

*NEW*

# ***SELENIUM***

## ***NOTES***

**MR.SRIKANTH**

# ***SATYA TECHNOLOGIES***

**SAI RAM XEROX**  
**AMEERPET;HYD**  
**9885440924**



## Ways of Testing:

↳ To test any application we have two ways of test process.

- i) Manual Testing
- ii) Automation Testing

### i) Manual Testing:

↳ Manual Testing is a process in which all the phases of software testing life cycle (STLC) like Test planning, test Design, test Execution result Analysis, bug tracking and reporting are completed successfully with human efforts.

Drawbacks :-

- 1) Time Consuming
- 2) Parallel almost all not possible
- 3) Testing the functionality again and again is not possible with the same interest.
- 4) No accuracy
- 5) tidiness

### ii) Automation Testing:

↳ Automation Testing is a process in which all the drawbacks of manual Testing is solved and provide the speed and accuracy in order to implements testing by using tool.

Drawbacks :-

- 1) All the Areas we can't recommended to the Automation Testing.  
    ① ~~which~~ Test Cases we can Convert to the Automation?
  - Re Testing Test Case
  - Regression Test Cases..

Q) Which Test Cases Cannot Convert into Automation?

→ Whenever application dynamically changes

→ when Event manual interaction is more required.

- |                     |                       |                        |
|---------------------|-----------------------|------------------------|
| (i) Functional Tool | (ii) Performance Tool | (iii) Management Tools |
| - Selenium          | - Load Runner (LR)    | - Quality Center (QC)  |
| - QTP/UFT           | - JMeter              |                        |
| - RFT               | - Silk                |                        |
- 

### Selenium:

Selenium is a open source tool, which is used for implementing the functional testing (or) web based application not for windows based application.

- It will support multiple language
- It will support multiple browsers
- It will support different operating system.

In Selenium, there is no installation just download the jar files which is available in website and configure them into our project in order to test our application.

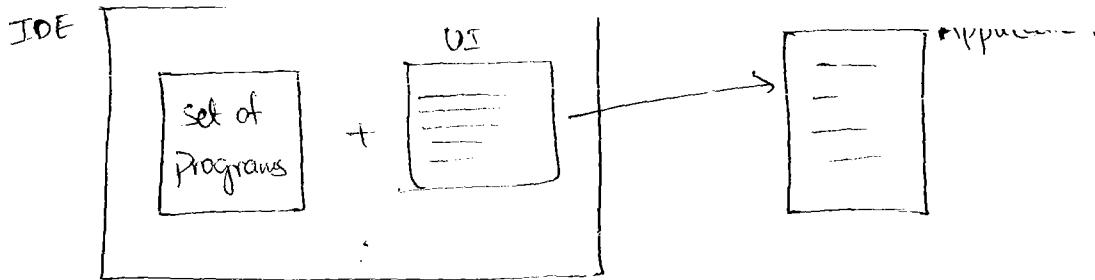
### Components of Selenium :

- > Selenium IDE
- > Selenium - RC / 1.0
- > Selenium WebDriver / 2.0
- > Selenium Grid
- > Selenium Appium / 3.0

### Selenium IDE :

It is a record and playback tool with user interface. It works only in Firefox browser not for other browsers.

- Selenium IDE is a combination of set of programs plus user interface.



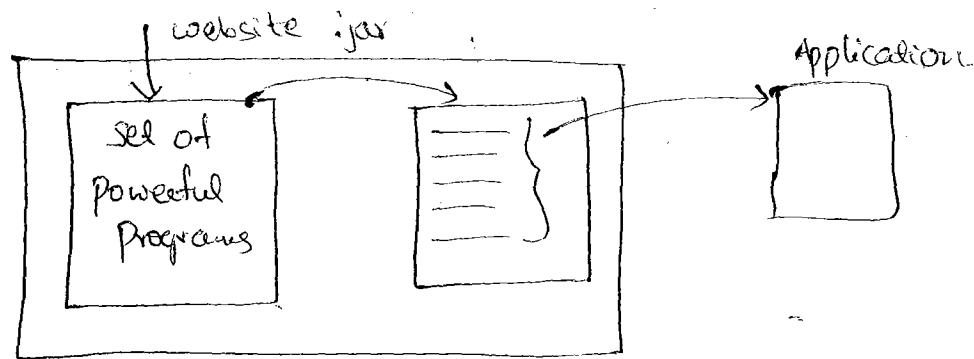
### Selenium RC (Remote Control) :

It is a old version of selenium which is used for implementing the functional testing on browser / web based application. It is developed based on Client Server Architecture in Selenium -RC, if you want to work.

### Selenium WebDriver 2.0 :

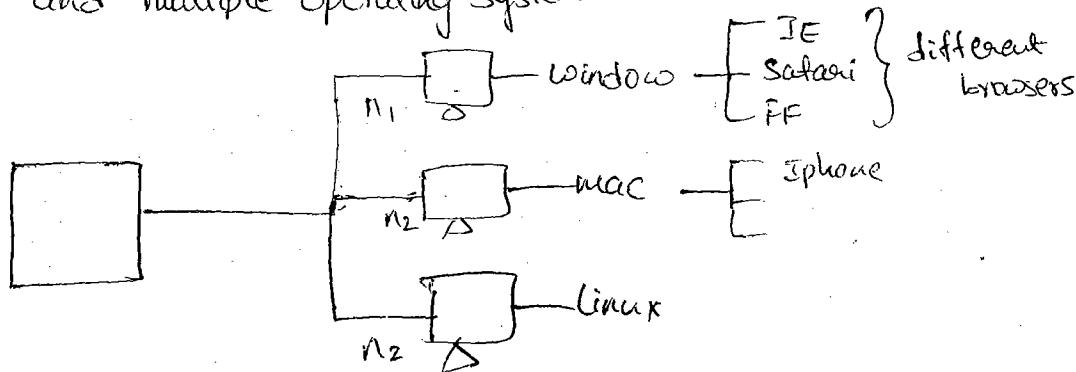
It is a Advanced version for the selenium (more advanced compare to selenium RC).

In Selenium webdriver, just download the jar files from the website and configure to the Eclipse in order to write the manual test Case to Automation Scripts.



### Selenium Grid :

It is used for executing the out test case parallel into multiple browsers and multiple operating System.



Android zipppum

It is used for automating the mobile applications.

29/8/2016

### Navigation for downloading and installing the JDK:

Step 1 :- Open the Google, Type the free download JDK.

Step 2 :- Click on the first link and select the licence agreement.

Step 3 :- Click on the required jdk .exe file and save it in your required loc.

Step 4 :- Double click on the .exe file and click on the next, next & finish, then it is showing successful message.

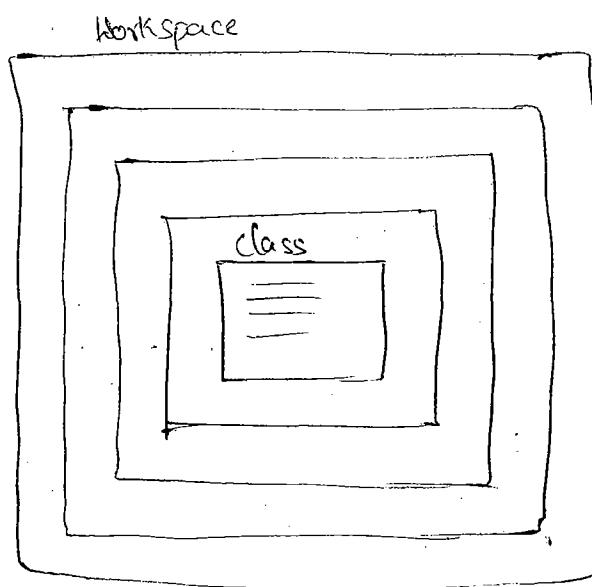
### Navigation for downloading the Eclipse :

Step 1 :- Open the Google, Type free download Eclipse and click on 1st link.

Step 2 :- Click on Luna package link under releases.

Step 3 :- Download the required Eclipse. Under Eclipse IDE for Java EE Developers, and click on download button and Save it in your pc.

### Navigation for creating work space :



Step 1 :- Extract the downloaded Eclipse

Step 2 :- Enter the workspace

Step 3 :- Click OK

Step 4 :- Then close the welcome screen.

Navigation for Creating the Project:

Step 1 :- Right click on the project

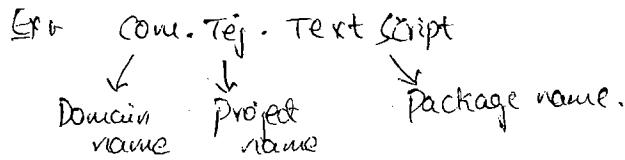
Step 2 :- Select New, click on Java Project

Step 3 :- Enter the name, click on finish

Navigation for Creating the Package:

Step 1 :- Right click on the "src" file, Select the "New", click on package.

Step 2 :- Enter the package name.



Step 3 :- Click on finish.

Navigation for Creating the class:

Step 1 :- Right click on the package, select the new, click on the class. Enter the name.

Step 2 :- Select the Public static void main()

Step 3 :- Click on finish.

```
Eg:- package com.Tejaswini.Carejava;  
public class SampleClass  
{  
    static String name = "Mahesh";  
    public static void main(String[] args)  
    {  
        System.out.println("my name is "+name);  
        System.out.println("Hi");  
    }  
}
```

27/10/2019

## Access Modifiers :

Public is used for within the project anywhere.

Private is used within the class only.

Default is used within the package only.

Protected is used in the package (all classes) and child classes in other packages also.

Ex : public String;  
private String;  
String; // default  
protected String;

## Data Types :

Byte is used to store the 8 bit ( $-2^7$  to  $+2^7$ )

byte a = 127;  
S.O.P(a);

\* Systo() → (short key for S.O.P)  
↓  
Ctrl + Space

Short is used to store the 16 bit ( $-2^{15}$  to  $2^{15}$ )

Short s = 12400;  
S.O.P(s);

Int is used to store 32 bit ( $-2^{31}$  to  $2^{31}$ )

int i = 12000000;  
S.O.P(i);

Long is used to store 64 bit ( $-2^{63}$  to  $2^{63}$ )

long l = 1222222222;  
S.O.P(l);

Float is used to store decimal values and is of 32 bit.

float f = 12.3463789f;  
S.O.P(f); // 12.346379.

double is used to store decimal values and is of 64 bit

~~float f=12.34;~~

double d = 12.34637898723d;

S.O.P(d); // 12.34637898723

boolean is used to store true/false

boolean b=true;

S.O.P(b); // true

boolean b=false;

S.O.P(b); // false.

char is used to store Numeric, Special chars and Alphabets of Only 1 char.

char c='A'; | char c='1'; | char c='&';  
S.O.P(c); // A | S.O.P(c); // 1 | S.O.P(c); // &

String is used to store bunch of chars and is not a datatype.

↓  
is a class.

String str="My name is Teju";

S.O.P(str);

## OBJECT ORIENTED PROGRAMMING LANGUAGE :-

> Any object in this world belongs to some class.

> Every object will have a set of properties and it can do some actions

> Properties are divided into 2 types (Variables)

1) Static Properties

2) Non-Static Properties

> Actions are divided into 2 types (Methods)

1) Static Action

2) Non-Static Action.

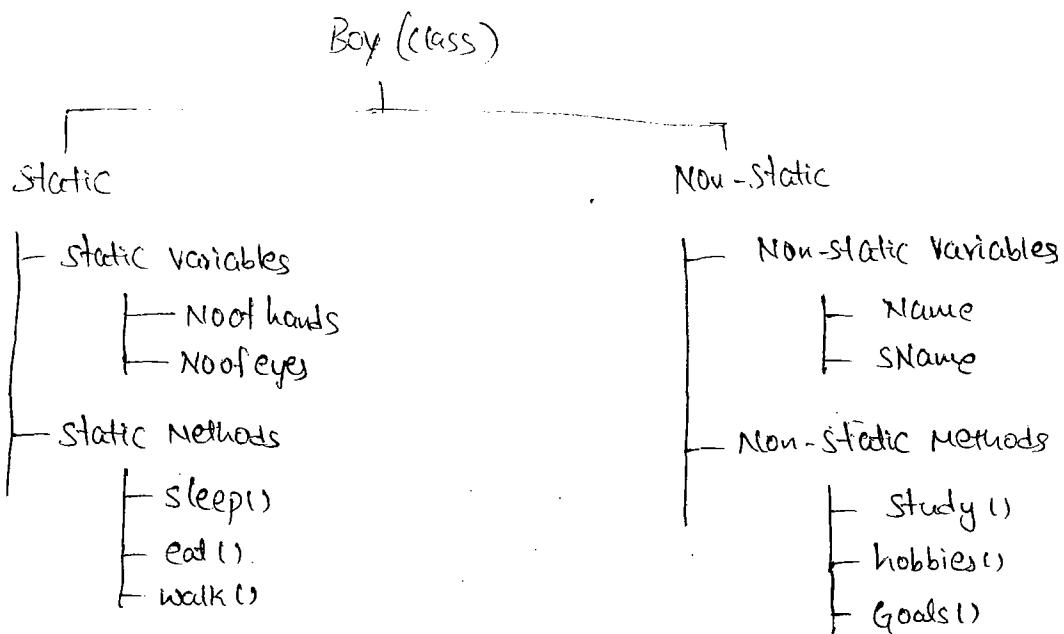
Static Properties :- If at all - the properties are common to all objects in that class, then we can call them as static properties.

Non-Static Properties :- If at all the properties are changing from object to

Some Actions (Static Functions) :- If all the actions are commonly done by all the objects in that class then we call them as static functions.

Non-Static Actions (Non-Static Functions) :- If all the objects are doing the actions which are changing from object to object in that class then we call them as non-static functions.

3/8/2016



// program

⇒ package com.Tejaswini.Corejava;

public class Boy

{

// Static Variables

public static int No.of.hands;

public static int No.of.eyes;

// Non-Static Variables

public String Name;

public String SName;

// Static Methods

public static void sleep()

{

S.O.P("Logic for sleep method");

public static void eat()

{

S.O.P("Logic for eat Method");

}

// Non-Static Methods

public void study()

{

S.O.P("Logic for Studying");

}

public void hobbies()

{

S.O.P("Logic for hobbies");

}

```

⇒ public class useBoy
{
    public static void main(String[] args)
    {
        // Accessing static info
        Boy.noOfhands = 2;
        Boy.noOfEyes = 2;
        System.out.println("No of hands: " + Boy.noOfhands);
        Boy.sleep();
        // Non-static info
        Boy b = new Boy();
        b.name = "Tejaswini";
        b.sname = "Adabala";
        System.out.println("Name is: " + b.name + b.sname);
        Boy b1 = new Boy();
        b1.name = "Sreekanth";
        b1.sname = "malagurla";
        System.out.println("Name is: " + b1.name);
        b1.study();
    }
}

```

⇒ This is different class  
Created under package.

> Accessing Static Info from one class to other class is done through "classname".

Syntax:

LCN > <VN|NN>;

Ex: Boy.noOfhands = 2;      // Variable  
          Boy.sleep();        // Method

Classname creates  
a new memory location.

→ Accessing non-static into from one class to other class is done through "reference".

Syntax:

```
<CN> ref = new <CN>();
```

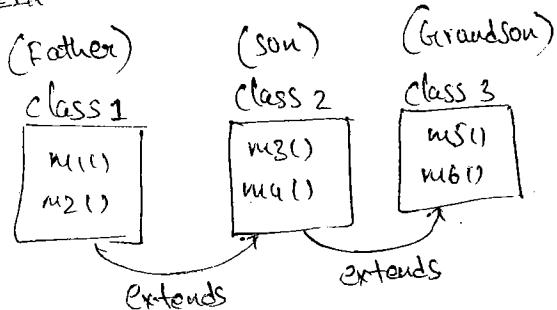
ref acts as a interface ~~to~~

```
Ex: Boy b = new Boy();  
    b.sleep();
```

Inheritance:

Inheritance is a concept provided in java, for extending the parent class properties to the child class.

Ex:



Class 4

```
Class3 c3 = new Class3();  
c3.m1();  
c3.m2();  
c3.m3();  
c3.m4();  
c3.m5();  
c3.m6();
```

By using  
"Extends"  
keyword.

// Example program under package - For inheritance.

(i) public class Father

{

  public void M1()

{

    S.O.P("Logic for M1");

}

  public void M2()

{

    S.O.P("Logic for M2");

}

}

(ii) public class Son Extends Father

{

  public void M3()

{

    S.O.P("Logic for M3");

}

  public void M4()

{

    S.O.P("Logic for M4");

}

}

(iii) public class Grandson extends Son.

```
{  
    public void M5()  
    {  
        S.O.p("Logic for M5");  
    }  
    public void M6()  
    {  
        S.O.p("Logic for M6");  
    }  
}
```

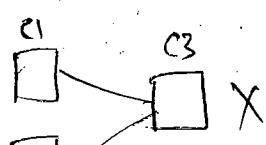
// Accessing Optimization by using "extends" keyword / Inheritance Concept:

```
package com.Tejaswini.Corejava;  
public class Inherited class  
{  
    public static void main (String[] args)  
    {  
        Grandson gs = new Grandson();  
        gs.M1();  
        gs.M2();  
        gs.M3();  
        gs.M4();  
        gs.M5();  
        gs.M6();  
    }  
}
```

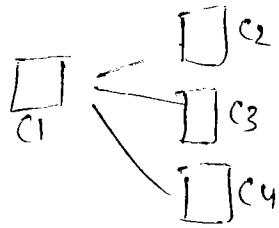
Output Logic for M1  
Logic for M2  
" " " M3  
" " " M4  
" " " M5  
" " " M6

#### NOTE :

> Multiple inheritance is not possible but we can solve this problem by using "interface" concept.



- single inheritance (One-one) & One-Many is possible.



11/9/2016

### Method / Function :

It is a set of statements which can perform the particular task effectively, efficiently and optimized way.

### Advantages :

- 1) Reusability of code
- 2) Managing very easily.

### Syntax for declaring Method

<Access Modifier> <void/DT> <Method Name>()

{  
Set of stmts;  
}

Ex: public void login()  
{  
    UN;  
    pwd;  
}

In above system, if you mention the void that method doesn't return any value.

In the place of void, if you mention any datatype that datatype value must be returned.

### Types of Method declarations: (4 types)

- i) with parameter, with Return type
- ii) without parameter, without Return type
- (iv) without parameter, with Return type

i) With parameters, with return type.

```
public int add(int x, int y)
{
    int z = x+y;
    return z;
}
```

ii) Without parameter, without Return Type.

```
public void info()
{
    S.O.P("Hi");
    S.O.P("Hello");
}
```

iii) With parameter, without Return type.

```
public void begin(UN, pwd)
{
    — UN;
    — pwd;
    — begin;
}
```

iv) Without Parameters, with Return type

```
public boolean flagStatus()
{
    boolean b=True;
    return b;
}
```

## Overloading:

If at all we have the same function more than one time in a class with different arguments (may be no. of arguments or/ type of arguments) then that concept is known as Overloading.

Ex- public class Overloading

```
{}
public int Add(int a, int b)
```

```

int c = a+b;
return c;
}

public int Add (int a, int b, int c)
{
    int d = a+b+c;
    return d;
}

public static void main (String [] args)
{
    Overloading ov = new Overloading();
    int Res = ov.Add (10,20);
    S.O.P ("Result is " + Res); //30
    Res = ov.Add (10,40,50);
    S.O.P ("Result is " + Res); //100
}

```

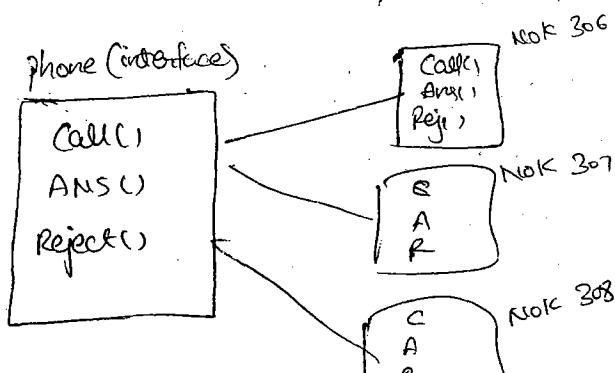
## Overriding:

If at all a function is existing in Parent class and same function with same arguments we write in Child class then that concept is known as Overriding.

## Interface:

It is a set of rules defined by us which has only declaration of functions (No logic/body of the func) and it will force us to override all the functions in the class, which implements this interface.

21/9/2016



Syntax:

Interface ref = new className();

// program.

⇒ public interface phone

{

    public void call();

    public void Ans();

    public void Reject();

}

⇒ public class Nokia306 implements phone.

{

    @Override public void reject()

{

        S.O.P ("logic for NOK 306<sup>reject</sup>");

}

    public void Ans()

{

        S.O.P ("logic for NOK306 Ans");

}

    public void call()

{

        S.O.P ("logic for NOK306 call");

}

⇒ public class Nok307 implements phone

{

    public void reject()

{

        S.O.P ("NOK307 reject logic");

}

    public void Ans()

{

        S.O.P ("NOK307 Ans logic");

}

    public void call()

{

        S.O.P ("NOK307 call logic");

}

To create user class we can use interface class.

public class InterfaceConcept

{

PSVM (String[] args)

{

Phone p = new NOK306();

p.call();

p.reject();

p.Ans();

Phone p1 = new NOK307();

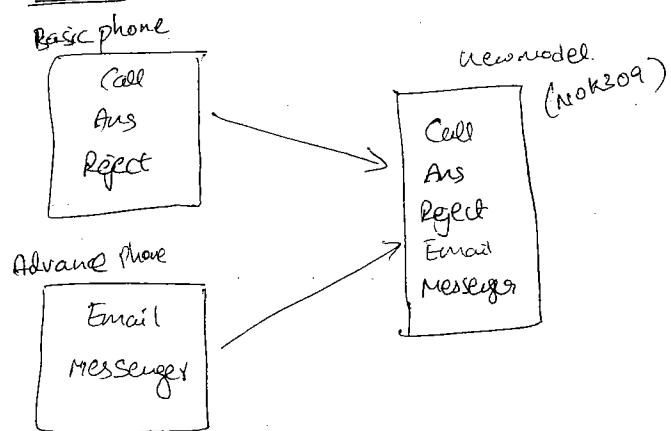
p1.call();

p1.Ans();

p1.reject();

}

### Multi Interface:



Step 1: public interface phone

{

    Public void call();

    Public void Ans();

    Public void Reject();

}

Step 2: public interface AdvancePhone

{

    Public void Email();

    Public void Messenger();

}

Step 3: public class NOK309 implements phone, AdvancePhone

{

    Public void reject();

{

        S.O.P ("Logic for NOK309");

}

    Public void call();

{

```
public void ans()
{
    S.o.p("Logic for Ans");
}
public void Email()
{
    S.o.p("Logic for NOK309 Email");
}
public void Messenger()
{
    S.o.p("Logic for Messenger");
}
```

Step 4: public class MultiInterface

```
{}
public String[] args)
{
    Phone p = new NOK309();
    p.call();
    p.Answer();
    p.reject();
}

Phone AP = new NOK309();
AP.Email();
AP.Messenger();
}
```

Selenium 2.0

J,

selenium 1.0/RC + Selenium 2.0/WP + Grid.

- WEB DRIVE :
- WebDrive is an interface developed by Selenium people which is being implemented by so many classes like WebDriver, InternetExplorerDriver, FirefoxDriver, ChromeDriver etc.
- At present people are using Selenium webdriver for functional Automation because it is so simple and more powerful than old version of Selenium PC. For example, In Selenium PC we need to start and stop the Selenium service through program but no need to do it Selenium webdriver.
- Navigation for Downloading the Selenium jar files:
- Step 1 : Open the browser and type "selenium.org", click go button
- Step 2 : Click on download tab.
- Step 3 : Under Selenium downloads, click on Previous Releases links
- Step 4 : Click on 2.53 version
- Step 5 : Click on "selenium - Service - Standalone - 2.53.1.jar" And save the file to the required location.
- Navigation for Configuring the Selenium jar files into the Project:
- Step 1 : Right click on "Project", click on the "Properties" and select the "Java Build Path".
- Step 2 : Then select "Library" tab and click on Add External Jars.
- Step 2 : Select Required jar file (downloaded) and click open and click OK.
- Step 3 : Navigate for below Manual Test case to Automation Script.
- || WAP for below Manual Test case to Automation Script.
- Step 1 : Right click on "Project", click on the "Properties" and select the "Java Build Path".
- Step 2 : Then select "Library" tab and click on Add External Jars.
- Step 3 : Open the Firefox browser.
- Step 4 : Open the Gmail account.
- Step 5 : Close the browser.

```

→ package com.tejaswini.webdriver;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
public class InvokeFirefoxApp
{
    public static void main(String[] args)
    {
        WebDriver driver = new FirefoxDriver();
        driver.get("http://Gmail.com");
        driver.manage().window().maximize();
        driver.close();
        driver.quit();
    }
}

```

## Working with IE Browser:

Nav is Similar for chrome, safari, Opera etc browsers

### Navigation

- Open "seleniumhq.org" and where you can observe "IE Driver Server" in download page.
- Click on 32-bit / 64-bit windows IE link and save the file (Zip) and Extract.

### Settings

#### Syntax

```
System.setProperty("webdriver.ie.driver", "path of IEDriverServer.exe");
```

### //Program

```

package com.tejaswini.webdriver;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.ie.InternetExplorerDriver;
public class IEBrowser
{
    public static void main(String[] args)
    {
        System.setProperty("webdriver.ie.driver", "E:/selenium_Download/IEDriverServer_X64_2.45.0/
IEDriverServer.exe");
    }
}

```

## Working with Chrome Browser.

```
package com.Tutorial.webdriver;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.chrome.ChromeDriver;  
public class chromeBrowser  
{  
    public static void main(String[] args)  
    {  
        System.setProperty("webdriver.chrome.driver", "E:/selenium/chromedriver/chromedriver.exe");  
        WebDriver driver = new ChromeDriver();  
        driver.get("http://gmail.com");  
    }  
}
```

## Navigation for downloading chromedriver Server.

- i) Open "seleniumhq.org" website , click on download option
- ii) click on GoogleChromeDriver version link and click on "chromedriver-win32.zip" and Save it to the required location.

⇒ To work on the Opera Browser , we can go for as below

```
import org.openqa.selenium.Opera.OperaDriver;  
WebDriver driver = new OperaDriver();  
driver.get("http://gmail.com");
```

⇒ To work with Safari browser ,

```
import org.openqa.selenium.Safari.SafariDriver;  
WebDriver driver = new SafariDriver();  
driver.get("http://gmail.com");
```

## Object Operational

### Overview of the Selenium:

While giving the instructions to Selenium we need to provide the information (properties of objects) to identify the object and also what operation to be performed on that object.

During execution Selenium will work in the following way.

- It will first identify the object using the information, which is provided by in the program.
- It will perform the action that is described in the instruction.

### Object Identification:

In Selenium, we can identify the object in the web application with the help of properties like id, name, linkname, class, Xpath, tagname etc.

- Install Firebug
  - Install Firepath
- } Add-ons for  
Firefox.

Firebug :- is a add-on of Firefox which is used for inspecting the web elements in the web application and knowing the info about them.

### Navigation for installing the Firebug:

Step 1 :- Open Google and type "Firebug"

Step 2 :- Click on the first link.

Step 3 :- Click on Add to Firefox button.

Step 4 :- Click on Allow button.

Step 5 :- Click on Install Now button.

Step 6 :- Click on the Restart Now button.

Firepath :- is a add-on of Firefox which is used for specially showing the xpath of a web element.

## Navigation for installing the firepath:

- Step 1 : Open "Google" and type "firepath".
- Step 2 : Click on the first link.
- Step 3 : Click on the add to Firefox button.
- Step 4 : Click on the allow.
- Step 5 : Click on Install Now.
- Step 6 : Click on Restart Now.

## Navigation for opening the firebug.

- Step 1 : Open any web page in Firefox browser.
- Step 2 : Select the Tools.
- Step 3 : Select the Web Developer.
- Step 4 : Select the Firebug and click on Open Firebug.
- Step 5 : Click on Inspect Element Icon () ; then select the required element in the web page.

## Syntax for identifying the element :

Webdriver.ref. findElement(By.Locator ("value")) . ~~SendKeys~~ Action ;

- Id
- Name
- Xpath
- LinkText
- Partial Link Text
- className
- Tag Name
- CSS Selector

- SendKeys()
- click()
- Isdisabled()
- Isenable()
- getText()

} Actions  
{  
webdriver  
Commands

Xpath : is a unique path for each element in the browser.

Example : Gmail Login TC

```
package com.tutorialspoint;  
public class GmailLogin_TC  
{  
    public static void main(String[] args)  
    {  
        WebDriver driver = new FirefoxDriver();  
        driver.get("http://google.com");  
        driver.manage().window.maximize();  
        // driver.findElement(By.partialLinkText("ail")).click();  
        driver.findElement(By.linkText("Gmail")).click();  
        //driver.findElement(By.  
        driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);  
        driver.findElement(By.name("Email")).sendKeys("testmail123456");  
        driver.findElement(By.id("Next")).click();  
        driver.findElement(By.xpath("//input[@id='passwd']")).sendKeys("selenium123");  
        driver.findElement(By.id("signIn")).click();  
    }  
}
```

7/9/2016

Example for Radio button

01	02	03	04
----	----	----	----

Public class RadioButton

{

Public static void main(String[] args) throws InterruptedException.

{

WebDriver driver = new FirefoxDriver();

driver.get("http://Selenium StreetKarth.blogspot.in");

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

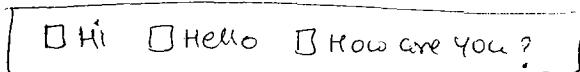
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

```

    driver.findElement(By.xpath("//3")).click();
    Thread.sleep(2000);
    driver.findElement(By.xpath("//4")).click();
    Thread.sleep(2000);
    driver.findElement(By.xpath("//1")).click();
}
}

```

### Example for Checkbox button



```
public class checkbox {
```

```
    public static void main(String[] args) throws InterruptedException {
```

```
}
```

```
    WebDriver driver = new FirefoxDriver();
```

```
    driver.get("http://seleniumguru.blogspot.com/checkbox");
```

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

```
    driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
```

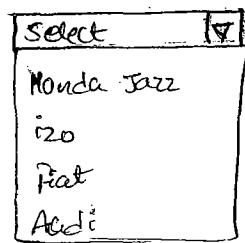
```
    driver.findElement(By.xpath("//1")).click();
```

```
    Thread.sleep(1000);
```

```
    driver.findElement(By.xpath("//3")).click();
```

```
}
```

### Example for Dropdown button:



```
public class Dropdown {
```

```
    public static void main(String[] args) throws InterruptedException {
```

```
}
```

```

webdriver driver = new FirefoxDriver();
driver.get("http://selesteekauth.blogspot.com/Dropdown");
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

- WebElement ddobj=driver.findElement(By.xpath("//label[@id='cars']"));
Select Oselect=new Select(ddobj);

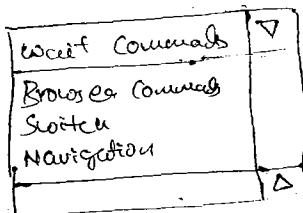
Oselect.selectByIndex(2);
Thread.sleep(2000);

Oselect.selectByVisibleText("Audi");
Thread.sleep(2000);

Oselect.selectByValue("120");
}
}

```

### Example for Listbox button:



Public class Listbox {

PSVM (String[] args) throws InterruptedException.

}

```

webdriver driver = new FirefoxDriver();
driver.get("http://selesteek.blogspot.com/listbox");
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

WebElement lobj = driver.findElement(By.xpath("//input"));
Select Oselect = new Select(lobj);

Oselect.selectByIndex(1);

```

model name: Ravi P271

o Select. SelectByIndex [1];

o Select. SelectedByIndex [2];

o Select. SelectByIndex [3];

o Select. DeselectByIndex [2];

}

## ⇒ Case Study 1:

http://Satyatech.com

Ex. Telugu Native  
Bharat Student

Name:

Gender: OM OF

Type:  ↴

Referred by:  Friends  
 Add's  
 enquiry..

Courses :

Selenium  
QTP  
Manual  
QC

Submit

// program.

```
public class Casestudy1
```

```
{
```

```
PSVM (String[] args)
```

```
{
```

```
webdriver driver = new FirefoxDriver();
```

```
driver.get ("http://Satyatech.com");
```

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

```
driver.findElement(By.linkText("Enquiry")).click();
```

```
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
```

```
driver.findElement(By.name("txtname")).sendKeys("Tejaswini");  
driver.findElement(By.ID("p")).click();
```

// driver.findElement

```
webElement dropobj = driver.findElement(By.name("Type"));
```

```
Select dselect = new Select(dropobj);
```

```
dselect.selectByIndex(1);
```

```
driver.findElement(By.ID("Friends")).click();
```

```
driver.findElement(By.ID("Eng")).click();
```

```
webElement listobj = driver.findElement(By.name("Cnd"));
```

```
Select lselect = new Select(listobj);
```

```
lselect.selectByVisibleText("selenium");
```

```
lselect.selectByIndex(2);
```

```
driver.findElement(By.ID("submit")).click();
```

3

4

// casestudy2 : Telugu Matrimony.com

Profile for

>

Name :

Native

Gender  M  F

SD

DOB

value

Mob   Num

value

Pwd

Xpath

Religion  >

dropdown (xpath)

Mother Tongue  >

" "

Continue

// casestudy3 : BharatStudent.com

Sign up free (Link)

About

Name :  First

Last

I am : OM OF

I live in :   Edit city

Birthday :  10  14  14

Occupation :

choose ID & Pwd

Email :

Pwd :

Cm Pwd :

Nickname :

Register

```

    {
        Public static void main (String[] args)
        {
            WebDriver driver = new FirefoxDriver();
            driver.manage().window().maximize();
            driver.get ("http://TeluguMatrimony.com");
            // driver.findElement(By.LinkText(""
            driver.manage().Timeouts().implicitlyWait (20, TimeUnit.SECONDS);
            WebElement dropobj = driver.findElement(By.name("Type"));
            Select dSelect = new Select (dropobj);
            dSelect.selectByIndex(1);
            dSelect.selectByVisibleText("Daughter");
            dSelect.selectByVisibleText("Son");
            driver.findElement(By.Name("Tentree")); sendkeys ("Rahul");
            driver.findElement(By.Id("m")) .click();
            WebElement dropobj = driver.findElement(By.ID(""));
            Select rSelect = new Select (dropobj);
            rSelect.selectByVisibleText("April");
            rSelect.selectByValue ("3");
            rSelect.selectByValue ("1991");
            driver.findElement(By.xpath("//@id='Mob Num')").Sendkeys ("9934887891");
            driver.findElement(By.xpath("//@id='pwd')").Sendkeys ("Matrimony123");
            WebElement dropobj = driver.findElement(By.xpath("//@id='Religion')");
            Select r1Select = new Select (r1Select);
            r1Select.selectByVisibleText("Hindu");
            WebElement dropobj = driver.findElement(By.xpath("//@id='Religion')");
            Select MSelect = new Select (MSelect);
            MSelect.selectByVisibleText("Telugu");
            driver.findElement(By.Id("Continue")).click();
        }
    }

```

## Property file :

Property file used for storing some general information and can be reused in any java file (class) just by loading it.

The following statements need to be used in that particular class.

After passing path of the properties file here we need to use instead of that / or \. (In java programming directories are separated by \ or /)

## Object Repository :

In Selenium, there is no Object repository concept but we will store all the object information in the properties file and will treat it as a object repository.

To retrieve any value from the properties file we need to follow the below syntax.

P.getProperty ("property Variable name");

## Navigation for Creating property file :

- 1) Click on project (Right click)
- 2) Select New → choose File.
- 3) Enter filename and click finish
- 4) Property file is created.

Ext // Propfile

url : http://gmail.com

un.text.loc = .//[@id='Email']

pwd.text.loc = password

next.btn.loc = next

sbtn.btn.loc = .//[@id='signIn']

un = testmail.no6

pwd = selenium123

⇒ System.getProperty ("user.dir") → This stat is used for getting the current directory during time of execution.

```

public class PropfileExample
{
    public static void main(String[] args)
    {
        // Associating the property file
        FileInputStream fis = new FileInputStream(System.getProperty("user.dir") + "/propfile");
        Properties p = new Properties();
        p.load(fis);

        WebDriver driver = new FirefoxDriver();
        driver.get(p.getProperty("url"));
        driver.findElement(By.linkText("Sign In")).click();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
        driver.findElement(By.xpath(p.getProperty("un.text.loc"))).sendKeys(p.getProperty("un"));
        driver.findElement(By.id(p.getProperty("next.btn.loc"))).click();
        driver.findElement(By.name(p.getProperty("pwd.text.loc"))).sendKeys(p.getProperty("pwd"));
        driver.findElement(By.xpath(p.getProperty("sbtn.btn.loc"))).click();
    }
}

```

9/9/2016

- Whenever one page having multiple elements that all the elements are having common properties. In this situation, to identify uniquely one element we can follow below example.

```

public class ExampleforFindElements
{
    public static void main(String[] args)
    {
        WebDriver driver = new FirefoxDriver();
        driver.manage().window().maximize();
        driver.get("http://SeleniumBreakwith.blogspot.com/checkboxes");
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    }
}

```

```
List<WebElements> ol = driver.findElements(By.name("gender"));
```

```
ol.get(1).click();
```

```
ol.get(2).click();
```

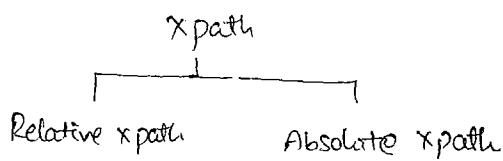
```
}
```

NOTE + For Single Element (whether to click or sendKeys) "findElement" is used.

For Multiple elements (.....) "findElements" is used.

XPATH:

↳ xpath is a unique locator for webelement in the web application.



Difference b/w Relative path and Absolute path:

→ Absolute path will start with root path (1)

Ex → Absolute xpath : html/body/div[1]/div[2]/div[2]/form/div[1]/div/div[1]/input[1]

→ Relative path will <sup>start</sup> from current path (11)

Ex → Relative xpath : .//\*[@id='email']

To identify objects by using xpath we will go for below ways

- 1) Attribute
- 2) text
- 3) Starts-with
- 4) Substring
- 5) Contains Constraints.

Creating Our own xpath:

Syntax 1:

```
//tagname[@attributename='attributevalue']
```

Ex   .//input[@name='email']   .//input[@id='pwd']

~~synonyms~~

## Multiple handling.

- //input[@id='Email' or @name='Email']
- //input[@type='text' or, @type='password'].

⇒ Many web sites create dynamic element on their web pages where Id's of the elements gets generated dynamically. Each time id gets generated differently. So to handle this situation we use some "JavaScript functions".

### i) Starts-with

- //\*[starts-with(@id, 'value')]

If first characters are constant and last letters are keep on changing then we go for starts-with.

Ex. ID =  $\frac{\text{const}}{\text{abc}} \frac{123}{\text{varies}}$

### ii) SubString

- //\*[substring(@id, 6)='signup']

If first letters are changing and last letters are constant then we go for substring.

Ex. ID =  $\frac{123}{\text{varies}} \frac{\text{const}}{\text{abc}}$

### iii) Constraints

If you want to identify object using partial property then we go for Constraints.

- //\*[contains(@id, 'link')]

is similar to partialLinkText  
where we can give incomplete link to full.

### Example for Starts-with method:

```

public class Xpathex1
{
    public void(String[] args)
    {
        WebDriver driver = new FirefoxDriver();
        driver.get("http://gmail.com");
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
        driver.findElement(By.xpath(".//input[@name='Email']")).sendKeys("testmail123");
        driver.findElement(By.xpath(".//input[@id='next']")).click();
        driver.findElement(By.xpath(".//input[@id='passwd']")).sendKeys("Selenium123");
        driver.findElement(By.xpath(".//input[@id='signIn']")).click();
        driver.findElement(By.xpath(".//*[@id='i1t']/div/div")).click();

        String str = driver.findElement(By.xpath(".//div[starts-with(@id,'i')]//h2/div[2]")).getText();
        System.out.println(str);
    }
}

```

### Example for Substring method:

```

String str = driver.findElement(By.xpath(".//div[contains(@id,'i')]//h2/div[2]")).getText();

```

### CSS Selector [Locator]:

CSS is a 'Cascading Style Sheets' and it is defined to display HTML in Structured and Colored styles are applied to webpage.

- CSS has more Advantage than XPath.
- CSS Selector is faster and simpler than the XPath.
- In IE, XPath works very slow, whereas CSS works faster when compared to XPath.

## Syntax:

Tagname [ attribute = 'attr value' ]

Ex: input [ id = 'E-mail' ]

input [ id = 'E-mail' ] [ name = 'E-mail' ].

## Regular Expressions:

Whenever particular object property value is changing regularly, due to that reason unable to identify the object on web element.

To overcome this Problem we are using regular expressions.

i) char: \n

Desc : Matches the beginning of input.

ii) char: \\$

Desc : Matches the end of input.

iii) char: \*

Desc : Matches the preceding character zero or more times.

Ex: "z0\*" matches either "z" (or) "zoo".

## // Example for CSS Selector

```
public class CSSSelectorExample
```

```
{
```

```
    public void(String[] args)
```

```
{
```

```
        WebDriver driver = new FirefoxDriver();
```

```
        driver.get("http://gmail.com");
```

```
        driver.findElement(By.cssSelector("input[id='Email']")).sendKeys("testmail.106");
```

```
        driver.findElement(By.cssSelector("input[id='Email'][name='E-mail']")).sendKeys("");
```

```
        driver.findElement(By.cssSelector("input[name^='Email']")).sendKeys("testmail.106");
```

```
        driver.findElement(By.cssSelector("input[name$='ail']")).sendKeys("testmail.106");
```

```
driver.findElement(By.CSSSelector("input[name*='mail']")).SendKeys("testuser106");
```

{  
}

## JavaScript in Detail :

Variable — is a name given by the memory location which holds the value if required that value we can change also.

Syntax :

```
<AM><DT><var name>;
```

```
Ex: public int i=10;
```

## Advantages :

- 1) Reusability
- 2) Managing very easily.

11/9/2016

Constant Variable — is a name given by the memory location which holds the value, If required that value, we can't edit.

Syntax :

```
final <DT><VN>=*value*;
```

```
Ex: final int i=10;
```

## Conditional statements : (Decision making statements)

In java, we have 2 types of Conditional Stmts.

→ if - else

→ Switch --- case

if - else : whenever we want to execute one block of stmts among two blocks then we can use If - Else.

Syntax : if (cond)

{

    block of stmts;

}

— condition is true then code or statement will be executed, otherwise returning else (stmt 2).

Switch-Case : whenever we want to execute a particular block of Stmt among many blocks then we will choose Switch Case Stmt.

Syntax : switch (choice)

{

case "c1" :

    block 1

    break;

case "c2" :

    block 2

    break;

case "c3" :

    block 3

    break;

⋮

Default :

    S.O.P(); // not mandatory

}

## Looping Statements:

Looping Stmt are used for executing a certain block of Stmt repeatedly and continuously for some number of times. In Java, we have three types of looping Stmt. i) for  
ii) while  
iii) do ... while

i) for loop : whenever we clearly know how many number of times the block should be repeated then use for loop.

Syntax : for (Initialization; Condition; Increment/Decrement)

{

    ⋮

}

```
Ex: for (int i=0; i<10; i++)  
{  
}
```

ii) while loop: is used for executing a block of statements repeatedly as long as Condition is being satisfied

Syntax:

```
while (cond)  
{  
    block of statements;  
}
```

Ex: while (i<5)  
{  
 = =  
}

iii) do...while loop: is used for executing a block of statements repeatedly as long as the condition is being satisfied, but first time execution will be done without checking the condition.

Syntax:      do  
{  
 ... block of statements;  
} while (cond);

Ex      do  
{  
 ...  
} while (i<5);

## Arrays:

Array is a special type of variable which can hold the multiple values.

In java, we have following types of arrays

- i) Single Dimension Array
- ii) Multi Dimension Array
- iii) Object Array

Syntax:

DT[] Arrayname = new DT[size];  
(Or,

DT Arrayname[] = new DT[size];

Ex:

```

public class SingletArray
{
    PSVM (String [] args)
    {
        int [] i = new int [4];
        i[0] = 5;
        i[1] = 6;
        i[2] = 7;
        i[3] = 8;
        for (int j=0 ; j < i.length ; j++)
        {
            S.O.P (i[j]);
        }
    }
}

```

## ii) Multi Dimension Array :

↳ to store more number of multiple values very easily. we should use this array.

Syntax:

datatype [][] Arrayname = new datatype [size][size];

(Or,

datatype Arrayname [][] = new datatype [size][size];

Ex: Public class MultidimArray

```

}
PSVM (String [] args)
{

```

int [][] i = new int [5][6];

i [0][0]=4;

```

    i[1][1] = 5;
    i[2][2] = 6;
    i[3][3] = 7;
    i[4][4] = 9;
    for (int j=0; j < i.length; j++)
    {
        for (int j2=0; j2 < i.length; j2++)
        {
            System.out.print(i[j][j2]);
        }
        System.out.println();
    }
}

```

### iii) Object Array:

It is used for storing the different datatype values in array.

#### Syntax:

```

Object[] ObjArryname = new Object[size];

```

Exn

```

public String[] args)
{
    Object[] obj = new Object[5];
    obj[0] = 12;
    obj[1] = 12.2;
    obj[2] = "selenium";
    obj[4] = 'E';
    obj[5] = 'True';

    for (i=0; i < obj.length; i++)
    {
        System.out.print(obj[i]);
    }
}

```

## Collection API

↳ is having so many classes in that very important for us is ArrayList, Hash Table, Hash Set.

### Advantage:

→ Dynamically growing the size of array.

12/9/2016

## Collection API

↳ ArrayList class, Hashtable class are predefined class of collection API in java. - for storing and retrieving values from array, we will use index.

Index starts from '0' and continues like '1, 2, 3, ...'.

ArrayList Class - is used for dynamically growing array.

### Syntax:

ArrayList<DT> arrayname = new ArrayList<DT>();

- for retrieving values from this we will use index (starts from '0' & continues)  
arrayname.get(index);
- To add data to this array we need to use below syntax

### Syntax:

arrayname.add(data);

NOTE: Here data is of same data type's data <sup>at</sup> ~~by~~ the time of declaration.

### Exa ArrayList class

public class ArrayListclass

{

    public void main(String[] args)

{

        ArrayList<String> al = new ArrayList<String>();

        al.add("ar");

    }

```

al.add("a3");
al.add("a4");
S.O.P(al.size()); //4
S.O.P(al.get(1)); //a2
}

```

Hashtable — is used for simulating dynamically growing array.

for storing and retrieving values we will use keys instead of index.

Syntax:

```
Hashtable<DT1, DT2> Arrayname = new Hashtable<DT1, DT2>();
```

- Data Type 1 is for 'keys'      } DT1, DT2  
 Data Type 2 is for 'values'      } are of same datatype's.

- To add data to this array we need to use the below Syntax.

```
Arrayname.put(data, data);
```

NOTE: here data is of same datatype's data ~~at~~ at the time of declaration.

~~Ex~~ — To access (retrieve) data from this array, we need to use the below Syntax.

Syntax:

```
Arrayname.get(keys);
```

Ex Hashtable class

```
public class Hashtableex
```

```
{
```

```
PSVM(String[] args)
```

```
{
```

```
Hashtable<String, String> ht = new Hashtable<String, String>();
```

```
ht.get ht.put("city1", "Hyd");
```

```
ht.put("city2", "Hyd");
```

```

S.o.p(ht.size()); // 3
S.o.p(ht.get("city1")); // Hyd
}
}

```

### Set

HashSet :- Duplicate values are not stored.

~~It is not~~

- To get the values from HashSet, no direct method is declared.

Hence we use "iterator" class

hasnext()  
next()

next() :- is used to go for next value and get the next value.

Initially it is in the top.

```
// system.out.println(itr.next());
```

hasnext :- is used for check values is available or not.

If value is available, then it gives "true" if true get value.

Otherwise it will give "false" and not give any error also.

Ex HashSet class

```

public class HashSeter
{
    public static void main(String[] args)
    {
        Set<String> s = new Set<String>();
        s.add("a1");
        s.add("a2");
        s.add("a3");
        s.add("a4");
        S.o.p(s.size());
    }
}
```

```
// Iterator i = new  
Iterator<String> itr = s.iterator();  
while (itr.hasNext())  
{  
    S.O.P(itr.next());  
}  
}  
}
```

### File System Object Concept:

↳ It is concept of providing java code which is used for creating a new file in the file system.

→ Writing the data into the file system and reading the data into the file system.

To do same in java they have provided 5 predefined class.

- 1) file
- 2) file Reader
- 3) buffered Reader
- 4) file writer
- 5) buffered writer

```
public class filewriter  
{  
    public void (String[] args) throws Exception  
    {  
        File f = new File ("D://selenium Note.txt");  
        // f. file  
        f. CreateNewFile();  
    }
```

```
filewriter fw = new filewriter ("D://seleniumnote.txt");
```

```
BufferedWriter bw = new BufferedWriter (fw);
```

```
bw.write ("Hi selenium");
```

```
bw.newLine();
```

```
bw.write ("Hi webdriver");
```

```
... ... ... ... ... file to permanent
```

```

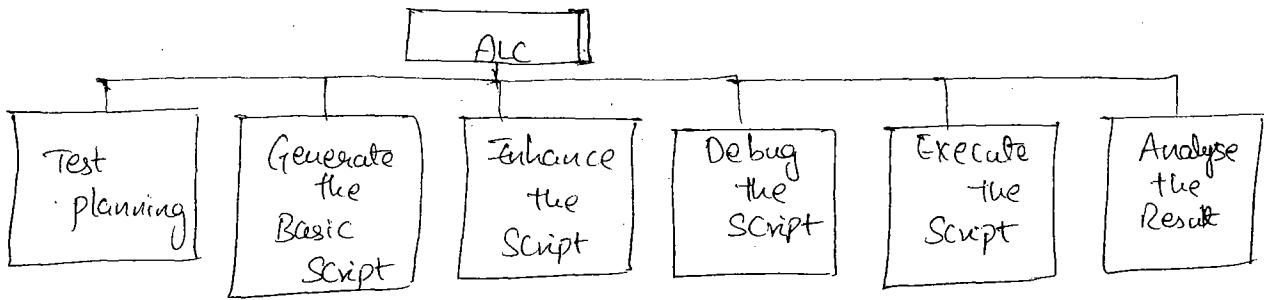
Ex 2) public class FileReader
{
    PSvm(String[] args) throws IOException
    {
        FileReader fr = new FileReader ("D:\\seleniumnotes.txt");
        BufferedReader br = new BufferedReader(fr);
        while (br.ready())
        {
            String str = br.readLine();
            S.O.P(str);
        }
    }
}

```

13/9/2016

### Automation Life cycle (ALC):

It is divided into 6 phases



#### 1) Test planning :

In this phase, the automation test lead will do the following

- He will identify the areas to be automated.
- He will analyze both positive flow and negative flow of the application.
- He will do resource planning & scheduling.
- He will prepare Automation test plan document with all the above analyzed information.

#### 2) Generating the Basic Script :

In this phase, the automation test engineer will generate basic script

- It is a first iteration of the automation

### 5) Enhancing the Script :

- i) Checking with Assertions
- ii) Data Driven Testing
- iii) Synchronization.

→ Ternary operator : (?)

Syntax:

Condition ? true result : false result;

It is alternative for (if - else) looping

→ getAttribute() :

↓

This method is used for getting the particular element property value during execution time.

Syntax:

element.getAttribute("property value");  
                                name

(i) Checking with Assertions:

Assertion is something like a checking statement which is used for checking whether the application is upto expectations or not.

Ex → finding the element, getting attribute value and comparing with expected value.

Public class checkingEx

{

    PSVM (String [] args)

{

        WebDriver d = new FirefoxDriver();

        d.get("http://cleartrip.com");

        d.manage().windows().maximize();

        d.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

        String EV = "Multicity";

        String OR = d.findElement(By.xpath(".//\*[@id='Multicity']]")).getAttribute("value");

// Example for handling the popup window

public class Popuper

{

PSVM(String[] args) throws InterruptedException

{

WebDriver d = new FirefoxDriver();

d.get("http://seleniumhq.org");

d.manage().windows().maximize();

d.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

d.findElement(By.xpath("//\*[@id='queryTop']/div/input[6]")).click();

Thread.sleep(3000);

Alert a = driver.switchTo().alert();

a.accept(); // Ok btn

}

15/9/2016

Multiple windows handling :

i) getwindowHandle() :

This method is used for getting the current window name.

Syntax :

driver.getWindowHandle();

ii) getWindowHandles() :

This method is used for getting the current multiple windows names.

Syntax :

driver.getWindowHandles();

iii) switchTo() :

This method is used for focussing the one window to another window or frame and so on.

For example about

Switch:

```
driver.switchTo().alert();
```

for window,

```
driver.switchTo().window(WindowName);
```

for frame,

```
driver.switchTo().frame(int arg()); //id
```

```
driver.switchTo().frame(String arg()); //name
```

```
driver.switchTo().frame(WebElement arg());
```

//Example for multi window handling.

```
public class Multiwinhandling
```

```
{ public static String freshid;
```

```
public Multiwinhandling(String[] args)
```

```
{}
```

```
Webdriver d = new FirefoxDriver();
```

```
driver.get("http://Telugumachinay.com");
```

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

```
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
```

```
driver.findElement(By.xpath(".//a[@id='close']/center/div[1]")).click();
```

```
Set<String> s = driver.getWindowHandles();
```

//Iterator itr =

```
Iterator<String> itr = s.iterator();
```

```
while(itr.hasNext()):
```

```
{
```

```
freshid = itr.next();
```

```
}
```

```
driver.switchTo().window(freshid);
```

```
driver.findElement(By.linkText("forgotten account?")).click();
```

```
}
```

It is a process of matching the speeds of both the tool and application in order to keep them in sync with each other to obtain proper testing results.

Here the main concept is making the selenium to wait till the main application allows it to perform the next operation. So it can be done in three ways.

→ By inserting the "implicitlyWait()" statements.

~~By Syntax:~~

WebdriverReference.manage().timeouts().implicitlyWait(time, TimeUnit);

{  
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

16/9/2016

→ By inserting "Explicit wait()" statements.

→ By inserting "Thread.sleep()" statements.

// Example for "Explicit wait()" statements.

Public class Explicitwait

{

    P S V m (String [ ] args) {  
    }

    Webdriver d = new FirefoxDriver();

    d.get("http://rediff.com");

    d.manage().window().maximize();

    WebDriverWait wait = new WebDriverWait(d, 20); // Explicit wait  
    // And starts here

    wait.until(ExpectedConditions.titleIs("Rediff.com")); // Title

    S.O.P ("Title is displayed");

    d.findElement(By.xpath(".//\*[@id='queryTOP']/div/input[6]")).click(); // search button

    wait.until(ExpectedConditions.alertIsPresent()); // alert (pop-up)

    Alert a = driver.switchTo().alert();

    a.accept();

3

// Example for Thread.sleep() statements.

```
public class Threadclass  
{  
    public void (String[] args) throws InterruptedException  
    {  
        System.out.println("Hi");  
        Thread.sleep(3000);  
        System.out.println("Hello");  
        System.out.println("How are you?");  
        Thread.sleep(3000);  
        System.out.println("Bye");  
    }  
}
```

#### 4) Debugging the Script:

It is a process of executing the test in a user-desired fashion with some temporary breaks, in order to identify the errors.

To do the same in Eclipse, there is a concept of "breakpoint" feature and "Step Commands".

##### Breakpoint:

→ It is used for breaking the execution temporarily.

##### STEP COMMANDS:

→ Divided into three types      i) Step Into  
                                      ii) Step Over  
                                      iii) Step return.

i) Step Into :- is used for executing a single step, if the step is a function call then it will make the pointer step into the function and breaks the execution at the first statement.

ii) Step Return :- is used only whenever pointer is inside the function, it will execute all the remaining statements inside the function from the position of the pointer and breaks the execution after stepping out of the function.

III) Step Over - is used for executing any step until it is finished.

for example, if it is a functional call statement it will execute all the statements inside the function and then breaks the execution.

// Example for debugging the Script.

```
public class Debugging  
{  
    public void info()  
    {  
        S.O.P ("Hi");  
        S.O.P ("Hello");  
    }  
  
    public static void main(String[] args)  
    {  
        breakpoint  
        Debugging db = new Debugging();  
        db.info();  
        S.O.P ("Welcome"); // breakpoint(0) is kept in order to stop the execution temporarily &  
        db.info();  
        S.O.P ("How are you ?");  
        db.info();  
        S.O.P ("Fine");  
        S.O.P ("Bye ");  
    }  
}
```

5) Test Execution / Executing the Script :

In this phase, one will execute the test.

6) Analysing the Result :

In this phase, one will analyse the result.

## MOUSEMOVE Action:

// Example for one mousemove action.

```

public class mousemove {
    §
    public void (String[] args) {
        §
        WebDriver d = new FirefoxDriver();
        d.get("http://flipkart.com");
        d.manage().window().maximize();
        WebElement obje = d.findElement(By.xpath("//select[@id='slectspdt']"));
    }
}

```

```

Actions actions = new Actions(driver);
actions.moveToElement(obje).perform(); // for one
}
}

```

// Example for double action for mousemove.

```
WebElement Obj1 = d.findElement(By.xpath("//path[1]"));
```

```
WebElement Obj2 = d.findElement(By.xpath("//path[2]"));
```

```
Actions actions = new Actions(d);
```

```
actions.moveToElement(Obj1).moveToElement(Obj2).perform();
```

17/9/2016

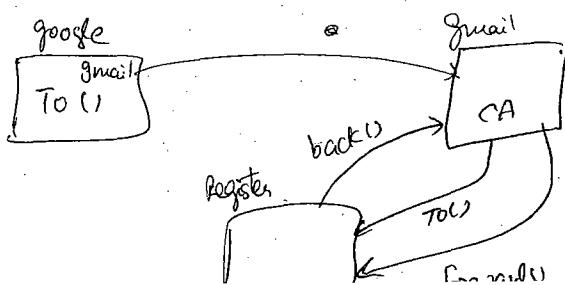
## Navigational Commands:

Navigate.To() — Navigate one page to another page.

Navigate.Back() — " one step backward.

Navigate.forward — " one step forward.

Navigate.Refresh — Refresh page.



to new page  
 ↓  
 page it can stop  
 forward & back & refresh

It also directs to new page  
 ↓  
 Page but it waits until page is loaded.  
 It can't stop back, forward & refresh.

### Interview Question

Difference b/w nav.to() & get().

### Navigational Commands

We have the following navigational Commands in Selenium webdriver.

- 1) Navigator().to()
- 2) Navigator().back()
- 3) Navigator().forward()
- 4) Navigate().refresh(),

### Difference between get() and navigate().to() To Open the webpage

- Get Method will get a page to load or get page source.
- Navigate will guide through the history like Refresh(), back(), forward(),  
for example, if we want to move the forward and do something functionality  
and back to the home page these can be achieved through navigator only.
- Driver.get()  
will wait till the whole page gets loaded

Driver.navigate()

↓  
will just redirect to that page and will not wait.

Exa public class NAVCNS

{

PSVM ( ) throws InterruptedException

{

WebDriver driver = new FirefoxDriver();

```

driver.manage().window().maximize();
driver.findElement(By.linkText("Create account")).click();
driver.navigate().back();
Thread.sleep(3000);
driver.navigate().forward();
Thread.sleep(3000);
driver.navigate().refresh();
}
}

```

// Example for handling the webTable Data:

```

public class WebTableData
{
    public String[] data
    {
        WebDriver driver = new FirefoxDriver();
        driver.get("http://www.cbschools.com/tags/tag-table.asp");
        driver.manage().window().maximize();
        WebElement table = driver.findElement(By.xpath("//table[@id='main']"));
        List<WebElement> rows = table.findElement(By.tagName("tr"));
        for(int i=0; i < rows.size(); i++)
        {
            List<WebElement> cell = rows.get(i).findElement(By.tagName("td"));
            for(int j=0; j < cell.size(); j++)
            {
                String str = cell.get(j).getText();
                System.out.println(str);
            }
        }
    }
}

```

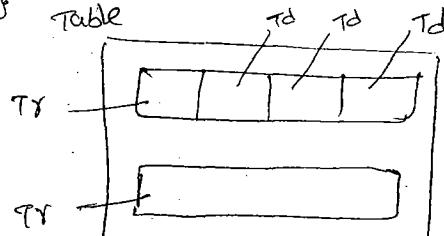


Table  
TR - Rows  
TD - Columns  
Element = driver.findElement

## Methods in Selenium Webdriver

### 1) getTitle():

- This method is used for getting title from current webpage.

Syntax:

```
WebDriver ref.getTitle();
```

### 2) clear():

- This method is used for clearing text field data.

Syntax:

```
element.clear();
```

### 3) isDisplayed():

- This method is used for verifying the element that exists or not. If available returns true otherwise false.

Syntax:

```
element.isDisplayed();
```

### 4) isEnabled():

- This method is used for verifying the element that is enabled or not. If enabled returns true otherwise false.

Syntax:

```
element.isEnabled();
```

### 5) isSelected():

- This method is used for verifying

Syntax:

```
element.isSelected();
```

### 6) getText():

This method is used for get text from the specific element.

Syntax:

```
element.getText();
```

### 7) click():

This method is used for click on element.

Syntax:

```
element.click();
```

### 8) equals():

This method is used for compare two string values. If both return same → true otherwise false.

Syntax:

This method is used for close browser.

Syntax : WebDriver ref.close();

10) quit():

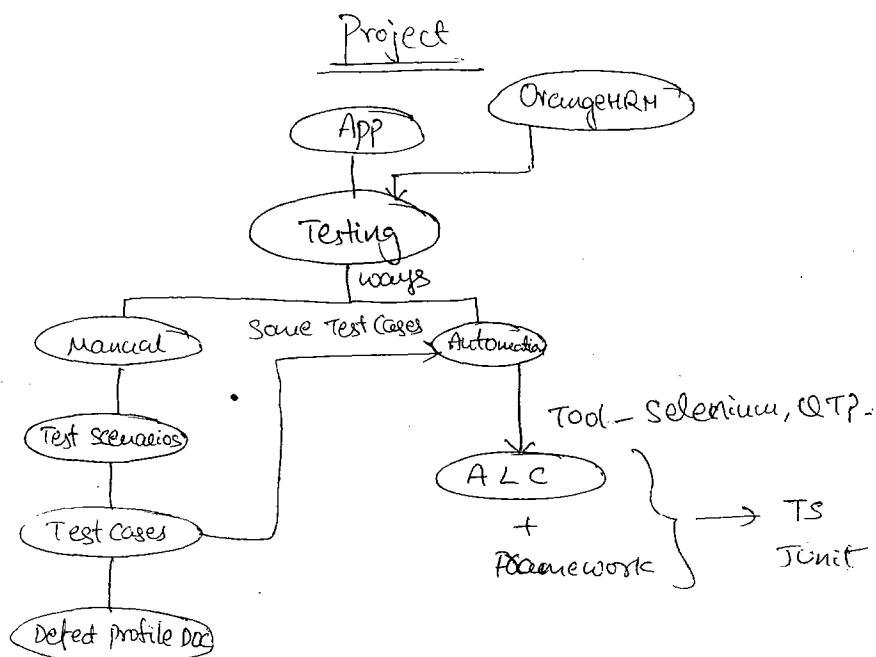
This method is used for close the object

Syntax : WebDriver ref.quit();

11) get():

To load the page.

Syntax : WebDriver ref.get("url");



## ORANGE HRM PROJECT IMPLEMENTATION

### Test Case 1:

→ Verify the Orange HRM Admin login.

Steps	Expected Result
1) Open Orange HRM	Verify Home page should be displayed with Username & password and login buttons.
2) Enter Valid username & password and Click login button	Verify OrangeHRM page displayed and verify below details like Welcome message, change password link and logout.
3) Click on logout	Verify whether Home page should be displayed
4) close browser	

## → Verify Add Employee Creations

Steps	Expected value
1) Open Orange form	Verify Homepage should be displayed with UN, pwd & login buttons.
2) Enter valid UN, pwd & click login button	Orange Hrm page should be displayed with welcome message, change pwd & logout options
3) focus on PIM	Verify employee list and add employee.
4) click on add-employee	Verify Add Employee page is displayed correctly
5) Enter the add employee details & click Save button.	Verify personal details page <del>is</del> is displayed or not.
6) Click on employee list	Verify employee list information page & Verify employee info in the table.
7) Click on logout and close the browser	Verify Homepage is displayed or not.

### Script for Testcase:

```
public class VerifyAdminLogin_TC1
{
    public void main(String[] args) throws InterruptedException
    {
        WebDriver driver = new FirefoxDriver();
        driver.get("http://OrangeHRM.uptoselenium.com/");
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

        // Verify title of homepage.
        if (driver.getTitle().equals("OrangeHrm"))
        {
            System.out.println("OrangeHrm home page is displayed");
        }

        // Verifying UN, pwd & login button.
        WebElement objun = driver.findElement(By.xpath("//input[@id='txtuser']"));
        WebElement objpwd = driver.findElement(By.xpath("//input[@id='txtuser']"));

    }
}
```

```
if (objun.isDisplayed() & objpwd.isDisplayed() & objlogin.isDisplayed())
```

```
{  
    S.o.p ("UN, pwd, login button is available");  
}
```

```
}  
String UN = "Sreekanth";
```

```
String pwd = "Sreekanth @ 2015";
```

```
objun.clear();
```

```
objun.sendKeys(un);
```

```
objpwd.clear();
```

```
objpwd.sendKeys(pwd);
```

```
objlogin.click();
```

```
// Verify title of Admin Home page.
```

```
if (driver.getTitle().equals("OrangeHRM"))
```

```
{  
    S.o.p ("OrangeHRM Admin homepage is displayed");  
}
```

```
}  
String weltext = driver.findElement(By.xpath("//div[@id='welcome']")).getText();
```

```
if (weltext.equals("welcome Sreekanth"))
```

```
{  
    S.o.p ("welcome Sreekanth is displayed");  
}
```

```
}  
Thread.sleep(5000);
```

```
driver.findElement(By.xpath("//a[@id='welcome']")).click();
```

```
"           (By.linkText("Logout")).click();
```

```
// Verify Title Home page.
```

```
if (driver.getTitle().equals("OrangeHRM"))
```

```
{  
    S.o.p ("OrangeHRM homepage is displayed");  
}
```

```
}  
driver.close();
```

```
S.o.p ("close the Browser");
```

```
driver.quit();
```

```
S.o.p ("close the object");  
}
```

```
}
```

## Log4J :

↳ Log4J Concept is very important to any application. It helps us collect information about how the application is running and also helps us to debug if any failure occurs.

### Procedure to Implement the Log4J Concept:

- 1) Download the Log4J jar files.
- 2) Configure the jar files into the project.
- 3) Create the .xml file.
- 4) Associate the .xml file into the class.
- 5) Design the output in your designed passion.
- 6) Create the Test Case.
- 7) Analyse the output/Result.

Exn

```
package com.orgnum.scripts;
import org.apache.log4j.Logger;
public class log4jconcept
{
    private static Logger log=Logger.getLogger(log4jconcept.class.getName());
    public static void main(String[] args)
    {
        DOMConfigurator.configure("log4j.xml");
        log.info("Hi");
        log.info("welcome");
    }
}
```

20/9/2018

## AutoIt

- Selenium web driver will support only webBased application. So, if you want any windowBased application action need to implement then we can use the Action class.

- Download the Auto it and install in our system and store in your desired location.

### Navigation for downloading Auto it :

- 1) Click on start
- 2) Click on all programs
- 3) " " Auto it v3
- 4) Here click on '4' option

#### i) Script Editor :

↳ used for writing the script Viewing the script & Modifying.

#### ii) Auto it Window Info :

↳ used for inspecting the elements on the application and showing the information.

#### iii) Run Script :

↳ used for executing the Auto it programs.

#### iv) Compile Script to exe :

↳ used to convert Auto it program to the .exe file.

- In Autoit already inbuilt ~~shor~~ commands are many and available using the commands perform the actions.

Ex : WinActivate.

WinActivate : is used for activating the window.

Syntax : WinActivate ("title" [, "text"])

Control Click :— These Commands is used for click on the required Control.

Syntax : ControlClick ("title", "text", \$Button")

Control Set Text :— is used for entering the some data into the text field.

// Example for calculator actions:

Run ("calc.exe")

Sleep(2000)

WinActivate ("[Title:calculator]", "")

Sleep(2000)

ControlClick ("[Title:calculator]", "", "Button9")

Sleep(2000)

ControlClick ("[Title:calculator]", "", "Button15")

Sleep(2000)

ControlClick ("[Title:calculator]", "", "Button21")

Sleep(2000)

ControlClick ("[Title:calculator]", "", "Button10")

Sleep(2000)

ControlClick ("[Title:calculator]", "", "Button28")

Sleep(4000)

WinClose ("[Title:calculator]", "")

// Example for uploading the image:

Sleep(1000)

WinActivate ("[Title:file upload]", "")

Sleep(1000)

ControlFocus ("[CLASS:#32770]", "", "Edit1")

Sleep(1000)

ControlFocus ("file upload", "", "Edit1")

Sleep(1000)

ControlSetText ("[Title:file upload]", "", "Edit1", C:\users\public\picture...)

Sleep(1000)

ControlClick ("[Title:file upload]", "", "Button1")

Sleep(1000)

MVIE. If you want execute the AutoIT programs or, use the AutoIT programs in the middle of WebDriver Script we will follow below syntax.

### Syntax :

```
Runtime.getRuntime().exec("C:/users ---")  
                                ↓  
path of the exe file.
```

### Test

#### Script for Testcase2 :

```
public class Addemp_Creation_TC2  
{  
    public static void main(String[] args)  
    {  
        WebDriver driver = new FirefoxDriver();  
        driver.get("http://OrangeHRM.uitSelenium.com/");  
        driver.manage().window().maximize();  
        driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);  
  
        // Verify title of home page.  
  
        if (driver.getTitle().equals("OrangeHRM"))  
        {  
            System.out.println("OrangeHRM home page is displayed");  
        }  
  
        // Verify un, pwd & login button  
  
        WebElement Objun = driver.findElement(By.xpath(".//[@id='txtusername']"));  
        WebElement Objpwd = " " (By.xpath(".//[@id='txtpassword']"));  
        WebElement Objlogin = " " (By.xpath(".//[@id='btalogin']"));  
  
        if (Objun.isDisplayed() & Objpwd.isDisplayed() & Objlogin.isDisplayed())  
        {  
            System.out.println("UN, pwd, login buttons are available");  
        }  
  
        String un = "Sreekanth";  
        String pwd = "Sreekanth@2015";
```

```
Objun.sendKeys();
```

```
Objun.sendKeys(Keys.ENTER);
```

```
Objlogin.click();
```

```
// Verify title is Admin Home page.
```

```
if(driver.getTitle().equals("OrangeHRM"))
```

```
{
```

```
S.O.P("OrangeHRM Admin home page is displayed");
```

```
}
```

```
String weltxt = driver.findElement(By.xpath("//*[@id='welcome']")).getText();
```

```
if(weltxt.equals("welcome Sreekanth"))
```

```
{
```

```
S.O.P("welcome Sreekanth is displayed");
```

```
}
```

```
// Mouseover Action
```

```
WebElement objPM = driver.findElement(By.xpath("//*[@id='Menu-pim-viewapping']"));
```

```
Actions actions = new Actions(driver);
```

```
Actions.moveToElement(objPM).perform();
```

```
Thread.sleep(3000);
```

```
driver.findElement(By.linkText("Add Employee")).click();
```

```
// checking Add Employee page.
```

```
String addemptxt = driver.findElement(By.xpath("//*[@id='content']/div/div[3]/h1")).getText();
```

```
if(addemptxt.equals("Add Employee"))
```

```
{
```

```
S.O.P("Add Employee details page is displayed");
```

```
}
```

```
String FN = "Tejaswini";
```

```
String LN = "Adabala";
```

```
driver.findElement(By.xpath("//*[@id='firstname']")).sendKeys(FN);
```

```
((By.xpath("//*[@id='lastname']")).sendKeys(LN);
```

```
((By.xpath("//*[@id='photofile']")).click();
```

// Auto it

```
Routine.getRuntime().exec("D://upload.exe");
Thread.sleep(8000);
driver.findElement(By.xpath("//*[@id='btNSave']")).click();
String pdtext = driver.findElement(By.xpath("//*[@id='pdMainContainer']/div[1]
if(pdtext.equals("Personal Details"))
{
    S.O.P("Personal Details page is displayed");
}
ObjPIM = driver.findElement(By.xpath("//*[@id='menu-pim-viewpimlist']"));
actions.moveToElement(ObjPIM).perform();
Thread.sleep(3000);
driver.findElement(By.linkText("Employee List")).click();
Thread.sleep(3000);
```

// Searching Emp List in table.

```
WebElement table = driver.findElement(By.xpath("//*[@id='resultTable']"));
List<WebElement> rows = table.findElements(By.tagName("tr"));
for(int i=1; i < rows.size(); i++)
{
    String ActLN = driver.findElement(By.xpath("//*[@id='resultTable']/tbody/
tr["+i+"]/td[4]/a")).getText();
    if(LN.equals(ActLN))
    {
        S.O.P(ActLN+" is displayed at "+i+" row");
        break;
    }
}
Thread.sleep(5000);
driver.findElement(By.xpath("//*[@id='welcome']")).click();
driver.findElement(By.linkText("Logout")).click();
```

## Framework

Framework is a generic workflow set of guidelines given by the group of experts in order to implement your task efficiently, effectively and optimized way.

### Types of Framework :

- 1) Modular framework
- 2) Data driven framework
- 3) Keyword driven framework
- 4) Hybrid framework
- 5) Page object module.

### Junit :

Junit is a software which provides some facilities through annotations.

- which makes testing job more easy. People generally call it as 'Framework'

#### There is

- No need to have any main method in junit, instead we are using static method in @beforeclass @Afterclass.
- Firstly, add junit into the project and work on it.

### // Example for Annotation list.

- @ Before class
- @ After class
- @ Before
- @ After
- @ Test
- @ Ignore
- @ Test.

## Navigation for Adding Junit library into the Project:

- 1) Right click (R.C) on the Project
- 2) Select Properties
- 3) Select java build path
- 4) click ~~and~~ add library button.
- 5) Select Junit
- 6) Select JUNIT4
- 7) Finish.

## // Example for Junit

```
public class JunitEx1
{
    /* TC1 --- BE (Balance Enquiry)
     * TC2 --- NT (Money Transfer)
     * TC3 --- WD ( withdraw )
```

Openapp

```
login --- BE --- logout
login --- NT --- logout
login --- WD --- logout
```

→ @Before class // Annotation

```
public static void OpenApp()
{
    S.O.P(" logic for Openapp--");
}
```

@Afterclass

```
public static void Closeapp()
{
    S.O.P(" logic for closeapp--");
}
```

@Before

(a) after

```
public void logout()
{
    S.O.P("logic for logout");
}

@Test
public void BE()
{
    S.O.P("logic for BE --");
}

@Test @Ignore
public void NT()
{
    S.O.P("logic for NT --");
}

@Test
public void WD()
{
    S.O.P("logic for WD --");
}
```

## II Example 2. for Junit

```
public class Junit-TC
{
    static WebDriver driver;

    → @BeforeClass
    public static void Openapp()
    {
        driver=new FirefoxDriver();
        driver.get("http://OrangeHRM.uitseleium.com/");
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(20,TimeUnit.SECONDS);
    }

    → @AfterClass
    public static void Closeapp() throws InterruptedException
    {
        driver.quit();
    }
}
```

```
driver.close();
driver.quit();
}

→ @ Before

public void login()
{
    if (driver.getTitle().equals("OrangeHRM"))
    {
        System.out.println("OrangeHRM home page is displayed");
    }
}
```

// Verify UN, pwd , login button.

```
WebElement objun=driver.findElement(By.xpath(".//*[@id='txtusername']]"));
String UN="Sreekanth";
String pwd="Sreekanth@2015";
objun.clear();
objun.sendKeys(UN);
objpwd.clear();
objpwd.sendKeys(pwd);
objlogin.click();
```

// Verify title of Admin home page.

```
if (driver.getTitle().equals("OrangeHRM"))
{
    System.out.println("Orange Admin home page is displayed");
}
```

→ @ After

```
public void logout() throws InterruptedException
{
    Thread.sleep(3000);
}
```

```
// Verify title of home page.  
if(driver.getTitle().equals("OrangeHRM"))  
{  
    System.out.println("OrangeHRM home page is displayed");  
}  
}
```

→ @Test

```
public void verifyAdminLogin() throws InterruptedException  
{  
    Thread.sleep(5000);  
    String weltext=driver.findElement(By.xpath("//*[@id='...']")).getText();  
    if(weltext.equals("welcome Sreekanth"))  
    {  
        System.out.println("welcome Sreekanth is displayed");  
    }  
}
```

22/9/2016

TestNG :

↓  
is one s/w, which provides some facilities to Annotations, which makes testing job easy. People generally call it as "framework".

Features of TestNG :

- Support for Annotations
- Support for Parameterization
- Advance Execution methodology that do not require test switches to be created.
- Support for Data Driven Testing using Data
- Enables user to Set execution provider for that test methods.
- Supports Thread Safe Environment while Executing multiple threads.
- Supports integration with various tools and plug-ins like build tools - Integrated Development.
- Facilitates user with effective means of report generation using

## Navigation for installing TestNG

- 1) Google → Type install Testing.
- 2) Click on 1<sup>st</sup> link (for eclipse 3.4 & above, enter <http://beust.com/eclipse/>)
- 3) In Eclipse menu bar "HELP" → install new software
- 4) work with ~~TestNG~~ → paste 'url' here click add.
- 5) select TestNG check box
- 6) Next, Next, finish.

After installing the TestNG we have to add TestNG to library:

- 1) Right click on project
- 2) click on properties
- 3) Select java Build Path
- 4) Select library tab
- 5) Click add libraries
- 6) Select TestNG
- 7) Next → finish → OK.

Annotations for TestNG:

@BeforeClass , @ BeforeMethod , @ Test.  
@ AfterClass              @ AfterMethod

// Program for TestNG

```
public class TestNG
{
    @BeforeClass
    public void openapp()
    {
        S.O.P("logic for open application");
    }

    @AfterClass
    public void closeapp()
    {
        S.O.P("logic for close application");
    }

    @BeforeMethod
    public void loginApp()
    {
    }
}
```

#### (@ AfterMethod)

```
public void logout()  
{  
    System.out.println("logic for logout");  
}
```

@ Test (priority = 2)

```
public void BE()  
{  
    System.out.println("logic for BE");  
    Reporter.log("logic for BE");  
}
```

@ Test (priority = 1, enabled = false)

```
public void WD()  
{  
    System.out.println("logic for WD");  
    Reporter.log("logic for WD");  
}
```

@ Test (priority = 3)

```
public void MT()  
{  
    System.out.println("logic for MT");  
    Reporter.log("logic for MT");  
}
```

Priority = 1 → is used to execute in ~~any~~ required order/way.

Enabled = false

↓  
Not executed in "hide" Node.

Q) What is the difference between 'Assert' and 'Verify' Commands?

#### Assert

- 1) This command is used for checking the condition whether it is true or false.
- 2) If condition is "Satisfied" i.e., true then only cond is executed otherwise it won't execute.

#### Verify

- 1) This command is used for checking the condition true or false.
- 2) If condition is satisfied (or) not satisfied, it is not a problem for executing the condition.

→ Before Test      ↴ Particular Test will be executed.  
After Test

### Difference between Assert & Verify:

Assert :— is a Command which checks whether the given condition is true or false, Let's say the given condition is true.

Let's say assert whether the given element is present on the webpage or not. If Condition is true, then the program control will proceed the next test step but if Cond is false, then the execution would stop and no further test would be executed.

Verify :— is a command which also checks whether the given condition is true or false. Irrespective of the condition being true or false, then the program execution doesn't halts i.e., any failures during verification could not stop the execution and all the test steps are Executed.

Import static org.openqa.Assert.\*;

```
Assert.Equals();
Assert.True();
Assert.False();
Assert.NotEquals();
```

### // Example for Assert Command

```
public class TestingAssertCMDs
```

```
{  
    static WebDriver driver;  
    WebDriverWait wait;
```

@Before Test

```
public void setup()
```

```
{  
    driver = new FirefoxDriver();  
    wait = new WebDriverWait(driver, 20);  
}
```

@ After Test

```
public void shutdown() throws InterruptedException
```

```
driver.close(); // Close browser
driver.quit(); // quit browser
}

@Text
public void VerifyOrangeHRM()
{
    driver.get("http://OrangeHRM.softselections.com");
    driver.manage().window().Maximize();
    "      "      timeouts implicitlyWait(      );
    wait.until(ExpectedConditions.titleIs("OrangeHRM"));

// Verify Home page
assertEquals(driver.getTitle(), "Orange HRM");
Reporter.log("OrangeHRM home page is displayed");

// Wait for elements
WebElement objun = driver.findElement(By.xpath("//input[@type='text']"));
    "      "      "      (By.xpath(      ));
    "      "      "      (By.xpath(      ));

assertTrue(objun.isDisplayed());
Reporter.log("UN displayed");

// Verify Pwd
assertTrue(objpwd.isDisplayed());
Reporter.log("Pwd displayed");

// Verify login
assertTrue(objsubmit.isDisplayed());
Reporter.log("Login displayed");
String UN = "freekauth";
String pwd = "freetkauth@2015";

// Type Username, Password
Objun.clear();
Objun.sendKeys(UN);
Objpwd.clear();
Objpwd.sendKeys(pwd);
```

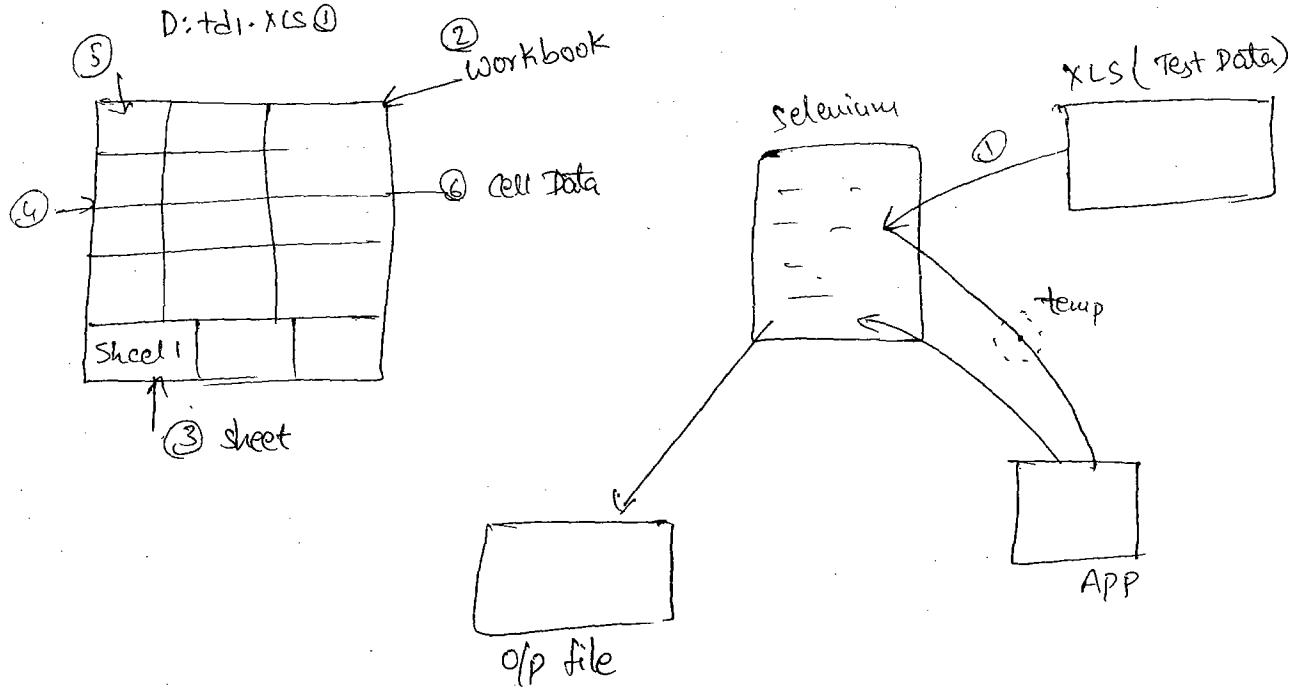
```

// Click on login
ObjSubmit.click();
String weltext = driver.findElement(By.xpath("//div[@id='welcome']")).getText();
String[] arr = weltext.split(" ");
assertEquals(arr[0], user);
Reporter.log("welcome Screen is displayed");
Thread.sleep(5000);
driver.findElement(By.xpath("//a[@id='welcome']")).click();
" " "(By.linkText("Logout")).click();
assertEquals(driver.getTitle(), "OrangeHRM");
Reporter.log("Logout is completed");
Reporter.log("OrangeHRM Home page is displayed");
driver.close();
Reporter.log("Close the Browser");
driver.quit();

```

Reporter.log("Close the Object");

### Data Driven Framework:



### Data Driven Testing | Framework:

It is a concept provided in Selenium in order to implement ~~Testing~~.

#### Resting:

↳ is a type of testing in which one will perform testing on the same functionality again & again with different sets of values in.

Generally we store the multiple sets of data in a excel file. To read data from the excel file, Apache people has developed "POI API". So we need to add the jar's of POI API to our project how we add Selenium jars and then can read the data in following way.

#### getSheetIndex():

↳ used for getting the index of any sheet by providing the SheetName.

#### getLastRowNum():

↳ is used for getting the last row number in the sheet. Here count starts with 1.

#### getLastCellNum():

↳ is used for getting the last no. of cell in that row. Here count starts with 0.

#### getSheetAt():

↳ is used for making selenium to focusing on the particular sheet based on the Index.

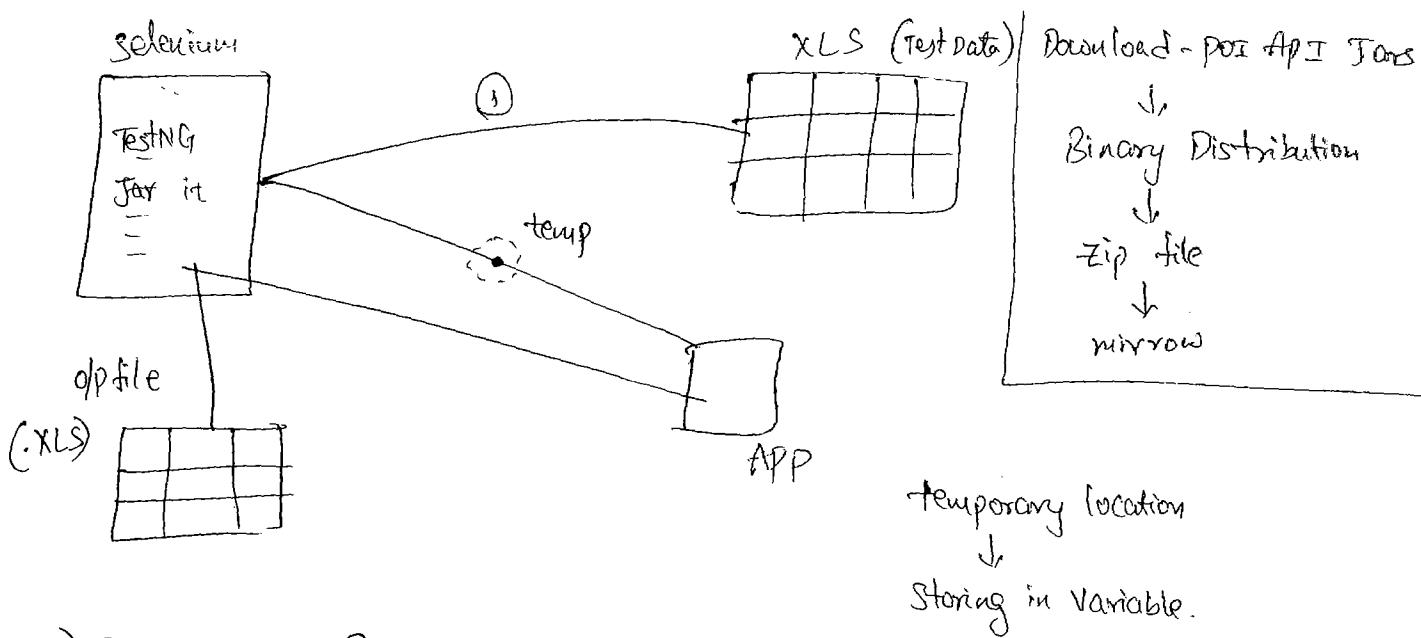
NOTE :- Above mentioned methods are few methods, but in "POI API" there are so many methods which are available and really useful for us.

- But we can also make (or) Create our own methods which are not in "POI API" then import it & can use it as you like.

Ex:- In "POI API", we don't have any method to get particular cell data based on following requirement in getting value from desired location, desired sheet, desired row & desired cell.

#### Procedure to Implement Data Driven Testing:

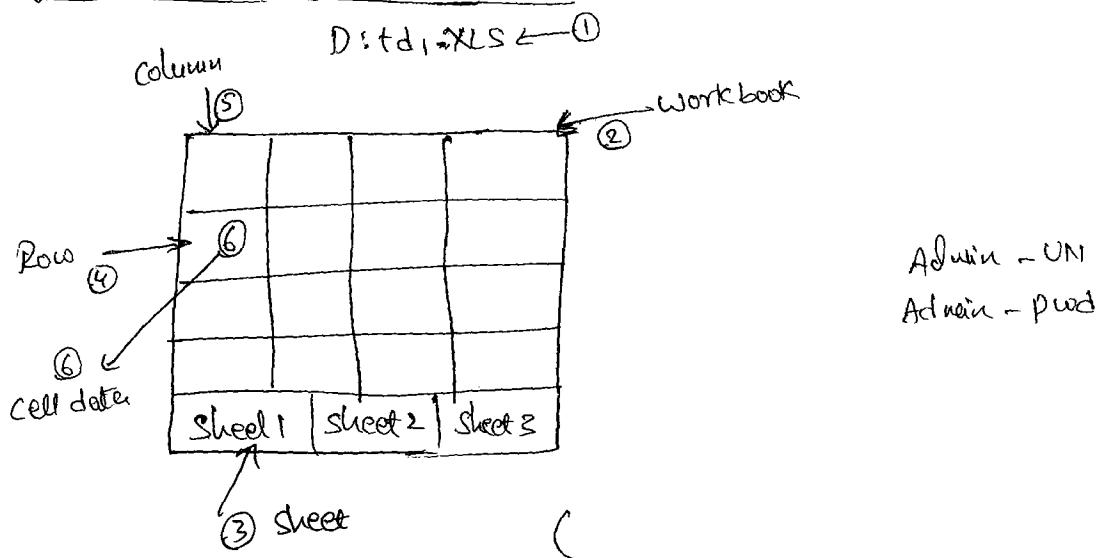
- 1) Download the POI-API Jar files
- 2) Configure the POI-API Jars into the Project.
- 3) Connect the Data into the XLS Sheet
- 4) Design the test Case.



3) Execute the Script

4) Analyse the result.

To get the Data from XLS Sheet:



24/9/2016

Data Driven Framework by using XLS:

```
Public class Basic_DDT
{
    PSVM (String[] args) throws InvalidFormatException
    {
        FileInputStream fis = new FileInputStream ("D:/td1.xls");
        Workbook wb = WorkbookFactory.create (fis);
        Sheet s = wb.getSheet ("sheet1");
    }
}
```

```

    for (int i=1; i<rc; i++)
    {
        Row r = s.getRow(i);
        int cc = r.getLastCellNum();
        for (int j=0; j<cc; j++)
        {
            Cell c = r.getCell(j);
            String str = c.getStringCellValue();
            s.o.p(str);
        }
    }
}

```

// Example for Data provider.

```

public class TestNGDDTEx
{
    @DataProvider
    public Object[][] getData() throws InvalidFormatException
    {
        Object[][] obj = new Object[6][2];
        FileInputStream fis = new FileInputStream("D:/reginfo.xls");
        Workbook wb = WorkbookFactory.create(fis);
        int st = wb.getSheetIndex("sheet1");
        Sheet s = wb.getSheetAt(st);
        int rowCount = s.getLastRowNum();
        for (int i=0; i<rowCount; i++)
        {
            Row r = s.getRow(i+1);
            int cellCount = r.getLastCellNum();
            for (int j=0; j<cellCount; j++)
            {
                Cell c = r.getCell(j);
                obj[i][j] = c.getStringCellValue();
            }
        }
        return obj;
    }
}

```

```

@("testc data provider = "getdata")
public void empcreation(String fn, String ln)
{
    WebDriver wd = new FirefoxDriver();
    wd.get("http://blaratstudent.com");
    wd.findElement(By.xpath("//html/body/div[3]/table[1]/tbody[1]")).click();
    wd.findElement(By.xpath("//*[@id='member name']")).SendKeys(fn);
    wd.close();
}

```

25/9/2016

— \* —

### Constructor:

- is used to Construct / Create a new object. The created object properties are easy to use and in optimized way.
- It doesn't have any return value.
- Name of the Constructor must be same as the name of the class.
- If you Create <sup>for</sup> class purpose then should declare reference

### Syntax:

Boy b = new Boy();

- Java will Create / generate default Constructor if there is no Constructor in the program.
- There are no parameters
- Doesn't Initialize any field.

// Example for Constructor.

```

package com.orgHrm.Scripts;
public class ConstExample
{
    public String name;
}

```

```

public ConstExample(String name, String sname)
{
    this.name = name;
    this.sname = sname;
}
public void info()
{
    System.out.println("my name is "+name);
}

```



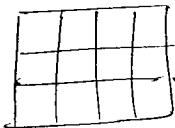
```

public class ConstConcept_TC1
{
    public static void main(String[] args)
    {
        ConstExample c = new ConstExample("Sreekanth", "Nalageula");
        c.info();
    }
}

```

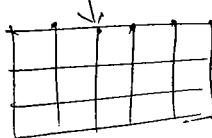
27/9/2016

Error sheet



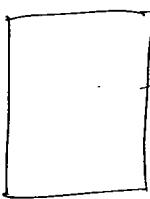
Driver Script

=  
-  
-  
-  
-

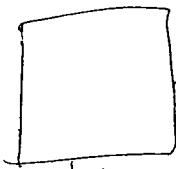


run

App



XML Reader



Constants

// Script for excel sheet (input file)

UN	Pwd	Result
Sreekanth	Sreekanth@2015	passed
Sreekanth	Sreekanth	failed
Sreekanth	Sreekanth@	failed

### Script

```
public class ExcelReader
{
    String path;
    Workbook wb;
    public XMLReader(String path) throws InvalidFormatException
    {
        this.path = path;
        FileInputStream fis = new FileInputStream(path);
        wb = workbookfactory.Create(fis);
    }
    public int getRowCount(String Sheetname)
    {
        return wb.getSheet(Sheetname).getLastRowNum() + 1;
    }
    public String cellData(String Sheetname, int rownum, int cellcol)
    {
        return wb.getSheet(Sheetname).getRow(rownum).getCell(cellcol).getStringCellValue();
    }
    public void setCellData(String Sheetname, int rownum, int cellcol, StringCellValue)
    {
        wb.getSheet(Sheetname).getRow(rownum).CreateCell(cellcol).setCellValue(CellValue);
        wb.write(new FileOutputStream(path));
    }
}

// Constants
public class Constants
{
    public static String pass = "PASS";
    public static String fail = "FAIL";
```

```

// DrivenScript
package pack1;
public class DrivenScript_TC1
{
    static String un;
    static String pwd;
    static String testdata;
    static String sheetname = "Sheet1";
    public void main(String[] args) throws InvalidFormatException
    {
        XcelReader reader = new XcelReader(system.getProperty("user.dir") + "/src/pack1/testdata.xls");
        int rc = reader.getRowCount("Sheet1");
        for(int r=1; r<rc; r++)
        {
            try
            {
                un = reader.getCellData(sheetname, r, 0);
            }
            catch(Exception e)
            {
                // e.printStackTrace();
            }
            try
            {
                pwd = reader.getCellData(sheetname, r, 1);
            }
            catch(Exception e)
            {
                // e.printStackTrace();
            }
            WebDriver driver = new FirefoxDriver();
            driver.get("http://OrangeHRM.softselenium.com");
            driver.manage().timeouts().implicitlyWait(10);
            driver.manage().window().maximize();
            driver.findElement(By.name("txtusername")).sendKeys(un);
            driver.findElement(By.name("txtusername").sendKeys(un));
            driver.findElement(By.name("txtpassword")).sendKeys(pwd);
        }
    }
}

```

```

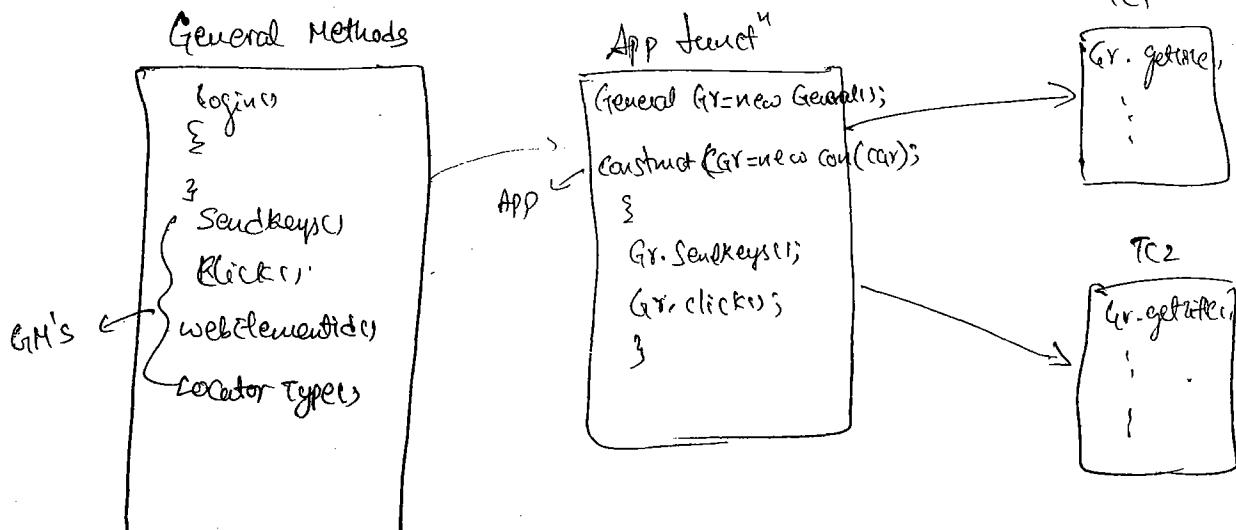
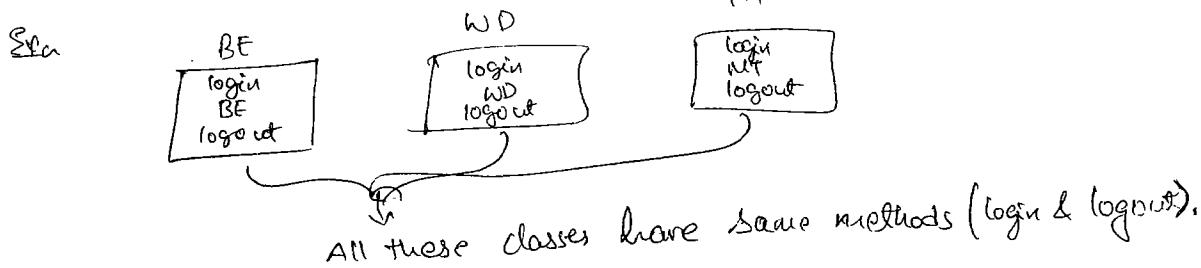
        Thread.sleep(5000);
    try {
        if(driver.findElement(By.xpath(".//*[@id='welcome']")).getText().equals("welcome
Sreekanth"));
    } else {
        XReader.setCellData(sheetname, r, 2, constants.pass);
        driver.findElement(By.xpath(".//*[ @id = 'welcome']")).click();
        ((By. LinkText ("logout"))).click();
        driver.close();
    }
}
catch(Exception e)
{
    XReader.setCellData(sheetname, r, 2, constants.fail);
    driver.close();
}

```

28/9/2016

### Modular Driver Framework:

↳ used to eliminate/avoid the duplicate code. This process is done by creating a class with all the methods that are used in our project for many times.

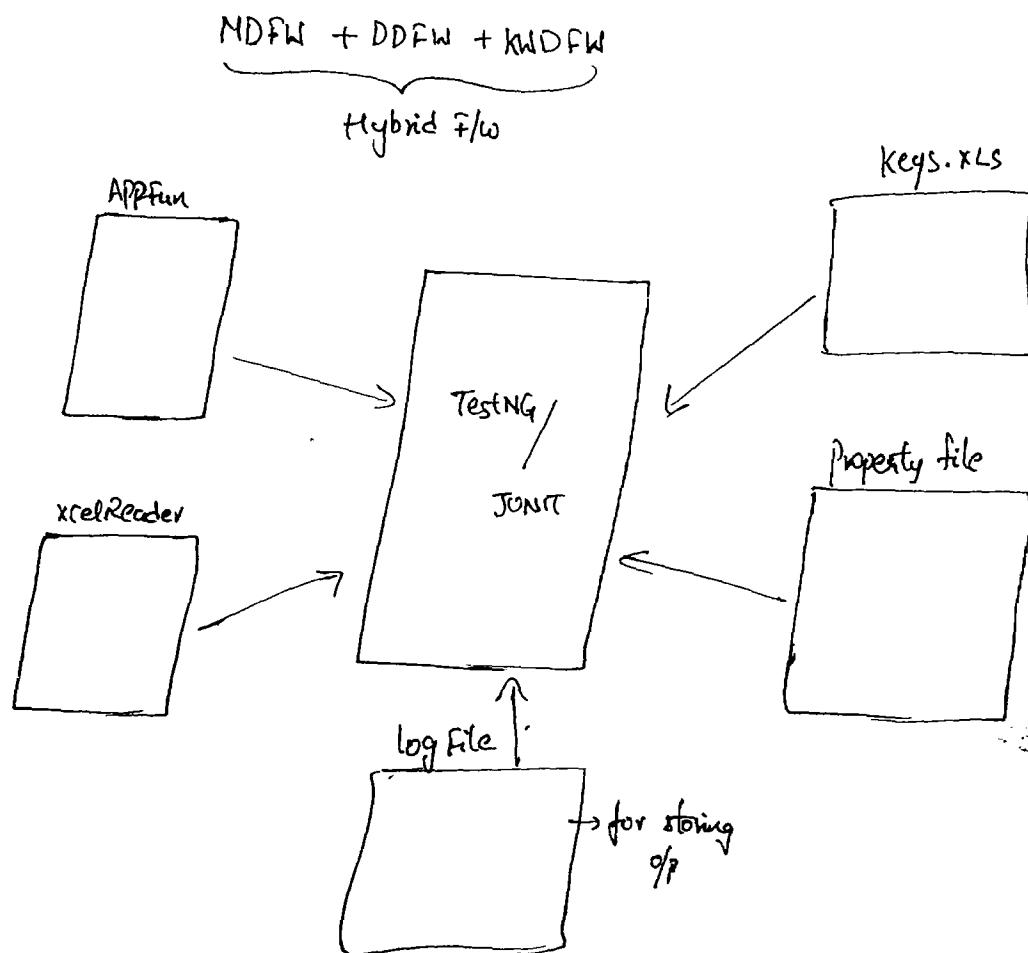


```

Row r = s.getRow(i+1);
int cellCounter = r.getLastCellIndex();
for (int j=1; j < cellCount; j++) // 1st col
{
    Cell c = r.getCell(j);
    String str = c.getStringCellValue();
    if (str.equals("Y"))
    {
        for (j=0; j < cellCount; j++) // 2nd col.
        {
            c = r.getCell(j);
            str = c.getStringCellValue();
            Method m = RefAPI.class.getMethod(str);
            m.invoke(null);
            break;
        }
    }
    else
    {
        System.out.println("Keyword is not matching");
    }
    break;
}
}

```

## Hybrid Framework:



10/10/2016

## Cross Browser Testing :

→ which is used to check multiple browsers one by one or, parallelly (at a time).

→ CBT is a process to test the web applications across multiple browsers.

→ CBT involves checking compatibility of your application across multiple web browsers and ensures that your web application works correctly across the different browsers.

// Example.

i) package com.orgbm.scripts;

```
public class CrossBrowserTesting VerifyText
{
    WebDriver driver;
    @Parameterized("browser")
    @Test
```

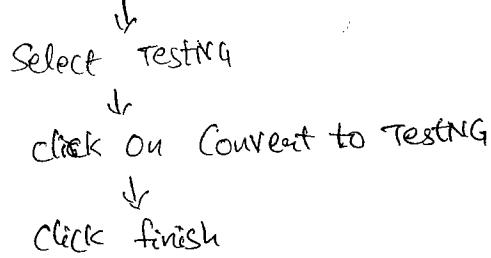
```

if (browserName.equalsIgnoreCase("firefox"))
{
    driver = new FirefoxDriver();
}
else if (browserName.equalsIgnoreCase("chrome"))
{
    System.setProperty("webdriver.chrome.driver", "E:/selenium-Dump/chromedriver.exe");
    driver = new ChromeDriver();
}
else if (browserName.equalsIgnoreCase("IE"))
{
    System.setProperty("webdriver.ie.driver", "E:/selenium-Dump/IEDriver.exe");
    driver = new InternetExplorerDriver();
}
driver.get("http://gmail.com");

```

### Navigation for CBT :-

- i) Create a class by naming VerifyText and write script as per following "above"
- ii) Right click on the above created class(VerifyText)



- iii) Open the Suite file and write as per following.

```

<Suite name="Suite" parallel="none">
  <test name="ffTest">
    <classes>
      <parameter name="browser" value="firefox"/>
      <class name="com.orgthrm.scripts.VerifyText"/>
    </classes>
  </test>
  <del test="on">
    <test name="chrome">
      <classes>

```

```

<parameter name="browser" value="chrome"/>
<class name="com.orgtum.Scripts.VerifyText"/>
</classes>
</test>
;
<test name="ie">
<classes>
<parameter name="browser" value="ie"/>
<class name="com.orgtum.Scripts.VerifyText"/>
</classes>
</test>
</suite>

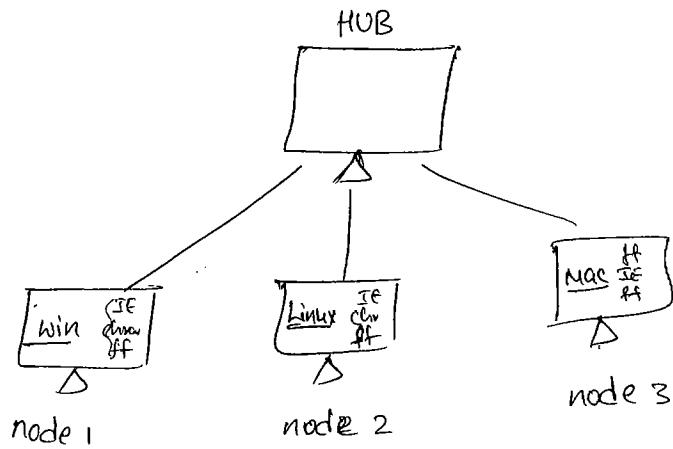
```

- Similarly we can also do for remaining browsers such as Opera, Safari, OC etc..

- iv) ~~Right click~~ Right click on VerifyTest → click on Run as  
 ↓  
 TestNG Suite

### Grid :

→ It is used for executing the Test Case parallelly into multiple browsers and multiple systems (OS).



## Procedure to Selenium Grid:

- 1) Download Selenium jar files.
- 2) Copy Paste into the HUB System and Node System.
- 3) Start the HUB
- 4) Start the Node and Register with the HUB.
- 5) Design the Test Case.
- 6) Execute the Test Case and Analyse the Result.

### Navigation to Start HUB :

- 1) Open cmd prompt
- 2) Go to seleniumhq.org and open Selenium Grid where you can find a command to start a hub.

java -jar selenium-server-standalone-2.53.0.jar --role hub

↓  
Enter the version that you installed in your system

### Navigation to Start Node and Register:

- 1) Open cmd prompt
- 2) Go to seleniumhq.org and open Selenium Grid where you can find a command to start the node.

java -jar selenium-server-standalone-2.53.0.jar -role node -hub

http://192.168.0.8:4444/grid/register

↓  
IP address

← To know your IP, In cmd prompt

↓  
Type ipconfig

### Designing Test Scripts that can Run on the Grid:

To design test scripts that will run on the grid, we need to use "Desired Capabilities" and the "Remote WebDriver" Objects.

QUESTION

Desired Capabilities :— is used to set the type of browser and OS that we will automate.

Remote WebDriver :— is used to set which node (or machine) that our test ~~will~~ will run against.

To use desired Capabilities Object, you must first import the package.

```
import org.openqa.selenium.remote.DesiredCapabilities;
```

To use Remote webDriver Object, you must import these packages.

```
import java.net.MalformedURLException;
import java.net.URL;
import org.openqa.selenium.remote.RemoteWebDriver;
```

Program for Grid.

```
public class Grid
{
    WebDriver Driver;
    String BaseURL;
    String NodeURL;
    DesiredCapabilities Capabilities;
```

@ Before Test

```
public void setup() throws MalformedURLException
{
```

```
    BaseURL = "http://Orangefarm.cftselinium.com/";
```

```
    NodeURL = "http://192.168.0.8:4444/wd/hub";
```

```
    Capability = DesiredCapabilities.firefox();
```

```
    Capability.setBrowserName("firefox");
```

```
    Capability.setPlatform(Platform.ANY);
```

```
    Driver = new RemoteWebDriver(new URL(NodeURL), Capability);
```

```
}
```

@ After Test

```

public void AtteateTest()
{
    Driver.quit();
}

@Test
public void ApplicationAccess2()
{
    Driver.get(BaseURL);
    Driver.findElement(By.name("txtusername")).Sendkeys("Sreekartha");
    Driver.findElement(By.name("txtpassword")).Sendkeys("Sreekartha@2015");
    Driver.findElement(By.name("submit")).click();
}

```

### MAVEN :

What is Maven?

- Maven is a Project Management tool which can manage the complete building life cycle.
- Maven simplifies and standardizes the project build process. By handling compilation, testing, library dependency, distribution, documentation and team collaboration.
- Maven Developers claim that Maven is more than just a build tool. We can think of Maven as a build tool with more.

### Navigation for Installing the Maven:

- Open Eclipse (Luna)
- Click on Help
- Click on Eclipse Market Place.
- Type Maven
- Find Maven for Luna
- Click install
- Click on Confirm
- Select Accept
- Click on finish.

## Creating the Maven Project:

- Click on file menu.
- Select new
- Click on other
- Open Maven folder
- Click on Maven Project
- Click next
- Select quick start
- Click next
- Enter Group ID (package name)
- Enter artifact ID (project name)
- Click on finish.

## Navigating for Adding the Dependencies into the pom.xml file:

- e.
- 1) Open Google
  - 2) Type 'Selenium webdriver Maven Dependency'
  - 3) Click on first link and copy the 'dependency files' as per following.

```
<dependency>
  <groupId> org.seleniumhq.selenium </groupId>
  <artifactId> selenium-java </artifactId>
  <version> 2.53.0 </version>
</dependency>
```

And Paste in the pom.xml file.

- 4) Similarly Other dependency files can be copied and paste in your pom.xml file.  
Like TestNG, Log4j, Junit, Selenium dependencies (above procedure)

### // Example for TestNG dependencies.

- Open Google, Type TestNG Maven Dependencies.
- Click on first link.
- 

```
<dependency>
  <groupId> org.testng </groupId>
  <artifactId> testng </artifactId>
```

```
  <scope> test </scope>
</dependency>
```

Some common dependency we write as per following.

<dependency>

<groupId> Log4J </groupId>

<artifactId> log4j </artifactId>

<version> 1.2.17 </version>

</dependency>

→ Same manner we can get required dependencies by using above process and then open pom.xml file.

In that file,

<dependencies>

{ above collected dependency files are copied and pasted here. }

</dependencies>

And then Save it → refresh the project.

→ All the jar files are created in the project.

3/10/2016

### Keyboard Actions

Handling file Keyboard Actions using selenium:

// Example 1 for Keyboard Actions

```
public class KeyboardActions
```

```
{
```

```
    public void(String[] args)
```

```
{
```

```
        WebDriver d = new FirefoxDriver();
```

```
        d.get("http://gmail.com");
```

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

```
        d.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
```

```
        driver.findElement(By.name("email")).sendKeys("tejaswini.adabala87");
```

```
        driver.findElement(By.name("Email")).sendKeys(Keys.TAB);
```

```
        driver.findElement(By.id("next")).sendKeys(Keys.Enter);
```

```
        driver.findElement(By.name("password")).sendKeys("Newlookoffice");
```

driver.findElement(By.id("sign\_in")).sendKeys(Keys.ENTER);

}

}

- To perform Ctrl+T, Ctrl+P, Ctrl+S .... (which is used for Open the New Tab, print and save)

Actions action = new Actions(d);

action.sendKeys(Keys.CONTROL).sendKeys("t").perform(); // opens New Tab

action.sendKeys(Keys.CONTROL).sendKeys("A").perform(); // selects all in page

- To perform Alt+F (It is used to open the file menu)

Actions action = new Actions(d);

action.sendKeys(Keys.ALT).sendKeys("F").perform();

- Keys enum is handled all the keyboard Actions like F1 - F12, Alt, Ctrl and so many Keyboard Actions.

### ROBOT Class :

→ It is "related to java class".

→ It is used for performing the keyboard Actions.

For Example,

```
public class Scrolling  
{
```

```
    public void (String [] args) throws InterruptedException, AWTException.
```

```
{
```

```
    WebDriver d = new FirefoxDriver();
```

```
    d.get("http://spreadsheet.com");
```

```
    Robot robot = new Robot();
```

```
    Thread.sleep(3000);
```

```
    robot.keyPress(KeyEvent.VK_DOWN);
```

```
}
```

Q) Why Sikuli?

Sikuli Automates anything you see on the screen. It uses image recognition to identify and control GUI Components. It is useful when there is no easy access to a GUI's internal (or) Source Code.

### Installation :

<http://www.sikuli.org/downloadrc3.html>

After installing Sikuli in your system in particular drive.

⇒ First → identify the "SikuliX" file → Double click on that file.  
↓  
SikuliIDE will be launching.

⇒ Record the your required actions and save it.

⇒ Click Run (P) to execute and save it.

↔ To write the Sikuli program in the Eclipse, we can follow below steps

- 1) Open the Eclipse.
- 2) Create the project, package and class.
- 3) Add Sikuli jar files (Same like WebDriver jar files)
- 4) Write the program as per following.

```
package pack;
import org.Sikuli.Script.Apps;
import org.Sikuli.Script.FindFailed;
import org.Sikuli.Script.Screen;
public class EX1
{
    public void main(String[] args) throws FindFailed
    {
        Screen S = new Screen();
        App.Open("calc.exe");
    }
}
```

```

S.click("C:/Users/Srikantu/Desktop/Calcapp1.sikuli/btn1.png");
S.click("C:/Users/Srikantu/Desktop/Calcapp1.sikuli/btn2.png");
S.click("C:/Users/Srikantu/Desktop/Calcapp1.sikuli/btn3.png");
S.click("C:/Users/Srikantu/Desktop/Calcapp1.sikuli/btn4.png");
App.close("calc.exe");
}
}

```

- In Case of Web Based application , You can add the WebDriver jar files also.

4/10/2016

// Example 2

```

public class E2
{
    public static void main(String[] args)
    {
        WebDriver d = new FirefoxDriver();
        d.get("http://orangehrm.uitselection.com");
        Screen s = new Screen();
        s.type("C:/Users/Sreekantu/Desktop/BackofDesktop/sikuli/org.lrm/un.png");
        s.type("C:/Users/Sreekantu/Desktop/BackofDesktop/sikuli/org.lrm/pwd.png");
        s.click("C:/Users/Srikantu/Desktop/BackofDesktop/sikuli/org.lrm/login.png");
        d.close();
        d.quit();
    }
}

```

Jenkins

// process for Executing Our Suite file through Cud prompt.

- 1) Create One project → package → class.
- 2) Write the your required Script by using @Test(Annotation)

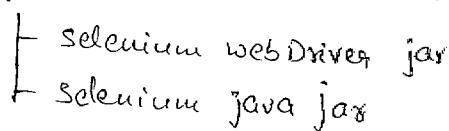
```

public class Sample
{
    @Test
    public void verification
}

```

Right click  $\rightarrow$  .class  $\rightarrow$  Select Testing  $\rightarrow$  Convert to TestNG  $\rightarrow$  Finish

4) Create the One "Lib" folder with all the jar files (which are needed)



5) Copy & paste this Lib folder into Our project location.

6) Open Cmd prompt and Come to the Our project location.

Right click  $\rightarrow$  project  $\rightarrow$  properties  $\rightarrow$  Resources  $\rightarrow$  Location  $\rightarrow$  copy & paste in cmd prompt

7) Type the Cmd as per following and click enter

Set Classpath = Path of Project dir with bin ; with lib \\*  
 $\downarrow$                           Loc of project       $\downarrow$                   bin folder       $\downarrow$                   lib folder

8) And then type the following command.

Java Org. testing. TestNG testing.xml  
 $\downarrow$   
Suite name

9) To Create ".bat" file for the Suite

i) Go to Project location  $\rightarrow$  Create One Notepad

ii) Open Notepad and write the following command and Save it as a

"run.bat".

java -cp bin;lib/\* Org. testing. TestNG testing.xml  
 $\downarrow$                           (easy way)  
Command to run the bat file

5/10/2016

JENKINS

Q) What is Jenkins?

Jenkins is an open-source Continuous integration Software tool written in Java programming language for testing and reporting an isolated changes in a large code base in real time. The JI enables developers to find and solve defects in a code base rapidly and to automate testing of their builds.

1) Go to "Google" → Type "Jenkins.io" → Click on "Download Jenkins"

↓  
Download Required War File

↓

2) Open "cmd prompt" → Type "java -jar path of Jenkins-war". Give it to required location

↑

3) Open "cmd prompt" → Type "java -jar path of Jenkins-war". Drag and drop Jenkins-war file

↓  
Click "Enter"

↓

4) Open the "Firefox browser" → Type "http://localhost:8080" → Click "go". Message displayed "Jenkins is up and running."

↓

↓  
Click "Enter"

↓

5) Open the "Firefox browser" → Type "http://localhost:8080" → Click "go". Message displayed "Jenkins is up and running."

↓

↓  
Click "Enter"

↓

6) Open "cmd prompt" → Type "java -jar path of Jenkins-war". Give it to required location

↑

↓  
Download Required War File

↓

7) Open "cmd prompt" → Type "java -jar path of Jenkins-war". Give it to required location

↓  
www.google.com for download The Jenkins:

8) Click on "Add Build Step" → Click on "Execute Windows batch Command" → Enter the ".bat" file name  
↓  
Click "Apply & Save" → Enter the "Custom workspace" → Enter "Project dir" → Click "OK" → Click on "Advanced option" → Select "Use custom workspace" → Enter "Project dir" → Click "Build Now" to execute

8) Click on Page Object Model (POM)



It is a design pattern to Create the Object Repository for Web UI.

elements (trtbox, btn, ...). Under this model for each page, each web page in an application there should be Corresponding "Page class".

This "Page class" will find the WebElements of that web page and also Contains "Page Methods" which perform Operations on these elements.

This Approach helps make the coding more "readability, maintainable, and Reusable".

// AppMethods.java

```
package pack1;
```

```
import org.openqa.selenium.support.FindBy;
```

```
public class AppMethods
```

```
{
```

```
    public WebDriver driver;
```

```
    @FindBy(xpath = ".//*[@id='txtusername'])")
```

```
    public WebElement objusername;
```

```
    @FindBy(xpath = ".//*[@id='txtpassword'])")
```

```
    public WebElement objpassword;
```

```
    @FindBy(xpath = ".//*[@id='login'])")
```

```
    public WebElement objSubmit;
```

```
    public AppMethods(WebDriver driver) // Constructor
```

```
{
```

```
        this.driver = driver;
```

```
}
```

```
    public void open(String url)
```

```
{
```

```
        driver.get(url);
```

```
}
```

UI

```
public void close() {
    driver.close();
}

public void login(String un, String pwd) {
    Objun.sendKeys(un);
    Objpwd.sendKeys(pwd);
    ObjSubmit.click();
}

public void getTitle() {
    return driver.getTitle();
}
```

## // TC1.java

```
import pack1.AppMethods;
public class TC1 {
    WebDriver driver = new FirefoxDriver();
    public AppMethods page;

    @BeforeMethod
    public void OpenBrowser() {
        page = pagefactory.initElement(driver, AppMethods.class); // ref. for Controller
        page.open("http://OrangeHRM.uit selenium.com");
    }

    @AfterMethod
    public void CloseBrowser() {
        page.close();
    }
}
```

```

public void OrangePRLogIn()
{
    page.login("Sreekanth", "Sreekanth@2015");
    String title = page.getTitle();
    S.O.P(title);
}

```

6/10/2016

## MOBILE AUTOMATION TESTING

### Types of Mobile Automation Tools:

#### 1) Appium : (only New Versions)

- ↳ test supports both iOS & Android
- ↳ Apk file is not required.
- ↳ supports multiple languages. (like Java, .Net, perl etc..)
- ↳ scalability (supports like Grid also)

( API 17 - 50+ versions  
Supported by Appium )

#### 2) Selendroid : (only Older versions)

- ↳ only Android
- ↳ Apk is required
- ↳ multiple lang (supports)
- ↳ scalability (works parallelly)

Supported by Selendroid  
( API 16 )

#### 3) iOS Driver:

#### 4) CALABASH:

⇒ Based on versions, we can choose which tool is required to use for given Version.

## Navigation for Android SDK

1) Open [SeliniumSreekarBlogsSpot.in](http://SeliniumSreekarBlogsSpot.in)

↓  
Download

↓  
Android SDK

2) Below, where you can find "platform"

↓  
Click on "Installer\_24.4.1-windows.exe"

↓  
Select 'I' box

↓  
Click on "download installer\_24.4.1-windows.exe"

3) C-Drive → Android → Tools (to use config the path)  
↓  
Copy path

↓  
System Variables (Set path)

new driver

## Screenshots

// Ex for taking screenshots

fileutils  
TakeScreenshot  
OutputType }  
}

```
public class SS
{
    public void m (String[] args) throws IOException
    {
        WebDriver driver = new FirefoxDriver();
        driver.get ("http://gmail.com");
        File scrfile = ((TakeScreenshot) driver).getScreenshotAs(OutputType.FILE);
        fileutils.Copyfile (scrfile, new File ("D:/Screenshots/SS1.png"));
    }
}
```

```

loadProperties(System.getProperty("user.dir") + "/src/Pack1/OR.properties");
XcelReader xreader = new
XcelReader(System.getProperty("user.dir") + "/src/Pack1/keywordexcel.xls");
    int rc = xreader.getRowsCount("Sheet1");
    for (int r = 1; r < rc; r++) {
        {
            flag = xreader.cellData(sheetname, r, 1);
            if (flag.equals("Y"))
            {
                try {
                    keyword = xreader.cellData(sheetname, r, 4);
                    ↑ → column no
                }
                catch (Exception e)
                {
                    //e.printStackTrace();
                }
                try {
                    testobject = xreader.cellData(sheetname, r, 5);
                    catch (Exception e)
                    {
                        //e.printStackTrace();
                    }
                try {
                   testdata = xreader.cellData(sheetname, r, 6);
                    catch (Exception e)
                    {
                        //e.printStackTrace();
                    }
                    AppFuncs kwords = new AppFuncs();
                    Method
                    m = kwords.getClass().getMethod(keyword, String.class, String.class);
                    String result = (String)
                    m.invoke(kwords, testobject, testdata);
                    xreader.setCellData(sheetname, r, 7, result);
                }
                else
                {
                    System.out.println("*****");
                }
            }
        }
    }
}

```

*(Note: getting string cell data from row in sheet column)*

*Keyword Driven Framework*

```
baseurl=http://www.gmail.com

#xpath
input_username=//input[@id='Email']
input_password=//input[@id='Passwd']
click_next=//input[@id='next']
click_login=//input[@id='signIn']
click_verify=//div[contains(text(), 'COMPOSE')

#data

click_verify=COMPOSE
data_username = testmail.n06
data_password = seleni33333
```

— XOX —

```
package Pack1;

public class Constants
{
    public static String pass="PASS";
    public static String fail="FAIL";
}
```

```
package Pack1;

import java.io.IOException;
import java.lang.reflect.InvocationTargetException;
import java.lang.reflect.Method;
import static Pack1.AppFuncs.*;

import org.apache.poi.openxml4j.exceptions.OpenXML4JException;
import Pack1.XcelReader;

public class DriverScript {
    static String keyword; ✓
    static String testobject; ✓
    static Stringtestdata; ✓
    static String flag; ✓
    static String sheetname="Sheet1";✓

    public static void main(String[] a) throws OpenXML4JException,
IOException, NoSuchMethodException, SecurityException,
IllegalAccessException, IllegalArgumentException,
InvocationTargetException{
```

```
package Pack1;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;

import org.apache.poi.openxml4j.exceptions.InvalidFormatException;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;

public class XcelReader {
    String path;
    Workbook wb; constructor

    ① public XcelReader(String path) throws InvalidFormatException,
    IOException
    {
        this.path=path; // creating instant object
        FileInputStream fis=new FileInputStream(path);
        wb=WorkbookFactory.create(fis);
    }

    ② public int getRowsCount(String sheetname){ To count the rows.
        return wb.getSheet(sheetname).getLastRowNum()+1;
    }

    ③ public String cellData(String sheetname,int rownum,int
    cellcol){ // go get the data from particular cell.
        return
        wb.getSheet(sheetname).getRow(rownum).getCell(cellcol).getStringCell
        Value();
    }

    ④ public void setCellData(String sheetname,int rownum,int
    cellcolumn,String celvalue) throws FileNotFoundException,
    IOException{ ① ↑ ② ^ ③ ~ ④
        wb.getSheet(sheetname).getRow(rownum).createCell(cellcolumn).setCellV
        alue(celvalue);

        wb.write(new FileOutputStream(path));
    }
}
```

*// creating of properties in File*

#url

-

```
        wd.get(pr.getProperty(testdata));
    } catch (Exception e) {
        e.printStackTrace();
        return fail;
    }
    return pass;
}

④ public String inputData(String testobject, String testdata) {
    try {
        WebElement el = wd
            .findElement(By.xpath(pr.getProperty(testobject)));
        el.clear();
        el.sendKeys(pr.getProperty(testdata));
    } catch (Exception e) {
        e.printStackTrace();
        return fail;
    }
    return pass;
}

public String click(String testobject, String testdata) {
    try {
        wd.findElement(By.xpath(pr.getProperty(testobject))).click();
        wd.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
    } catch (Exception e) {
        e.printStackTrace();
        return fail;
    }
    return pass;
}

public String verifyText(String testdata, String testobject) {
    try {
        String text = wd.findElement(By.xpath(pr.getProperty(testobject)))
            .getText();
        if (!text.equals(pr.getProperty(testdata))) {
            return fail;
        }
    } catch (Exception e) {
        e.printStackTrace();
        return fail;
    }
    return pass;
}
}
```

→ This deals pass to Test argument

0	1	2	3	4	5	6	7
A	B	C	D	E	F	G	H
0 TCID	STATUS	TSID	DESCRIPTION	KEYWORD	TESTOBJECT	TESTDATA	RESULT
1 TC1-LOGIN	Y	TS1	opening browser	openBrowser			PASS
2 TC1-LOGIN	Y	TS2	opening url	openUrl		baseurl	PASS
3 TC1-LOGIN	Y	TS3	enter input data	inputData	input_username	data_username	PASS
4 TC1-LOGIN	Y	TS4	click next	click	click_next		PASS
5 TC1-LOGIN	Y	TS5	enter input data	inputData	input_password	data_password	PASS
6 TC1-LOGIN	Y	TS6	enter input data	click	click_login		PASS
7 TC1-LOGIN	Y	TS7	verifying test	verifyText	click_verify	click_verify	FAIL

package Pack1;

```
import static Pack1.Constants.*;
import java.io.FileInputStream;
import java.io.IOException;
import java.util.Properties;
import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
```

public class AppFuncs {

```
    static WebDriver wd;
    static FileInputStream fip;
    static Properties pr;
```

} Declaration & Required & Reusable class.

① public static String loadProperties(String ppath) throws IOException

```
{
    fip = new FileInputStream(ppath);
    pr = new Properties();
    pr.load(fip);
    return null;
}
```

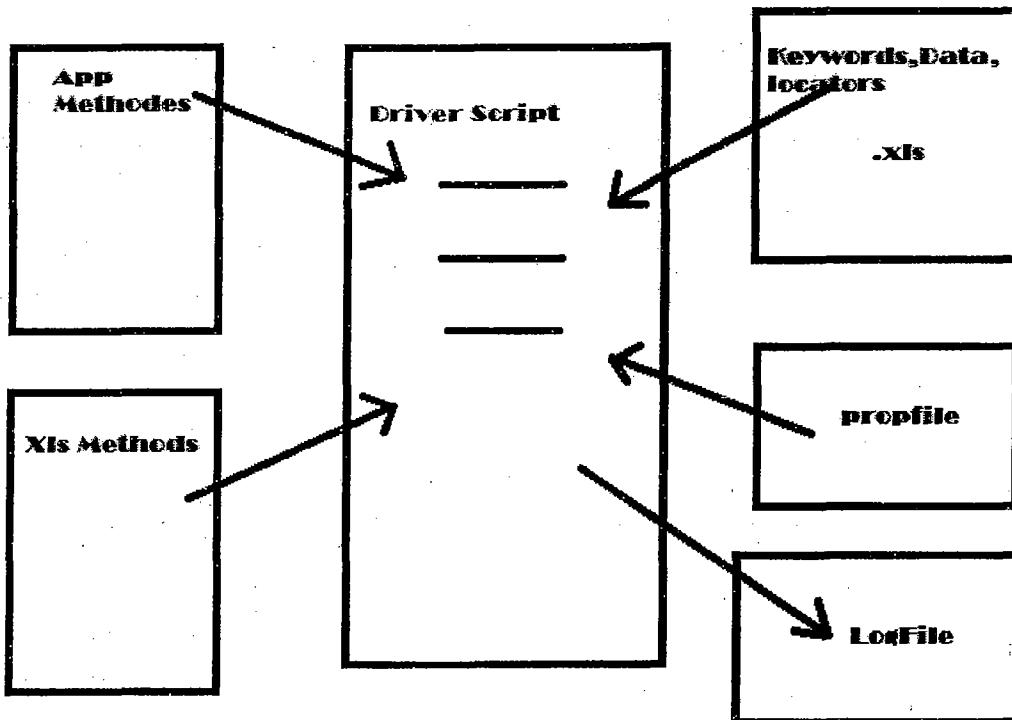
② public String openBrowser(String testobject, String testdata)

```
{
try {
    wd = new FirefoxDriver();
    wd.manage().timeouts().implicitlyWait(30,
TimeUnit.SECONDS);
}
catch (Exception e)
{
    e.printStackTrace();
    return fail;
}
return pass;
}
```

③ public String openUrl(String testobject, String testdata) {

```
try {
```

- ↳ HybridFW
  - ↳ src
    - ↳ Pack1
      - ↳ AppFuncs.java
      - ↳ Constants.java
      - ↳ DriverScript.java
      - ↳ XcelReader.java
      - ↳ keywordexcel.xls
      - ↳ OR.properties
    - ↳ JRE System Library [JavaSE-1.6]
    - ↳ Referenced Libraries
    - ↳ TestNG



## **1. What is automation and Advantages of automation**

The process of converting the manual test cases to test scripts by using any automation tool is known as Automation

Advantages:

1. It saves time by executing the test cases without manual effort
2. CTC(Cost to the company) can be saved
3. We can maintain Accuracy by repeating the same task in same manner
4. Bugs can be identified
5. We can report the bugs to the developer
6. We can ensure for quality

## **2. What are the components available in Selenium**

Selenium contains 4 components

1. Selenium IDE
2. Selenium RC
3. Selenium WebDriver
4. Selenium Grid

## **3. Why should we go for Selenium instead of QTP (or) How is Selenium different from commercial browser automation tools?**

1. Selenium is an open source automation tool
2. It supports multiple languages like Java, C#, Perl, Python, Ruby, HTML and PHP
3. It supports Firefox, IE, Google chrome, Safari and Opera
4. Supports Windows, Linux and Mac
5. Supports Web and mobile Applications
6. Its very flexible and extendable

## **4. What is Selenium IDE**

1. IDE stands for integrated Development environment.
2. It is for Record and Run the Scripts
3. Selenium IDE is an add on for Firefox
4. Its accountable for user actions
5. Recorded script can be viewed in the supported languages like HTML, JAVA, C#, Ruby, Python
6. Recorded script can be run against other browsers also by using Selenium RC or Webdriver

**5. How to capture screen shot in web driver.**

We can capture screenshot by using below two lines:

```
File f = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
 FileUtils.copyFile(f, new File("D:\\screenshot.png"));
```

**6. How to maximize the browser in web driver.**

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

**7. How to run selenium server from command prompt.**

1. Go to command prompt
2. Give the path, Where the selenium server is saved ex: cd E:Library
3. Use the command "java -jar selenium-server-standalone-2.53.0.jar"

**8. What is Selenium RC**

RC stands for Remote Control. It is a server and it launches browsers. It acts as API and library. It controls the entire automation

**9. How many test cases can be automated per day.**

It always depends on the application and test cases. But on average we can automate 5 to 6 test cases per day. Ex: 1. For analyzing the test cases 2. Developing the script 3. Debugging and executing the script 4. Stabilizing the script

**10. What challenges you have faced with Selenium.**

Challenge means problems or issues 1. Maintaining synchronization is a challenge in Selenium. 2. Handling Desktop, Flex, Flash elements is challenge with selenium. 3. Proving the user defined Results a challenge 4. Taking the data from the application is a challenge

**11. How to handle SSL Certificate issue in Firefox with Webdriver (or) How do you handle the secured connection error in HTTPS?**

```
FirefoxProfile fp = new FirefoxProfile();
fp.setAcceptUntrustedCertificates(true);
```

```
fp.setAssumeUntrustedCertificateIssuer(false);  
driver=new FirefoxDriver(profile);
```

## **12. How to handle SSL certification issue in IE ?**

Add the below command after opening the browser.

```
driver.navigate().to("javascript:document.getElementById('overridelink').click()");
```

## **13. How to handle SSL certification issue in Chrome ?**

Add the below command after opening the browser.

## **14. What is the latest version of Selenium available**

selenium-server-standalone-3.0.0.jar

## **15. How to change the port number for Selenium RC**

Syntax: -jar seleniumJarfileName -port anyFourdigitNo

Ex: -jar selenium-server-standalone-2.53.0.jar -port 1234

## **16. How do you start Selenium RC from command prompt?**

1. Save the selenium jarfile (Selenium-server-standalone-2.34.0.jar) in any folder (Ex: D:/Library)

2. Open command prompt

3. Navigate to the path where you have saved the jar file and follow the below steps

D:

cd D:/Library

java -jar selenium-server-standalone-2.34.0.jar

## **17. What locators available in Selenium RC**

ID

Name

CSS (Cascade style sheet)

XPATH (Relative xpath and Absolute xpath)

Dom

## **18. What locators available in Selenium Webdriver**

- ID
- Name
- CSS
- XPath
- Classname
- TagName
- LinkText
- Partial Link Text

## **19. How to create DefaultSelenium object and what parameters needs to be pass**

```
DefaultSelenium selenium= new DefaultSelenium("localhost",4444,"*firefox","http://");
```

localhost – It is the machine name and selenium server should be configured in the specific machine

4444 – It's selenium port number.

\*firefox – Its is the command to launch firefox

http:// – Protocol to open URL

## **20. How many types of Xpaths are available**

Xpath is two types:

1. Relative XPath
2. Absolute XPath

## **21. What is the difference between single and double slash in Xpath.**

"// " is the starting point of the Xpath.

"/" will navigate into the tag

Ex: //html/head/body/div1/div2/.....

## **22. How to handle Alerts and Confirmation Alerts in WebDriver**

We have to navigate to Alert or Confirmation as below,

```
driver.switchTo().alert()
```

To click OK or Cancel on Alert or Confirmation Alert then follow as below

```
driver.switchTo().alert().accept(); — To click OK
```

```
driver.switchTo().alert().dismiss(); — To click Cancel
```

### **23. How to Handle Popup in Webdriver**

We have to navigate to the popup as below

```
driver.switchTo().window("Window Name");
```

To navigate from Popup to main window

```
driver.switchTo().window("Main Window Name");
```

### **24. How to handle dynamically changing popup in Webdriver**

Dynamic means the Name is not constant. It changes frequently

Use the below approach to handle Dynamically changing popup

```
Set<String> s=driver.getWindowHandles();
```

```
Object popup[] = s.toArray();
```

```
driver.switchTo().window(popup[1].toString());
```

### **25. Is it possible to handle multiple popups in Webdriver**

Yes.

We can handle it by using the command

```
driver.getWindowHandles();
```

### **26. How to capture window name**

```
driver.getWindowHandle();
```

### **27. How to launch Firefox, Safari and Opera with Webdriver**

Firefox, Safari and Opera we be launch by using below commands.

```
WebDriver driver=new FirefoxDriver();
```

```
WebDriver driver=new OperaDriver();
```

```
WebDriver driver=new SafariDriver();
```

### **28. How to launch InternetExplorer.**

For IE, we need the supported “Internet Explorer Driver Server”. It can be downloaded from the below site:

<http://docs.seleniumhq.org/download/>

Below code is to launch IE.

```
System.setProperty("webdriver.ie.driver", "E:\\Library\\IEDriverServer.exe");
driver=new InternetExplorerDriver();
driver.get("http://gmail.com");
```

## 29. How to launch GoogleChrome

For GoogleChrome, We need the supported “ChromeDriver” exe file. Can be downloaded from the below site:

<http://code.google.com/p/chromedriver/downloads/list>

Below code is to launch GoogleChrome

```
System.setProperty("webdriver.chrome.driver", "E:\\Library\\chromedriver.exe");
driver=new ChromeDriver();
driver.get("http://gmail.com");
```

## 30. Brief about the Disadvantages of Selenium

Selenium supports Web applications. It doesn't support Desktop applications

## 31. How to handle Desktop, Flex and Flash objects in Selenium.

We can use SIKULI as a add-on for Selenium to support Desktop, Flex and Flash objects

Sikuli is an open source automation tool developed on JAVA

Can be downloaded from:<https://launchpad.net/sikuli/+download>

## 32. How to take data from excel(xls) file

```
FileInputStream fi=new FileInputStream("Path of the excel file");
```

```
Workbook w=Workbook.getWorkbook(fi);
```

```
Sheet s=w.getSheet(0);
```

It will read the excel file upto the sheet. To take the data from sheet, use below command,

```
s.getCell(columnID, rowID).getContents()
```

```
ex: s.getCell(0, 1).getContents();
```

## 33. How to create Excel file

```
FileOutputStream fo=new FileOutputStream("Path to create xls file");
```

```
WritableWorkbook wb=Workbook.createWorkbook(fo);
```

```
WritableSheet ws=wb.createSheet("Results", 0);
```

```
Label un=new Label(0,0,"Username");
ws.addCell(un);
wb.write();
wb.close();
```

#### **34. How to print data from notepad(txt file)**

```
File f=new File("E:\\data2.txt");
FileReader fr=new FileReader(f);
BufferedReader br=new BufferedReader(fr);
String str;
while((str=br.readLine())!=null)
{
System.out.println(str);
}
```

#### **35. How to create and write data into txt file**

```
File f=new File("E:\\data2.txt");
FileWriter fw=new FileWriter(f);
BufferedWriter bw=new BufferedWriter(fw);
bw.write("Nagesh");
bw.newLine();
bw.write("hyderabad");
bw.newLine();
bw.write("AP");
bw.close();
fw.close();
```

#### **36. What is Ant or Apache Ant**

Ant is a Java based build management tool.

The process of converting the source code to software system (executable code) is known as build.

There are number of steps to convert the source code to executable code. Ant will manages all the steps with Build.xml file.

Steps to convert the Source code to software:

1. Taking the source code from src repository
2. Preparing a build area

3. Compiling the source code
4. Build the compiled code to executable code

### **37. What is Maven**

Maven is also Java based build management tool. It is the advanced version of Ant.

### **38. What is the differences between Ant and Maven**

1. Maven is widely preferred than Ant. Ant is an older tool.
2. Ant doesn't come with formal conventions such as a common project directory.  
Maven consists of conventions.
3. Ant is procedural, Maven is declarative. It means in Ant, you have to specify the order what should have to be done, whereas Maven takes care of all the directories once the files are stored in the pom.xml file.
4. Another difference that can be seen is that Ant does not have a life cycle whereas Maven has a life cycle.
5. The scripts in Ant are not reusable whereas Maven comes with reusable plugins.

### **39. What are the disadvantages of using Selenium as testing tool**

It doesn't support desktop or windows based applications.

But we can overcome by using the tool SIKULI

### **40. How do you handle Ajax controls using selenium? i. Eg. By typing in search engine how do you capture the auto suggestion**

Ajax control means when we enter some text in the google text box, it displays the auto suggested values. That text box is an ajax control.

Type the value in the textbox and capture all the suggested values in a string.

Split the string and take the values

### **41. How do you select the 2nd item in a List box or drop down.**

```
new Select(driver.findElement(By.id("id"))).selectByIndex(index);
```

### **42. Brief about your framework**

Its Hybrid framework where its a combination of Keyword driven (TestNG) and DataDriven framework.

Advantage of Hybrid framework is, we can utilize the advantages of both Keyword driven and Datadriven frameworks

**43. What is the difference between assert and Verify Commands?**

**Verify** command will not stop execution of test case if verification fails. It will log an error and proceed with execution of rest of the test case.

**Assert** command will stop execution of test case if verification fails. It will log an error and will not proceed with execution of rest of the test case

**44. Explain about your reporting method**

Bug Reporting is always a manual process. Getting the test report from the automation tool and analyze the report for bugs. If identified, report the bugs by using QC or Bugzilla or JIRA

**45. What is the difference between Selenium RC and Webdriver**

1. We can access all the latest browsers
2. Not necessary to start Selenium server to run the webdriver programs
3. Webdriver by default maintains page load synchronization, for page refresh we need to handle it
4. Its having auto scroll down action into the application while executing the script
5. We can develop both webdriver commands and RC in a single class
6. We can effectively take the information from application

**46. What are all things can not be done through selenium IDE**

We can't perform below testings

1. Regression testing
2. Retesting
3. Static Testings like GUI and Validations
4. DB Testing
5. It doesn't provide Test Report

**47. Brief about Selenium Grid.**

It is for parallel execution. We can launch all the browsers parallelly and the scripts can be executed on it parallelly

#### **48. How to use selenium for performance testing**

Selenium is functional testing automation tool. Its not for performance testing. We can generate only the load by using Selenium.

#### **49. How to get all the links in http://google.co.in**

Link means anchor tag in any application.

Identify all the links by using FindElements and tagname.

Get all the links to List variable and take the targetted link by using for loop and if condition

#### **50. Is it possible to handle multiple popups in selenium?**

Yes we can handle it with the command getWindowHandles().

Take all the window names into Set<String> variable and convert it to array.

Use the array index to navigate to specific window by using  
driver.switchTo().window(ArrayIndex);

#### **51. Difference between Junit and TestNg framework**

##### JUnit Vs TestNG

1. We need to implement all the test methods very independently in Junit. Not necessary in TestNg
2. JUnit contains very limited annotations like @Beforeclass, @Before, @Test, @After and @AfterClass. TestNG contains multiple annotations like @BeforeSuite, @BeforeTest, @Beforeclass, @BeforeMethod, @Test, @AfterMethod @AfterClass, @AfterTestand @AfterSuite
3. TestNG provides html result file by default where as in JUnit it wont provide html result file by default

#### **52. If the default port of selenium is busy then which port you use?**

Selenium by default uses 4444 port no. If the port is already used by any other server then we can change it to any other 4 digit no.

Ex: 5555 or 1234 or 6666

### **53. How much time we can save with automation**

It always depends on the project and test cases. But on average we can save around 50% to 60% of the time.

### **54. What is automation Lifecycle or Automation approach or automation plan**

1. Do the POC for the project
2. Execute or review all the manual test cases to get the functional knowledge of the project and to identify the test cases for automation
3. Develop the framework
4. Prepare the test scripts for all the identified test cases
5. Integrate the scripts with Framework for execution
6. Before executing identify the build changes, if available update the scripts
7. Execute the scripts and capture all the results in separate folder
8. Analyze the test report and identify the bugs
9. Report the bugs manually to the developer by using some reporting tool

### **55. Write a program to get all the text boxes in mail.in register page with webdriver**

```
List<WebElement> str=driver.findElements(By.tagName("textbox tagname"));  
System.out.println(str.size());  
for (int i = 0; i < str.size(); i++) {  
    System.out.println(str.get(i).getAttribute("id")); //To print ids  
}
```

### **56. How to execute scripts on other machines Remotely**

```
URL url = new URL( "http", "localhost", 4444, "/wd/hub" );  
DesiredCapabilities capabilities =DesiredCapabilities.internetExplorer();  
capabilities.setJavascriptEnabled(true); WebDriver driver = new  
RemoteWebDriver(url,capabilities); driver.get("http://www.google.com");  
Note: Provide IP address of other machines instead of localhost
```

### **57. How to Mouse over on one element by using webdriver**

```
WebDriver driver=new FirefoxDriver();  
driver.manage().window().maximize();  
driver.get("http://spicejet.com");
```

```
Actions a=new Actions(driver);
WebElement str=driver.findElement(By.linkText("About Us"));
a.moveToElement(str).build().perform();
```

## 58. How to Scroll down and Scroll up in the browser

```
WebDriver driver=new FirefoxDriver();
driver.manage().window().maximize();
driver.get("http://spicejet.com");
Actions a=new Actions(driver);
a.keyDown(Keys.CONTROL).sendKeys(Keys.END).build().perform();
Thread.sleep(1000);
a.keyDown(Keys.CONTROL).sendKeys(Keys.HOME).build().perform();
a.keyDown(Keys.ALT).sendKeys(Keys.F1).build().perform();
```

## 1 ) What Is Selenium WebDriver/Selenium 2?

Selenium WebDriver software testing tool is well designed object oriented API which is developed to automate web and mobile applications testing process. WebDriver API is bigger than Selenium RC but its Architecture is simple and easy to understand compared Selenium RC API.

We can automate our web application's software testing process using selenium webdriver.

We can say it is advanced version of selenium RC software testing tool because some limitations of selenium RC has been overcome in selenium WebDriver software testing tool.

WebDriver is designed to provide better support for dynamic changing pages. Example : Web page elements of software web application is changing without reloading the page. In this case WebDriver works better.

Selenium Webdriver software testing tool is more faster than Selenium RC software testing tool as it is directly interacting with web browsers and mimic the behavior of a real user. Example : User clicks on button of web page or moving mouse on main menu to get the sub menu list. WebDriver works same.

All popular browser vendors are active participants in selenium WebDriver's development and all of them have their own engineers team to improve this framework.

You can include answers of question 2, question 3, question 4 and question 5 in answer of this question if interviewer need more detail on selenium webdriver.

## **2 : Tell Me WebDriver Supported Browsers?**

Selenium WebDriver API has a many different drivers to test your web application In different browsers. List of Webdriver browser drivers are as bellow.

Selenium WebDriver Supported Browsers

Firefox Driver - For Mozilla Firefox browser

Internet Explorer Driver - For Internet Explorer browser

Chrome Driver - For Google Chrome browser

HtmlUnit Driver - GUI-Less(Headless) browser for Java programs

Opera Driver - For Opera browser

## **3 : Tell Me WebDriver Supported Mobile Application Testing Drivers?**

We can get support of mobile software application testing using Selenium webdriver. Selenium WebDriver supports bellow given drivers to test mobile application.

selenium supported mobile app testing drivers

AndroidDriver

OperaMobileDriver

iPhoneDriver

## **4 : Which Programming Languages Supported By Selenium WebDriver To Write Test Cases?**

Selenium WebDriver Is very wast API and It support many different languages to write test cases for your software web application. List of WebDriver supported languages are as bellow.

Selenium supported languages

Java , C# , Python , Ruby ,Perl, PHP

## **5 : Which Different Element Locators Supported By Selenium WebDriver?**

Selenium WebDriver supports below given element locators.

Selenium supported element locators

XPath Locator -> CSSSelector Locator -> ClassName Locator -> ID Locator -> Name Locator ->  
LinkText Locator -> PartialLinkText Locator -> TagName Locator ->

## **6 : What are the benefits of automation testing ?**

We can get below given benefits If automate our software testing process.

**Fast Test Execution :** Manual software testing process Is time consuming. Automation tests are faster and takes less time to execute tests compared to manual test execution.

**Re-usability Of Test Cases :** You need to prepare automation test cases only one time. Then you can use same test cases for all upcoming version release of software application. However you need to modify your test cases If there Is any flow change of business logic changes In software. But It Is less time consuming.

**Testing Cost Reduction :** You have to put human efforts only one time to automate your software test process. Latter on automation tool will work for you at place of human resource.

**Better Test Coverage In Each Version Release:** You have to Implement test scenarios only once In your automation test cases. Latter on you can execute same test cases In all upcoming release. So each scenarios will be tested In every version release.

**Easy For Compatibility Testing :** It Is easy to run same tests In combination of different OS and browser environments using automation tools.

## **7 : Does Selenium WebDriver Support Record And Playback Facility?**

No. WebDriver do not have any record and playback facility. But you can record your tests In one of the selenium version called Selenium IDE and then you can export your recorded tests In webdriver compatible format as per your preferred language.

## **8 : Which Operating systems support Selenium WebDriver?**

At present, Mainly below given operating systems support Selenium WebDriver.

Windows - Windows XP, Windows 7, Windows 8 and Windows 8.1

Apple OS X

Linux - Ubuntu. Other versions of linux should support too.

### **9 : Selenium WebDriver Is Paid Or Open Source Tool? Why you prefer to use It?**

All versions of selenium software testing tool are open source. You can use any version of selenium In free of charge.

I choose to use It because

Open Source.

It has multi-browser support.

Multi-OS support.

Multi types of locators support. So If one not works, We can use another type.

Web as well mobile application testing support.

Many testers are using selenium WebDriver to automate their testing process. So getting solution of any complex Issue very easily on Internet.

It Is extendable and flexible.

Continues support from WebDriver's development team to Improve the API and resolve current Issues.

### **10 : Which OpenSource Framework Is Supported In WebDriver With Java?**

Bellow given 2 java frameworks are supported by selenium WebDriver.

JUnit

TestNG

### **11 : Can you tell me the syntax to open/launch Firefox browser In WebDriver software testing tool?**

We can open new Mozilla Firefox browser Instance using bellow given syntax In WebDriver software testing tool.

```
WebDriver driver = new FirefoxDriver();
```

### **12 : What Is XPath and what Is use of It In WebDriver?**

In Selenium WebDriver software testing tool, XPath is used to locate the elements. Using XPath, We can navigate through elements and attributes In an XML document to locate software webpage elements like buttons, text box, links, Images etc..

### **13 : Which tool you are using to find the XPath of any element?**

I am using Mozilla Firefox AddOns FireBug and FirePath to find the XPath of software web elements.

### **14 : What is the difference between absolute XPath and relative XPath?**

**Absolute XPath** : Absolute XPath Is the full path starting from root node and ends with desired descendant element's node. It will start using single forward slash(/) as below.

Example Of Absolute XPath :

```
/html/body/div[3]/div[2]/div[2]/div[2]/div[2]/div[2]/div/div[4]/div[1]/div/div/div/div[1]/div/div/div[1]/div[2]/form/table/tbody/tr[1]/td/input
```

Above XPath Is absolute XPath of calc result box given on THIS PAGE. It starts top node html and ends with input node.

**Relative XPath** : Instead of starting from root node, Relative XPath starts from any In between node or current element's node(last node of element). It will start using double forward slash(//) as below.

Example Of Relative XPath :

```
//input[@id='Resultbox']
```

Above XPath Is relative XPath of same calc result box given on THIS PAGE.

### **15 : How To Handle Dynamic Changing IDs In XPath.**

Example : //div[@id='post-body-3647323225296998740']/div[1]/form[1]/input[1]

In this XPath "3647323225296998740" Is changing every time when reloading the page.  
How to handle this situation?

There are many different alternatives In such case.

**Alternative 1** : Look for any other attribute which Is not changing every time In that div node like name, class etc. So If this div node has class attribute then we can write xpath as below.

```
//div[@class='post-body entry-content']/div[1]/form[1]/input[1]
```

**Alternative 2** : You can use absolute xpath(full xpath) where you not need to give any attribute names In xpath.

```
/html/body/div[3]/div[2]/div[2]/div[2]/div[2]/div[2]/div/div[4]/div[1]/div/div/div[1]/div/div/div[1]/div[2]/div[1]/form[1]/input[1]
```

**Alternative 3 :** Use starts-with function. In this xpath's ID attribute, "post-body-" part remain same every time. So you can use xpath as bellow.

```
//div[starts-with(@id,'post-body-')]/div[1]/form[1]/input[1]
```

**Alternative 4 :** Use contains function. Same way you can use contains function as bellow.

```
div[contains(@id,'post-body-')]/div[1]/form[1]/input[1]
```

## **16 : How to press ENTER key button on text box In selenium webdriver?**

To press ENTER key using selenium WebDriver software automation tool, We need to use selenium Enum Keys with Its constant ENTER as bellow.

```
driver.findElement(By.xpath("//input[@id='gbqfq']")).sendKeys(Keys.ENTER);
```

## **17 : How many types of waits available In selenium WebDriver**

There are two types of waits available In selenium WebDriver software automation testing tool.

Implicit Wait

Explicit Wait

## **18 : What Is Implicit Wait In Selenium WebDriver?**

Sometimes, Elements are taking time to be appear on software web application page. Using Implicit wait In webdriver software testing test case, We can poll the DOM for certain amount of time when some element or elements are not available Immediately on webpage.

**Implicit Wait Example :**

```
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
```

If you will write above syntax In your test, Your WebDriver test will wait 10 seconds for appearing element on page.

## **19 : What Is Explicit Wait In Selenium WebDriver?**

Using explicit wait code In selenium webdriver software automation testing tool, You can define to wait for a certain condition to occur before proceeding further test code execution.

### **Explicit Wait Example :**

```
WebDriverWait wait = new WebDriverWait(driver, 20);  
wait.until(ExpectedConditions.elementToBeClickable(By.xpath("//input[@id='gbqfq']")));
```

Above code will wait for 20 seconds for targeted element to be displayed and enabled or we can say clickable.

## **20 : I wants to pause my test execution for fix 10 seconds at specific point. How can I do It?**

You can use `java.lang.Thread.sleep(long milliseconds)` method to pause the software test execution for specific time. If you wants to pause your test execution for 10 seconds then you can use bellow given syntax In your test.

```
Thread.sleep(10000);
```

## **21 : How does selenium RC software testing tool drive the browser?**

When browser loaded In Selenium RC, It ‘injected’ javascript functions into the browser and then It Is using javascript to drive the browser for software application under test.

## **22 : How does the selenium WebDriver drive the browser?**

Selenium webdriver software testing tool works like real user Interacting with software web page and Its elements. It Is using each browser's native support to make direct calls with browser for your software application under test. There Is not any Intermediate thing In selenium webdriver to Interact with web browsers.

## **23 : Do you need Selenium Server to run your tests In selenium WebDriver?**

It depends. If you are using only selenium webdriver API to run your tests and you are running your all your tests on same machine then you do not need selenium server because In this case, webdriver can directly Interact with browser using browser's native support.

You need selenium server with webdriver when you have to perform bellow given operations with selenium webdriver.

When you are using remote or virtual machine to run webdriver tests for software web application and that machine have specific browser version that is not on your current machine.

When you are using selenium-grid to distribute your webdriver's test execution on different remote or virtual machines.

**24 : Below given syntax will work to navigate to specified URL In WebDriver? Why?**

```
driver.get("www.google.com");
```

No. It will not work and show you an exception like : "Exception in thread "main" org.openqa.selenium.WebDriverException: f.QueryInterface is not a function" when you run your test.

You need to provide http:// protocol with URL In driver.get method as below.

```
driver.get("http://www.google.com");
```

Now It will work.

**25 : Tell me a reason behind below given WebDriver exception and how will you resolve It?**

"Exception in thread "main" org.openqa.selenium.NoSuchElementException: Unable to locate element"

You will get this exception when WebDriver Is not able to locate element on the page of software web application using whatever locator you have used In your test. To resolved this Issue, I will check below given things.

First of all I will check that I have placed Implicit wait code In my test or not. If you have not placed Implicit timeout In your test and any element Is taking some time to appear on page then you can get this exception. So I will add below given line at beginning of my test case code to wait for 15 seconds for element to be present on page. In 70% cases, this step will resolved Issue. View Practical Example Of Implicit Wait.

```
driver.manage().timeouts().implicitlyWait(15, TimeUnit.SECONDS);
```

Another reason behind this Issue Is element's ID Is generated dynamically every time when reloading the page. If I have used element's ID as an element locator or used It In xpath to locate the element then I need to verify that ID of element remains same every time or It Is changing? If It Is changing every time then I have to use alternative element locating method. In 20% cases, This step will resolve your Issue..

If Implicit wait Is already added and element locator Is fine then you need to verify that how much time It(element) Is taking to appear on page. If It Is taking more than 15 seconds then you have to put explicit wait condition with 20 or more seconds wait period as below. In 5 to 10% cases, This step will resolve your Issue..

```
WebDriverWait wait = new WebDriverWait(driver, 25);
```

```
wait.until(ExpectedConditions.elementToBeClickable(By.cssSelector("#submitButton")));
```

**26 : Can we automate desktop software application's testing using selenium WebDriver?**

No. This is the biggest disadvantage of selenium WebDriver API. We can automate only web and mobile software application's testing using selenium WebDriver.

**27 : Can you tell me the alternative driver.get() method to open URL in browser?**

We can use anyone from below given two methods to open URL in web browser in selenium webdriver software testing tool.

```
driver.get()
```

```
driver.navigate().to()
```

**28 : Can you tell me a difference between driver.get() and driver.navigate() methods?**

Main and mostly used functions of both methods are as below.

```
driver.get()
```

driver.get() method is generally used for Open URL of software web application.

It will wait till the whole page gets loaded.

```
driver.navigate()
```

driver.navigate() method is generally used for navigate to URL of software web application, navigate back, navigate forward, refresh the page.

It will just navigate to the page but wait not wait till the whole page gets loaded.

**29 : WebDriver has built in Object Repository. Correct me if I am wrong.**

No. WebDriver do not have any built in object repository till now. But yes, I am using java .properties file in my framework to store all required element objects in my tests. .

**30 : Can you tell me syntax to set browser window size to 800(Width) X 600(Height)?**

We can set browser window size using setSize method of selenium webdriver software testing tool. To set size at 800 X 600, Use below given syntax in your test case.

```
driver.manage().window().setSize(new Dimension(500,500));
```

**31 : Can you tell me the names of different projects of selenium software automation testing tool?**

At present, Selenium software automation testing tool has four different projects as bellow.

**Selenium IDE** : It Is Firefox add-on which allows you to record and playback your software web application's tests In Firefox browser.

**Selenium RC** : It Is software web application automation tool which allows you to write your tests In many different programming languages.

**Selenium WebDriver** : It Is well designed object oriented API developed to automate web and mobile software application testing process. You can write your tests In different languages too In selenium webdriver.

**Selenium Grid** : Grid allows you to execute your tests In parallel by distributing them on different machines having different browser/OS combinations.

**32 : I wants to use java language to create tests with Selenium WebDriver. Can you tell me how to get latest version of WebDriver?**

You can download language specific latest released client drivers for selenium WebDriver software testing tool at <http://docs.seleniumhq.org> official website. For java language, You will get bunch of jar files In zip folder. And then you can add all those jar files In your project's java build path as a referenced libraries to get support of webdriver API.

**33 : Can you tell me the usage of "submit" method In selenium WebDriver?**

We can use submit method to submit the forms In selenium WebDriver software automation testing tool. Example : Submitting registration form, submitting LogIn form, submitting Contact Us form ect.. After filling all required fields, We can call submit method to submit the form. .

**34 : Do you have faced any Issue with "submit" method any time?**

Yes, I have faced Issue like submit method was not working to submit the form. In this case, Submit button of form was located outside the opening `<form>` and closing `</form>` tags. In this case submit method will not works to submit the form.

Also If submit button Is located Inside opening `<form>` and closing `</form>` tags but that button's type tag's attribute Isn't submit then submit method will not work. It(type tag's attribute) should be always submit.

**35 : What Is the syntax to type value In prompt dialog box's Input field using selenium WebDriver?**

Prompt dialog Is just like confirmation alert dialog but with option of Input text box as bellow.

To Input value In that text box of prompt dialog, You can use bellow given syntax.

```
driver.switchTo().alert().sendKeys("Purushotham");
```

**36 : When I am running software web application's tests In Firefox Browser using selenium webdriver, It Is not showing me any bookmarks, addons, saved passwords etc. In that browser. Do you know why?**

Yes. It Is because all those bookmarks, addons, passwords etc.. are saved In your regular browser's profile folder so when you launch browser manually, It will use existing profile settings so

It will show you all those stuffs. But when you run your software web application's tests In selenium webdriver, It Is opening new browser Instance with blank/new profile. So It will not show you bookmarks and all those things In that browser Instance.

You can create custom firefox profile and then you can use It In selenium webdriver test. In your custom profile, you can set all required bookmarks, addons etc.. [VIEW THIS EXAMPLE](#) to know how to set custom profile of firefox browser.

**37 : Arrange bellow given drivers In fastest to slowest sequence?**

Firefox Driver, HtmlUnit Driver, Internet Explorer Driver.

HTMLUnit Driver Is faster than all other drivers because It Is not using any UI to execute test cases of software web application. Internet Explorer driver Is slower than Firefox and HtmlUnit driver. So, Fastest to slowest driver sequence Is as bellow.

HtmlUnit Driver , Firefox Driver and Internet Explorer Driver

**38 : What Is Ajax?**

Asynchronous JavaScript and XML Is full form of AJAX which Is used for creating dynamic web pages very fast for software web applications. Using ajax, We can update page behind the scene by exchanging small amounts of data with server asynchronously. That means, Using ajax we can update page data Without reloading page.

**39 : How to handle Ajax In selenium WebDriver?**

Generally we are using Implicit wait In selenium WebDriver software automation tests to wait for some element to be present on page. But Ajax call can not be handled using only Implicit wait In your test because page not get reloaded when ajax call sent and received from server and we can not assume how much time It will take to receive ajax call from server.

To handle ajax call In selenium WebDriver software automation tests, We needs to use webdriver's FluentWait method or Explicit Waits which can wait for specific amount of time with specific condition. You can get different Explicit Waits examples links on THIS PAGE.

**40 : On Google search page, I wants to search for some words without clicking on Google Search button. Is It possible In WebDriver? How?**

Yes we can do It using WebDriver sendKeys method where we do not need to use Google Search button. Syntax Is as bellow.

```
driver.findElement(By.xpath("//input[@id='gbqfq']")).sendKeys("Search Syntax",Keys.ENTER);
```

In above syntax, //input[@id='gbqfq'] Is xPath of Google search text field. First It will enter "Search Syntax" text In text box and then It will press Enter key on same text box to search for words on Google.

**41 : What kind of software testings are possible using selenium WebDriver? We can use It for any other purpose except testing activity?**

Generally we are using selenium WebDriver for functional and regression testing of software web applications.

**42 : Give me any five different xPath syntax to locate bellow given Input element.**

```
<input id="fk-top-search-box" class="search-bar-text fk-font-13 ac_input" type="text" autofocus="autofocus" value="" name="q" />
```

Five xPath syntax for above element of software web application page are as bellow.

```
//input[@id='fk-top-search-box']
```

```
//input[contains(@name,'q')]
```

```
//input[starts-with(@class, "search-bar-text")]
```

```
//input[@id='fk-top-search-box' or @name='q']
```

```
//input[starts-with(@id, 'fk-top-search-box') and contains(@class,'fk-font-13')]
```

**43 : Can you tell me two drawbacks of xPath locators as compared to cssSelector locator?**

Two main disadvantage of xPath locator as compared to cssSelector locator are as bellow.

It Is slower than cssSelector locator.

xPath which works In one browser may not work In other browser for same page of software web application because some browsers (Ex. IE) reads only Lower-cased tag name

and Attribute Name. So If used It In upper case then It will work In Firefox browser but will not work In IE browser. Every browser reads xPath In different way. In sort, do not use xPath locators In your test cases of software web application If you have to perform cross browser testing using selenium WebDriver software testing tool.

**44 : Why xPath locator Is much more popular than all other locator types In WebDriver?**

xPath locators are so much popular In selenium webdriver test case development because It Is very easy to learn and understand for any new user.

There are many functions to build xPath In different ways like contains, starts-with etc.. So If one Is not possible you will have always another option to build xPath of any element.

Presently many tools and add-ons are available to find xpath of any element.

**45 : Give me WebDriver's API name using which we can perform drag and drop operation.**

That API's name Is Advanced User Interactions API using which we can perform drag and drop operation on page of software web application. Same API can help us to perform some other operations too like moveToElement, doubleClick, clickAndHold, moveToElement, etc..

**46 : Can we perform drag and drop operation In Selenium WebDriver? Tell me a syntax to drag X element and drop It On Y element.**

Yes, We can perform drag and drop operation using selenium webdriver software testing tool's Advanced User Interactions API. Syntax Is like below.

```
new Actions(driver).dragAndDrop(X, Y).build().perform();
```

**47 : Do you have faced any technical challenges with Selenium WebDriver software test automation?**

Yes, I have faced below given technical challenges during selenium webdriver test cases development and running for software web application.

Sometimes (Not always), Some elements like text box, buttons etc. are taking more time(more than given Implicit wait time) to appear on page of software web application or to get enabled on page. In such situation, If I have used only Implicit wait then my test case can run fine on first run but It may fail to find element on second run. So we need provide special treatment for such elements so that webdriver script wait for element to be present or get enabled on page of software web application during test execution. We can use Explicit wait to handle this situation.

Handling dynamic changing ID to locate element Is tricky. If element's ID Is changing every time when you reload the software web application page and you have to use that ID In

XPath to locate element then you have to use functions like starts-with(@id,'post-body-') or contains(@id,'post-body-') In XPath.

Clicking on sub menus which are getting rendered on mouse hover of main menu is somewhat tricky. You need to use webdriver's Actions class to perform mouse hover operation.

If you have to execute your test cases in multiple browsers then one test case can run successfully in Firefox browser but same test case may fail in IE browser due to the timing related issues (nosuchelement exception) because test execution in Firefox browser is faster than IE browser. You can resolve this issue by increasing implicit wait time when you run your test in IE browser.

Above issue can arise due to the unsupported XPath in IE browser. In this case, You need to use OTHER ELEMENT LOCATING METHODS (ID, Name, CSSSelector etc.) to locate element.

Handling JQuery elements like moving pricing slider, date picker, drag and drop etc.. is tricky. You should have knowledge of webdriver's Advanced User Interactions API to perform all these actions.

Working with multiple Windows, Frames, and some tasks like Extracting data from web table, Extracting data from dynamic web table, Extracting all Links from page, Extracting all text box from page are also tricky and time consuming during test case preparation.

There is not any direct command to upload or download files from web page using selenium webdriver. For downloading files using selenium webdriver, You need to create and set Firefox browser profile with webdriver test case.

Webdriver do not have any built in object repository facility. You can do it using java .properties file to create object repository as described in.

Webdriver do not have any built in framework or facility using which we can achieve below given tasks directly : 1. Capturing screenshots, 2. generating test execution log, 3. reading data from files, 4. Generating test result reports, Manage test case execution sequence. To achieve all these tasks, We have to use external services with webdriver like Log4J to generate log, Apache POI API to read data from excel files, TestNG XSLT reports to generate test result reports. TestNG to manage test case execution, .properties file to create object repository. All these tasks are very time consuming.

#### **48 : Can you tell me a syntax to close current webdriver instance and to close all opened webdriver instances?**

Yes, To close current WebDriver instance, We can use Close() method as below.

```
driver.close();
```

If there are opened multiple webdriver Instances and wants to close all of them then we can use webdriver's quit() method as bellow in software automation test.

```
driver.quit();
```

**49 : Is It possible to execute javascript directly during software test execution? If Yes then tell me how to generate alert by executing javascript In webdriver script?**

Yes, we can execute javascript during webdriver software test execution. To generate alert, You can write bellow given code In your script.

```
JavascriptExecutor javascript = (JavascriptExecutor) driver;  
javascript.executeScript("alert('Javascript Executed.');");
```

**50 : Give me a syntax to read javascript alert message string, clicking on OK button and clicking on Cancel button.**

We can read alert message string as bellow.

```
String alrtmsg = driver.switchTo().alert().getText();
```

We can click on OK button of alert as bellow.

```
driver.switchTo().alert().accept();
```

We can click on Cancel button of alert as bellow.

```
driver.switchTo().alert().dismiss();
```

**51 : Tell me a scenario which we can not automate In selenium WebDriver.**

1. Bitmap comparison Is not possible using selenium webdriver software testing tool.
2. Automating captcha Is not possible. (Few peoples says we can automate captcha but I am telling you If you can automate any captcha then It Is not a captcha).
3. We can not read bar code using selenium webdriver software testing tool.

**52 : What Is JUnit?**

Java software developers use JUnit as unit testing framework to write repeatable tests for java programming language. It Is very simple and open source framework and we can use It In selenium webdriver test scripts creation to manage them In well manner.

**53 : Which Is the latest version of JUnit.**

Current latest version of JUnit Is 4.12-beta-2. This can change In future. To check latest released version of JUnit, You can Visit JUnit Official WebSite.

**54 : Tell me different JUnit annotations and Its usage.**

JUnit has bellow given different annotations.

**@Test** : @Test annotation Is useful to Identify method as a Test method from software automation test script.

**@Before** : @Before annotation method will be executed before each and every @Test method.

**@After** : @After annotation method will be executed after each and every @Test method.

**@BeforeClass** : @BeforeClass annotation method will be executed before all @Test methods In a class(Means before first @Test method).

**@AfterClass** : @AfterClass annotation method will be executed after all @Test method In a class(Means after last @Test method).

**@Ignore** : @Ignore annotation Is useful to exclude @Test method from execution.

**@Test(timeout=1000)** : You can set @Test method execution timeout. This @Test method fails Immediately when Its execution time cross 1000 milliseconds.

**55 : Write sample JUnit @Test method that passes when expected ArithmeticException thrown.**

Sample JUnit @Test to pass on expected ArithmeticException Is as bellow.

```
@Test(expected = ArithmeticException.class)

public void excOnDivision() {

    int i = 5/0;

}
```

**56 : Write sample JUnit @Test method that fails when unexpected ArithmeticException thrown.**

Sample JUnit @Test to fail on unexpected ArithmeticException Is as bellow.

```
@Test

public void excOnDivision() {

    int i = 5/0; }
```

**57 : What are the advantages of TestNG over JUnit.**

Advantages of TestNG over JUnit JUnit are as bellow.

TestNG Annotations are simple and Easy to understand.

Easy to parameterize the software test cases In TestNG.

Easy to run software automation test cases In parallel.

Can generate Interactive XSLT test execution reports using TestNG.

**58 : Tell me main features of JUnit.**

JUnit features are as bellow.

JUnit Is unit software testing framework. So It helps software developers to create and run unit test cases very easily.

There are many different annotations available In JUnit. Using all those annotations, we can Identify and configure webdriver software test case very easily.

JUnit supports many different assertions using which we can compare webdriver software automation test's expected and actual result.

We can create test suite for multiple test cases to run all of them In one go using JUnit.

We can generate webdriver test execution HTML reports using JUnit. .

**59 : What are different assertions supported by JUnit?**

List of JUnit assertions as bellow.

assertEquals

assertFalse

assertTrue

assertNull

assertNotNull

assertSame

assertNotSame

assertArrayEquals

**60 : How to create and run JUnit test suite for selenium WebDriver?**

For creating JUnit software test suite, we have to create test cases class files and one separate test suite file. Then we can write syntax like bellow In test suite file to run test suite.

```
@RunWith(Suite.class)  
  
@SuiteClasses({ junittest1.class, junittest2.class })  
  
public class junittestsuite {  
  
}
```

In above example, junittest1.class and junittest2.class are test case class names.

**61 : For what purpose, assertTrue and assertFalse assertions are used?**

In selenium webdriver software test automation, We need to assert Boolean conditions true and false. We can assert both these conditions using assertTrue and assertFalse JUnit assertions.

**62 : Can you give me example of JUnit assertEquals assertion?**

Example of JUnit assertEquals assertion Is as bellow. It assert that values of actTotal and expTotal are equal or not.

```
public void sumExample() {  
  
    int val1 = 10;  
  
    int val2 = 20;  
  
    int expTotal = 35;  
  
    int actTotal = 0;  
  
    actTotal = val1 + val2;  
  
    assertEquals(actTotal, expTotal);  
  
}
```

**63 : What Is TestNG?**

TestNG Is Open Source(Freeware) framework which Is Inspired from NUnit and JUnit with Introducing few new features and functionality compared to NUnit and JUnit to make It easy to use and more powerful.

We can use TestNg with selenium webdriver software testing tool to configure and run test cases very easily, easy to understand, read and manage test cases, and to generate HTML or XSLT test reports.

**64 : Can you describe major features of TestNG?**

TestNG has many major features like support of @DataProvider annotation to perform data driven testing on software web application, can set test case execution dependency, test case grouping, generate HTML and XSLT test execution report for software web application etc

**65 : Describe the similarities and difference between JUnit and TestNG unit testing frameworks.**

You can find all the similarities and difference between JUnit and TestNG framework on THIS PAGE.

**66 : How to Install TestNG In Eclipse? How do you verify that TestNg Is Installed properly In Eclipse?**

To Install TestNG software unit testing framework In Eclipse,

**67 : What are different annotations supported by TestNG ?**

TestNG supports many different annotations to configure Selenium WebDriver software automation test.

**68 : What Is the usage of testng.xml file?**

In selenium WebDriver software testing tool, We are using testng.xml file to configure our whole test suite In single file. Few of the tasks which we can specify In testng.xml file are as bellow.

We can define software testing test suite using set of test cases to run them from single place.

Can Include or exclude test methods from software web application's test execution.

Can specify a group to include or exclude.

Can pass parameter to use In test case of software web application.

Can specify group dependencies.

Can configure parallel test execution for software web application.

Can define listeners.

## **69 : How to pass parameter with testng.xml file to use It In test case?**

We can define parameter In testng.xml file using syntax like bellow.

```
<parameter name="browser" value="FFX" />
```

Here, name attribute defines parameter name and value defines value of that parameter.  
Then we can use that parameter In selenium webdriver software automation test case using bellow given syntax.

```
@Parameters {"browser"})
```

## **70 : I have a test case with two @Test methods. I wants to exclude one @Test method from execution. Can I do It? How?**

Yes you need to specify @Test method exclusion In testng.xml file as bellow.

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >

<suite name="Test Exclusion Suite">

<test name="Exclusion Test" >

<classes>

<class name="Your Test Class Name">

<methods>

<exclude name="Your Test Method Name To Exclude"/>

</methods>

</class>

</classes>

</test>

</suite>
```

You need to provide @Test method name In exclude tag to exclude It from execution.

## **71 : Tell me syntax to skip @Test method from execution.**

You can use bellow given syntax Inside @Test method to skip It from test execution.

```
throw new SkipException("Test Check_Checkbox Is Skipped");
```

It will throw skip exception and @Test method will be sipped immediately from execution.

### **73 : How to set priority of @Test method? What Is Its usage?**

In your software web application's test case, you can set priority for TestNG @Test annotated methods as bellow.

```
@Test(priority=0)
```

Using priority, We can control @Test method execution manner as per our requirement. That means @Test method with priority = 0 will be executed 1st and @Test method with priority = 1 will be executed 2nd and so on.

### **74 : Tell me any 5 assertions of TestNG which we can use In selenium webdriver software testing tool.**

There are many different assertions available In TestNG but generally I am using bellow given assertions In my test cases.

assertEquals

assertNotEquals

assertTrue

assertFalse

assertNull

assertNotNull

### **75 : Can you tell me usage of TestNG Soft Assertion In selenium webdriver software testing tool?**

Using TestNG soft assertion, We can continue our test execution even if assertion fails. That means on failure of soft assertion, remaining part of @Test method will be executed and assertion failure will be reported at the end of @Test method.

### **76 : How to write regular expression In testng.xml file to search @Test methods containing "product" keyword.**

Regular expression to find @Test methods containing keyword "product" Is as bellow In selenium webdriver software testing tool.

```
<methods>
```

```
    <include name=". *product.*"/>
```

```
</methods>
```

**77 : Which time unit we provide In time test? minutes? seconds? milliseconds? or hours?  
Give Example.**

Time unit we provide on @Test method level or test suite level Is In milliseconds.

**78 : What Is the syntax to get value from text box and store It In variable.**

Most of the time, String In text box will be stored as value. So we need to access value attribute(getAttribute) of that text box as shown In bellow example.

String Result =

```
driver.findElement(By.xpath("//input[@id='Resultbox']")).getAttribute("value");
```

**79 : What Is the difference between findelement and findElements ?**

findElement Is useful to locate and return single element from page of software web application while findElements Is useful to locate and return multiple elements from software web page.

**80 : Tell me looks like XPath of sibling Input element which Is after Div in the DOM.**

XPath for above scenario will be something like bellow.

```
//div/following-sibling::input
```

**81 : Tell me looks like CSSSelector path of sibling Input element which Is after Div in the DOM.**

CSSSelcor path will looks like bellow.

```
css=div + input
```

**82 : What Is Parallelism In TestNG?**

In general software term, Parallelism means executing two part of software program simultaneously or executing software program simultaneously or we can say multithreaded or parallel mode. TestNG has same feature using which we can start multiple threads simultaneously In parallel mode and test methods will be executed In them.

**83 : What are the benefits of parallelism over normal execution?**

Using parallelism facility of TestNG In selenium webdriver,

Your software test execution time will be reduced as multiple tests will be executed simultaneously.

Using parallelism, We can verify multithreaded code In software application.

**84 : I wants to run test cases/classes In parallel. Using which attribute and value I can do It?**

You have to use parallel = classes attribute In testng.xml to run software web app tests parallel. .

**85 : What Is dependency test In TestNG?**

Dependency Is very good feature of testng using which we can set software test method as dependent test method of any other single or multiple or group of test methods. That means depends-on method will be executed first and then dependent test method will be executed. If depends-on software test method will fail then execution of dependent test method will be skipped automatically. TestNG dependency feature will works only If depends-on test method Is part of same class or part of Inherited base class.

**86 : What Is the syntax to set test method dependency on multiple test methods.**

We can set test method's dependency on multiple test methods as bellow.

```
@Test(dependsOnMethods={"Login","checkMail"})  
  
public void LogOut() {  
  
    System.out.println("LogOut Test code.");  
  
}
```

Above test method Is depends on Login and checkMail test methods. .

**87 : What Is the syntax to set test method disabled.**

We can use attribute enabled = false with @Test annotation to set test method disabled. Syntax Is as bellow.

```
@Test(enabled = false)  
  
public void LogOut() {  
  
    System.out.println("LogOut Test code.");  
  
}
```

**88 : In XPath, I wants to do partial match on attribute value from beginning. Tell me two functions using which I can do It.**

We can use bellow given two functions with XPath to find element for software web page using attribute value from beginning.

contains()

starts-with()

**89 : I have used findElements In my software test case. It Is returning NoSuchElementException when not element found. Correct me If I am wrong.**

It Is Incorrect. findElements will never return NoSuchElementException. It will return just an empty list.

**90 : My Firefox browser Is not Installed at usual place. How can I tell FirefoxDriver to use It?**

If Firefox browsers Is Installed at some different place than the usual place then you needs to provide the actual path of Firefox.exe file as bellow.

```
System.setProperty("webdriver.firefox.bin","C:\\Program Files\\Mozilla
Firefox\\Firefox.exe");
driver =new FirefoxDriver();
```

**91 : How to create custom firefox profile and how to use It In selenium webdriver software test?**

**92 : What versions of Internet Explorer are supported by selenium WebDriver software testing tool?**

Till date, Selenium WebDriver software testing tool supports IE 6, 7, 8, 9, 10 and 11 with appropriate combinations of Windows 7, Vista or XP.

**93 : Tell me the class name using which we can generate Action chain.**

The WebDriver class name Using which we can generate Action chain Is "Actions". VIEW USAGE OF ACTIONS CLASS with practical example on how to generate series of actions to drag and drop element of software web application.

**94 : Do you know method name using which we can builds up the actions chain?**

Method name of Actions class to build up actions chain Is "build()". THIS EXAMPLE will show you how to build drag and drop element on software web page by x,y pixel actions using build() method.

**95 : When we can use Actions class In Selenium WebDriver test case?**

Few of the examples are bellow where can use actions class to perform operations In software web application. Drag and drop element -

Drag and drop by x,y pixel offset -

Selecting JQuery selectable Items -

Moving JQuery slider -

Re-sizing JQuery re-sizable element -

Selecting date from JQuery date picker -

**96 : Can we capture screenshot In Selenium WebDriver software testing tool? How?**

We can use selenium webdriver TakesScreenshot method to capture screenshot. Java File class will be used to store screenshot In your system's local drive.

**97 : Selenium WebDriver has any built In method using which we can read data from excel file?**

No, Selenium webdriver software testing tool do not have any built In functionality using which we can read data from excel file.

**98 : Do you know any external API name using which we can read data from excel file?**

We can use jxl API (Java Excel API) to read data from excel file.

We can use one more powerful API known as Apache POI API to read and write data In excel file. I have created data driven framework using Apache POI API.

**99 : Tell me any 5 webdriver common exceptions which you faced during software test case execution.**

WebDriver's different 5 exceptions are as bellow.

TimeoutException - This exception will be thrown when command execution does not complete In given time.

NoSuchElementException - WebDriver software testing tool will throw this exception when element could not be found on page of software web application.

NoAlertPresentException - This exception will be generated when webdriver tries to switch to alert popup but there is not any alert present on page.

ElementNotSelectableException - It will be thrown when webdriver is trying to select unselectable element.

ElementNotVisibleException - Thrown when webdriver is not able to interact with element which is available In DOM but it is hidden.

StaleElementReferenceException -

**100 : Tell me different ways to type text In text box In selenium software test.**

We can type text In text box of software web application page using bellow given ways In selenium test.

1.Using .SendKeys() method

```
driver.findElement(By.xpath("//input[@id='fname']")).sendKeys("Using sendKeys");
```

2. Using JavascriptExecutor

```
((JavascriptExecutor)driver).executeScript("document.getElementById('fname').value='Using JavascriptExecutor'");
```

3. Using Java Robot class

```
driver.findElement(By.xpath("//input[@id='fname']")).click();
```

```
Robot robot = new Robot();
```

```
robot.keyPress(KeyEvent.VK_U);
```

```
robot.keyPress(KeyEvent.VK_S);
```

```
robot.keyPress(KeyEvent.VK_I);
```

```
robot.keyPress(KeyEvent.VK_N);
```

```
robot.keyPress(KeyEvent.VK_G);
```

```
robot.keyPress(KeyEvent.VK_SPACE);
```

```
robot.keyPress(KeyEvent.VK_R);
```

```
robot.keyPress(KeyEvent.VK_O);
```

```
robot.keyPress(KeyEvent.VK_B);
```

```
robot.keyPress(KeyEvent.VK_O);
```

```
robot.keyPress(KeyEvent.VK_T);
```

**101 : Tell me different ways to verify element present or not on page.**

We can check If element Is present or not on page of software we application using bellow given 2 simple ways.

1. Using .size() method

```
Boolean elePresent = driver.findElements(By.id("ID of element")).size() != 0;
```

If above syntax return "false" means element is not present on page and "true" means element is present on page.

## 2. Using .isEmpty() method

```
Boolean elePresent = driver.findElements(By.id("ID of element")).isEmpty();
```

If this returns "true" means element is not present on page and "false" means element is present on page.

## **102 : Why we need to customize Firefox browser profile In webdriver test?**

Webdriver launch fresh browser Instance when we run software tests In Firefox browser using selenium webdriver. Fresh browser Instance do not have any Installed add-ons, saved passwords, bookmarks and any other user preferences. So we need to create custom profile of Firefox browser to get any of this thing In Webdriver launched browser.

## **103 : How to customize Firefox browser profile for webdriver software test?**

You can do it in two different ways.

You can create your desired firefox browser profile before running software automation test and then you can use it in your selenium webdriver software test. .

You can customize your firefox browser profile run time before launching webdriver's Firefox browser Instance. .

## **104 : What is Difference betweengetAttribute() and getText()?**

getAttribute() method is useful to read software web app element's attribute value like id, name, type etc. .

getText() method is useful to read text from element or alert..

## **105 : What is the difference between WebDriver and Remote WebDriver?**

Simple answer for this tions is as bellow.

WebDriver : Webdriver is an Interface or we can say software testing tool using which we can create automated test cases for web application and then run on different browsers like IE, Google chrome, Firefox etc.. We can create test cases in different languages..

Remote WebDriver : Remote WebDriver is useful to run test cases in same machine or remote machines using selenium Grid.

**106 : I have total 200 test cases. I wants to execute only 20 test cases out of them. Can I do It In selenium WebDriver? How?**

Yes. If you are using TestNG with selenium webdriver software testing tool then you can do Is using grouping approach as described In THIS PAGE. Create separate group for those 20 test cases and configure testng.xml file accordingly to run only those 20 test cases.

Also If you are using data driven framework then you can configure It In excel file. You can configure such data driven framework at your own by following steps given on THIS PAGE.

**107 : Can you tell me three different ways to refresh page. Do not use .refresh() method.**

We can refresh browser In many different ways. Three of them are as bellow.

```
driver.get(driver.getCurrentUrl());  
driver.navigate().to(driver.getCurrentUrl());  
driver.findElement(By.xpath("//h1[@class='title']")).sendKeys(Keys.F5);
```

**108 : I wants to pass parameter In software test case through testng.xml file. How can I do It?**

You can use <parameter> node under <test> node In testng.xml file with parameter name and value. Then you can use @Parameters annotation with parameter name In your test case of software web application.

**109 : My page contains file upload field but I am not able to upload file using selenium webdriver software testing tool. Is there any other way using which I can upload file In selenium test?**

If you are not able to upload file using selenium webdriver then you can create file upload script In AutoIT and then you can use It In selenium webdriver software test. You can refer bellow given articles to learn more about It.

What Is AutoIT V3

Steps To Download And Install AutoIT V3

Creating Autolt Script To Upload File On Web Page

Upload File In Selenium WebDriver Using Autolt

**110 : I wants to set size and position of my browser window. Do you know how to do it in selenium webdriver?**

We can use setSize function to set size of window and setPosition function to set position of browser window..

**111 : Is there any way to get size and position of browser window in selenium webdriver?**

Yes.. We can get it using webdriver functions getSize and getPosition.

**112 : I wants to scroll my software web application page by 300 pixel. Tell me how can i do it?**

We can use javascript executor with window.scrollBy(x,y) to scroll page in x or y directions.

**1) What is Automation Testing?**

Automation testing or Test Automation is a process of automating the manual process to test the application/system under test. Automation testing involves use to a separate testing tool which lets you create test scripts which can be executed repeatedly and doesn't require any manual intervention.

**2) What are the benefits of Automation Testing?**

Benefits of Automation testing are:

1. Supports execution of repeated test cases
2. Aids in testing a large test matrix
3. Enables parallel execution
4. Encourages unattended execution
5. Improves accuracy thereby reducing human generated errors
6. Saves time and money

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

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

**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)** – 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** – 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** – Selenium Grid is used to distribute your test execution on multiple platforms and environments concurrently.

### **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

### **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.

### **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 support 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	Selenium RC requires server to be started before executing the	WebDriver doesn't require any server to be started before

Feature	Selenium IDE	Selenium RC	WebDriver
	executing the test scripts	test scripts	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	Selenium RC is slower than WebDriver as it doesn't communicates directly with the	WebDriver communicates directly with the web browsers. Thus making

Feature	Selenium IDE	Selenium RC	WebDriver
	launches the test. Thus, the IDE and browser communicates directly	browser; rather it sends selenese commands over to Selenium Core which in turn communicates with the browser.	it much faster.

### 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.

### 9) What is Selenese?

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

### 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:

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

### 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 halt i.e. any failure during verification would not stop the execution and all the test steps would be executed.

### 12) What is an Xpath?

Xpath 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.

**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.

**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://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.

**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.

**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.

**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

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

There are two types of waits available in WebDriver:

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.

## 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");
```

## 22) How can you find if an element is 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")).isSelected();
```

#### `isEnabled():`

```
boolean searchIconEnabled = driver.findElement(By.id("gbqfb")).isEnabled();
```

## 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();
```

## 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);
```

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

Following are the navigation commands:

**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");
```

**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.

**27) How to handle frame in WebDriver?**

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();
```

### 28) When do we use findElement() and findElements()?

**findElement()**: findElement() is used to find the first element in the current web page matching to the specified locator value. Take a note that only first matching element would be fetched.

**Syntax:**

```
WebElement element = driver.findElements(By.xpath("//div[@id='example']//ul//li"));
findElements(): findElements() is used to find all the elements in the current web page matching to the specified locator value. Take a note that all the matching elements would be fetched and stored in the list of WebElements.
```

**Syntax:**

```
List <WebElement> elementList
=driver.findElements(By.xpath("//div[@id='example']//ul//li"));
```

### 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

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

// Fetching the size of the list
int listSize = elementList.size();
for (int i=0; i<listSize; i++)
{
    // Clicking on each service provider link
    serviceProviderLinks.get(i).click();
}

// Navigating back to the previous page that stores link to service providers
driver.navigate().back();
}
```

### 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 it 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 it return any value.

### **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.

### **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. 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();
```

### **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.

### **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"));
```

### **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:**

```
// Instantiating Action Interface  
Actions actions=new Actions(driver);  
  
// howering on the dropdown  
actions.moveToElement(driver.findElement(By.id("id of the dropdown"))).perform();  
  
// Clicking on one of the items in the list options
```

```
WebElement subLinkOption=driver.findElement(By.id("id of the sub link"));
subLinkOption.click();
```

### 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");
```

### 37) How to capture screenshot in WebDriver?

```
public class CaptureScreenshot {
    WebDriver driver;
    @BeforeTest
    public void setUp() throws Exception {
        driver = new FirefoxDriver();
        driver.get("https://google.com");
    }
    @AfterTest
    public void tearDown() throws Exception {
        driver.quit();
    }

    @Test
    public void test() throws IOException {
        // Code to capture the screenshot
        File scrFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
        // Code to copy the screenshot in the desired location
        FileUtils.copyFile(scrFile, newFile("C:\\CaptureScreenshot\\google.jpg"));
    }
}
```

### 38) What is Junit?

Junit is a unit testing framework introduced by Apache. Junit is based on Java.

### 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.

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

TestNG 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

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

##### **Setting Priority in TestNG**

###### **Code Snippet**

```
package testng;
import org.testng.annotations.*;
public class settingpriority {
    @test(priority=0)
    public void method1() {
    }
    @test(priority=1)
    public void method2() {
    }
    @test(priority=2)
    public void method3() {
    }
}
```

###### **Test Execution Sequence:**

1. Method1
2. Method2
3. Method3

#### **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.

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

#### **Advantage of Test Automation framework**

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

### **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.

### **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.

### **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

# JXL jar	POI jar
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

#### 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

#### 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.

#### 49) Can captcha be automated?

No, captcha and bar code reader cannot be automated.

#### 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.

### **1. What is selenium webdriver?**

WebDriver is designed to provide a simpler, more concise programming interface in addition to addressing some limitations in the Selenium-RC API. Selenium-WebDriver was developed to better support dynamic web pages where elements of a page may change without the page itself being reloaded. WebDriver's goal is to supply a well-designed object-oriented API that provides improved support for modern advanced web-app testing problems. Selenium-WebDriver makes direct calls to the browser using each browser's native support for automation.

### **2. How to write the tests in selenium WebDriver?**

In selenium WebDriver, depending on the programming language, I used different test framework. In Java, I used TestNG, and in Java, I used JUnit. In either programming language, I defined browser webdriver in setup method, and wrote test steps in test method and dispose and close the webdriver in the tear down method.

### **3. How to configure selenium webdriver in eclipse?**

In eclipse, I created java projects and added JUnit or TestNG classes. In the project reference, I added JUnit or TestNG jar file. In the test class, I used webdriver in setup, test and teardown methods. Sometimes, I used webdriver in beforeclass, beforemethod, aftermethod, afterclass sections.

### **4. What are the prerequisites to run selenium webdriver?**

Depending on the programming language, reference files should be added to the test solutions in C# or test projects in Java. For example, in C#, I added webdriver dlls and in Java, I added Selenium-client-driver.jar file. And also, we should have programming IDE like visual studio or eclipse to run webdriver.

### **5. What is the difference between selenium 1.0 and webdriver?**

Selenium 1.0 needs Selenium RC to run a test. However, webdriver can directly launch a browser and run tests.

### **6. What are the advantages of selenium webdriver?**

Selenium WebDriver is very flexible to use with Java, .Net, Python, Ruby or html languages. QA engineers who have good coding skills can use it very effectively.

### **7. What are the disadvantages of selenium webdriver over selenium 1.0?**

Since selenium web driver requires coding skills, QA engineers should have some knowledge of program development in Java, .Net, or other languages.

## **8. How to handle multiple windows in selenium webdriver?**

We can use web driver's windows handler to identify each window and use switch method to pick the window for test.

## **9. How to navigate with browser buttons in selenium webdriver?**

We can use web driver's back or forward method to simulate browser's navigation button functionality.

## **10. Which are the locators used for recognizing the objects in selenium webdriver?**

In webdriver, we can use element id, name, css, xpath, link text, partial link text and DOM to locate elements.

## **11. How to run the tests in internet explorer using selenium webdriver?**

When setup a webdriver in the code, we can select InternetExplorerDriver to use ID. If we want to use of the latest and greatest features of the WebDriver "InternetExplorerDriver", we need to download Internet Explorer Server.

## **12. How to run the tests in firefox using selenium webdriver?**

In the setup method, we select FirefoxDriver for the webDriver.

## **13. How to run the tests in google chrome using selenium webdriver?**

Sometimes, webdriver cannot launch chrome directly, so (1) we can use Desired Capabilities of WebDriver, put chrome browser application path in the code ; (2) we need to have chromedriver.exe file in the application path. Alternatively, we can manually start chrome driver service, and then launch the test in chrome.

## **14. How to run the tests without a browser or with HTML unit driver in selenium webdriver?**

```
WebDriver driver = new HtmlUnitDriver();
```

## **15. How to run Selenium 1.0 tests in webdriver?**

We can use WebDriverBackedSelenium to run Selenium 1.0 tests in webdriver.

```
WebDriver driver = new FirefoxDriver();
```

```
Selenium selenium = new WebDriverBackedSelenium(driver, "http://www.yoursite.com");
```

## **16. How to convert selenium 1.0 tests to webdriver tests?**

We can use WebDriverBackedSelenium to run Selenium 1.0 tests in webdriver.

```
WebDriver driver = new FirefoxDriver();  
Selenium selenium = new WebDriverBackedSelenium(driver, "http://www.yoursite.com");
```

## **17. What is webdriver backed selenium?**

WebDriver backed Selenium is API that enables running Selenium 1.0 tests in web driver.

## **18. When to use web driver backed selenium?**

When we have existing tests in Selenium 1.0 (RC), if we want to avoid using Selenium RC, instead we want to use web driver, we need to use web driver backed selenium.

## **19. Which version of selenium IDE supports webdriver?**

Any version higher than 2.0 supports webdriver.

## **20. How to invoke an application in webdriver?**

We can use Process to invoke application in the code using web driver.

## **21. Which of Selenium IDE commands not supported in webdriver?**

It depends on the format of conversion functionality of Selenium IDE to web driver. sometimes, not all IDE script can be converted to web driver without any problem.

## **22. Where to download selenium webdriver?**

Selenium WebDriver libraries can be download from <http://www.seleniumhq.org> website.

## **23. What is implicit and explicit wait in selenium webdriver?**

### **Explicit wait:**

An explicit waits is code you define to wait for a certain condition to occur before proceeding further in the code. The worst case of this is Thread.sleep(), which sets the condition to an exact time period to wait. There are some convenience methods provided that help you write code that will wait only as long as required. WebDriverWait in combination with ExpectedCondition is one way this can be accomplished.

### **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 set, the implicit wait is set for the life of the WebDriver object instance.

## **25. What Selenese commands can be used to help debug a regexp?**

Text Pattern matching commands can be used to help debug a regex. For example, Verify Title regexp:\*.Film will match any text with \*.Film pattern.

**26. What is one big difference between SilkTest and Selenium, excluding the price?**

Silk Test is a test automation solution for development, quality and business teams who need to deliver software faster.

With Silk Test you can create and execute tests across multiple platforms and devices to ensure that your applications work exactly as intended.

In Selenium, we create tests with Selenium IDE and run it with multiple platforms by using web driver.

**27. Which browsers can Selenium IDE be run in?**

Selenium IDE is firefox add on, so it runs on Firefox browser.

**28. If a Selenium function requires a script argument, what would that argument look like in general terms?**

That should be java script argument.

**29. If a Selenium function requires a pattern argument, what five prefixes might that argument have?**

\* which translates to “match anything,” i.e., nothing, a single character, or many characters.

[ ] (character class) which translates to “match any single character found inside the square brackets.” A dash (hyphen) can be used as a shorthand to specify a range of characters (which are contiguous in the ASCII character set). A few examples will make the functionality of a character class clear:

[aeiou] matches any lowercase vowel

[0-9] matches any digit

[a-zA-Z0-9] matches any alphanumeric character

**30. What is the regular expression sequence that loosely translates to "anything or nothing?"**

\* which translates to “match anything,” i.e., nothing, a single character, or many characters.

**31. What is the globbing sequence that loosely translates to "anything or nothing?"**

glob: label

**32. What does a character class for all alphabetic characters and digits look like in regular expressions?**

[a-z A-Z 0-9] matches any alphanumeric character

**33. What does a character class for all alphabetic characters and digits look like in globbing?**

regexp: \*[a-z A-Z 0-9]

**34. What must one set within SIDE in order to run a test from the beginning to a certain point within the test?**

We need to set a break point in the test, so the test will pause at the break point.

**35. What does a right-pointing green triangle at the beginning of a command in SIDE indicate?**

That triangle indicates that test paused at that step, or will start from that step.

**36. How does one get rid of the right-pointing green triangle?**

Right click on that line and remove break point.

**37. How can one add vertical white space between sections of a single test?**

User can input comment line between test steps, and in the comment line, add spaces.

**38. What Selenium functionality uses wildcards?**

Regular expression can use wildcards.

**39. Which wildcards does SIDE support?**

\*, ?, :

**40. What are the main types of regular expression quantifiers you know?**

\* which translates to “match anything,” i.e., nothing, a single character, or many characters.

[ ] (character class) which translates to “match any single character found inside the square brackets.” A dash (hyphen) can be used as a shorthand to specify a range of characters (which are contiguous in the ASCII character set). A few examples will make the functionality of a character class clear:

[aeiou] matches any lowercase vowel

[0-9] matches any digit

[a-z A-Z 0-9] matches any alphanumeric character

**41. What regular expression special character(s) means "any character?"**

\* symbol means any character

**42. What distinguishes between an absolute and relative URL in SIDE?**

Absolute URL means the URL starts with http or https. Selenium IDE adds the argument in the open statement and creates an absolute URL.

Relative URL means that in baseURL, we can put the main domain, and in the open statement, user should put relative URL;

For example, if main url is http://mysite.com, in the relative url, we can put http://beta.mysite.com in the open statement.

**43. How would one access a Selenium variable named "count" from within a JavaScript snippet?**

`${count}`

**44. What Selenese command can be used to display the value of a variable in the log file, which can be very valuable for debugging?**

We can use echo command to log information in the test result output.

**45. If one wanted to display the value of a variable named answer in the log file, what would the first argument to the previous command look like?**

`echo ${variableName}`

**46. Where did the name "Selenium" come from?**

Actually, Selenium is a chemical element name. But Thoughtworks used it for test automation framework api.

**47. Which Selenium command(s) simulates selecting a link?**

Selenium has several commands to verify links. For simulating selecting a link, we can use `findElement(By.Link("LinkName"))`

**48. Which two commands can be used to check that an alert with a particular message popped up?**

Use command VerifyAlertPresent

**49. What does a comment look like in Column view?**

Comments display in pink color

**50. What does a comment look like in Source view?**

<!--This is second comment-->  
<!--This is a comment-->

**51. What are Selenium tests normally named (as displayed at the top of each test when viewed from within a browser)?**

In the browser, the test name in Selenium IDE will be displayed.

**52. What command simulates selecting the browser's Back button?**

goBack command

**53. If the Test Case frame contains several test cases, how can one execute just the selected one of those test cases?**

Users can use "Play current test case" function of Selenium IDE

**54. What globbing functionality is NOT supported by SIDE?**

Globbing supports \* or ?, so other patterns are not supported by globbing. We need to use Regular Expressions for that.

**55. What is wrong with this character class range? [A-z]**

It should be [a-z]; A should be written as lower case.

**56. What are three ways of specifying an uppercase or lowercase M in a Selenese pattern?**

We can use globbing, regular expression, and exact.

Uppercase: [A-Z]

Lowercase: [a-z]

**57. What does this regular expression match?**

**regexp:[1-9][0-9],[0-9]{3},[0-9]{4}**

If we use the number 1

match: 11,111,1111

**58. What are two ways to match an asterisk within a Selenese regexp?**

\* or any symbol with \*

**59. What is the generic name for an argument (to a Selenese command) which starts with //?**

Xpath

**60. What Selenese command is used to choose an item from a list?**

Select command

**61. How many matches exist for this pattern?**

**regexp:[13579][02468]**

25 (basically, any odd number from this group [13579] and any even number from this [02468])

**62. What is the command associated with testing an alert?**

AssertAlert, AssertAlertNoPresent

**63. How can one get SIDE to always record an absolute URL for the open command's argument?**

In the Selenium IDE options, check "Record Absolute URL" option.

**64. What Selenese command and argument can be used to transfer the value of a JavaScript variable into a SIDE variable?**

Use "store" command

**65. How would one access the value of a SIDE variable named name from within a JavaScript snippet used as the argument to a Selenese command?**

`${variableName}`

**What are the set of tools available with Selenium?**

Selenium has four set of tools – Selenium IDE, Selenium 1.0 (Selenium RC), Selenium 2.0 (WebDriver) and Selenium Grid. Selenium Core is another tool but since it is available as part of Selenium IDE as well as Selenium 1.0, it is not used in isolation.

**Which Selenium Tool should I use?**

It entirely boils down to where you stand today in terms of using Selenium. If you are entirely new to Selenium then you should begin with Selenium IDE to learn Selenium location strategies and then move to Selenium 2 as it is the most stable Selenium library and

future of Selenium. Use Selenium Grid when you want to distribute your test across multiple devices. If you are already using Selenium 1.0 than you should begin to migrate your test scripts to Selenium 2.0

### **What is Selenium IDE?**

Selenium IDE is a firefox plug-in which is (by and large) used to record and replay test in firefox browser. Selenium IDE can be used only with firefox browser.

### **Which language is used in Selenium IDE?**

Selenium IDE uses html sort of language called Selenese. Though other languages (java, c#, php etc) cannot be used with Selenium IDE, Selenium IDE lets you convert test in these languages so that they could be used with Selenium 1.0 or Selenium 2.0

### **What is Selenium 1.0?**

Selenium 1.0 or Selenium Remote Control (popularly known as Selenium RC) is library available in wide variety of languages. The primary reason of advent of Selenium RC was incapability of Selenium IDE to execute tests in browser other than Selenium IDE and the programmatical limitations of language Selenese used in Selenium IDE.

### **What is Selenium 2.0?**

Selenium 2.0 also known as WebDriver is the latest offering of Selenium. It provides

- ② better API than Selenium 1.0
- ② does not suffer from java script security restriction which Selenium 1.0 does
- ② supports more UI complicated UI operations like drag and drop

### **What is Selenium Grid?**

Selenium grid lets you distribute your tests on multiple machines and all of them at the same time. Hence you can execute test on IE on Windows and Safari on Mac machine using the same test script (well, almost always). This greatly helps in reducing the time of test execution and provides quick feedback to stakeholders.

### **What are two modes of views in Selenium IDE?**

Selenium IDE can be opened either in side bar (View > Side bar > Selenium IDE) or as a pop up window (Tools > Selenium IDE). While using Selenium IDE in browser side bar it cannot record user operations in a pop up window opened by application.

## **Can I control the speed and pause test execution in Selenium IDE?**

Selenium IDE provides a slider with Slow and Fast pointers to control the speed of execution.

## **Where do I see the results of Test Execution in Selenium IDE?**

Result of test execution can be views in log window in Selenium IDE –

## **Where do I see the description of commands used in Selenium IDE?**

Commands of description can be seen in Reference section –

## **Can I build test suite using Selenium IDE?**

Yes, you can first record individual test cases and then group all of them in a test suite. Following this entire test suite could be executed instead of executing individual tests.

## **What verification points are available with Selenium?**

There are largely three type of verification points available with Selenium –

- ☒ Check for page title
- ☒ Check for certain text
- ☒ Check for certain element (text box, drop down, table etc)

## **I see two types of check with Selenium – verification and assertion, what's the difference between tow?**

- ☒ A verification check lets test execution continue even in the wake of failure with check, while assertion stops the test execution. Consider an example of checking text on page, you may like to use verification point and let test execution continue even if text is not present. But for a login page, you would like to add assertion for presence of text box login as it does not make sense continuing with test execution if login text box is not present.

## **I don't see check points added to my tests while using Selenium IDE☒, how do I get them added to my tests?**

You need to use context menu to add check points to your Selenium IDE tests –

## **How do I edit tests in Selenium IDE?**

There are two ways to edit tests in Selenium IDE; one is the table view while other looking into the source code of recorded commands –

## **What is syntax of command used in Selenium?**

There are three entities associated with a command –

- ② Name of Command
- ② Element Locator (also known as Target)
- ② Value (required when using echo, wait etc)

## **There are tons of Selenium Command, am I going to use all of them**

This entirely boils down to operations you are carrying out with Selenium. Though you would definitely be using following Selenium Commands more often –

- ② Open: opens a web page.
- ② click/clickAndWait: click on an element and waits for a new page to load.
- ② Select: Selects a value from a drop down value.
- ② verifyTitle/assertTitle: verifies/asserts page title.
- ② verify/assert ElementPresent: verifies/asserts presence of element, in the page.
- ② verify/assert TextPresent verifies/asserts expected text is somewhere on the page.

## **How do I use html id and name while using Selenium IDE**

Html id and name can be used as it is in selenium IDE. For example Google search box has name – “q” and id – “list-b” and they can be used as target in selenium IDE –

## **What is XPath? When would I have to use XPath in Selenium IDE?**

XPath is a way to navigate in xml document and this can be used to identify elements in a web page. You may have to use XPath when there is no name/id associated with element on page or only partial part of name/ide is constant.

Direct child is denoted with - /

Relative child is denoted with - //

Id, class, names can also be used with XPath –

- ② //input[@name='q']
- ② //input[@id='lst-b']
- ② //input[@class=' lst']

If only part of id/name/class is constant than “contains” can be used as –

```
② //input[contains(@id,'lst-ib')]
```

### **What is CSS location strategy in Selenium?**

CSS location strategy can be used with Selenium to locate elements, it works using cascade style sheet location methods in which -

Direct child is denoted with – (a space)

Relative child is denoted with - >

Id, class, names can also be used with XPath –

```
② css=input[name='q']
```

```
② css=input[id='lst-ib'] or input#lst-ib
```

```
② css=input[class=' lst'] or input.lst
```

If only part of id/name/class is constant than “contains” can be used as –

```
② css=input[id*=' lst-ib ']
```

Element location strategy using inner text

```
② css = a:contains('log out')
```

### **There is id, name, XPath, CSS locator, which one should I use?**

If there are constant name/id available than they should be used instead of XPath and CSS locators. If not then css locators should be given preference as their evaluation is faster than XPath in most modern browsers.

### **I want to generate random numbers, dates as my test data, how do I do this in Selenium IDE?**

This can be achieved by executing java script in Selenium. Java script can be executed using following syntax –

```
type  
css=input#s  
javascript{Math.random()}
```

And for date –

```
type  
css=input#s  
javascript{new Date()}
```

### **Can I store result of an evaluation and use it later in my test?**

You can use “store” command to achieve this. You can save result of an evaluation in a variable and use it later in your Selenium IDE script. For example we can store value from a text box as following, and later use it to type it in another text box –

```
storeText  
css=input#s  
var1  
type  
css=input#d  
${var1}
```

**I have stored result of an evaluation; can I print it in IDE to check its value?**

You can use echo command as following to check the stored value in Selenium IDE –

```
storeText  
css=input#s  
var1  
echo  
${var1}
```

**Can I handle java script alert using Selenium?**

You could use verify/assertAlert to check presence of alert on page. Since selenium cannot click on “Ok” button on js alert window, the alert itself does not appear on page when this check is carried out.

**Selenium has recorded my test using XPath, how do I change them to css locator?**

You can use drop down available next to Find in Selenium to change element locator used by Selenium –

**I have written my own element locator, how do I test it?**

You can use Find button of Selenium IDE to test your locator. Once you click on it, you would see element being highlighted on screen provided your element locator is right Else one error message would be displayed in log window.

**I have written one js extension; can I plug it in Selenium and use it?**

You could specify your js extension in “Options” window of Selenium IDE –

**How do I convert my Selenium IDE tests from Selenese to another language?**

You could use Format option of Selenium IDE to convert tests in another programming language –

**I have converted my Selenium IDE tests to java but I am not able to execute them, execution options as well as Table tab of Selenium IDE is disabled??**

This is because Selenium IDE does not support execution of test in any other language than Selenese (language of Selenium IDE). You can convert Selenium IDE in a language of your choice and then use Selenium 1.0 to execute these tests in that language.

**I want to use only Selenese as my test script language but still want to execute tests in other browsers, how do I do that?**

You can execute your Selenese test in another browser by specifying the “-htmlSuite” followed by path of your Selenese suite while starting the Selenium Server. Selenium Server would be covered in details in the section about Selenium RC.

**I have added one command in middle of list of commands, how do I test only this new command?**

You can double click on newly added command and Selenium IDE would execute only that command in browser.

**Can I make Selenium IDE tests begin test execution from a certain command and not from the very first command?**

You could set a command as “start” command from context menu. When a command is set as start command then a small green symbol appears before command. Same context menu can be used to toggle this option

**Are there other tools available outside Selenium IDE to help me test my element locators**

You could use XPath checker - <https://addons.mozilla.org/en-US/firefox/addon/xpath-checker/> to test your XPath locators and Firefinder (a firebug add on) to test your CSS locators

<https://addons.mozilla.org/en-US/firefox/addon/firefinder-for-firebug/>  
Firefinder can also be used to test XPath locators.

**What is upcoming advancement in Selenium IDE?**

The latest advancement in Selenium IDE would be to have capabilities of converting Selenium IDE tests in WebDriver (Selenium 2.0) options. This would help generating quick and dirty tests for Selenium 2.0

**How can I use looping option (flow control) in Selenium IDE**

Selenese does not provide support for looping, but there is an extension which could be used to achieve same. This extension can be downloaded from here -  
<http://51elliot.blogspot.com/2008/02/selenium-ide-goto.html>

This extension can be added under “Selenium IDE Extension” section to use loop feature in Selenium IDE.

**Can I use screen coordinate while using click command? I want to click at specific part of my element.**

You would need to use clickAt command to achieve. clickAt command accepts element locator and x, y coordinates as arguments –  
clickAt(locator, coordString)

### **How do I verify presence of drop down options using Selenium?**

Use assertSelectOptions as following to check options in a drop down list –  
assertSelectOptions

### **Can I get data from a specific html table cell using Selenium IDE?**

Use storeTable command to get data from a specific cell in an html table, following example  
store text from cell 0,4 from an html table –

```
storeTable  
css=#tableId.0.4  
textFromCell
```

### **I want to make Selenium IDE record and display css locator followed by other locators, is it possible to give high priority to css locator in Selenium IDE?**

You can change default behavior of Selenium IDE > element locator preference by creating js file with following –

```
LocatorBuilders.order = ['css:name', 'css:id', 'id', 'link', 'name', 'xpath:attributes'];
```

And add this file under “Selenium IDE Extension” under Selenium Options.

### **My application has dynamic alerts which don't always appear, how do I handle them?**

If you want to simulate clicking “ok” on alert than use – chooseOkOnNextConfirmation and

if you want to simulate clicking “cancel” on alert than use -

```
chooseCancelOnNextConfirmation ()
```

### **Can I right click on a locator?**

You can use command - contextMenu ( locator) to simulate right click on an element in web page.

### **How do I capture screen shot of page using Selenium IDE?**

Use command – captureEntirePageScreenshot to take screen shot of page.

### **I want to pause my test execution after certain command.**

Use pause command which takes time in milliseconds and would pause test execution for specified time – pause ( waitTime )

### **I used open command to launch my page, but I encountered time out error?**

This happens because open commands waits for only 30 seconds for page to load. If your application takes more than 30 sec then you can use “setTimeout ( timeout )” to make selenium IDE wait for specified time, before proceeding with test execution.

### **What's the difference between type and typeKeys commands?**

type command simulates enter operations at one go while typeKeys simulates keystroke key by key.

typeKeys could be used when typing data in text box which bring options (like Google suggestion list) because such operation are not usually simulated using type command.

### **What is Selenium RC (also known as Selenium 1.0)?**

Selenium RC is an offering from SeleniumHQ which tries to overcome following draw backs of Selenium IDE –

- ② Able to execute tests only with Firefox
- ② Not able to use full-fledged programming language and being limited to Selenese

### **What are the main components of Selenium RC?**

Selenium RC has two primary components –

- ② Client libraries which let you writes tests in language of your preference i.e. java, C#, perl, php etc
- ② Selenium sever which acts as a proxy between browser and application under test (aut)

### **Why do I need Selenium Server?**

Selenium uses java script to drives tests on a browser; Selenium injects its own js to the response which is returned from aut. But there is a java script security restriction (same origin policy) which lets you modify html of page using js only if js also originates from the same domain as html. This security restriction is of utmost important but spoils the working of Selenium. This is where Selenium server comes to play an important role.

Selenium server stands between aut and browser and injects selenium js to the response received from aut and then it is delivered to browser. Hence browser believes that entire response was delivered from aut.

### **What is Selenium core? I have never used it!!!**

Selenium core is the core js engine of Selenium which executes tests on browser, but because of same origin policy it needs to be deployed on app server itself, which is not always feasible. Hence Selenium core is not used in isolation. Selenium IDE as well as Selenium RC use Selenium core to drive tests while over coming same origin policy. In case of Selenium IDE tests are run in context of browser hence it is not hindered by same origin policy and with Selenium RC, Selenium Server over comes same origin policy.

### **Where is executable for Selenium RC, how do I install it?**

Installation is a misnomer for Selenium RC. You don't install Selenium RC you only add client libraries to your project. For example in case of Java you add client driver and Selenium server jars in Eclipse or IntelliJ which are Java editors.

### **I have downloaded Selenium Server and Client libraries, how do I start Selenium Server?**

To start Selenium Server, you need to navigate to installation directory of Selenium server and execute following command –

```
Java -jar .jar
```

This will start Selenium server at port 4444 by default.

Notice that you need to have Java version 1.5 or higher available on your system to be able to use Selenium server.

### **On my machine port 4444 is not free. How do I use another port?**

You can specify port while running the selenium server as –

```
Java -jar .jar –port 5555
```

### **I am new to programming; can Selenium generate sample code from my Selenese scripts?**

You can first record tests in Selenium IDE and then use format option to convert them in a language of your choice.

### **Can I start Selenium server from my program instead of command line**

If you are using Java then you can start Selenium server using SeleniumServer class. For this you need to instantiate SeleniumServer and then call start() method on it.

```
seleniumServer = new SeleniumServer();
seleniumServer.start();
```

### **And how do I start the browser?**

While using Java you need to create instance of DefaultSelenium class and pass it four parameters –

```
selenium = new DefaultSelenium(serverHost, serverPort, browser, appURL);
selenium.start();
```

Herein you need to pass,

- host where Selenium server is running,
- port of Selenium server,
- browser where tests are to be executed and
- application URL

**I am not using java to program my tests, do I still have to install java on my system?**

Yes, since Selenium server is written in java you need java to be installed on your system to be able to use it. Even though you might not be using java to program your tests.

**What are the browsers offering from Selenium?**

Following browsers could be used with Selenium –

- \*firefox
- \*firefoxproxy
- \*pifirefox
- \*chrome
- \*iexploreproxy
- \*iexplore
- \*firefox3
- \*safariproxy
- \*googlechrome
- \*konqueror
- \*firefox2
- \*safari
- \*piiexplore
- \*firefoxchrome
- \*opera
- \*iehta
- \*custom

Here pi and proxy stand for proxy injection mode of Selenium server

**During execution of tests I see two browser windows; one my test application another window shows commands being executed. Can I limit it to just one window?**

Yes you can instruct Selenium Server to launch just one window. To do so you must specify –singleWindow while starting the Selenium server as –

```
Java -jar .jar --singleWindow
```

**My tests usually time out. Can I specify bigger time out?**

This usually happens while using open method with Selenium. One way to overcome this is to use setTimeOut method in your test script another way is to specify time out while starting Selenium server as following –

```
Java -jar .jar --timeout
```

**My system is behind corporate network, how do I specify my corporate proxy?**

You can specify your corporate proxy as following while starting Selenium Server –

```
java -jar .jar -Dhttp.proxyHost= -Dhttp.proxyPort= -Dhttp.proxyUser= -  
Dhttp.proxyPassword=
```

**I am switching domains in my test scripts. I move from “yahoo.com” to “google.com” and my tests encounter permission denied error?**

Changing domains is also a security restriction from java script. To overcome this you should start your Selenium server in proxy injection mode as –

```
java -jar .jar --proxyInjectionMode
```

Now Selenium server would act as proxy server for all the content going to test application

**Can I execute my Selenese tests in another browser using Selenium server? I want to use only Selenese script my tests but still want to execute my test in non firefox browsers**

Yes you can. To do so you need to specify following parameters while starting Selenium server –

- Browser
  - Test domain
  - Path to html suite (your Selenese tests) and
  - Path to result
- ```
java -jar <>.jar -htmlSuite
```

**Can I log more options during test execution?**

If you want to log browser side option during test execution then you should start Selenium server as following –

```
java -jar <>.jar --browserSideLog
```

**Can I use Selenium RC on my UNIX/Mac also?**

You can use Selenium RC on any system which is capable of running Java. Hence you can use it on RC and UNIX machines also

**I want to test my scripts on new experimental browser, how do I do that?**

You can specify \*custom followed by path to browser to execute your tests on any browser while starting Selenium server

\*custom

**I executed my tests cases but where is the test report?**

Selenium RC itself does not provide any mechanism for test reporting. Test reporting is driven from the framework you use for Selenium. For example with java client driver of Selenium –

- ② if you are using JUnit then you can use ant plug-in of JUnit to generate test report
- ② if you are using TestNG then TestNG generates reports for you

Same reporting option is available with PHP unit and other client libraries you are using.

**How do I use recovery scenarios with Selenium? I used to use them with QTP.**

Power of recovery scenarios lies with the programming language you use. If you are using java then you can use Exception handling to overcome same. For example if you are reading data from a file and file is not available then you should keep your code statements in “try catch” block so that test execution could continue even in the wake of errors. Such mechanism entirely boils down the errors you want to recover from, while being able to continue with test execution.

**How do I iterate through options in my test script.**

You can use loop features of the programming language, for example you can use “for” loop in java as following to type different test data in a text box –

```
// test data collection in an array  
String[] testData = {"test1", "test2", "test3"};  
  
// iterate through each test data  
for (String s : testData) {  
    selenium.type("elementLocator", testData);  
}
```

**Can I execute java script from my tests? I want to count number of images on my page.**

You can use method getEval() to evaluate java script. For example if you want to count number of images then you can pass following dom statement to getEval() as following –

```
selenium.getEval("window.document.images.length");
```

Or to get All anchor objects from a page

```
selenium.getEval("window.document.getElementsByTagName('a');");
```

**Is there a way for me to know all available options when I start Selenium Server?**

If you want to see all options available while starting Selenium server then you should use option “-h” while starting Selenium server -

```
Java -jar .jar -h
```

It would display you all the options which you can use while starting the Selenium server.

### **I have created my own firefox profile; can I execute my test scripts on it**

You may like to create your own firefox profile because Selenium always creates a clean firefox profile while executing the tests and none of your FF settings and plug-in are considered with this clean profile. If you want to execute tests on FF with your settings then you should create custom profile for FF.

To be able to execute tests on a custom firefox profile you should specify its path while starting Selenium server. For example if your new profile is stored at “awesome location” in your directory then you should start Selenium server as following –

```
Java -jar .jar -firefoxProfileTemplate "awesome location"
```

### **How do I capture server side log from Selenium server?**

Start your Selenium Server as following –

```
java -jar .jar -log selenium.log
```

And Selenium would start logging server side info, i.e.

```
20:44:25 DEBUG [12] org.openqa.selenium.server.SeleniumDriverResourceHandler -  
Browser 12345/:top frame1 posted START NEW
```

### **What are Heightened Privileges Browsers?**

Firefox and IE have browser modes which are not restricted by java script's same origin policy. These browsers are known as browsers with elevated security privileges. In case of Firefox it is known as chrome (It's not the Google browser) and in case of IE it is known as iehta

### **My application has lots of pop up window, how do I work with them?**

You need to know the Window ID of pop window to be able to work with them.

First you need to bring control on pop up window; execute selenium commands there, close the pop up window and then bring control back to main window. Consider following example where click on an image brings a pop up window –

```
// click on image brings pop up window  
selenium.click("css=img");  
  
// wait for pop up window identified using anchor target "ss"  
selenium.waitForPopUp("ss", getWaitPeriod());  
selenium.selectWindow("ss");  
  
// Some more operations on popup window  
// Close the pop up window and Select the main application window  
// Main window is selected by adding null as argument  
selenium.close();
```

```
selenium.selectWindow("null");
// continue with usual operation
```

**While trying to execute my tests with firefox I encountered following error – “Firefox Refused Shutdown While Preparing a Profile”. How do I solve it?**

This message simply means that Selenium is not able to launch FF browser as it is already running on your system. To overcome this you should close all running instances of FF browser.

You should also check your system process if there is any hidden FF profile running which is not visible on screen. You should kill all FF processes and following this your tests should run smooth

**While trying to execute my tests with firefox I encountered following error – “Firefox Refused Shutdown While Preparing a Profile”. How do I solve it?**

This message simply means that Selenium is not able to launch FF browser as it is already running on your system. To overcome this you should close all running instances of FF browser.

You should also check your system process if there is any hidden FF profile running which is not visible on screen. You should kill all FF processes and following this your tests should run smooth

**My application uses Ajax heavily how do I use Selenium RC to work with Ajax operations?**

Ajax operations don't reload a page like normal form submission but they make http rets behind the scene. You cannot use waitForPageToLoad for such operations and instead should use conditional wait for change in state of application. This could as well mean waiting for presence of an element before continuing with test operations. Consider following example in which type operation triggers Ajax operation which is followed by conditional wait for presence of a text box –

```
// type operation brings element "q" on screen without loading the page
selenium.type("elementLocator", "testData");
```

```
// conditional wait for element "q"
for (int second = 0; second++) {
    if (second >= 60) fail("timeout");
    try { if (selenium.isElementPresent("q")) break; } catch (Exception e) {}
    Thread.sleep(1000);
}
```

**How do I upload a file using Selenium? I need to upload a word file during test execution.**

If you are using Firefox then you can use “type” command to type in a File Input box of upload file. But type operation does not work with IE and you would have to use “Robot” class in java to work make file upload work.

### **Why do I get “permission denied” error during execution of Selenium tests?**

The primary reason of permission denied error is same origin policy restriction from java script. To overcome this error you can use browsers with elevated security privileges. In case of Firefox you should use \*chrome and in case of IE you should use \*iehta as browser for working with Selenium.

### **I am not able to use “style” attribute to locate element with IE browser**

This is because IE expects attribute values to be in caps while other browsers expect it to be lower case letters. Hence

//tr[@style="background-color:yellow"] works with other browsers

//tr[@style="BACKGROUND-COLOUR:yellow"] works with IE

### **Are there any technical limitations while using Selenium RC?**

Besides notorious “same origin policy” restriction from js, Selenium is also restricted from exercising anything which is outside browser. For example you cannot click on “Tools” option of your browser by just using Selenium.

### **But my tests need me to exercise objects outside browser, how do I achieve it?**

You can use Robot class in java to achieve this, but it would be dirty solution even if you get through this.

### **Does Selenium have any offering for mobile browsers?**

Selenium 2.0 (WebDriver) provides iPhone as well Android drivers which could be used to drive tests on mobile browsers

### **How does Selenium RC stand with other commercial tools?**

The biggest advantage of Selenium RC is that it is absolutely free and has vast support of languages and browsers (almost always). Selenium lags when it comes to test reporting as Selenium does not have any in built reporting but this can be easily achieved using the programming language you decide to work on with Selenium. A bigger drawback is not being able to exercise objects which are outside browser window, for example clicking on folder on your desktop.

### **How does Selenium RC stand with other commercial tools?**

The biggest advantage of Selenium RC is that it is absolutely free and has vast support of languages and browsers (almost always). Selenium lags when it comes to test reporting as Selenium does not have any in built reporting but this can be easily achieved

### **Can I just use Selenium RC to drive tests on two different browsers on one operating system without using Selenium Grid?**

If you are using java client driver of Selenium then java testing framework TestNG lets you achieve this. You can set tests to be executed in parallel using “parallel=test” attribute and

define two different tests, each using a different browser. Whole set up would look as (notice the highlighted sections for browser in test suite)–

```
xml version="1.0" encoding="utf-8"?>
DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >

<suite name="Test Automation" verbose="10">

<parameter name="serverHost" value="localhost" />
<parameter name="appURL" value="http://tamil.yahoo.com"/>
<parameter name="proxyInjection" value="false" />
<parameter name="serverPort" value="4444"/>

<test name="Web Test1">
<parameter name="browser" value="*chrome">
<parameter name="m" value="1">parameter>
<classes><class name="com.core.tests.TestClass1">class>classes>
    test>

<test name="Web Test2">
<parameter name="browser" value="*iehta">
<parameter name="m" value="2">parameter>
<classes><class name="com.core.tests.TestClass1">class>classes>
    test>

<u>suite>
```

### Selenium Grid tions

**How do I cut down text execution time for my selenium tests? I want to execute my tests on a combination of different machines and browsers.**

Selenium grid is your friend. Selenium grid lets you distribute tests across browsers and machines of your choice.

The remaining tions on Selenium Grid are applicable before Selenium 2 was release. To get latest information on Selenium Grid please follow Grid 2 doc.

### **How does Selenium grid works?**

Selenium grid uses combination of Selenium RC servers to execute tests in multiple browsers on different machine. Herein one Selenium RC server works as hub while other RC servers work as slaves, which could be controlled by hub. Whenever there is a ret for a specific configuration for test execution then hub looks for a free RC slave server and if available then test execution begins on it. Once test execution is over then RC slave server would be available for next set of test execution.

**Can you show me one diagram which describes functionality of Selenium grid?**

In the following diagram Selenium hub is controlling three Selenium RC servers which are running for configurations –

- ② IE on Windows
- ② FF on Linux
- ② FF on windows

**Which jar files are needed to works with Selenium GRID?**

You need to download and add following jar files to your Selenium set up to be able to work with Selenium. These jar files are –

- ② selenium-grid-remote-control-standalone-.jar
- ② selenium-grid-hub-standalone-.jar
- ② selenium-grid-tools-standalone-.jar

**How do I start Selenium Grid hub from my machine?**

You should have “ant” set up on your system to be able to work with Grid. Once you have downloaded Selenium Grid, navigate to its distribution directory and execute following command -

```
ant launch-hub
```

This would start grid hub on port 4444 locally. You can verify this by navigating to following URL - <http://localhost:4444/console>

**How do I start Selenium Grid Slave Server from my system?**

Navigate to download directory of Selenium gird and execute following command –

```
ant launch-remote-control
```

This would launch remote control server at port 555. At this point if you navigate to <http://localhost:4444/console> then you would see this remote control listed under “Available Remote Controls”

**How do I start Selenium grid slave on a different port than 5555?**

You can use option “-Dport” followed by port number to start grid slave on a specific port.

```
ant -Dport=1111 launch-remote-control  
ant -Dport=2222 launch-remote-control
```

**How do I start grid RC slaves on a different system than my local host so that hub could control and contact a specific configuration?**

You should specify following configuration while starting RC slave –

```
ant -Dport= -Dhost= -DhubURL= launch-remote-control
```

Herein “hostname” is the host where RC slave is running and “hub url” is URL of machine where grid hub is running.

**How do I specify an environment while starting grid slave machine?**

You could specify an environment using “-Denvironment” while starting a slave machine.  
ant -Denvironment=”Safari on Mac” launch-remote-control  
Herein Safari on Mac is the environment which would be used to recognize configuration of RC slave.

### **How do I use machine specific configuration in my Selenium tests?**

You could specify machine specific configuration while instantiating Selenium as –  
Selenium = new DefaultSelenium("localhost", 4444, \*\*'Safari on Mac'\*\*,  
'http://yahoo.com');

And then you use this selenium instance to carryout operation on your web application.

**But how does my tests know that ‘Safari on Mac’ mean a safari browser? How does mapping between names like ‘Safari on Mac’ and original browser options available in Selenium work?**

Selenium grid uses a file called “grid\_configuration.yml” which defines configurations of all browsers. You would have to add this in your project. This file looks like –

### **How does Selenium grid hub keeps in touch with RC slave machine?**

Selenium grid hub keeps polling all RC slaves at predefined time to make sure they are available for testing. If not then Selenium hub disconnect any unavailable RC slaves and makes it clear that any RC slave is not available for testing. The deciding parameter is called – “remoteControlPollingIntervalInSeconds” and is defined in “grid\_configuration.yml” file.

### **My RC becomes unresponsive at times and my Selenium grid hub keeps waiting for RC slave to respond. How do I let hub know to give up on RC slave after a certain time?**

You could state Selenium grid hub to wait for RC slave for a predefined time, and if RC slave does not responds with in this time then hub disconnects test execution on that slave. This parameter is called “sessionMaxIdleTimeInSeconds” and this parameter can be defined in “grid\_configuration.yml” file.

### **What if my hub goes down while Selenium RC slaves are up and running?**

There is one heart beat mechanism from RC slave to hub which is reciprocal to mechanism used by hub to slave machines. RC slaves use a parameter called “hubPollerIntervalInSeconds” to keep track of running grid hub. This parameter can be defined while starting the hub as –  
ant -DhubPollerIntervalInSeconds= launch-hub  
if hub does not respond within this time then RC slaves deregister themselves from hub.

### **Can Selenium grid be used for performance testing?**

Selenium grid is for functional testing of application across different configuration. Performance testing is usually not carried out on actual devices but on a simulated http ret/response mechanism. If you want to use Selenium Grid for performance testing then you would have to invest heavily on s/w and h/w infrastructure.

**Are there additional logs available while working with Selenium grid?**

You can find Selenium grid hub logs in “log/hub.log” and Remote Control logs in “log/rc-\* .log” folder.

**There are various options available while starting a Selenium server, how do I use them with Selenium grid?**

You can use “seleniumArgs” Java property while launching the remote control and specify any of the option which you would with normal Selenium set up. For example you can run Selenium RC slave in single window mode using following command –

```
ant -DseleniumArgs="-singleWindow -debug" launch-remote-control
```

**I see duplicate entries in my hub console, same RC slave listed more than once :-O**

This is because you are killing RC slave very ferociously. For example you if you just close the console without actually ending the process in more civil manner. To avoid this you should kill all RC slaves in more civil manner. If you encounter this error then you should restart Selenium hub and RC slaves would them register themselves to hub.

**How do I specify my corporate proxy while starting Selenium grid hub or slave machines?**

You could use setting for “http.proxyHost” abd “http.proxyPort” while starting hub and remote control machines –

```
ant -Dhttp.proxyHost= -Dhttp.proxyPort= launch-hub
```

```
ant -Dhttp.proxyHost= -Dhttp.proxyPort= launch-remote-control
```

**How do I use Selenium Grid while using Java, .Net or Ruby**

With java you can take advantage of parallel testing capabilities of TestNG to drive your Selenium grid tests

With .Net you can use “Gallio” to execute your tests in parallel

With Ruby you can use “DeepTest” to distribute your tests

**How about the test report when I use Selenium grid for test execution?**

This entirely boils down to framework you use to write your tests. For example in case of java, TestNG reports should suffice.

**Web Driver (Selenium 2.0) tions****What is Selenium 2.0? I have heard this buzz word many times.**

Selenium 2.0 is consolidation of two web testing tools – Selenium RC and WebDriver, which claims to give best of both words – Selenium and WebDriver. Selenium 2.0 was officially released only of late.

**Why are two tools being combined as Selenium 2.0, what's the gain?**

Selenium 2.0 promises to give much cleaner API than Selenium RC and at the same time not being restricted by java script Security restriction like same origin policy, which have been haunting Selenium from long. Selenium 2.0 also does not warrant you to use Selenium Server.

### **So everyone is going to use Selenium 2.0?**

Well no, for example if you are using Selenium Perl client driver than there is no similar offering from Selenium 2.0 and you would have to stick to Selenium 1.0 till there is similar library available for Selenium 2.0

### **So how do I specify my browser configurations with Selenium 2.0?**

Selenium 2.0 offers following browser/mobile configuration –

- ❑ AndroidDriver,
- ❑ ChromeDriver,
- ❑ EventFiringWebDriver,
- ❑ FirefoxDriver,
- ❑ HtmlUnitDriver,
- ❑ InternetExplorerDriver,
- ❑ iPhoneDriver,
- ❑ iPhoneSimulatorDriver,
- ❑ RemoteWebDriver

And all of them have been implemented from interface WebDriver. To be able to use any of these drivers you need to instantiate their corresponding class.

### **How is Selenium 2.0 configuration different than Selenium 1.0?**

In case of Selenium 1.0 you need Selenium jar file pertaining to one library for example in case of java you need java client driver and also Selenium server jar file. While with Selenium 2.0 you need language binding (i.e. java, C# etc) and Selenium server jar if you are using Remote Control or Remote WebDriver.

### **Can you show me one code example of setting Selenium 2.0?**

Here is java example of initializing firefox driver and using Google Search engine –

```
protected WebDriver webDriver;
```

```
//@BeforeClass(alwaysRun=true)
public void startDriver(){
    webDriver = new FirefoxDriver();
    // Get Google search page and perform search on term "Test"
    webDriver.get("http://www.google.com");
    webDriver.findElement(By.name("q")).sendKeys("Test");
    webDriver.findElement(By.name("btnG")).click();
```

### **Which web driver implementation is fastest?**

HTMLUnitDriver. Simple reason is HTMLUnitDriver does not execute tests on browser but plain http ret – 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

### **What all different element locators are available with Selenium 2.0?**

Selenium 2.0 uses same set of locators which are used by Selenium 1.0 – id, name, css, XPath but how Selenium 2.0 accesses them is different. In case of Selenium 1.0 you don't have to specify a different method for each locator while in case of Selenium 2.0 there is a different method available to use a different element locator. Selenium 2.0 uses following method to access elements with id, name, css and XPath locator –

```
driver.findElement(By.id("HTMLId"));
driver.findElement(By.name("HTMLname"));
driver.findElement(By.cssSelector("cssLocator"));
driver.findElement(By.xpath("XPathLocator"));
```

### **How do I 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.

### **Can I simulate pressing key board keys using Selenium 2.0?**

You can use "sendKeys" command to simulate key board keys as –

```
element.sendKeys(" and some", Keys.ARROW_UP);
```

You can also use "sendKeys" to type in text box as –

```
HTMLelement.sendKeys("testData");
```

### **How do I clear content of a text box in Selenium 2.0?**

You can use "clear" method on text box element to clear its content –

```
textBoxElement.clear();
```

### **How do I select a drop down value using Selenium2.0?**

To select a drop down value, you first need to get the select element using one of element locator and then you can select element using visible text –

```
Select selectElement = new Select(driver.findElement(By.cssSelector("cssSelector")));
selectElement.selectByVisibleText("India");
```

### **What are offering to deal with popup windows while using Selenium 2.0?**

You can use “switchTo” window method to switch to a window using window name. There is also one method “getWindowHandles” which could be used to find all Window handles and subsequently bring control on desired window using window handle –

```
webDriver.switchTo().window("windowName");
```

```
for (String handle : driver.getWindowHandles()) {  
    driver.switchTo().window(handle);  
}
```

### **How about handling frames using Selenium 2.0?**

You can use “switchTo” frame method to bring control on an HTML frame –

```
driver.switchTo().frame("frameName");
```

You can also use index number to specify a frame –

```
driver.switchTo().frame("parentFrame.4.frameName");
```

This would bring control on frame named – “frameName” of the 4<sup>th</sup> sub frame names “parentFrame”

### **Can I navigate back and forth in a browser in Selenium 2.0?**

You can use Navigate interface to go back and forth in a page. Navigate method of WebDriver interface returns instance of Navigation. Navigate interface has methods to move back, forward as well as to refresh a page –

```
driver.navigate().forward();  
driver.navigate().back();  
driver.navigate().refresh();
```

### **What is the order of fastest browser implementation for WebDriver?**

HTMLUnitDriver is the fastest browser implementation as it does not involves interaction with a browser, This is followed by Chrome and Firefox

driver and then IE driver which is slower than FF driver and runs only on Windows.

### **Is it possible to use Selenium RC API with Selenium 2.0?**

You can emulate Selenium 1.0 API with Selenium 2.0 but not all of Selenium 1.0 methods are supported. To achieve this you need to get Selenium instance from WebDriver and use Selenium methods. Method executions might also be slower while simulating Selenium 1.0 with in Selenium 2.0

### **Can you show me one example of using Selenium 1.0 in Selenium 2.0?**

Code Sample:

```
// Create web driver instance  
WebDriver driver = new FirefoxDriver();
```

```

// App URL
String appUrl = "http://www.google.com";

// Get Selenium instance
Selenium selenium = new WebDriverBackedSelenium(driver, appUrl);

// Tests using selenium
selenium.open(appURL);
selenium.type("name=q", "testData");
selenium.click("name=btnG");

// Get back the WebDriver instance
WebDriver driverInstance = ((WebDriverBackedSelenium)
selenium).getUnderlyingWebDriver();

```

**I had support of lots of browsers while using Selenium 1.0 and it seems lacking with Selenium 2.0, for example how do I use < awesome> browser while using Selenium 2.0?**

**Answer**

There is a class called Capabilities which lets you inject new Capabilities in WebDriver. This class can be used to set testing browser as Safari –

```

//Instantiate Capabilities
Capabilities cap = new DesiredCapabilities()

//Set browser name
cap.setBrowserName("this awesome browser");

//Get your browser execution capabilities
CommandExecutor executor
= new SeleneseCommandExecutor("http:localhost:4444/", "http://www.google.com/", cap);

//Setup driver instance with desired Capabilities
WebDriver driver = new RemoteWebDriver(executor, cap);

```

**Are there any limitations while injecting capabilities in WebDriver to perform tests on a browser which is not supported by WebDriver?**

Major limitation of injecting Capabilities is that “findElement” command may not work as expected. This is because WebDriver uses Selenium Core to make “Capability injection” work which is limited by java script security policies.

**Can I change User-Agent while using FF browser? I want to execute my tests with a specific User-Agent setting.**

You can create FF profile and add additional Preferences to it. Then this profile could be passed to Firefox driver while creating instance of Firefox –

```
FirefoxProfile profile = new FirefoxProfile();
profile.addAdditionalPreference("general.useragent.override", "User Agent String");
WebDriver driver = new FirefoxDriver(profile);
```

**Is there any difference in XPath implementation in different WebDriver implementations?**

Since not all browsers (like IE) have support for native XPath, WebDriver provides its own implementation for XPath for such browsers. In case of HTMLUnitDriver and IEDriver, html tags and attributes names are considered lower cased while in case of FF driver they are considered case in-sensitive.

**My application uses ajax highly and my tests are suffering from time outs while using Selenium 2.0.**

You can state WebDriver to implicitly wait for presence of Element if they are not available instantly. By default this setting is set to 0. Once set, this value stays till the life span of WebDriver object. Following example would wait for 60 seconds before throwing ElementNotFoundException –

```
WebDriver driver = new FirefoxDriver();
driver.manage().timeouts().implicitlyWait(60, TimeUnit.SECONDS);
WebElement element = driver.findElement(By.id("elementID"));
```

**What if I don't want to use implicit wait and want to wait only for presence of certain elements?**

You can use explicit wait in this situation to wait for presence of certain element before continuing with test execution. You can use “WebDriverWait” and “ExpectedCondition” to achieve this –

```
WebDriver driver = new FirefoxDriver();
WebElement myDynamicElement = (new WebDriverWait(driver,
60)).until(new ExpectedCondition<WebElement>(){
    @Override
    public WebElement apply(WebDriver d) {
        return d.findElement(By.id("myDynamicElement"));
    }
});
```

This is going to wait up to 60 seconds before throwing ElementNotFoundException.

**What is RemoteWebDriver? When would I have to use it?**

RemoteWebDriver is needed when you want to use HTMLUnitDriver. Since HTMLUnitDriver runs in memory, you would not see a browser getting launched –

```
// Create HTMLUnitDriver instance  
WebDriver driver = new HtmlUnitDriver();  
  
// Launch Yahoo.com  
driver.get("http://www.yahoo.com");
```

### **What all languages available to be used with WebDriver?**

Java and C# are on the forefront of WebDriver languages. Support is also available for Python and Ruby. There is also one java script library available for Firefox.

### **How do I handle java script alert using WebDriver?**

WebDriver would support handling js alerts using Alert interface.

```
// Bring control on already opened alert  
Alert alert = driver.switchTo().alert();  
  
// Get the text of the alert or prompt  
alert.getText();  
  
// Click ok on alert  
alert.accept();
```

### **Could I safely execute multiple instances of WebDriver implementations?**

As far as HTMLUnitDriver and FF drivers are concerned, each instance would be independent of other. In case of IE driver there could be only one instance of IE driver running on Windows. If you want to execute more than one instance of IE driver then you should consider using RemoteWebDriver and virtual machines.

### **Is it possible to interact with hidden elements using WebDriver?**

Since WebDriver tries to exercise browser as closely as real users would, hence simple answer is No, But you can use java script execution capabilities to interact with hidden elements. But beware, this is not how an user interacts with application.

### **I have all my tests written in Selenium 1.0 (Selenium RC), why should I migrate to Selenium 2.0 (WebDriver)?**

Because –

- ❑ WebDriver has more compact and object oriented API than Selenium 1.0
- ❑ WebDriver simulates user behaviour more closely than Selenium 1.0, for example if a text box is disabled WebDriver would not be able to type text in it while Selenium 1.0 would be
- ❑ WebDriver is supported by Browser vendor themselves i.e. FF, Opera, Chrome etc

## **My XPath and CSS locators don't always work with Selenium 2.0, but they used to with Selenium 1.0.**

In case of XPath, it is because WebDriver uses native browser methods unless it is not available. And this cause complex XPath to be broken. In case of Selenium 1.0 css selectors are implemented using Sizzle Library and not all the capabilities like “contains” are available to be used with Selenium 2.0

## **How do I execute Java Script in Selenium 2.0?**

You need to use JavaScriptExecutor to execute java script in Selenium 2.0, For example if you want to find tag name of an element using Selenium 2.0 then you can execute java script as following –

```
WebElement element = driver.findElement(By.id("elementLocator"));
String name = ((JavascriptExecutor) driver).executeScript( "return arguments[0].tagName", element);
```

## **Why does not my java script execution return any value?**

This might happen when you forget to add “return” keyword while executing java script. Notice the “return” keyword in following statement –

```
((JavascriptExecutor) driver).executeScript("return window.title;");
```

## **Are there any limitations from operating systems while using WebDriver?**

While HTMLUnitDriver, FF Driver and Chrome Driver could be used on all operating systems, IE Driver could be used only with Windows.

## **Give me architectural overview of WebDriver.**

WebDriver tries to simulate real user interaction as much as possible. This is the reason why WebDriver does not have “fireEvent” method and “getText” returns the text as a real user would see it. WebDriver implementation for a browser is driven by the language which is best to driver it. In case of FF best fit languages are Javascript in an XPCOM component and in IE it is C++ using IE automation. Now the implementation which is available to user is a thin wrapper around the implementation and user need not know about implementation.

## **What is Remote WebDriver Server?**

Remote WebDriver Server has two components – client and server. Client is WebDriver while Server is java servlet. Once you have downloaded selenium-server-standalone-.jar file you can start it from command line as –

```
java -jar selenium-server-standalone-<version-number>.jar
```

## **Is there a way to start Remote WebDriver Server from my code?**

First add Remote WebDriver jar in your class path. You also need another server called "Jetty" to use it. You can start sever as following –

```
WebApplicationContext context = new WebApplicationContext();
context.setContextPath("");
context.setWar(new File("."));
server.addHandler(context);

context.addServlet(DriverServlet.class, "/wd/*");

SelectChannelConnector connector = new SelectChannelConnector();
connector.setPort(3001);
server.addConnector(connector);

server.start();
```

### **But what are the advantages of using Remote WebDriver over WebDriver?**

You can use Remote WebDriver when –

- ② When you want to execute tests on a browser not available to you locally
- ② Introduction to extra latency to tests

But there is one disadvantage of using Remote WebDriver that you would need external servlet container.

### **Can you show me code example of using Remote WebDriver?**

```
// Any driver could be used for test
DesiredCapabilities capabilities = new DesiredCapabilities();

// Enable javascript support
capabilities.setJavascriptEnabled(true);

// Get driver handle
WebDriver driver = new RemoteWebDriver(capabilities);

// Launch the app
driver.get("http://www.google.com");
```

### **What are the modes of Remote WebDriver**

Remote WebDriver has two modes of operations –

*Client Mode:* This is where language bindings connect to remote instance. FF drive and RemoteWebDriver clients work this way.

*Server Mode:* In this mode language bindings set up the server. ChromeDriver works this way.

### **What Design Patterns could be used while using Selenium 2.0?**

These three Design Patterns are very popular while writing Selenium 2.0 tests –

1. Page Objects – which abstracts UI of web page
2. Domain Specific Language – which tries to write tests which could be understood by a normal user having no technical knowledge
3. Bot Style Tests – it follows “command-like” test scripting

### **So do I need to follow these Design patterns while writing my tests?**

Not at all, these Design Patterns are considered best practices and you can write your tests without following any of those Design Patterns, or you may follow a Design Pattern which suites your needs most.

### **Is there a way to enable java script while using HTMLUnitDriver?**

Use this –

```
HtmlUnitDriver driver = new HtmlUnitDriver();
driver.setJavascriptEnabled(true);
```

or this –

```
HtmlUnitDriver driver = new HtmlUnitDriver(true);
```

### **Is it possible to emulate a browser with HTMLUnitDriver?**

You can emulate browser while using HTMLUnitDriver but it is not recommended as applications are coded irrespective of browser you use. You could emulate Firefox 3 browser with HTMLUnitDriver as –

```
HtmlUnitDriver driver = new HtmlUnitDriver(BrowserVersion.FIREFOX_3);
```

Or you can inject desired capabilities while instantiating HTMLUnitDriver as –

```
HtmlUnitDriver driver = new HtmlUnitDriver(capabilities);
```

### **How do I use iPhone Driver?**

You should start iPhone SDK and build iPhone driver. Download iPhone development tools and provision profile. Now iPhone driver can connect through HTTP to the iPhone simulator.

You can also run simulator on another machine in your network and WebDriver could connect to it remotely.

### **Is it possible to convert Selenium IDE test to WebDriver test?**

~~For now there is no formatter available to convert Selenium IDE tests to corresponding WebDriver tests, hence simple answer is No. Yes WebDriver style of code can be generated from Selenium IDE~~

### **Can WebDriver handle UntrustedSSLCertificates?**

This feature is currently supported in Firefox browser and is awaiting implementation in IE and Chrome drivers.

### **Can I carry out multiple operations at once while using WebDriver?**

You can use Builder pattern to achieve this. For example if you want to move an element from one place to another you can use this –

```
Actions builder = new Actions(driver);

Action dragAndDrop = builder.clickAndHold(element)
    .moveToElement(otherElement)
    .release(otherElement)
    .build();

dragAndDrop.perform();
```

### **How do I simulate keyboard keys using WebDriver?**

There is a KeyBoard interface which has three methods to support keyboard interaction –

- `sendKeys(CharSequence)`- Sends character sequence
- `pressKey(Keys keyToPress)` - Sends a key press without releasing it.
- `releaseKey(Keys keyToRelease)` - Releases a modifier key

### **What about Mouse Interaction?**

Mouse interface lets you carry out following operations –

- `click(WebElement element)` – Clicks an element

- `doubleClick(WebElement element)` - Double-clicks an element.
- `void mouseDown(WebElement element)` - Holds down the left mouse button on an element.
- `mouseUp(WebElement element)` - Releases the mouse button on an element.
- `mouseMove(WebElement element)` - Moves element from current location to another element.
- `contextClick(WebElement element)` - Performs a context-click (right click) on an element.

### **How does Android Webdriver works?**

Please note Android WebDriver has been deprecated in favor of Selendroid. Appium is also a popular library to automate both Android and Appium m-sites and app. Android WebDriver uses Remote WebDriver. Client Side is test code and Server side is application installed on android emulator or actual device. Here client and server communicate using JSON wire protocol consisting of Rest rets.

### **What are the advantages of using Android WebDriver?**

Android web driver runs on Android browser which is best real user interaction. It also uses native touch events to emulated user interaction.

But there are some drawbacks also like, it is slower than headless WebKit driver. XPath is not natively supported in Android web view.

### **Is there a built-in DSL (domain specific language) support available in WebDriver?**

There is not, but you can easily build your own DSL, for example instead of using –

```
webDriver.findElement(By.name("q")).sendKeys("Test");
```

You can create a more composite method and use it –

```
public static void findElementAndType(WebDriver webDriver, String elementLocator, String testData) {
```

```
    webDriver.findElement(By.name(elementLocator)).sendKeys(testData);
}
```

And now you just need to call method `findElementAndType` to do type operation. You may also like to check STF for this

### **What is grid2?**

Grid2 is Selenium grid for Selenium 1 as well as WebDriver, This allows to –

- ② Execute tests on parallel on different machines
- ② Managing multiple environments from one point

### **How do I start hub and slaves machines in grid 2?**

Navigate to you selenium server standalone jar download and execute following command –

```
java -jar selenium-server-standalone-.jar -role hub
```

And you start Slave machine by executing following command –

```
Java -jar selenium-server-.jar -role webdriver -hub http://localhost:4444/grid/register -  
port 6666
```

### **And how do I run tests on grid?**

You need to use the RemoteWebDriver and the DesiredCapabilities object to define browser, version and platform for testing. Create Targeted browser capabilities as –

```
DesiredCapabilities capability = DesiredCapabilities.firefox();
```

Now pass capabilities to Remote WebDriver object –

```
WebDriver driver = new RemoteWebDriver(new URL("http://localhost:4444/wd/hub"),  
capability);
```

Following this, hub will take care of assigning tests to a slave machine

### **What parameters can be passed to grid2?**

You can pass following parameters to grid 2 –

- -port 4444 (default 4444)
- -nodeTimeout (default 30) the timeout in seconds before the hub automatically releases a node that hasn't received any rets for more than the specified number of seconds.
- -maxConcurrent 5 (5 is default) The maximum number of browsers that can run in parallel on the node.

## **Selenium Tool Implementation Misc tions**

### **How do I implement data driven testing using Selenium?**

Selenium, unlike others commercial tools does not have any direct support for data driven testing. Your programming language would help you achieving this. You can use jxl library in case of java to read and write data from excel file. You can also use Data Driven Capabilities of TestNG to do data driven testing.

### **What is equivalent to test step, test scenario and test suite in Selenium.**

If you are using Java client driver of Selenium then TestNG test method could be considered equivalent to test step, a test tag could be considered as test scenario and a suite tag could be considered equivalent to a test suite.

### **How do I get attribute value of an element in Selenium?**

You could usegetAttribute method

With Selenium 1.0 –

```
String var = selenium.getAttribute("css=input[name='q']@maxlength");
System.out.println(var);
```

With Selenium 2.0 –

```
String var =
webDriver.findElement(By.cssSelector("input[name='q']")).getAttribute("maxlength")
System.out.println(var);
```

### **How do I do database testing using Selenium?**

Selenium does not support database testing but your language binding does. For example while using java client driver you can use java data base connectivity (jdbc) to establish connection to data base, fetch/write data to data base and doing data comparison with front end.

### **I completed test execution and now I want to email test report.**

If you are using “ant” build tool then you can use “mail” task to deliver your test results. Similar capabilities are available in Continuous Build Integration tools like – Hudson.

## **How do I make my tests more comprehensible?**

Selenium tests which are written as –

```
selenium.click("addForm:_ID74:_ID75:0:_ID79:0:box");
```

Make it tough to understand the element which is being exercised upon.

Instead of hard coding element locator in tests you should externalize them. For example with java you can use properties file to contain element locators and then locator reference is given in test script. Following this approach previous test step would look as –

```
selenium.click(PolicyCheckbox);
```

And this is far more comprehensible.

## **Why should I use Page Object?**

Page object is a design pattern which distinguishes the code carrying out operations on page and code which carries out tests (assertion/verification). While implementing page object you abstract functioning of a page or part of it in a dedicated “Classs” which is then used by test script to perform actions on page and reach a stage when actual test could be performed.

Advantage of using page object is the fact that if application lay out changes then you only need to modify the navigation part and test would function intact.

### **1: How do you start Selenium RC from command line?**

Selenium command line example

```
// Simple way to start Selenium RC is
```

```
java -jar selenium-server.jar
```

```
// Run a suite of Selenese scripts in a browser
```

```
java -jar selenium-server.jar -htmlSuite
```

### **2: On my machine port 4444 is not free. How do Use another port?**

Selenium run on specific port

```
// You can specify port while running the selenium server as –
```

```
Java -jar selenium-server.jar –port 5555
```

### 3: What is selenium server vs selenium hub?

Selenium server is a standalone application for using a single server as a test node. Selenium hub acts as a proxy in front of one or more Selenium node instances. A hub + node(s) is called a Selenium grid. Running Selenium server is similar to creating a Selenium grid from a hub and a single node on the same host.

### 4: How do you connect to Data base from selenium?

Connecting to database is language dependent. In the below example, we assume that Java is being used.

A Connection object represents a connection with a database. When we connect to a database by using connection method, we create a Connection Object, which represents the connection to the database. An application may have one or more than one connections with a single database or many connections with different databases.

We can use the Connection object for the following things:

- 1). It creates the Statement, PreparedStatement and CallableStatement objects for executing the SQL statements.
- 2). It helps us to Commit or roll back a jdbc transactionn.
- 3). If you want to know about the database or data source to which you are connected then the Connection object gathers information about the database or data source by the use of DatabaseMetaData.
- 4). It helps us to close the data source. The Connection.isClosed() method returns true only if the Connection.close() has been called. This method is used to close all the connection.

Firstly we need to establish the connection with the database. This is done by using the method DriverManager.getConnection(). This method takes a string containing a URL. The DriverManager class, attempts to locate a driver that can connect to the database represented by the string URL. Whenever the getConnection() method is called the DriverManager class checks the list of all registered Driver classes that can connect to the database specified in the URL.

#### **Syntax:**

Selenium database connection example

```
String url = "jdbc: odbc: makeConnection";
```

```
Connection con = DriverManager.getConnection(url, "userID", "password");
```

## **5: What locators are available in Selenium RC?**

1. ID
2. Name
3. CSS (Cascade style sheet)
4. XPATH (Relative xpath and Absolute xpath)
5. Dom

## **6: How do you verify an object present in multiple pages?**

Check on each page assertTrue(selenium.isElementPresent(locator));

## **7: What is the difference between single and double slash in xpath?**

If xpath starts selection from the document node, it'll allow you to create 'absolute' path expressions.

e.g. "/html/body/p" matches all the paragraph elements

If xpath starts selection matching anywhere in the document, then it'll allow you to create 'relative' path expressions.

e.g. "//p" matches all the paragraph elements

## **8: Did you write any User Extensions?**

User extensions are stored in a separate file that we will tell Selenium IDE or Selenium RC to use. Inside there the new function will be written in JavaScript.

Because Selenium's core is developed in JavaScript, creating an extension follows the standard rules for prototypal languages. To create an extension, we create a function in the following design pattern.

Selenium user extension sample

Java

```
// sample
Selenium.prototype.doFunctionName = function(){
}
```

The "do" in front of the function name tells Selenium that this function can be called as a command for a step instead of an internal or private function.

## **9: How do you verify the presence of an element after the successful page loading?**

It can be achieved with the following line of code.

Just mention some time value to check the element (in seconds) like:

```

Java Sample
Java

    public void waitForElementPresent(String element, int timeout)
        throws Exception {

        for (int second = 0; second++) {

            if (second >= timeout)

                fail("Timeout. Unable to find the Specified element" + eleme
nt);

            try {

                if (selenium.isElementPresent(element))
                    break;

            } catch (Exception e) {
            }

            Thread.sleep(1000);

        }

    }

```

#### **10: How do you describe Selenium Grid?**

Selenium Grid is a tool that dramatically speeds up functional testing of web-apps by leveraging your existing computing infrastructure. It allows you to easily run multiple tests in parallel, on multiple machines, in an heterogeneous environment.

Based on the excellent Selenium web testing tool, Selenium Grid allows you to run multiple instances of Selenium Remote Control in parallel. Even better, it makes all these Selenium Remote Controls appear as a single one, so your tests do not have to worry about the actual infrastructure. Selenium Grid cuts down on the time required to run a Selenium test suite to a fraction of the time that a single instance of Selenium instance would take to run.

#### **11: How to start the selenium server from your language class?**

```

Selenium server sample
Java
try {

    seleniumServer = new SeleniumServer();

    seleniumServer.start();

```

```
} catch (Exception e) {  
    e.printStackTrace();  
}
```

## **12: What are the verification points available in Selenium?**

There are largely three types of verification points available with Selenium –

- Check for page title
- Check for certain text
- Check for certain element (text box, drop down, table etc.)

## **13: What is XPath? When would I have to use XPath in Selenium?**

XPath is a way to navigate in xml document and this can be used to identify elements in a web page. You may have to use XPath when there is no name/id associated with element on page or only partial part of name/ide is constant.

Direct child is denoted with – /

Relative child is denoted with – //

Id, class, names can also be used with XPath –

- //input[@name='q']
- //input[@id='lst-ib']
- //input[@class=' lst']

If only part of id/name/class is constant than “contains” can be used as –

- //input[contains(@id,'lst-ib')]

## **14: What is CSS location strategy in Selenium?**

CSS location strategy can be used with Selenium to locate elements, it works using cascade style sheet location methods in which –

Direct child is denoted with – (a space)

Relative child is denoted with – >

Id, class, names can also be used with XPath –

- css=input[name='q']
- css=input[id='lst-ib'] or input#lst-ib
- css=input[class=' lst'] or input.lst

If only part of id/name/class is constant than “contains” can be used as –

- `css=input[id*='lst-ib ']`

Element location strategy using inner text

- `css = a:contains('log out')`

#### **15: There is id, name, XPath, CSS locator, which one should I use?**

If there are unique names or identifier available then they should be used instead of XPath and CSS locators. If not then css locators should be given preference as their evaluation is faster than XPath in most modern browsers.

#### **16: What is the mechanism to handle multiple popups in selenium?**

Multiple popups can be handled by using the command `getWindowHandles()`.

Then store all the window names into `Set<String>` variable and transform it into an array.

Next by using the array index, you can navigate to specific window by using

```
driver.switchTo().window(ArrayIndex);
```

#### **17: How do you handle Ajax controls using selenium?**

Let's consider an example. Say the google test box which is an ajax control and when we enter some text into it, then it displays the auto suggested values.

To work with such controls, you need to capture all the suggested values in a string after entering the value in textbox. Then just split the string and take the values.

#### **18: What are the advantages of Selenium Web driver over Selenium RC?**

Selenium RC's architecture is quite complicated while WebDriver's architecture is simpler than Selenium RC's.

Though Selenium RC is slower since it uses an additional JavaScript program called Selenium Core. On the contrary, WebDriver is faster than Selenium RC since it speaks directly to the browser and uses browser's own engine to control it.

Selenium Core, just like other JavaScript codes, can access disabled elements. Web Driver interacts with page elements in a more realistic way.

Selenium RC's API set is already evolved but contains redundancies and often confusing commands. WebDriver APIs are simpler and do not contain any redundant or confusing commands.

Selenium RC cannot support the headless HtmlUnit browser. It needs a real, visible browser to operate on. Web Driver can support the headless HtmlUnit browser.

Selenium RC has built-in test result generator and it automatically generates an HTML file of test results. Web Driver has no built-in command that automatically generates a Test Results File.

### **19: State any difference between “GET” and “NAVIGATE”?**

Get method will get a page to load or get page source or get text that's all. Whereas navigate will guide through the history like refresh, back, forward. For example if we want to move forward and do some functionality and back to the home page. This can be achieved through navigate() method only. driver.get() will wait till the whole page gets loaded and driver.navigate() will just redirect to that page and will not wait.

### **20: How is the implicit Wait different from Explicit wait?**

Implicit Wait sets internally a timeout that will be used for all consecutive Web Element searches. It will try lookup the element again and again for the specified amount of time before throwing a NoSuchElementException if the element could not have been found. It does only this and can't be forced into anything else – it waits for elements to show up.

Explicit Wait or just Wait is a one-timer used by you for a particular search. It is more extendible in the means that you can set it up to wait for any condition you might like. Usually, you can use some of the prebuilt Expected Conditions to wait for elements to become clickable, visible, invisible, etc., or just write your own condition that suits your needs.

### **21: How to Handle Alerts/Popups in Selenium WebDriver?**

There are two types of alerts which are commonly referred–

- Windows based alert pop ups
- Web based alert pop ups

#### **Web based alert pop ups.**

1. WebDriver offers the users with a very efficient way to handle these pop ups using Alert interface.
2. void dismiss() – The dismiss() method clicks on the “Cancel” button as soon as the pop up window appears.
3. void accept() – The accept() method clicks on the “Ok” button as soon as the pop up window appears.
4. String getText() – The getText() method returns the text displayed on the alert box.

5. void sendKeys(String stringToSend) – The sendKeys() method enters the specified string pattern into the alert box.

#### **Windows based alert pop ups.**

Handling window based pop-ups have always been a little tricky as we know Selenium is an automation testing tool which supports only web application testing, that means, it doesn't support windows based applications and window alert is one of them.

1. Robot class is a java based utility which emulates the keyboard and mouse actions and can be effectively used to handling window based pop up with the help of keyboard events.
2. The keyPress and keyRelease methods simulate the user pressing and releasing a certain key on the keyboard respectively.

#### **22: How to take a screenshot with Selenium WebDriver?**

```
Import org.apache.commons.io.FileUtils;  
  
WebDriver driver = new FirefoxDriver();  
  
driver.get("http://www.google.com/");  
  
File scrFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);  
  
// Now you can do whatever you need to do with it, for example copy somewhere  
  
FileUtils.copyFile(scrFile, new File("c:\\tmp\\Screenshot.png"));
```

#### **23: How to address SSL Certificate issue in Firefox with WebDriver (or) how do you manage the secured connection error in HTTPS?**

SSL Certificate issue in Firefox with WebDriver  
Java

```
FirefoxProfile profile = new FirefoxProfile();  
  
profile.setAcceptUntrustedCertificates(true);  
  
profile.setAssumeUntrustedCertificateIssuer(false);  
  
driver=new FirefoxDriver(profile);
```

#### **24: How to handle SSL certification issue in IE?**

Handle SSL certification issue in IE

Java

```
// Add the below command after opening the browser.
```

```
driver.navigate().to("javascript:document.getElementById('overridelink').click()");
```

#### **26: How to handle AJAX controls in WebDriver?**

AJAX stands for Asynchronous JavaScript and XML. It does not rely on the extra overhead of opening and closing tags that is needed to create valid XML. Most of the time WebDriver automatically handle the Ajax controls and calls/ Incase if it is not able to handle, you can follow the below way to handle.

```
Waiting for Ajax Control  
//Waiting for Ajax Control
```

```
WebElement AjaxElement = (new WebDriverWait(driver, 10)).until(ExpectedConditions.presenceOfElementLocated(By. &lt;locatorType&gt;("&lt;locator Value&gt;")));
```

## **27: How to Mouse over a submenu item of header menu?**

With the actions object you should first move the menu title, and then move to the popup menu item and click it. Don't forget to call actions.perform() at the end. Here's some sample Java code:

### **Mouse over a submenu item of header menu**

```
Actions actions = new Actions(driver);  
  
WebElement menuHoverLink = driver.findElement(By.linkText("Menu heading"));  
  
actions.moveToElement(menuHoverLink);  
  
WebElement subLink = driver.findElement(By.cssSelector("#headerMenu .subLink"));  
  
actions.moveToElement(subLink);  
  
actions.click();  
  
actions.perform();
```

## **28: Can you broadly classify TDD, BDD and DDD frameworks and What's the Difference?**

You would have heard of all these acronyms buzzing all around. Here I'll briefly explain them and tell how exactly they will help in the system test life cycle.

TDD – Test Driven Development.

It's also called test-driven design, is a method of software development in which unit testing is repeatedly done on source code. Write your tests watch it fails and then refactor it. The concept is we write these tests to check if the code we wrote works fine. After each test, refactoring is done and then the same or a similar test is performed again. The process is iterated as many times as necessary until each unit is functionally working as expected. TDD was introduced first by XP. I believe I have explained enough in simple terms.

## BDD – Behavior Driven Development.

Behavior-driven development combines the general techniques and principles of TDD with ideas from domain-driven design. Its purpose is to help the folks devising the system (i.e. the developer) identify appropriate tests to write—that is, tests that reflect the behavior desired by the stakeholders.

## DDD-Domain Driven Development.

DDD is about mapping business domain concepts into software artifacts. A DDD framework offers following benefits:

- Helps the team to create a common model, between the business and IT stakeholders
- The model is modular, extensible and easy to maintain as the design reflects the business model.
- It improves the re-usability and testability of the business domain objects.

## 29: What is Data driven framework & Keyword Driven?

### Data driven framework.

In this framework test case logic resides in test Scripts. 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.

The keyword/table driven framework requires the development of data tables and keywords. They are 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.

## 30: Explain the advantages of TestNG over Junit?

### Advantages of TestNG over Junit.

1. In Junit we have to declare @BeforeClass and @AfterClass. It is a constraint Junit where as in TestNG there is no constraint like this.
2. Additional Levels of setUp/tearDown level are available in TestNG.
  1. @ Before/AfterSuite
  2. @Before/AfterTest and
  3. @Before/AfterGroup
3. There is no need to extend any class in TestNG.
4. There is no method name constraint in TestNG as in Junit.
5. In TestNG we can tell the test that one method is dependent on another method whereas in Junit this is not possible.
6. Grouping of test cases is available in TestNG whereas the same is not available in Junit. Execution can be done based on Groups. For example, if you have defined many

cases and segregated them by defining 2 groups as Sanity and Regression. And if you only want to execute the “Sanity” cases then just tell TestNG to execute the “Sanity”. TestNG will automatically execute the cases belonging to the “Sanity” group.

7. Also TestNG supports parallel test case execution.

### **31: What are the different Parameters for @Test annotation?**

Parameters are keywords that modify the annotation’s function.

### **32: Can we run group of test cases using TestNG?**

Test cases in group using TestNG will be executed with the below options.

If you want to execute the test cases based on one of the group like regression test or smoke test-

```
@Test(groups = {"regressiontest", "smoketest"})
```

### **33: Which web driver implementation is the fastest?**

HTMLUnitDriver. Simple reason is HTMLUnitDriver does not execute tests on browser but use plain http ret-response. It is far quicker than launching a browser and executing tests.

### **34: Is it possible to use Selenium RC API with Selenium 2.0?**

You can emulate Selenium 1.0 API with Selenium 2.0. But not all of the Selenium 1.0 methods are supported. To achieve this you need to get Selenium instance from WebDriver and use Selenium methods. Method executions might also be slower while simulating Selenium 1.0 with in Selenium 2.0.

### **35: How do I use Selenium Grid while using Java, .Net or Ruby?**

- With java you can take advantage of parallel testing capabilities of TestNG to drive your Selenium grid tests.
- With .Net you can use “Gallio” to execute your tests in parallel.
- With Ruby you can use “DeepTest” to distribute your tests.

## **1: What is Selenium 2.0**

Webdriver is open source automation tool for web application. WebDriver is designed to provide a simpler, more concise programming interface in addition to addressing some limitations in the Selenium-RC API.

## **2: What is cost of webdriver, is this commercial or open source.**

selenium is open source and free of cost.

**3:** how you specify browser configurations with Selenium 2.0?

Following driver classes are used for browser configuration

- AndroidDriver,
- ChromeDriver,
- EventFiringWebDriver,
- FirefoxDriver,
- HtmlUnitDriver,
- InternetExplorerDriver,
- iPhoneDriver,
- iPhoneSimulatorDriver,
- RemoteWebDriver

**4:** How is Selenium 2.0 configuration different than Selenium 1.0?

In case of Selenium 1.0 you need Selenium jar file pertaining to one library for example in case of java you need java client driver and also Selenium server jar file. While with Selenium 2.0 you need language binding (i.e. java, C# etc) and Selenium server jar if you are using Remote Control or Remote WebDriver.

**5:** Can you show me one code example of setting Selenium 2.0?

below is example

```
WebDriver driver = new FirefoxDriver();

driver.get("https://www.google.co.in/");

driver.findElement(By.cssSelector("#gb_2 > span.gbts")).click();
```

**6:** Which web driver implementation is fastest?

HTMLUnitDriver. Simple reason is HTMLUnitDriver does not execute tests on browser but plain http ret – 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

**7:** What all different element locators are available with Selenium 2.0?

Selenium 2.0 uses same set of locators which are used by Selenium 1.0 – id, name, css, XPath but how Selenium 2.0 accesses them is different. In case of Selenium 1.0 you don't have to specify a different method for each locator while in case of Selenium 2.0 there is a different method available to use a different element locator. Selenium 2.0 uses following method to access elements with id, name, css and XPath locator –

**8: How do I submit a form using Selenium?**

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

el.submit();
```

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

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

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

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

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

el.clear();
```

**11: How to execute java scripts function.**

```
JavascriptExecutor js = (JavascriptExecutor) driver;

String title = (String) js.executeScript("pass your java scripts");
```

**12: How to select a drop down value using Selenium2.0?**

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

```
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()
```

**14: How to capture element image using Selenium 2.0?**

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

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

int totalRow = rows.size();
```

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

```
String title = driver.getTitle()
```

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

```
String pagesource = driver.getPageSource()
```

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

```
String currentURL = driver.getCurrentUrl()
```

**19: How to assert text 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);
```

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

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

```
//get test from element and stored in text variable
```

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

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

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

```
Actions builder = new Actions(driver);
```

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

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

```
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();
```

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

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

**24: How to verify pdf content using selenium 2.0?**

Go through below post:

**25: How to capture video of running scripts in selenium 2.0?**

**26: How to verify response 200 code using selenium 2.0?**

**27: How to verify image using selenium 2.0?**

**28: How to handle http authentication in selenium 2.0?**

**29: Can we automate HTML 5 video using selenium 2.0?**

**30: What is different between findElement and findElements**

**1 What is Selenium IDE?**

Selenium IDE is an integrated development environment for Selenium tests. It is implemented as a Firefox extension, and has a recording feature, which will keep account of user actions as they are performed and store them as a reusable script to play back. Selenium-IDE also offers full editing of test cases for more precision and control.

**2 Can Selenium test a application on iPhone's Mobile Safari browser?**

Selenium can handle Mobile Safari browser. There is experimental Selenium iPhone Driver for running tests on Mobile with Safari on the iPhone and iPad and iPod Touch.[sociallocker]

**3 Can Selenium test an application on Android browser?**

Selenium can handle Android browser.

**4 What tests can selenium do?**

Selenium could do functional, regression, and load of web based applications.

**5 What are the disadvantages of Selenium?**

Disadvantages of Selenium:

- Limitation in terms of browser support (It runs only in Mozilla). Scripts written using Selenium IDE can be used for other browsers only if it is used with Selenium RC or Selenium Core.
- We can't run recorded script if it is converted to Java, C#, Ruby etc.
- Not allowed to write manual scripts like conditions and Loops for Data Driven Testing
- There is no option to verify images.

## **6 What are the technical challenges with selenium?**

As you know Selenium is a free ware open source testing tool. There are many challenges with Selenium.

1. Selenium supports only web based applications.
2. It doesn't support any non web based (Like Win 32, Java Applet, Java Swing, .Net Client Server etc) applications.
3. When you compare selenium with QTP, Silk Test, Test Partner and RFT, there are many challenges in terms of maintainability of the test cases.
4. Since Selenium is a freeware tool, there is no direct support if one is in trouble with the support of applications.
5. There is no object repository concept in Selenium, so maintainability of the objects is very high
6. There are many challenges if one have to interact with Win 32 windows even when you are working with Web based applications.
7. Bitmap comparison is not supported by Selenium.
8. Any reporting related capabilities, you need to depend on third party tools.
9. You need to learn any one of the native language like (.Net, Java, Perl, Python, PHP, Ruby) to work efficiently with the scripting side of selenium.

## **7 How to run test case recorded using Selenium IDE in other browsers?**

Running test case recorded with the help of Selenium -Remote Control.

## **8 What is the selenium's recording language?**

Selenium's recording language is "HTML".

## **9 What are the steps to run automation using selenium?**

The very basic steps are:

1. Record the test steps using selenium-IDE.
2. Modify the script according to the testing needs. Add validation points, Java Scripts, Time-out etc.
3. Run the test.
4. View the result after test run complete analyze.

## **10 What are the capabilities of Selenium IDE?**

Selenium IDE (Integrated Development Environment) works similar to commercial tools like QTP, Silk Test and Test Partner etc.

The below points describes well about Selenium IDE.

1. Selenium IDE is a Firefox add-on.
2. Selenium IDE can support recording the clicks, typing, and other actions to make a test cases.
3. Using Selenium IDE, a tester can play back the test cases in the Firefox browser.
4. Selenium IDE supports exporting the test cases and suites to Selenium RC.
5. Debugging of the test cases with step-by-step can be done.
6. Breakpoint insertion is possible.
7. Page abstraction functionality is supported by Selenium IDE.
8. Selenium IDE can supports an extensibility capability allowing the use of add-ons or user extensions that expand the functionality of Selenium IDE.

## **11 What is WebDriver?**

WebDriver is a tool for writing automated tests of websites. It aims to mimic the behaviour of a real user, and as such interacts with the HTML of the application.

## **12 So, is it like Selenium? Or Sahi?**

The aim is the same (to allow you to test your webapp), but the implementation is different. Rather than running as a Javascript application within the browser (with the limitations this brings, such as the “same origin” problem), WebDriver controls the browser itself. This means that it can take advantage of any facilities offered by the native platform.

## **13 What is Selenium 2.0?**

Selenium 2 is the latest version of the Selenium project, and includes the IDE, Server and WebDriver.

## **14 How do I migrate from using the original Selenium APIs to the new WebDriver APIs?**

A common question when adopting Selenium 2 is what’s the correct thing to do when adding new tests to an existing set of tests? Users who are new to the framework can begin by using the new WebDriver APIs for writing their tests. But what of users who already have suites of existing tests? This guide is designed to demonstrate how to migrate your existing tests to the new APIs, allowing all new tests to be written using the new features offered by WebDriver.

## **15 Which browsers does WebDriver support?**

The existing drivers are the ChromeDriver, InternetExplorerDriver, FirefoxDriver, OperaDriver and HtmlUnitDriver. For more information about each of these, including their relative strengths and weaknesses, please follow the links to the relevant pages. There is also support for mobile testing via the AndroidDriver, OperaMobileDriver and iPhoneDriver.

## **16 What does it mean to be “developer focused”?**

We believe that within a software application’s development team, the people who are best placed to build the tools that everyone else can use are the developers. Although it

should be easy to use WebDriver directly, it should also be easy to use it as a building block for more sophisticated tools. Because of this, WebDriver has a small API that's easy to explore by hitting the "autocomplete" button in your favourite IDE, and aims to work consistently no matter which browser implementation you use.

## **17 How do I execute Javascript directly?**

We believe that most of the time there is a requirement to execute Javascript there is a failing in the tool being used: it hasn't emitted the correct events, has not interacted with a page correctly, or has failed to react when an XMLHttpRequest returns. We would rather fix WebDriver to work consistently and correctly than rely on testers working out which Javascript method to call.

We also realise that there will be times when this is a limitation. As a result, for those browsers that support it, you can execute Javascript by casting the WebDriver instance to a JavascriptExecutor. In Java, this looks like:

```
WebDriver driver; // Assigned elsewhere  
  
JavascriptExecutor js = (JavascriptExecutor) driver;  
  
js.executeScript("return document.title");
```

Other language bindings will follow a similar approach. Take a look at the [UsingJavascript](#) page for more information.

## **18 Why is my Javascript execution always returning null?**

You need to return from your javascript snippet to return a value, so:

```
js.executeScript("document.title");
```

will return null, but:

```
js.executeScript("return document.title");
```

will return the title of the document.

## **19 My XPath finds elements in one browser, but not in others. Why is this?**

The short is that each supported browser handles XPath slightly differently, and you're probably running into one of these differences. The long is on the [XpathInWebDriver](#) page.[/sociallocker]

### **1) What is test automation or automation testing?**

Automation testing is used to automate the manual testing. It is a process of automating the manual process to test the application/system under test. It uses separate testing tools which facilitate you to create test scripts which can be executed repeatedly and doesn't need any manual intervention.

### **2) What are the advantages of automation testing?**

- It supports execution of repeated test cases.
- It facilitates parallel execution.
- It aids in testing a large test matrix.

- It improves accuracy because there are no chances of human errors.
- It saves time and money.

### 3) What is selenium? What are the different components of selenium?

Selenium is one of the most popular automated testing suites. It is browser automation tool which lets you automated operations like click, type and selection from a drop down of a web page. It is designed to support and encourage automation testing of functionalities of web based applications and a wide range of browsers and platforms. It is one of the most accepted tools amongst the testing professional due to its existence in the open source community.

Selenium is not just a single tool rather than it is a package of several testing tools and that's why it is referred as a suite. Each of these tools is designed to cater different testing and test environment requirement.

### 4) What is Selenium 1.0?

Selenium 1.0 is popularly known as Selenium Remote Control (Selenium RC). It is a library available in wide variety of languages. The main reason to use Selenium RC was that Selenium IDE was incapable to execute tests in browsers other than Selenium IDE and the limitation of language Selenese used in Selenium IDE.

### 5) What is Selenium IDE?

Selenium IDE is a firefox plug-in. It is used to record and replay tests in firefox browser. It is used only with firefox browser.

### 6) What is selenium 2.0?

Selenium 2.0 is a tool which is a combination of web testing tools Selenium RC and WebDriver.

### 7) Why is selenium selected as a test tool?

Selenium is used as a testing tool because:

- It is free and open source.
- It has a large user base and helping communities.
- Compatible on different platforms i.e. Windows, Mac OS, Linux etc.
- Have cross browser compatibility (Chrome, Firefox, IE, Safari etc.)
- Support multiple programming languages ( Java, C#, Ruby, PERL, Python etc.)
- Support distributed testing.

Selenium supports two types of testing:

- Functional Testing
  - Regression Testing
- 9) What are the limitations of selenium?

Selenium has following limitations:

- It can be used only to test web based application.
- Mobile applications can not be tested using selenium.
- You can not test captcha and bar code by using selenium.
- The user must have the knowledge of programming language for using selenium.
- Reports can only be generated using third party tools like TestNG or Junit.

10) What is selenese?

Selenese is a language which is used for writing test script in selenium IDE.

11) Describe the different types of locators in Selenium?

Locator is an address which identifies a web element uniquely within the web page.

There are different types of locators in Selenium to identify web elements accurately and precisely.

These are:

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

12) What is an Xpath?

Xpath is used to locate a web element based on its XML path. It can be used to locate HTML elements.

13) Which is the latest Selenium tool?

The latest Selenium tool is WebDriver.

**14) What are the different types of Drivers that WebDriver contains?**

These are the different drivers available in WebDriver:

- FirefoxDriver
- InternetExplorerDriver
- ChromeDriver
- SafariDriver
- OperaDriver
- AndroidDriver
- iPhoneDriver
- HtmlUnitDriver

**15) What is Selenium Grid?**

Selenium Grid facilitates you to distribute your tests on multiple machines and all of them at the same time. So, you can execute tests on Internet Explorer on Windows and Safari on Mac machine using the same test script. It reduces the time of test execution and provides quick feedback.

**16) Where the result is seen of Test Execution in Selenium IDE?**

The result of test execution is displayed in a Log Window in Selenium IDE.

**17) Can you edit tests in Selenium IDE?**

Yes, tests in Selenium IDE can be edited. There are two ways to edit tests in Selenium IDE.

- By Table views
- Looking into the source code

**18) Can you use html id and name while using Selenium IDE?**

Yes, You can use html id and name as it is available in Selenium IDE.

**19) What are the WebDriver supported Mobile Testing Drivers?**

WebDriver supported "mobile testing drivers" are:

- AndroidDriver
- IphoneDriver
- OperaMobileDriver

20) What are the popular programming languages supported by Selenium WebDriver to write Test Cases?

- o JAVA
- o PHP
- o Python
- o C#
- o Ruby
- o Perl

21) What is the difference between assert and verify commands?

**Assert:** Assert command checks if the given condition is true or false. If the condition is true, the program control will execute the next phase of testing, and if the condition is false, execution will stop and nothing will be executed.

**Verify:** Verify command also checks if the given condition is true or false. It doesn't halt program execution i.e. any failure during verification would not stop the execution and all the test phases would be executed.

22) What is the difference between "/" and "//" in Xpath?

**Single Slash "/":** Single slash is used to create Xpath with absolute path.

**Double Slash "//":** Double slash is used to create Xpath with relative path.

23) How to type in a text box using Selenium?

To type in a text box, we use **sendKeys("String to be entered")** to insert string in the text box.

Syntax:

```
WebElement username = drv.findElement(By.id("Email"));
// entering username
username.sendKeys("sth");
```

24) What is JUnit?

JUnit is an open source testing framework for java applications. It is introduced by Apache.

25) What are the four parameters that you need to pass in Selenium?

The four parameters that you have to pass in Selenium are:

1. Host
2. Port Number
3. Browser
4. URL

26) How can you "submit" a form using Selenium ?

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

27) What is the difference between close() and quit()?

The close() method closes the current browser only whereas quit() method closes all browsers opened by WebDriver.

**1) What are the annotations used in TestNG ?**

Ans: @Test, @BeforeSuite, @AfterSuite, @BeforeTest, @AfterTest, @BeforeClass, @AfterClass, @BeforeMethod, @AfterMethod.

**2) How do you read data from excel ?**

```
FileInputStream fis = new FileInputStream("path of excel file");
```

```
Workbook wb = WorkbookFactory.create(fis);
```

```
Sheet s = wb.getSheet("sheetName");
```

```
String value = s.getRow(rowNum).getCell(cellNum).getStringCellValue();
```

**3) What is the use of xpath ?**

it is used to find the WebElement in web page. It is very useful to identify the dynamic web elements.

**4) What are different types of locators ?**

There are 8 types of locators and all are the static methods of the By class.

```
By.id(), By.name(), By.tagName(), By.className(), By.linkText(), By.partialLinkText(),
By.xpath, By.cssSelector().
```

**5) What is the difference between Assert and Verify?**

Assert- it is used to verify the result. If the test case fail then it will stop the execution of the test case there itself and move the control to other test case.

Verify- it is also used to verify the result. If the test case fail then it will not stop the execution of that test case.

**6) What is the alternate way to click on login button?**

use submit() method but it can be used only when attribute type=submit.

**7) How do you verify if the checkbox/radio is checked or not ?**

We can use isSelected() method.



```
driver.findElement(By.xpath("xpath of the checkbox/radio button")).isSelected();
```

If the return value of this method is true then it is checked else it is not.

**8) How do you handle alert pop-up ?**

To handle alert pop-ups, we need to 1st switch control to alert pop-ups then click on ok or cancel then move control back to main page.

Syntax-

```
String mainPage = driver.getWindowHandle();
```

```
Alert alt = driver.switchTo().alert(); // to move control to alert popup
```

```
alt.accept(); // to click on ok.
```

```
alt.dismiss(); // to click on cancel.
```

//Then move the control back to main web page-

```
driver.switchTo().window(mainPage); → to switch back to main page.
```

**9) How do you launch IE/chrome browser?**

**10) How to perform right click using WebDriver?**

Use Actions class

```
Actions act = new Actions(driver); // where driver is WebDriver type
```

```
act.moveToElement(webElement).perform();
```

```
act.contextClick().perform();
```

**12) Give the example for method overload in WebDriver.**

frame(string), frame(int), frame(WebElement).

**13) How do you upload a file?**

To upload a file we can use sendKeys() method.

```
driver.findElement(By.xpath("input field")).sendKeys("path of the file which u want to upload");
```

**14) How do you click on a menu item in a drop down menu?**

If that menu has been created by using select tag then we can use the methods selectByValue() or selectByIndex() or selectByVisibleText(). These are the methods of the Select class.

If the menu has not been created by using the select tag then we can simply find the xpath of that element and click on that to select.

**15) How do you simulate browser back and forward ?**

```
driver.navigate().back();
```

```
driver.navigate().forward();
```

**16) How do you get the current page URL ?**

```
driver.getCurrentUrl();
```

**17) What is the difference between '/' and '//' ?**

//- it is used to search in the entire structure.

/- it is used to identify the immediate child.

**18) What is the difference between findElement and findElements?**

Both methods are abstract method of WebDriver interface and used to find the WebElement in a web page.

findElement() – it used to find the one web element. It return only one WebElement type.

findElements()- it used to find more than one web element. It return List of WebElements.

**19) How do you achieve synchronization in WebDriver ?**

We can use implicit wait.

Syntax- driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

Here it will wait for 10sec if while execution driver did not find the element in the page immediately. This code will attach with each and every line of the script automatically. It is not required to write every time. Just write it once after opening the browser.

**20) Write the code for Reading and Writing to Excel through Selenium ?**

```
FileInputStream fis = new FileInputStream("path of excel file");
Workbook wb = WorkbookFactory.create(fis);
Sheet s = wb.getSheet("sheetName");
String value = s.getRow(rowNum).getCell(cellNum).getStringCellValue(); // read data
s.getRow(rowNum).getCell(cellNum).setCellValue("value to be set"); //write data
FileOutputStream fos = new FileOutputStream("path of file");
wb.write(fos); //save file
```

**21) How to get typed text from a textbox ?**

usegetAttribute("value") method by passing arg as value.

```
String typedText = driver.findElement(By.xpath("xpath of box")).getAttribute("value");
```

**22) What are the different exceptions you got when working with WebDriver ?**

ElementNotVisibleException, ElementNotSelectableException, NoAlertPresentException, NoSuchElementException, NoSuchWindowException, TimeoutException, WebDriverException etc.

**23) What are the languages supported by WebDriver ?**

Python, Ruby, C# and Java are all supported directly by the development team. There are also webdriver implementations for PHP and Perl.

**24) How do you clear the contents of a textbox in selenium ?**

Use clear() method.

```
driver.findElement(By.xpath("xpath of box")).clear();
```

**25) What is a Framework ?**

A framework is set of automation guidelines which help in

Maintaining consistency of Testing, Improves test structuring, Minimum usage of code, Less Maintenance of code, Improve re-usability, Non Technical testers can be involved in code, Training period of using the tool can be reduced, Involves Data wherever appropriate.

There are five types of framework used in software automation testing:

1-Data Driven Automation Framework

2-Method Driven Automation Framework

3-Modular Automation Framework

4-Keyword Driven Automation Framework

5-Hybrid Automation Framework , its basically combination of different frameworks.  
(1+2+3).

**26) What are the prerequisites to run selenium webdriver?**

JDK, Eclipse, WebDriver(selenium standalone jar file), browser, application to be tested.

**27) What are the advantages of selenium webdriver?**

- a) It supports with most of the browsers like Firefox, IE, Chrome, Safari, Opera etc.
- b) It supports with most of the language like Java, Python, Ruby, C# etc.
- b) Doesn't require to start server before executing the test script.
- c) It has actual core API which has binding in a range of languages.
- d) It supports of moving mouse cursors.
- e) It support to test iphone/Android applications.

**28) What is WebDriverBackedSelenium ?**

WebDriverBackedSelenium is a kind of class name where we can create an object for it as below:

```
Selenium wbdriver= new WebDriverBackedSelenium(WebDriver object name, "URL path of website")
```

The main use of this is when we want to write code using both WebDriver and Selenium RC , we must use above created object to use selenium commands.

**29) How to invoke an application in webdriver ?**

```
driver.get("url"); or driver.navigate().to("url");
```

**30) What is Selenium Grid ?**

Selenium-Grid allows you to run your tests on different machines against different browsers in parallel. That is, running multiple tests at the same time against different machines, different browsers and operating systems. Essentially, Selenium-Grid support distributed test execution. It allows for running your tests in a distributed test execution environment.

**31) How to get the number of frames on a page ?**

```
List<WebElement> framesList = driver.findElements(By.xpath("//iframe"));
int numOfFrames = frameList.size();
```

**32) How do you simulate scroll down action ?**

Use java script to scroll down-

```
JavascriptExecutor jsx = (JavascriptExecutor)driver;
```

```
jsx.executeScript("window.scrollBy(0,4500)", ""); //scroll down, value 4500 you can change as per your req
```

```
jsx.executeScript("window.scrollBy(450,0)", ""); //scroll up
```

**33) What is the command line we have to write inside a .bat file to execute a selenium project when we are using testng ?**

```
java -cp bin;jars/* org.testng.TestNG testng.xml
```

**34) Which is the package which is to be imported while working with WebDriver ?**

```
org.openqa.selenium
```

**35) How to check if an element is visible on the web page ?**

use isDisplayed() method. The return type of the method is boolean. So if it return true then element is visible else not visible.

```
1 driver.findElement(By.xpath("xpath of elemnt")).isDisplayed();
```

**36) How to check if a button is enabled on the page ?**

Use isEnabled() method. The return type of the method is boolean. So if it return true then button is enabled else not enabled.

```
1 driver.findElement(By.xpath("xpath of button")).isEnabled();
```

**37) How to check if a text is highlighted on the page ?**

To identify weather color for a field is different or not-

```
String color = driver.findElement(By.xpath("//a[text()='Shop']")).getCssValue("color");
```

```
String backcolor =
```

```
driver.findElement(By.xpath("//a[text()='Shop']")).getCssValue("background-color");
```

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

```
System.out.println(backcolor);
```

Here if both color and back color different then that means that element is in different color.

**38) How to check the checkbox or radio button is selected ?**

Use isSelected() method to identify. The return type of the method is boolean. So if it return true then button is selected else not enabled.

```
driver.findElement(By.xpath("xpath of button")).isSelected();
```

**39) How to get the title of the page ?**

Use getTitle() method.

Syntax- driver.getTitle();

**40) How do u get the width of the textbox ?**

```
driver.findElement(By.xpath("xpath of textbox ")).getSize().getWidth();
```

```
driver.findElement(By.xpath("xpath of textbox ")).getSize().getHeight();
```

**41) How do u get the attribute of the web element ?**

driver.getAttribute("src") will give you the src attribute of this tag. Similarly, you can get the values of attributes such as title, alt etc.

Similarly you can get CSS properties of any tag by using getCssValue("some property name").

**42) How to check whether a text is underlined or not ?**

Identify by getCssValue("border-bottom") or sometime getCssValue("text-decoration") method if the

cssValue is 'underline' for that WebElement or not.

ex- This is for when moving cursor over element that is going to be underlined or not-

```
public class UnderLine {  
    public static void main(String[] args) {  
        WebDriver driver = new FirefoxDriver();  
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);  
  
        driver.get("https://www.google.co.in/?gfe_rd=ctrl&ei=bXAwU8jYN4W6iAf8zIDgDA&amp;gws_rd=cr");  
  
        String cssValue= driver.findElement(By.xpath("//a[text()='Hindi']")).getCssValue("text-decoration");  
  
        System.out.println("value"+cssValue);  
  
        Actions act = new Actions(driver);  
  
        act.moveToElement(driver.findElement(By.xpath("//a[text()='Hindi']"))).perform();  
  
        String cssValue1= driver.findElement(By.xpath("//a[text()='Hindi']")).getCssValue("text-decoration");  
  
        System.out.println("value over"+cssValue1);  
  
        driver.close();  
    }  
}
```

**43) How to change the URL on a webpage using selenium web driver ?**

```
driver.get("url1");
driver.get("url2");
```

**44) How to hover the mouse on an element ?**

```
Actions act = new Actions(driver);
```

```
act.moveToElement(webElement); //webElement on which you want to move cursor
```

**45) What is the use of getOptions() method ?**

getOptions() is used to get the selected option from the dropdown list.

**46) What is the use of deSelectAll() method ?**

It is used to deselect all the options which have been selected from the dropdown list.

**47) Is WebElement an interface or a class ?**

WebDriver is an Interface.

**48) FirefoxDriver is class or an interface and from where is it inherited ?**

FirefoxDriver is a class. It implements all the methods of WebDriver interface.

**49) Which is the super interface of webdriver ?**

SearchContext.

**50) What is the difference b/w close() and quit()?**

close() – it will close the browser where the control is.

quit() – it will close all the browsers opened by WebDriver.

**1) Can we enter text without using sendKeys() ?**

Yes we can enter text without using sendKeys() method. We have to use combination of javascript and wrapper classes with WebDriver extension class, check the below code-

```
public static void setAttribute(WebElement element, String attributeName, String value)
{
    WrapsDriver wrappedElement = (WrapsDriver) element;
    JavascriptExecutor driver = (JavascriptExecutor)wrappedElement.getWrappedDriver();
    driver.executeScript("arguments[0].setAttribute(arguments[1],arguments[2])",
element, attributeName, value);
}
```

call the above method in the test script and pass the text field attribute and pass the text you want to enter.

**2) There is a scenario whenever “Assert.assertEquals()” function fails automatically it has to take screenshot. How can you achieve this ?**

By using EventFiringWebDriver.

```
EventFiringWebDriver eDriver=new EventFiringWebDriver(driver);
File srcFile = eDriver.getScreenshotAs(OutputType.FILE);
```

```
FileUtils.copyFile(srcFile, new File(imgPath));
```

**3) How do you handle https website in selenium ?**

By changing the setting of FirefoxProfile.

```
public class HTTPSSecuredConnection {  
    public static void main(String[] args){  
        FirefoxProfile profile = new FirefoxProfile();  
        profile.setAcceptUntrustedCertificates(false);  
        WebDriver driver = new FirefoxDriver(profile);  
        driver.get("url");  
    }  
}
```

**4) How to login into any site if its showing any authentication popup for user name and pass ?**

pass the username and password with url.

<http://username:password@url>

ex- <http://creyate:jamesbond007@alpha.creyate.com>

**5) What is the name of Headless browser.**

HtmlUnitDriver.

**6) Open a browser in memory means whenever it will try to open a browser the browser page must not come and can perform the operation internally.**

use HtmlUnitDriver.

ex-

```
public class Memory {  
    public static void main(String[] args) {  
        HtmlUnitDriver driver = new HtmlUnitDriver(true);  
        driver.setJavascriptEnabled(false);  
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);  
        driver.get("https://www.google.co.in/");  
        System.out.println(driver.getTitle());  
    }  
}
```

**7) What are the benefits of using TestNG ?**

- a) TestNG allows us to execute of test cases based on group.
- b) In TestNG Annotations are easy to understand.
- c) Parallel execution of Selenium test cases is possible in TestNG.
- d) Three kinds of report generated
- e) Order of execution can be changed

- f) Failed test cases can be executed
- g) Without having main function we can execute the test method.
- h) An xml file can be generated to execute the entire test suite. In that xml file we can rearrange our execution order and we can also skip the execution of particular test case.

**9) How do you send ENTER/TAB keys in WebDriver ?**

use click() or submit() [submit() can be used only when type='submit']) method for ENTER.  
Or use Actions class to press keys.

For Enter-

```
act.sendKeys(Keys.RETURN);
```

For Tab-

```
act.sendKeys(Keys.ENTER);
```

where act is Actions class type. ( Actions act = new Actions(driver); )

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

**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.

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

- 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.

**12) How to switch back from a frame ?**

use method defaultContent().

Syntax – driver.switchTo().defaultContent();

**13) How to type text in a new line inside a text area ?**

Use \n for new line.

ex- webelement.sendKeys("Sanjay\_Line1.\n Sanjay\_Line2.");

it will type in text box as-

Sanjay\_Line1.

Sanjay\_Line2.

**14) What is the use of AutoIT tool ?**

Some times while doing testing with selenium, we get stuck by some interruptions like a window based pop up. But selenium fails to handle this as it has support for only web based application. To overcome this problem we need to use AutoIT along with selenium script. AutoIT is a third party tool to handle window based applications. The scripting language used is in VBScript.

**15) How to perform double click using WebDriver ?**

use doubleClick() method.

Syntax- Actions act = new

Actions(driver);

act.doubleClick(webelement);

**16) How to press Shift+Tab ?**

String press = Keys.chord(Keys.SHIFT,Keys.ENTER);

webelement.sendKeys(press);

**17) What is the use of contextClick() ?**

It is used to right click.

**18) What is the difference b/w getWindowHandles() and getWindowHandle() ?**

getWindowHandles()- is used to get the address of all the open browser and its return type is Iterator<String>.

getWindowHandle()- is used to get the address of the current browser where the control is and return type is String.

**19) How do you accommodate project specific methods in your framework ?**

1st go through all the manual test cases and identify the steps which are repeating. Note down such steps and make them as methods and write into ProjectSpecificLibrary.

**20) What are different components of your framework ?**

Library- Assertion, ConfigLibrary, GenericLibrary, ProjectSpecificLibrary, Modules.

Drivers folder, Jars folder, excel file.

**21) What are the browsers supported by Selenium IDE ?**

Mozilla FireFox only. Its an Firefox add on.

**22) What are the limitations of Selenium IDE ?**

- a) It does not support looping or conditional statements. Tester has to use native languages to write logic in the test case.
- b) It does not support test reporting, you have to use selenium RC with some external reporting plugin like TestNG or JUnit to get test execution report.
- c) Error handling is also not supported depending on the native language for this.
- d) Only support in Mozilla Firefox only. Its an Firefox add on.

**23) How to check all checkboxes in a page ?**

```
List<webElement> chkBox =
driver.findElements(By.xpath("//htmltag[@attbute='checkbox']"));
for(int i=0; i<=chkBox.size(); i++){
chkBox.get(i).click();
}
```

**24) Count the number of links in a page.**

use the locator By.tagName and find the elements for the tag //a then use loop to count the number of elements found.

Syntax- int count = 0;

```
List<webElement> link = driver.findElements(By.tagName("a"));
```

```
System.out.println(link.size()); // this will print the number of links in a page.
```

**25) How do you identify the Xpath of element on your browser ?**

And- to find the xpath , we use Firebug addons on firefox browser and to identify the xpath written we use Firepath addons.

Syntax- //htmltag[@attname='attvalue'] or //html[text()='textvalue'] or //htmltag[contains(text(),'textvalue')] or //htmltag[contains(@attname,'attvalue')]

**26) What is Selenium Webdriver ?**

WebDriver is the name of the key interface against which tests should be written in Java. All the methods of WebDriver have been implemented by RemoteWebDriver.

**27) What is Selenium IDE ?**

Selenium IDE is a complete integrated development environment (IDE) for Selenium tests. It is implemented as a Firefox Add-On, and allows recording, editing, and debugging tests. It was previously known as Selenium Recorder.

Scripts may be automatically recorded and edited manually providing autocompletion support and the ability to move commands around quickly.

Scripts are recorded in Selenese, a special test scripting language for Selenium. Selenese provides commands for performing actions in a browser (click a link, select an option), and for retrieving data from the resulting pages.

**28) What are the flavors of selenium ?**

selenium IDE, selenium RC, Selenium WebDriver and Selenium Grid.

**29) What is selenium ?**

Its an open source Web Automation Tool. It supports all types of web browsers. Despite being open source its actively developed and supported.

**30) Advantages of selenium over other tools ?**

- a) Its free of cost,
- b) it supports many languages like Java, C#, Ruby, Python etc.,
- c) it allows simple and powerful DOM-level testing etc.

**31) What is main difference between RC and webdriver ?**

Selenium RC injects javascript function into browsers when the web page is loaded.

Selenium WebDriver drives the browser using browser's built-in support.

**32) Why you choose webdriver over RC ?**

- a) Native automation faster and a little less prone to error and browser configuration,
- b) Does not Requires Selenium-RC Server to be running
- c) Access to headless HTMLUnitDriver can allow really fast tests
- d) Great API etc.

**33) Which one is better xpath or CSS ?**

xpath.

**34) How will you handle dynamic elements ?**

By writing relative xpath.

**35) what are the different assertions or check points used in your script ?**

The common types of validations are:

- a) Is the page title as expected,
- b) Validations against an element on the page,
- c) Does text exist on the page,
- d) Does a javascript call return an expected value.

method used for validation – Assert.assertEquals();

**36) What is actions class in WebDriver ?**

Actions class is used to control the actions of mouse.

**37) What is the difference between @BeforeMethod and @BeforeClass ?**

@BeforeMethod- this will execute before every @Test method.

@BeforeClass- this will execute before every class.

**38) What are the different attributes for @Test annotation ?**

alwaysRun, dataProvider, dependsOnMethods, enabled, expectedExceptions, timeOut etc.

ex- @Test(expectedExceptions = ArithmeticException.class)

```
@Test(timeOut = 2000)
```

**39) Can we run group of test cases using TestNG ?**

yes.

**40) What is object repository ?**

An object repository is a very essential entity in any UI automation tool. A repository allows a tester to store all the objects that will be used in the scripts in one or more centralized locations rather than letting them be scattered all over the test scripts. The concept of an object repository is not tied to WET alone. It can be used for any UI test automation. In fact, the original reason why the concept of object repositories were introduced was for a framework required by QTP.

**41) What are oops concepts ?**

- a) Encapsulation,
- b) Abstraction,
- c) Polymorphism,
- d) Inheritance.

**42) What is inheritance ?**

Inherit the feature of any class by making some relations between the class/interface is known as inheritance.

**43) What is difference between overload and override ?**

The methods by passing different arguments list/type is known as overloading of methods while having the same method signature with different method body is known as method overriding.

**44) Does java supports multiple inheritance ?**

Interface supports multiple inheritance but class does not support.

**45) Write a java program for swapping of two numbers ?**

```
public class Swapping{  
    public static void main(String[] args){  
        Scanner in = new Scanner(System.in);  
        System.out.println("enter the 1st num");  
        int a = in.nextInt();  
        System.out.println("enter the 2nd num");  
        int b = in.nextInt();  
        System.out.println("before swapping a="+a+" and b= "+b);  
        int x = a;  
        a = b;  
        b = x;  
  
        System.out.println("After swapping a="+a+" and b= "+b);  
    }  
}
```

**46) Write a java program for factorial of a given number.**

```
public class Factorial{  
    public static void main(String args[]){  
        Scanner in = new Scanner(System.in);  
        System.out.println("enter the num for which u want the factorial");  
        int num = in.nextInt();  
        for(int i=num-1; i>0; i-- ){  
            num = num*i;  
        }  
        System.out.println(num);  
    }  
}
```

**47) What are the different access specifiers in Java?**

private, default, protected and public.

**48) Why do we go for automation testing ?**

Reasons-

- a) Manual testing of all work flows, all fields, all negative scenarios is time and cost consuming.
- b) It is difficult to test for multi lingual sites manually.
- c) Automation does not require human intervention. We can run automated test unattended(Overnight).
- d) Automation increases speed of test execution.
- e) Automation helps increase test coverage.
- f) Manual testing can become boring and hence error prone.

**49) What is testing strategy ?**

A Test Strategy document is a high level document and normally developed by project manager. This document defines "Software Testing Approach" to achieve testing objectives. The Test Strategy is normally derived from the Business Requirement Specification document.

**50) write a code to make use of assert if my username is incorrect.**

```
try{  
    Assert.assertEquals(expUserName, actUserName);  
}catch(Exception e){  
    System.out.println("name is invalid");  
}
```

**1. What is Selenium?**

Selenium is a suite of tools for browser automation. It is composed of "IDE", a recording and playback mechanism, "WebDriver" and "RC" which provide APIs for browser automation in a wide variety of languages, and "Grid", which allows many tests using the APIs to be run in parallel. It works with most browsers, including Firefox from 3.0 up to 7, Internet Explorer 8, Google Chrome, Safari and Opera 11.5

## **2. Describe technical problems that you had with Selenium tool?**

As with any other type of test automation tools like SilkTest, HP QTP, Watir, Canoo Webtest, Selenium allows to record, edit, and debug tests cases. However there are several problems that seriously affect maintainability of recorded test cases, occasionally Quality Assurance Engineers complain that it takes more time to maintain automated test cases than to perform manual testing; however this is an issue with all automated testing tools and most likely related to improper testing framework design. Another problem is complex ID for an HTML element. If IDs is auto-generated, the recorder test cases may fail during playback. The work around is to use XPath to find required HTML element. Selenium supports AJAX without problems, but QA Tester should be aware that Selenium does not know when AJAX action is completed, so ClickAndWait will not work. Instead QA tester could use pause, but the snowballing effect of several 'pause' commands would really slow down total testing time of test cases. The best solution would be to use waitForElement.

## **3. What test can Selenium do?**

Selenium could be used for the functional, regression, load testing of the web based applications. The automation tool could be implemented for post release validation with continuous integration tools like Jenkins, Hudson, QuickBuild or CruiseControl.

## **4. What is the price of Selenium license per server?**

Selenium is open source software, released under the Apache 2.0 license and can be downloaded and used without charge.

## **5. How much does Selenium license cost per client machine?**

Selenium is open source software, released under the Apache 2.0 license and can be downloaded and used without charge.

## **6. Where to download Selenium?**

Selenium can be downloaded and installed for free from [seleniumhq.org](http://seleniumhq.org)

## **7. What is the latest version of Selenium?**

The latest versions are Selenium IDE 1.3.0, Selenium Server (formerly the Selenium RC Server) 2.8.0, Selenium Client Drivers Java 2.8.0, Selenium Client Drivers C# 2.8.0, Selenium Client Drivers Ruby 2.8.0, Selenium Client Drivers Python 2.8.1, Selenium Grid 1.0.8.

## **8. What is Selenium IDE?**

Selenium IDE is a Firefox add-on that records clicks, typing, and other actions to make a test cases, which QA Tester can play back in the Firefox browser or export to Selenium RC. Selenium IDE has the following features: record/play feature, debugging with step-by-step and breakpoints, page abstraction functionality, an extensibility capability allowing the use of add-ons or user extensions that expand the functionality of Selenium IDE

## **9. What are the limitations of Selenium IDE?**

Selenium IDE has many great features and is a fruitful and well-organized test automation tool for developing test cases, in the same time Selenium IDE is missing certain vital features of a testing tool: conditional statements, loops, logging functionality, exception handling, reporting functionality, database testing, re-execution of failed tests and screenshots taking capability. Selenium IDE doesn't work for IE, Safari and Opera browsers.

## **10. What does SIDE stand for?**

Selenium IDE. It was a very tricky interview question.

## **11. What is Selenium Remote Control (RC) tool?**

Selenium Remote Control (RC) is the powerful solution for test cases that need more than simple browser actions and linear execution. Selenium-RC allows the developing of complex test scenarios like reading and writing files, querying a database, and emailing test reports. These tasks can be achieved by tweaking test cases in your preferred programming language.

## **12. What are the advantages using Selenium as testing tool?**

If QA Tester would compare Selenium with HP QTP or Micro Focus SilkTest, QA Engineer would easily notice tremendous cost savings for Selenium. In contrast to expensive SilkTest license or QTP license, Selenium automation tool is absolutely free. It means that with almost no investment in purchasing tools, QA Team could easily build the state of the art test automation infrastructure. Selenium allows developing and executing test cases in various programming languages including .NET, Java, Perl, RubyPython, PHP and even HTML. This is a great Selenium advantage, most likely your software developers already know how to develop and maintain C# or Java code, so they transfer coding techniques and best practices to QA team. Selenium allows simple and powerful DOM-level testing and in the same time could be used for testing in the traditional waterfall or modern Agile environments. Selenium would be definitely a great fit for the continuous integration tools Jenkins, Hudson, CruiseControl, because it could be installed on the server testing box, and controlled remotely from continuous integration build.

## **13. What is Selenium Grid?**

Selenium Grid extends Selenium RC to distribute your tests across multiple servers, saving you time by running tests in parallel.

## **14. What is Selenium WebDriver?**

Selenium WebDriver is a tool for writing automated tests of websites. It is an API name and aims to mimic the behaviour of a real user, and as such interacts with the HTML of the application. Selenium WebDriver is the successor of Selenium Remote Control which has

been officially deprecated.

**15. How many browsers are supported by Selenium IDE?**

Test Engineer can record and playback test with Selenium IDE in Firefox.

**16. Can Selenium test an application on iPhone's Mobile Safari browser?**

Selenium should be able to handle Mobile Safari browser. There is experimental Selenium iPhone Driver for running tests on Mobile Safari on the iPhone, iPad and iPod Touch.

**17. Can Selenium test an application on Android browser?**

Selenium should be able to handle Android browser. There is experimental Selenium Android Driver for running tests in Android browser.

**18. What are the disadvantages of using Selenium as testing tool?**

Selenium weak points are tricky setup; dreary errors diagnosis; tests only web applications

**19. How many browsers are supported by Selenium Remote Control?**

QA Engineer can use Firefox 7, IE 8, Safari 5 and Opera 11.5 browsers to run actual tests in Selenium RC.

**20. How many programming languages can you use in Selenium RC?**

Several programming languages are supported by Selenium Remote Control - C# Java Perl PHP Python Ruby

**21. How many testing framework can QA Tester use in Selenium RC?**

Testing frameworks aren't required, but they can be helpful if QA Tester wants to automate test cases. Selenium RC supports Bromine, JUnit, NUnit, RSpec (Ruby), Test::Unit (Ruby), TestNG (Java), unittest (Python).

**22. How to developer Selenium Test Cases?**

Using the Selenium IDE, QA Tester can record a test to comprehend the syntax of Selenium IDE commands, or to check the basic syntax for a specific type of user interface. Keep in mind that Selenium IDE recorder is not clever as QA Testers want it to be. Quality assurance team should never consider Selenium IDE as a "record, save, and run it" tool, all the time anticipate reworking a recorded test cases to make them maintainable in the future.

**23. What programming language is best for writing Selenium tests?**

The web applications may be written in Java, Ruby, PHP, Python or any other web framework. There are certain advantages for using the same language for writing test cases as application under test. For example, if the team already have the experience with Java, QA Tester could always get the piece of advice while mastering Selenium test cases in Java.

Sometimes it is better to choose simpler programming language that will ultimately deliver better success. In this case QA testers can adopt easier programming languages, for example Ruby, much faster comparing with Java, and can become experts as soon as possible.

#### **24. Have you read any good books on Selenium?**

There are several great books covering Selenium automation tool, you could check the review at Best Selenium Books: Top Recommended page

#### **25. Do you know any alternative test automation tools for Selenium?**

Selenium appears to be the mainstream open source tool for browser side testing, but there are many alternatives. Canoo Webtest is a great Selenium alternative and it is probably the fastest automation tool. Another Selenium alternative is Watir, but in order to use Watir QA Tester has to learn Ruby. One more alternative to Selenium is Sahi, but it has confusing interface and small developers community.

#### **26. Compare HP QTP vs Selenium?**

When QA team considers acquiring test automation to assist in testing, one of the most critical decisions is what technologies or tools to use to automate the testing. The most obvious approach will be to look to the software market and evaluate a few test automation tools. Read Selenium vs QTP comparison

#### **27. Compare Borland Silktest vs Selenium?**

Check Selenium vs SilkTest comparison

#### **28. How to test Ajax application with Selenium**

Ajax interview tions could be tough for newbie in the test automation, but will be easily cracked by Selenium Tester with a relevant experience. Read the detailed approach at Testing Ajax applications with Selenium in the right way

#### **29. How can I learn to automate testing using Selenium?**

Don't be surprised if the interviewer asks you to describe the approach for learning Selenium. This interviewer wants to hear how you can innovative software test automation process the company. Most likely they are looking for software professional with a good Selenium experience, who can do Selenium training for team members and get the team started with test automation. I hope this Selenium tutorial will be helpful in the preparation for this Selenium interview tion.

#### **30. What are the main components of Selenium testing tools?**

Selenium IDE, Selenium RC and Selenium Grid

**31. What is Selenium IDE?**

Selenium IDE is for building Selenium test cases. It operates as a Mozilla Firefox add on and provides an easy to use interface for developing and running individual test cases or entire test suites. Selenium-IDE has a recording feature, which will keep account of user actions as they are performed and store them as a reusable script to play back.

**32. What is the use of context menu in Selenium IDE?**

It allows the user to pick from a list of assertions and verifications for the selected location.

**33. Can tests recorded using Selenium IDE be run in other browsers?**

Yes. Although Selenium IDE is a Firefox add on, however, tests created in it can also be run in other browsers by using Selenium RC (Selenium Remote Control) and specifying the name of the test suite in command line.

**34. What are the advantage and features of Selenium IDE?**

- a. Intelligent field selection will use IDs, names, or XPath as needed
- b. It is a record & playback tool and the script format can be written in various languages including C#, Java, PERL, Python, PHP, HTML
- c. Auto complete for all common Selenium commands
- d. Debug and set breakpoints
- e. Option to automatically assert the title of every page
- f. Support for Selenium user-extensions.js file

**35. What are the disadvantage of Selenium IDE tool?**

- a. Selenium IDE tool can only be used in Mozilla Firefox browser.
- b. It is not playing multiple windows when we record it.

**36. What is Selenium RC (Remote Control)?**

Selenium RC allows the test automation expert to use a programming language for maximum flexibility and extensibility in developing test logic. For example, if the application under test returns a result set and the automated test program needs to run tests on each element in the result set, the iteration / loop support of programming language's can be used to iterate through the result set, calling Selenium commands to run tests on each item. Selenium RC provides an API and library for each of its supported languages. This ability to use Selenium RC with a high level programming language to develop test cases also allows the automated testing to be integrated with the project's automated build environment.

**37. What is Selenium Grid?**

Selenium Grid in the selenium testing suit allows the Selenium RC solution to scale for test suites that must be run in multiple environments. Selenium Grid can be used to run multiple instances of Selenium RC on various operating system and browser configurations.

**38. How Selenium Grid works?**

Selenium Grid sent the tests to the hub. Then tests are redirected to an available Selenium RC, which launch the browser and run the test. Thus, it allows for running tests in parallel with the entire test suite.

**39. What you say about the flexibility of Selenium test suite?**

Selenium testing suite is highly flexible. There are multiple ways to add functionality to Selenium framework to customize test automation. As compared to other test automation tools, it is Selenium's strongest characteristic. Selenium Remote Control support for multiple programming and scripting languages allows the test automation engineer to build any logic they need into their automated testing and to use a preferred programming or scripting language of one's choice.

Also, the Selenium testing suite is an open source project where code can be modified and enhancements can be submitted for contribution.

**40. What test can Selenium do?**

Selenium is basically used for the functional testing of web based applications. It can be used for testing in the continuous integration environment. It is also useful for agile testing

**41. What is the cost of Selenium test suite?**

Selenium test suite a set of open source software tool, it is free of cost.

**42. What browsers are supported by Selenium Remote Control?**

The test automation expert can use Firefox, IE 7/8, Safari and Opera browsers to run tests in Selenium Remote Control.

**43. What programming languages can you use in Selenium RC?**

C#, Java, Perl, PHP, Python, Ruby

**44. What are the advantages and disadvantages of using Selenium as testing tool?**

Advantages: Free, Simple and powerful DOM (document object model) level testing, can be used for continuous integration; great fit with Agile projects.

Disadvantages: Tricky setup; dreary errors diagnosis; can not test client server applications.

**45. What is difference between QTP and Selenium?**

Only web applications can be testing using Selenium testing suite. However, QTP can be used for testing client server applications. Selenium supports following web browsers: Internet Explorer,

Firefox, Safari, Opera or Konqueror on Windows, Mac OS X and Linux. However, QTP is limited to Internet Explorer on Windows.

QTP uses scripting language implemented on top of VB Script. However, Selenium test suite has the flexibility to use many languages like Java, .Net, Perl, PHP, Python, and Ruby.

**46. What is difference between Borland Silk test and Selenium?**

Selenium is completely free test automation tool, while Silk Test is not. Only web applications can be tested using Selenium testing suite. However, Silk Test can be used for testing client server applications. Selenium supports following web browsers: Internet Explorer, Firefox, Safari, Opera or Konqueror on Windows, Mac OS X and Linux. However, Silk Test is limited to Internet Explorer and Firefox.

Silk Test uses 4Test scripting language. However, Selenium test suite has the flexibility to use many languages like Java, .Net, Perl, PHP, Python, and Ruby.

**47. What is the difference between an assert and a verify with Selenium commands?**

Effectively an “assert” will fail the test and abort the current test case, whereas a “verify” will fail the test and continue to run the test case.

**48. If a Selenium function requires a script argument, what would that argument look like in general terms?**

StoreEval(script, variable) and storeExpression(expression, variableName)

**49. If a Selenium function requires a pattern argument, what five prefixes might that argument have?**

glob, regexp, exact, regexp

**50. What is the regular expression sequence that loosely translates to "anything or nothing?"**

(.\* i.e., dot star) This two-character sequence can be translated as “0 or more occurrences of any character” or more simply, “anything or nothing.”

**51. What is the globbing sequence that loosely translates to "anything or nothing?"**

(\*) which translates to “match anything,” i.e., nothing, a single character, or many characters.

**52. What does a character class for all alphabetic characters and digits look like in regular expressions?**

[0-9] matches any digit

[A-Za-z0-9] matches any alphanumeric character

[A-Za-z] matches any alphabet character

**53. What does a character class for all alphabetic characters and digits look like in globbing?**

[0-9] matches any digit

[a-zA-Z0-9] matches any alphanumeric character

[a-zA-Z] matches any alphabet character

**54. What must one set within SIDE in order to run a test from the beginning to a certain point within the test?**

Set Toggle BreakPoint.

**55. What does a right-pointing green triangle at the beginning of a command in SIDE indicate?**

Play Entire Test Suite

**56. Which wildcards does SIDE support?**

\*, []

**57. What are the four types of regular expression quantifiers which we've studied?**

Ans : \* quantifier: 0 or more of the preceding character (or group)

+ quantifier: 1 or more of the preceding character (or group)

? quantifier: 0 or 1 of the preceding character (or group)

{1,5} quantifier: 1 through 5 of the preceding character (or group)

**58. What regular expression special character(s) means "any character?"**

(.\*)

**59. What distinguishes between an absolute and relative URL in SIDE?**

Absolute URL: Its is base url and this represent domain address.

Relative URL: (Absolute URL + Page Path).

Open command uses Base URL (Absolute URL) to navigate web page.

**60. How would one access a Selenium variable named "count" from within a JavaScript snippet?**

`\${count}

**61. What Selenese command can be used to display the value of a variable in the log file, which can be very valuable for debugging?**

echo()

**62. If one wanted to display the value of a variable named `in` in the log file, what would the first argument to the previous command look like?**

`echo()`

**63. Which Selenium command(s) simulates selecting a link?**

`click`, `clickAndWait`, `ClickAt`, `ClickAtandWait`, `DoubleClick`, `DoubleClickandWait`,  
`doubleClickAt`, `doubleClickAtandWait`

**64. Which two commands can be used to check that an alert with a particular message popped up?**

The following commands are available within Selenium for processing Alerts:

- `getAlert()`
- `assertAlert()`
- `assertAlertNotPresent()`
- `assertAlertPresent()`
- `storeAlert()`
- `storeAlertPresent()`
- `verifyAlert()`
- `verifyAlertNotPresent()`
- `verifyAlertPresent()`
- `waitForAlert()`
- `waitForAlertNotPresent()`
- `waitForAlertPresent()`

The `AlertPresent()` and `AlertNotPresent()` functions check for the existence or not of an alert – regardless of its content. The `Alert()` functions allow the caller to specify a pattern which should be matched. The `getAlert()` method also exists in Selenium RC, and returns the text from the previous Alert displayed.

## **1. What are the different Selenium components?**

The suite package constitutes of the following sets of tools:

- **Selenium Integrated Development Environment (IDE)** – 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** – 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** – Selenium Grid is used to distribute your test execution on multiple platforms and environments concurrently.

## **2. What test can selenium do?**

Selenium could be used for the functional, regression, load testing of the web based application. The automation tool could be implemented for the post release validation with a continuous integration tol like Jenkins, Hudson, QuickBuild or cruiseControl.

## **3. What programming languages are best for writing Selenium tests?**

The Web Application may be written in JAVA, Ruby, PHP, Python or any other web framework. There are certain advantages for writing test cases as application tests. For example, if the team already has the experience with Java, QA Tester could always get the piece of advice while mastering Selenium test cases in Java. When QA Testers use the same programming language with application developers, they can get support from developers to develop the test application quicker.

## **4. What are the test types supported by the Selenium?**

Types of tests supported by Selenium are listed below:

Selenium could be used for testing the web based applications. The test types can be supported are:

1. **Functional**—Functional testing is a quality assurance (QA) process and a type of black-box testing that bases its test cases on the specifications of the software component under test.
2. **Regression**— Regression testingis a type of softwaretesting that seeks to uncover new software bugs, or regressions, in existing functional and non-functional areas of a system after changes such as enhancements, patches or configuration changes, have been made to them.
3. **Load testing**— Load testingis the process of putting demand on a software system or computing device and measuring its response.Load testing is performed to determine a system's behavior under both normal and anticipated peak load conditions.

The automation tool could be implemented for post release validation with continuous integration tools like:

1. **Jenkins**— It is an open source continuous integration tool written in Java. The project was forked from Hudson after a dispute with Oracle.
2. **Hudson**— It is a continuous integration (CI) tool written in Java, which runs in a servlet container, such as Apache Tomcat or the GlassFish application server.
3. **QuickBuild**— QuickBuild provides an easy and flexible way to set up builds. When manage builds for many projects, QuickBuild's ability to inherit and override build definitions makes the job incredibly easy. With the help of scripting and variables, QuickBuild is able to address almost all kinds of build scenarios.
4. **CruiseCont**— It is a Java-based framework for a continuous build process. It includes, but is not limited to, plugins for email notification, Ant, and various source control tools. A web interface is provided to view the details of the current and previous builds. It allows one to perform a continuous integration of any software development process.

## **5. How do I submit a form using Selenium?**

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

## 6. 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 methods.
- **@After:** Method annotated as `@After` lets the system know that this method shall be executed every time after each of the test methods.
- **@BeforeClass:** Method annotated as `@BeforeClass` lets the system know that this method shall be executed once before any of the test methods.
- **@AfterClass:** Method annotated as `@AfterClass` lets the system know that this method shall be executed once after any of the test methods.
- **@Ignore:** Method annotated as `@Ignore` lets the system know that this method shall not be executed.

## 7. Can Selenium test an application on Android browser?

Selenium should be able to handle Android browser. There is a Selenium Android Driver for running tests in the Android browser.

You can use Selendroid or Appium framework to test native apps or web apps in the Android browser.

Here is some code sample for it.

```
@BeforeClass

public static void startSelendroidServer() throws Exception {
    if (selendroidServer != null) {
        selendroidServer.stopSelendroid();
    }

    SelendroidConfiguration config = new SelendroidConfiguration();
    config.addSupportedApp("selendroid-test-app-0.9.0.apk");
    selendroidServer = new SelendroidLauncher(config);
    selendroidServer.launchSelendroid();

    SelendroidCapabilities caps = new SelendroidCapabilities("io.selendroid.testapp:0.9.0");
    driver = new SelendroidDriver(caps);
}
```

## 8. When do we use `findElement()` and `findElements()`?

**findElement():** findElement() is used to find the first element in the current web page matching to the specified locator value. Take a note that only first matching element would be fetched.

**Syntax:**

```
WebElement element =driver.findElements(By.xpath("//div[@id='example']//ul//li"));
```

**findElements():** findElements() is used to find all the elements in the current web page matching to the specified locator value. Take a note that all the matching elements would be fetched and stored in the list of WebElements.

**Syntax:**

```
List <WebElement> elementList  
=driver.findElements(By.xpath("//div[@id='example']//ul//li"));
```

## 1. What are the driver.close and driver.quit in WebDriver? Which is more preferable?

driver.close and driver.quit are two different methods for closing the browser session in Selenium WebDriver.

- **driver.close** – It closes the browser window on which the focus is set.
- **driver.quit** – It basically calls driver.dispose method which in turn closes all the browser windows and ends the WebDriver session gracefully.

We should use driver.quit whenever we want to end the program. It will close all opened browser windows and terminates the WebDriver session. If we do not use driver.quit at the end of the program, WebDriver session will not close properly and files would not be cleared off memory. This may result in memory leak errors.

## 2. How is Selenium 2.0 configuration different than Selenium 1.0?

In case of Selenium 1.0 we need a Selenium jar file pertaining to one library, for example, in case of Java you need a Java client driver and also Selenium server jar file.

## 3. Can tests recorded using Selenium IDE be run in other browsers?

Yes. Although Selenium IDE is a Firefox add on, however, the tests created in it can also be run in other browsers by using Selenium RC (Selenium Remote Control) and specifying the name of the test suite in command line.

For example:

**with command line**

```
c:\Program Files\Mozilla Firefox\firefox.exe" -chrome "chrome://selenium-  
ide/content/selenium/TestRunner.html?baseUrl=http://localhost&test=file:///c:\te  
sts\testsuite.html&auto=false"
```

**With Java program**

```
C:\Program Files (x86)\Java\jre6\bin\java.exe" -jar c:\seltest\selenium-server-  
standalone-2.18.0.jar -htmlSuite "*firefox" "http://127.0.0.1"  
"c:\seltest\mytestsuite.html" "c:\seltest\logs\results-firefox.html" -port 5555
```

## 4. 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. 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();
```

## 5. How to set the test case priority in TestNG?

Setting Priority in TestNG

**Code Snippet**

```
package TestNG;

import org.testng.annotations.*;

public class SettingPriority {

    @Test(priority=0)
    public void method1() {
    }

    @Test(priority=1)
    public void method2() {
    }

    @Test(priority=2)
    public void method3() {
    }
}
```

**Test Execution Sequence:**

1. Method1
2. Method2
3. Method3

## 6. How to automate radio button in Selenium 2.0?

```
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()
```

## 7. How to handle the AJAX popup window?

By using getWindowHandles() and obj.switchTo.window(windowid), we can handle popups using explicit wait and driver.switchTo.window("name") commands for your requirements.

## Selenium Overview

### 1. Which is the only browser in which Selenium IDE can be used ?

- A) Safari B) Firefox C) Google Chrome D) Internet Explorer

Firefox

### 2. Selenium supports which kinds of applications?

- A) Web Based applications B) Stand alone applications C) Both D) None of the above

Web Based Applications

### 3. Find the type of text pattern used in Selenium

- A) Chronological Sequence B) Globbing C) Simulation D) None of the above

Globbing

Will be updated with more tions then and there. Keep visiting

## Selenium RC

### 1. Which of the following command is a “Onevent Handler”

- A) focus() B) assertAlert() C) fireEvent() D) alert()
- fireEvent()

### 2. Selenium server is developed in

- A) Perl B) Java C) Python D).Net

Ans is Java

## Selenium WebDriver

### 1. Which Driver implementation will allow headless mode

- A) FireFoxDriver() B) HtmlUnitDriver() C) SafariDriver D) ChromeDriver
- : HTMLUnitDriver

Will be updated with more tions then and there. Keep visiting

## **1) What is Selenium IDE ?**

Selenium IDE (Integrated Development Environment) is an ideal tool used to develop selenium test scripts... It is the only flavor of selenium which allows us to record user actions on browser window... Here, the scripts are recorded in 'Selenese'(a set of selenium commands like Click, assertTextPresent, storeText, etc.,..) It is not only a time-saver but also an excellent way of learning Selenium scripts syntax... It is a Firefox add-On...

## **2) How to install Selenium IDE ?**

I use seleniumhq.org site to download any software regarding selenium.. Here, if we click on the version number of selenium ide it starts downloading the add-ons, just we have to click on 'install now' button and restart the firefox browser.. With-in 5 minutes we can install selenium ide...

## **3) How do you open/start selenium-ide after installation ?**

First launch/open firefox browser and then click on 'Tools' tab in the menu bar.. You can see an option called 'selenium ide' , click on it, a small window will be opened where you can start recording scripts...

## **4) Features of selenium IDE..**

detailed features of selenium ide

- Its main feature is record and playback..
- Debugging features by setting breakpoints, startpoints, pause..
- Identifies element using id, name, xpath etc..
- Auto complete for all common selenium commands..
- It has an option of asserting title of every page automatically..
- Support for selenium user-extensions.js file..
- Option to run single test or multiple tests(suite).
- It has a feature of exporting testcase/suite into different formats like C#, Java, Ruby, Python..

## **5) Advantages and disadvantages of selenium IDE..**

**Advantages :**

- Freeware..
- Easy to install..
- It is the only flavor of selenium that allows us to record user actions on browser window..

- Scripts recorded in IDE can be converted into other languages like Java, C#, Python and Ruby..
- It is not only a time saver but also an excellent way of learning scripts syntax..
- We can insert comments in the middle of the script for better understanding and debugging..
- Context menu, which allows us to pick from a list of assertions and verifications for the selected location..

**Disadvantages:**

- Main disadvantage of IDE is that it runs only in firefox browser..
- Selenium IDE can execute scripts created in selenese only.
- Cannot upload files..
- It does not directly support loops and conditions..
- Reading from external files like .txt, .xls is not possible..

**6) What are the selenium locators and what is the tool you use to identify element ?**

Selenium locators are the way of finding HTML element on the page to perform Selenium actions... We use firebug(for firefox) to identify elements as it is more popular and powerful web development tool.. It inspects HTML and modify style and layout in real-time.. We can edit, debug and monitor CSS, HTML and Javascript live in any web page.. ((click here to download firebug))

-->For Internet Explorer we can choose debugbar.. It views HTML DOM tree, we can view and edit tab attributes..

**7) How do you locate elements in IDE ?**

I will focus on the unique attribute values like id, name or other structural information that is stable enough to withstand frequent changes to the web application.. I strongly recommend CSS selectors as locating strategy.. They are considerably faster than xpath and can find the most complicated objects in any HTML document..

**8) What is selenese ?**

Selenium set of commands that run our test is called Selenese.. A sequence of these commands is a test script.. There are three types of selenese..

1. **Actions :** They perform some operations like clicking a link or typing text in text box or selecting an option from drop-down box etc..

2. **Assertions** : They verify that the state of application conforms to what is expected..

Ex: 'verify that this checkbox is checked', 'make sure that the page title is X'..

3. **Accessors** : Checks the state of application and store the results in a variable.. Ex:

storeText, storeTitle, etc..

### **9) How do you add check points or verification points ?**

They are called as Assertions in selenium.. 'assert', 'verify' and 'waitFor' are the commands used to add check points..

Ex: assertText, verifyElementPresent, waitForTextPresent, etc..

### **10) Name some commands that you use frequently in IDE ?**

Open, click, type, select, assertText, assertTitle, assertTextPresent, verifyText, verifyTextPresent, verifyTitle, waitForText, waitForTextPresent, waitForTitle, store, storeText, storeTitle, check, uncheck, pause, mouseover, etc..

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

When an 'assert' fails, the test is aborted whereas when 'verify' fails, the test will continue execution logging the failure..

'assert' is used when the expected value is mandatory to continue with next set of steps..

However 'verify' is used when the expected value is optional to continue with the next set of steps..

### **12) Difference between waitFor and pause commands ?**

'waitFor' command waits for some condition to become true..

For example, 'waitForPageToLoad(20000)'-- it will wait upto 20 thousand milliseconds to load required page.. If the page is loaded before 20,000ms then it jumps to next step to execute.. If the page is not loaded before 20,000ms then it stops the execution due to time-out error..

--> pause command stops the execution of the test until the specified time..

Ex: `pause(10000)`-- It stops the execution of test for 10 thousand milliseconds.. After completing 10,000ms it jumps to next command to execute.. I prefer 'waitFor' command than 'pause'..

### **13) How do you export tests from Selenium IDE to RC ?**

First i will open the test in IDE, which should be exported to RC.. There after i'll select 'File' from the menu bar.. when we mouseover on 'Export Test Case As' in the file menu, we could see different languages like C#, Java, Python and Ruby.. Select the language you want to export your test and provide the name to save it..

### **1: What is Selenium?**

Selenium is a browser based functional test automation tool. It is basically a library which you can use in your program to test a web application. It is important to note that selenium is mainly used for browser automation. It is NOT used for unit testing or API testing.

Selenium Webdriver has many language bindings, which means you can write your tests in your favorite programming language and using the respective selenium bindings.

### **2: Another interesting selenium interview is how many languages does selenium WebDriver support?**

The main ones are: Java, C#, php, Ruby, Python

### **3: In Selenium WebDriver, how do you select an item from a drop-down menu?**

We can select an item from the drop-down menu by Value, by Index or by Visible Text.

Example:

```
<select id="cars">
<option value="vo">Volvo</option>
<option value="sa">Saab</option>
<option value="me">Mercedes</option>
```

```
<option value="au">Audi</option>
</select>
```

```
WebElement cars = driver.findElement(By.id("cars"));
Select car = new Select(cars)
```

**//select by value**

```
car.selectByValue("vo"); //this will select Volvo from drop-down
```

**//select by index**

```
car.selectByIndex(2); //this will select Saab
```

**//select by visible text**

```
car.selectByVisibleText("Audi") //this will select Audi
```

**4: What is the difference between driver.get() and driver.navigate.to()**

You can use both methods to navigate to a url.

Because navigating to a URL is very common, then driver.get() is a convenient way. However it does the same function as the driver.navigate().to("url")

The driver.navigate() also has other functions, such as

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

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

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

**5: What is the difference between implicit wait and explicit wait in selenium webdriver?**

This question is asked in almost all selenium interview questions because many web applications use AJAX.

Selenium Webdriver introduces the concept of waits for AJAX based applications. There are two waiting methods, implicit wait and explicit wait

**Implicit wait:**

When an implicit wait is implemented in tests, if WebDriver cannot find an element in the Document Object Model (DOM), it will wait for a defined amount of time for the element to appear in the DOM. In other terms, an implicit wait polls the DOM for a certain amount of time when trying to find an element or elements if they are not immediately available.

Implicit waits can slow down your tests, because once set, the implicit wait is set for the life of the WebDriver object's instance. This means that when an application responds normally, it will still wait for each element to appear in the DOM which increases the overall execution time.

Another downside is if for example you set the waiting limit to be 5 seconds and the elements appears in the DOM in 6 seconds, your tests will fail because you told it to wait a maximum of 5 seconds.

An example of an implicit wait is:

```
1 driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
```

#### **Explicit wait:**

Explicit waits are better than implicit wait. Unlike an implicit wait, you can write custom code or conditions for wait before proceeding further in the code.

An explicit wait can be used where synchronization is needed, for example the page is loaded but we are still waiting for a call to complete and an element to appear.

Selenium WebDriver provides WebDriverWait and ExpectedCondition classes for implementing an explicit wait. The ExpectedCondition class provides a set of predefined conditions to wait before proceeding further in the code.

An example of an explicit wait is:

```
WebDriverWait wait = new WebDriverWait(driver, 10);
wait.until(ExpectedConditions.titleContains("selenium"));
```

To summarize:

Implicit wait time is applied to all elements in your script and Explicit wait time is applied only for particular specified element.

## **6: How would you count the number of elements on a page?**

This is a common selenium interview question because in many cases you want to click on an item from a list. So it is important to know how to count the elements and select the correct one from the list.

We first need to locate the node element where the items are listed. For example, in the html code below:

```
<ul  
id="movies">  
    <li>movie  
    title 1</li>  
    <li>movie  
    title 2</li>  
    .  
    .  
    .  
    <li>movie  
    title 50</li>  
  </ul>
```

The root element can be located using

```
List<WebElement> movies = driver.findElements(By.cssSelector("ul#movies li"))
```

Then we can use the `.size()` to get the number of `<li>` elements

```
int numberOfMovies = movies.size();
```

## **7: How can you check if a check-box or a radio button is selected?**

We can use the `.isSelected()` method, e.g.

```
driver.FindElement(By.id("id_of_checkbox")).isSelected();
```

## **8: Is there a way to do drag and drop in selenium WebDriver?**

Yes, we can use the following code to do drag and drop

```
Actions action = new Actions(driver);
WebElement start = driver.findElement(By.cssSelector("div.source"));

WebElement target = driver.findElement(By.cssSelector("div.target"));
action.dragAndDrop(start,target).perform();
```

**9: How would you check if an element is visible on the page?**

We can use `isDisplayed()` method. The return type of the method is boolean. So if it return true then element is visible otherwise it is not.

```
driver.findElement(By.id("id_of_element")).isDisplayed();
```

**10: How to check if a button is enabled on the page ?**

We can Use `isEnabled()` method. The return type of the method is boolean. So if it return true then button is enabled else not enabled.

```
driver.findElement(By.id("id_of_element")).isEnabled();
```

Selenium is an open source software test automation tool. Here you will get all the Selenium interview tions and s:

***Q1. What do you know about Selenium?***

Selenium is a set of different software tools each with a different approach to supporting test automation. The entire suite of tools results in a rich set of testing functions specifically geared to the needs of testing of web applications of all types. These operations are highly flexible, allowing many options for locating UI elements and comparing expected test results against actual application behavior. One of Selenium's key features is the support for executing one's tests on multiple browser platforms.

Selenium Suite of projects includes:

Selenium IDE: For Record and Run

Selenium Core: more a component of selenium than a standalone project. Without going into the project history, Selenium was once just a collection of .js files that automated a browser. No one uses these directly; they're just there for legacy reasons.

Selenium1: known as. Selenium RC or Remote Control now deprecated

Selenium2: Updated version of Selenium RC known as. Selenium Webdriver

Selenium-Grid: Selenium Grid allows you to run your tests in parallel, that is, different tests can be run at the same time on different remote machines.

## ***Q2. What is the difference between Selenium Remote Control and Selenium Server?***

While discussing the core difference between Selenium RC and Selenium WebDriver we will focus on some related terms as well:

Selenium Core: more a component of selenium than a stand alone project. Without going into the project history, Selenium was once just a collection of .js files that automated a browser. No one uses these directly, they're just there for legacy reasons.

Selenium IDE: a Firefox plugin for record/playback. You may want to start with this, to get used to the api, but you'll outgrow it soon

Selenium RC and when you do outgrow it, you'll use Selenium Remote Control. Selenium 1.x is a client-server architecture. You use the RC libraries to program tests that communicate with the server, and the server relays those commands to a browser.

Selenium Grid: a way to run Selenium testing on a distributed network of computers. Good for speeding things up once you've got a lot of tests.

Cubic Test: An eclipse-based tool that leverages selenium for testing. Not sure how popular it is.

Bromine: a web based script and test management tool. Uses selenium RC to run tests.

Then we get to the Selenium 2 beta. Selenium 2 is a major departure from the Selenium 1 model because it doesn't require a Selenium server. I say 'require' because it's optional to run the tests remotely on another computer. Selenium Server Standalone is the server you'd use for this. It's compatible with Selenium-RC as well as Selenium 2 for remote purposes.

You may have seen Selenium 2 referred to as WebDriver. WebDriver was another project that was merged a couple years ago and became the basis for Selenium 2. That's why Selenium 2 has a WebDriver interface, sometimes called the "WebDriver" api to distinguish from Selenium-RC.

## ***Q3. What are the test types supported by Selenium?***

Selenium could be used for testing the web based applications. The test types can be supported are:

1. functional,

2. regression,
3. load testing

The automation tool could be implemented for post release validation with continuous integration tools like:

1. Jenkins,
2. Hudson,
3. QuickBuild
4. CruiseCont

***Q4. What are the technical challenges with selenium?***

As you know Selenium is a free ware open source testing tool. There are many challenges with Selenium.

1. Selenium supports only web based applications
2. It doesn't support any non-web based (Like Win 32, Java Applet, Java Swing, .Net Client Server etc) applications
3. When you compare selenium with QTP, Silk Test, Test Partner and RFT, there are many challenges in terms of maintainability of the test cases
4. Since Selenium is a freeware tool, there is no direct support if one is in trouble with the support of applications
5. There is no object repository concept in Selenium, so maintainability of the objects is very high
6. There are many challenges if one have to interact with Win 32 windows even when you are working with Web based applications
7. Bitmap comparison is not supported by Selenium
8. Any reporting related capabilities, you need to depend on third party tools
9. You need to learn any one of the native language like (.Net, Java, Perl, Python, PHP, Ruby) to work efficiently with the scripting side of selenium.

***Q5. What is the Difference between Absolute path & Relative path?***

Absolute path will start with root path (/) and Relative path will from current path (//)

Absolute xPath:

```
/html/body/div[3]/div[2]/div[2]/div[2]/div[2]/div[2]/div/div[4]/div[1]/div/div[@id='main']/div[@id='Blog1']/div[1]/div[1]/div/div[1]/div/h3/a
```

Relative xPath : //h3/a[text()='Working on New Window']

***Q6.What are the features of TestNG?***

TestNG is a testing framework designed to simplify a broad range of testing needs, from

unit testing (testing a class in isolation of the others) to integration testing (testing entire systems made of several classes, several packages and even several external frameworks, such as application servers). You can use test suite annotations, to automatically generate test run report and much more.

***Q7. Please tell me the difference between implicitly Wait and Explicit wait.***

Implicit Wait sets internally a timeout that will be used for all consecutive Web Element searches. It will try lookup the element again and again for the specified amount of time before throwing a NoSuchElementException if the element could not have been found. It does only this and can't be forced into anything else – it waits for elements to show up.

Explicit Wait or just Wait is a one-timer used by you for a particular search. It is more extendible in the means that you can set it up to wait for any condition you might like. Usually, you can use some of the prebuilt Expected Conditions to wait for elements to become clickable, visible, invisible, etc., or just write your own condition that suits your needs.

***Q8. Tell me some TestNG Annotations.***

**@BeforeSuite:** The annotated method will be run before all tests in this suite have run.

**@AfterSuite:** The annotated method will be run after all tests in this suite have run.

**@BeforeTest:** The annotated method will be run before any test method belonging to the classes inside the <test> tag is run.

**@AfterTest:** The annotated method will be run after all the test methods belonging to the classes inside the <test> tag have run.

**@BeforeGroups:** The list of groups that this configuration method will run before. This method is guaranteed to run shortly before the first test method that belongs to any of these groups is invoked.

**@AfterGroups:** The list of groups that this configuration method will run after. This method is guaranteed to run shortly after the last test method that belongs to any of these groups is invoked.

**@BeforeClass:** The annotated method will be run before the first test method in the current class is invoked.

**@AfterClass:** The annotated method will be run after all the test methods in the current class have been run.

**@BeforeMethod:** The annotated method will be run before each test method.

**@AfterMethod:** The annotated method will be run after each test method.

***Q8. What is Selenium IDE? Mention some disadvantage.***

Selenium IDE is basic tool that is generally used for Record and Run Test for Learning purpose.

- Selenium IDE (Integrated Development Environment) is a Firefox add on which supports record/playback mechanism as some other commercial tools like QTP/UFT, TestComplete etc.
- By using selenium IDE user can record and playback test cases in Firefox browser.
- Tester can export the test cases from selenium IDE in any of the native language supported by selenium. These tests can then be modified according to the test case.
- Tester can insert breakpoints in IDE.

**Disadvantages of Selenium IDE**

- It does not support looping or conditional statements. Tester has to use native languages to write logic in the test case.
- It does not support test reporting, you have to use selenium WebDriver with some external reporting plugin like TestNG or JUnit to get test execution report.
- Error handling is also not supported depending on the native language for this.
- Only support in Mozilla Firefox only. Its an Firefox add on.

***Q9. What are the advantage and features of Selenium IDE?***

- Intelligent field selection will use IDs, names, or XPath as needed
- It is a record & playback tool and the script format can be written in various languages including C#, Java, PERL, Python, PHP, HTML
- Auto complete for all common Selenium commands
- Debug and set breakpoints
- Option to automatically assert the title of every page
- Support for Selenium user-extensions.js file

***Q10. Name some test automation tools apart from Selenium?***

- HP Unified Functional Testing(UFT)/QTP (Commercial)
- IBM Rational Functional Tester (Commercial)
- TestComplete (By Smart Bear)
- HP WinRunner (Commercial)
- SilkTest (Commercial)
- HTTP Test Tool (Open Source)
- WATIR (Open Source)
- Maverryx (Open Source)

- eggPlant (Commercial)

#### ***Q11. What is WebDriver?***

The primary new feature in Selenium 2.0 is the integration of the WebDriver API. WebDriver is designed to provide a simpler, more concise programming interface in addition to addressing some limitations in the Selenium-RC API. Selenium-WebDriver was developed to better support dynamic web pages where elements of a page may change without the page itself being reloaded. WebDriver's goal is to supply a well-designed object-oriented API that provides improved support for modern advanced web-app testing problems.

#### ***Q13 What are the types of text patterns available in Selenium?***

There are three types of patterns available in Selenium:

1. globbing
2. regular expressions
3. exact

#### ***Q14. What are globs? How does Selenium Support Globbing?***

We used globs while searching files in DOS by using the command \*.doc or US\*.doc and so on. So let's see how Selenium supports globbing.

Globbing in Selenium is possible with the following set of characters

- \* (asterisk): It is used to match any number of characters. E.g. \*.doc will indicate a.doc or arkansas.doc
- ? (question mark)

It is used to match a single character. E.g. Se?.doc will indicate Set.doc. It supports just a single character.

- [ ] (square brackets)

Typically a square bracket is used to denote a class of characters or set of characters.

- [a-z] means a set of lowercase alphabets
- [A-Z] means a set of uppercase alphabets
- [0-9] indicates a set of numeric values

#### ***Q15. How to use globs in Selenium?***

Suppose we want to find/ search for the text "America" or "American", then we could use it as Ame\*

In Selenium, that would mean the following:

Command	Target	Value
verifyTextPresent	Glob: Ame*	

### ***Q16. Comparative discussion Between UFT/QTP vs Selenium?***

Only web applications can be tested using Selenium testing suite. However, UFT/QTP can be used for testing client server applications as well as Web Applications. Selenium supports following web browsers: Internet Explorer, Firefox, Safari, Opera or Konqueror on Windows, Mac OS X and Linux. However, UFT/QTP supports Internet Explorer/Chrome/Firefox on Windows.

QTP uses scripting language implemented on top of VB Script. However, Selenium test suite has the flexibility to use many languages like Java, .Net, Perl, PHP, Python, and Ruby.

#### **1) Which is the command used for displaying the values of a variable into the output console or log?**

The command used for displaying the values of a variable into the output console or log—  
echo

If you want to display a constant string. The below mentioned command can be used  
echo <constant string>

ex: echo “The sample message”

If you want to display the value of a variable it can be written like below

echo \${<variable name>}

ex: echo \${var1}

#### **2) What are the capabilities of Selenium WebDriver or Google WebDriver or Selenium 2.0?**

Capabilities of Selenium WebDriver or Google WebDriver or Selenium 2.0 are:

One should use WebDriver when requiring improved support for

1. Mult-browser testing including improved functionality for browsers not well-supported by Selenium-1.0.
2. Handling multiple frames, multiple browser windows, pop-ups, and alerts.
3. Page navigation.
4. Drag-and-drop.
5. AJAX-based UI elements.

#### **3) Which are the browsers supported by Selenium RC?**

Browsers supported by Selenium RC are:

1. \*firefox
2. \*mock
3. \*firefoxproxy

4. \*pifirefox
5. \*chrome
6. \*iexploreproxy
7. \*iexplore
8. \*firefox3
9. \*safariproxy
10. \*googlechrome
11. \*konqueror
12. \*firefox2
13. \*safari
14. \*piexplore
15. \*firefoxchrome
16. \*opera
17. \*iehta
18. \*custom

#### **4) What are the Operating Systems supported by Selenium?**

Operating Systems supported by Selenium are:

##### **Selenium IDE**

Works in Firefox 2+ Start browser, run tests Run tests

Operating Systems Supported:

1. Windows,
2. OS X
3. Linux
4. Solaris
5. Others whichever supports Firefox 2+

##### **Selenium Remote Control**

Used for starting browser and run tests

Operating Systems Supported:

1. Windows,
2. OS X
3. Linux
4. Solaris
5. Others

##### **Selenium Core**

Used for running tests

Operating Systems Supported:

1. Windows,
2. OS X
3. Linux
4. Solaris
5. Others

#### **5) What is the difference between an assert and a verify with Selenium commands?**

**Assert:** Will fail and abort the current test execution.

**Verify:** Will fail and continue to run the test execution.

**6) If a Selenium function requires a script argument, what would that argument look like in general terms?**

StoreEval(script, variable) and storeExpression(expression, variableName).

**7) If a Selenium function requires a pattern argument, what five prefixes might that argument have?**

Five prefixes that Selenium pattern argument are:

glob, regexp, exact, regexpi.

**8) Why Selenium RC is used?**

We use Selenium RC for:

Selenium-IDE does not directly support:

1. Condition statements.
2. Iteration.
3. Logging and reporting of test results.
4. Error handling, particularly unexpected errors.
5. Database testing.
6. Test case grouping.
7. Re-execution of failed tests.
8. Test case dependency.
9. Capture screen shots on test failures.

The reason behind why Selenium-IDE does not support the above mentioned requirements is IDE supports only HTML language. Using HTML language we cannot achieve the above mentioned requirements. Because HTML does not support conditional, looping and external source connectives.

To overcome the above mentioned problems Selenium RC is used.

**9) How many testing frameworks can QA Tester use in Selenium RC?**

Testing frameworks aren't required, but they can be helpful if QA Tester wants to automate test cases. Selenium RC supports Bromine, JUnit, NUnit, RSpec (Ruby), Test::Unit (Ruby), TestNG (Java), unittest (Python).

**10) What is the architecture of Selenium RC?**

The Selenium Server launches and kills browsers and acts as an HTTP proxy for browser tests. Client libraries for various programming languages, each of which instructs the Selenium Server in how to test the AUT by passing it your test script's Selenium commands.

The client libraries communicate with the Server passing each Selenium command for execution. Then the server passes the Selenium command to the browser using Selenium-Core JavaScript commands. The browser, using its JavaScript interpreter, executes the Selenium command, which effectively, runs the check you specified in your Selenese test script.

**11) Does Selenium support mobile internet testing?**

Selenium supports Opera and opera is used in most of the Smart phones. So whichever

Smart phone supports opera, selenium can be used to test. So, one can use Selenium RC to run the tests on mobiles.

## **12) What are the basic annotations used to run TestNG tests in Selenium?**

The basic annotations used to run TestNG tests in Selenium RC:

1. `@BeforeClass`: The annotated method with `@BeforeClass` will be run before the first test method in the current class is invoked.
2. `@AfterClass`: The annotated method with `@AfterClass` will be run after all the test methods in the current class have been run.
3. `@BeforeMethod`: The annotated method with `@BeforeMethod` will be run before each test method.
4. `@AfterMethod`: The annotated method with `@AfterMethod` will be run after each test method.
5. `@Test`: Marks a class or a method `@Test` with as part of the test.

## **13) What is the difference between Thread.Sleep() and Selenium.setSpeed()?**

**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.

## **14) How to configure Selenium RC with eclipse to run Junit Tests?**

- 1) Download eclipse. click here to download the software
- 2) Open eclipse -> Workspace Launcher window will open
- 3) Create a workspace by giving meaningful name
- 3) Click on Workbench
- 4) Create a project of type java
- 5) Create a package under src folder of the package
- 6) Add Junit to the build path
- 7) Add selenium rc java client driver to the build path
- 8) Now drag and drop your test script (.which is exported from Selenium IDE) to the package created.

## **15) Actual end user simulation, Is the test conducted using this tool equivalent to an end user action?**

Selenium performs actions in the background on the browser. It modifies the DOM structure of the HTML page in order to perform actions on the page. To be more precise it executes JavaScript on UI objects within the webpage to perform actions like click, type, select etc. This is the reason why you can execute tests with the browser minimized.

QTP claims to perform end user simulation, in other words executing QTP scripts are equivalent to a person performing those steps manually on the application.

## **16) Is it possible to use Selenium for multi-user Load Testing?**

Yes, but it requires a LOT of hardware. We recommend you check out BrowserMob, which does load testing with real browsers and is powered by Selenium.

## **What is Selenium? What is the difference between Selenium WebDriver and Selenium IDE?**

Selenium is an automation tool which is used to test different applications mainly web applications, browsers and platforms. Selenium is a set of tools:

**Selenium Integrated Development Environment:** It's a record and playback tool designed to record the user actions and playback the user actions that are performed.

**Selenium Remote Control:** It is used to write automated web applications UI in any programming languages such as Java.

**Selenium WebDriver:** It is introduced in Selenium 2.0, the robust technology that was used in Selenium webdriver uses browser native commands.

**Selenium Grid:** It is also introduced with Selenium 2.0, it is mainly used to distribute the test in different systems and manage from a single point.

## **What are the browsers and environments selenium supports?**

Selenium Supporting Browsers: Firefox 27, 26, 24, 17; Internet explorer 6, 7, 8, 9, 10, 11; Safari Driver 5.1x, Opera Driver 12.x and chrome driver.

Selenium Supporting Environments: Windows xp, 7, 8, 8.1 and Windows 10 and Linux.

## **What are the programming languages Selenium supports?**

Selenium supports C#, Java, JavaScript, Objective C, Perl, PHP, Python, Ruby.

## **How much you rate yourself on Selenium out of 10?**

Well, don't overrate yourself for this question, give the what you feel and don't give too less as well.

## **Why did you choose Selenium as your tool for automation?**

These days most of the company's preferring Selenium as their functional testing tool as it is an open source software and is widely used and you can do different types of automation testing and it has large online community support and you can develop our scripts with different programming languages such as C#, Java, Ruby, Python etc.

## **What are the differences between Selenium and UFT or QTP?**

Selenium	UFT
Selenium is an open source tool	UFT is a commercial automation tool by HP.
You can test only web applications	You can test different applications and web browsers.
You can develop the scripts only on VB script	You can develop the scripts with Java, Perl, Ruby, PHP etc.
Selenium has good online community support	UFT has technical support by HP

### **What are the different types of testing you can do with Selenium?**

By using selenium you can do functional testing, smoke testing, regression testing, integration testing, Load and performance testing.

### **What is Selenese?**

Selenese is a set of commands that were performed by a selenium tool on the web application.

### **What are the different type of locators or elements available in Selenium?**

ID, Name, Link, CSS, XPATH, DOM and location by element id etc.

### **What is assertion and what are the different types of assertions available in Selenium?**

You can perform different assertions on the application such as text, alert, editable, checked etc. One of the popular assertion and my favorite is assertTrue(message, condition).

### **What are the different web drivers available in selenium?**

FirefoxDriver, ChromeDriver, SafariDriver, OperaDriver, InternetExplorerDriver, iPhoneDriver, IphoneSimulatorDriver, RemoteWebDriver.

These are some of the frequently asked questions in selenium.

1

Selenium WebDriver Interview Questions & Answers

#### **1) What is Selenium?**

Ans: Selenium is a suite of software tools to automate web browsers across many platforms (Different Operation Systems like MS Windows, Linux Macintosh etc.). It was launched in

2004, and it is open source Test Tool suite.

## **2) What is Selenium 2.0?**

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

2.0

Selenium 1.0 + WebDriver = Selenium 2.0

## **3) What is Selenium WebDriver?**

Ans: Selenium WebDriver is a tool for writing automated tests of websites. It is an API name and aims to mimic the behavior of a real user, and as such interacts with the HTML of the application. Selenium WebDriver is the successor of Selenium Remote Control which has been

officially deprecated.

## **What is cost of WebDriver, is this commercial or open source?**

Selenium is an open source and free of cost.

## **How you specify browser configurations with Selenium 2.0?**

Following driver classes are used for browser configuration

AndroidDriver,

ChromeDriver,

EventFiringWebDriver,

FirefoxDriver,

HtmlUnitDriver,

InternetExplorerDriver,

2

iPhoneDriver,

iPhoneSimulatorDriver,

RemoteWebDriver

## **How is Selenium 2.0 configuration different than Selenium 1.0?**

Ans: In case of Selenium 1.0 you need Selenium jar file pertaining to one library for example in

case of java you need java client driver and also Selenium server jar file. While with Selenium

2.0 you need language binding (i.e. java, C# etc) and Selenium server jar if you are using Remote Control or Remote WebDriver.

## **Can you show me one code example of setting Selenium 2.0?**

Ans: Below is example

```
WebDriver driver = new FirefoxDriver();
driver.get("https://www.google.co.in/");
driver.findElement(By.cssSelector("#gb_2 > span.gbts")).click();
driver.findElement(By.cssSelector("#gb_1 > span.gbts")).click();
driver.findElement(By.cssSelector("#gb_8 > span.gbts")).click();
driver.findElement(By.cssSelector("#gb_1 > span.gbts")).click();
```

## **Which web driver implementation is fastest?**

Ans: HTMLUnitDriver. Simple reason is HTMLUnitDriver does not execute tests on browser but plain http ret – 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

## **What all different element locators are available with Selenium 2.0?**

Ans: Selenium 2.0 uses same set of locators which are used by Selenium 1.0 – id, name, css, XPath but how Selenium 2.0 accesses them is different. In case of Selenium 1.0 you don't have

to specify a different method for each locator while in case of Selenium 2.0 there is a different

method available to use a different element locator. Selenium 2.0 uses following method to access elements with id, name, css and XPath locator –

### **How do I submit a form using Selenium?**

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

### **How to capture screen shot in Webdriver?**

```
File file= ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
FileUtils.copyFile(file, new File("c:\\name.png"));
```

### **How do I clear content of a text box in Selenium 2.0?**

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

### **How to execute java scripts function?**

```
JavascriptExecutor js = (JavascriptExecutor) driver;
String title = (String) js.executeScript("pass your java scripts");
```

### **How to select a drop down value using Selenium2.0?**

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();
}
}

```

### **How to automate radio button in Selenium 2.0?**

Ans:

```

WebElement el = driver.findElement(By.id("Radio button id"));
//to perform check operation
el.click()
//verifiy to radio button is check it return true if selected else false
el.isSelected()

```

### **How to capture element image using Selenium 2.0?**

click link for

### **17: How to count total number of rows of a table using Selenium 2.0?**

```

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

```

### **18: How to capture page title using Selenium 2.0?**

```

String title = driver.getTitle()

```

### **19: How to store page source using Selenium 2.0?**

```

String pagesource = driver.getPageSource()
20: How to store current url using selenium 2.0?
String currentURL = driver.getCurrentUrl()
21: How to assert text 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);
22: How to get element attribute using Selenium 2.0?
WebElement el = driver.findElement(By.id("ElementID"));
//get test from element and stored in text variable
String attributeValue = el.getAttribute("AttributeName");
23: How to double click on element using selenium 2.0?
WebElement el = driver.findElement(By.id("ElementID"));
Actions builder = new Actions(driver);
builder.doubleClick(el).build().perform();
24: How to perform drag and drop in selenium 2.0?
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();
25: How to maximize window using selenium 2.0?
driver.manage().window().maximize();
26: How to verify PDF content using selenium 2.0?
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());
PDFParserPDFParser = new PDFParser(fileToParse);
PDFParser.parse();
//savePDF text into strong variable
StringPDFtxt = new PDFTextStripper().getText(pdfParser.getPDDocument());

//closePDFParser object
PDFParser.getPDDocument().close();
After applying above code, you can store allPDF file content into "pdftxt" string variable.
Now
you can verify string by giving input. As if you want to verify "Selenium or WebDiver" text.
Use
below code.

Assert.assertTrue(pdftxt.contains("Selenium or WebDiver"))

```

## **27: How to verify response 200 code using selenium 2.0?**

Ans: I will explain you to verify HTTP response code 200 of web application using java webdriver. As webdriver does not support direct any function to verify page response code.

But

using "WebClient" of HtmlUnit API we can achieve this.

Html unit API is GUI less browser for java developer, using WebClient class we can send request to application server and verify response header status.

Below code, I used in my webdriver script to verify response 200 of web application

```
String url = "http://www.google.com/";  
WebClient webClient = new WebClient();  
HtmlPage htmlPage = webClient.getPage(url);  
//verify response  
Assert.assertEquals(200,htmlPage.getWebResponse().getStatusCode());  
Assert.assertEquals("OK",htmlPage.getWebResponse().getStatusMessage());  
If HTTP authentication is required in web application use below code.  
String url = "Application Url";
```

```
WebClient webClient = new WebClient();  
DefaultCredentialsProvider credential = new DefaultCredentialsProvider();
```

```
//Set some example credentials  
credential.addCredentials("UserName", "Password");  
webClient.setCredentialsProvider(credential);
```

```
HtmlPage htmlPage = webClient.getPage(url);  
//verify response  
Assert.assertEquals(200,htmlPage.getWebResponse().getStatusCode());  
Assert.assertEquals("OK",htmlPage.getWebResponse().getStatusMessage());
```

## **28: How to verify image using selenium 2.0?**

I have explained that how to verify images in webdriver using java. As webdriver does not provide direct any function to image verification, but we can verify images by taking two screen

shots of whole web page using "TakesScreenshot" webdriver function, one at script creation time

and another at execution time,

In below example I have created a sample script in which first I captured a Google home page

screen shot and saved (GoogleInput.jpg) into my project, Another screen shot "GoogleOutput.jpg" captured of same page at test executing time and saved into project. I compared both images if they are not same then test will fail.

Here is sample code for same

```
package com.test;  
import java.awt.image.BufferedImage;  
import java.awt.image.DataBuffer;  
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.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.Assert;
import org.testng.annotations.AfterSuite;
import org.testng.annotations.BeforeSuite;
import org.testng.annotations.Test;
public class ImageComparison {
    public WebDriver driver;
    private String baseUrl;

    @BeforeSuite
    public void setUp() throws Exception {
        driver = new FirefoxDriver();
        baseUrl = "https://www.google.co.in/";
        driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
    }

    @AfterSuite
    public void tearDown() throws Exception {
        driver.quit();
    }

    @Test
    public void testImageComparison()
            throws IOException, InterruptedException
    {
        driver.navigate().to(baseUrl);
        File screenshot = ((TakesScreenshot)driver).
                getScreenshotAs(OutputType.FILE);
        Thread.sleep(3000);
        FileUtils.copyFile(screenshot, new File("GoogleOutput.jpg"));
        File fileInput = new File("GoogleInput.jpg");
        File fileOutPut = new File("GoogleOutput.jpg");
        BufferedImage bufFileInput = ImageIO.read(fileInput);
        DataBuffer dafileInput = bufFileInput.getData().getDataBuffer();
        int sizefileInput = dafileInput.getSize();
        BufferedImage bufFileOutPut = ImageIO.read(fileOutPut);
        DataBuffer dafileOutPut = bufFileOutPut.getData().getDataBuffer();
        int sizefileOutPut = dafileOutPut.getSize();
        Boolean matchFlag = true;
        if(sizefileInput == sizefileOutPut) {
            for(int j=0; j<sizefileInput; j++) {
                if(dafileInput.getElem(j) != dafileOutPut.getElem(j)) {

```

```
matchFlag = false;
break;
}
}
}
else
matchFlag = false;
Assert.assertTrue(matchFlag, "Images are not same");
}
}
```

## 29: How to handle http authentication in selenium 2.0?

Road to handle http authentication in webdriver

One of my applications had Http authentication for security purpose and I had need to automate

using webdriver. As we know http authentication is not a part of DOM object so we cannot handle it using webdriver. Here is approach to handle such type situation.

We need to pass http credential with URL to skip http popup authentication window. Below is

URL format.

`http://username:passwork@applicationURL`

User above formatted URL in webdriver get method:

```
driver.get("http://username:passwork@applicationURL")
```

After using above approach, http authentication popup window disappear. But in Internet explorer, it raise error with message "wrong format url". To accept same type url in internet explorer browser we need to add a DWORD value named exploere.exe and iexplore.exe with

value data 0 in below registry.

To open registry, open "regeidt" from run command.

For all users of the program, set the value in the following registry key:

`HKEY_LOCAL_MACHINE\Software\Microsoft\Internet`

`Explorer\Main\FeatureControl\FEATURE_HTTP_USERNAME_PASSWORD_DISABLE`

For the current user of the program only, set the value in the following registry key:

`HKEY_CURRENT_USER\Software\Microsoft\Internet`

`Explorer\Main\FeatureControl\FEATURE_HTTP_USERNAME_PASSWORD_DISABLE`

If you are using 64 bit machine, set the value in the following registry key.

`HKEY_CURRENT_USER\Software\WOW6432Node\Microsoft\Internet`

`Explorer\Main\FeatureControl\FEATURE_HTTP_USERNAME_PASSWORD_DISABLE`

## 1. What is the Difference between Manual Testing and Automation Testing?

- Tests conducted by testers manually is called Manual Testing, wherein they compare the expected and actual functionality are in accordance with the test cases.

- Tests conducted by using software tool is called Automation Testing, wherein the expected results are fed into the tool to be compared with the actual output of software being tested.
- Refer link for more Q&A on Automation Testing. ([Link to Automation Testing QA](#))

## **2. What are the Popular tools used for automation testing?**

- Selenium
- HP UFT(Formerly QTP)
- Rational Robot
- IBM Rational Functional Tester

## **3. Explain the advantages of Automation Testing**

- Reduced test execution time
- Can be executed in cross platforms test cases (different OS, browsers, environment, etc)
- Simplifies complex functional tests
- Automating repetitive tasks
- Enables parallel execution

## **4. What is Selenium Testing?**

Selenium is a web browser automation tool, supports numerous scripting language and technologies.

## **5. What are the major Selenium components?**

- Selenium IDE
- Selenium RC
- Selenium Webdriver
- Selenium Grid

## **6. What are the testing types supported by Selenium?**

Selenium supports Functional and Regression Testing

## **7. What is Selenium IDE**

Selenium IDE is Integrated Development Environment, which is implemented as Firefox extension. It helps us record, edit and debug the tests.

**8. Explain the advantages of Selenium IDE?**

- Easy to Record and Playback
- Command auto-complete
- Can set breakpoints while debug mode
- Assert, Verify and WaitFor options

**9. Language used in Selenium IDE**

Language used is Selenese

**10. Does Selenium IDE support multiple test case execution?**

Selenium IDE supports suite creation wherein the testcases created under a Suite is executed one by one. Test suite is a simple HTML file listing the tests to be executed in a user-defined format.

**11. Can tests created in Selenium IDE run in different browser?**

Selenium IDE created test cases can be executed as suite under Selenium RC using command line interface which invokes Selenium RC Server.

Ex, `java -jar selenium-server.jar -htmlSuite "*firefox" "http://www.google.com"`  
`"c:\absolute\path\to\my\HTMLSuite.html" "c:\absolute\path\to\my\results.html"`

**12. What is the difference between Assert and Verify in Selenium IDE?**

- When an “assert” command fails, the test is aborted.

- Whereas, when a “verify” command fails, the test will continue execution of next command in test after logging the failure.

#### **13. What is a Xpath?**

Xpath is a XML path language, which describes a way to traverse through XML document to locate nodes. Same is used to locate elements in HTML.

#### **14. What is the difference between '/' and '//' in Xpath?**

'/' used to indicate absolute path of an element, majorly when the user knows exact path of an element. Whereas '//' used to indicate relative path of an element, majorly to locate an element from any part of the HTML file.

#### **15. What is Selenium RC?**

Selenium Remote Control (RC) is the first selenium project before Selenium Webdriver. It's a test tool that allows you to write automated web application user interface tests in any programming language against any HTTP website using any mainstream JavaScript-enabled browser.

#### **16. What are the Selenium RC supported languages/client drivers?**

Supported languages/client drivers are :

- Java
- Ruby
- Python
- Perl
- PHP
- .Net

#### **17. Explain the advantages of Selenium**

- Open Source
- iUser communities for addressing the issues faced

- Cross browser compatibility
- Multiple platform compatibility
- Supports multiple programming/scripting languages

## **18. What are the different type of locators used in Selenium**

Locators are used to identify elements within a web page, types are listed below :

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

## **19. What is Selenium Grid?**

Selenium Grid allows us to run tests on different machines against different browsers in parallel. Here different machines can be of different OS and supports distributed test execution.

## **20. What is the Difference between Selenium1 and Selenium2?**

Selenium1 is Selenium RC and Selenium2 is Selenium Webdriver, major difference is that the Selenium RC communicates to browser via Server, whereas the Webdriver makes direct calls to browser native support for automation.

## **21. What are the Browser launch commands used in Webdriver**

Following are some major browser related launch commands :

```
WebDriver driver = new FirefoxDriver(); //Mozilla
```

```
WebDriver driver = new ChromeDriver(); //Chrome
```

```
WebDriver driver = new InternetExplorerDriver(); //Internet Explorer
```

```
WebDriver driver = new RemoteWebDriver(); // Grid  
WebDriver driver = new HtmlUnitDriver(); //Headless browser
```

## 22. List down the Webdriver supported drivers

### SeleniumHQ drivers :

- FirefoxDriver
- InternetExplorerDriver
- SafariDriver
- HtmlUnitDriver

### Third party drivers :

- ChromeDriver
- OperaDriver
- AndroidDriver
- iPhoneDriver
- Etc

## 23. Explain in detail different 'Wait' operations in Selenium Webdriver

There are two wait types available in Selenium Webdriver - Implicit Wait and Explicit Wait

### Implicit Wait

An implicit wait is to tell Webdriver to provide a default waiting time when trying to find an element or elements if they are not immediately available. The default wait time is 0. For Example,

```
WebDriver driver = new FirefoxDriver();  
  
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);  
  
driver.get("http://somedomain/url_that_delays_loading");  
  
WebElement myDynamicElement = driver.findElement(By.id("myDynamicElement"));
```

### Explicit Wait

Used to halt the execution till the time a particular condition is met. Unlike Implicit waits, Explicit waits are applied for a particular instance only. For example,

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

WebElement element =
wait.until(ExpectedConditions.elementToBeClickable(By.id("someid")));
```

#### **24. What are the differences between findElement() and findElements() ?**

- **findElement()** : It finds the first element within the current page using the given locating mechanism. It returns a single WebElement.
- **findElements()** : Using the given locating mechanism, find all the elements within the current page. It returns a list of web elements.

#### **25. What is the Command to enter string into textbox using Selenium Webdriver**

```
webElement.sendKeys("String");
```

##### **Example**

```
WebDriver driver = new FirefoxDriver();

driver.get("www.google.com");

WebElement searchName=driver.findElement(By.id("gs_htif0"));

searchName.sendKeys("FirstName");
```

#### **26. How to switch between frames in Webdriver?**

Select frame by Index :

```
driver.switchTo().frame(int arg0);
```

Select frame by Name or ID :

```
driver.switchTo().frame(String arg0);
```

Select frame by WebElement :

```
driver.switchTo().frame(WebElement frameElement);
```

## **27. How handle pop up in web application using Selenium Webdriver?**

```
Alert alert = driver.switchTo().alert();
```

To cancel the message :

```
alert.dismiss();
```

To accept the message :

```
alert.accept();
```

To get string from alert message :

```
String msg = alert.getText();
```

To send string to alert box :

```
alert.sendKeys("Hi");
```

## **28. Can Selenium Webdriver handle windows pop up?**

No, but a simple program called AutoIT can generate .exe file to be called in webdriver script.

## **29. Write AutoIT script to upload a file**

Wait 10 seconds for the Upload window to appear

```
WinWait("[CLASS:#32770]","",10)
```

Set input focus to the edit control of Upload window using the handle returned by WinWait

```
ControlFocus("File Upload","","Edit1")
```

Set the File name text on the Edit field

```
ControlSetText("File Upload", "", "Edit1", "Complete file path")
```

Click on the Open button

```
ControlClick("File Upload", "", "Button1");
```

### **30. How to call AutoIT script from Selenium Webdriver?**

```
Runtime.getRuntime().exec("complete AutoIt .exe filepath");
```

### **31. Write down Selenium script to take screenshot**

There are multiple ways to achieve the same, one of them is listed below :

```
WebDriver driver = new FirefoxDriver();
driver.get("http://www.google.com/");
File scrFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
// Now you can do whatever you need to do with it, for example copy somewhere
FileUtils.copyFile(scrFile, new File("c:\\tmp\\screenshot.png"));
```

### **32. Name the fastest web driver**

```
WebDriver driver = new HtmlUnitDriver(); //Headless browser
```

### **33. What are the Different frameworks available for JAVA Webdriver**

- Junit
- TestNG

### **34. Best Automation Framework type preferred for Webdriver**

Hybrid Framework – Its combination of both Data and Keyword Driven Framework

### **35. List down the basic browser commands**

Get Webpage :

```
driver.get("www.google.com");
```

Get Title :

```
driver.getTitle();
```

Get Current URL :

```
driver.getCurrentUrl();
```

Get page source :

```
driver.getPageSource();
```

Close current window :

```
driver.close();
```

Quit the browser :

```
driver.quit();
```

### **36. What is TestNG?**

TestNG is similar to Junit and an open source automated testing framework

### **37. What are the advantages of TestNG?**

- Ability to produce HTML reports
- iAnnotations
- Grouping and prioritizing made easy
- Parallel execution of test

- Parametrization

### **38. List annotations in TestNG**

@BeforeSuite: The method will be run before all tests in this suite.

@AfterSuite: The method will be run after all tests in suite.

@BeforeTest: The method will be run before any test method belonging to the classes.

@AfterTest: The method will be run after all the test methods belonging to the classes.

@BeforeGroups: The list of groups that this configuration method will run before.

@AfterGroups: The list of groups that this configuration method will run after.

@BeforeClass: The method will be run before the first test method in the current class is invoked.

@AfterClass: The method will be run after all the test methods in the current class have been run.

@BeforeMethod: The method will be run before each test method.

@AfterMethod: The method will be run after each test method.

@Test: The method is a part of a test case.

### **39. What are the exceptions faced in Webdriver**

Some of the exceptions faced are :

- ElementNotFoundException,
- ElementNotSelectableException,
- NoAlertPresentException,
- NoSuchElementException,
- NoSuchWindowException,
- TimeoutException,
- WebDriverException etc

### **40. Explain the ways to handle radio button**

For radio button click :

```
WebElement radioButton = driver.findElement(By.id("radioID"));
```

```
radioButton.click();
```

To check if the radio button is selected :

```
List radioButton = driver.findElements(By.name("radioName"));
boolean rValue = radioButton.get(0).isSelected();
```

NOTE : rValue will be set to true if selected, else false.

#### 41. Explain the ways to handle Dropdown list

**Selection :**

```
dropDown.selectByVisibleText
dropDown.selectByIndex
```

**Deselect :**

```
dropDown.deselectAll();
dropDown.deselectByIndex(index);
dropDown.deselectByValue(value);
dropDown.deselectByVisibleText(text);
```

#### 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 plug-in
- **Web Driver and RC**: It provide the APIs for a variety of languages like Java, .NET, PHP, etc. With most of the browsers Web driver 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

#### What are the Components of Selenium ?

- Selenium IDE
- Selenium Remote Control
- Selenium Grid

#### Why and When To Automate?

- Frequent regression testing
- Repeated test case Execution is required
- User Acceptance Tests
- Faster Feedback to the developers
- Reduce the Human Effort
- Test same application on multiple environments

### **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
- Link text
- PartialLink Text
- Xpath etc

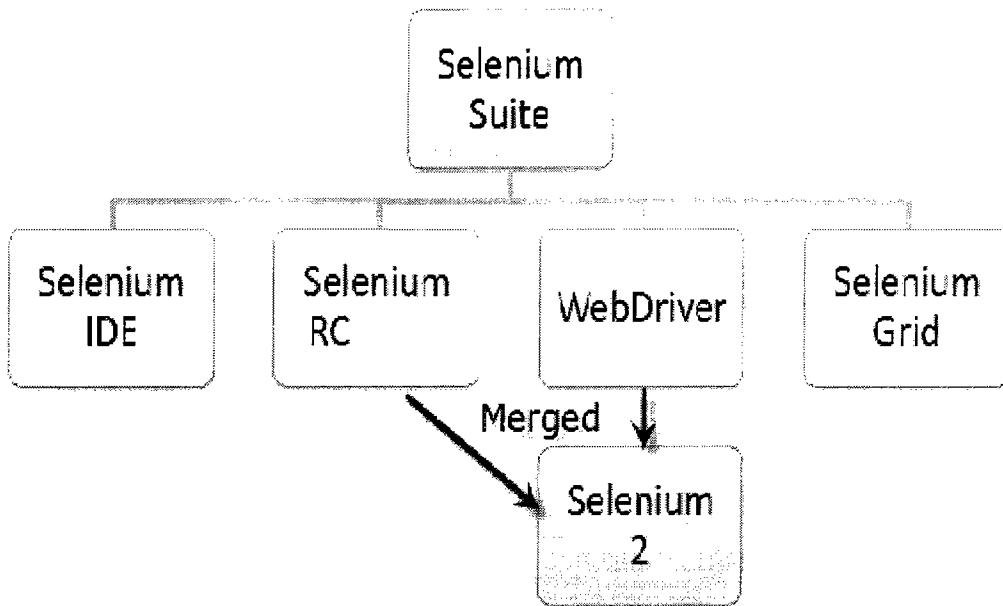
### **What are the Features of Selenium IDE ?**

- Record and playback
- Intelligent field selection will use IDs, names, or XPath as needed
- Auto complete for all common Selenium commands
- Walk through test cases and test suites.
- Debug and set breakpoints
- Save tests as HTML, Ruby scripts, or other formats
- Support for Selenium user-extensions.js file
- Option to automatically assert the title of every page

- Rollup common commands

### **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 “waif For”.



### **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.

### **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

### **What are the Selenese Commands in Selenium ?**

- clicking a link - click or clickAndWait commands
- entering values - type command

- selecting options from a drop-down listbox - select command
- clicking checkboxes or radio buttons - click command

### **What is Test Runner ?**

Test Runner allows you to run the test case in a browser loaded with the Selenium-Core Test Runner. Test runner is invoked by clicking the below Shown button in the IDE.

### **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.

### **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 know as Annotations. Variables, parameters, packages, methods and classes are annotated some of the JUnit annotations which can be useful are

- Test
- Before
- After
- Ignore
- Before Class
- After Class
- Run With

### **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 know as Annotations. Variables, parameters, packages, methods and classes are annotated some of the JUnit annotations which can be useful are

- Test
- Before
- After

- Ignore
- Before Class
- After Class
- Run With

### **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

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

Both will delay the speed of execution.

**Thread. Sleep () :** It will stop the current (java) thread for the specified period of time. It's done only once

- It takes a single argument in integer format  
Ex: `thread.sleep(2000)`- It will wait for 2 seconds
- It waits only once at the command given at sleep

**SetSpeed () :** For specific amount of time it will stop the execution for every selenium command.

### **Explain Selenium WaitFor Commands ?**

- **waitForPageToLoad :** This command will make the script to wait till the page loads. Syntax is `waitForPageToLoad(timeout);` Time out is the maximum time the script will wait for the page to load.
- **waitForAlert :** This command will wait for the alert message to appear
- **waitForTable:** This command will wait for the Web table to completely load in the page
- **waitForTitle:** This command will for the page Title to appear on the browser.
- **Other waitFor commands :** Selenium has several other wait command like `waitForText,waitForPopup` and so on. These commands are generically called Synchronization commands

### **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

**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

**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

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

**Datadriven 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.

**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 ned to switch to each and every frame and search for locator which we want.  
Then break the loop

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

**Implicit Wait:** Sets a timeout for all successive Web Element searches. For the specified amount of time it will try looking for element again and again before throwing a NoSuchElementException. It waits for elements to show up.

**Explicit Wait :** It is a one-timer, used for a particular search.

### **What tests can selenium do?**

Selenium could do functional, regression, and load of web based applications.

### **What is the selenium's recording language?**

Selenium's recording language is "HTML".

### **What is WebDriver?**

WebDriver is a tool for writing automated tests of websites. It aims to mimic the behaviour of a real user, and as such interacts with the HTML of the application.

### **Can Selenium test a application on iPhone's Mobile Safari browser?**

Selenium can handle Mobile Safari browser. There is experimental Selenium iPhone Driver for running tests on Mobile with Safari on the iPhone and iPad and iPod Touch.

#### **1. How do you start Selenium RC?**

simple way to start selenium rc is

java -jar selenium-server.jar

to run a suite of Selenese scripts in a browser

java -jar selenium-server.jar -htmlSuite

#### **2. How do you connect Data base from selenium**

Connecting to database is lanugage dependent. if we are using Java

A Connection object represents aconnection with a database. When we connect to a database by using connection method, we create a Connection Object, which represents theconnection to the database. An application may have one or more than one connections with a single database or many connections with different databases.

We can use the Connection object for the following things:

1). It creates the Statement, PreparedStatement and CallableStatement objects for executing the SQL statements.

2). It helps us to Commit or roll back a jdbc transactionn.

3). If you want to know about the database or data source to which you are connected then the Connection object gathers information about the database or data source by the use of DatabaseMetaData.

4). It helps us to close the data source. The Connection.isClosed() method returns true only if the Connection.close() has been called. This method is used to close all the connection.

Firstly we need to establish the connection with the database. This is done by using the method DriverManager.getConnection(). This method takes a string containing a URL. The DriverManager class, attempts to locate a driver that can connect to the database represented by the string URL. Whenever the getConnection() method is called the DriverManager class checks the list of all registered Driver classes that can connect to the database specified in the URL.

Syntax:

```
String url = "jdbc:odbc:makeConnection";
Connection con = DriverManager.getConnection(url, "userID", "password");
```

**3. How do you handle Ajax controls using selenium?**

Eg. By typing in search engine how do you capture the auto suggestion

**4. How do you select the 2nd item in a List box or drop down.**

```
selenium.select(selectLocator, optionLocator)
selectLocator – a locator for the select element
optionLocator – a locator for the option element
```

**5. How do you identify an object using selenium?**

isElementPresent(String locator)

isElementPresent takes a locator as the argument and if found returns a boolean value of True.

**6. How do you capture an element loading after the page load?**

provide a time to check the element( in seconds) Like :

```
public void waitForElementPresent(String element, int timeout) throws Exception {
for (int second = 0;; second++) {
if (second >= timeout)
fail("Timeout. Unable to find the Specified element" + element);
try {
if (selenium.isElementPresent(element)) break;
} catch (Exception e) {}
Thread.sleep(1000);
}
}
```

**7. Brief about your framework**

**8. What is the difference between assert and Verify Commands?**

There are two mechanisms for validating elements that are available on the application under test. The first is assert: this allows the test to check if the element is on the page. If it is not available, then the test will stop on the step that failed. The second is verify: this also allows the test to check whether the element is on the page, but if it isn't, then the test will carry on executing.

**9. Explain about your reporting method**

**10. How do you verify an object presented in multiple pages.**

**Ans:**

Check on each page

```
assertTrue(selenium.isElementPresent(locator));
```

**12. What is the difference between single and double slash in Xpath**

**Ans:**

/

1. start selection from the document node
2. allows you to create 'absolute' path expressions
3. e.g. "/html/body/p" matches all the paragraph elements

//

1. start selection matching anywhere in the document
2. allows you to create 'relative' path expressions
3. e.g. "//p" matches all the paragraph elements

### **13.Brief about Selenium Client.**

### **14.Did you write any User Extensions.**

User extensions are stored in a separate file that we will tell Selenium IDE or Selenium RC to use. Inside there the new function will be written in JavaScript.

Because Selenium's core is developed in JavaScript, creating an extension follows the standard rules for prototypal languages. To create an extension, we create a function in the following design pattern.

```
Selenium.prototype.doFunctionName = function(){  
    //  
}
```

The "do" in front of the function name tells Selenium that this function can be called as a command for a step instead of an internal or private function.

### **15.What are all things can not be done through selenium IDE**

1. Easy record and playback.
2. Intelligent field selection will use IDs, names, or XPath as needed.
3. Autocomplete for all common Selenium commands.
4. Walk through tests.
5. Debug and set breakpoints.
6. Save tests as HTML, Ruby scripts, or any other format.
7. Support for Selenium user-extensions.js file.
8. Option to automatically assert the title of every page.

**Advantages:** Selenium IDE is the only flavor of Selenium, which allows you to record user action on browser window. It can also record user actions in most of the popular languages like Java, C#, Perl, Ruby etc. This eliminates the need of learning new vendor scripting language. For executing scripts created in these languages, you will need to use Selenium Remote Control. If you do not want to use Remote Control than you will need to create your test scripts in HTML format.

**Disadvantages:** Biggest drawback of Selenium IDE is its limitation in terms of browser support. Though Selenium scripts can be used for most of the browser and operating system, Scripts written using Selenium IDE can be used for only Firefox browser if it is not used with Selenium RC or Selenium Core.

### **16.Brief about Selenium Grid.**

Selenium Grid is a tool that dramatically speeds up functional testing of web-apps by leveraging your existing computing infrastructure. It allows you to easily run multiple tests in parallel, on multiple machines, in an heterogeneous environment.

Based on the excellent Selenium web testing tool, Selenium Grid allows you to run multiple instances of Selenium Remote Control in parallel. Even better, it makes all these Selenium Remote Controls appear as a single one, so your tests do not have to worry about the actual infrastructure. Selenium Grid cuts down on the time required to run a Selenium test suite to a fraction of the time that a single instance of Selenium instance would take to run.

**17. Is it possible to start the Remote engine more than once in an instance?**

**18. How to use selenium for performance testing**

**19. How to start the selenium server from your language class?**

```
try {  
    seleniumServer = new SeleniumServer();  
    seleniumServer.start();  
} catch (Exception e) {  
    e.printStackTrace();  
}
```

**20. Is it possible to handle multiple pop ups in selenium?**

**22. Give the benefits of SQL Stored procedure in selenium?**

**23. What are the difficulties or challenge you faced in selenium?**

**24. How do you check a single test method in multiple browser?**

**25. What is the annotation you use to connect the Spread sheet in Junit.**

**26. Brief about Junit annotations.**

**27. How can we speed up the selenium script execution?**

**28. If the default port of selenium is busy then which port you use?**

**29. Explain types of SQL joins.**

**30. How do you handle the secured connection error in HTTPS?**

**31. How do you compare two strings or values are same.**

### **What is Selenium?**

Selenium is a suite of tools for browser automation. It is composed of "IDE", a recording and playback mechanism, "WebDriver" and "RC" which provide APIs for browser automation in a wide variety of languages, and "Grid", which allows many tests using the APIs to be run in parallel. It works with most browsers, including Firefox 5, Internet Explorer 8, Google Chrome, Safari and Opera 9

### **Describe technical problems that you had with Selenium tool?**

As with any other type of test automation tools like SilkTest, HP QTP, Watir, Canoo Webtest, Selenium allows to record, edit, and debug tests cases. However there are several problems that seriously affect maintainability of recorded test cases, occasionally Quality Assurance Engineers complain that it takes more time to maintain automated test cases than to perform manual testing; however this is an issue with all automated testing tools and most likely related to improper testing framework design. Another problem is complex ID for an HTML element. If IDs are auto-generated, the recorder test cases may fail during playback. The work around is to use XPath to find required HTML element. Selenium supports AJAX without problems, but QA Tester should be aware that Selenium does not know when AJAX action is completed, so ClickAndWait will not work. Instead QA tester could use

pause, but the snowballing effect of several 'pause' commands would really slow down total testing time of test cases. The best solution would be to use `waitForElement`.

### **What test can Selenium do?**

Selenium could be used for the functional, regression, load testing of the web based applications. The automation tool could be implemented for post release validation with continuous integration tools like Jenkins, Hudson, QuickBuild or CruiseControl.

### **What is the price of Selenium license per server?**

Selenium is open source software, released under the Apache 2.0 license and can be downloaded and used without charge.

### **How much does Selenium license cost per client machine?**

Selenium is open source software, released under the Apache 2.0 license and can be downloaded and used without charge.

### **Where to download Selenium?**

Selenium can be downloaded and installed for free from [seleniumhq.org](http://seleniumhq.org)

### **What is the latest version of Selenium?**

The latest version of Selenium Core, Selenium IDE, Selenium RC and Selenium Grid can be found on Selenium download page

### **What is Selenium IDE?**

Selenium IDE is a Firefox add-on that records clicks, typing, and other actions to make a test cases, which QA Tester can play back in the Firefox browser or export to Selenium RC. Selenium IDE has the following features: record/play feature, debugging with step-by-step and breakpoints, page abstraction functionality, an extensibility capability allowing the use of add-ons or user extensions that expand the functionality of Selenium IDE

### **What are the limitations of Selenium IDE?**

Selenium IDE has many great features and is a fruitful and well-organized test automation tool for developing test cases, in the same time Selenium IDE is missing certain vital features of a testing tool: conditional statements, loops, logging functionality, exception handling, reporting functionality, database testing, re-execution of failed tests and screenshots taking capability. Selenium IDE doesn't support IE, Safari and Opera browsers.

### **What does SIDE stand for?**

Selenium IDE. It was a very tricky interview question.

### **What is Selenium Remote Control (RC) tool?**

Selenium Remote Control (RC) is the powerful solution for test cases that need more than simple browser actions and linear execution. Selenium-RC allows the developing of complex test scenarios like reading and writing files, querying a database, and emailing test reports. These tasks can be achieved by tweaking test cases in your preferred programming language.

### **What are the advantages using Selenium as testing tool?**

If QA Tester would compare Selenium with HP QTP or Micro Focus SilkTest, QA Engineer would easily notice tremendous cost savings for Selenium. In contrast to expensive SilkTest license or QTP license, Selenium automation tool is absolutely free. It means that with almost no investment in purchasing tools, QA Team could easily build the state of the art test automation infrastructure. Selenium allows developing and executing test cases in various programming languages including .NET, Java, Perl, RubyPython, PHP and even HTML. This is a great Selenium advantage, most likely your software developers already know how to develop and maintain C# or Java code, so they transfer coding techni and best practices to QA team. Selenium allows simple and powerful DOM-level testing and in the same time could be used for testing in the traditional waterfall or modern Agile environments. Selenium would be definitely a great fit for the continuous integration tools Jenkins, Hudson, CruiseControl, because it could be installed on the server testing box, and controlled remotely from continuous integration build.

### **What is Selenium Grid?**

Selenium Grid extends Selenium RC to distribute your tests across multiple servers, saving you time by running tests in parallel.

### **How many browsers are supported by Selenium IDE?**

Test Engineer can record and playback test with Selenium IDE in Firefox.

### **Can Selenium test an application on iPhone's Mobile Safari browser?**

Selenium should be able to handle Mobile Safari browser. There is experimental Selenium iPhone Driver for running tests on Mobile Safari on the iPhone, iPad and iPod Touch.

### **Can Selenium test an application on Android browser?**

Selenium should be able to handle Android browser. There is experimental Selenium Android Driver for running tests in Android browser.

### **What are the disadvantages of using Selenium as testing tool?**

Selenium weak points are tricky setup; dreary errors diagnosis; tests only web applications

### **How many browsers are supported by Selenium Remote Control?**

QA Engineer can use Firefox 5, IE 8, Safari 3 and Opera 9 browsers to run actual tests in Selenium RC.

### **How many programming languages can you use in Selenium RC?**

Several programming languages are supported by Selenium Remote Control - C#  
Java Perl PHP Python Ruby

### **How many testing framework can QA Tester use in Selenium RC?**

Testing frameworks aren't required, but they can be helpful if QA Tester wants to automate test cases. Selenium RC supports Bromine, JUnit, NUnit, RSpec (Ruby), Test::Unit (Ruby), TestNG (Java), unittest (Python).

### **How to developer Selenium Test Cases?**

Using the Selenium IDE, QA Tester can record a test to comprehend the syntax of Selenium IDE commands, or to check the basic syntax for a specific type of user interface. Keep in mind that Selenium IDE recorder is not clever as QA Testers want it to be. Quality assurance team should never consider Selenium IDE as a "record, save, and run it" tool, all the time anticipate reworking a recorded test cases to make them maintainable in the future.

### **What programming language is best for writing Selenium tests?**

The web applications may be written in Java, Ruby, PHP, Python or any other web framework. There are certain advantages for using the same language for writing test cases as application under test. For example, if the team already have the experience with Java, QA Tester could always get the piece of advice while mastering Selenium test cases in Java. Sometimes it is better to choose simpler programming language that will ultimately deliver better success. In this case QA testers can adopt easier programming languages, for example Ruby, much faster comparing with Java, and can become experts as soon as possible.

### **Have you read any good books on Selenium?**

There are several great books covering Selenium automation tool, you could check the review at Best Selenium Books: Top Recommended page

### **Do you know any alternative test automation tools for Selenium?**

Selenium appears to be the mainstream open source tool for browser side testing, but there are many alternatives. Canoo Webtest is a great Selenium alternative and it is probably the fastest automation tool. Another Selenium alternative is Watir, but in order to use Watir QA Tester has to learn Ruby. One more alternative to Selenium is Sahi, but it has confusing interface and small developers community.

### **Compare HP QTP vs Selenium?**

When QA team considers acquiring test automation to assist in testing, one of the most critical decisions is what technologies or tools to use to automate the testing. The most obvious approach will be to look to the software market and evaluate a few test automation tools. Read Selenium vs QTP comparison

### **How to test Ajax application with Selenium**

Ajax interview tions could be tough for newbie in the test automation, but will be easily cracked by Selenium Tester with a relevant experience. Read the detailed approach at Testing Ajax applications with Selenium in the right way

### **How can I learn to automate testing using Selenium?**

Don't be surprised if the interviewer asks you to describe the approach for learning Selenium. This interviewer wants to hear how you can innovative software test automation process the company. Most likely they are looking for software professional with a good Selenium experience, who can do Selenium training for team members and get the team started with test automation. I hope this Selenium tutorial will be helpful in the preparation for this Selenium interview tion.

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

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

#### **2). Difference between Absolute path & Relative path?**

Absolute path will start with root path (/) and Relative path will from current path (//)

#### **3). How does u handle dynamic elements without using xpath?**

By using classname or css.

#### **4). Which repository you have used to store the test scripts?**

I have created scripts in excel file and store them in Test cases folder under src .

#### **5). Detail about TestNG Test Output folder.**

It is the directory where reports are generated. Every time tests run in a suite, TestNG creates index.html and other files in the output directory.

#### **6). Can we run group of test cases using TestNG?**

Test cases in group in Selenium using TestNG will be executed with the below options.

If you want to execute the test cases based on one of the group like regression test or smoke test

```
@Test(groups = {"regressiontest", "smoketest"})
```

**7) What is Selenium? What are the different Selenium components?**

.Selenium Integrated Development Environment (IDE) – 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 – 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 – Selenium Grid is used to distribute your test execution on multiple platforms and environments concurrently.

**8). Difference between the selenium1.0 and selenium 2.0?**

Selenium 1 = Selenium Remote Control.

Selenium 2 = Selenium Web driver, which combines elements of Selenium 1 and Web driver.

**9). Difference between find element () and findelements ()?**

findElement() :

Find the first element within the current page using the given “locating mechanism”.

Returns a single WebElement.

findElements() :

Find all elements within the current page using the given “locating mechanism”.

Returns List of Web Elements.

**10) What are the testing types that can be supported by Selenium?**

Selenium supports the following types of testing:

Functional Testing

Regression Testing

**11) 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.

**12). What is the default time for selenium Ide and webdriver?**

Default timeout in selenium ide is 30 seconds.

**13). Write down scenarios which we can't automate?**

Barcode Reader, Captcha etc.

**14) 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.

**15). In TestNG I have some test's Test1-Test2- Test3-Test4-Test5I want to run my execution order is Test5-Test1-Test3-Test2-Test4.How do you set the execution order can you explain for that?**

Use priority parameter in @test annotation or TestNG annotations.

**16). What is default port no?**

4444

**17). Does Selenium support https protocols?**

Yes

**18) 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

**19) Which is the latest Selenium tool?**

WebDriver

**20. Majorly asked test scenario with framework in Interviews?**

Majorly asked are:

- Login for Gmail scenario
- Goggle search and finding no of results
- Downloading a file and save it
- Checking mails and deleting them
- Do shopping in flipkart.com

**21). Selenium support mobile applications?**

No, it is browser automation tool, it only automates Websites opening in mobile browser, and mobile APPs

can't be automated.

**22). Name 5 different exceptions you had in selenium web driver and mention what instance you got it and how do you resolve it?**

- WebDriverException
- NoAlertPresentException
- NoSuchElementException
- NoSuchElementException
- TimeoutException

**23). How do you manage the code versions in your project?**

- Using SVN or other versioning tools

**24). How do you create html test report from your test script?**

- I would see below 3 ways:
- Junit: with the help of ANT.
- TestNG: using inbuilt default.html to get the HTML report. Also XLST reports from ANT, Selenium, TestNG combination.
- Using our own customized reports using XSL jar for converting XML content to HTML.

**25) When do we use findElement() and findElements()?**

findElement(): findElement() is used to find the first element in the current web page matching to the specified locator value. Take a note that only first matching element would be fetched.

findElements(): findElements() is used to find all the elements in the current web page matching to the specified locator value. Take a note that all the matching elements would be fetched and stored in the list of WebElements.

**26). What is the difference between single and double slash inXpath?**

/

1. It starts selection from the document node
2. It Allows you to create 'absolute' path expressions
3. e.g "/html/body/p" matches all the paragraph elements

//

1. It starts selection matching anywhere in the document
2. It Allows you to create 'relative' path expressions
3. e.g "//p" matches all the paragraph elements

**27). What are the test types supported by Selenium?**

Selenium supports UI and functional testing. As well it can support performance testing for reasonable load using selenium grid.

**28).Tell me some TestNG Annotations.**

@Test,@Parameters,@Listeners,@BeforeSuite,@AfterSuite,@BeforeTest,@AfterTest,@DataProvider,@BeforeGroups,@AfterGroups,@BeforeClass,@AfterClass,@BeforeMethod,@AfterMethod,@Factory

**29). Which is the best way to locate an element?**

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

**30). What are the features of TestNG?**

TestNG is a testing framework designed to simplify a broad range of testing needs, from unit testing (testing a class in isolation of the others) to integration testing (testing entire systems made of several classes, several packages and even several external frameworks, such as application servers). You can use test suite, annotations, automatically generation of report and much more.

**31). In what situation selenium finding element get fails?**

- Element loading issue
- Dynamic id of web element

**32). Please tell me the difference b/w implicitly Wait and Explicit wait.**

Implicit Wait sets internally a timeout that will be used for all consecutive Web Element searches. It will try lookup the element again and again for the specified amount of time before throwing a NoSuchElementException if the element could not have been found. It does only this and can't be forced into anything else – it waits for elements to show up.

Explicit Wait or just Wait is a one-timer used by you for a particular search. It is more extendible in the means that you can set it up to wait for any condition you might like. Usually, you can use some of the prebuilt Expected Conditions to wait for elements to become clickable, visible, invisible, etc., or just write your own condition that suits your needs.

**33).What is the difference between driver.Close() and driver.Quit () method?**

Close() – It is used to close the browser or page currently which is having the focus.

Quit() – It is used to shut down the web driver instance or destroy the web driver instance (Close all the windows)

**34). Difference between flex and flash application.**

In flash there is no code just based on creativity(design) we will complete the work(time consuming process) whereas flex contain some small functions which is integrated with mxml,PHP..(no tool is there to develop flex we want to use the properties of css and style sheet)

**35) Can captcha be automated?**

No, captcha and bar code reader cannot be automated.

**36).What is TestNG?**

So far we had been doing Selenium tests without generating a proper format for the test

results. From this point on, we shall tackle how to make these reports using a test framework called TestNG.

TestNG is a testing framework that overcomes the limitations of another popular testing framework called JUnit. The “NG” means “Next Generation”. Most Selenium users use this more than JUnit because of its advantages. There are so many features of TestNG, but we will only focus on the most important ones that we can use in Selenium. Advantages of TestNG over JUnit

There are three major advantages of TestNG over JUnit:

Annotations are easier to understand

Test cases can be grouped more easily

Parallel testing is possible

**37). How to disable cookies in browser.**

- Using deleteAllVisibleCookies() in selenium

**38). What is Selenese?**

Selenese is HTML language based command, which is used in Selenium IDE.

**39). Differences between QTP and selenium.**

- 1) Selenium generates a proxy while starting browser. QTP does not
- 2) QTP uses only Vb script. Selenium is available in many languages
- 3) QTP is paid and selenium is free.
- 4) You can run script from a particular line in QTP but in selenium, you cannot.
- 5) Selenium works on all browsers. QTP only works on IE, mozilla. Support from chrome has been introduced lately.
- 6) QTP is more organized and user friendly
- 7) Selenium requires more technical skills

**40. How does u handle dynamic elements without using xpath (with example?)**

- By using classname or css.

**41).. Which repository you have used to store the test scripts?**

I have created scripts in excel file and store them in Test cases folder under src .

**42). Detail about TestNG Test Output folder.**

It is the directory where reports are generated. Every time tests run in a suite, TestNG creates index.html and other files in the output directory.

**43). How do you create html test report from your test script?**

- I would see below 3 ways:
- Junit: with the help of ANT.
- TestNG: using inbuilt default.html to get the HTML report. Also XLST reports from ANT,

Selenium, TestNG combination.

- Using our own customized reports using XSL jar for converting XML content to HTML.

4) QTP can also be used on desktop based applications but selenium cannot be used

#### **44) What is Junit?**

Junit is a unit testing framework introduced by Apache. Junit is based on Java.

#### **45) 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.

#### **46) 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.

### **1. What is Selenium?**

Selenium is a set of tools that supports rapid development of test automation scripts for web based applications. Selenium testing tools provides a rich set of testing functions specifically designed to fulfil needs of testing of a web based application.

### **2. What are the main components of Selenium testing tools?**

Selenium IDE, Selenium Webdriver, Selenium RC and Selenium Grid

### **3. What is Selenium IDE?**

Selenium IDE is for building Selenium test cases. It operates as a Mozilla Firefox add on and provides an easy to use interface for developing and running individual test cases or entire test suites. Selenium-IDE has a recording feature, which will keep account of user actions as they are performed and store them as a reusable script to play back.

### **4. What is the use of context menu in Selenium IDE?**

It allows the user to pick from a list of assertions and verifications for the selected location.

### **5. Can tests recorded using Selenium IDE be run in other browsers?**

Yes. Although Selenium IDE is a Firefox add on, however, tests created in it can also be run in other browsers by using Selenium RC (Selenium Remote Control) and specifying the name of the test suite in command line.

## **6. What are the advantage and features of Selenium IDE?**

1. Intelligent field selection will use IDs, names, or XPath as needed
2. It is a record & playback tool and the script format can be written in various languages including

C#, Java, PERL, Python, PHP, HTML

3. Auto complete for all common Selenium commands
4. Debug and set breakpoints
5. Option to automatically assert the title of every page
6. Support for Selenium user-extensions.js file

## **7. What is Selenium RC (Remote Control)?**

Selenium RC allows the test automation expert to use a programming language for maximum flexibility and extensibility in developing test logic. For example, if the application under test returns a result set and the automated test program needs to run tests on each element in the result set, the iteration / loop support of programming language's can be used to iterate through the result set, calling Selenium commands to run tests on each item.

Selenium RC provides an API and library for each of its supported languages. This ability to use Selenium RC with a high level programming language to develop test cases also allows the automated testing to be integrated with the project's automated build environment.

## **8. What is Selenium Grid?**

Selenium Grid in the selenium testing suit allows the Selenium RC solution to scale for test suites that must be run in multiple environments. Selenium Grid can be used to run multiple instances of Selenium RC on various operating system and browser configurations.

## **9. How Selenium Grid works?**

Selenium Grid sent the tests to the hub. Then tests are redirected to an available Selenium RC, which launch the browser and run the test. Thus, it allows for running tests in parallel with the entire test suite.

## **10. What you say about the flexibility of Selenium test suite?**

Ans. Selenium testing suite is highly flexible. There are multiple ways to add functionality to

Selenium framework to customize test automation. As compared to other test automation tools, it is Selenium's strongest characteristic. Selenium Remote Control support for multiple programming and scripting languages allows the test automation engineer to build any logic they need into their automated testing and to use a preferred programming or scripting language of one's choice. Also, the Selenium testing suite is an open source project where code can be modified and enhancements can be submitted for contribution.

#### **11. What test can Selenium do?**

Selenium is basically used for the functional testing of web based applications. It can be used for testing in the continuous integration environment. It is also useful for agile testing

#### **12. What is the cost of Selenium test suite?**

Selenium test suite a set of open source software tool, it is free of cost.

#### **13. What browsers are supported by Selenium Remote Control?**

The test automation expert can use Firefox, IE 7/8, Safari and Opera browsers to run tests in Selenium Remote Control.

#### **14. What programming languages can you use in Selenium RC?**

C#, Java, Perl, PHP, Python, Ruby

#### **15. What are the advantages and disadvantages of using Selenium as testing tool?**

Advantages: Free, Simple and powerful DOM (document object model) level testing, can be used

for continuous integration; great fit with Agile projects.

Disadvantages: Tricky setup; dreary errors diagnosis; cannot test client server applications.

#### **16. What is difference between QTP and Selenium?**

Only web applications can be testing using Selenium testing suite. However, QTP can be used for testing stand alone applications. Selenium supports following web browsers:

Internet Explorer, Firefox, Safari, Opera on Windows, Mac OS X and Linux. However, QTP is limited to Internet Explorer on Windows. QTP uses scripting language implemented on top of VB Script. However, Selenium test suite has the flexibility to use many languages like Java, .Net, Perl, PHP, Python, and Ruby.

## **17. What is difference between Borland Silk test and Selenium?**

Selenium is completely free test automation tool, while Silk Test is not. Only web applications can be tested using Selenium testing suite. However, Silk Test can be used for testing client server applications. Selenium supports following web browsers: Internet Explorer, Firefox, Safari, Opera on Windows, Mac OS X and Linux. However, Silk Test is limited to Internet Explorer and Firefox. Silk Test uses 4Test scripting language. However, Selenium test suite has the flexibility to use many languages like Java, .Net, Perl, PHP, Python, and Ruby.

### **. What is selenium?**

Selenium is a browser automation tool that is used for automating web based applications. Selenium WebDriver uses browser's native methods to perform the automation task e.g. to click a button, selenium WebDriver will execute the selected browser's method that in turn will click the button.

### **. What are different forms of selenium?**

Selenium comes in 4 forms-

1. Selenium WebDriver - Selenium WebDriver is used to automate web applications using browser's native methods.
2. Selenium IDE - An IDE for selenium that works on record and play back principle.
3. Selenium RC - Selenium Remote Control(RC) is officially deprecated by selenium and it used to work on javascript to automate the web applications.
4. Selenium Grid - Allows selenium tests to run in parallel across multiple machines.

### **. How to execute javascript in selenium?**

JavaScript can be executed in selenium using JavaScriptExecutor. Sample code for javascript execution-

```
WebDriver driver = new FirefoxDriver();
if (driver instanceof JavascriptExecutor) {
    ((JavascriptExecutor)driver).executeScript("{JavaScript Code}");
}
```

### **. How to handle alerts in selenium?**

In order to accept or dismiss an alert box the alert class is used. This requires first switching to the alert box and then using accept() or dismiss() command as the case may be.

```
1 Alert alert = driver.switchTo().alert();
2 alert.accept();
```

**. Explain the difference between close and quit command.**

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

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

**. Explain the difference between implicit wait and explicit wait.**

Ans. An implicit wait, while finding an element waits for a specified time before throwing NoSuchElementException in case element is not found. The timeout value remains valid throughout the webDriver's instance and for all the elements.

```
driver.manage().timeouts().implicitlyWait(180, TimeUnit.SECONDS);
```

On the other hand, Explicit wait is applied for a specified element only-

```
WebDriverWait wait = new WebDriverWait(driver, 5);
wait.until(ExpectedConditions.presenceOfElementLocated(ElementLocator));
```

**. How to do drag and drop in selenium?**

Using Action class, drag and drop can be performed in selenium. Sample code-

```
Actions builder = new Actions(driver);
Action dragAndDrop = builder.clickAndHold(SourceElement)
.moveToElement(TargetElement)
.release(TargetElement)
.build();
dragAndDrop.perform();
```

**. What is HtmlUnitDriver?**

HtmlUnitDriver is the fastest WebDriver. Unlike other drivers (FireFoxDriver, ChromeDriver etc), the HtmlUnitDriver is non-GUI, while running no browser gets launched.

**. How to handle hidden elements in Selenium webDriver?**

Using javaScript executor we can handle hidden elements-

```
(JavascriptExecutor(driver)).executeScript("document.getElementsByClassName(ElementLo
cator).click();");
```

**. What are commonly used TestNG annotations?**

The commonly used TestNG annotations are-

- o @Test- @Test annotation marks a method as Test method.

- @BeforeSuite- The annotated method will be run only once before all tests in this suite have run.
- @AfterSuite-The annotated method will be run only once after all tests in this suite have run.
- @BeforeClass-The annotated method will be run only once before the first test method in the current class is invoked.
- @AfterClass-The annotated method will be run only once after all the test methods in the current class have been run.
- @BeforeTest-The annotated method will be run before any test method belonging to the classes inside the <test> tag is run.
- @AfterTest-The annotated method will be run after all the test methods belonging to the classes inside the <test> tag have run.

### **1. What is Selenium?**

Wikipedia defines Selenium as “a portable software testing framework for web applications. Selenium provides a record/playback tool for authoring tests without learning a test scripting language.”

In other words, Selenium is a browser automation tool that allows you to automate operations such as Type, Click, and Selection from a web page drop down.

### **2. How does Selenium score over QTP?**

- Selenium is an open source tool and scores over QTP which is a completely commercial tool.
- Selenium supports Firefox, IE, Opera, Safari on operating systems like Windows, Mac, Linux, among others while QTP is limited to Internet Explorer on the Windows OS.
- Selenium supports an array of programming languages such as Ruby, Perl, Python among others while QTP supports just VB script.

### **3. What is Selenium 1.0?**

Selenium 1.0 is also known as Selenium Remote Control. It is essentially a virtual library that is available in multiple languages.

### **4. What is Selenium 2.0?**

- Selenium 2.0 is also known as WebDriver. This is the latest offering of Selenium that aims to:
- Provide a better, more robust API compared to Selenium 1.0
- Overcome the Java script security restriction that hinders the performance of Selenium 1.0.
- Supports more UI operations such as drag and drop

### **5. What is Selenium IDE?**

Selenium IDE is a record and playback tool which is distributed as a Firefox plugin. Selenium IDE can be used only with Firefox browser.

### **6. Which is the latest Selenium tool?**

Selenium WebDriver

### **7. List the different Selenium tools.**

- Selenium Integrated Development Environment (IDE)

- Selenium Remote Control (RC)
- Selenium WebDriver
- **Selenium Grid**

8. What kind of testing does Selenium support?

Selenium supports Functional Testing and Regression Testing.

#### **9. What is Selenese?**

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

#### **10. What are the different ways in which Selenium IDE can be opened?**

Selenium IDE can be opened either through the side bar or as a pop up window.

#### **11. What is Selenium Grid?**

Selenium grid is the functionality that lets you distribute your tests simultaneously across multiple machines. Selenium Grid helps in reducing the time taken for test execution, and provides instant feedback to stakeholders.

#### **12. When should one use Selenium Grid?**

Selenium Grid could be used to execute the same or different test scripts on multiple platforms and browsers in parallel. It helps achieve distributed test execution, testing under different environments and saving execution timelines.

#### **13. What is difference between Assert and Verify commands?**

The Assert command checks if a particular condition is true or false. If the condition is true, the program control will execute the next test step. If the condition is false, the execution would stop and no further tests would be executed.

The Verify command also checks if a particular condition is true or false. Irrespective of the condition being true or false, the program execution doesn't stop.

#### **14. List the different types of locators in Selenium.**

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

#### **15. What is the function of Xpath?**

Xpath is a locator that 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, similar to HTML tags.

#### **16. Explain the difference between single and double slash in X-path.**

A single slash ( / ) begins selection from the document node, while a double slash ( // ) begins selection matching anywhere in the document.

**17. List the different types of Drivers in WebDriver.**

- FirefoxDriver
- InternetExplorerDriver
- ChromeDriver
- SafariDriver
- OperaDriver
- AndroidDriver
- iPhoneDriver
- HtmlUnitDriver

**18. What is the difference between Verification and Assertion?**

Verification and Assertion are two Check functionalities. Verification allows test execution to continue even when Check fails, while assertion stops the test execution.

**19. List the parameters that one needs to pass in Selenium.**

- Host
- Port Number
- Browser
- URL

**20. Explain the difference between Implicit Wait and Explicit Wait.**

In Selenium, the Implicit Wait function sets a timeout for all successive web element searches. For a certain amount of time, it will attempt to look for an element repeatedly before throwing a NoSuchElementException. ExplicitWait, on the other hand, is a one-time function that is used for a particular search.

**21. How does one submit a form in Selenium?**

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

**22. How does one execute Java scripts function in Selenium?**

```
JavascriptExecutor js = (JavascriptExecutor) driver; String title = (String) js.executeScript("pass your java scripts");
```

**23. How does one calculate the number of rows using Selenium 2.0?**

```
JavascriptExecutor js = (JavascriptExecutor) driver;
```

```
String title = (String) js.executeScript("pass your java scripts");
```

**24. How does one capture a page title in Selenium 2.0?**

```
String title = driver.getTitle()
```

**13 --> What type locators use in selenium webdriver?**

The locators elements of Selenium Webdriver as run as following with java syntax and example.

Method: By ID

Syntax: `driver.findElement(By.id(<element ID>))`

Description: Locates an element using the ID attribute. The most efficient way and preferred way to locate an element on a web page is By ID. ID will be the unique on web page which can be easily identified. IDs are the safest and fastest locator option and should always be the first choice even when there are multiple choices, It is like an Employee Number or Account which will be unique.

Example:

```
<input type="text" id="email" name="email" class="inputtext">  
<div id="main_menu">Home.</div>
```

Selenium Webdriver write as:

```
WebElement loginemail= driver.findElement(By.id("email"));  
String menuname= driver.findElement(By.id("main_menu")).getText();  
driver.findElement(By.id("email")).sendKeys("abcd@gmail.com");
```

Method: By class name

Syntax: driver.findElement(By.className(<element class>)),  
driver.findElements(By.className(<element class>))

Description: Locates an element using the Class attribute. There may be multiple elements with the same name. We use findElement() method for single element class name. We use findElements() for multiple elements with the same name.

Example:

```
<input type="text" id="email" name="email" class="inputtext">  
<input type="password" id="pass" name="pass" class="inputtext">
```

or

```
<input name="register" class="required" type="text"/>  
<div class="mobile">8801911444444.</div>
```

Selenium Webdriver write as:

```
List<WebElement> noinputf = table.findElements(By.className("inputtext"));
```

or

```
WebElement loginemail= driver.findElement(By.className("required"));  
WebElement loginemail= driver.findElement(By.className("mobile")).getText();
```

What is automation testing?

Automation testing is a process of automating the existing manual process to test the application. Separate tools are available for automation testing which lets you create test scripts which can be executed repeatedly and does not require any manual intervention.

### **What are the advantages/benefits of automation testing?**

- Decrease costs and reduces test time.
- Increase Test Efficiency, Test Speed and Software Quality.
- Improve accuracy.
- Reduce human-generated errors.

### **What is the need of automation testing?**

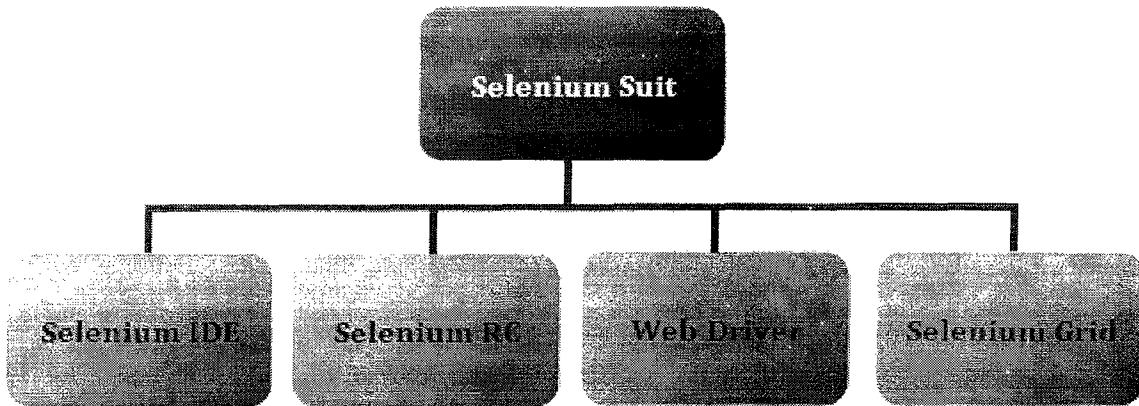
It is a challenge for any company to continuously maintain the quality and efficiency of the application. So with the use of automation testing, we can improve the quality and efficiency of our application continuously. Automated test cases can run fast frequently, which is cost-effective for software products with a long maintenance life.

### **What is Selenium?**

Selenium is an open source automation tool which provides the facility to automate web application. It is composed of Selenium IDE. It's a tool for recording & playing back. Selenium is designed to support automation testing of function aspects of web application and platforms. It supports multiple programming languages like

- Java
- .Net
- Python
- Perl
- Ruby etc.

Selenium is not a single tool rather a package of several testing tools. It has four components.



#### Selenium IDE:

Selenium IDE is an Integrated Development Environment for selenium tests. It is implemented as a Firefox-Add-On and provides the features for recording, editing and debugging tests. It was previously known as Selenium Recorder.

#### Selenium RC:

Selenium RC is known as a Selenium Remote Control. It is a server that provides the facility to write automated web application UI tests in desired programming language against any HTTP website using any JavaScript-enabled browser.

#### Web Driver:

Web Driver is designed in a way to provide a simple, more abbreviated programming interface in addition to addressing some limitations in Selenium RC. Web Driver was developed to providing the better support to dynamic web pages where elements of a page may change without the page itself being reloaded.

#### Selenium Grid:

Selenium Grid allows you to execute your tests on different machines against different browsers at the same time. Selenium Grid supports distributed test execution. It allows to execute your tests in distributed environment.

- See more at:

<http://modernpathshala.com/Learn/Selenium/Interview#sthash.XcOZdjYL.dpuf>

#### What are the limitations of Selenium?

- Selenium does not support Mobile Application testing.
- Selenium supports only web based application testing.

- We can't test Captcha & Bar Code using Selenium.
- We can generate test reports using third party tools like JUnit or TestNG.

### **What are the types of testing that can be supported by selenium?**

- Regression Testing.
- Functional Testing.

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

Locators are used to identify the web elements accurately in a web page. Different types of locators are available in Selenium:

Different types of locators are available in Selenium are as following:

### **What are the different types of drivers are available in WebDriver?**

- FireFoxDriver.
- ChromeDriver.
- InternetExplorerDriver.
- SafariDriver.
- OperaDriver.
- AndroidDriver.
- HtmlUnitDriver.
- iPhoneDriver.

### **How to set a value in textbox using Selenium?**

Use the following code to set the value in the text box.

```
WebElement element = driver.findElement(By.id("elementId"));
element.sendKeys("Value");
```

### **How to clear a particular textbox using Selenium?**

Use following code to clear text box values.

```
WebElement element = driver.findElement(By.id("elementId"));
element.clear();
```

Note: findElement() is used to search an element in current page matching to the specified value. We can identify the element by id or by name.

### **How to select drop down value using selenium?**

Use the following code to select a value in dropdown control.

```
Select dropDown = new Select(driver.findElement(By.id("drop_down_id")));
```

#### **Select By VisibleText:**

```
dropDown.selectByVisibleText("Text");
```

#### **Select By Index:**

```
dropDown.selectByIndex(1);
```

#### **Select By Value:**

```
dropDown.selectByValue("Value");
```

How will you find that an element is displayed on the screen?

Web Driver provides the methods to check the visibility of the web elements. These can be button, checkboxes, labels, radio buttons etc.

- isDisplayed()
- isSelected()
- isEnabled()

#### **Syntax:-**

##### **isDisplayed():**

```
Boolean isDisplayed = driver.findElement(By.id("item_id")).isDisplayed();
```

##### **isSelected():**

```
Boolean isSelected = driver.findElement(By.id("item_id")).isSelected();
```

##### **isEnabled():**

```
Boolean isEnabled = driver.findElement(By.id("item_id")).isEnabled();
```

### **How to click on a hyperlink using link text?**

Use below code to click on the hyperlink.

```
Driver.findElement(By.linkText("www.modernpathshala.com")).click();
```

### **What is the difference between Single Slash "/" and Double Slash "//"?**

#### **Single Slash(/):**

It starts the selection from document node and allows you to create absolute path expressions.

**e.g. "/html/body/a"** matches all the a tag elements.

#### **Double Slash(//):**

It starts the selection matching anywhere in the document and allows you to create relative path expressions.

**e.g. "//a"** matches all the a tag elements.

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

Get command is used to get the inner text of the specified web element and its return string value.

```
String value = driver.findElement(By.id("Input_Field_Id")).getText();
```

### **What is the difference between verify and assert commands?**

**Verify:** Verify command checks whether an element is on the page or not, 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 tests fails.

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

### **What are the parameter you have to pass in Selenium?**

The parameters which have to be passed in Selenium are listed below

1. Host
2. Port Number
3. Browser
4. URL

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

Both are used to delay the speed of execution.

**Sleep():** It suspends the current java thread for specified period of time. But one Thread.sleep() will set up wait only once. It takes the single argument in integer format.

**Syntax:** Thread.sleep(3000) – It will wait for 3 seconds.

**Example:** Suppose we have 3 selenium operations.

Operation1

Operation2

Operation3

If we want to set delay time 3000 milliseconds for each of these. It's will be done as-

```
Thread.sleep(3000);
```

Operation1

```
Thread.sleep(3000);
```

Operation2

```
Thread.sleep(3000);
```

Operation3

**SetSpeed():** It stops the execution for every selenium command for a specific amount of time. It takes single argument.

**Syntax:** selenium.setSpeed("3000");

**Example:** Suppose we have 3 selenium operations.

Operation1

Operation2

Operation3

We want to set delay time 3000 for each of these. It will be done as:-

```
selenium.setSpeed("3000");
```

Operation1

Operation2

Operation3

## **How can we submit a form using Selenium?**

You can submit form in Selenium using below code snippet

```
driver.findElement(By.id("id")).submit();
```

### **What is difference between findElement() & findElements()?**

#### **findElement():**

1. It finds the element on the current page as per the given element locator mechanism.
2. It returns only the single element.

**Syntax:** `WebElement element = webdriver.findElement(By.id("item_id"));`

1. If the element not found on the page then it will throw NoSuchElementException.

#### **findElements():**

1. It finds all the element on the current page as per the given element locator mechanism.
2. It returns the list of elements.

**Syntax:** `List<WebElement> element = webdriver.findElements(By.id("item_id"));`

1. If any element not found on current page as per the given element locator mechanism, It will return empty list.

### **What is the difference between click() & submit() in Selenium?**

**Click()** & **Submit()** both are used to click button in web page.

Selenium Web Driver provides the special method **submit()** to submit any form. **Submit()** method works same as clicking on submit button.

**Submit():** We can use **submit()** method for only submit form. There are some restrictions that are given below:-

1. Element's type="submit" & button should be inside `<form>` tag, then only **submit()** method will work.
2. If element's type="button", then **submit()** will not work.
3. If button is outside of `<form>` tag, then **submit()** will not work.

**Click():** We can use **click()** method to click on any button. There is no restrictions for click.

1. Element's type="button" or type="submit", then click method will work for both type.
2. If button is inside the <form> tag or outside the <form> tag, then click method will work.

### **What are different types of annotation which are useful?**

Different types of annotations which are useful are listed below-

1. Test
2. Before
3. After
4. Ignore
5. BeforeClass
6. AfterClass
7. RunWith

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

Selenium WebDriver provides the two types of wait:

1. Implicit Wait
2. Explicit Wait

**Implicit Wait:** Implicit provides the facility to poll the DOM for a certain amount of time when trying to find the element if they are not immediately. Default time setting is 0.

#### **Syntax:**

```
WebDriver driver = new FirefoxDriver();

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

driver.get("https://www.modernpathshala.com/");

WebElement element = driver.findElement(By.id("Element_Id"));
```

**Explicit Wait:** An explicit waits is code you define to wait for a certain condition to occur before proceeding further in the code. The worst case of this is time.sleep(), which sets the condition to an exact time period to wait.

#### **Syntax:**

```
WebDriver driver = new FirefoxDriver();
driver.get("https://www.modernpathshala.com/");
WebElement myDynamicElement = (new WebDriverWait(driver, 10)).until(ExpectedConditions.presenceOfElementLocated(By.id("myDynamicElement")));
```

### **What is difference between driver.close() and driver.quit()?**

#### **driver.close():**

It is used to close the browser or page currently which is having the focus.

#### **driver.quit():**

It is used to shut down the web driver instance or destroy the web driver instance means close all the browser windows. Basically it calls the driver.dispose method which in turn closes all the browser windows and ends the web driver session gracefully.

### **What is the TextNG?**

TestNG is a testing framework for Java programming language and designed to simplify a broad range of testing needs, from unit testing to integration testing(testing entire system made of several classes and packages). In other words, TestNG is designed to cover all categories of testing like unit, end-to-end, functional, integration etc. where **NG** means **Next Generation**. It is an open source framework and inspired by JUnit. It is similar to Junit and designed to be better than Junit.

### **Instantiate Specific Browser Or Client**

You may need this code in every selenium script. So I thought of keeping it here handy for the popular ones.

#### **Android Driver**

```
WebDriver driver = new AndroidDriver()
```

#### **iPhone Driver**

```
WebDriver driver = new iPhoneDriver();
```

#### **HTML Unit**

```
WebDriver driver = new HtmlUnitDriver()
```

### **Check If An Element Exists**

You may need to perform a action based on a specific web element being present on the web page. You can use below code snippet to check if an element with id “element-id” exists

on web page.

```
driver.findElements(By.id("element-id")).size()!=0
```

### How To Check If An Element Is Visible With WebDriver

The above mentioned method may be good to check if a element exists on page. However sometimes a element may be not visible, therefore you can not perform any action on it. You can check whether an element is visible or not using below code.

```
WebElement element = driver.findElement(By.id("element-id"));
if(element instanceof RenderedWebElement) {
    System.out.println("Element visible");
} else {
    System.out.println("Element Not visible");
}
```

### Refresh Page

This may be required often. Just a simple refresh of the page equivalent to a browser refresh.

```
driver.navigate().refresh();
```

### Navigate Back And Forward On The Browser

Navigating the history of browser can be easily done using below two methods. The names are self explanatory.

```
//Go back to the last visited page
driver.navigate().back();
```

```
//go forward to the next page
driver.navigate().forward();
```

### Wait For Element To Be Available

The application may load some elements late and your script needs to stop for the element to be available for next action. You can perform this check using below code.  
In below code, the script is going to wait maximum 30 seconds for the element to be available. Feel free to change the maximum number per your application needs.

```
WebDriverWait wait = new WebDriverWait(driver, 30);
WebElement element =
wait.until(ExpectedConditions.elementToBeClickable(By.id("id123")));
```

### Focus On A Input Element On Page

Doing focus on any element can be easily done by clicking the mouse on the required element. However when you are using selenium you may need to use this workaround instead of mouse click you can send some empty keys to a element you want to focus.

```
WebElement element = driver.findElement(By.id("element-id"));
//Send empty message to element for setting focus on it.
element.sendKeys("");
```

### Overwrite Current Input Value On Page

The sendKeys method on WebElement class will append the value to existing value of element. If you want to clear the old value. You can use clear() method.

```
WebElement element = driver.findElement(By.id("element-id"));
element.clear();
element.sendKeys("new input value");
```

### Mouseover Action To Make Element Visible Then Click

When you are dealing with highly interactive multi layered menu on a page you may find this useful. In this scenario, an element is not visible unless you click on the menu bar. So below code snippet will accomplish two steps of opening a menu and selecting a menu item easily.

```
Actions actions = new Actions(driver);
WebElement menuElement = driver.findElement(By.id("menu-element-id"));
actions.moveToElement(menuElement).moveToElement(driver.findElement(By.xpath("//of-menu-item-element"))).click();
```

## Extract CSS Attribute Of An Element

This can be really helpful for getting any CSS property of a web element.  
For example, to get background color of an element use below snippet

```
String bgcolor = driver.findElement(By.id("id123")).getCssValue("background-color");
```

and to get text color of an element use below snippet

```
String textColor = driver.findElement(By.id("id123")).getCssValue("color");
```

## Find All Links On The Page

A simple way to extract all links from a web page.

```
List link = driver.findElements(By.tagName("a"));
```

## Take A Screenshot On The Page

The most useful one. Selenium can capture the screenshot of any error you want to record.  
You may want to add this code in your exception handling logic.

```
File snapshot =((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
```

## Execute A JavaScript Statement On Page

If you love JavaScript, you are going to love this. This simple JavascriptExecutor can run any javascript code snippet on browser during your testing. In case you are not able to find a

way to do something using web driver, you can do that using JS easily.  
Below code snippet demonstrates how you can run a alert statement on the page you are testing.

```
JavascriptExecutor jsx = (JavascriptExecutor) driver;  
jsx.executeScript("alert('hi')");
```

### Upload A File On A Page

Uploading a file is a common use case. As of now there is not webdriver way to do this, however this can be easily done with the help of JavascriptExecutor and little bit of JS code.

```
String filePath = "path\\to\\file\\for\\upload";  
JavascriptExecutor jsx = (JavascriptExecutor) driver;  
jsx.executeScript("document.getElementById('fileName').value='" + filePath + "'");
```

### Scroll Up, Down Or Anywhere On A Page

Scrolling on any web page is required almost always. You may use below snippets to do scrolling in any direction you need.

```
JavascriptExecutor jsx = (JavascriptExecutor) driver;  
//Vertical scroll - down by 100 pixels  
jsx.executeScript("window.scrollBy(0,100)", "");  
//Vertical scroll - up by 55 pixels (note the number is minus 55)  
jsx.executeScript("window.scrollBy(0,-55)", "");  
//Horizontal scroll - right by 20 pixels  
jsx.executeScript("window.scrollBy(20,0)", "");  
//Horizontal scroll - left by 95 pixels (note the number is minus 95)  
jsx.executeScript("window.scrollBy(-95,0)", "");
```

### Get HTML Source Of A Element On Page

If you want to extract the HTML source of any element, you can do this by some simple Javascript code.

```
JavascriptExecutor jsx = (JavascriptExecutor) driver;
```

```
String elementId = "element-id";
String html =(String) js.executeScript("return document.getElementById('" + elementId +
"').innerHTML;");
```

## Switch Between Frames In Java Using Webdriver

Multiple iframes are very common in recent web applications. You can have your webdriver script switch between different iframes easily by below code sample

```
WebElement frameElement = driver.findElement(By.id("id-of-frame"));
driver.switchTo().frame(frameElement);
```

Web driver is really a powerful functional testing tool. I have not looked into any other libraries and tools after I have started using Webdriver ( selenium ) . The learning curve is really short for any java developer.

Earlier version of Selenium required a separate remote server to run the scripts. The latest version does not even require that. You can directly run a selenium webdriver script using webdriver library. This makes life of a developer even more easy.

For a quick start you may want to try the Selenium IDE. It has recording feature, that can record user actions on browser and then generate code in any supported language like Java, Python, Ruby and .NET

## 3.What is Selenium 2.0?

- i. Selenium 2. is the latest offering of Selenium.It is known as Web Driver.
- ii. It provides API's better than Selenium 1.0
- iii. It supports more UI operations.
- iv. It is not suffering from Java Script security restrictions which is available in Selenium 1.0.

## 4.What are the element locators available in Selenium to locate elements on the web page?

The element locators are:-

- i. Html id
- ii. Html name
- iii. XPath locator
- iv. Css Locator

## **5.What are the verification points available with Selenium?**

Three types of verification points are available in selenium:

- i. Check for page Title
- ii. Check for Certain text
- iii. Check for certain element(Table, dropdown text box etc)

## **6.What is XPath?**

XPath is a way to navigate in XML document.It is also used to identify elements in a web page.

## **7.What is the difference between type and typeKeys commands?**

type commands simulates enter operations at one go while type Keys simulates Keystroke key by key.typekeys could be used when typing data in textbox which bring options because such operations are not usually simulated using type command.

Hey guys, here we come-up with Selenium software interview questions set, this set contains top 80 most frequently asked Selenium framework interview questions during Testing interview. So if you are a tester and preparing for Selenium webdriver interview questions then this set is for you, If you carefully go through all 80 questions then you will earn good confidence before your interview so must go through all these questions and share your interview questions if you have more interview questions we will add in this series.

### **1). What is Automation Testing?**

Automation testing or Test Automation is a process of automating the manual process to test the application/system under test. Automation testing involves use of a separate testing tool which lets you create test scripts which can be executed repeatedly and doesn't require any manual intervention.

### **2). What are the benefits/advantages of Automation Testing?**

Benefits of Automation testing are:

1. Supports execution of repeated test cases
2. Aids in testing a large test matrix
3. Enables parallel execution
4. Encourages unattended execution
5. Improves accuracy thereby reducing human generated errors
6. Saves time and money
7. Better Quality Software

**3). What are the disadvantages of Automation Testing?**

1. High investment is needed in the tools and training
2. High man power requirement for test preparations
3. A lot of testing areas left uncovered

**4). Explain 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

**5). 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.

**6). Which is the latest Selenium tool?**

WebDriver

**7). What are the testing types that can be supported by Selenium?**

Selenium supports the following types of testing:

1. Functional Testing
2. Regression Testing

**8). 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

## **9). What are the different types of waits available in WebDriver?**

There are two types of waits available in WebDriver:

- 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.

## **10). 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.

## **11). 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.

## **13). What is Selenese?**

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

## **14). 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.

## **15). 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 among 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) – 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 – 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 – Selenium Grid is used to distribute your test execution on multiple platforms and environments concurrently.

## **16). What is an Xpath?**

Xpath 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.

## **17). 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");
```

## **18). How can you find if an element is 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():

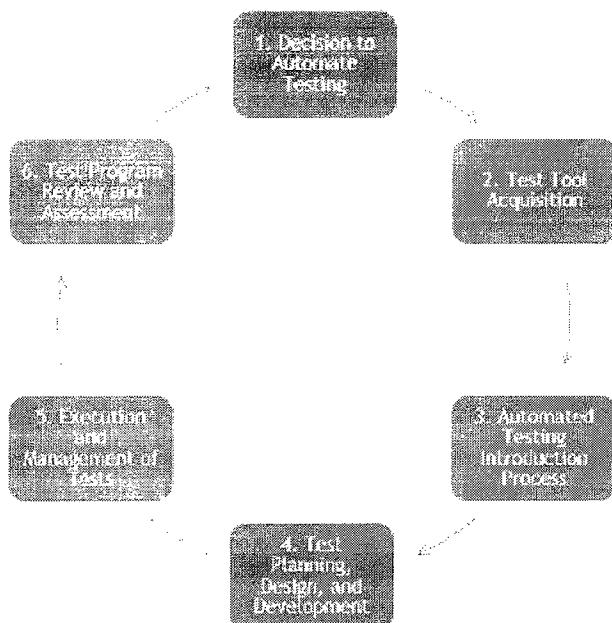
```
boolean buttonSelected = driver.findElement(By.id("gbqfba")).isSelected();
```

isEnabled():

```
boolean searchIconEnabled = driver.findElement(By.id("gbqfb")).isEnabled();
```

### 19). Can you explain the phase of Automation Testing LifeCycle?

- Outline the potential benefits and test tool proposal
- Test tool evaluation and selection
- Steps necessary to outline automated testing to the project
- Identifies the test procedure standards, defines the tests, defines development standard
- Test plans are executed
- This is done throughout the life-cycle



### 20). 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:

**21). 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

**22). 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.

**23). What are the steps to run automation using selenium?**

The very basic steps are:

1. Record the test steps using selenium-IDE.
2. Modify the script according to the testing needs. Add validation points, Java Scripts, Time-out etc.
3. Run the test.
4. View the result after test run complete analyze.

**24). What are the capabilities of Selenium IDE?**

Selenium IDE (Integrated Development Environment) works similar to commercial tools like QTP, Silk Test and Test Partner etc.

The below points describes well about Selenium IDE.

1. Selenium IDE is a Firefox add-on.

2. Selenium IDE can support recording the clicks, typing, and other actions to make a test cases.
3. Using Selenium IDE, a tester can play back the test cases in the Firefox browser.
4. Selenium IDE supports exporting the test cases and suites to Selenium RC.
5. Debugging of the test cases with step-by-step can be done.
6. Breakpoint insertion is possible.
7. Page abstraction functionality is supported by Selenium IDE.
8. Selenium IDE can supports an extensibility capability allowing the use of add-ons or user extensions that expand the functionality of Selenium IDE

#### **25). 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

```
// Storing the list
List elementList = driver.findElements(By.xpath("//div[@id='example']/ul//li"));
// Fetching the size of the list
int listSize = elementList.size();
for (int i=0; i <
// Clicking on each service provider link
serviceProviderLinks.get(i).click();
// Navigating back to the previous page that stores link to service providers
driver.navigate().back();
}
```

#### **26). 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 it 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 it return any value.

#### **27). 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();
```

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.

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);
```

## **28). What are the different types of navigation commands?**

Following are the navigation commands:

`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");
```

**29). How to get title?**

```
driver.getTitle();~To Print: System.out.println( driver.getTitle());
```

**30). 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.

**31). 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.

**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. 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();
```

**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.

**34). 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.

### **35). 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"));
```

### **36). 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.

### **37). How to handle frame in WebDriver?**

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");
```

### **38). When do we use findElement() and findElements()?**

**findElement():** findElement() is used to find the first element in the current web page matching to the specified locator value. Take a note that only first matching element would be fetched.

Syntax:

```
WebElement element =  
driver.findElements(By.xpath("//div[@id='example']//ul//li"));
```

findElements():

findElements() is used to find all the elements in the current web page matching to the specified locator value. Take a note that all the matching elements would be fetched and stored in the list of WebElements.

Syntax:

```
List elementList =  
driver.findElements(By.xpath("//div[@id='example']//ul//li"));
```

### **39). What are the advantages of Automation framework?**

Advantage of Test Automation framework

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

### **40). 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.

### **41). 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.

### **42). 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:

```
// Instantiating Action Interface  
Actions actions=new Actions(driver);  
// howering on the dropdown  
actions.moveToElement(driver.findElement(By.id("id of the dropdown"))).perform();  
// Clicking on one of the items in the list options
```

```
WebElement subLinkOption=driver.findElement(By.id("id of the sub link"));
subLinkOption.click();
```

**43). Can captcha be automated?**

No, captcha and bar code reader cannot be automated.

**44). 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” ..

**45). 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)

**46). What are the advantages of Selenium?**

- It supports C#, PHP, Java, Perl, Python
- 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

**47). 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");
```

**48). 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

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

:Both will delay the speed of execution.

Thread.sleep () : It will stop the current (java) thread for the specified period of time. Its done only once

- It takes a single argument in integer format

Ex: thread.sleep(2000)- It will wait for 2 seconds

- It waits only once at the command given at sleep

`SetSpeed ()` : For specific amount of time it will stop the execution for every selenium command.

- It takes a single argument in integer format

Ex: `selenium.setSpeed("2000")`- It will wait for 2 seconds

- Runs each command after setSpeed delay by the number of milliseconds mentioned in set Speed

This command is useful for demonstration purpose or if you are using a slow web application

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

#### **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.

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

#### **51). What is Selenium 2.0 ?**

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

#### **52). 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.

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

**53). 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

**54). 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 launched 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

**55). 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

**56). 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

**57). 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

Selenium IDE captures 3 options?

Command, Target, Value

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

Implicit Wait:Sets a timeout for all successive Web Element searches. For the specified amount of time it will try looking for element again and again before throwing a NoSuchElementException. It waits for elements to show up.

Explicit Wait : It is a one-timer, used for a particular search./p>

**59). 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.

**60). 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);
```

**61). Can we use Selenium grid for performance testing?**

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

**62). Can Selenium test an application on Android browser?**

Selenium can handle Android browser.

### **63). Which browsers does WebDriver support?**

The existing drivers are the ChromeDriver, InternetExplorerDriver, FirefoxDriver, OperaDriver and HtmlUnitDriver. For more information about each of these, including their relative strengths and weaknesses, please follow the links to the relevant pages. There is also support for mobile testing via the AndroidDriver, OperaMobileDriver and iPhoneDriver

### **64). What tests can selenium do?**

Selenium could do functional, regression, and load of web based applications.p>

### **65). 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 ned to switch to each and every frame and search for locator which we want.

Then break the loop

### **66). How do I execute Javascript directly?**

We believe that most of the time there is a requirement to execute Javascript there is a failing in the tool being used: it hasn't emitted the correct events, has not interacted with a page correctly, or has failed to react when an XMLHttpRequest returns. We would rather fix WebDriver to work consistently and correctly than rely on testers working out which Javascript method to call.

We also realise that there will be times when this is a limitation. As a result, for those browsers that support it, you can execute Javascript by casting the WebDriver instance to a JavascriptExecutor. In Java, this looks like:

```
WebDriver driver; // Assigned elsewhere  
  
JavascriptExecutor js = (JavascriptExecutor) driver;  
  
js.executeScript("return document.title");
```

Other language bindings will follow a similar approach. Take a look at the UsingJavascript page for more information.

### **67). My XPath finds elements in one browser, but not in others. Why is this?**

The short is that each supported browser handles XPath slightly differently, and you're probably running into one of these differences. The long is on the XpathInWebDriver page..

### **68). Explain what is the difference between find elements () and find element () ?**

**find element ():**

It finds the first element within the current page using the given “locating mechanism”. It returns a single WebElement

**findElements () :**

Using the given “locating mechanism” find all the elements within the current page. It returns a list of web elements.

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

### **69). What is Selenium IDE?**

Selenium IDE is an integrated development environment for Selenium tests. It is implemented as a Firefox extension, and has a recording feature; which will keep account of user actions as they are performed and store them as a reusable script to play back. Selenium-IDE also offers full editing of test cases for more precision and control.

### **70). Why is my Javascript execution always returning null?**

You need to return from your javascript snippet to return a value, so:

js.executeScript("document.title");

will return null, but:

js.executeScript("return document.title");

will return the title of the document.

### **71). What is WebDriver?**

WebDriver is a tool for writing automated tests of websites. It aims to mimic the behaviour of a real user, and as such interacts with the HTML of the application

### **72). Can Selenium test a application on iPhone's Mobile Safari browser?**

Selenium can handle Mobile Safari browser. There is experimental Selenium iPhone Driver for running tests on Mobile with Safari on the iPhone and iPad and iPod Touch.

### **73). What are the disadvantages of Selenium?**

Disadvantages of Selenium:

- Limitation in terms of browser support (It runs only in Mozilla). Scripts written using Selenium IDE can be used for other browsers only if it is used with Selenium RC or Selenium Core.
- We can't run recorded script if it is converted to Java, C#, Ruby etc.
- Not allowed to write manual scripts like conditions and Loops for Data Driven Testing
- There is no option to verify images

**74). Explain what is Datadriven framework and Keyword driven?**

Datadriven 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.

Keyworddriven 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.

**76). 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

**77). 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:abc@www.gmail.com`

**78). What is the selenium's recording language?**

Selenium's recording language is "HTML".

**79). What does it mean to be "developer focused"?**

We believe that within a software application's development team, the people who are best placed to build the tools that everyone else can use are the developers. Although it should be easy to use WebDriver directly, it should also be easy to use it as a building block for more sophisticated tools. Because of this, WebDriver has a small API that's easy to explore by hitting the "autocomplete" button in your favorite IDE, and aims to work consistently no matter which browser implementation you use.

**80). What are the steps to run automation using selenium?**

The very basic steps are:

1. Record the test steps using selenium-IDE.
2. Modify the script according to the testing needs. Add validation points, Java Scripts, Time-out etc.
3. Run the test.
4. View the result after test run complete analyze.

**1) What is Selenium?**

> Selenium is a suite of tools to automate web browsers, Selenium supports web Applications testing only, doesn't support Windows based client/server Applications testing.

> Selenium tools are mainly used for Functional and Regression Testing of Web Applications.

**2) What are the Components or tools of Selenium?**

Selenium IDE

Selenium RC

Selenium WebDriver

Selenium Grid

Note:

Selenium 1.0 (Selenium IDE + Selenium RC + Selenium Grid)

Selenium 2.0 (Selenium IDE + Selenium RC + Selenium WebDriver + Selenium Grid)

### **3) What is selenium IDE?**

- > It is a test tool in Selenium suite, used record and play back test cases.
- > It has IDE to create and execute test cases, but it doesn't support programming to enhance test cases.
- > Selenium IDE supports Mozilla Firefox browser only.

### **4) What is Selenium WebDriver?**

- > It is test tool in Selenium suite, using element locators and Webdriver methods we can create and execute test cases.
- > Selenium Webdriver supports Programming (Either Java or Perl or PHP or C#.Net or Python or Ruby) to enhance test cases.
- > Selenium WebDriver supports Mozilla Firefox, IE, Google Chrome, Safari etc...Browsers.
- > It supports MS Windows, UNIX and Macintosh operating environments.

### **5) What is Selenium Grid?**

- > Selenium Grid is a tool in Selenium suite to execute test cases in Parallel,

### **6) What are the advantages of Selenium?**

- > Selenium suite of tools are open source tools, anybody can download and use with free of cost.
- > Selenium supports various operating environments like MS Windows, UNIX, Macintosh etc...
- > Selenium supports various Programming languages to enhance test cases.
- > Selenium supports various Browsers.

### **7) What are the disadvantages of Selenium?**

- > Since it is Open source suite of tools no reliable support from anybody.
- > No centralized maintenance of Objects.

- > New features may not work properly.
- > It doesn't support Windows based Client/server applications.
- > Limited support for Image testing.

## **8) How to configure Selenium?**

### **Selenium IDE:**

Launch Firefox browser, Download Selenium IDE from [www.seleniumhq.org](http://www.seleniumhq.org) and install.

### **Selenium WebDriver:**

- > Download Eclipse IDE and extract
- > Download Java software and install.
- > Download Selenium Webdriver Java language bindings from Seleniumhq.org website.
- > Create Java project in Eclipse and add selenium WebDriver jar file to Java project in Eclipse.

## **9) What is JUnit?**

- > JUnit is a Testing framework used for Unit testing
- > JUnit is also used with Selenium for Functional Testing.
- > Using JUnit with Selenium we can group test cases, execute test batches and generate detailed test reports.

## **10) What is TestNG?**

- > TestNG is a test automation framework used in Selenium to group test cases, executes series of test cases and generate detailed test reports.

## **11) What about Selenium certification?**

- > Since it is open source software, no authorized certification exam for Selenium.
- > Some organizations are conducting Selenium certification exam but no value guarantee.

## **12) What is Selenium license?**

- > All Selenium projects released under Apache 2.0 License.
- > Anybody can download and use Selenium suite of tools with free of cost.
- > Anybody can update the source code and use.
- > Selling updated code is not allowed, this is the primary objective of Apache 2.0 license.

## **13) What is ANT?**

- > Apache Ant is a tool that automates the software build process. It also supports testing.
- > Ant is used for code compilation, deployment, and execution process.

## **14) What is Jenkins and what is the usage of Jenkins in Selenium?**

- > Jenkins is an open source continuous integration tool written in Java.
- > It is cross-platform and can be used on Windows, Linux, Mac OS and Solaris environments.
- > Running Selenium tests in Jenkins allows us to run our tests every time our software changes and deploy the software to a new environment when the tests pass.
- > Jenkins can schedule our tests to run at specific time.

## **15) What is Maven?**

- > Maven is a build automation tool used primarily for Java projects.
- 

### **Choose Selenium tools and others:**

For Functional and Regression Testing of your web bases applications choose following tools and plug-ins.

- > Eclipse IDE to create and execute Java programs/WebDriver & TestNG Test cases.
- > Java Programming language to enhance Selenium WebDriver Test cases.
- > Selenium WebDriver to recognize elements (objects) in our web application and perform operations on elements.
- > Firebug and Firepath to inspect elements in Mozilla Firefox browser.

> TestNG Framework for grouping test cases, executing series of test cases and generating test reports.

> Maven

## **Interviews on Selenium Fundamentals**

### **1) What is Selenium?**

- Selenium is an Open Source Software, used for Test Automation.
- Selenium is a suite of tools that automate web browsers.
- Selenium suite of tools are mainly used for Functional and Regression Testing of Web Applications.
- Selenium supports MS Windows, Linux, Macintosh etc... operating environments.
- Selenium supports Mozilla Firefox, IE, Chrome, Safari etc... browsers.
- Selenium supports Java, .NET, PHP, Perl, Python and Ruby languages for enhancing Test cases, We can use any one the language.
- Most of the Selenium testers using Java Programming Language.

### **2) When Selenium was launched?**

- Selenium was first came to life in 2004.
- Selenium WebDriver was launched in 2006.
- In 2008, the whole Selenium Team decided to merge WebDriver and Selenium RC to form a more powerful tool called Selenium 2.

#### **Selenium 1**

Selenium IDE + Selenium RC + Selenium Grid

#### **Selenium 2**

Selenium IDE + Selenium RC + Selenium WebDriver + Selenium Grid

**Note:** Selenium RC is still being developed but only for maintenance.

- We can download Selenium Tools from [www.seleniumhq.org](http://www.seleniumhq.org) website.

### **3) What is Selenium License?**

All Selenium projects are licensed under the Apache 2.0 License, but It is Open source anybody can download and use selenium suite(Selenium IDE, Selenium RC, Selenium WebDriver and Selenium Grid) of tools with free of cost.

- Anybody can download and use Selenium tools.
- Anybody can update the source code and use.
- Selling updated code is not allowed, it is the primary objective of Apache License.

### **4) What are Selenium Tools or Selenium Components?**

- Selenium IDE
- Selenium RC
- Selenium Webdriver
- Selenium Grid

### **5) What is Selenium IDE?**

- Selenium IDE, IDE stands for Integrated Development Environment is a tool to create and execute test cases.
- It is a Firefox plugin and provides an easy-to-use interface for developing automated test cases.

#### **Features of Selenium IDE:**

- Record and Play back Test Cases.
- Execute Test Suites.
- Edit Test Scripts

#### **Drawbacks of Selenium IDE:**

- It supports Mozilla Firefox Browser only.
- Data Driven Testing (executing tests using multiple sets of test data) is not possible.
- Test Results are not generated using Selenium IDE (It generates summary only).
- Random Test Cases execution is not possible.
- Selenium IDE doesn't support Flow control Statements.

- It doesn't support programming (Conditional statements, loop statements etc...) for enhancing Test cases.
- It is not suitable for complex Test case design.
- It doesn't support random Test case execution.

## 6) What is Selenium RC?

- Selenium RC(Remote Control) solves the limitations of Selenium IDE.
- It supports various programming languages like Java, C#, PHP, Perl, Python and Ruby languages to enhance Test cases.

### Advantages of Selenium RC:

- It supports Random Test cases execution
- Supports Flow control statements and other programming features to enhance Tests.
- Supports various Browsers for Test case execution.

### Disadvantage of Selenium RC:

- Test Case execution time is more because every client ret first send to Selenium Server after that send to Browser.
- Executing Tests in parallel is not possible.

## 7) What is Selenium WebDriver?

- Selenium WebDriver fits in the same role as RC did and Webdriver overcomes the limitations of Selenium RC.
- It is a most important tool in Selenium suite.
- It has programming interface allows us to create and execute Test cases against different browsers (ex: Firefox, IE, Google Chrome etc...)
- WebDriver supports various programming languages to enhance Test cases.
- WebDriver supports various operating environments to create and execute Test cases(Ex: MS Windows, UNIX and Macintosh etc...)
- WebDriver supports Data Driven testing, Cross browser testing.

- Webdriver is faster than other tools of Selenium suite.
- WebDriver supports Parallel test execution with the help of TestNG.
- WebDriver doesn't have IDE, only Programming interface.
- WebDriver doesn't have built in Result reporting facility, it provides summary only. (\* with the help of TestNG we can generate HTML test reports.)
- No object Repository in selenium WebDriver (It is for entire Suite), so no centralized maintenance of Objects).
- Using Element locators, Webdriver methods and Programming features we can create and execute Test cases.

## **8) What is Selenium Grid?**

- Selenium Grid is only for Test case execution, not used for Test design.
- Selenium Grid allows us to run our Test cases in parallel, different tests can be run at the same time on different remote machines.

## **9) What are the differences between Selenium and UFT?**

Selenium and UFT (formerly QTP) both used for Functional and Regression Testing.

- Selenium supports Web Applications only whereas UFT supports Desktop and Web Applications.
- Selenium is an Open source software but UFT is a Vendor tool.
- Selenium supports MS Windows, Linux and Macintosh etc... operating environments but UFT supports Windows operating environment only.
- Selenium supports various programming and scripting languages (Java, .Net, PHP, Perl, Python and Ruby) for enhancing test cases but UFT supports VBScript only.

## **10) What are the Advantages of Selenium?**

- It is an Open source Software, so License cost.
- It supports various operating environments (MS Windows, Linux, Macintosh etc...) to create and execute Test cases.
- It supports various programming and scripting languages (Java, .Net, PHP, Perl, Python and Ruby) for enhancing test cases.

- It supports Cross Browser Testing.
- It supports Parallel Test Execution.

#### **11) What are the Disadvantages of Selenium?**

- It supports Web Applications Testing only, doesn't support Desktop Applications.
- Since it is an Open source software, no reliable technical support from anybody.
- New features may not work properly.
- No centralized maintenance of objects/elements.

#### **12) What are the Testing Frameworks used in Selenium?**

- JNuit, TestNG etc ... Testing frameworks used in Selenium.
- JNunit will help to execute Test Batches and generate Test Reports.
- TestNG framework is used to group Test cases, Execute Test suites, and generate Test Reports.

#### **13) What about Selenium Certification?**

- Since it is an open source software, No Authorized Certification for Selenium.
- Some organizations are conducting Selenium certification exam but no value guaranty for them in the IT Industry.

#### **14) How to Configure Selenium?**

First Choose Selenium Tools, Programming language and Testing framework for Test Automation.

Suppose we selected,

Selenium WebDriver for Test case design.

Java Programming language for enhancing Test cases.

TestNG framework for grouping test cases, executing Test suites and generating Test Reports.

Configure Selenium:

- Download and Install Java

- Download and extract Eclipse IDE to create and execute Java programs.
- Download and Install Firebug and Firepath (Firefox plug ins) to inspect Elements.
- No plug ins are required for IE and Google chrome browsers, they have built in developer tools.
- Download Selenium WebDriver Java language bindings from [www.seleniumhq.org](http://www.seleniumhq.org) web site.
- Add WebDriver jar files to Java Project in Eclipse IDE.
- Download TestNG software from Eclipse IDE and Install.

## **Interviews on Selenium Test Process**

### **1) What are the phases in Software Test Process?**

- Software Test phases may vary from one company to another and one project to another.
- As per Industry standards, there are four important phases in Software Test process, They are Test Planning, Test Design, Test Execution and Test Closure.

### **2) What are the phases in Selenium Test Process?**

- Test Planning, Generating basic Test cases, Enhancing Test Cases, Running & Debugging Test cases, Analyzing Test Results and Reporting Defects.

### **3) What are the important tasks in Test Planning phase?**

- Get AUT (Application Under Test) Environment details (UI design, and Database) from development team.
- Analyze the AUT in terms of UI elements Identification.
- Select Test cases for Automation
- Select Testing Framework (JUnit or TestNG) and configure.
- Configure Defect Reporting environment.

### **4) How to create Test cases using Selenium IDE?**

- Using Recording feature we can create Test cases in Selenium IDE.

- Using Element locators and Selenese commands we can type Test steps instead of using Recording feature.

### **5) How to create Test cases using Selenium WebDriver?**

- Using Element locators and WebDriver Methods we can create Test cases in Selenium WebDriver.

### **6) How to inspect Elements/objects?**

- If it is Mozilla Firefox Browser, using Firebug and Firepath plug ins we inspect Elements/Objects.
- If it is IE or Google Chrome browser, using built in Developer tools (F12) we can inspect elements.

### **7) How to enhance Selenium IDE Test cases?**

- Using Selenese Verification commands we can insert verification points.
- We can add comments

### **8) How to enhance Selenium WebDriver Test Cases?**

- By inserting verification points, Adding comments, Exception handling, Synchronization, Parametrization and using functions (reusable components) we can enhance Test cases.
- Using Java or other supported programming language features we can enhance Test cases.
- Using JUnit or TestNG Annotations also we can enhance WebDriver Test cases.

### **9) How to conduct Batch Testing?**

- In Selenium IDE, we can create Test suites and execute Test suites.
- Using either JUnit or TestNG, we can conduct Batch Testing in selenium Webdriver.

### **10) How to debug Test cases?**

- By step by step Test case execution we can locate errors.
- Debugging Test cases is optional task in Selenium Test process.

- Whenever Test case is not showing any errors and not providing correct output then we debug the Test case.

### **11) How to report Defects in selenium?**

- There is no Defect Tracking tool integrated with Selenium, after analyzing the Test Results then report defects using your company Defect management system.
- Selenium is nor responsible for Defect reporting, Tester has to report defects.

### **12) How to Select Test cases for Automation?**

No restricts for selecting Test cases for Automation, generally we select Three categories of test cases.

- Tests that we have to execute on every build (Sanity tests).
- Tests that we have to execute on every modified build(Regression Tests).
- Tests that we have execute using multiple sets of Test data (Data Driven Tests).

### **13) How to conduct Cross Browser Testing?**

- Selenium IDE doesn't support Cross browser Testing.
- Selenium WebDriver supports Cross browser Testing.
- Use Browser drivers to conduct Cross browser Testing.
- For Mozilla Firefox we no need to download browser driver, For IE, Google chrome and other browsers we have to download browser driver and set path.

### **14) How to conduct Parallel Testing?**

- Using multiple machines (Computers) and Selenium Grid we can conduct Parallel Test execution.

### **15) How to conduct Data driven Testing?**

- Using Text file or Excel file as resource we can conduct Data driven Testing.
- Using TestNG DataProvider Annotation also we can conduct Data Driven Testing.

## **Java for Selenium Interview s**

## **1) Why we prefer Java for Selenium?**

- Selenium written in Java, it doesn't mean that Java only more compatible with selenium, we can use other supported languages also.
- Good support for Selenium with Java, You can get more help documents and implementations from Internet.
- Majority of Selenium Testers(nearly 77%) using Java, so knowledge sharing is very easy.
- Java is platform independent language, we can use it on any Operating environment.

## **2) How much Java is required for Selenium?**

- For Test Automation using Selenium Core Java knowledge is sufficient, Advanced Java not required.
- Java Basics and Object Oriented Concepts are enough.
- Java Basics are Data Types, Variables, Modifiers, Operators, Flow Control statements, Functions, Comments etc...
- Java OOPS concepts are Abstraction, Polymorphism, Inheritance, and Encapsulation.

## **3) What is Java?**

- Java Programming Language was developed by Sun Microsystems in 1995, Now it is subsidiary of Oracle corporation.
- Java is an Object Oriented programming Language, In Java, everything is an Object. Java can be easily extended since it is based on the Object model.
- Java is a Platform independent Language. It can be compiled and interpreted.
- Java is Simple, It is easy to learn and implement.
- Java is Secureable, using Java we can develop virus free and tamper free systems.

## **4) How to Set up Java Environment for Windows Operating Environment?**

- Download Java (JDK) software and Install.
- Set Path Environment Variable to access Java from any directory.

- Download and extract Eclipse IDE to write and execute Java Programs.

For more information visit: <http://www.gcreddy.com/2015/09/java-environment-setup-for-selenium.html>

### **5) What are the Advantages of Java or usage of Java?**

- Java is used to Develop Desktop Applications (Ex: Acrobat Reader)
- Java is used to Develop Web Applications.
- Java is used to Develop Enterprise Applications (ex: banking, Insurance Applications).
- Java is used to Develop Mobile applications.
- Java is used to Develop Embedded systems.
- Java is used to Develop Smart cards.
- Java is used to Develop Games Software etc...

### **6) What are the Java language Syntax Rules?**

- Java is case sensitive language(deference is there in between upper and lower case letters)
- First letter of a Class Name should be in upper case.
- Method names should start with lower case letter
- Java Program file name should exactly match with class name.
- public static void main (String args[])]- Java program execution starts from main method, which is mandatory in every java program.
- Every statement should end with semi colon symbol.

### **7) What is the primary purpose of Comments?**

To make the code readable.

### **8) What is Data Type?**

Data type is a classification of the type of data that a variable or object can hold in Computer programming.

### **9) What are the 8 primitive data types in Java?**

- Integer Data types
  - i) byte (8 bits)
  - ii) short (16 bits)
  - iii) int (32 bits)
  - iv) long (64 bits)
- Rational Data types (Numbers with decimal places)
  - v) float (32 bits)
  - vi) double (64 bits)
- Characters
  - vii) char (16 bits)
- Conditional
  - viii) Boolean

## **9) What are Java Modifiers?**

- Modifiers are keywords that we add to those definitions to change their meanings.
- Java has 2 types of modifiers, One is Access Modifiers another is Non-Access Modifiers.
- Access Modifiers are public, private, default and protected.
- Non-Access Modifiers are static, final, abstract etc...

## **10) What are the types of Variables in Java?**

Java has 3 types of variables:

- i) Instance variables  
(A variable that is declared inside the class but outside the method.)
- ii) Local variables  
(A variable that is declared inside the Method.)
- iii) Static / class variables  
(A variable that is declared as static, It cannot be local.)

## **11) What are the categories of operators in Java?**

Operators are used to perform mathematical, Comparison and Logical operations.

Important categories of Operators in Java:

- i) Assignment Operators
- ii) Arithmetic operators
- iii) Relational operators

iv) Logical Operators  
Etc...

## **12) What are the Conditional statements in Java?**

Types of Conditional statements in Java:

- i) if statement
- ii) switch statement

## **13) What are the Loop statements in Java?**

Whenever we want execute a block of statements multiple times then we use Loop structures.

There are four types of Loop structures in Java.

- i) for loop
- ii) while loop
- iii) do while loop
- iv) Enhanced for loop

## **14) What is String?**

- String is a sequence of characters written double quotes.
- String may have Alfa bytes, numbers and special characters.

## **15) Explain about Arrays in Java?**

- Java Array is an Object that holds a fixed number of values of a single data type.
- The length of an Array is established when the Array is created.
- Array length is fixed, Java Array has Zero based index.

## **16) What are Java Methods?**

- Method is a set of statements to perform an Operation.
- Methods also known as Procedures or Functions.
- In Structure programming we use Functions (Built in and User defined)
- In Object Oriented programming we use Methods (Built in and User defined)

## **17) What is Exception handling in Java?**

- An exception is an event, which occurs during execution of a program.
- Exception handling is mechanism to handle exceptions.

## **18) What is Inheritance?**

- It is a process of Inheriting (reusing) the class members (Variables and Methods) from one class to another class is called Inheritance.
- Non static (Object Level) class members only can be inherited.
- The class where the class members are getting inherited is called as Super class / parent class / Base class.
- The class to which the class members are getting inherited is called Sub class / Child class / Derived class.
- The inheritance between Super class and Sub class is achieved using "extends" keyword.

## **19) What is Polymorphism?**

Polymorphism means, existence of Object behavior in many forms.

There are two types of Polymorphism in Java:

- i) Compile Time Polymorphism / Static binding / Method overloading
- ii) Run Time Polymorphism / Dynamic binding / Method overriding

## **20) What is Abstraction?**

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

## **21) What is Java Interface?**

- Interface is a Java type definition block which is 100% abstract.
- All the Interface methods are by default public and abstract.
- Static and final modifiers are not allowed for interface methods.
- In Interfaces variables have to initialize at the time of declaration.

## **22) What is Encapsulation?**

It is a process of wrapping code and data into a single unit.

Encapsulation is the technique making the fields in a class private and providing access via public methods

- It provides us control over the data.
- By providing setter and getter methods, we can make a class read only or write only.
- If we don't define setter method then read only.

## Selenium WebDriver Interview s

### 1) Give a detailed introduction about Selenium WebDriver?

**Selenium 1** (Selenium IDE + selenium RC + Selenium Grid)

**Selenium 2** (Selenium IDE + Selenium RC + Selenium WebDriver + Selenium Grid)

**Note:** Selenium WebDriver merged with Selenium 1 and called as Selenium 2.

- It is a most important tool in Selenium suite.
- It has programming interface allows us to create and execute Test cases against different browsers (ex: Firefox, IE, Google Chrome etc...)
- WebDriver supports various programming languages(Java, .Net, PHP, Perl, Python and Ruby) to enhance Test cases.
- WebDriver supports various operating environments (MS Windows, UNIX and Macintosh etc...) to create and execute Test cases.
- WebDriver supports Data Driven testing, Cross browser testing.
- Webdriver is faster than other tools in Selenium suite.
- WebDriver supports Parallel test execution with the help of TestNG.
- WebDriver doesn't have IDE, only Programming interface.
- WebDriver doesn't have built in Result reporting facility, it provides summary only. (\* with the help of TestNG we can generate HTML test reports.)
- No object Repository in selenium WebDriver (This limitation is for entire Suite), so no centralized maintenance of Objects).

- Using Element locators, Webdriver methods and Java programming features we can create and execute Test cases.

## **2) How to set up Selenium WebDriver Environment?**

- Download and install Java (JDK) software - to enhance test cases using Java programming features.
- Set Path environment variable- to access Java from any directory.
- Download Eclipse IDE and extract
- Download WebDriver Java Language binding and add Webdriver jar files (in Eclipse)
- Install Firebug and FirePath plug ins for Mozilla Firefox browser to inspect Elements.

**Note:** For Internet Explorer and Google chrome, no need to install any plug ins, They have built in Developer tools.

**Note 2:** Element locators and WebDriver methods are common for all browsers, browser driver only varies from one browser to another.

**Note 3:** Firefox driver is default driver in Webdriver, For IE and Chrome we need to download drivers.

Download Selenium WebDriver Java binding from [www.seleniumhq.org](http://www.seleniumhq.org) website and extract.

### **Navigation for adding Webdriver jar files in Eclipse.**

Create Java project in Eclipse

- > Select src and right click
- > Build path
- > Configure Build Path
- > select Libraries tab
- > Click "Add External JARs"
- > Browser path of the WebDriver jars
- > Add

### **3) How to create Test cases using Selenium WebDriver?**

- Using Element locators, WebDriver methods and Java programming features we can create Test cases.
- Element locators for recognizing objects/elements.
- WebDriver Methods are used perform operations on Elements or objects.
- Java Programming for enhancing Test cases.

### **4) What are the Testing frameworks support Selenium WebDriver with Java?**

JUnit

TestNG

We can use either JUnit or TestNG testing framework with Java and Webdriver

### **5) What are the Element Locators that Selenium WebDriver supports to recognize Elements?**

Selenium Webdriver supports 8 Element locators to recognize Objects/Elements.

- i) id
- ii) name
- iii) className
- iv) tagName
- v) linkText
- vi) partialLinkText
- vii) cssSelector
- viii) xpath

Note: We can use any one unique locator to recognize the Object/Element.

### **6) What is Cross Browser Testing?**

Testing web applications using multiple browsers, Cross browser testing involves checking compatibility of the application.

## **7) How to conduct Cross Browser Testing using WebDriver?**

- Using Browser Drivers we can conduct Cross browser testing.
- For Mozilla Firefox, we no need to download the browser driver.
- For IE and Chrome etc... browsers download the browser drivers and set path.

Create Browser Driver Object for Mozilla Firefox:

```
WebDriver driver = new FirefoxDriver();
```

Create Browser Driver Object for Internet Explorer:

```
System.setProperty("webdriver.ie.driver", "E:/IEDriverServer.exe");
WebDriver driver = new InternetExplorerDriver();
```

Create Browser Driver Object for Google Chrome:

```
System.setProperty("webdriver.chrome.driver", "E:/chromedriver.exe");
WebDriver driver = new ChromeDriver();
```

## **8) What are the important operations on Browser object?**

Operation: Open URL

```
WebDriver code: driver.get("http://google.com");
```

Operation: Return Browser Title

```
WebDriver code: String s = driver.getTitle();
```

Operation: Return Current URL

```
WebDriver code: String s = driver.getCurrentUrl();
```

Operation: Close focused Browser

```
WebDriver code: driver.close
```

Operation: Close all Browsers that opened by WebDriver

```
WebDriver code: driver.quit
```

## **9) What are the Elements/Objects in Web Applications?**

Link

Button

Image, Image Link, Image Button

Text box

Edit Box

Text Area

Check box

Radio Button  
Drop down box  
List box  
Combo box  
Web table /HTML table  
Frame  
Etc...

## **10) Write a Test case using Selenium WebDriver?**

Manual Test Case:

Test Case Name: Admin Login

Steps:

- i) Launch the Browser and navigate to www.gcrit/build3/admin/
- ii) Enter user name
- iii) Enter Password
- iv) Click Login button

Verification point:

---

Check the existence of "Logoff" link, if exists then pass otherwise fail

Selenium Test Case:

```
WebDriver driver = new FirefoxDriver();
driver.get("http://www.gcrit.com/build3/admin/");
driver.findElement(By.name("username")).sendKeys("admin");
driver.findElement(By.name("password")).sendKeys("admin@123");
driver.findElement(By.id("fdb1")).click();

if (driver.findElement(By.linkText("Logoff")).isDisplayed()){
    System.out.println("Login Successful");
}
else {
    System.out.println("Login Failed");
}
```

## **11) How to conduct Batch Testing using WebDriver?**

Using TestNG Annotations we can group Test cases and execute series of Test cases.

## **12) How to handle Link element?**

Operation: Click Link

```
WebDriver code: driver.findElement(By.linkText("Gmail")).click();
```

Operation: Check the link existance

WebDriver code:

```
boolean a =  
driver.findElement(By.xpath(".//*[@id='gbw']/div/div/div[1]/div[1]/a")).isDisplayed();  
System.out.println(a); //true
```

Operation: Check the link enabled status

WebDriver code:

```
boolean b =  
driver.findElement(By.xpath(".//*[@id='gbw']/div/div/div[1]/div[1]/a")).isEnabled();  
System.out.println(b); //true
```

Operation: Return the Link Name

WebDriver code:

```
String c = driver.findElement(By.xpath(".//*[@id='gbw']/div/div/div[1]/div[1]/a")).getText();  
System.out.println(c); //Gmail
```

## **13) How to handle Button element?**

### **Important operations on Buttons:**

Enabled status

Display status

Click

Return name of the Button

type of the object

Example:

```
public static void main(String[] args) {  
    WebDriver driver = new FirefoxDriver();  
    driver.get("https://gmail.com");  
    WebElement button = driver.findElement(By.id("next"));  
    boolean a = button.isDisplayed();  
    boolean b = button.isEnabled();  
    button.getAttribute("name");  
    button.getAttribute("type");  
    button.click();  
    System.out.println(a);
```

```
System.out.println(b);
}
```

#### 14) How to handle Edit box element?

##### Important Operations on Edit box:

Enter a Value,

Clear the Value,

Check enabled status,

Check edit box existence,

Get the value etc...

---

##### Example:

```
public static void main(String[] args) {
    WebDriver driver = new FirefoxDriver();
    driver.get("https://gmail.com");
    //Enter a Value
    driver.findElement(By.id("Email")).sendKeys("ABCD123");

    //Return the Value
    String value = driver.findElement(By.id("Email")).getAttribute("name");

    //Return Type of the Object
    String value2 = driver.findElement(By.id("Email")).getAttribute("type");

    //Return display status
    boolean a = driver.findElement(By.id("Email")).isDisplayed();

    //Return Enabled status
    boolean b = driver.findElement(By.id("Email")).isEnabled();

    System.out.println(value);
    System.out.println(value2);
    System.out.println(a);
    System.out.println(b);
    // Clear the Value
    driver.findElement(By.id("Email")).clear();
}
```

#### 15) How to handle Images?

### **Three types of Image elements in Web Environment**

- 1) General Image (No functionality)
- 2) Image Button (Submits)
- 3) Image Link (Redirects to another page/location)

#### **Example:**

```
public static void main(String[] args) {  
    WebDriver driver = new FirefoxDriver();  
    driver.get("https://google.com");  
    String Title = driver.findElement(By.id("hplogo")).getAttribute("Title");  
    System.out.println(Title);  
  
    driver.navigate().to("http://newtours.demoaut.com/");  
    driver.findElement(By.name("login")).click();  
  
    driver.navigate().to("http://www.seleniumhq.org/");  
    driver.findElement(By.xpath("//*[@id='choice']/tbody/tr/td[2]/center/a/img")).click();  
}
```

### **16) How to handle Radio Button element?**

#### **Operations on Radio Button**

- i) Select Radio Button
- ii) Verify if the Radio Button is Displayed or not?
- iii) Verify if the Radio Button is enabled or not?
- iv) Verify if the Radio Button is Selected or not?

---

#### **Example:**

```
public static void main(String[] args) {  
    WebDriver driver = new FirefoxDriver();  
    driver.get("http://www.gcrit.com/build3/create_account.php?osCsid=boh81f11tmud1134jn  
p5ne7r20");  
    boolean a =  
    driver.findElement(By.xpath("//*[@id='bodyContent']/form/div/div[2]/table/tbody/tr[1]/td  
[2]/input[1]")).isDisplayed();  
    System.out.println(a); //true  
  
    boolean b =  
    driver.findElement(By.xpath("//*[@id='bodyContent']/form/div/div[2]/table/tbody/tr[1]/td  
[2]/input[1]")).isEnabled();  
    System.out.println(b); //true
```

```

boolean c =
driver.findElement(By.xpath(".//*[@id='bodyContent']/form/div/div[2]/table/tbody/tr[1]/td[2]/input[1]")).isSelected();
System.out.println(c); //false

driver.findElement(By.xpath(".//*[@id='bodyContent']/form/div/div[2]/table/tbody/tr[1]/td[2]/input[1]")).click();

boolean d =
driver.findElement(By.xpath(".//*[@id='bodyContent']/form/div/div[2]/table/tbody/tr[1]/td[2]/input[1]")).isSelected();
System.out.println(d); //true

driver.findElement(By.xpath(".//*[@id='bodyContent']/form/div/div[2]/table/tbody/tr[1]/td[2]/input[1]")).clear();
}

```

## **17) How to handle Check box element?**

### **Operations on Check box**

- i) Check if the check box is displayed or not?
- ii) Check if the check box is enabled or not?
- iii) Check if the check box is Selected or not?
- iv) Select the Check box
- v) Unselect the Check box

### **Example:**

```

public static void main(String[] args) {
WebDriver driver = new FirefoxDriver();
driver.get("file:///E:/HTMLExamples/MultipleCheckbox.html");

boolean display = driver.findElement(By.xpath("html/body/input[2]")).isDisplayed();
System.out.println("Displayed Status: " + display);

boolean enabled = driver.findElement(By.xpath("html/body/input[2]")).isEnabled();
System.out.println("Enabled Status: " + enabled);

boolean check = driver.findElement(By.xpath("html/body/input[2]")).isSelected();
System.out.println("Check Status: " + check);

driver.findElement(By.xpath("html/body/input[2]")).click();

```

```

boolean check2 = driver.findElement(By.xpath("html/body/input[2]")).isSelected();
System.out.println("Check Status: " + check2);

driver.findElement(By.xpath("html/body/input[2]")).click();

boolean check3 = driver.findElement(By.xpath("html/body/input[2]")).isSelected();
System.out.println("Check Status: " + check3);

}

```

### **18) How to handle Frames?**

- HTML frames are used to divide our Browser window into multiple sections, where each section can load a separate html document.
- Frames are sections of Web page displayed on top window.
- Whenever we access the page then focus is on the top window.

**Switch to a frame is done in two ways:**-----

#### **i) Using frame index**

Syntax:

```
driver.switchTo.frame(int index);
```

#### **ii) Using frame name**

Syntax:

```
driver.switchTo.frame(String name);
```

Example:

```
//Using Frame index
```

```

public static void main(String[] args) {
    WebDriver driver = new FirefoxDriver();
    driver.get("http://seleniumhq.github.io/selenium/docs/api/java/index.html");
    driver.switchTo().frame(0);
    driver.findElement(By.xpath("html/body/div[2]/ul/li[1]/a")).click();
}
-----
```

```
//Using Frame name
```

```

public static void main(String[] args) {
    WebDriver driver = new FirefoxDriver();
    driver.get("http://seleniumhq.github.io/selenium/docs/api/java/index.html");
    driver.switchTo().frame("packageListFrame");
    driver.findElement(By.xpath("html/body/div[2]/ul/li[1]/a")).click();
}
```

---

**Switch from a frame to Top window:**

```
driver.switchTo().defaultContent();
```

Example:

```
public static void main(String[] args) {  
    WebDriver driver = new FirefoxDriver();  
    driver.get("http://seleniumhq.github.io/selenium/docs/api/java/index.html");  
  
    //Switch from Top window to 3rd frame  
    driver.switchTo().frame(2);  
    driver.findElement(By.xpath("html/body/div[3]/table/tbody[2]/tr[1]/td[1]/a")).click();  
    //Switch from 3rd frame to Top window  
    driver.switchTo().defaultContent();  
    //Switch from Top window to 1st frame  
    driver.switchTo().frame(0);  
    driver.findElement(By.xpath("html/body/div[2]/ul/li[1]/a")).click();  
}  
}
```

**19) How to handle Web Table?****Important Operations on Web Table:**

Get cell value

Rows Count

Cells Count

**Example:**

```
public static void main(String[] args) {  
    WebDriver driver = new FirefoxDriver();  
    driver.get("file:///E:/HTMLExamples/htmlTable.html");  
  
    String s = driver.findElement(By.xpath(".//*[@id='students']/tbody/tr[2]/td[2]")).getText();  
    System.out.println(s);  
  
    WebElement htmlTable = driver.findElement(By.id("students"));  
  
    List <WebElement> rows = htmlTable.findElements(By.tagName("tr"));  
    int i = rows.size();  
    System.out.println(i);  
  
    List <WebElement> cells = htmlTable.findElements(By.tagName("td"));  
    int j = cells.size();  
    System.out.println(j);  
}
```

## **20) How to handle Multiple Browsers?**

We can Handle Multiple Browsers using Browser window handle.

## **21) How to handle duplicate elements/objects?**

We can Handle duplicate objects/elements using index, index starts from 0 to 9 and Top left of the web page to Bottom right.

## **TestNG Interview s**

### **1) What is TestNG?**

TestNG is a testing framework inspired from JUnit and NUnit but introducing some new functionalities that make it more powerful and easier to use.

### **2) What is TestNG in Selenium?**

Using TestNG framework in Selenium we can,

- > Generate Detailed (HTML) Test Reports.
- > Group Test cases.
- > Parallel Test execution.
- > Parameterize Tests
- > Execute multiple classes / programs using XML file etc...

### **3) How to install TestNG in Eclipse?**

In Eclipse IDE,

Help menu -> Install New Software -> Click Add

- > Enter name as "TestNG"
- > Enter url as: "<http://beust.com/eclipse/>"
- > Select "TestNG"
- > Next > Next > accept the Agreement > finish

### **4) Give a TestNG Program example?**

```
public class TestNGexample {
```

```
@Test  
public void verifyTitle() {  
    WebDriver driver = new FirefoxDriver();  
    driver.get("http:\\\\gmail.com");  
    String Actual = driver.getTitle();  
    Assert.assertEquals(Actual, "Gmail");  
}  
}
```

**Note:**

- > main method is not required for TestNG programs.
- > TestNG program contains methods that contain @ annotations.
- > If we don't write @Test Annotation then the method won't be executed.

**5) What are the important TestNG Annotations?**

Important TestNG Annotations are:

@Test - Annotation for every Test (Method)

@BeforeMethod - pre-condition for every test case in a program.

@AfterMethod - Post condition for every test case in a program.

@BeforeClass - pre condition for all test cases in a program/class

@AfterClass - post condition for all test cases in a program/class

@BeforeTest - pre condition for all test cases in multiple classes/programs

@AfterTest - post condition for all test cases in multiple classes / programs

**6) How to create multiple Test cases in a program/class?**

```
public class TestNGexample {
```

```
    @Test
```

```
    public void testA () {
```

```
        Assert.assertEquals("Gmail", "Gmail");
```

```
    }
```

```
    @Test
```

```
    public void testC() {
```

```
        Assert.assertEquals("Gmail", "Gmail");
```

```
    }
```

```
    @Test
```

```
    public void testB() {
```

```
        Assert.assertEquals("abc", "abc");
```

```
}
```

```
}
```

As per TestNG program:

A

C

B

---

Execution Flow:

A

B

C

---

If you want to control the Test execution flow then use "priority" Attribute.

Syntax:

```
@Test (priority = number")
```

Example:

```
public class TestNGexample {  
  
    @Test (priority=1)  
    public void testA () {  
        Assert.assertEquals("Gmail", "Gmail");  
    }  
  
    @Test (priority=2)  
    public void testC() {  
        Assert.assertEquals("Gmail", "Gmail");  
    }  
  
    @Test(priority=3)  
    public void testB() {  
        Assert.assertEquals("abc", "abc");  
    }  
}
```

## 7) How to execute multiple program or classes?

Using XML file we can execute multiple Java programs or classes at a time.

## 8) How to create XML file?

Create XML file

Select Java Project and Right click

- > New
- > Other
- > Enter XML and Select XML file
- > Enter File Name
- > Finish

### 9) Give an example for executing multiple programs or classes?

XML File

```
-----  
<?xml version="1.0" encoding="UTF-8"?>  
<suite name = "Ecommerce suite">  
  <test name ="Sanity Tests">  
    <classes>  
      <class name="javaExamples.Class1"/>  
      <class name="javaExamples.Class2"/>  
  
    </classes>  
  </test>  
</suite>  
-----
```

Class 1:

```
public class Class1 {  
    @BeforeClass  
    public void login(){  
        System.out.println("Login Successful");  
    }  
    @AfterClass  
    public void logout(){  
        System.out.println("Logout Successful");  
    }  
    @Test (priority=1)  
    public void search(){  
        System.out.println("Search Successful");  
    }  
    @Test(priority=2)  
    public void advancedSearch(){  
        System.out.println("Advanced Search Successful");  
    }  
    @Test(priority=3)  
    public void buyProducts(){  
        System.out.println("Buying Products Successful");  
    }  
}
```

```
}
```

---

Class 2:

```
public class Class2 {  
    @BeforeClass  
    public void login(){  
        System.out.println("Login Successful");  
    }  
    @AfterClass  
    public void logout(){  
        System.out.println("Logout Successful");  
    }  
    @Test  
    public void accountSummary(){  
        System.out.println("Account Summary Successful");  
    }  
    @Test  
    public void fundTransfer(){  
        System.out.println("Fund Transfer Successful");  
    }  
    @Test  
    public void billPayment(){  
        System.out.println("Bill Payment Successful");  
    }  
}
```

---

**10) Give an example for parallel Test execution?**

XML File

```
<suite name = "Parallel Test suite" parallel = "classes" thread-count = "2">  
    <test name = "Sanity Test">  
        <classes>  
            <class name = "seleniumTests.Class1"/>  
            <class name = "seleniumTests.Class2"/>  
        </classes>  
    </test>  
</suite>
```

---

parallel = "methods": TestNG will run all the methods in separate threads.  
parallel = "classes" : TestNG will run all the methods in the same class in the same thread.  
parallel = "tests" : TestNG will run all the methods in the same <test> tag in the same thread.

---

Class 1

```
public class Class1 {  
    @BeforeTest  
    public void login(){  
        System.out.println("Login Successful");  
    }  
    @AfterTest  
    public void logout(){  
        System.out.println("Logout Successful");  
    }  
    @Test (priority = 1)  
    public void search(){  
        System.out.println("Search Successful");  
    }  
    @Test (priority = 2)  
    public void advancedSearch(){  
        System.out.println("Advanced Search Successful");  
    }  
    @Test (priority = 3)  
    public void buyProducts(){  
        System.out.println("Buying Products Successful");  
    }  
    @Test (priority = 4)  
    public void testCase(){  
        System.out.println("Test Case in "+getClass().getSimpleName()  
        + " With Thread id: " + Thread.currentThread().getId());  
    }  
}
```

---

## Class 2

```
public class Class2 {  
    /*@BeforeClass  
    public void login(){  
        System.out.println("Login Successful");  
    }  
    @AfterClass  
    public void logout(){  
        System.out.println("Logout Successful");  
    }*/  
    @Test (priority = 1)  
    public void accountSummary(){  
        System.out.println("Account Summary Successful");  
    }  
    @Test (priority = 2)  
    public void fundTransfer(){  
        System.out.println("Fund Transfer Successful");  
    }  
}
```

```

    @Test (priority = 3)
    public void billPayment(){
        System.out.println("Bill Payment Successful");
    }
    @Test (priority = 4)
    public void testCase(){
        System.out.println("Test Case in "+getClass().getSimpleName()
        + " With Thread id: " + Thread.currentThread().getId());
    }
}
-----
```

### **11) Give an Example for grouping Test cases?**

XML File

```

<suite name ="suite">
<test name ="test">

    <groups>
        <run>
            <include name = "sanity"/>
        </run>
    </groups>

    <classes>
        <class name = "seleniumTests.GroupTests"/>
        <class name = "seleniumTests.GroupTests2"/>
    </classes>

</test>
</suite>
```

-----  
Java Program/Class file

```

@BeforeTest (groups ={"sanity","regression"})
    public void login(){
        System.out.println("Login Successful");
    }
    @AfterTest (groups ={"sanity","regression"})
    public void logout(){
        System.out.println("Logout Successful");
    }

    @Test (groups ={"sanity"})
    public void search(){
        System.out.println("Search Successful");
    }
    @Test (groups ={"sanity","regression"})
```

```

public void advancedSearch(){
    System.out.println("Advanced Search Successful");
}
@Test (groups ={"sanity", "regression"})
public void buyProducts(){
    System.out.println("Buying Products Successful");
}
@Test (groups ={"regression"})
public void abcd(){
    System.out.println("Abcd Successful");
}
@Test (groups ={"regression"})
public void xyza(){
    System.out.println("Xyza Successful");
}
@Test (groups ={"regression"})
public void asdf(){
    System.out.println("Asdf Successful");
}
}
-----
```

## **12) Give an example for Data Driven Testing?**

If you want work with excel, then download third party jar files

Ex: jxl

```

public class DataDriven {
    @Test (dataProvider="testdata")
    public void Addition(String val1, String val2, String val3){
        int a = Integer.parseInt(val1);
        int b = Integer.parseInt(val2);
        int c = Integer.parseInt(val3);
        int result = a + b + c;
        System.out.println(result);
    }
}

@DataProvider(name="testdata")
public Object [][] readExcel() throws BiffException, IOException {
    File f = new File("C:/Users/gcreddy/Desktop/Input.xls");
    Workbook w = Workbook.getWorkbook(f);
    Sheet s = w.getSheet(0);
    int rows = s.getRows();
    int columns = s.getColumns();
    //System.out.println(rows);
    //System.out.println(columns);
```

```

String Inputdata [] [] = new String [rows] [columns];
for (int i = 0; i<rows; i++){
    for (int j = 0; j<columns; j++){
        Cell c = s.getCell(j, i);
        Inputdata [i] [j] = c.getContents();
        //System.out.println(Inputdata[i][j]);
    }
}
return Inputdata;
}
}

```

## **2. Tell me some TestNG Annotations.**

@Test,@Parameters,@Listeners,@BeforeSuite,@AfterSuite,@BeforeTest,@AfterTest,  
 @DataProvider,@BeforeGroups,@AfterGroups,@BeforeClass,@AfterClass,  
 @BeforeMethod,@AfterMethod,@Factory

## **3. What are desiredcapabilities?**

Desired Capabilities help to set properties for the Web Driver. A typical use case would be to  
 set the path for the Firefox Driver if your local installation doesn't correspond to the default settings.

## **5. Difference between Web driver listener and TestNG Listener.**

TestNG and Web driver Listener have different interfaces to implement and call them. They both  
 modify respective behaviour. You can use Listeners in Annotation. Below 2 URL gives the detailed list of listener and their interfaces.

<http://testng.org/doc/documentation-main.html#testng-listeners>

<http://selenium.googlecode.com/git/docs/api/java/org/openqa/selenium/support/events/AbstractWebDriverEventListener.html>

## **9. What are the features of TestNG?**

TestNG is a testing framework designed to simplify a broad range of testing needs, from unit testing (testing a class in isolation of the others) to integration testing (testing entire systems made of several classes, several packages and even several external frameworks, such as application servers). You can use test suite, annotations, automatically generation of report and much more.

**11. In what situation selenium finding element get fails?**

- Element loading issue
- Dynamic id of web element

**12. What is the difference between "GET" and "NAVIGATE" to open a web page in selenium web driver?**

Get method will get a page to load or get page source or get text that's all whereas navigate will guide through the history like refresh, back, forward. For example if we want to move forward and do some functionality and back to the home page this can be achieved through navigate() only. driver.get will wait till the whole page gets loaded and driver.navigate will just redirect to that page and will not wait

**14. How we can retrieve the dynamically changing Ids?**

When we login Facebook the login label's id changes dynamically thus resulting in failure.

We have a hierarchy of locators and Facebook is dynamic in nature, so we are not able to use "id" for identification for after that we have remaining 7 locator's for that :2. xpath ().. 3. name..4. css.. 5. link text.. 6. partiallinktext...7.tag name. so u can use any one for identifying it. Most probably u can use "xpath" or "css-locator" and if there r tag then link text or partial-link text. it depend on u . But we never use id's in Ajax application because it's not possible.

**15.What is the difference between driver.Close() and driver.Quit () method?**

Close() - It is used to close the browser or page currently which is having the focus.

Quit() - It is used to shut down the web driver instance or destroy the web driver instance (Close all the windows)

**17. What is the basic use of Firefox profiles and how can we use them using selenium?**

A profile in Firefox is a collection of bookmarks, browser settings, extensions, passwords, and history; in short, all of your personal settings.

We use them to change user agent, changing default download directory, changing versions etc.

<http://code.google.com/p/selenium/wiki/FirefoxDriver>

## **18. Customize the name of file going to be downloaded?**

You have to download AUTO IT.exe file and has to be install

and later you have create .au3 file (in this file you have to specify the commands in VB script like your file name, where have to save, it will be easy may be 3 or 4 steps ) using AUTOIT...then right click the .au3 file you have to compile ....after that you will get the .exe file with the name of .au3 file ..In eclipse you will give the code like this

```
<----ProcessBuilder ps = new ProcessBuilder("path of the .exe file of au3") .start();--->
```

## **19. How to handle internationalisation through web driver?**

```
FirefoxProfile profile = new FirefoxProfile();
profile.setPreference("intl.accept_languages","jp");
```

Web driver driver = new FirefoxDriver(profile); driver.get(google.com) will open google in Japanese Lang

## **20. How to overcome same origin policy through web driver?**

- Proxy server.

```
DesiredCapabilities capability=new DesiredCapabilities.firefox();
capability.setCapability(CapabilityType.PROXY,"your desire proxy")
WebDriver driver=new FirefoxDriver(capability);
```

## **21. How to put text in Facebook search box using selenium web driver.**

- driver.findElement(By.xpath("//div[contains(@class, '\_586i')]")).sendKeys("abc");

## **22. Difference between flex and flash application.**

In flash there is no code just based on creativity(design) we will complete the work(time consuming process) whereas flex contain some small functions which is integrated with mxml,PHP..(no tool is there to develop

flex we want to use the properties of css and style sheet)

### **23. What is Error Collector in TestNG? What is its use?**

This class allows the collection of errors during the process of retrieving the test data for the test method parameters

<http://testngdatabind.sourceforge.net/apidocs/net/sf/testng/databinding/core/error/ErrorCollector.html>

### **24. How can we get the font size, font color, font type used for a particular text on a web page using Selenium web driver?**

```
driver.findElement(By.XPath("Xpath ")).GetCssValue("font-size");
driver.findElement(By.XPath("Xpath ")).GetCssValue("font-colour");
driver.findElement(By.XPath("Xpath ")).GetCssValue("font-type");
driver.findElement(By.XPath("Xpath ")).GetCssValue("background-colour");
```

### **26. How to prepare Customized html Report using TestNG in hybrid framework.**

Below are the 3 ways:

- Junit: with the help of ANT.
- TestNG: using inbuilt default.html to get the HTML report. Also XST reports from ANT, Selenium, TestNG combination.
- Using our own customized reports using XSL jar for converting XML content to HTML.

### **27. "What's the hierarchy of TestNG annotations?**

Explain me about annotation hierarchy & execution order?

[Click Here for](#)

### **29. Is it possible test web services using selenium?**

Using Jmeter we can test how one website is talking to each other means time taken to send data, feeds, messages from one website to other website. Jmeter does a nice job of doubling for performance and api tests.

### **30. How to refresh a page without using context click?**

- 1.Using sendKeys(Keys method
- 2.Using navigate.refresh() method
- 4.Using get() method
- 5.Using sendKeys() method

[Click Here for Detailed](#)

### **31. Can u send a code for printing in selenium?**

There are two cases:

**Case1.** Any hyperlink/button on a web page, n clicking that link/button a print dialog box opens. (Performing an action on web page)

**Case2.** or do u want to open print dialog box within ur own script, not by performing any action on web page.

So If Case 1: just a call for WebElement.click() event will work to open it.

If Case 2: Call a Printer Job object (Use Awt API).

For code: Google it.

<http://code.google.com/p/selenium/issues/detail?id=1815>

### **32. How to find broken images in a page using Selenium Web driver.**

1. Get xpath and then using tag name; get all the links in the page
2. Click on each and every link in the page
3. In the target page title, look for 404/500 error.

[Click Here for Sample Program Code](#)

### **33. How to handle Ajax popup window?**

By using getWindowHandles() and obj.switchTo.window(windowid) we can handle popups using explicit wait and driver.switchTo.window("name") commands for your requirements.

### **36. How to handle colors in web driver?**

Use getCssValue(arg0) function to get the colors by sending 'color' string as an argument.

Example

```
String col = driver.findElement(By.id(locator)).getCssValue("color");
```

### **38. How to get text from captcha image??**

```
driver.findElement(By.xpath(".///*[@id='SkipCaptcha']")).click();
```

```
String attr = ie.findElement(By.xpath(".///*[@id='SkipCaptcha']")).getAttribute("value");
```

```
System.out.println("The value of the attribute 'Name' is " + attr);
```

### **39. Is there a way to click hidden LINK in web driver?**

```
String Block1 = driver.findElement(By.id("element ID"));
```

```
JavascriptExecutor js1=(JavascriptExecutor)driver;
```

```
js1.executeScript("${"+Block1+"}.css({'display':'block'});");
```

### **40. What Class Extends Web Driver?**

AndroidDriver, ChromeDriver, EventFiringWebDriver, FirefoxDriver, HtmlUnitDriver, InternetExplorerDriver, iPhoneDriver, PhantomJSDriver, RemoteWebDriver, SafariDriver

### **41. What are the APIs that support Web Driver?**

API are nothing but collection of all selenium commands for Locating UI Elements (WebElements), Fetching a Page, User Input etc...

### **42. How to disable cookies in browser.**

Using deleteAllVisibleCookies() in selenium

### **44. How to change user agent in Firefox by selenium web driver.**

```
FirefoxProfile profile = new FirefoxProfile();
```

```
profile.setPreference("general.useragent.override", "some UA string");
```

```
Web Driver driver = new FirefoxDriver(profile);
```

### **45. What is Selenese?**

Selenese is HTML language based command, which is used in Selenium IDE.

**47. What is the MOST challenging test problem in my career in Automation?**

In my career

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

**48. "Suppose developer changed the existing image to new image with same xpath. Is test case pass or fail?" Pass**

**49. How to handle network latency using selenium?**

- Using driver.manage.pageLoadingtime for network latency

**50. How does u handle dynamic elements without using xpath (with example?)**

- By using classname or css.

**51. What are the different types of driver implementation?**

AndroidDriver, AndroidWebDriver, ChromeDriver, EventFiringWebDriver, FirefoxDriver, HtmlUnitDriver, InternetExplorerDriver, iPhoneDriver, iPhoneSimulatorDriver, RemoteWebDriver, SafariDriver, WebDriverBackedSelenium

**53. Which repository you have used to store the test scripts?**

I have created scripts in excel file and store them in Test cases folder under src .

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

You can get the total number of <tr> tags within a <td> tag by giving the xpath of the <td> element by using this function -

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

Now you can use a for each loop to loop through each of the <tr> tags in the above list and then read each value by using getText() method.

## **56. Detail about TestNG Test Output folder.**

It is the directory where reports are generated. Every time tests run in a suite, TestNG creates index.html and other files in the output directory.

## **57. In frame if no frame Id as well as no frame name then which attribute I should consider throughout our script.**

You can go like this.....driver.findElements(By.xpath("//iframe"))...

Then it will return List of frames then switch to each and every frame and search for the locator which you want then break the loop

## **58. What is object repository?**

It is collection of object names their properties, attributes and their values .It may be excel, XML,property file or text file

## **60. What is the difference between @beforemethod and @beforeclass.**

In JUnit4 @Before is used to execute set of preconditions before executing a test.

For example, if there is a need to open some application and create a user before executing a test, then this annotation can be used for that method. Method that is marked with @Before will be executed before executing every test in the class.

If a JUnit test case class contains lot of tests which all together need a method which sets up a precondition and that needs to be executed before executing the Test Case class then we can utilise “@BeforeClass” annotation.

## **61. What are the different Parameters for @Test annotation?**

Parameters are keywords that modify the annotation's function.

For more details Go to: <http://testng.org/doc/documentation-main.html#parameters>

## **62. Can we run group of test cases using TestNG?**

Test cases in group in Selenium using TestNG will be executed with the below options.

If you want to execute the test cases based on one of the group like regression test or smoke test

```
@Test(groups = {"regressiontest", "smoketest"})
```

For more details please see: <http://testng.org/doc/documentation-main.html#test-groups>

## **65. What are the different assertions in SIDE?**

**Assertions** are like Assessors, but they verify that the state of the application conforms to what is expected. Examples include "make sure the page title is X" and "verify that this check box is checked".

## **66. How to store a value which is text box using web driver?**

```
driver.findElement(By.id("your Textbox")).sendKeys("your keyword");
```

## **67. How to handle alerts and confirmation boxes.**

Confirmation boxes and Alerts are handled in same way in selenium.

```
var alert = driver.switchTo().alert();  
  
alert.dismiss(); //Click Cancel or Close window operation  
  
alert.accept(); //Click OK
```

Handle Confirmation boxes via JavaScript,

```
driver.executeScript("window.confirm = function(message){return true;};");
```

## **68. How to mouse hover on an element?**

```
Actions action = new Actions(webdriver);  
  
WebElement we = webdriver.findElement(By.xpath("html/body/div[13]/ul/li[4]/a"));  
  
action.moveToElement(we).moveToElement(webdriver.findElement(By.xpath("/expression-here"))).click().build().perform();
```

## **69. How to switch between the windows?**

```
private void handlingMultipleWindows(String windowTitle) {  
  
    Set<String> windows = driver.getWindowHandles();  
  
    for (String window : windows) {  
  
        driver.switchTo().window(window);
```

```
if (driver.getTitle().contains(windowTitle)) { return; } }
```

## 70. How to switch between frames?

## 71. What is actions class in web driver?

Actions class with web Driver help is Sliding element, Resizing an Element, Drag & Drop, hovering a mouse, especially in a case when dealing with mouse over menus.

## 72. Difference between the selenium1.0 and selenium 2.0?

Selenium 1 = Selenium Remote Control.

Selenium 2 = Selenium Web driver, which combines elements of Selenium 1 and Web driver.

## 73. Difference between find element () and findelements ()?

### **findElement() :**

Find the first element within the current page using the given "locating mechanism".

Returns a single WebElement.

### **findElements() :**

Find all elements within the current page using the given "locating mechanism".

Returns List of Web Elements.

## 75. What is the default time for selenium Ide and webdriver?

Default timeout in selenium ide is 30 seconds.

## 76. Write down scenarios which we can't automate?

Barcode Reader, Captcha etc.

**77. In TestNG I have some test's Test1-Test2-Test3-Test4-Test5I want to run my execution order is Test5-Test1-Test3-Test2-Test4.How do you set the execution order can you explain**

**for that? Use priority parameter in @test annotation or TestNG annotations.**

## 78. Differences between jxl and ApachePOI.

- jxl does not support XLSX files
- jxl exerts less load on memory as compared to ApachePOI
- jxl doesn't support rich text formatting while ApachePOI does.
- jxl has not been maintained properly while ApachePOI is more up to date.
- Sample code on Apache POI is easily available as compare to jxl.

## **79. How to ZIP files in Selenium with an Example?**

### **80. What is default port no?**

4444

### **81. If Default port no is busy how to change port no?**

We can use any port number which is valid.. First create an object to remote control configuration.

Use 'setPort' method and provide valid port number(4545,5555,5655, etc).. There after attach this

remote control configuration object to selenium server..i.e

```
RemoteControlConfiguration r= new RemoteControlConfiguration();
r.setPort(4567);
SeleniumServer s= new SeleniumServer(r);
```

## **82. Does Selenium support https protocols?**

Yes

## **83. Majorly asked test scenario with framework in Interviews?**

Majorly asked are:

- Login for Gmail scenario
- Goggle search and finding no of results
- Downloading a file and save it
- Checking mails and deleting them
- Do shopping in flipkart.com

#### **84. Selenium support mobile applications?**

No, it is browser automation tool, it only automates Websites opening in mobile browser, and mobile APPs

can't be automated.

#### **85. What is wraps Driver?**

For casting selenium instance to selenium2 (webdriver). wraps driver is used.

#### **86. Can you explain Junit Annotation? If there are 1000 test cases. 500 test cases are executed. How will you execute the rest of the test cases by using annotation?"**

The annotations generated with JUnit 4 tests in Selenium are:

1. @Before public void method() - Will perform the method() before each test. This method can prepare the test
2. @Test public void method() - Annotation @Test identifies that this method is a test method.environment,e.g. read input data, initialize the class)
3. @After public void method() - Test method must start with test@Before - this annotation is used for executing a method before

#### **87. Difference between assert and verify in selenium web driver.**

- When an "assert" fails, the test will be aborted. Assert is best used when the check value has to pass for the test to be able to continue to run log in.
- Where if a "verify" fails, the test will continue executing and logging the failure. Verify is best used to check non critical things. Like the presence of a headline element.

#### **88. "I want to find the location of ""b"" in the below code, how can I find out without using xpath, name, id, csslocator, index.<div>**

```
<Button>a</button>
<Button>b</button>
<Button>c</button>
</div>
② driver.findElement(By.xpath("//*[contains(text(),'b')]")).click(); or
② //div/button[contains(text(),'b')]
```

**89. How to do Applet testing using selenium?**

Please see below URLs:

<http://docs.codehaus.org/display/FEST/Selenium>

<https://code.google.com/p/festselenium/>

**90. Name 5 different exceptions you had in selenium web driver and mention what instance  
you got it and how do you resolve it?**

- WebDriverException
- NoAlertPresentException
- NoSuchElementException
- NoSuchElementException
- TimeoutException

**91. How do you manage the code versions in your project?**

- Using SVN or other versioning tools

**92. Latest version of Firefox and selenium in market and the version on which you are testing  
which you are testing.**

- FF Latest version till Dec,2013 for windows7,64 bit :26.0.I use FF 25.0.1 (ur ans. may differ)
- Selenium web driver latest version till dec,2013- 2.39.0 I use selenium 2.37 see latest at

<http://www.seleniumhq.org/download/>

**93. How to know all the methods supported in  
web driver  
and its syntax.**

- In Org.openqa.selenium package, web driver interface has all the main methods that can be used in Selenium Web driver

- [HTTP://docs.seleniumhq.org/docs/03\\_webdriver.jsp](HTTP://docs.seleniumhq.org/docs/03_webdriver.jsp)

**94. How do you create html test report from your test script?**

- I would see below 3 ways:
- Junit: with the help of ANT.
- TestNG: using inbuilt default.html to get the HTML report. Also XLST reports from ANT, Selenium, TestNG combination.
- Using our own customized reports using XSL jar for converting XML content to HTML.

**95. List the browsers, OS supported by the Selenium**

**Windows Linux Mac**

IE Y NA NA

FF Y Y Y

Safari Y N Y

Opera Y Y Y

Chrome Y Y Y

**96. Can you explain Selenium Mobile Automation?**

**97. What mobile devices it may Support?**

Selenium Web driver supports all the mobile devices operating on Android, IOS operating Systems

- Android – for phones and tablets (devices & emulators)
- iOS for phones (devices & emulators) and for tablets (devices & emulators)

**99. What are the test types supported by Selenium?**

Selenium supports UI and functional testing. As well it can support performance testing for reasonable load using selenium grid.

**100. In what all case we have to go for “JavaScript executor”.**

Consider FB main page after you login. When u scrolls down, the updates get loaded. To handle this activity, there is no selenium command. So you can go for javascript to set the scroll down value like `driver.executeScript("window.scrollBy(0,200)", "");`

