and the purpose of Application under test. Everybody should be able to understand the intent of feature file by reading the first Feature step. This part is basically kept brief.

**#3) Scenario:**

Basically a scenario represents a particular functionality which is under test. By seeing the scenario user should be able to understand the intent behind the scenario and what the test is all about. Each scenario should follow given, when and then format. This language is called as “gherkin”.

1. **Given:**As mentioned above, given specifies the pre-conditions. It is basically a known state.
2. **When**: This is used when some action is to be performed. As in above example we have seen when the user tries to log in using username and password, it becomes an action**.**
3. **Then:**The expected outcome or result should be placed here. For Instance: verify the login is successful, successful page navigation.
4. **Background:**Whenever any step is required to perform in each scenario then those steps needs to be placed in Background. For Instance: If user needs to clear database before each scenario then those steps can be put in background.
5. **And**: And is used to combine two or more same type of action.

**Example:**

**Feature**: Login Functionality Feature

**Scenario**: Login Functionality  
**Given** user navigates to SOFTWARETETINGHELP.COM  
**When** user logs in using Username as “USER”  
**And**password as “password”  
**Then** login should be successful  
**And** Home page should be displayed

**Example of Background:**

**Background:**

**Given** user logged in as databases administrator  
**And** all the junk values are cleared

**#4) Scenario Outline:**

Scenario outlines are used when same test has to be performed with different data set. Let’s take the same example. We have to test login functionality with multiple different set of username and password.

**Feature**: Login Functionality Feature

In order to ensure Login Functionality works,  
I want to run the cucumber test to verify it is working

**Scenario Outline**: Login Functionality

**Given** user navigates to SOFTWARETESTINGHELP.COM  
**When** user logs in using Username as <**username**> and Password <**password**>  
**Then** login should be successful

**Examples:**  
|username         |password          |  
|Tom                     |password1        |  
|Harry                   |password2        |  
|Jerry                    |password3        |

**Note:**

1. As shown in above example column names are passed as parameter to **When** statement.
2. In place of Scenario, you have to use Scenario Outline.
3. Examples are used to pass different arguments in tabular format. Vertical pipes are used to separate two different columns. Example can contain many different columns.

**#5) Tags:**

Cucumber by default runs all scenarios in all the feature files. In real time projects there could be hundreds of feature file which are not required to run at all times.

**For instance**: Feature files related to smoke test need not run all the time. So if you mention a tag as smokeTest in each feature file which is related to smoke test and run cucumber test with @SmokeTest tag . Cucumber will run only those feature files specific to given tags. Please follow the below example. You can specify multiple tags in one feature file.

**Example of use of single tags:**

**@SmokeTest**

**Feature**: Login Functionality Feature

In order to ensure Login Functionality works,  
I want to run the cucumber test to verify it is working

**Scenario Outline**: Login Functionality

**Given** user navigates to SOFTWARETESTINGHELP.COM  
**When** user logs in using Username as <**username**> and Password <**password**>  
**Then** login should be successful

**Examples:**  
|username         |password          |  
|Tom     |password1        |  
|Harry   |password2        |  
|Jerry    |password3        |

Different phases of agile testing

## The Phases of Agile Testing

### 1.0 Unit Testing

The developers do the unit testing of their code. Think of this as being an initial test to see whether the code works at a high level. These tests should be automated as far as possible to reduce the testing time.

### 2.0 Integration Testing

Integration testing and system testing start together. As soon as the second sprint is over, the system and integration testing of that sprint, together with the other ones start.

### 3.0 Regression Testing

After each period of integration testing there is a short cycle of regression testing. As the development is automated test driven, the system should be working well at this stage and hence there would be no need for a long period of regression testing.

Please click here for the remainder of the Phases of [Agile Testing](http://www.my-project-management-expert.com/agile-testing-1.html) as well as the Key Points and Values of Agile Testing.

What is abstraction?

**Abstraction** is a process of hiding the implementation details and showing only functionality to the user.

**Ways to achieve Abstaction**

There are two ways to achieve abstraction in java

1. Abstract class (0 to 100%)
2. Interface (100%)

# Difference between HashMap and Hashtable

HashMap and Hashtable both are used to store data in key and value form. Both are using hashing technique to store unique keys.

But there are many differences between HashMap and Hashtable classes that are given below.

|  |  |
| --- | --- |
| **HashMap** | **Hashtable** |
| 1) HashMap is **non synchronized**. It is not-thread safe and can't be shared between many threads without proper synchronization code. | Hashtable is **synchronized**. It is thread-safe and can be shared with many threads. |
| 2) HashMap **allows one null key and multiple null values**. | Hashtable **doesn't allow any null key or value**. |
| 3) HashMap is a **new class introduced in JDK 1.2**. | Hashtable is a **legacy class**. |
| 4) HashMap is **fast**. | Hashtable is **slow**. |
| 5) We can make the HashMap as synchronized by calling this code Map m = Collections.synchronizedMap(hashMap); | Hashtable is internally synchronized and can't be unsynchronized. |
| 6) HashMap is **traversed by Iterator**. | Hashtable is **traversed by Enumerator and Iterator**. |
| 7) Iterator in HashMap is **fail-fast**. | Enumerator in Hashtable is **not fail-fast**. |
| 8) HashMap inherits **AbstractMap** class. | Hashtable inherits **Dictionary** class. |

# How Java Garbage Collection Works?

Java garbage collection is an automatic process to manage the runtime memory used by programs. By doing it automatic JVM relieves the programmer of the overhead of assigning and freeing up memory resources in a program.

## **Java Garbage Collection GC Initiation**

Being an automatic process, programmers need not initiate the garbage collection process explicitly in the code. System.gc() andRuntime.gc() are hooks to request the JVM to initiate the garbage collection process.

* 1. **Difference between Private Public Protected**
* Methods declared public in a superclass also must be public in all subclasses.
* Methods declared protected in a superclass must either be protected or public in subclasses; they cannot be private.
* Methods declared private are not inherited at all, so there is no rule for them.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Access Modifier** | **within class** | **within package** | **outside package by subclass only** | **outside package** |
| **Private** | Y | N | N | N |
| **Default** | Y | Y | N | N |
| **Protected** | Y | Y | Y | N |
| **Public** | Y | Y | Y | Y |

2) Write a program for :

\*

\*\*

\*\*\*

\*\*\*\*

3) Difference between Overriding and Overloading

4)

# Difference between final, finally and finalize

There are many differences between final, finally and finalize. A list of differences between final, finally and finalize are given below:

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Final** | **finally** | **finalize** |
| 1) | Final is used to apply restrictions on class, method and variable. Final class can't be inherited, final method can't be overridden and final variable value can't be changed. | Finally is used to place important code, it will be executed whether exception is handled or not. | Finalize is used to perform clean up processing just before object is garbage collected. |
| 2) | Final is a keyword. | Finally is a block. | Finalize is a method. |

## **Java final example**

1. **class** FinalExample{
2. **public** **static** **void** main(String[] args){
3. **final** **int** x=100;
4. x=200;//Compile Time Error
5. }}

## **Java finally example**

1. **class** FinallyExample{
2. **public** **static** **void** main(String[] args){
3. **try**{
4. **int** x=300;
5. }**catch**(Exception e){System.out.println(e);}
6. **finally**{System.out.println("finally block is executed");}
7. }}

## **Java finalize example**

1. **class** FinalizeExample{
2. **public** **void** finalize(){System.out.println("finalize called");}
3. **public** **static** **void** main(String[] args){
4. FinalizeExample f1=**new** FinalizeExample();
5. FinalizeExample f2=**new** FinalizeExample();
6. f1=**null**;
7. f2=**null**;
8. System.gc();
9. }}

**Findelements to click multiple links in a page**

WebElement ParentElement= Driver.findElement(By.cssSelector(OR.getProperty(Object)));

List<WebElement> schedule= ParentElement.findElements(By.tagName("a"));

for(int i=1;i<schedule.size();i++){

schedule.get(i).click();

Thread.sleep(5000L);

}

**Clicking 3rd button:**

List<WebElement> a=driver.findElements(By.cssSelector("button[@class='button' and @type = 'button']"));

// a.get(0).click();

// a.get(1).click();

// a.get(2).click();

a.get(3).click();

# Why do we use Desired Capabilities in Selenium WebDriver

The Desired Capabilities Class helps us to tell the webdriver, which environment we are going to use in our test script.

## **What is Desired Capability**

The desired capability is a series of key/value pairs that stores the browser properties like browsername, browser version, the path of the browser driver in the system, etc. to determine the behaviour of the browser at run time.

* Desired capability can also be used to configure the driver instance of Selenium WebDriver.
* We can configure driver instance like FirefoxDriver, ChromeDriver, InternetExplorerDriver by using desired capabilities.

Desired Capabilities are more useful in cases like:

* In mobile application automation, where the browser properties and the device properties can be set.
* In Selenium grid when we want to run the test cases on a different browser with different operating systems and versions.

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

There are different types of desired capabilities methods and some of them are as follows.

* getBrowserName()
* setBrowserName()
* getVersion()
* setVersion()
* getPlatform()
* setPlatform()
* getCapability Method
* setCapabilityMethod

**dataProvider() annotation in TestNG**

public class SendMailTest {

// to

// subject

// mail body

@Test(dataProvider="getData")

public void testMailSending(String to , Integer sub, String msg){

System.out.println(to +" --- " + sub + " --- " + msg);

}

@DataProvider

public Object[][] getData(){

// Selenium framework - read the data from xls file and put it in Object array

// rows - number of time test has to be repeated

// cols - number of parameters in test data

Object[][] data = new Object[2][3];

// 1st row

data[0][0] = "xyz@gmail.com";

data[0][1] = 123;

data[0][2] = "howzz life";

// 2nd row

data[1][0] = "abc@gmail.com";

data[1][1] = 444;

data[1][2] = "life is good";

return data;

}

}

**2) What are the 2 files required to execute a Cucumber test scenario?**

The 2 files required to execute a Cucumber test scenario are

* Features
* Step Definition

**3) What is feature file in Cucumber? What does feature file consist of ?**

Feature file in cucumber consist of parameters or conditions required for executing code, they are

* Feature
* Scenario
* Scenario Outline
* Given
* When
* Then

**4) Give an example of behaviour driven test in plain text?**

* **Feature:**Visit **XYZ** page in abc.com
* **Scenario :**Visit abc.com
* **Given:**  I am on abc.com
* **When:**I click on XYZ page
* **Then:**I should see ABC page

**5) Explain what is Scenario Outline in feature file?**

**Scenario Outline**:  Same scenario can be executed for multiple sets of data using scenario outline.  The data is provided by a tabular structure separated by (I   I).

**6) What is step definition in Cucumber?**

A step definition is the actual code implementation of the feature mentioned in feature file.

**TestNG Annotations**

* @Test :  To mark a method as a test method
* @BeforeMethod: Executes before each test (@Test) method
* @AfterMethod: Executes after each test method
* @BeforeClass: Executes before the first test method in the current class
* @AfterClass: Executes after all the test methods in a current class
* @BeforeTest: Executes before any test methods of available classes which are inside the <test> tag in the testng.xml in this suite
* @AfterTest: Executes after all the test methods of available classes which are inside the <test> tag in the testng.xml in this suite
* @BeforeSuite: Executes before all the tests in this suite
* @AfterSuite: Executes after all tests executed in this current suite
* @BeforeGroup: Executes before the first test method run in that specified group
* @AfterGroup: Executes after the end of all the test methods executed in that specified group

TestNG have more annotations like @BeforeGroup, @AfterGroup, @Listeners, @Parameters, @DataProvider, @Factory.  These will be covered in next posts.

**Sequence of Execution of the annotations:**

1. BeforeSuite
2. BeforeTest
3. BeforeClass
4. BeforeMethod
5. Test
6. AfterMethod
7. AfterClass
8. AfterTest
9. AfterSuite