RAJ JADHAV

MORE HELP -
+91 9623753538
LIKE----------
HTTPS://WWW.FACEBOOK.COM/THECAREERHELPLINE/

17

Selenium

RAJ JADHAV

MORE HELP -
+91 9623753538
LIKE ----------
HTTPS://WWW.FACEBOOK.COM/THECAREERHELPLINE/

01/01/2017

Appendix

Selenium Interview Questions : Page 1 **Important Que**

1. Describe Frame work design & Ov erall flow diagram using in your project.
2. How to handle windows authentication popup with & without using third party tool.
3. What are different types of exception in selenium WebDriv er?
4. What are different types of methods in Selenium ?
5. Can we enter text without using sendKeys() if yes, how?
6. Write a program for string rev erse. (Imp)
7. Write a program for Fibonacci.
8. Write a program to check whether a number is prime .
9. How to find more than one web element in the list?
10. What is the differenc e between driver.close() and driv er.quit command ?
11. Difference between findElement/findElements ?
12. How to Count the number of l inks in a page ?
13. Does java supports multiple inheritance ?
14. Difference between assert and v erify in selenium Webdriv er
15. How to take a screen shot using selenium Webdriver ?
16. How will you execute your login script using chrome browser from your editor using selenium ?
17. Can we use multiple catch in try, how?
18. Difference between webdriv er/firefoxDriv er ?
19. What is the differenc e between "GET" and "NAVIGATE" to open a web page in selenium web driv er?
20. How can we get the font size, font colour ,font type used for a particular text on a web page using selenium web driv er ?
21. How do you manage the code versions in your project ?
22. How do we handle dyna mic elements without using X path ?
23. How to handle alerts and confirmation boxes.
24. How to handle colours in Webdriver ?
25. How to press shift +tab ?
26. How to take a screenshot in selenium.
27. Is there a way to click hidden link in Webdriv er.
28. Login for Gmail scenario ?
29. What are the technical challenges that you faced with selenium ?
30. Difference between flex and flash application
31. What classes extends Webdriv er
32. What is action class in Webdriv er ?
33. What is selense
34. What is SIDE
35. what is the hierarchy of TestNG annotation
36. What is Selenium ? What are the different Selenium components?
37. What is an Xpath?
38. How to refresh a page without using context click?
39. How to handle autocom plete box in web driv er?
40. Difference between the selenium 1.0 and selenium 2.0 ??
41. How to switch between the windows (getWindowHandle and getWindowHandles)

Selenium Interview Questions : PAGE 2

42. WRITE A CODE FOR NUMBER OF CHARACTER IN STRING ?
43. HOW TO READ PARTICULAR CELL FROM HTML TABLE
44. HOW TO CONNECT JAVA APPLICATION WITH ORACLE DATABASE
45. HOW TO SWITCH BETWEEN FRAMES IN SELENIUM WEBDRIVER
46. WHAT ARE THE DIFFERENT TYPES OF LOCATORS IN SELENIUM ?
47. HOW TO HANDLE AJAX POPUP WINDOW
48. DIFFERENCE BETWEEN INTERFACE AND ABSTRACT CLASS
49. HOW TO CONNECT TO EXCEL CODE FOR EXCEL(POI) API TO READ EXCEL,
50. RETRY EXECUTING ONLY FAILED TESTS USING TESTNG
51. RELATIV E XPATH METHOD
52. PARALLEL EXECUTION OF TEST METHODS IN TESTNG
53. PARALLEL EXECUTION OF CLASSES IN TESTNG
54. TESTING IN MULTIPLE BROWSERS USING SELENIUM AND TESTNG
55. FIND OUT BROKEN LINKS ON WEBSITE USING SEL ENIUM WEBDRIV ER AND HTTP CLIENT
56. FIND BROKEN / INVALID IMAGES ON A PAGE
57. HOW TO HANDLE INTERNATIONALISATION THROUGH WEB DRIV ER?
58. PAGE OBJECT MODEL | POM
59. TAKING SCREENSHOT ONLY FOR FAILED TESTS
60. WAIT COMMANDS IN WEBDRIV ER
61. WHAT IS TESTNG LISTENER ??
62. HOW TO SELECT A ELEMENT FROM DROPDOWN ?
63. WHAT IS WEBDRIV ER EV ENT LISTENER ??
64. HOW DOES THE SELENIUM WEBDRIV ER WORK?
65. HOW DO YOU CREATE HTML TEST REPORT FROM Y OUR TEST SCRIPT
66. HOW DO YOU HANDLE HTTPS WEBSITE IN SELENIUM
67. HOW THE TESTNG INTERACT WITH SELENIUM CORE
68. HOW TO CHANGE USER AGENT IN FIIREFOX BY SELENIUM WEBDRIVER
69. HOW TO READ PARTICULAR CELL FROM HTML TABLE
70. HOW TO CHECK ALL CHECK BOXES IN A PAGE
71. HOW TO DISABLE COOKIES IN A BROWSER
72. HOW TO GET THE NAME OF BROWSER USING WEBDRIV ER?
73. HOW TO HANDLE AJAX POPUP WINDOW
74. HOW TO HANDLE AUTO COMPLETE BOX IN WEBDRIV ER
75. HOW TO HANDLE NETWORK LATENCY USING SELENIUM
76. HOW TO KNOW ALL THE METHODS SUPPORTED IN WEBDRIV ER AND ITS SY NTAX
77. HOW TO LOGIN INTO ANY SITE IF ITS SHOWING ANT AUTHENTICA TION POP UP FOR USER NAME AND PASSWORD DURING LAUNCH OF THE URL
78. HOW TO MOUSER HOV ER ON AN ELEMENT
79. HOW TO PASS PARAMETERS FROM TESTNG.XML INTO TEST CASES
80. HOW TO PERFORM DOUBLE CLICK USING WEBDRIV ER
81. HOW TO PREPARE CUSTOMIZED HTML REPORT USING TESTNG IN HYBRID FRAMEWORK
82. HOW TO PUT TEXT IN ANY SEARCH BOX USING SELENIUM WEBDRIV ER
83. HOW TO REFRESH A PAGE WITHOUT USING CONTEXT CLICK ?
84. HOW TO RUN TESTS IN MULTIPLE BROWSER PA RALLEL (SELENIUM GRID)

85. How to store a value which is text boxes using Webdriver ?
86. How to switch between the windows
87. How to type next in a new line inside a text area
88. How to work with dynamic webtable
89. How to work with radio button in Webdriver
90. How we can retrieve the dynamically changing ads
91. If a selenium function requires a script argument ,what would that argument look like in a general terms ?
92. If TestNG I have some test's Test-1 Test-2 Test-3 Test-4 Test-5 I want to run my execution order is Test-5 Test-4 Test-3 Test-2 Test-1 .how do you set the execution order can you explain for that ?
93. Latest version of firefox and selenium in market and he version on which you are testing
94. List the browsers, OS supported by the selenium
95. Please tell me the difference b/w implicit wait and explicit wait
96. Provide Detail about testNG test Output folder.
97. Suppose Developer changed the existing image to new image with same x path .Is test case pass or fail
98. There is a scenario whenever "assert.assertequals()" function fails automatically it has to take screenshot .how can you achieve this
99. What are oops concept
100. What are benefits of using testNG
101. What are browser supported by selenium IDE
102. What are different access specifier in java
103. Whar are the different assertions or check points used in your script
104. What are the different parameters For @ test annotation
105. What are different types of driver implementation
106. What are the features of TestNG?
107. What are the operating system supported by selenium
108. What are the possible scenarios where selenium finding element get fails
109. What is the basic use of firefox profiles and how can we use them using selenium
110. What is the default time for selenium ide and Webdriver
111. What is the differenc e b/w get window handle and get window handles
112. What is difference between get and navigate to open a web page in selenium Webdriver
113. What is difference between @ befor method and @ before class\
114. What is difference between absolute path and relative path
115. What is difference between driver.close and driver .quit
116. What is difference between selenium RC and Webdriver
117. What is difference between single and double slash
118. What is difference between thread.sleep () and seleniumsetspeed("2000")
119. what is difference between thread.sleep() and selenium.Setspeed()
120. what is most challenging test problem in my career automation
121. what mobile devices it support
122. which are the different methods to locate an element
123. which one is better one xpath or css
124. which repository you have used to store the test scripts
125. why we refer firefox driver to the Webdriver inheritance
126. write a code to make use of assert if my user name is incorrect
127. write a java program for swapping of two numbers

128. **WRITE DOWN SCENARIOS WHICH WE CAN'T AUTOMATE.**
129. **DIFFERENCES BETWEEN SELENIUM WEBDRIVER RC , IDE AND RC ?**
130. **DO SHOPPING IN FLIPKART ?**
131. **WHAT DOES DESIRED CAPABILITES DO?**
132. **WHAT IS ACTIONS CLASS IN WEBDRIVER**
133. **WHAT IS DIFFERENCE BETWEEN OV ERLOAD AND OV ERRIDE**
134. **WHAT IS DIFFERENCE BETWEEN WEBDRIV ER AND LISTENER AND TESTNG LISTENER**
135. Read particular cell from html table
136. ⇨ Logic for looping the table, xapth to iterate rows & columns
137. **Can we run Group of test cases using TestNG**
138. **Ans:Yes**
139. **checking mails and deleting them**
140. **Code for opening firefox browser**
141. **Does selenium support https protocols ?**
142. **Downloading a file and save it?**
143. **Explain any 3 testNG annotation**
144. **Google search and finding no of results**
145. TestNG- Write sample code to select browser depending on parameter given in testing.xml
146. Parameter annotation, how to add system property for chrome/iedriver.exe for chrome & IE
147. How to check result with expected
148. Assert.assertEquals().
149. implicit wait and explicit wait- WebDriverWait or Thread.sleep()
150. Can we execute java code without main()no, only can compile
151. . oops concept and diff between overloading and overriding
152. Find the top from array list(10 number) using java code
153. check logic for **Fibonacci** series, even/odd number, top value from 5th number
154. what are the access modifier and difference.public, private, protected, default
155. What is abstract class with example
156. Polymorphism
157. Inheritance
158. Collection-set,list,map
159. Exception-exception,error,assertion
160. Oops-Type casting of object
161. Wrapper classes

Wrapper classes allows us to convert the primitive types into an object type.

Java is not 100% object oriented programming language because of the 8 primitive types. Then wrapper classes are introduced to give the primitive types an object form. So the primitive types can also be stored as an object of its respective wrapper class

The 8 primitive types and its wrapper classes are,

byte.- By te
int - Integer short
- Short
long- Long
float - Float
double - Double
char - Character
boolean - Boolean

all this wrapper classes are available in java.lang package

Now if you want to store an integer as an object type you can write it as

Integer i=new Integer(10);

This is known as **Boxing,** converting a primitive type into an object.

Now to get that integer value back,

int a=i.intValue();

This operation is known as **Unboxing** converting the value of a wrapper class object into a primitive type.

After java 1.5/5 Boxing and Unboxing became automatic. You can directly assign the primitive as an object of wrapper class.

Integer i=10;

Before java 1.5/5, databases stores only Object class objects so to store a primitive type into an ArrayList or Vector wrapper classes are used. Primitive types are converted into wrapper class object through boxing and upcasted to Object type to store in the database.

The significance of wrapper class comes when you want to write a program which will work with any type of value. To write such a program declare the arguments as Object type since Object class is extended by all the other class in java Object class type can store any kind of objects.

Lets take an example, you want to write a program which will work if you pass any kind of values

```
public static void printHello(Object a)
  {
   Sy stem.out.println("Hello");
  }
```

This program will work with all kind of values. Since Object class is the super most class, Object type variable can accept any objects. the wrapper class comea into picture when you pass a primitive type value. Imagine you are calling the function with a value 10

```
printHello(10);
```

Now the 10 will be boxed and becomes an object of Integer class(wrapper class) since Integer extends Object the method will work fine.

There are many other uses which comes while overriding the build in methods like compare(), equals() etc because all these functions have Object type as parameters.

162. Constructor-abstract class
163. Fibonacii,String reverse,remove duplicate from array,sort array, find min-max value from array
164. JDBC
165. TestNG-
166. Annotations
167. Listeners
168. Parameterization/Dataprovider
169. Parallel run- threading(XML)
170. Factory
171. how to create testng.xml programitacally
172. Webdriver-
173. Eventfiringwebdriver
174. Capabilities
175. IE issues & resolution-6-7
176. Execptions
177. Class interface eg.
178. Synchronization-waits
179. Frame
180. xpath,css-functions-contains,sibilings,starts-ends with,relative-absolute xpath,nth-child,nth-of-type

1. AS A TESTER WHAT SHOULD BE YOUR APPROACH WHEN REQUIREMENTS CHANGE CONTINUOUSLY?
2. LIST OUT THE PROS AND CONS OF EXPLORATORY TESTING (USED IN AGILE) AND SCRIPTED TESTING?
3. EXPLAIN THE DIFFERENCE BETWEEN EXTREME PROGRAMMING AND SCRUM?
4. WHAT IS AN EPIC, USER STORIES AND TASK?
5. EXPLAIN WHAT IS RE-FACTORING?
6. EXPLAIN HOW YOU CAN MEASURE THE VELOCITY OF THE SPRINT WITH VARYING TEAM CAPACITY?
7. MENTION THE KEY DIFFERENCE BETWEEN SPRINT BACKLOG AND PRODUCT BACKLOG?
8. IN AGILE MENTION WHAT IS THE DIFFERENCE BETWEEN THE INCREMENTAL AND ITERATIVE DEVELOPMENT?
9. EXPLAIN WHAT IS SPIKE AND ZERO SPRINT IN AGILE? WHAT IS THE PURPOSE OF IT?
10. WHAT IS TEST DRIVEN DEVELOPMENT?
11. PROTOTYPES AND WIREFRAMES ARE WIDELY USED AS PART OF?
12. EXPLAIN WHAT IS APPLICATION BINARY INTERFACE?
13. EXPLAIN IN AGILE, BURN-UP AND BURN-DOWN CHART?
14. EXPLAIN WHAT IS SCRUM BAN?
15. WHAT IS STORY POINTS/EFFORTS/ SCALES?
16. EXPLAIN WHAT IS TRACER BULLET?
17. WHAT IS A TEST STUB?
18. WHAT ARE THE DIFFERENCES BETWEEN RUP (RATIONAL UNIFIED PROCESS) AND SCRUM METHODOLOGIES?
19. WHY CONTINUOUS INTEGRATION IS IMPORTANT FOR AGILE?
20. WHAT TESTING IS DONE DURING AGILE?
21. Explain what is Velocity in Agile?
22. WHAT ARE THE QUALITIES OF A GOOD AGILE TESTER SHOULD HAVE?
23. WHO ARE ALL INVOLVED IN THE AGILE TEAM?
24. MENTION IN DETAIL WHAT ARE THE ROLE'S OF SCRUM MASTER?
25. MENTION WHAT ARE THE AGILE QUALITY STRATEGIES?
26. MENTION WHAT ARE THE TOOLS THAT CAN BE USEFUL FOR SCREENSHOTS WHILE WORKING ON AGILE PROJECTS?
27. MENTION WHAT ARE THE ADVANTAGES OF MAINTAINING CONSISTENT ITERATION LENGTH THROUGHOUT THE PROJECT?
28. IF A TIME BOX PLAN NEEDS TO BE REPRIORITIZED WHO SHOULD RE-PRIORITIES IT?
29. MENTION WHAT SHOULD A BURNDOWN CHART SHOULD HIGHLIGHT?
30. Mention what is the difference between Scrum and Agile?
31. Mention what are the challenges involved in AGILE software development?
32. When not to use Agile?
33. EXPLAIN HOW CAN YOU IMPLEMENT SCRUM IN AN EASY WAY TO YOUR PROJECT?
34. EXPLAIN WHAT DOES IT MEAN BY PRODUCT ROADMAP?

Manual Important Question
Difference bwtween Desktop application and Web Based Application

Question 1 : Frame work design-Overall flow diagram.(Asked)

Automation workflow for the application can be presented as follows:

- First of all it is required to identify tasks that an application has to accomplish.
- Second, a set of necessary input data has to be created.
- Third, expected results have to be defined in order one can judge that an application (a requested feature) works correspondingly.
- Fourth, Executes a test.
- Finally, Compares expected results with actual results, and decides whether the test has been passed successfully.

Environment Specifications:

- Selenium WebDriver (Supports all major browsers, we use Mozilla, chrome and IE)
- Eclipse IDE
- Java
- TestNG
- **AutoIT** Tool (Used to handle Windows popup for Document Uploads and Downloads.)
- **JExcel or Apache POI** to perform operations with excel like read, write and update the excel sheet

Framework has the following tools:

1. Selenium - Selenium is a well know open source testing framework, which is widely used for testing Web-based applications. It has different components and in that WebDriver has rendered the Selenium Remote Control obsolete, and is commonly referred to as Selenium 2.0.Selenium WebDriver supports most of all browsers to run your test cases and many programming languages like C#, Java, Python, Ruby, .Net, Perl, PHP, etc.. To create and modify your test scripts.

2. Eclipse IDE: Eclipse is an integrated development environment (IDE) for Java. The Eclipse IDE is the most known product of the Eclipse Open Source project.

3. TestNG - Is a testing framework inspired from JUnit and NUnit. It has extended new functionalities which made it more powerful and easier than the other testing frameworks. It supports ReportNG (simple HTML reporting plug-in) and XLST (Graphical / Pictorial reports) plug-ins to customize or extend the default TestNG reporting style.

4. AutoIT - AutoIT v3 is a freeware BASIC-like scripting language designed for automating the Windows GUI and general scripting. It uses a combination of simulated keystrokes, mouse movement and window/control manipulation in order to automate tasks which are not possible with selenium.

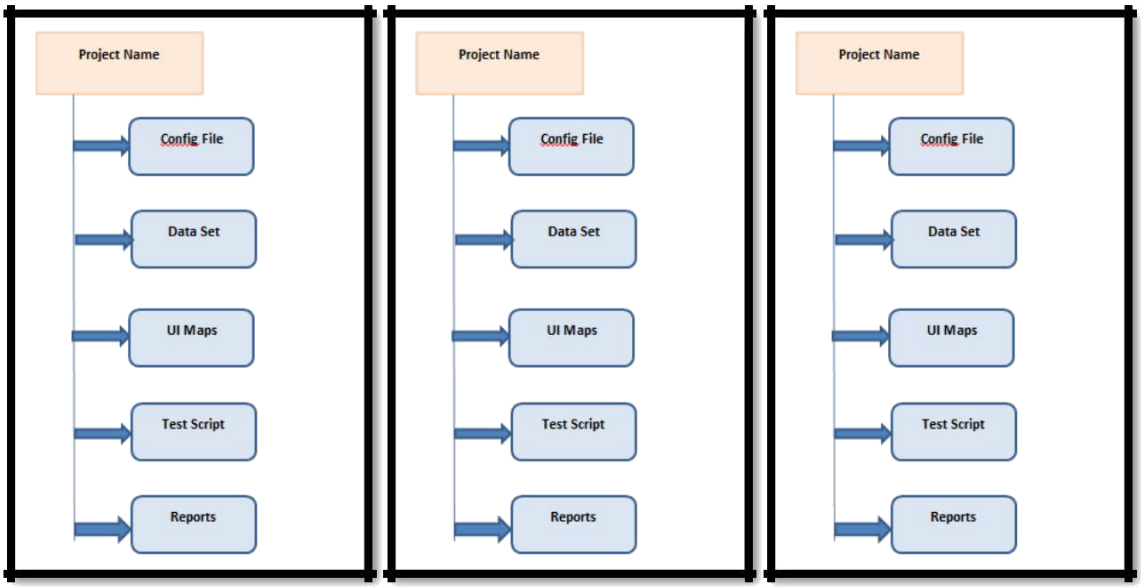
File Formats Used in the Framework:

- **Properties file** – We use properties file to store and retrieve the UI elements of an application or a website and data set file paths. It contains id of the elements, name, Xpath or Css selector etc.

- **Excel files** – Excel files are used to pass multiple sets of data to the application.

- **Xml file** – Is used to execute the test scripts. Based on the package or classes or Tests mentioned in the xml file scripts will be executed.

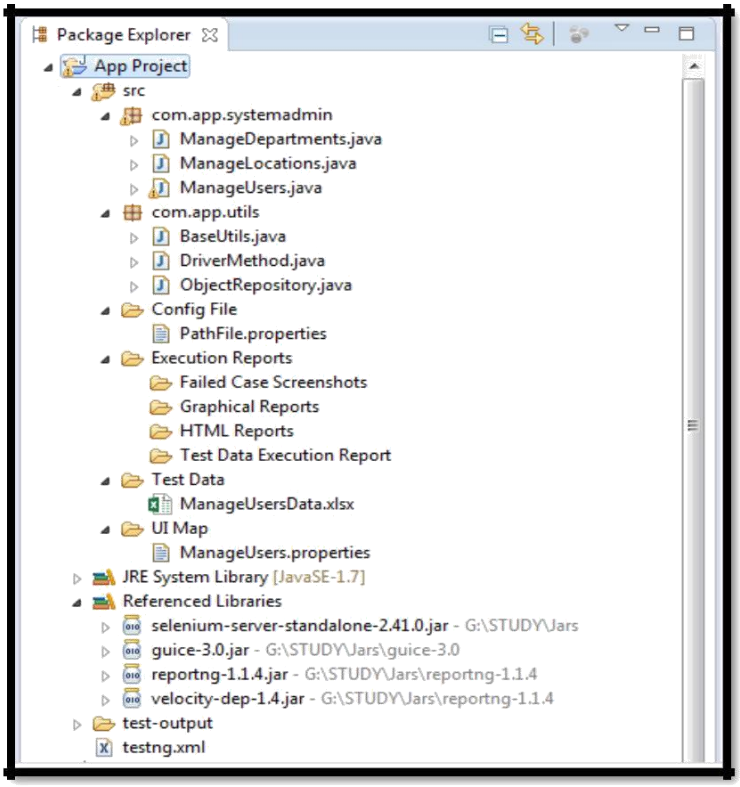
The following figure explains physical structure of files required for Test Automation Framework



1. **UI Map / Object Repository-contains** all objects
2. **Data Set / Test Data** –external data
3. **Test Automation Scripts-scripts**
4. **Reports / Executed Results**
5. **TestNG xml file**

The Project Folder Structure:

- All the basic required folders are created with the sub folders and classes under each folder



The Following explains the structure in detail:-

1. UI Map / Object Repository

UI Map is a concept for defining, storing, and serving UI elements of an application or a website. The UI Map properties file contains a set-value'of'keypairs, where key is an alias of the a value is the locator.

We will create properties file for every single page and capture all the UI elements present on the page and use it as per needs.

UI Map or Object Repository Samples: -

Role - System Admin

ManageUsers.Properties

ManageDepartments.Properties

ManageLocations.Properties

Example:-

ManageUsers.Properties

ManageUsers.properties

```
#These properties belongs to Manage Users
#---------------------------------------

#PageTitle
userPageTitle=Manage Users

#To Click on Add User in Manage Users Grid
addUserButton=name=add user

#To enter user first name
firstName=id=fname

#To enter user middle name
middleName=id=mname

#To enter user last name
lastName=id=lname

#To Enter email address
emailAddress=id=email

#To enter contact number details
contactNumber=name=contact

#To click on submit button
submitButton=name=submit
```

Why and How to use UI Map / Object Repository / OR

To perform any action we need locators. Tool will perform any action based on the locators. For each and every action tool depends on these locators. If the tool does not identify the locators, it simply throws an error as Locator / Element Not Found or Identified.

In order to make sure the tool executes smoothly, we need to provide accurate unique identifier / locators.

We need to keep all the locators at one place where we can easily modify the locators / identifies if there are any UI changes in the application.

If not, it will become difficult to change even one locator, as it will be used at many different places in test scripts.

It is always better to keep locators in a separate file and at easily accessible location. It should be defined in such a way if any new person joins in the middle of the project, he/she should be capable of making changes / adding any new locators to the file.

In the above example, we have created properties file for each function which will store all the UI elements.

Example: In Manage Users, we will store all the locators' / UI element used and accessed in "Add Users", "Edit Users" andvalue"DeleteisthelocatorUsers". tes

Click here for more..

2. Data Set / Test Data

Data set stores the data files, Script reads test data from external data sources and executes test based on it. Data sets increases test coverage by performing testing with various inputs and reduce the number of overall test scripts needed to implement all the test cases.Click here for more..

3. Test Automation Scripts

A test is considered as a single action or a sequence of actions, that defines whether a specific feature meets functional requirements. It has multiple test files / packages / class files which will be executed based on the configurations defined in testng.xml. Click here for more..

4. Reports / Executed Results

Test report/results is a document which contains summary of test activities. After execution is completed, it is very important to communicate the test results and findings to the project manager along with the screenshots for failed tests and with that decisions can be made for the release. Click here for More..

5. TestNG xml file

In order to create a test suite and run separate test cases, we need framework which drives the automation. Here testng.xml can be called as "driver" which drives several test cases automated using selenium code. Advantage of using TestNG with Selenium is of running multiple test cases from multiple classes using xml configuration file . Click here for More..

Properties File

In Selenium .properties files are mainly used to store GUI locators / elements, and also global fields like database configuration details

'.properties' files are mainly used in Java programs to maintain project configuration data, database config or project settings etc. Each parameter in properties file is stored as a pair of strings, in key and value format, where each key is on one line. You can easily read properties from some file using object of type Properties.

Below is a example program which demonstrate to read the data from .properties file using Java.

```
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.Properties;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;

public class ReadFileData {

  public static void main(String[] args)  {
           File file = new File("D:/Dev/ReadData/src/datafile.properties");
                  FileInputStream fileInput = null;
                  try {
                           fileInput = new FileInputStream(file);
                  } catch (FileNotFoundException e) {
                           e.printStackTrace();
                  }
                  Properties prop = new Properties();

                  //load properties file
                  try {
                           prop.load(fileInput);
                  } catch (IOException e) {
                           e.printStackTrace();
                  }

                  WebDriver driver = new FirefoxDriver();
```

```
                    driver.get(prop.getProperty("URL"));
                    driver.findElement(By.id("Email")).sendKeys(prop.getProperty("username"));
                    driver.findElement(By.id("Passwd")).sendKeys(prop.getProperty(     "password"));
                    driver.findElement(By.id("SignIn")).click();


                    System.out.println("URL ::" + prop.getProperty("URL"));
                    System.out.println("User name::" +prop.getProperty( "username"));
               System.out.println("Password::" +prop.getProperty("password"));
  }
}
```

The below is the Output after executing the program: We are passing the properties values to the webdriver and printing the values at end

```
URL ::http://gmail.com
User name::testuser
Password::password123
```

We can also get all the key value pairs in the properties file using the below java program:

```
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.Enumeration;
import java.util.Properties;

public class ReadFileData {

         public static void main(String[] args) {

                            File file = new File

("D:/Dev/ReadData/src/datafile.properties");
                            FileInputStream fileInput = null;
                            try {
                                      fileInput = new FileInputStream(file);
                            } catch (FileNotFoundException e)
                                      { e.printStackTrace();
                            }
```

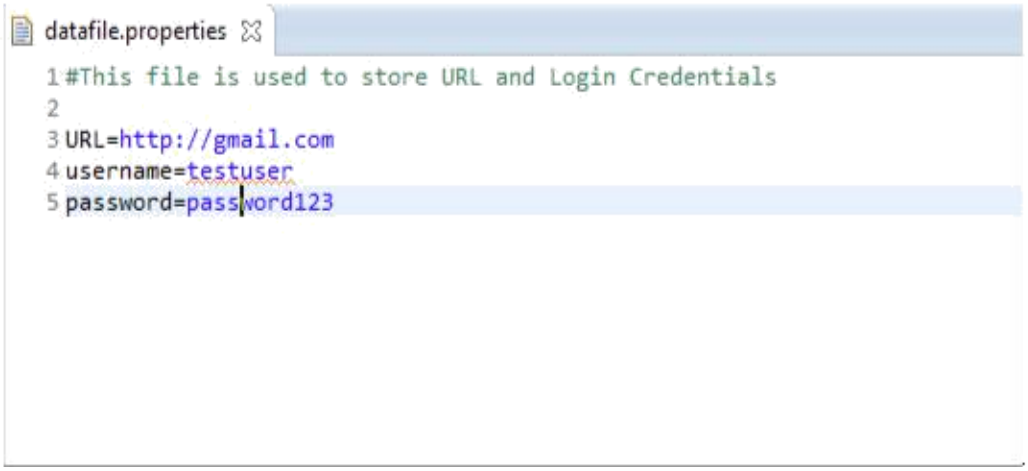
```
                Properties prop = new Properties();
                try {
                        prop.load(fileInput);
                } catch (IOException e) {
                        e.printStackTrace();
                }

                Enumeration KeyValues = prop.keys();
                while (KeyValues.hasMoreElements()) {
                        String key = (String) KeyValues.nextElement();
                        String value = prop.getProperty(key);
                        System.out.println(key + ":- " + value);
                }
        }
}
```

We have not used individual property name to get the KeyValue in the above program. It will get all the key values which are available in the .properties file

The Output of the above program:

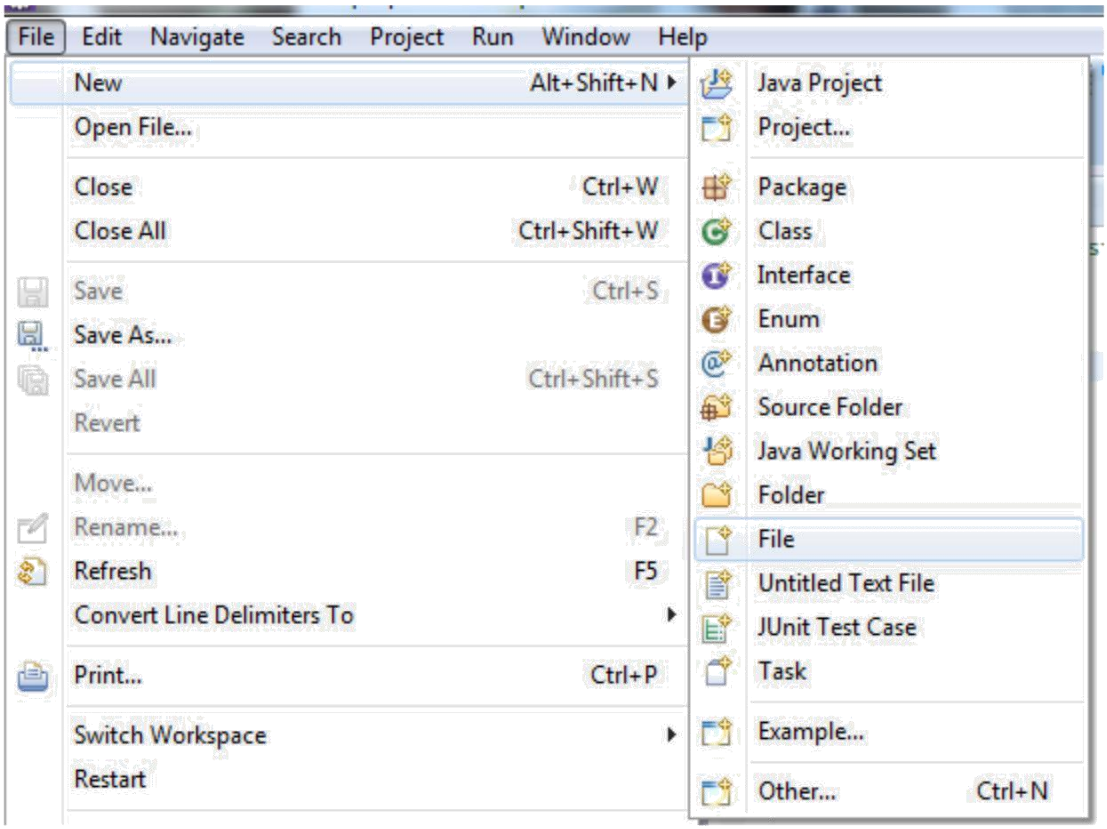
```
URL:- http://gmail.com
password:- password123
username:- testuser
```

The below is the sample properties file that was used for the above tests.

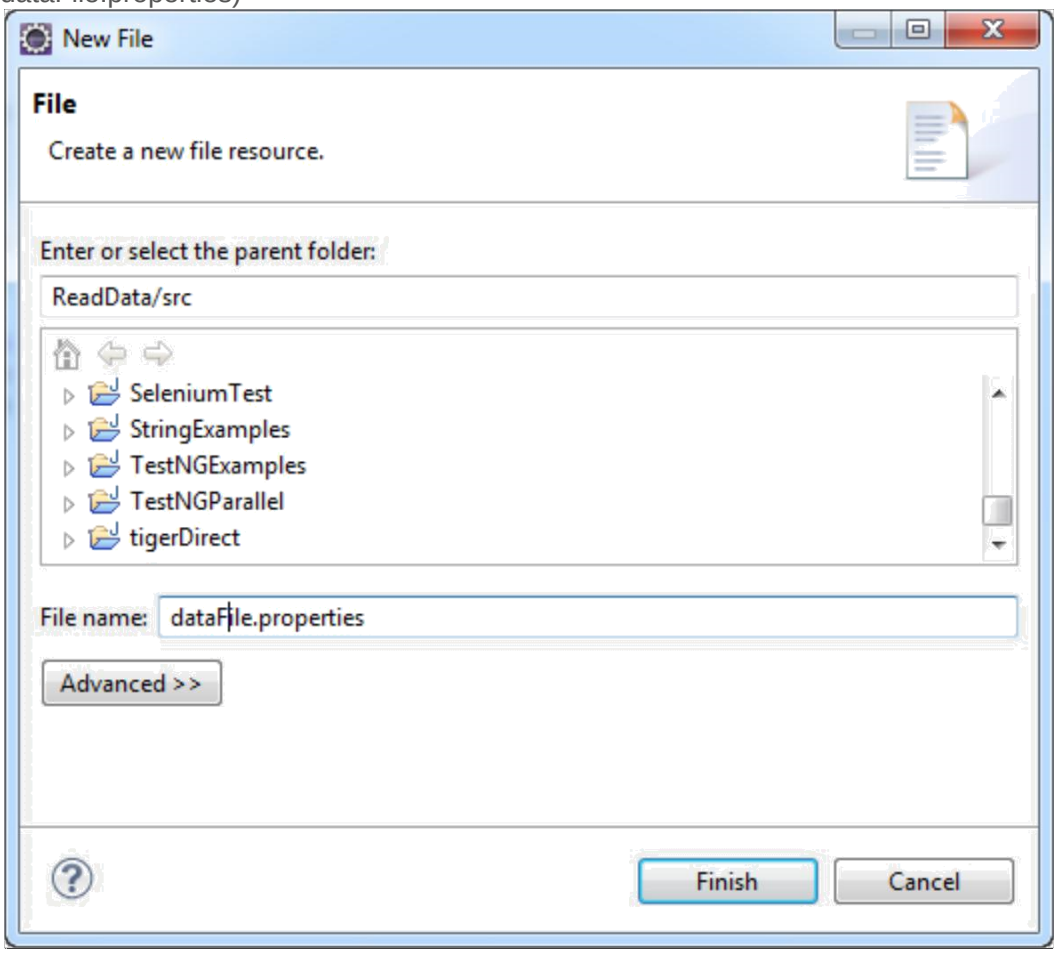


How to create properties file?

Navigate to File menu -> New -> Click on File


File
Edit
Navigate
Search
Project
Run
Window
Help
New
Alt+Shift+N
Open File...
Close
Ctrl+W
Close All
Ctrl+Shift+W
Save
Ctrl+S
Save As...
Save All
Ctrl+Shift+S
Revert
Move...
Rename...
F2
Refresh
F5
Convert Line Delimiters To
Print...
Ctrl+P
Switch Workspace
Restart
Java Project
Project...
Package
Class
Interface
Enum
Annotation
Source Folder
Java Working Set
Folder
File
Untitled Text File
JUnit Test Case
Task
Example...
Other...
Ctrl+N

The only thing we need to do is, give the file name and give the extension as '.properties'. (Example: dataFile.properties)



Question 2: Handle windows authentication popup without using any third party tool .

Using Autoit (Third Party Tool)
Download and in Autoit editor write script through below command
ControlFocus()
ControlSet()
ControlClick()
Save file and Compile Script with your version and after that you get .exe file.
Java code for executing .exe file
Runtime.getRuntime().exec(\\Path)
It throws checked exception IOexception.

Another way of doing this is using Robot class

Robot Class is available under java.awt.package.

Methods in Robot Class can be effectively used to do the interaction with pop-ups in Web Applications. Selenium does not provide support to handle browser pop-ups or the native operating system pop-ups. To handle this kind of pop-up, we need help of Robot Class. This is also used while we need to handle file upload and download activity using selenium Webdriver.

Some of the popular methods under Robot Class are:
.keyPress();
.mousePress();
.mouseMove();
.keyRelease();
.mouseRelease();

First of all create the object of the Robot Class as following:

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

1. **.keyPress()**
robot.keyPress(KeyEvent.VK_ESC);
This will press Escape key on keyboard.
2. **.keyRelease()**
robot.keyRelease(KeyEvent.VK_CAPS_LOCK);
This will release the CAPS_LOCK key.
3. .**mousePress()**
robot.mousePress(InputEvent.BUTTON1_MASK);
This will press Left mouse button.
4. **.mouseRelease()**
robot.mouseRelease(InputEvent.BUTTON1_MASK);
This will release Left mouse button.
5. **.mouseMove()**
robot.mouseMove(coordinates.getX(), coordinates.getY());
This will move the mouse pointer to X and Y co-ordinates.

```
package com.easy.upload;
import java.awt.Robot;
import java.awt.Toolkit;
import java.awt.datatransfer.StringSelection;
import java.awt.event.KeyEvent;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.annotations.Test;

public class UploadFileRobot {

	String URL = "application URL";
	@Test
	public void testUpload() throws InterruptedException
	{
		Webdriver driver = new FirefoxDriver();
		driver.get(URL);
		WebElement element =
		driver.findElement(By.name("uploadfile")); element.click();
		uploadFile("path to the file");
		Thread.sleep(2000);
	}

	/**
     * This method will set any parameter string to the system's
     clipboard. */
	public static void setClipboardData(String string) {
		//StringSelection is a class that can be used for copy and paste
operations.
		   StringSelection stringSelection = new StringSelection(string);

Toolkit.getDefaultToolkit().getSystemClipboard().setContents(stringSelection, null);
		}

	public static void uploadFile(String fileLocation)
	 { try {
		//Setting clipboard with file location
	    setClipboardData(fileLocation);
	    //native key strokes for CTRL, V and ENTER
	    keys Robot robot = new Robot();

	    robot.keyPress(KeyEvent.VK_CONTROL);
	    robot.keyPress(KeyEvent.VK_V);
	    robot.keyRelease(KeyEvent.VK_V);
	    robot.keyRelease(KeyEvent.VK_CONTROL);
	    robot.keyPress(KeyEvent.VK_ENTER);
	    robot.keyRelease(KeyEvent.VK_ENTER);
	 } catch (Exception exp) {
		exp.printStackTrace();
	 }
    }}
```

Scenario-
1-Open Firefox and Create profile in monster.com
2- Click on upload button and select the file and
save Lets implement the same

```
import java.awt.datatransfer.StringSelection;
import java.awt.event.KeyEvent; import
java.awt.*;
import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.annotations.Test;

public class FileUpload {

@Test
public void Fileupload() throws AWTException, InterruptedException
{

// Start browser
 WebDriver driver = new FirefoxDriver();
// maximize browser
 driver.manage().window().maximize();
  // Specify the file location with extension
 StringSelection sel = new StringSelection("C:\\Users\\Desktop\\1.doc");
   // Copy to clipboard
 Toolkit.getDefaultToolkit().getSystemClipboard().setContents(sel,null);
 System.out.println("selection" +sel);
// Open Monster.com
 driver.get("http://my.monsterindia.com/create_account.html");
 Thread.sleep(2000);
 // This will scroll down the page
 JavascriptExecutor js = (JavascriptExecutor)driver;
 js.executeScript("scroll(0,350)");
// Wait for 5 seconds

// This will click on Browse button
 driver.findElement(By.id("wordresume")).click();
 System.out.println("Browse button clicked");
 // Create object of Robot class


  // Press Enter
 robot.keyPress(KeyEvent.VK_ENTER);
// Release Enter
 robot.keyRelease(KeyEvent.VK_ENTER);
  // Press CTRL+V
 robot.keyPress(KeyEvent.VK_CONTROL);
 robot.keyPress(KeyEvent.VK_V);
// Release CTRL+V
 robot.keyRelease(KeyEvent.VK_CONTROL);
 robot.keyRelease(KeyEvent.VK_V);
 Thread.sleep(1000);
        // Press Enter
 robot.keyPress(KeyEvent.VK_ENTER);
 robot.keyRelease(KeyEvent.VK_ENTER);
}}
```

Question 3 : Different types of exception in selenium Webdriver.(Asked)

https://selenium.googlecode.com/git/docs/api/py/common/selenium.common.exceptions.html

Examples:

```
class WebDriverException(Exception):
 class ErrorInResponseException(WebDriverException):
 class InvalidSwitchToTargetException(WebDriverException):
 class NoSuchFrameException(InvalidSwitchToTargetException):
 class NoSuchWindowException(InvalidSwitchToTargetException):
 class NoSuchElementException(WebDriverException):
 class NoSuchAttributeException(WebDriverException):
 class StaleElementReferenceException(WebDriverException):
 class InvalidElementStateException(WebDriverException):
 class NoAlertPresentException(WebDriverException):
  class ElementNotVisibleException(InvalidElementStateException):
class ElementNotSelectableException(InvalidElementStateException):
 class InvalidCookieDomainException(WebDriverException):
 class UnableToSetCookieException(WebDriverException):
 class RemoteDriverServerException(WebDriverException):
 class TimeoutException(WebDriverException):
 class MoveTargetOutOfBoundsException(WebDriverException):
 class UnexpectedTagNameException(WebDriverException):
 class InvalidSelectorException(NoSuchElementException):
 class ImeNotAvailableException(WebDriverException):
 class ImeActivationFailedException(WebDriverException):
```

WebDriverException

```
WebDriver Exception comes when we try to perform any action on the
non-existing driver.

WebDriver driver = new InternetExplorerDriver();

driver.get("http://google.com");

driver.close();

driver.quit();
```

NoAlertPresentException

```
When we try to perform an action i.e., either accept() or dismiss()
which is not required at a required place; gives us this exception.

try{

driver.switchTo().alert().accept();

}

catch (NoAlertPresentException E){

E.printStackTrace();

}
```

NoSuchWindowException

```
When we try to switch to an window which is not present gives us
this exception:

WebDriver driver = new InternetExplorerDriver();

driver.get("http://google.com");

driver.switchTo().window("Yup_Fail");

driver.close();

In the above snippet, line 3 throws us an exception, as we are
trying to switch to an window that is not present.
```

NoSuchFrameException

```
Similar to Window exception, Frame exception mainly comes during
switching between the frames.

WebDriver driver = new InternetExplorerDriver();
```

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

driver.switchTo().frame("F_fail");

driver.close();

In the above snippet, line 3 throws us an exception, as we are
trying to switch to an frame that is not present.
```

NoSuchElementException

```
This exception is thrown when we WebDriver doesn't find the web-
element in the DOM.

WebDriver driver = new InternetExplorerDriver();

driver.get("http://google.com");

driver.findElement(By.name("fake")).click();
```

TimeoutException

```
Thrown when a command does not complete in enough time.
```

StaleElementException

Most automation tools depend on the concept of the page has finished loading. With AJAX and Web 2.0 this has become a grey area. META tags can refresh the page and Javascript can update the DOM at regular intervals.

For Selenium this means that StaleElementException can occur. StaleElementException occurs if I find an element, the DOM gets updated then I try to interact with the element.

Actions like:
`driver.findElement(By.id("foo")).click();`
are not atomic. Just because it was all entered on one line, the code generated is no different than:

```
By fooID = By.id("foo");
WebElement foo =
driver.findElement(fooID); foo.click();
```

If Javascript updates the page between the *findElement* call and the *click* call then I'll get a StaleElementException. It is not uncommon for this to occur on modern web pages. It will not happen consistently however. The timing has to be just right for this bug to occur.

Generally speaking, if you know the page has Javascript which automatically updates the DOM, you should assume a StaleElementException will occur. It might not occur when you are writing the test or running it on your local machine but it will happen. Often it will happen after you have 5000 test cases and haven't touched this code for over a year. Like most developers, if it worked yesterday and stopped working today you'll look at what you changed recently and never find this bug.

So how do I handle it? I use the following *click* method:

```
public boolean retryingFindClick(By by) {
         boolean result = false;
        int attempts = 0;
        while(attempts < 2) {
            try {
                driver.findElement(by).click();
                result = true;
                break;
            } catch(StaleElementException e) {
            }
            attempts++;
        }
        return result;
}
```

This will attempt to find and click the element. If the DOM changes between the find and click, it will try again. The idea is that if it failed and I try again immediately the second attempt will succeed. If the DOM changes are very rapid then this will not work. At that point you need to get development to slow down the DOM change so this works or you need to make a custom solution for that particular project.

The method takes as input a locator for the element you want to click. If it is successful it will return true. Otherwise it returns false. If it makes it past the*click* call, it will return true. All other failures will return false.

Personally, I would argue this should always work. If the developers are refreshing the page too quickly then it will be overloading the browser on the client machine.

Question 4: Different types of method in Selenium(Asked)

get()	close()
getCurrentUrl();	quit()
getTitle()	getWindowHandle()
findElements()	getWindowHandles()
findElement()	navigate()
getPageSource()	manage()

Question 5: Can we enter text without using sendKeys()?

Yes by using JavascriptExecutor

```
WebDriver driver = new FirefoxDriver();
JavascriptExecutor executor = (JavascriptExecutor)driver;
executor.executeScript("document.getElementById("textbox_id").value='new value';);
```

Question 6: Write a program for string reverse (Asked)

```
class ReverseString
{
   public static void main(String args[])
   {
      String reverse = "";
      String s="manas is a good boy";
      int length = s.length();
      for ( int i = length - 1 ; i >= 0 ; i-- )
      reverse = reverse + s.charAt(i);
      System.out.println("Reverse of entered string is: "+reverse);
   }
}
```

Question 7: Write a program for Fibonacci.

```
public class fibonacci {

public static void main(String[] args) {

    int a=0;
    int b=1;
    long  c=65;
    System.out.print(a+" "+b);
    for (int d=2;d<c;d++)
    {
        int f=a+b;
        a=b;
        b=f;
        System.out.print(" "+f);

    }
    }
}
```

Question 8: Write a program to check number is Prime.

```
    if (n<2){
        System.out.println("No is not prime");
        }
    else
    {
    for(i=2;i<=n/2;i++)
    {
        res=n%i;
        if(res==0)
        {
            flag=false;
            break;
        }
    }
    if(flag)
        System.out.println(n + " is Prime Number");
    else
        System.out.println(n + " is not Prime Number");
}
```

Question 9: 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();
}
```

Question 10: What is the difference between driver.close() and driver.quit command?

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

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

Question 11: Difference between findElement/findElements ?

findElement () will return only single WebElement and if that element is not located or we use some wrong selector then it will throw NoSuchElement exception.

findElements() will return List of WebElements – for this we need to give locator in such a way that it can find multiple elements and will return you list of webelements then using List we can iterate and perform our operation.

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

 public class findElementDemo {
 public static void main(String[] args) throws Exception {

   WebDriver driver=new FirefoxDriver();
   driver.manage().window().maximize();
   driver.get("http://google.com");
   Thread.sleep(5000);
   List<WebElement> links=driver.findElements(By.xpath(".//*[@id='menu-top']/li"));
   System.out.println("Total element is "+links.size());
Iterator<WebElement> i1=links.iterator();
while(i1.hasNext)
WebElement ele1=i1.next();
String name=ele1.getText();
System.out.println("Elements name is "+name);
 }
 }
}
```

Question 12:How to Count the number of links in a page ?

Iterator and advanced for loop can do similar job; However, the inconsistency on page navigation within a loop can be solved using array concept.

```
private static String[] links = null;
private static int linksCount = 0;
driver.get("www.google.com");
List<WebElement> linksize = driver.findElements(By.tagName("a"));
linksCount = linksize.size();
System.out.println("Total no of links Available: "+linksCount);
links= new String[linksCount];
System.out.println("List of links Available: ");
// print all the links from webpage
for(int i=0;i<linksCount;i++)
{
links[i] = linksize.get(i).getAttribute("href");
System.out.println(all_links_webpage.get(i).getAttribute("href"));
}
// navigate to each Link on the webpage
for(int i=0;i<linksCount;i++)
{
driver.navigate().to(links[i]);
Thread.sleep(3000);
}
```

Capture all links under specific frame|class|id and Navigate one by one

```
driver.get("www.xyz.com");
WebElement element = driver.findElement(By.id(Value));
List<WebElement> elements = element.findElements(By.tagName("a"));
int sizeOfAllLinks = elements.size();
System.out.println(sizeOfAllLinks);
for(int i=0; i<sizeOfAllLinks ;i++)
{
System.out.println(elements.get(i).getAttribute("href"));
}
for (int index=0; index<sizeOfAllLinks; index++ ) {
getElementWithIndex(By.tagName("a"), index).click();
driver.navigate().back();
}

public WebElement getElementWithIndex(By by, int index) {
WebElement element = driver.findElement(By.id(Value));
List<WebElement> elements = element.findElements(By.tagName("a"));
return elements.get(index);
}
```

Capture all links

```java
driver.get(baseUrl + "https://www.google.co.in");
List<WebElement> all_links_webpage = driver.findElements(By.tagName("a"));
System.out.println("Total no of links Available: " + all_links_webpage.size());
int k = all_links_webpage.size();
System.out.println("List of links Available: ");
for(int i=0;i<k;i++)
{
if(all_links_webpage.get(i).getAttribute("href").contains("google"))
{
String link = all_links_webpage.get(i).getAttribute("href");
System.out.println(link);
}
```

Question 13:Does java supports multiple inheritance ?

No, but You can achieve it through interface

Inheritance:

- The concept of getting properties of one class object to another class object is known as inheritance.
- Here properties mean variable and methods.

Types of Inheritance:

- Multiple inheritance.
- Multilevel inheritance.

Multiple inheritance:

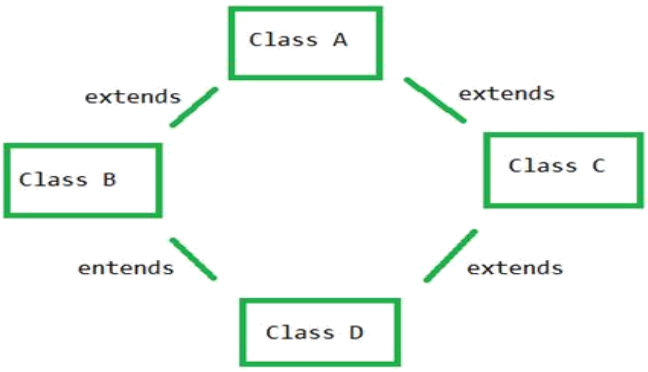
- The concept of getting the properties from multiple class objects to sub class object with same priorities is known as multiple inheritance.
- Java Doesn't Support multiple Inheritance.

Diamond problem:

- In multiple inheritance there is every chance of multiple properties of multiple objects with the same name available to the sub class object with same priorities leads for the ambiguity.

```
1. //Multiple inheritance program
2. Class A{
3. }
4. Class B extends A{
5. public void show(){
6. }
```

```
7.  }
8.  Class C extends A{
9.  public void show(){
10.}
11.}
12.Class D extends B,C{  // not supported by java leads to syntax error.
13.}
```



- We have two classes B and c which are inheriting A class properties.
- Here Class D inheriting B class and C class So properties present in those classes will be available in java.
- But both classes are in same level with same priority.
- If we want to use show() method that leads to ambiguity
- This is called diamond problem.
- Because of multiple inheritance there is chance of the root object getting created more than once.
- Always the root object i.e object of object class hast to be created only once.
- Because of above mentioned reasons multiple inheritance would not be supported by java. Thus
- in java a class cannot extend more than one class simultaneously. At most a class can extend only one class.
- So these are the reasons that java does not supports multiple inheritance. Even though having this much reasons some people say and we also gets some doubts when we are using interfaces. lets me clear this one also. our immediate question should be.

Is Java supports multiple inheritance using interfaces?

- Before that we need to know about interfaces.
- Interfaces having fully abstract functionality.
- Means methods in interfaces by default public abstract methods so we need to implement these methods in classes which are extending this interface.

```
1.  /Multiple inheritance program
2.  interface A{
3.  public void  show();
```

```
4.  }
5.  interface B{
6.  public void  display();
7.  }
8.  Class C Implements A,B{
```

- Here it seems we are achieving multiple inheritance by using interfaces. but we are not.
- Syntactically it seems to multiple inheritance but the actual implementation of multiple inheritance is not there. how can i say that? let me clear

Difference between interfaces and inheritance: in multiple inheritance

- Inheritance means getting the properties from one class object to another class object.
- Means if any class extending another class then the super class methods can be used in sub classes happily.
- But in interfaces the methods in interfaces are fully abstract so we need to provide functionality in our extended class and use it .seems both are opposite right? yes.
- That is the reason we can say interfaces are only supports syntactical multiple inheritance which is not implementation of multiple inheritance.

So finally java supports only syntax of multiple inheritance does not supports implementation of multiple inheritance. Inheritance is like debit and interface is like credit but interface has its own importance in other concepts like server side programming.

Question 14:Difference between assert and verify in selenium Webdriver

- 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.

Question 15:How to take a screen shot using selenium Webdriver

```
import java.io.File;
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.By;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.annotations.Test;

    public class takeScreenShotExample{
        public WebDriver driver;

      @Test
      public void openBrowser() throws Exception {
         driver = new FirefoxDriver();
```

```
        driver.manage().window().maximize();
        driver.get("http://www.google.com");
        try{
              //the below statement will throw an exception as the element is
not found, Catch block will get executed and takes the screenshot.
               driver.findElement(By.id("testing")).sendKeys("test");

                 //if we remove the below comment, it will not return exception
and screen shot method will not get executed.
               //driver.findElement(By.id("gbqfq")).sendKeys("test");
        }
        catch (Exception e){
              System.out.println("I'm in exception");
//calls the method to take the screenshot.
              getscreenshot();
        }
     }

     public void getscreenshot() throws Exception
     {
            File scrFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
          //The below method will save the screen shot in d drive with name
"screenshot.png"
            FileUtils.copyFile(scrFile, new File("D:\\screenshot.png"));
     }
 }
```

Question 16:How will you execute your login script using chrome browser from your editor using selenium?

Download the chromedriver from this path: http://code.google.com/p/chromedriver/downloads/list and store in the directory

- For Chrome browser we need to set the chromedriver path in the editor which is as shown below:

```
System.setProperty("webdriver.chrome.driver","C:\\Users\\Public\\Documents\\S
elenium\\chromedriver.exe"); WebDriver

driver = new ChromeDriver();
```

Question 17:Can we use multiple catch in try, how ?

Yes

```
class TestMultipleCatchBlock1{
```

```
public static void main(String
            args[]){ try{
                        int a[]=new int[5];
                        a[5]=30/0;
                  }
                        catch(Exception e)
                        {System.out.println("common task
                        completed");} catch(ArithmeticException e)
                        {System.out.println("task1 is completed");}
                        catch(ArrayIndexOutOfBoundsException e)
                        {System.out.println("task 2 completed");}
                        System.out.println("rest of the code...");
                  }
            }
```

Question 18:Difference between webdriver/firefoxDriver

interface and instance

WebDriver is an interface which contain several abstract methods such as get(…), findElementBy(…) etc. We simply create reference of web Driver and we can assign objects (Firefox driver, CromeDriver, IEDriver, Andriod driver etc) to it.

Ex :

WebDriver driver = new FireFoxDriver();————(1)

If we are using (1) we can do the same thing by using

FireFoxDriver driver = new FireFoxDriver();————(2)

We can use (1) and (2) for same purpose but if we want to switch to another browser in same program then again we have to create the object of other class as for example

CromeDriver driver = new CromeDriver();.

creating object of several class is not good. So we create the reference of WebDriver and we assign the objects of another class as for example

WebDriver driver; // it is created only one time in the program

driver = new FireFoxDriver();// any where in the program driver = new CromeDriver(); // any where in the program

Question 19: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.Forexample 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

Question 20:How can we get the font size, font colour ,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);
```

Question 21:How do you manage the code versions in your project ?

Using Perforce

Question 22:How do we handle dynamic elements without using X path ?

While automating any web application using any automation tool, be it open-source like selenium webdriver or commercial like UFT/QTP. we have to identify locators for elements which we need to interact with. It could be ID, Name, CSS Selector, XPath or combination of all these. It is quite straight forward to identify locators for static elements which are clearly defined with static IDs. But in some applications, we come across dynamic elements and it becomes quite challenging to identify locators for such dynamic elements.

Dynamic elements are those elements which have identifiers that are dynamically generated. Dynamic identifiers are normally used for buttons, text-fields and buttons. Let us take an example of a button whose ID is in following format…

```
// Dynamic Element Locators
<button id="Submit-901" />
<button id="Submit-902" />
<button id="Submit-903" />
```

In this test scenario, we can observe that element ID is not static. There is a number combined with text that auto increments on user action. So, we can expect that on every script execution there will be a new ID for the element.

There are multiple approaches which can be used to handle dynamic elements but there is no definite one. An approach might work in one scenario and might not work in another. It all depends on the code, element type, locator and test script requirements.

1. Absolute Xpath

Xpath Position or Absolute Xpath are most frequently used to resolve the dynamic element issues. Only problem with using XPath locators is that they are very fragile. They are most prone to breakage in case of change in web page. This factor could get worse exponentially as the test suite size and complexity increases. Below is an example of Absolute XPath and XPath Position

```
web_element_name=html/body/div[30]/div[2]/div[2]/div/div/div/div[1]/table/tbody/tr/td[2]/table/tbody/tr/td[1]/table/tbody/tr/td[1]/table/tbody/tr[2]/td[2]/em/button
```

```
//p[6]/label[2]/div/ins
```

2.Identify Element by starting Text

If the dynamic elements have a definite pattern to them, then we can also use JavaScript functions like “starts-with” or “contains” in our element locators to separate the dynamic part of locator from static part.

For example, in case of dynamic submit button Id example which we discussed earlier, we can apply 'starts-with' function to access this locator irrespective of its dynamic part. XPath: //button[starts-with(@id, 'Submit-')]

3. Identify Element by containing Text

Similarly, in some scenarios where dynamic element is surrounded by a static value, we can use 'contains' function. For example we have following element locators…

```
<input class="new-userfield-001">
<input class="old-userfield-002">
```

As we can see 'usefield' part of element is static, so we can apply 'contains' function to access this element locator as shown below…

XPath: //input[contains(@class, 'suggest')].

4. Identify Element by Index

If there are multiple elements present on page with same locator then we can use following Java code in our selenium WebDriver script to interact with element of particular index.

```
driver.findElements(By.xpath("//*submit")).get(0).click();
```

5. Identify Element with reference of a closest stable element

We can use the DOM structure to find the closest stable element first and then this stable element can be used as a reference element to find the required element.

XPATH: //span1/../following-sibling::div//button1

DOM structure could be found using Firefox extension like Firebug and FirePath. But in complex and large applications this approach is difficult to use because of large DOM structure.

6. Identify Element by stable preceding Text

For web elements like text field and text areas we can identify them by using the stable text labels nearby. This approach might not be possible in all scenarios but it does resolve the dynamic element issues where possible. Example of this approach is shown below.

```
//label1/following::input
```

Question 23: How to handle alerts and confirmation boxes.

Apart from switching between windows and frames, you may have to handle various modal dialogs in a web application. For this, WebDriver provides an API to handle alert dialogs. The API for that is as follows:

Alert alert()

The preceding method will switch to the currently active modal dialog on the web page. This returns an Alert instance where appropriate actions can be taken on that dialog. If there is no dialog currently present, and you invoke this API, it throws back a NoAlertPresentException.

The Alert interface contains a number of APIs to execute different actions. The following list discusses them one after the other:

void accept(): This is equivalent to the **OK** button action on the dialog. The corresponding **OK** button actions are invoked when the accept() action is taken on a dialog.

void dismiss(): This is equivalent to clicking on the **CANCEL** action button.

java.lang.S tring getText(): This will return the text that appears on the dialog. This can be used if you want to evaluate the text on the modal dialog.

void sendKeys(java.lang.S tring keysToSend): This will allow the developer to type in some text into the alert if the alert has some provision for it.

Question 24:How to handle colours in Webdriver ?

Get the value of a given CSS property. Color values should be returned as rgba strings, so, for example if the "background-color" property is set as "green" in the HTML source, the returned value will be "rgba(0, 255, 0, 1)". Note that shorthand CSS properties (e. g. background, font, border, border-top, margin, margin-top, padding, padding-top, list-style, outline, pause, cue) are not returned, in accordance with the DOM CSS2 specification - you should directly access the longhand properties (e.g. background -color) to access the desired values.

Then this is not a Selenium specific question, this is just a general programming question about how to parse string `rgba(102,102,102)` to three number.

```
// Originally untested code, just the logic.
// Thanks for Ripon Al Wasim's correction.

String color = driver.findElement(By.xpath("//div[@class='gb_e gb_f
gb_g gb_xb']/a")).getCssValue("color");

String[] numbers = color.replace("rgba(", "").replace(")",
"").split(","); int r = Integer.parseInt(numbers[0].trim());
int g = Integer.parseInt(numbers[1].trim());
int b = Integer.parseInt(numbers[2].trim());
System.out.println("r: " + r + "g: " + g + "b: " + b);

String hex = "#" + Integer.toHexString(r) +
Integer.toHexString(g) + Integer.toHexString(b);
System.out.println(hex);
```

Question 25:How to press shift +tab

```
public class SendKeys
	{
		public static void main(String[] args)
		{
			WebDriver driver = new FirefoxDriver();
			driver.get("http://www.google.com");
			WebElement searchBox = driver.findElement(By.name("q"));
			searchBox.sendKeys(Keys.chord(Keys.SHIFT, "packt publishing"));
			}
		}
```

Question 26: How to take a screenshot in selenium.

```
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"));
```

Question 27: Is there a way to click hidden link in Webdriver.

There is a easier way to work around the problem using `JavascriptExecutor`.
For example:

```
document.getElementsByClassName('post-tag')[0].click();
```

The above javascript would click on the "Selenium" tag on the top right of this page (next to your question), even if it were hidden.

All you need to do is issue this JS instruction via the `JavascriptExecutor` interface like so:

```
(JavascriptExecutor(webdriver)).executeScript("document.getElementsByClassName('post-tag')[0].click();");
```

Question 28: Login for Gmail scenario ?

```
package login;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
public class Login1 {
public static void main(String[] args) {
// Create a new instance of the Firefox driver
WebDriver driver = new FirefoxDriver();
//  Wait For Page To Load
// Put a Implicit wait, this means that any search for elements on the page
//could take the time the implicit wait is set for before throwing exception
driver.manage().timeouts().implicitlyWait(10,
TimeUnit.SECONDS); // Navigate to URL
driver.get("https://mail.google.com/");
// Maximize the window.
driver.manage().window().maximize();
// Enter UserName
driver.findElement(By.id("Email")).sendKeys(" YOUR USER NAME");
// Enter Password
driver.findElement(By.id("Passwd")).sendKeys("YOUR PASSWORD");
// Wait For Page To Load
driver.manage().timeouts().implicitlyWait(60, TimeUnit.SECONDS);
// Click on 'Sign In' button
driver.findElement(By.id("signIn")).click();
```

```
//Click on Compose Mail.
driver.findElement(By.xpath("//div[
icon present in the top right navigational Bar
driver.findElement(By.xpath("//div[@class='gb_1 gb_3a gb_nc gb_e']/div/a")).click();
//Click on 'Logout' Button@class='z0']/div")).click();
// Click on the image
driver.findElement(By.xpath("//*[@id='gb_71']")).click();
//Close the browser.
driver.close();
}
}
```

Question 29:What are the technical challenges that you faced with selenium?

- There are quite a few technical challenges we faced in the initial stages of implementation; as my insurance application has a restriction – it opens only in Internet Explorer. With this said, locating the UI elements (test objects) via., developer tools & view source which was really challenging.
- Also there are many of the objects which doesn't have the id, name; also the application is mostly in form tables, so we had to go for Xpath-Relative using developer tools post that we found an Fire-IEBrowser1.4 (VB Scripted excel) that helped in getting the Xpath-absolute & relative very easily.
- We had many obstacles in locating few UI Elements, for which we have used AUI.
- It took almost 20 working days to sort out all the issue and finish an end to end business transaction; with all the solutions found to the major problems we had automated all the 21 live stream products within a short span of time.
- Also mention about other challenges in action builder, selenium crashes, non-support to window handles, file uploads, IDE to webdriver conversion challenges etc
- 

Question 30:Difference between flex and flash application.

There is not big difference between the flex and flash. Flash is more oriented or specially designed software for designer because they can create anything without using any code. Coding is the advantage for the designer to save his time but he can also perform the same work without coding. Flash provides number of tools for drawing to create graphics or timeline method for changing those graphics according to the requirement. Flash is used to create web ads, banner for websites, banner for social messages, games, and so on. Using Action Script in the flash we can make our design more lively or realistic. The biggest disadvantage of flash is you have to spend countless hours for creating attractive framework for your project or website. It is really a time consuming process.

Where as flex is more oriented or created to keep in mind the developers. It has includes almost every feature of web development. Developers who have good knowledge of html, css and javascript or little bit designing knowledge can easily understand the flex. You can create graphics by using inbuilt components or functions. The prime function of php is to create business type applications. If you want create complex application in other software it will take number of hours to complete but flex provides the framework applications which includes inbuilt components to design or develop complex application and with the use of components you will save your time. You can use several languages in one application like mxml which is similar to html, action script or php, dotnet and so on. Flex compiles these languages in one SWF file. The only disadvantage of flex is that it doesn't provide any tool for designing but with the use of stylesheet, properties or components we can develop or design attractive design for our applications.

Question 31:What classes extends Webdriver

Interface WebDriver

- **All Superinterfaces:**

  SearchCont ext

  All Known Implementing Classes:

  AndroidDriver, AndroidWebDriver, ChromeDr iver, EventFiringWebDr iver, FirefoxDriver, HtmlUnitDr iver, Inter netExplorer Driver, IPhoneDr iver, IPhoneSimulator Driver, RemoteWebDriver, SafariDr iver

Question 32:What is action class in Webdriver ?

In struts 2, action class is **POJO** (Plain Old Java Object).POJO means you are not forced to implement any interface or extend any class. Generally, **execute** method should be specified that represents the business logic. The simple action class may look like:

```
public class Welcome
 {
public String execute()
{
   return "success";
}
}
```

Question 33:What is selense

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

Question 34:What is SIDE

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.

Interface Hierarchy

- org.testng.annotations.**IAnnotation**
  - org.testng.annotations.**IDataProviderAnnotation**
  - org.testng.annotations.**IExpectedExceptionsAnnotation**
  - org.testng.annotations.**IObjectFactoryAnnotation**
  - org.testng.annotations.**IParameterizable**
    - org.testng.annotations.**IFactoryAnnotation** (also extends org.testng.internal.annotations.IDataProvidable)
    - org.testng.annotations.**ITestOrConfiguration**
      - org.testng.annotations.**IConfigurationAnnotation**
      - org.testng.annotations.**ITestAnnotation** (also extends org.testng.internal.annotations.IDataProvidable)
  - org.testng.annotations.**IParametersAnnotation**
- org.testng.internal.annotations.IDataProvidable
  org.testng.annotations.**IFactoryAnnotation** (also extends org.testng.annotations.IParameterizable)
  org.testng.annotations.**ITestAnnotation** (also extends org.testng.annotations.ITestOrConfiguration)

Question 35:what is the hierarchy of TestNG annotation

Annotation Type Hierarchy

- org.testng.annotations.**Parameters** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**Listeners** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**Test** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**AfterMethod** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**BeforeTest** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**BeforeMethod** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**Optional** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**AfterTest** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**Guice** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**BeforeGroups** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**ExpectedExceptions** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**TestInstance** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**NoInjection** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**AfterSuite** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**AfterClass** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**AfterGroups** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**DataProvider** (implements java.lang.annotation.Annotation)

- org.testng.annotations.**BeforeSuite** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**BeforeClass** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**Factory** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**Configuration** (implements java.lang.annotation.Annotation)
- org.testng.annotations.**ObjectFactory** (implements java.lang.annotation.Annotation)

Question 36: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.

Question 37: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.

Question 38:How to refresh a page without using context click?

1.Using sendKeys.Keys method
driver.get("https://accounts.google.com/SignUp");
driver.findElement(By.id("firstname-placeholder")).sendKeys(Keys.F5);

2.Using navigate.refresh() method
driver.get("http://ruchi-myseleniumblog.blogspot.in/2013/12/100-selenium-interview-questions.html");
driver.navigate().refresh();

3.Using navigate.to() method
driver.get("http://ruchi-myseleniumblog.blogspot.in/2014/01/selenium-hybrid-framework-using.html");
driver.navigate().to(driver.getCurrentUrl());

4.Using get() method
driver.get("http://ruchi-myseleniumblog.blogspot.in/2013/12/basic-core-java-interview-questions.html");
driver.get(driver.getCurrentUrl());

5.Using sendKeys() method
driver.get("https://accounts.google.com/SignUp");
driver.findElement(By.id("firstname-placeholder")).sendKeys("\uE035");

Question 39:How to handle autocomplete box in web driver?

driver.findElement(By.id("your searchBox")).sendKeys("your partial keyword"); Thread.sleep(3000);
List <WebElement> listItems = driver.findElements(By.xpath("your list item locator"));
listItems.get(0).click();
driver.findElement(By.id("your searchButton")).click();

Question 40:Difference between the selenium 1.0 and selenium 2.0 ??

Question What is difference between QTP and Selenium?

Ans:

Category | QTP | Selenium / Web-Driver

License Type | Commercial Software | Open Source

Scripting Language | VB script | Java, Python, Ruby, C#, JS, Perl

Browsers Support | IE, FF(for running), Chrome, Android, iOS | IE, FF, Safari, Opera, Chrome, Android, IOS

Supporting App's | WebApp's, SAP, Activex, Windows, Siebel | Only Web Applications, Flex Supporting OS | Windows | Windows, Linux, Mac

- QTP allows storing of objects using OR and Se works in descriptive mode only as of today.
- Se does not support the window dialog boxes as it can work only in web.
- Se supports the largest number of browsers and multiple OS whereas QTP doesn't
- Selenium support wide programming languages whereas QTP supports Vbscript only.
- QC-QTP integration allows easier management; Se can integrate with Jira, others

What are the disadvantages of using Selenium as testing tool? Ans:

I would say there would be more challenges while using selenium as testing tool rather than disadvantages: below would be some of the major challenges that I know of

- Script Development would be challenging when selenium breaks as we get a minimal or no support at times being opensource.
- Integrating with other external tools such as test management would be challenging.
- Framework development shall be very complex at the beginning as none is available as professional and stable!
- only web, file uploads (window handlers) does not work, changing object properties may be difficult to manage,
- Not so tester friendly at this stage

Question 41:Difference between the getWindowHandle and getWindowHandles.

Method Name: getWindowHandles()

Syntax: Set getWindowHandles()
Example: driver.getWindowHandles();
Purpose: Return a set of window handles which can be used to iterate over all the open windows of this Webdriver instance by passing them to switchTo().WebDriver.Options.window() Returns: A set of window handles which can be used to iterate over all the open windows.

Method Name: getWindowHandle()

Syntax: String getWindowHandle()
Example: driver.getWindowHandle();
Parameter: Return an opaque handle to this window that uniquely identifies it within this driver instance. This can be used to switch to this window at a later date

switchTo
WebDriver. TargetLocator switchTo()
The next future commands will be performed to a different frame or window. Returns:
A Target Locator which can be used to switch or select a frame or window

Commands for different Object Present In UI

Object/Property	WebDriver Command
Text Box	getText(), isDisplayed(), isEnabled(), sendKeys()
List Box	isSelected(), selectByVisibleText(), selectByValue(),selectByIndex(), deselectAll()
Radio Button	isDisplayed(), isEnabled(), click()
Check Box	isDisplayed(), isEnabled(), click()
Button	click(), submit(),
Link	isDisplayed(), isEnabled(), click()
Text	getText()

Class Hierarchy

- java.lang.Object
  - org.openqa.selenium.**By**
    - org.openqa.selenium.**By.ByClassName** (implements java.io.Serializable)
    - org.openqa.selenium.**By.ByCssSelector** (implements java.io.Serializable)
    - org.openqa.selenium.**By.ById** (implements java.io.Serializable)
    - org.openqa.selenium.**By.ByLinkText** (implements java.io.Serializable)
    - org.openqa.selenium.**By.ByName** (implements java.io.Serializable)
    - org.openqa.selenium.**By.ByPartialLinkText** (implements java.io.Serializable)
    - org.openqa.selenium.**By.ByTagName** (implements java.io.Serializable)
    - org.openqa.selenium.**By.ByXPath** (implements java.io.Serializable)
  - org.openqa.selenium.**Cookie** (implements java.io.Serializable)
  - org.openqa.selenium.**Cookie.Builder**
  - org.openqa.selenium.**Dimension**
  - org.openqa.selenium.**Point**
  - org.openqa.selenium.**Proxy**
  - java.lang.Throwable (implements java.io.Serializable)
    - java.lang.Exception
      - java.lang.RuntimeException
        - org.openqa.selenium.**WebDriverException**
          - org.openqa.selenium.**ImeActivationFailedException**
          - org.openqa.selenium.**ImeNotAvailableException**
          - org.openqa.selenium.**InvalidCookieDomainException**
          - org.openqa.selenium.**InvalidElementStateException**
            - org.openqa.selenium.**ElementNotVisibleException**
          - org.openqa.selenium.**NotFoundException**
            - org.openqa.selenium.**NoAlertPresentException**
            - org.openqa.selenium.**NoSuchContextException**
            - org.openqa.selenium.**NoSuchElementException**
              - org.openqa.selenium.**InvalidSelectorException**
            - org.openqa.selenium.**NoSuchFrameException**
            - org.openqa.selenium.**NoSuchWindowException**
          - org.openqa.selenium.**SessionNotCreatedException**
          - org.openqa.selenium.**StaleElementReferenceException**
          - org.openqa.selenium.**TimeoutException**
          - org.openqa.selenium.**UnableToSetCookieException**
          - org.openqa.selenium.**UnhandledAlertException**
          - org.openqa.selenium.**UnsupportedCommandException**

Question 42:Write a code for number of character in string ?

```
String s = "Hello My name is Joel";
int counter = 0;
for( int i=0; i<s.length(); i++ ) {
    if( s.charAt(i) == 'l' ) {
        counter++;
    }
}
```

There are multiple solutions. For such a low level task you should probably do a performance test to see which method is faster. For instance, regular expressions are very slow in java.

Implementation 1[edit]

This code will count the number of commas in a string.

```
      String s = "Count, the number,, of commas.";
      int numberOfCommas = s.replaceAll("[^,]","").length();
```

Implementation 2[edit]

```
public int count(String input, String countString){
  return input.split("\\Q"+countString+"\\E", -1).length - 1;
}
```

Implementation 3[edit]

```
public int count(String sourceString, char
    lookFor) { if (sourceString == null) {
        return -1;
    }
    int count = 0;
    for (int i = 0; i < sourceString.length(); i++)
        { final char c = sourceString.charAt(i);
        if (c == lookFor) {
            count++;
        }
    }
    return count;
}
```

Implementation 4[edit]

```
public static int count(String word, Character ch)
{
      int pos = word.indexOf(ch);
      return pos == -1 ? 0 : 1 + count(word.substring(pos+1),ch);
}
```

Question 43:How to Read particular cell from html table ?

```
public class SeleniumTable {

      public static void main(String[] args)
      {
            WebDriver driver = new InternetExplorerDriver();
            driver.get("http://localhost/test/test.html");

            WebElement table_element =
            driver.findElement(By.id("testTable")); List<WebElement>
tr_collection=table_element.findElements(By.xpath("id('testTable')/tbody/tr"));

            System.out.println("NUMBER OF ROWS IN THIS
TABLE = "+tr_collection.size());
            int row_num,col_num;
            row_num=1;
            for(WebElement trElement : tr_collection)
            {
                  List<WebElement>
td_collection=trElement.findElements(By.xpath("td"));
                  System.out.println("NUMBER OF COLUMNS="+td_collection.size());
                  col_num=1;
```

```
                    for(WebElement tdElement : td_collection)
                    {
                          System.out.println("row # "+row_num+", col #
"+col_num+ "text="+tdElement.getText());
                          col_num++;
                    }
                    row_num++;
              }
        }
}
```

Question 44:How to Connect Java Application with Oracle database

There are 5 steps to connect any java application with the database in java using JDBC. They are as follows:

- Register the driver class
- Creating connection
- Creating statement
- Executing queries
- Closing connection

Example to Connect Java Application with Oracle database

In this example, system is the username and oracle is the password of the Oracle database.

```
import java.sql.*;
class OracleCon{
public static void main(String args[]){
try{
//step1 load the driver class
Class.forName("oracle.jdbc.driver.OracleDriver");

//step2 create  the connection object
Connection con=DriverManager.getConnection(

"jdbc:oracle:thin:@localhost:1521:xe","system","oracle");

//step3 create the statement object
Statement stmt=con.createStatement();

//step4 execute query
ResultSet rs=stmt.executeQuery("select * from
emp"); while(rs.next())
System.out.println(rs.getInt(1)+"  "+rs.getString(2)+"  "+rs.getString(3));

//step5 close the connection object
con.close();
```

```
}catch(Exception e){ System.out.println(e);}

}
}
```

Question 45: How to switch between frames in Selenium WebDriver?

WebDriver's `driver.switchTo().frame()` method takes one of the three possible arguments:

- A number.
  Select a frame by its (zero-based) index. That is, if a page has three frames, the first frame would be at index `0`, the second at index `1` and the third at index `2`. Once the frame has been selected, all subsequent calls on the WebDriver interface are made to that frame.
- A name or ID.
  Select a frame by its name or ID. Frames located by matching name attributes are always given precedence over those matched by ID.
- A previously found `WebElement`.
  Select a frame using its previously located WebElement.

In Selenium to work with iFrames, we have different ways to handle frame depending on the need. Please look at the below ways of handling frames

driver.switchTo().frame(int arg0);

Select a frame by its (zero-based) index. That is, if a page has multiple frames (more than 1), the first frame would be at index "0", the second at index "1" and so on. Once the frame is selected or navigated , all subsequent calls on the WebDriver interface are made to that frame. i.e the driver focus will be now on the frame. What ever operations we try to perform on pages will not work and throws element not found as we navigated / switched to Frame.

Parameters: Index - (zero-based) index
Returns: driver focused on the given frame (current frame)
Throws: NoSuchFrameException - If the frame is not found.

Example: if iframe id=webklipper-publisher- widget-container-frame, it can be written as driver.switchTo().frame("webklipper-publisher- widget-container-frame"); Below is the code snippet to work with switchToFrame using frame id.

```
public void switchToFrame(int frame) {
                    try {
                              driver.switchTo().frame(frame);
```

```
                        System.out.println("Navigated to frame with id " + frame);
                } catch (NoSuchFrameException e) {
                        System.out.println("Unable to locate frame with id " +
                                        frame + e.getStackTrace());
                } catch (Exception e) {
                        System.out.println("Unable to navigate to frame with id " +
                                        frame + e.getStackTrace());
                }
        }
```

driver.switchTo().frame(String arg0);

Select a frame by its name or ID. Frames located by matching name attributes are always given precedence over those matched by ID.
Parameters: name Or Id - the name of the frame or the id of the frame element.
Returns: driver focused on the given frame (current frame)
Throws: NoSuchFrameException - If the frame is not found

Below is the example code snippet using frame name.

```
public void switchToFrame(String frame) {
                try {
                        driver.switchTo().frame(frame);
                        System.out.println("Navigated to frame with name " +
                frame); } catch (NoSuchFrameException e) {
                        System.out.println("Unable to locate frame with id " +
                                        frame + e.getStackTrace());
                } catch (Exception e) {
                        System.out.println("Unable to navigate to frame with id " +
                                        frame + e.getStackTrace());
                }
        }
```

driver.switchTo().frame(WebElement frameElement);

Select a frame using its previously located WebElement.
Parameters: frameElement - The frame element to switch to.
Returns: driver focused on the given frame (current frame).
Throws: NoSuchFrameException - If the given element is neither an iframe nor a frame element.
And StaleElementReferenceException - If the WebElement has gone stale.

Below is the example code to send an Element to the and switch.

```
public void switchToFrame(WebElement frameElement) {
```

```
                try {
                        if (isElementPresent(frameElement)) {
                                driver.switchTo().frame(frameElement);
                                System.out.println("Navigated to frame with element "+
frameElement);
                        } else {
                                System.out.println("Unable to navigate to frame with
element "+ frameElement);
                        }
                } catch (NoSuchFrameException e) {
                        System.out.println("Unable to locate frame with element "
+ frameElement + e.getStackTrace());
                } catch (StaleElementReferenceException e) {
                        System.out.println("Element with " + frameElement + "is
not attached to the page document" + e.getStackTrace());
                } catch (Exception e) {
                        System.out.println("Unable to navigate to frame with element "
+ frameElement + e.getStackTrace());
                }
        }
```

Some times when there are multiple Frames (Frame in side a frame), we need to first switch to the parent frame and then we need to switch to the child frame. below is the code snippet to work with multiple frames.

```
public void switchToFrame(String ParentFrame, String ChildFrame)
                { try {
                        driver.switchTo().frame(ParentFrame).switchTo().frame(ChildFrame);
                        System.out.println("Navigated to innerframe with id " + ChildFrame
                                        + "which is present on frame with id" +
ParentFrame);
                } catch (NoSuchFrameException e) {
                        System.out.println("Unable to locate frame with id " + ParentFrame +
                                        " or " + ChildFrame + e.getStackTrace());
                } catch (Exception e) {
                        System.out.println("Unable to navigate to innerframe with id "
                                        + ChildFrame + "which is present on frame with
id"
                                        + ParentFrame + e.getStackTrace());
                }
        }
```

After working with the frames, main important is to come back to the web page. if we don't switch back to the default page, driver will throw an exception. Below is the code snippet to switch back to the default content.

```
public void switchtoDefaultFrame() {
                    try {
                              driver.switchTo().defaultContent();
                              System.out.println("Navigated back to webpage from frame");
                    } catch (Exception e)
                              { System.out
                                                  .println("unable to navigate back to main
webpage from frame"
                                                                        + e.getStackTrace());
                    }
          }
```

Question 46: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

Question 47:How to handle ajax popup window

AJAX stands for Asynchronous JavaScript and AJAX allows the Web page to retrieve small amounts of data from the server without reloading the entire page. In AJAX driven web applications, data is retrieved from server without refreshing the page.

When we perform any action on Ajax controls, using Wait commands will not work as the page is not actually refreshed here. Pausing the test execution using threads for a certain period of time is also not a good approach as web element might appear later or earlier than the stipulated period of time depending on the system's responsiveness, load or othmoment,r leadsuncontrolled to test failures.

The best approach would be to wait for the required element in a dynamic period and then continue the test execution as soon as the element is found/visible.

This can done achieved with WebDriverWait in combination with ExpectedCondition , the best way to wait for an element dynamically, checking for the condition every second and continuing to the next command in the script as soon as the condition is met.

There are many methods which are available to use with wait.until(ExpectedConditions.anyCondition); The below is the image for the number of methods which are available.

The below are the few which we use regularly when testing an application : -

Syntax:

```
WebDriverWait wait = new WebDriverWait(driver, waitTime);
wait.until(ExpectedConditions.presenceOfElementLocated(locator));
```

The above statement will check for the element presence on the DOM of a page. This does not necessarily mean that the element is visible.

Syntax:

```
WebDriverWait wait = new WebDriverWait(driver, waitTime);
wait.until(ExpectedConditions.visibilityOfElementLocated(locator));
```

The above syntax will for the element present in the DOM of a page is visible.

Some times we may also need to check if the element is invisible or not. To do this we need use the below :

Syntax:

```
WebDriverWait wait = new WebDriverWait(driver, waitTime);
wait.until(ExpectedConditions.invisibilityOfElementLocated(locator));
```

Some times you will get an exception as ""'org.openqa.selenium.WebDriverException: Element is not clickable at point (611, 419). Other element would receive the click:'. The below one is used to wait for the element to be clickable.

Syntax:

```
WebDriverWait wait = new WebDriverWait(driver, waitTime);
wait.until(ExpectedConditions.elementToBeClickable(locator));
```

Please click here for WebDriver Waits example and Synchronization in WebDriver

Below is the example program to handle Ajax controls with Wait statements. Based on the application loading time, we can increase or decrease the wait time.

```
package com.pack.ajax;


import org.openqa.selenium.By;
```

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait; import
org.testng.Assert;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;

public class AjaxExample {

        private String URL = "http://demos.telerik.com/aspnet-ajax/
                                ajax/examples/loadingpanel/explicitshowhide/defaultcs.aspx";

        WebDriver driver;
        WebDriverWait wait;

        @BeforeClass
        public void setUp() {
                driver=new FirefoxDriver();
                driver.manage().window().maximize();
                driver.navigate().to(URL);
        }

        @Test
        public void test_AjaxExample() {

                /*Wait for grid to appear*/
                By container = By.cssSelector(".demo-container");
                wait = new WebDriverWait(driver, 5);
                wait.until(ExpectedConditions.presenceOfElementLocated(container));

                /*Get the text before performing an ajax call*/
                WebElement noDatesTextElement =
driver.findElement(By.xpath("//div[@class='RadAjaxPanel']/span"));
                String textBeforeAjaxCall = noDatesTextElement.getText().trim();

                /*Click on the date*/
                driver.findElement(By.linkText("1")).click();
```

```
                    /*Wait for loader to disappear */
                    By loader = By.className("raDiv");
                    wait.until(ExpectedConditions.invisibilityOfElementLocated(loader));

                    /*Get the text after ajax call*/
                    WebElement selectedDatesTextElement =
driver.findElement(By.xpath("//div[@class='RadAjaxPanel']/span"));
                    wait.until(ExpectedConditions.visibilityOf(selectedDatesTextElement));
                    String textAfterAjaxCall = selectedDatesTextElement.getText().trim();

                    /*Verify both texts before ajax call and after ajax call text.*/
                    Assert.assertNotEquals(textBeforeAjaxCall, textAfterAjaxCall);

                    String expectedTextAfterAjaxCall = "Thursday, January 01, 2015";

                    /*Verify expected text with text updated after ajax call*/
                    Assert.assertEquals(textAfterAjaxCall, expectedTextAfterAjaxCall);
          }

}
```

Question 48:Difference Between Interface and Abstract Class

- Abstract classes can have *con stants, members, method stubs (methods without a body) and defined methods*, whereas interfaces can only have *constants* and *methods stubs*.

- Methods and members of an abstract class can be defined with *any visibility*, whereas all methods of an interface must be defined as `public` (they are defined public by default).

- When inheriting an abstract class, a *concrete* child class *must define the abstract methods*, whereas an an abstract class can extend another abstract class and abstract methods from the parent class don't have to be defined.
- Similarly, an interface extending another interface is *not responsible for implementing methods*from the parent interface. This is because interfaces cannot define any implementation.
- A child class can only *extend a single class* (abstract or concrete), whereas an interface can extend or a class can *implement multiple other interfaces*.
- A child class can define abstract methods with the *same or less restrictive visibility*, whereas a class implementing an interface must define the methods with the exact same visibility (public)

=>Use of xpath functions or 1 value is changing dynamically in xpath & handle it in for loop, findElements syntax, looping

Given 2 java classes. In one class there are 5-6 method. From class 2 call all methods defined in class1 one by one

⇨ Ask not to create object of class1. Without creating object call the methods. [Java reflection]

There is sticky header on page & element hides behind that header on page scroll. Click on such element

=>Syntax of java script executor, Page scrolling by finding X,Y co-ordinates

There is drop down & depending on drop down selection value in text box is loading(ajax loading). Verify selection in both fields are correct. .(Asked)

=>Check code for synchronization (Explicit wait) code & logic to compare both values

Question 49:Read excel & fill application form. Click submit & verify success page. Take screen shot on failure.(Asked)

Code for Excel(POI) api to read excel, code for assertions, TakeScreenshot syntax

- HSSF (Horrible SpreadSheet Format) – reads and writes Microsoft Excel (XLS) format files. It can read files written by Excel 97 onwards; this file format is known as the *BIFF 8* format. As the Excel file format is complex and contains a number of tricky characteristics, some of the more advanced features cannot be read.
- XSSF (XML SpreadSheet Format) – reads and writes Office Open XML (XLSX) format files. Similar feature set to HSSF, but for Office Open XML files

```
public class Excel1 {

 public static void main(String []args){

  try {
  // Specify the path of file
  File src=new File("filepath/excelsheetname.xlsx");

   // load file
   FileInputStream fis=new FileInputStream(src);

   // Load workbook
   XSSFWorkbook wb=new XSSFWorkbook(fis);

   // Load sheet- Here we are loading first
      sheetonly XSSFSheet sh1= wb.getSheetAt(0);

  // getRow() specify which row we want to read.
  // and getCell() specify which column to read.
  // getStringCellValue() specify that we are reading String data.
```

```
 System.out.println(sh1.getRow(0).getCell(0).getStringCellValue());
  } catch (Exception e) {
   System.out.println(e.getMessage());
  }
public class Excel2 {
      public static void main(String []args){
             try {
                   // Specify the file path which you want to create or
write File src=new File("./testdata/test.xlsx");
                   // Load the file

                   // load the workbook
XSSFWorkbook wb=new XSSFWorkbook(fis);
             // get the sheet which you want to modify or
create XSSFSheet sh1= wb.getSheetAt(0);
// getRow specify which row we want to read and getCell which column

                   // here createCell will create column
                   // and setCellvalue will set the value
sh1.getRow(0).createCell(2).setCellValue("2.41.0");
sh1.getRow(1).createCell(2).setCellValue("2.5");
sh1.getRow(2).createCell(2).setCellValue("2.39");

                   // here we need to specify where you want to save file
FileOutputStream fout=new FileOutputStream(new File("location
of file/filename.xlsx"));

                   // finally write content
wb.write(fout);
                   // close the file
```

```
fout.close();
 }
            catch (Exception e) {

System.out.println(e.getMessage());

                    }}}
```

Double click on element after which application opens 2 Windows. Perform this scenario & then enter value in 2nd window.

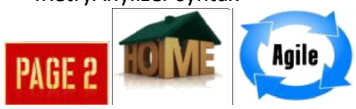
⇨ Action class code to double click & getWindowHandles code

Syntax to add browser capability. Code to set security zones at same level in IE

⇨ Code to add capability, capability name to set security zone

TestNG-Code to Rerun failed test cases only

⇨ IRetryAnylizer syntax



Question 50:Retry executing only Failed Tests using TestNG

There may be many reasons for a Test case getting failed, may be due to element not found or time out exception or stale element exception etc. Normally in automation after executing scripts/tests, we will check for the results and if the test fails because of above reasons we will re -run then again.

Instead of that we can ask testNG to execute the failed test cases again for X (we can define) number of times and check for the updated results.

To achieve this we need to implement TestNG IRetryAnalyzer. Below is the simple code snippet:

```
package com.pack.test;


import org.testng.IRetryAnalyzer;
import org.testng.ITestResult;


public class Retry implements IRetryAnalyzer {
    private int retryCount = 0;
    private int maxRetryCount = 1;


// Below method returns 'true' if the test method has to be retried else 'false'
//and it takes the 'Result' as parameter of the test method that just ran
    public boolean retry(ITestResult result)
        { if (retryCount < maxRetryCount) {
            System.out.println("Retrying test " + result.getName() + " with status "
```

```
                    + getResultStatusName(result.getStatus()) + " for the " + (retryCount+1) + "
time(s).");
            retryCount++;
            return true;
        }
        return false;
    }

    public String getResultStatusName(int status) {
            String resultName = null;
            if(status==1)
                    resultName = "SUCCESS";
            if(status==2)
                    resultName = "FAILURE";
            if(status==3)
                    resultName = "SKIP";
                    return resultName;
    }
}
```

Create an other class 'RetryListener' by implementing 'IAnnotationTransformer'. We need to setRetryAnalyzer for iTestAnnotation. In the example below, add the above class name.

```
package com.pack.test;

import java.lang.reflect.Constructor;
import java.lang.reflect.Method;

import org.testng.IAnnotationTransformer;
import org.testng.IRetryAnalyzer;
import org.testng.annotations.ITestAnnotation;

public class RetryListener implements IAnnotationTransformer {

        @Override
        public void transform(ITestAnnotation testannotation, Class testClass,
                        Constructor testConstructor, Method testMethod)    {
                IRetryAnalyzer retry = testannotation.getRetryAnalyzer();

                if (retry == null)   {
                        testannotation.setRetryAnalyzer(Retry.class);
                }

        }
```

```
}
```

Let us see the example by executing simple tests below. In the below test verifyForgotPasswordPage, We are trying to verify the wrong text which will return False and the Test fails.

Now the second test will be executed for the value that we defined for 'maxRetryCount' in Retry Class.

```
package com.pack.test;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.Assert;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;

public class TestNGExampleTests {
        WebDriver driver;
        String baseURL = "https://www.linkedin.com/";

        @BeforeClass
        public void setup() {
                driver = new FirefoxDriver();
                driver.manage().window().maximize();
        }

        @Test(priority=1)
        public void verifyLoginPageText() {
                driver.navigate().to(baseURL);
                System.out.println("Verify login page test started");
                WebElement element = driver.findElement(By.cssSelector(".header>h2"));
                String headerText = element.getText();
                Assert.assertEquals(headerText, "Get started – it’s free.");
        }

        @Test(priority=2)
        public void verifyForgotPasswordPage() {
                driver.navigate().to(baseURL);
                System.out.println("Verify Forgot password page test started");
                WebElement element = driver.findElement(By.linkText("Forgot your
                password?")); element.click();
                WebElement pageTextElement = driver.findElement(By.cssSelector(".flow-login-
content>fieldset>h1"));
                String pageText = pageTextElement.getText();
```

```
                    Assert.assertEquals(pageText, "Wrong text");
                    //Assert.assertEquals(pageText, "Change your password");
          }
}
```

After executing the above program, We need to add the Listener to testng.xml file. Below is syntax to add listener for RetryListener.

```
<listeners>
        <listener class-name="com.pack.test.RetryListener"/>
  </listeners>
```

After executing the above program, the output should look like below. The verifyForgotPasswordPage test will be executed for two times as we have defined 'maxRetryCount = 1', hence when the test fails, it will execute again for one time. Now the total count of tests it will show as 3, Failures 2. But actually we have only two tests. To handle this we need to adjust the count by implementing 'ITestListener' and update the count from 'onFinish' method.

```
[TestNG] Running:
  D:\Selenium\TestNgSample\testng.xml


Verify login page test started
Verify Forgot password page test started
Retrying test verifyForgotPasswordPage with status FAILURE for the 1 time(s).
Verify Forgot password page test started


===============================================
Parallel test runs
Total tests run: 3, Failures: 2, Skips: 0
===============================================
```

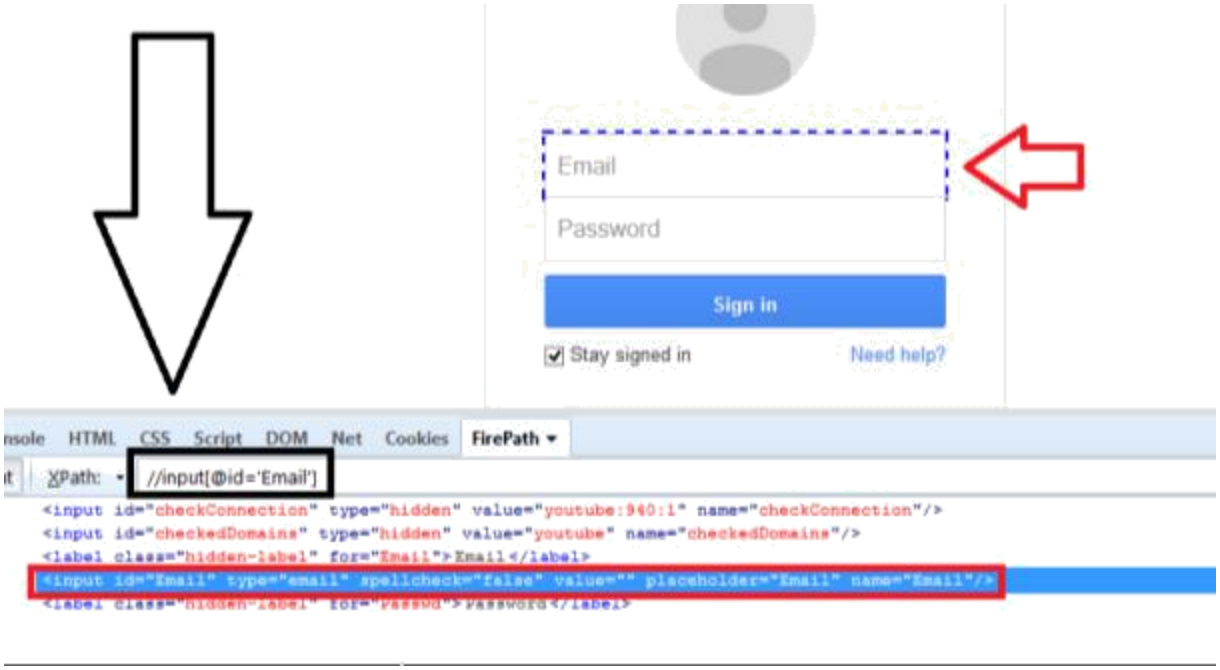
Question 51:Relative XPath method

Using single attribute

1 // tagname[@attribute-name=’value1’]
2
3
4
5 Example
6
7 // a [@href=’http://www. google.com’]
8
9
10//input[@id=’name’]
11
12
13//input[@name=’username’]

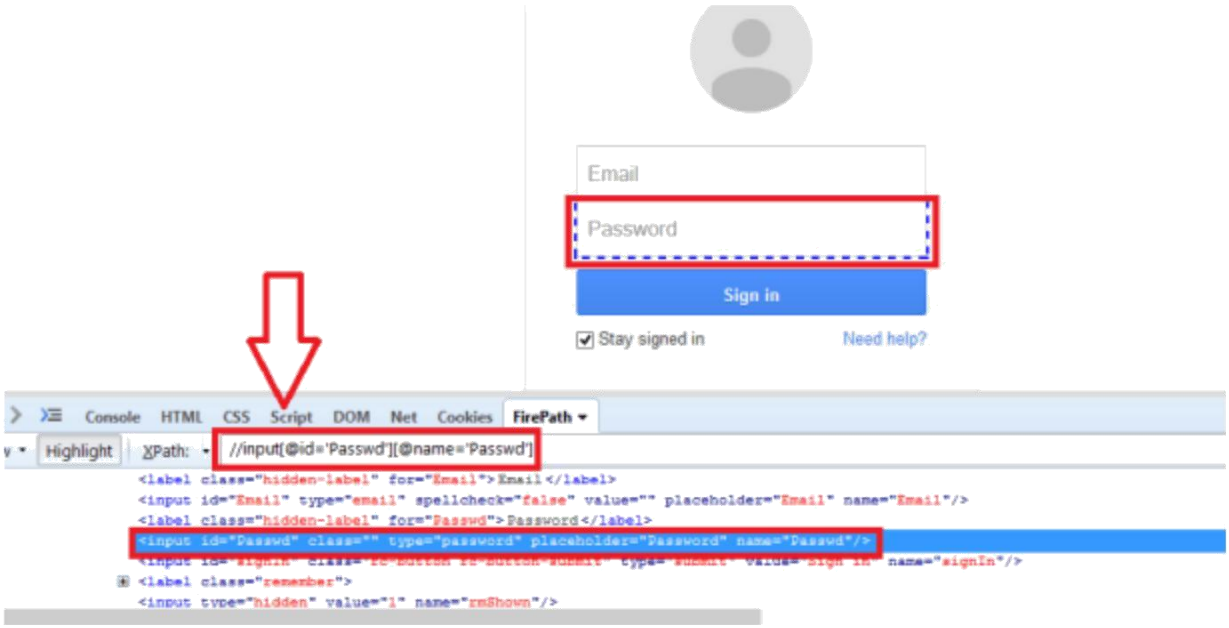
```
14
15
16      //img[@alt='sometext']
```

And so on.



Using multiple attribute

```
1       //tagname[@attribute1='value1'][attribute2='value2']
2
3
4
5       //a[@id='id1'][@name='namevalue1']
6
7       //img[@src=''][@href='']
```



And so on.

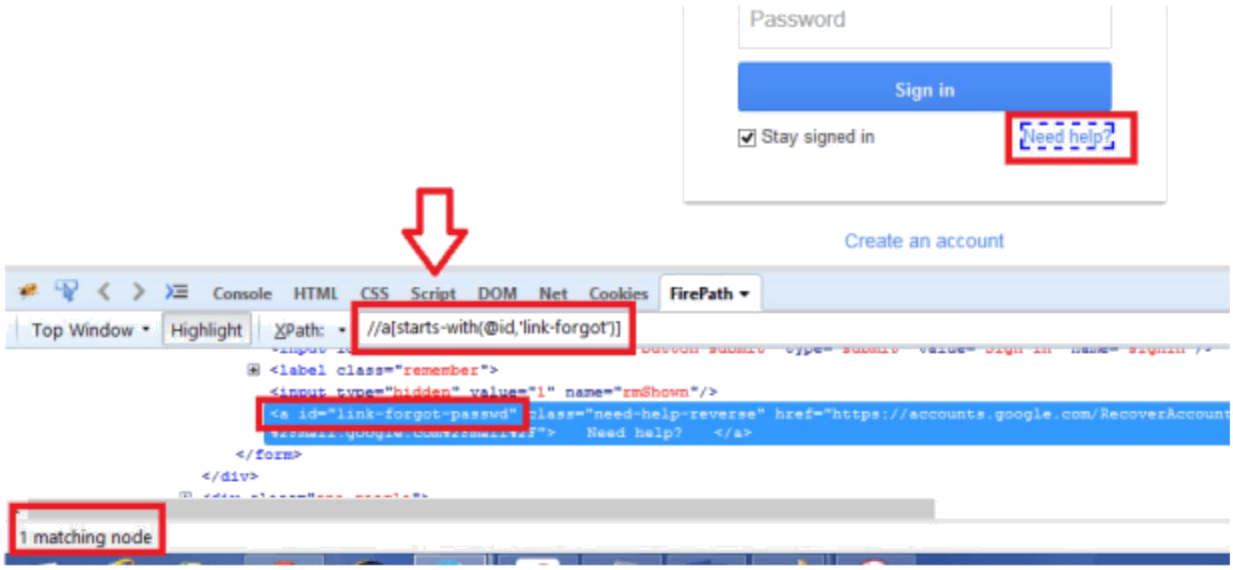
Using contains method

```
1       Syntax
2
3       //tagname[contains(@attribute,'value1')]
4
5       //input[contains(@id,'')]
6
7       //input[contains(@name,'')]
8
9       //a[contains(@href,'')]
10
11//img[contains(@src,'')]
12
13//div[contains(@id,'')]
```

And so on.

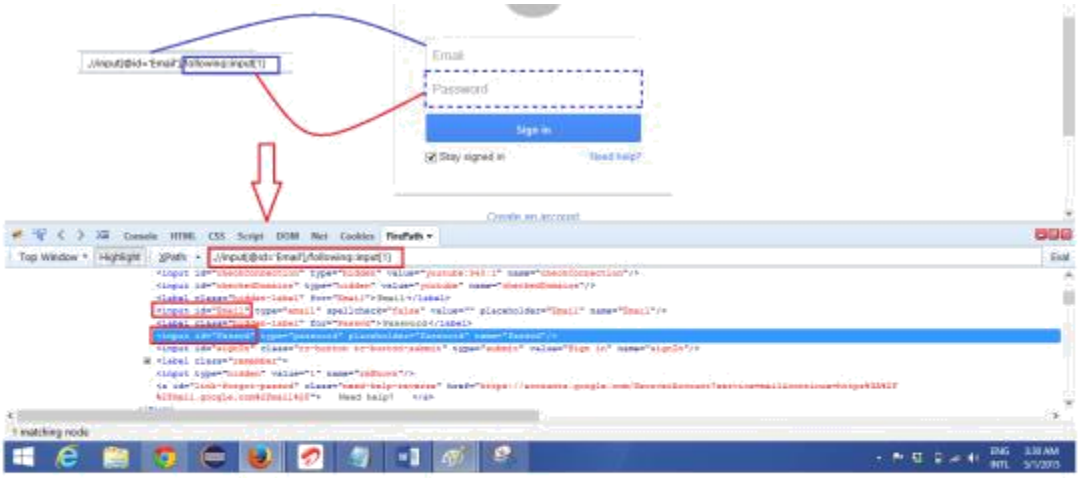
Using starts-with method

```
1       //tagname[starts-with(@attribute-name,'')]
2
3
4
5       //id[starts-with(@id,'')]
6
7       //a[starts-with(@href='')]
8
9       //img[starts-with(@src='')]
10
11//div[starts-with(@id='')]
12
13//input[starts-with(@id='')]
14
15//button[starts-with(@id,'')]
16
17And so on.
```



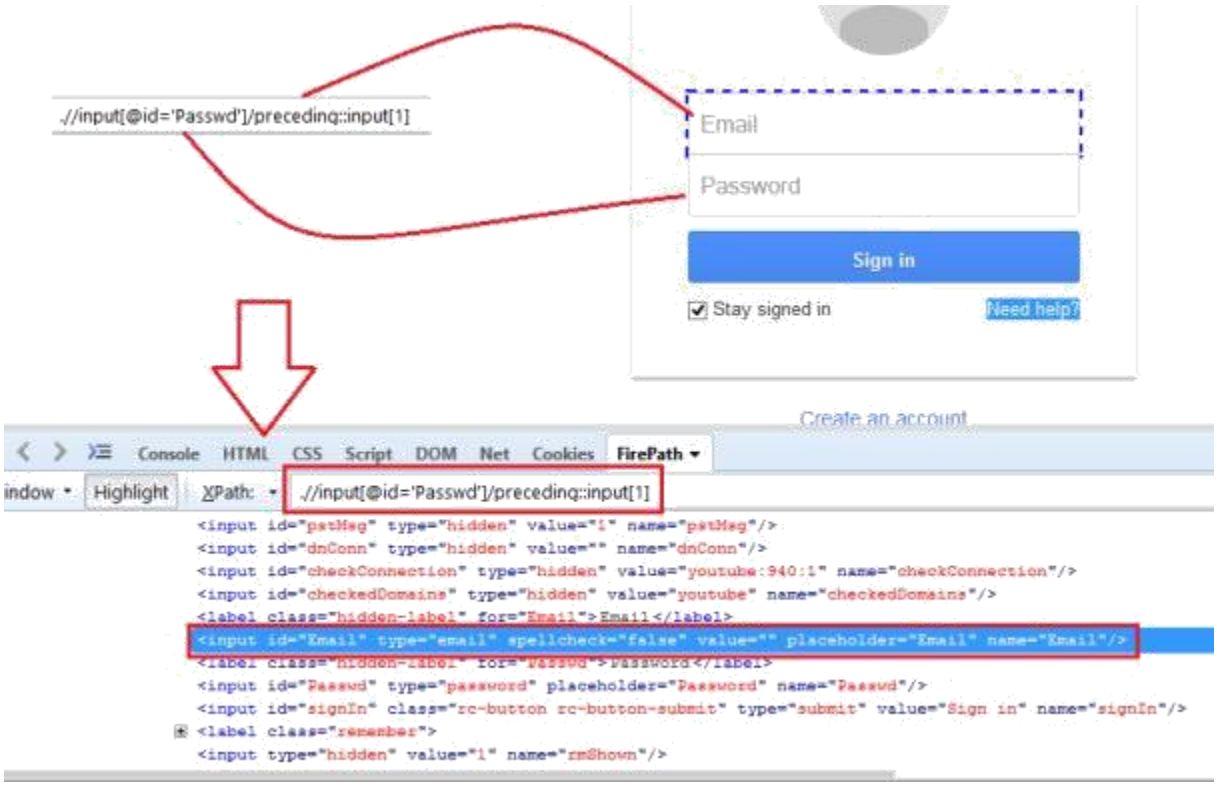
Using Following node

```
1   Xpath/following::again-ur-regular-path
2
3   //input[@id=’’]/following::input[1]
4
5   //a[@href=’’]/following::a[1]
6
7   //img[@src=’’]/following::img[1]
```



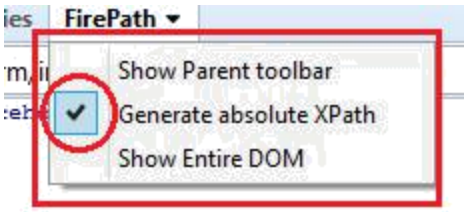
Using preceding node

1 Xpath/preceding::again-ur-regular-path
2
3 //input[@id=’’]/ preceding::input[1]
4
5 //a[@href=’’]/ preceding::a[1]
6
7 //img[@src=’’]/ preceding::im g[1]



Absolute XPath method

1 -/html/head/body/div/input



Relative and Absolute XPath method

```
1    //parent -xpath/absolute xpath
2
3    //input[@id='section']/div/input
```

Question 52: Parallel Execution of test methods in TestNG

TestNG provides an ability to run test methods, test classes and tests in parallel. By using parallel execution, we can reduce the 'execution time' as tests are started and executed simultaneously in different threads.

In testNG we can achieve parallel execution by two ways. One with testng.xml file and we can Configure an independent test method to run in multiple threads.

First let us look at basic example for Parallel Execution of Test Methods using testng.xml.

We will create a class with Two test methods and try to execute in different threads.

```
package com.parallel;
import org.testng.annotations.Test;


public class TestParallelOne {
```

```
        @Test
        public void testCaseOne() {
                //Printing Id of the thread on using which test method got
                executed System.out.println("Test Case One with Thread Id:- "
                                        + Thread.currentThread().getId());
        }

        @Test
        public void testCaseTwo() {
                ////Printing Id of the thread on using which test method got
                executed System.out.println("Test Case two with Thread Id:- "
                                        + Thread.currentThread().getId());
        }
}
```

The below is the simple testng.xml file, if you observe, we are defining two attributes 'parallel' and 'thread-count' at suite level. As we want test methods to be executed in parallel, we have provided 'methods'. And 'thread-count' attribute is to used to pass the number of maximum threads to be created.

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Parallel test suite" parallel="methods" thread-
  count="2"> <test name="Regression 1">
    <classes>
      <class name="com.parallel.TestParallelOne"/>
    </classes>
  </test>
</suite>
```

Now run the above example and see the output, it should look like below :

```
<terminated> testng.xml [TestNG] C:\Program Files\Java\jre7\bin\javaw.exe
[TestNG] Running:
  D:\Dev\TestNGParallel\testng.xml

Test Case One with Thread Id:- 9
Test Case two with Thread Id:- 10

===============================================
Parallel test suite
Total tests run: 2, Failures: 0, Skips: 0
===============================================
```

The above result shows that two methods executed using different threads. When we run the same testng.xml again, the result may vary. The assigning of the thread is take care by the processor. So we can't say which thread is going to execute which method.

If say example, now there is an other test method which is added in the class.

Now if we execute the same testng.xml, we will get the output as below:

```
<terminated> testng.xml [TestNG] C:\Program Files\Java\jre7\bin\javaw.exe
[TestNG] Running:
  D:\Dev\TestNGParallel\testng.xml

Test Case two with Thread Id:- 10
Test Case One with Thread Id:- 9
Test Case two with Thread Id:- 9

===============================================
Parallel test suite
Total tests run: 3, Failures: 0, Skips: 0
===============================================
```

Here Two threads are created. Which ever thread completes the execution of one method, will pick and execute the other test method.

Question 53:Parallel Execution of Classes in TestNG

TestNG provides an ability to run test classes in parallel. By using parallel execution of classes, each class will be started and executed simultaneously in different threads.

Let us look at basic example for Parallel Execution of Classes using testng.xml.

We will create a Two classes with Two test methods each and try to execute in different threads.

Create class and name it as : TestParallelClassOne.java

```
package com.parallel;


import org.testng.annotations.Test;


public class TestParallelClassOne {


        @Test
        public void testCaseOne() {
                // Printing class name and Id of the thread on using which test method got
executed
```

```
                System.out.println("Test Case One in " + getClass().getSimpleName()
                                + " with Thread Id:- " + Thread.currentThread().getId());
        }

        @Test
        public void testCaseTwo() {
                 //Printing class name and Id of the thread on using which test method got
executed
                System.out.println("Test Case two in " + getClass().getSimpleName()
                                + " with Thread Id:- " + Thread.currentThread().getId());
        }

}
```

Create class and name it as : TestParallelClassTwo.java

```
package com.parallel;

import org.testng.annotations.Test;

public class TestParallelClassTwo {

        @Test
        public void testCaseOne() {
                //Printing class name and Id of the thread on using which test method got
executed
                System.out.println("Test Case One in " + getClass().getSimpleName()
                                + " with Thread Id:- " + Thread.currentThread().getId());
        }

        @Test
        public void testCaseTwo() {
                //Printing class name and Id of the thread on using which test method got
executed
                System.out.println("Test Case Two in " + getClass().getSimpleName()
                                + " with Thread Id:- " + Thread.currentThread().getId());
        }

}
```

The below is the testng.xml file

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Parallel test suite" parallel="classes" thread-count="2">
```

```
  <test name="Test 1">
    <classes>
      <class name="com.parallel.TestParallelClassOne"/>
      <class name="com.parallel.TestParallelClassTwo"/>
    </classes>
  </test>
</suite>
```

In the above testng.xml file, we have defined two attributes 'parallel' and 'thread-count' at suite level. As we want classes to be executed in parallel, we have provided 'parallel="classes". And 'thread - count' attribute is to used to pass the number of maximum threads to be created.

Now run the above example and see the output, it should look like below :

```
<terminated> testng.xml [TestNG] C:\Program Files\Java\jre7\bin\javaw.exe ((
  D:\Dev\TestNGParallel\testng.xml

Test Case One in TestParallelClassOne with Thread Id:- 9
Test Case One in TestParallelClassTwo with Thread Id:- 10
Test Case two in TestParallelClassOne with Thread Id:- 9
Test Case Two in TestParallelClassTwo with Thread Id:- 10

===============================================
Parallel test suite
Total tests run: 4, Failures: 0, Skips: 0
===============================================
```

The above result shows that two classes executed using different threads. When we run the same testng.xml again, the result may vary. As The thread scheduler dispatches the various threads on the available processors, and each thread gets some processor time, each in his turn. But the processor, the order and the time assigned to each thread is up to the OS thread scheduler, and we does not guarantee the order of execution

Question 54:Testing in multiple browsers using selenium and testng

In testing, it is always important to test application in different browsers. We can perform automation on multiple browsers using selenium and testng. If there are more number of tests that need to be executed parallelly on different browsers also, we can do this using testng. We will look into the below examples in detail:

Below is the sample test which will only run one browser at a time.

```
package com.pack;


import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebDriverException;
import org.openqa.selenium.chrome.ChromeDriver;
```

```
import org.openqa.selenium.firefox.FirefoxDriver; import
org.openqa.selenium.ie.InternetExplorerDriver; import
org.testng.Assert;
import org.testng.annotations.AfterTest;
import org.testng.annotations.BeforeTest;
import org.testng.annotations.Parameters;
import org.testng.annotations.Test;

public class ParallelTest {

	private WebDriver driver;
	String baseURL = "http://www.google.com/";

	@Parameters({ "browser" })
	@BeforeTest
	public void openBrowser(String browser) {
		try {
			if (browser.equalsIgnoreCase( "Firefox")) {
				driver = new FirefoxDriver();
			} else if (browser.equalsIgnoreCase("chrome")) {
				System.setProperty("webdriver.chrome.driver",
					"D:/Dev/Jars/chromedriver.exe");
				driver = new ChromeDriver();
			} else if (browser.equalsIgnoreCase("IE")) {
				System.setProperty("webdriver.ie.driver" ,
					"D:/Dev/Jars/IEDriverServer.exe");
				driver = new InternetExplorerDriver();
			}

		} catch (WebDriverException e) {
			System.out.println(e.getMessage());
		}

	}

	@Test
	public void login_TestCase() {
		driver.navigate().to(baseURL);
		//do something
```

```
        }

        @Test
        public void search_TestCase() {
                driver.navigate().to(baseURL);
            //do something
        }

        @AfterTest
        public void closeBrowser() {
                driver.quit();
        }
}
```

In the above code, we have OpenBrowser method with BeforeTest annotation along with parameter 'browser'. In the xml we will define three tests tags to run each test with different browser. We will compare the browser value with the parameter value and based on that we will create the driver instance. We have now defined three tests with three browsers (Firefox, Google Chrome and Internet Explorer)

You can find more details about defining parameters in testng

Below is the testng.xml file which will run all the tests which are defined. We are passing parameter value with browser name for each test. Tests will be executed in there browsers one by one.

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Parallel test suite">
  <test name="Firefox Test">
  <parameter name="browser" value="Firefox"/>
    <classes>
      <class name="com.pack.ParallelTest"/>
    </classes>
  </test>
  <test name="Chrome Test">
  <parameter name="browser" value="chrome"/>
    <classes>
      <class name="com.pack.ParallelTest"/>
    </classes>
  </test>
    <test name="Internet Explorer Test">
    <parameter name="browser" value="IE"/>
    <classes>
```

```
      <class name="com.pack.ParallelTest"/>
    </classes>
  </test>
</suite>
```

We can also use TestNG to execute tests Simultaneously by defining the"parallel" attribute to "tests" to run all the tests in defined browser with the help of testng.xml configuration file.

```
<suite name="Parallel test suite" parallel="tests">
```

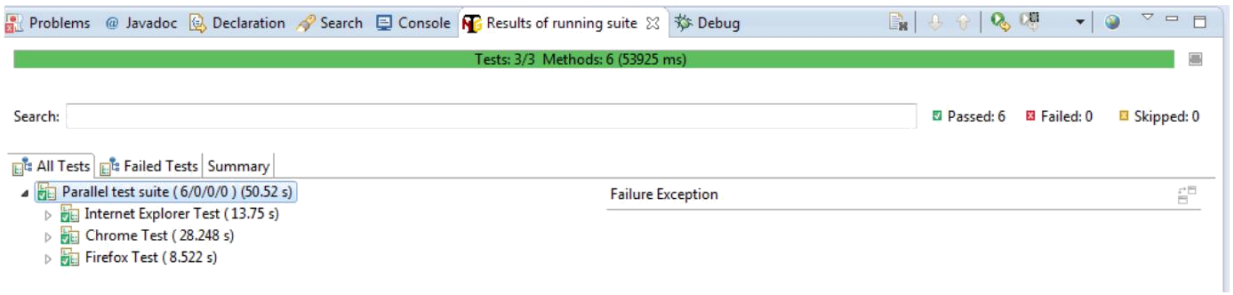
We just need to update the single line in above testng.xml. By defining parallel="tests" in suite tag, tests will get executed simultaneously. In order to execute to execute tests simultaneously it is recommended to have good system configuration, so that the execution time can be saved.

You can find more details about Executing multiple tests using testng

It looks like below:

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Parallel test suite" parallel="tests">
  <test name="Firefox Test">
  <parameter name="browser" value="Firefox"/>
    <classes>
      <class name="com.pack.ParallelTest"/>
    </classes>
  </test>
  <test name="Chrome Test">
  <parameter name="browser" value="chrome"/>
    <classes>
      <class name="com.pack.ParallelTest"/>
    </classes>
  </test>
    <test name="Internet Explorer Test">
    <parameter name="browser" value="IE"/>
    <classes>
      <class name="com.pack.ParallelTest"/>
    </classes>
  </test>
</suite>
```

After executing you can view the report, which will look like below:



Question 55:Find out broken links on website using selenium webdriver and HTTP Client

Earlier we have seen working with finding broken images, now here we will see finding invalid URLs. Here a valid URL will always have a status with 200. We have different HTTP status codes which are used for different purposes. You can check Wiki page for more information on HTTP Status Codes

Here 2xx class of status codes indicates that the action request by client was received and processed successfully without any issues.

And 4xx class of status code is mainly intended for cases in which the client seems to have erred.

And 5xx class of status codes are intended for cases in which the server seems to have erred.

The following are the list of different HTTP status codes.

The following are the list of HTTP response status codes		
Type	Status Codes	Examples
Informational	1xx	100 Continue, 101 Switching Protocols
Success	2xx	200 - OK , 201 - Created, 202 Accepted
Redirection	3xx	300 Multiple Choices, 301 Moved Permanently, 302 - Found
Client Error	4xx	400 Bad Request, 403 - Forbidden , 404 - Not Found , 422 - Unprocessable Entity
Server Error	5xx	500 - Internal Server Error , 503 - Service Unavailable

By just seeing the Links in the UI, we may not be able to confirm if that link is working or not until we click and verify it.

To achieve this, we can use HTTPClient library to check status codes of the URLs on a page. You need to download and add it to the build path.

If request was NOT processed correctly, then the HTTP status codes may return any of the above listed codes but not a 200 status code. We can easily say whether the link is broken or not with status codes.

Now let us jump into the example, First we will try to find all anchor tags on the page by using Webdriver. By using the below syntax:

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

We need to iterate through each link and verify request response Status codes and it should be 200 if not, we will increment invalid links count

Let us look into the example :

```
package com.linked;

import java.util.List;

import org.apache.http.HttpResponse;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.HttpClientBuilder;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.annotations.AfterClass; import
org.testng.annotations.BeforeClass; import
org.testng.annotations.Test;

public class FindBrokenLinksExample {

        private WebDriver driver;
        private int invalidLinksCount;

        @BeforeClass
        public void setUp() {

                driver = new FirefoxDriver();
                driver.get("http://google.com");
        }

        @Test
        public void validateInvalidLinks() {

                try {
```

```
                        invalidLinksCount = 0;
                        List<WebElement> anchorTagsList = driver.findElements(By
                                        .tagName("a"));
                        System.out.println("Total no. of links are "
                                        + anchorTagsList.size());
                        for (WebElement anchorTagElement : anchorTagsList)
                                { if (anchorTagElement != null) {
                                        String url =
anchorTagElement.getAttribute("href");
                                        if (url != null && !url.contains("javascript"))
{
                                                verifyURLStatus(url);
                                        } else {
                                                invalidLinksCount++;
                                        }
                                }
                        }

                        System.out.println("Total no. of invalid links are "
                                        + invalidLinksCount);

                } catch (Exception e) { e.printStackTrace();
                        System.out.println(e.getMessage());

                }
        }

        @AfterClass
        public void tearDown() {
                if (driver != null)
                        driver.quit();
        }

        public void verifyURLStatus(String URL) {

                HttpClient client = HttpClientBuilder.create().build();
                HttpGet request = new HttpGet(URL);
                try {
                        HttpResponse response = client.execute(request);
```

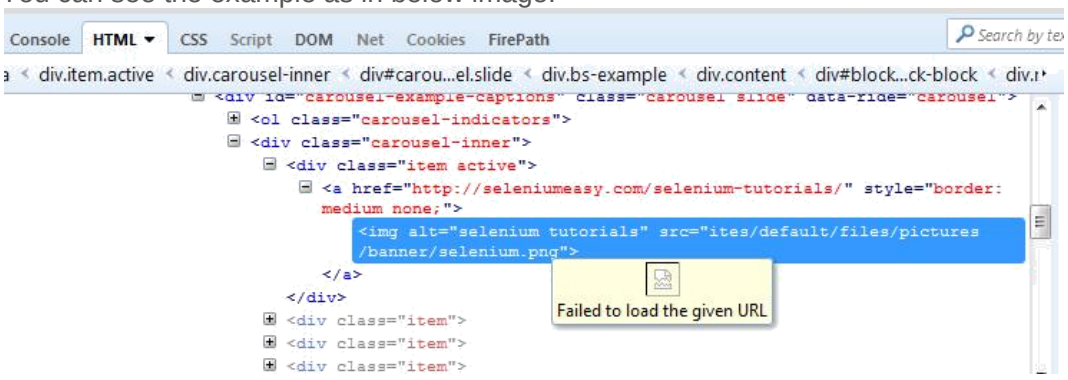
```
                    // verifying response code and The HttpStatus should be 200 if
not,
                    // increment invalid link count
                    ////We can also check for 404 status code
like response.getStatusLine().getStatusCode() == 404
                if (response.getStatusLine().getStatusCode() != 200)
                        invalidLinksCount++;
                } catch (Exception e) {
                    e.printStackTrace();
                }
        }
}
```

Question 56:Find Broken / Invalid Images on a Page

There are cases where we have seen image loading is failed due to many reasons, the most see is "Image not loading - Failed to load the given URL" becausenotlocatedofin theimagesam file is location as that is specified or may be image file is corrupted. And it will be very difficult to identify invalid images when there are many in the applications.

To achieve this, we can use HTTPClient library to check status codes of the images on a page. If they don't load correctly, then it will be registered with likely a 404 but not a 200 status code. We can easily say whether the link is broken or not with status codes.

You can see the example as in below image.



You can download Apache HTTPClient from here and add to you build path.

First we will try to find all images on the page by using Webdriver. Below is the syntax:

```
List<WebElement> imagesList = driver.findElements(By.tagName("img"));
```

Now iterate through each image and verify response code with HttpStatus and it should be 200 if not, increment invalid images count. We can get the response code using below statement:

```
response.getStatusLine().getStatusCode()
```

Let us look into the example :

```
package com.linked;

import java.util.List;

import org.apache.http.HttpResponse;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.HttpClientBuilder;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.annotations.AfterClass; import
org.testng.annotations.BeforeClass; import
org.testng.annotations.Test;

public class FindBrokenImages {

        private WebDriver driver;
        private int invalidImageCount;

        @BeforeClass
        public void setUp() {
                driver = new FirefoxDriver();
                driver.get("http://google.com");
        }

        @Test
        public void validateInvalidImages() {
                try {
                        invalidImageCount = 0;
                        List<WebElement> imagesList
= driver.findElements(By.tagName("img"));
                        System.out.println("Total no. of images are " +
imagesList.size());
                        for (WebElement imgElement : imagesList) {
                                if (imgElement != null) {
                                        verifyimageActive(imgElement);
```

```
                                        }
                                }
                                System.out.println("Total no. of invalid images are "   +
invalidImageCount);
                        } catch (Exception e) {
                                e.printStackTrace();
                                System.out.println(e.getMessage());
                        }
                }

                @AfterClass
                public void tearDown() {
                        if (driver != null)
                                driver.quit();
                }

                public void verifyimageActive (WebElement imgElement) {
                        try {
                                HttpClient client = HttpClientBuilder.create().build(); HttpGet
                                request = new HttpGet(imgElement.getAttribute("src"));
                                HttpResponse response = client.execute(request);
                                // verifying response code he HttpStatus should be 200 if not,
                                // increment as invalid images count
                                if (response.getStatusLine().getStatusCode() != 200)
                                        invalidImageCount++;
                        } catch (Exception e) {
                                e.printStackTrace();
                        }
                }
}
```

Question 57:How to handle internationalisation through web driver?

FirefoxProfile profile = new FirefoxProfile();
profile.set Preference("intl.accept_languages","jp");
Web driver driver = new FirefoxDriver(profile); driver.get(google.com)

Question 58:Page Object Model | POM

Creating Selenium test cases can result in an unmaintainable project. One of the reasons is that too many duplicated code is used. Duplicated code could be caused by duplicated functionality and this will result in duplicated usage of locators. The disadvantage of duplicated code is that the project is less maintainable. If some locator will change, you have to walk through the whole test code to adjust locators where necessary. By using the page object model we can make non -brittle test code and reduce or eliminate duplicate test code. Beside of that it improves the readability and allows us to create interactive documentation. Last but not least, we can create tests with less keystroke. An implementation of the page object model can be achieved by separating the abstraction of the test object and the test scripts.

Note: We will follow the same example which we have used in ***First Test Case***. Let's assume it our base test case and implement the Page Object Model (POM) in it.

How to do it…

1. Create***New a****Package*‘‘ file and name**pageObjects’**itas ,‘by right click on the Project and select **New** > **Package**. We will be creating different packages for Page Objects, Utilities, Test Data, Test Cases and Modular actions. It is always recommended to use this structure, as it is easy to understand, easy to use and easy to maintain.

2. Create a ***‘New Class‘*** file and refer the name to the actual page from the test object, by right click on the above created Package and select **New** > **Class**. In our case it is **Home Page** and **LogIn Page**.

```
package pageObjects;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

public class Home_Page {

	private static WebElement element = null;

	public static WebElement lnk_MyAccount(WebDriver driver){

		element = driver.findElement(By.id("account"));

		return element;

	}

	public static WebElement lnk_LogOut(WebDriver driver){

		element = driver.findElement(By.id("account_logout"));
```

```
            return element;
      }
}
```

3. Now create a **Static Method** for each **Element** (Object) in the Home Page. Each method will have an**Argument** (driver) and a **Return** value (element).

Driver is being passed as an **Argument** so that Selenium is able to locate the element on the browser (driver).

Element is returned, so that an **Action** can be performed on it.

Method is declared as **Public Static**, so that it can be called in any other method without **instantiate**the class.

Follow the same rule for creating **LogIn Page** class.

```
package pageObjects;

import org.openqa.selenium.*;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

public class LogIn_Page {

        private static WebElement element = null;

    public static WebElement txtbx_UserName(WebDriver driver){

         element = driver.findElement(By.id("log"));

         return element;

         }

     public static WebElement txtbx_Password(WebDriver driver){

         element = driver.findElement(By.id("pwd"));

         return element;

         }

     public static WebElement btn_LogIn(WebDriver driver){

         element = driver.findElement(By.id("login"));

         return element;

         }

}
```

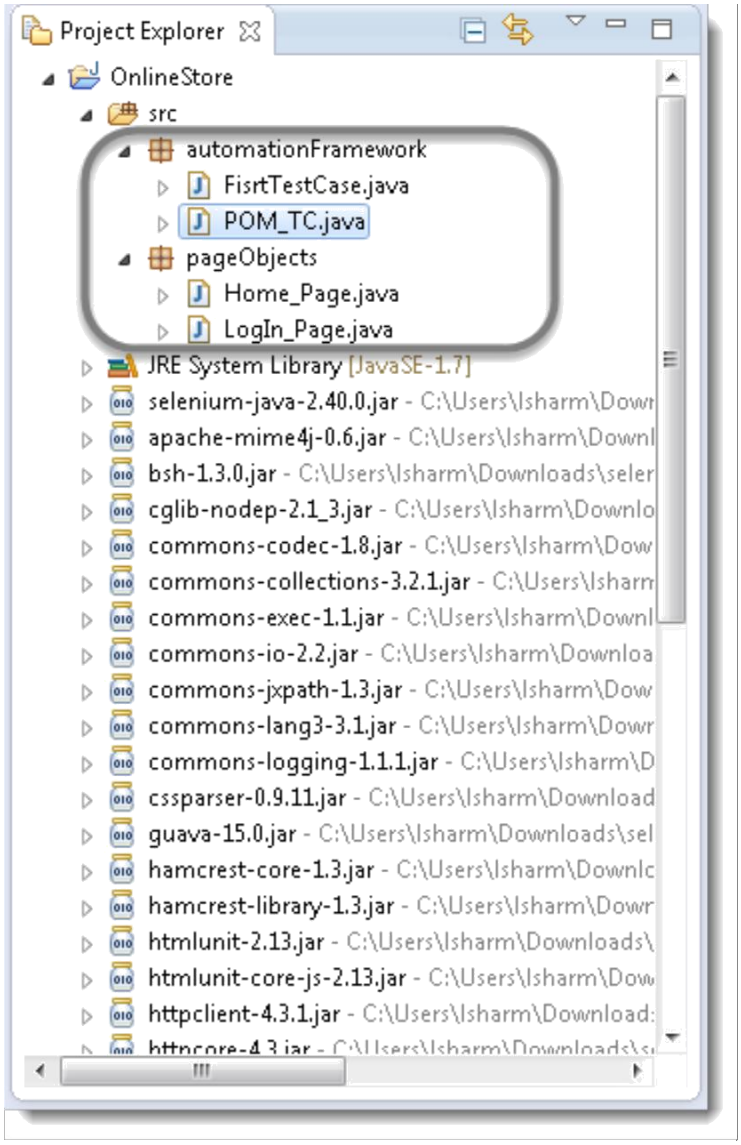
4) Create*New*a*Class*“ and name**POM**it_**TC**asby right clickautomationFrameworkthe“ Package and select **New** > **Class**. We will be creating all our test cases under this package.
Now convert your old ***First Test Case*** in to the new Page Object Model test case. package

```
package automationFramework;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

// Import package pageObject.*

import pageObjects.Home_Page;

import pageObjects.LogIn_Page;

public class PageObjectModel {

      private static WebDriver driver = null;

      public static void main(String[] args) {

            driver = new FirefoxDriver();

             driver.manage().timeouts().implicitlyWait(10,

             TimeUnit.SECONDS); driver.get("http://www.store.demoqa.com");

             // Use page Object library now

             Home_Page.lnk_MyAccount(driver).click();

             LogIn_Page.txtbx_UserName(driver).sendKeys("testuser_1");

             LogIn_Page.txtbx_Password(driver).sendKeys("Test@123");

             LogIn_Page.btn_LogIn(driver).click();

             System.out.println(" Login Successfully, now it is the time to Log Off
buddy.")

             Home_Page.lnk_LogOut(driver).click();

             driver.quit();

       }

}
```

You will notice that once you type Home_Page in your test script and the moment you press dot, all the methods in the Home Page will display. We can expose methods in order to reduce duplicated

code. We are able to call these method multiple times. This will ensure a better maintainable test code, because we only have to make adjustments and improvements in one particular place.

Your Project explorer window will look like this now.



Queation 59:Taking ScreenShot ONLY for Failed Tests

We have discussed about simple way of taking a screen shot earlier. Now in this tutorial, we will see how to take screen shot ONLY for failed tests.

To do this, first we need to create a class and then implement *TestNG 'ITestListener'*. We will have a method called *'onTestFailure'*. We need to add the code to take the screen shot in this method.

Instead of just taking the screen shot, we will get the Test method name and take a screen shot with test name and place it is destination folder.

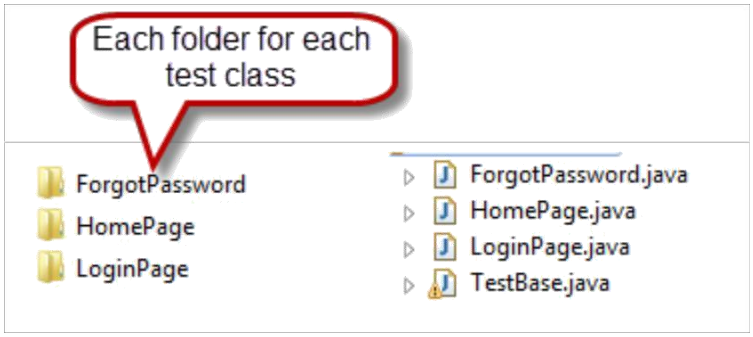
```
package com.pack.listeners;

import java.io.File;
import java.io.IOException;

import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.testng.ITestContext;
import org.testng.ITestListener;
import org.testng.ITestResult;

import com.pack.sample.TestBase;

public class TestListener implements ITestListener {
        WebDriver driver=null;
        String filePath = "D:\\SCREENSHOTS";
    @Override
    public void onTestFailure(ITestResult result) {
        System.out.println("***** Error "+result.getName()+" test has failed
        *****"); String methodName=result.getName().toString().trim();
        takeScreenShot(methodName);
    }

    public void takeScreenShot(String methodName) {
        //get the driver
        driver=TestBase.getDriver();
         File scrFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE); //The
        below method will save the screen shot in d drive with test method name
            try {
                                    FileUtils.copyFile(scrFile, new
File(filePath+methodName+".png"));
                                    System.out.println("***Placed screen shot in "+filePath+"
***");
                            } catch (IOException e) {
                                    e.printStackTrace();
```

```
                        }
    }
        public void onFinish(ITestContext context) {}

    public void onTestStart(ITestResult result) {    }

    public void onTestSuccess(ITestResult result) {    }

    public void onTestSkipped(ITestResult result) {    }

    public void onTestFailedButWithinSuccessPercentage(ITestResult result) {    }

    public void onStart(ITestContext context) {    }
}
```

Before executing the above program, we need to add *TestListener* class in testng.xml file as below:

```
<listeners>
      <listener class-name="com.pack.listeners.TestListener"/>
  </listeners>
```

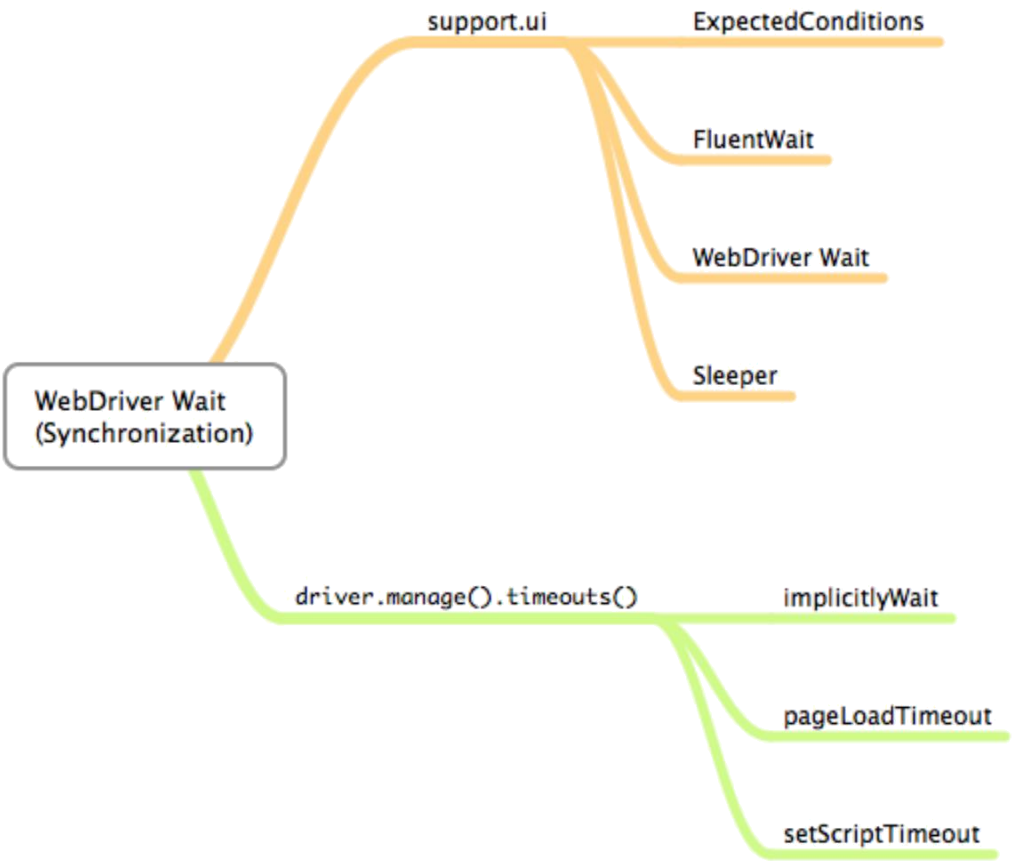
In the Next tutorial, we will try to extend the way of taking the screen shot and placing it by creating a folder with Test Class name and Screens shots of that particular test class as shown in below screen shot. Click here for example on Take Screenshot and place it in a folder with Test Class name



Question 60: Wait commands in WebDriver

Listing out the different WebDriver Wait statements that can be useful for an effective scripting and can avoid using the *Thread.sleep()* comamnds

After few searches and digging into the WebDriver Java doc, I managed to design a mindmap of the different WebDriver commands available




```
WebDriver.manage().timeouts()
implicitlyWait
[sourcecode language=”java”]
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")); [/sourcecode]
```

The ImplicitWait will tell the webDriver to poll the DOM for a certain duration when trying to find the element, this will be useful when certain elements on the webpage will not be available immediately and needs some time to load.

By default it ill take the value to 0, for the life of the WebDriver object instance through out the test script.

```
pageLoadTimeout
[sourcecode language=”java”]
driver.manage().timeouts().pageLoadTimeout(100, SECONDS);
[/sourcecode]
```

Sets the amount of time to wait for a page load to complete before throwing an error. If the timeout is negative, page loads can be indefinite.

```
setScriptTimeout
[sourcecode language=”java”]
driver.manage().timeouts().setScriptTimeout(100,SECONDS);
[/sourcecode]
```

Sets the amount of time to wait for an asynchronous script to finish execution before throwing an error. If the timeout is negative, then the script will be allowed to run indefinitely.

_____

```
Support.ui
FluentWait
[sourcecode language=”java”]
// Waiting 30 seconds for an element to be present on the page, checking
// for its presence once every 5 seconds.
Wait<WebDriver> wait = new FluentWait<WebDriver>(driver)
.withTimeout(30, SECONDS)
.pollingEvery(5, SECONDS)
.ignoring(NoSuchElementException.class);
WebElement foo = wait.until(new Function<WebDriver, WebElement>()
{ public WebElement apply(WebDriver driver) {
return driver.findElement(By.id("foo"));
}
});
[/sourcecode]
```

Each FluentWait instance defines the maximum amount of time to wait for a condition, as well as the frequency with which to check the condition. Furthermore, the user may configure the wait to ignore specific types of exceptions whilst waiting, such as NoSuchElementExceptions when searching for an element on the page.

```
ExpectedConditions
[sourcecode language=”java”]
WebDriverWait wait = new WebDriverWait(driver, 10);
WebElement element = wait.until(ExpectedConditions.elementToBeClickable(By.id("someid")));
[/sourcecode]
```

Models a condition that might reasonably be expected to eventually evaluate to something that is neither null nor false.
Examples : Would include determining if a web page has loaded or that an element is visible.

Note that it is expected that ExpectedConditions are idempotent. They will be called in a loop by the WebDriverWait and any modification of the state of the application under test may have unexpected side-effects.
WebDriverWait will be used as we used in the Expected conditions code snippet as above.

sleeper is something same as the Thread.sleep() method, but this with an Abstraction around the thread.sleep() for better testability

Question 61:/What is TestNG Listener

In last post we have discussed WebDriver Listener and hope you understood the actual concept of listeners, if not I will strongly recommend you that please go through WebDriver Listener also.
This is very common question also in interv iews that what is the actual difference between TestNG Listener and Webdriver Listener.
Let's discuss now- When we talk about TestNG Listener we will use **ITestListener** Interface.
This interface has some methods which we need to override as per our requirement.

For example- take some method of **ITestListener**

```
1-@Override
    public void onFinish(ITestContext arg0) {
   }
 2-    @Override
    public void onStart(ITestContext arg0) {
        }
 3- @Override
    public void onTestFailure(ITestResult arg0) {
    }
```

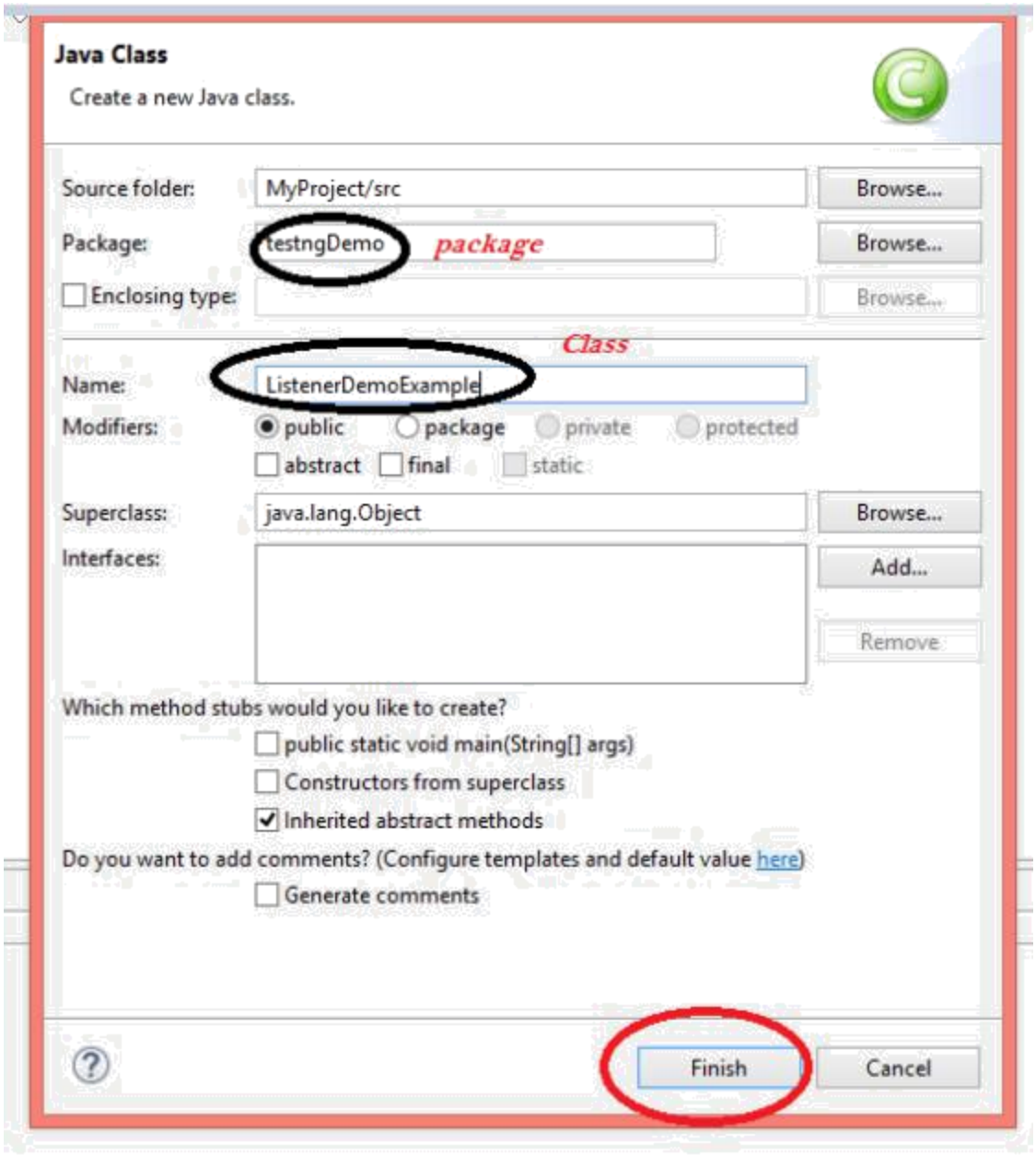
etc..

How to implement TestNG Listener

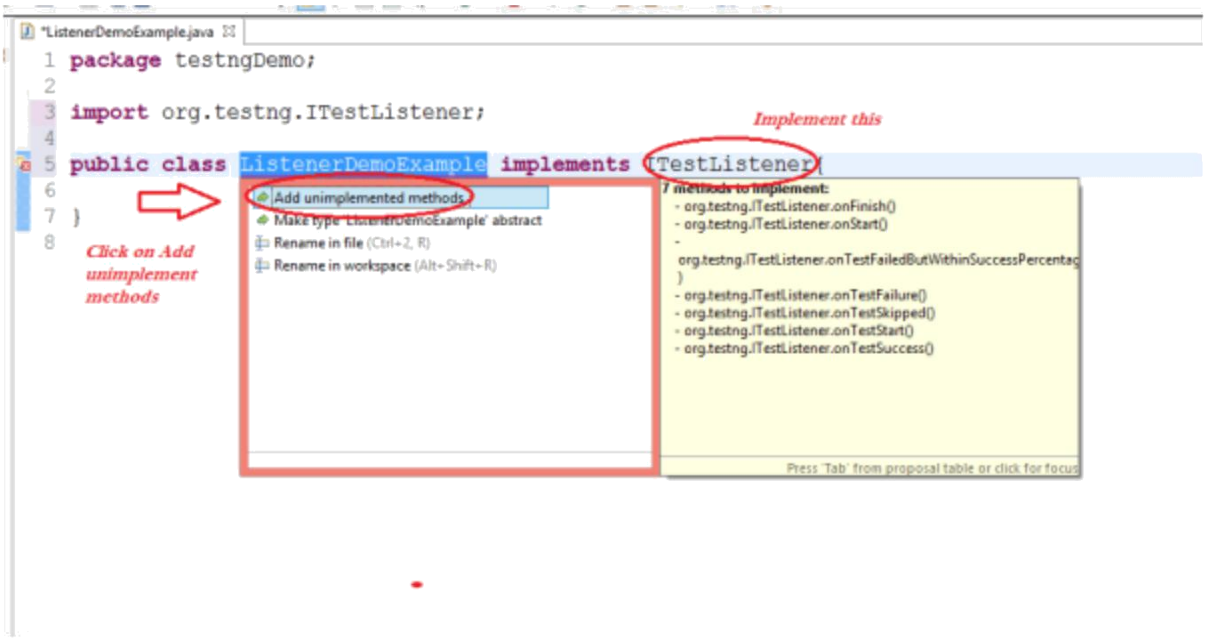
Its not that tough as compared to WebDriver Listeners we simply have to implement **ITestListener** interface and override all method as per requirement and its not required that you should have some statements in all methods but yes you should use some of that method Wisely

Scenario-
1 -If testcase is failing then what action should be performed
2- If testcase is skipped then what should be action requested and so far..

Step 1- Create a simple class and implement ITestListener listener



Step 2- Override all unimplemented methods



Once you implement all your java program will look like

```
package testngDemo;

import org.testng.ITestContext;
import org.testng.ITestListener;
import org.testng.ITestResult;

public class ListenerDemoExample implements ITestListener{

    @Override
    public void onFinish(ITestContext arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void onStart(ITestContext arg0) {
        // TODO Auto-generated method stub

    }
```

```
    @Override
    public void onTestFailedButWithinSuccessPercentage(ITestResult arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void onTestFailure(ITestResult arg0)
{       // TODO Auto-generated method stub

    }

    @Override
    public void onTestSkipped(ITestResult arg0)
{       // TODO Auto-generated method stub

    }

    @Override
    public void onTestStart(ITestResult arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void onTestSuccess(ITestResult arg0)
{       // TODO Auto-generated method stub

    }

}
```

So let me implement some of the methods and I will start my test.

I will override following method onTestStart, onTestSuccess(), onTestFailure()

I have written a small test for facebook login and I am failing this testcase forcefully so that we can check how different events will be fired if testcase pass or fail

Program

```
package testngDemo;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.ITestContext;
import org.testng.ITestListener;
import org.testng.ITestResult;
import org.testng.annotations.Test;

public class ListenerDemoExample implements ITestListener{

    // This is dummy tescase for facebook login
    @Test
    public void loginFB(){

        WebDriver driver=new FirefoxDriver();
        driver.get("http://www.facebook.com");
        driver.manage().window().maximize();
        driver.findElement(By.id("email")).sendKeys("mukesh@facebook.com");
        driver.findElement(By.id("wronglocator")).sendKeys("dont-tell");
        driver.findElement(By.id("loginbutton")).click();
    }



    @Override
    public void onFinish(ITestContext arg0) {
        // TODO Auto-generated method stub

    }
```

```
    @Override
    public void onStart(ITestContext arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void onTestFailedButWithinSuccessPercentage(ITestResult arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void onTestFailure(ITestResult arg0) {

        System.out.println("Screen shot captured====="+arg0.toString());

    }

    @Override
    public void onTestSkipped(ITestResult arg0) {
        // TODO Auto-generated method stub


    }

    @Override
    public void onTestStart(ITestResult arg0) {
        // TODO Auto-generated method stub
        System.out.println("TestCase started====" +arg0.toString());

    }

    @Override
    public void onTestSuccess(ITestResult arg0) {
        // TODO Auto-generated method stub
```

```
        System.out.println("Congrates Testcase has been passed===="+ arg0.toString());

    }

}
```

Step 3- It not completed yet now we need to modify our xml file that is known as testng.xml and we have to specify this listener and we have to execute that xml file only.

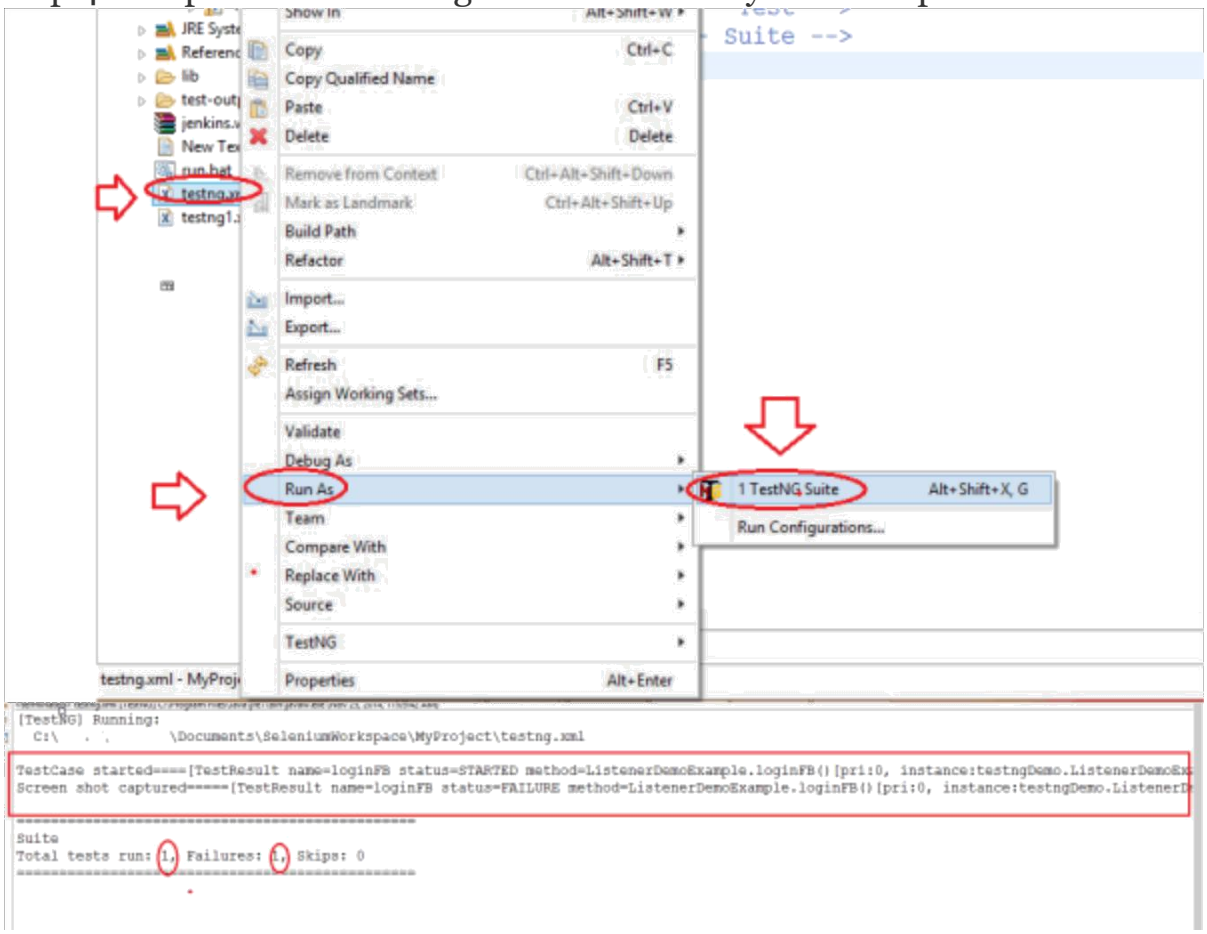
We have predefined listeners tag so that we will use today.

```
<listeners>
<listener class-name="testngDemo.ListenerDemoExample"/>
</listeners>
```

so your complete testng.xml file will look like.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Suite" parallel="none">
<listeners>
<listener class-
name="testngDemo.ListenerDemoExample"/> </listeners>
  <test name="Test">
    <classes>
      <class name="testngDemo.ListenerDemoExample"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

Step 4- Simple run this testng.xml file and analyze the output



Question 62 :How to select a element from dropdown ?

```
Select dropdown = new Select(driver.findElement(By.id("identifier")));
```

Once this is done you can select the required value in 3 ways. Consider an HTML file like this

```
<html>
<body>
<select id = "designation">
<option value = "MD">MD</option>
<option value = "prog"> Programmer </option>
<option value = "CEO"> CEO </option>
</option>
</select>
<body>
</html>
```

Now to identify dropdown do

```
Select dropdown = new Select(driver.findElement(By.id("designation")));
```

To select its option say 'Programmer' you can do

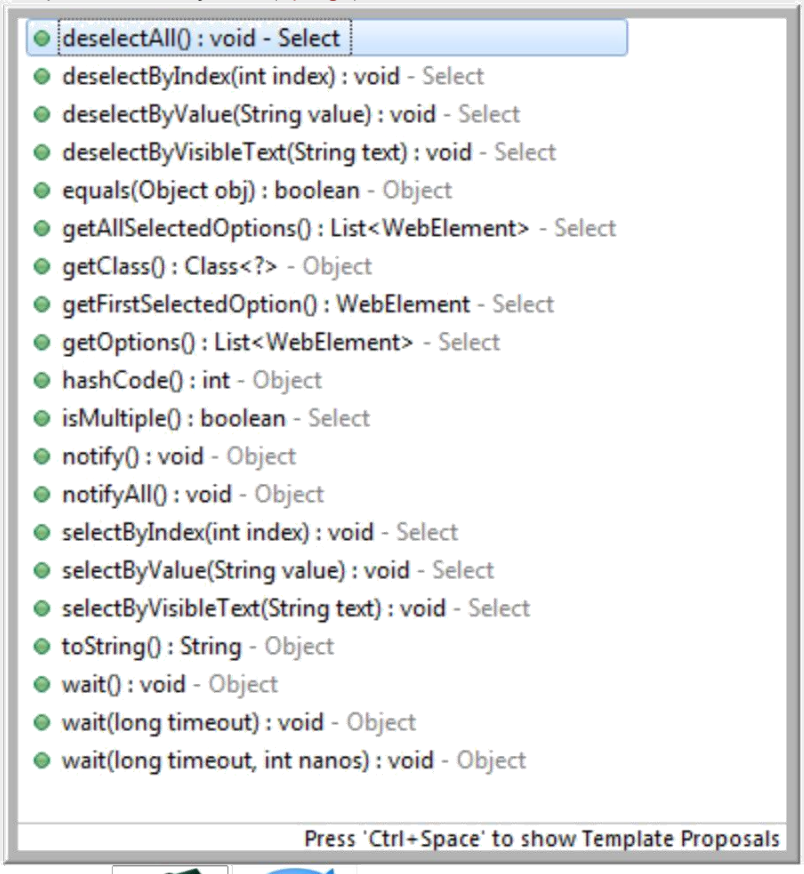
```
dropdown.selectByVisibleText("Programmer ");
```

or

```
 dropdown.selectByIndex(1);
```

or

```
dropdown.selectByValue("prog");
```



Question 63 :Webdriver Event Listener ?

Selenium webdriver has ability to track different events such as 'beforeNavigateTo' , 'afterNavigateTo' , 'beforeClickOn' , 'afterClickOn', 'onException' and so on. When ever we develop test scripts we can write our own implementation for handling events during the execution.

For example, we can count the number of clicks that we have performed in the script by adding logic to count clicks after each click 'afterClickOn'. We can also when ever there is an exception using 'onException'.

Let us now look into example by taking the help of class " EventFiringWebDriver" - which is a wrapper around an arbitrary WebDriver instance which supports registering of a WebDriverEventListener for logging purposes.

Now firstly we have to initialize 'EventFiringWebDriver' by passing webdriver instance and then we need to register event listener for EventFiringWebDriver instance.

We also need to provide implementation for the events. We can do this by two ways, which ever is needed for our project.

One way is by using WebDriverEventListener Interface If we use this, it will implement all the events that are supported by Webdriver.

An example for this is shown below, when we use this, it will implement all the events like 'beforeNavigateTo', 'afterNavigateTo', 'beforeClickOn' , 'afterClickOn', 'onException' and more such events

```
package com.example;

/************************************ PURPOSE ********************************

 - This class implements the WebDriverEventListener, which is included under events.
 The purpose of implementing this interface is to override all the methods and define
certain useful Log statements
 which would be displayed/logged as the application under test is being run.

 Do not call any of these methods, instead these methods will be invoked
 automatically as an when the action done (click, findBy etc).

 */

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.events.WebDriverEventListener;

public class WebEventListener implements WebDriverEventListener {

        public void beforeNavigateTo(String url, WebDriver driver) {
                System.out.println("Before navigating to: '" + url + "'");
        }

          public void afterNavigateTo(String url, WebDriver driver)
               { System.out.println("Navigated to:'" + url + "'");
        }

        public void beforeChangeValueOf(WebElement element, WebDriver driver) {
                System.out.println("Value of the:" + element.toString()
```

```
                                        + " before any changes made");
        }

        public void afterChangeValueOf(WebElement element, WebDriver driver) {
                System.out.println("Element value changed to: " + element.toString());
        }

        public void beforeClickOn(WebElement element, WebDriver driver) {
                System.out.println("Trying to click on: " + element.toString());
        }

        public void afterClickOn(WebElement element, WebDriver driver) {
                System.out.println("Clicked on: " + element.toString());
        }

        public void beforeNavigateBack(WebDriver driver) {
                System.out.println("Navigating back to previous page");
        }

        public void afterNavigateBack (WebDriver driver) {
                System.out.println("Navigated back to previous page");
        }

        public void beforeNavigateForward(WebDriver driver) {
                System.out.println("Navigating forward to next page");
        }

        public void afterNavigateForward(WebDriver driver) {
                System.out.println("Navigated forward to next page");
        }

        public void onException(Throwable error, WebDriver driver) {
                System.out.println("Exception occured: " + error);
        }

        public void beforeFindBy(By by, WebElement element, WebDriver driver) {
                System.out.println("Trying to find Element By : " + by.toString());
        }
```

```
        public void afterFindBy(By by, WebElement element, WebDriver driver) {
                System.out.println("Found Element By : " + by.toString());
        }

        /*
         * non overridden methods of WebListener
         class */
        public void beforeScript(String script, WebDriver driver)
        { }

        public void afterScript(String script, WebDriver driver) {
        }

}
```

And other way is by using Class AbstractWebDriverEventListener If we use this class we can include only those events which we are interested.

An example for this is shown below, if you observer we have implemented only 5 events, 'beforeNavigateTo', 'afterNavigateTo', 'beforeClickOn' , 'afterClickOn' and 'onException'

```
package com.example;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.events.AbstractWebDriverEventListener;

public class WebEventListener extends AbstractWebDriverEventListener {

        public void beforeNavigateTo(String url, WebDriver driver) {
                System.out.println("Before navigating to: '" + url + "'");
        }

          public void afterNavigateTo(String url, WebDriver driver)
               { System.out.println("Navigated to:'" + url + "'");
        }

        public void beforeClickOn(WebElement element, WebDriver driver) {
                System.out.println("Trying to click on: " + element.toString());
        }
```

```
        public void afterClickOn(WebElement element, WebDriver driver) {
                System.out.println("Clicked on: " + element.toString());
        }

        public void onException(Throwable error, WebDriver driver)
                { System.out.println("Error occurred: " + error);
        }
}
```

Now let us try to use in our program. From the above two options, here we will use 'AbstractWebDriverEventListener' and implement the above 5 methods to understand the example

```
package com.example;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.events.EventFiringWebDriver;
import org.openqa.selenium.support.ui.ExpectedConditions; import
org.openqa.selenium.support.ui.WebDriverWait; import
org.testng.Assert;
import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;

public class EventListenerExample {

        private WebDriver driver;
        private EventFiringWebDriver e_driver;
        private WebEventListener eventListener;
        private WebDriverWait wait;
        private String appURL = "http://www.google.com";
        public int waitTime = 10;
        private String headerText = "One account. All of Google.";
        private String errMessage = "The email and password you entered don't match.";

        @BeforeClass()
        public void setUp() {

                // Initializing instance of Firefox WebDriver
                driver = new FirefoxDriver();
```

```
                wait = new WebDriverWait(driver, waitTime);

                // Initializing EventFiringWebDriver using Firefox WebDriver
                instance e_driver = new EventFiringWebDriver(driver);

                // Now create object of EventListerHandler to register it with
EventFiringWebDriver
                eventListener = new WebEventListener();

                e_driver.register(eventListener);

                e_driver.manage().window().maximize();
                e_driver.get(appURL);
        }

        @Test
        public void testEventsONE() {
                System.out.println("***** Executing Test ONE ***** ");
                e_driver.findElement(By.linkText("Gmail")).click();
                String pageHeaderText = e_driver.findElement(By.tagName( "h1")).getText();
                Assert.assertTrue(pageHeaderText.equalsIgnoreCase(headerText));
        }

        @Test
        public void testEventsTWO()  {
                System.out.println("***** Executing Test Two *****
                "); //Entering user name and clicking on next
                e_driver.findElement(By.id("Email")).sendKeys("username");
                e_driver.findElement(By.id("next")).click();

                wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("Passwd")));
                //Entering password and clicking on sign-in
                e_driver.findElement(By.id("Passwd")).sendKeys("password");
                e_driver.findElement(By.id("signIn")).click();

                //Get the error message and validate it
                String pageHeaderText =
e_driver.findElement(By.id("errormsg_0_Passwd")).getText();
                Assert.assertTrue(pageHeaderText.equalsIgnoreCase(errMessage));
        }
```

```
        @AfterClass()
        public void tearDown() {
                if (e_driver != null) {
                        e_driver.quit();
                }
        }

}
```

In the above example, 'setUp' method has driver.get(appURL) which will trigger 'beforeNavigateTo' and 'afterNavigateTo' methods and prints the URL. 'testEventsONE' and 'testEventsTWO' test methods has multiple events which will trigger 'beforeClickOn' and 'afterClickOn' events.

After executing the above program, below is the output :

```
[TestNG] Running:
  C:\Users\Easy\AppData\Local\Temp\testng-eclipse-310156716\testng-customsuite.xml


Before navigating to:- 'http://www.google.com'
Navigated to:-'https://www.google.co.in/?gfe_rd=cr&ei=dymBVv6EBrPv8weejrnwDg&gws_rd=ssl...
Executing Test ONE *****
Trying to click on:-  link text: Gmail
Clicked on:-  link text: Gmail
***** Executing Test Two *****
Trying to click on:-  id: next
Clicked on:-  id: next
Trying to click on:-  id: signIn
Clicked on:-  id: signIn
PASSED: testEventsONE
PASSED: testEventsTWO


===============================================
    Default test
    Tests run: 2, Failures: 0, Skips: 0
===============================================



===============================================
Default suite
Total tests run: 2, Failures: 0, Skips: 0
===============================================
```

People often ask the difference between WebDriverEventListener and Log4j, Here both are used for logging purposes BUT, WebDriverEventListener will trigger ONLY webdriver events and Using Log4j we will print all the messages that we provide and along with webdriver events.

Question 64 :How does the Selenium WebDriver work?

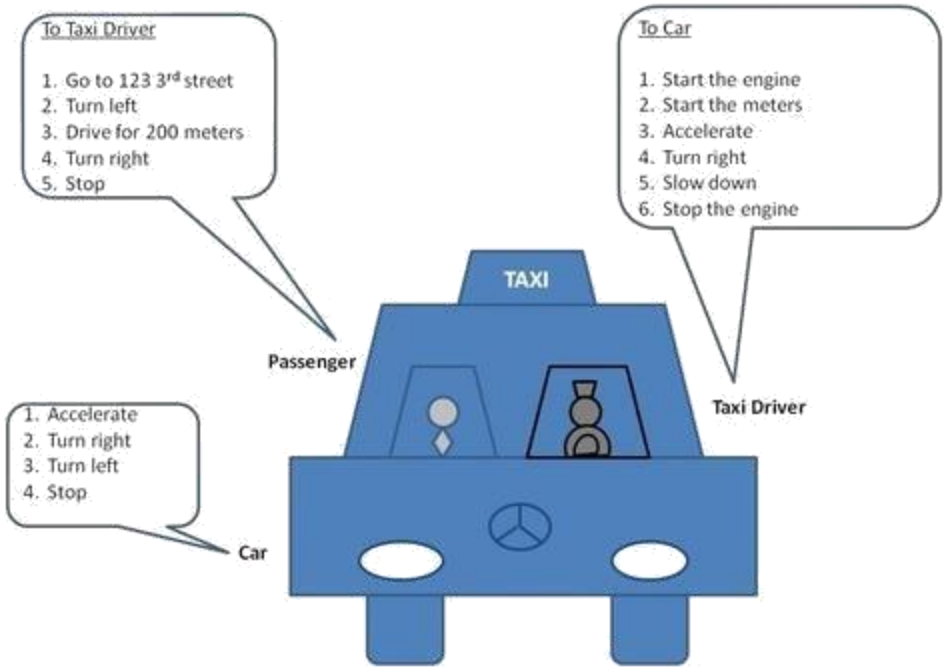
There are 2 ways of explaining how Selenium WebDriver works:

- easy
- technical

Selenium WebDriver drives a browser the same way a taxi driver drives a cab.
In taxi driving, there are typically 3 actors:

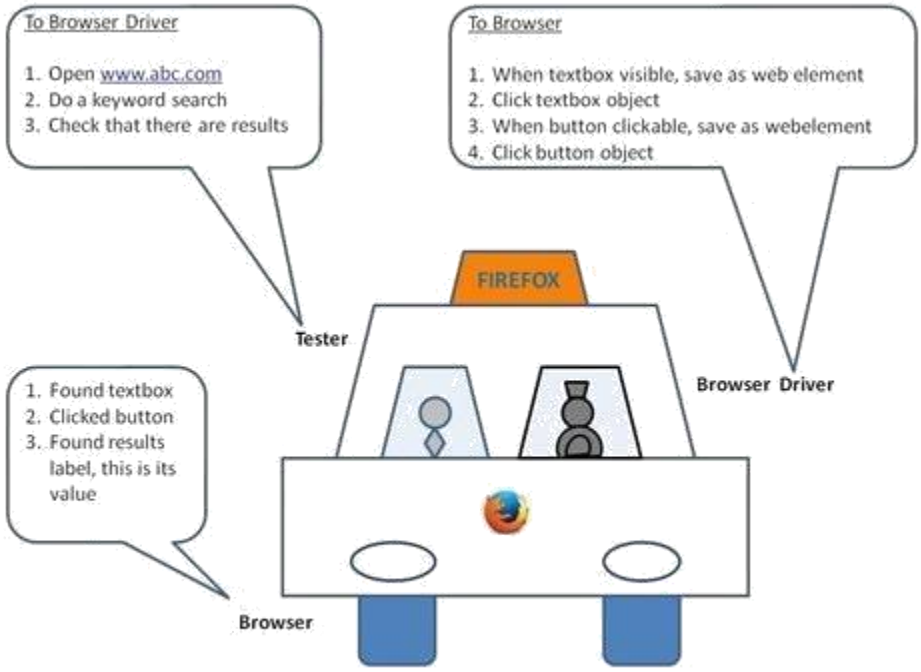
- **the client;** he tells the taxi driver where he wants to go and how to get there
- **the taxi driver;** he executes the client's requests; the taxi driver sends his own requests to the car
- **the car;** the car executes the taxi driver's requests

The client gets to the destination through dialogues that happen between the client - taxi driver and taxi driver - car.



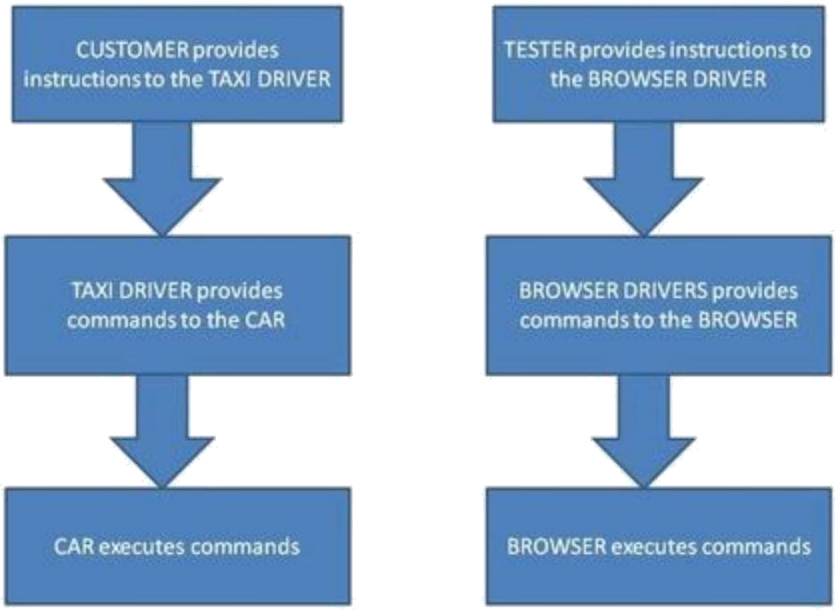
In test automation with Selenium WebDriver (and other tools), there are 3 actors as well:

- **test engineer that writes the automation code**; the automation code sends requests to the browser driver component
- **the browser driver component**; it executes the test engineer requests; it sends its own request to the browser
- **the browser**; it executes the browser driver requests



So this is the analogy:

1. the test engineer is like a taxi client
1. the browser driver is like a taxi driver
1. the browser is like a taxi



The technical explanation is not using analogies.
Like most technical explanations, there are no photos either :(
When the automation script is executed, the following steps happen:

- **for each Selenium command, a HTTP request is created and sent to the browser driver**
- the browser driver **uses a HTTP server for getting the HTTP requests**
- **the HTTP server** determines the **steps needed for implementing the Selenium command**
- the implementation steps are executed on the browser
- the execution status is sent back to the HTTP server
- the **HTTP server sends the status back to the automation script**

1) As a tester what should be your approach when requirements change continuously?

When requirement keeps changing, continuously agile tester should take following approach

- Write generic test plans and test cases, which focuses on the intent of the requirement rather than its exact details
- To understand the scope of change, work closely with the product owners or business analyst

- Make sure team understand the risks involved in changing requirements espec ially at the end of the sprint
- Until the feature is stable, and the requirements are finalized, it is best to wait if you are going to automate the feature
- Changes can be kept to a minimum by negotiating or implement the changes in the next sprint

2) List out the pros and cons of exploratory testing (used in Agile) and scripte d testing?

-

	Pros	**Cons**
Exploratory Testing	– It requires less preparation- Easy to modify when requirement changes - Works well when documentation is scarce	– Presenting progress and Coverage to project management is difficult
Scripted Testing	– In case testing against legal or regulatory requirements it is very useful	– Test preparation is usually time-consuming- Same steps are tested over and again- When requirement changes it is difficult to modify

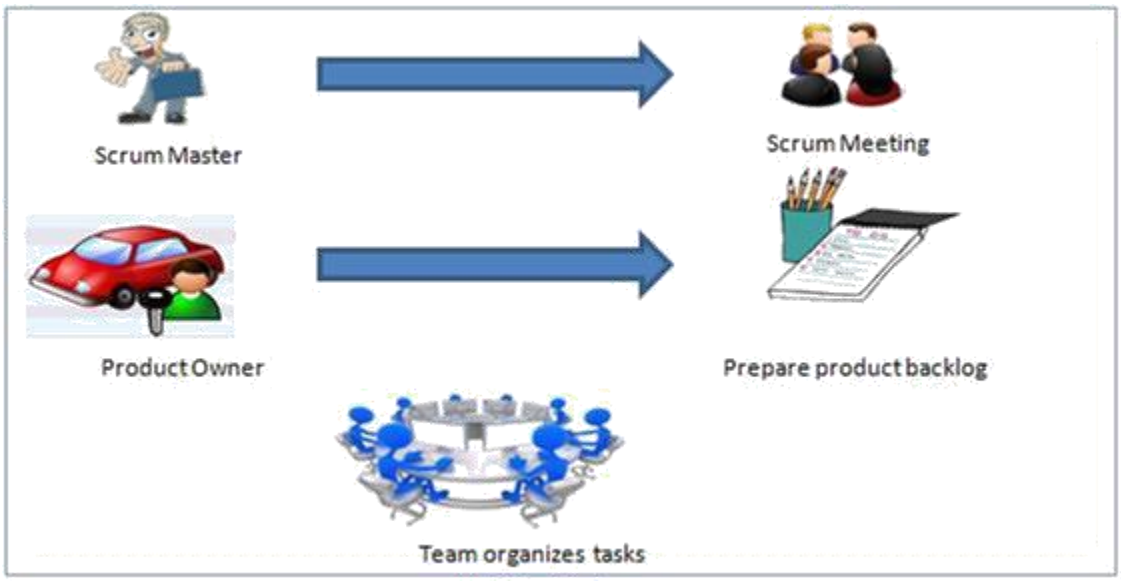
3) Explain the difference between Extreme programming and Scrum?

Scrum	Extreme Programing (XP)
– Scrum teams usually have to work in iterations called sprints which usually last up to two weeks to one month long	– XP team works in iteration that last for one or two weeks
– Scrum teams do not allow change into their sprints	– XP teams are more flexible and change their iterations
– In scrum, the product owner prioritizes the product backlog but the team decides the sequence in which they will develop the backlog items	– XP team work in strict priority order, features developed are prioritized by the customer
– Scrum does not prescribe any engineering practices	– XP does prescribe engineering practices

4) What is an epic, user stories and task?

Epic: A customer described software feature that is itemized in the product backlog is known as epic. Epics are sub-divided into stories

User Stories: From the client perspective user stories are prepared which defines project or business functions, and it is delivered in a particular sprint as expected. **Task:** Further down user stories are broken down into different task



5) Explain what is re-factoring?

To improve the performance, the existing code is modified; this is re -factoring. During re-factoring the code functionality remains same

6) Explain how you can measure the velocity of the sprint with varying team capacity?

When planning a sprint usually, the velocity of the sprint is measured on the basis of professional judgement based on historical data. However, the mathematical formula used to measure the velocity of the sprint are,

- first – completed story points **X** team capac ity: If you measure capacity as a percentage of a 40 hours weeks
- Second – completed story points **/** team capacity: If you measure capacity in man-hours

For our scenario second method is applicable.

7) Mention the key difference between sprint backlog and product backlog?

Product backlog: It contains a list of all desired features and is owned by the product owner

Sprint backlog: It is a subset of the product backlog owned by development team and commits to deliver it in a sprint. It is created in Sprint Planning Meeting

8) In Agile mention what is the difference between the Incremental and Iterative development?

Iterative: Iterative method is a continuous process of software development where the software development cycles are repeated (Sprint & Releases) till the final product is achieved.

Release 1: Sprint 1, 2… n

Release n: Sprint 1, 2….n

Incremental: Incremental development segregates the system functionality into increments or portions. In each increment, each segment of functionality is delivered through cross-discipline work, from the requirements to the deployment.

9) Explain what is Spike and Zero sprint in Agile? What is the purpose of it?

Sprint Zero: It is introduced to perform some research before initiating the first sprint. Usually this sprint is used during the start of the project for activities like setting development environment, preparing product backlog and so on.

Spikes: Spikes are type of stories that are used for activities like research , exploration, design and even prototyping. In between sprints, you can take spikes for the work related to any technical or design issue. Spikes are of two types Technical Spikes and Functional Spikes.

10) What is test driven development?

Test driven development or TDD is also known as test-driven design. In this method, developer first writes an automated test case which describes new function or improvement

and then creates small codes to pass that test, and later re -factors the new code to meet the acceptable standards.

11) Prototypes and Wireframes are widely used as part of?

Prototypes and Wireframes are prototypes that are widely used as part of Empirical Design

12) Explain what is Application Binary Interface?

Across different system platforms and environments a specification defining requirements for portability of applications in binary form is known as Application Binary Interface

13) Explain in Agile, burn-up and burn- down chart?

To track the project progress burnup and burn down, charts ar e used Burnup Chart: It shows the progress of stories done over time Burndown Chart: It shows how much work was left to do overtime

14) Explain what is Scrum ban?

Scrum ban is a software development model based on Scrum and Kanban. It is specially designed for project that requires frequent maintenance, having unexpected user stories and programming errors. Using these approach, the team's workfl allows minimum completion time for each user story or programming error.

15) What is story points/efforts/ scales?

It is used to discuss the difficulty of the story without assigning actual hours. The most common scale used is a Fibonacci sequence (1,2,3,5,8,13,….100) use linear scale (1,2,3,4….),8 ……Powers)and ofcloth2(1,2,4,size(XS, S ,M,L, XL

16) Explain what is tracer bullet?

The tracer bullet is a spike with the current architecture, the current set of best practices, current technology set which results in production quality code. It is not a throw away c ode but might just be a narrow implementation of the functionality.

17) What is a test stub?

A test stub is a small code that replaces an undeveloped or fully developed component within a system being tested. Test stub is designed in such a way that it mimics the actual component by generating specifically known outputs and substitute the actual component.

18) What are the differences between RUP (Rational Unified Process) and Scrum methodologies?

RUP	SCRUM
– Formal Cycle is defined across four phases, but some workflows can be concurrent	– Each sprint is a complete cycle
– Formal project plan, associated with multiple iterations is used.	– No end to end project plan. Each next iteration plan is determined at the end of the current iteration
– Scope is predefined ahead of the project start and documented in the scope document. During the project, scope can be revised.	– It uses a project backlog instead of scope scrum
– Artifacts include Scope Document, formal functional requirements package, system architecture document, development plan, test scripts, etc.	– Operational software is the only formal artifacts
– Recommended for long term, large, enterprise level projects with medium to high complexity	– Recommended for quick enhancements and organization that are not dependent on a deadline

19) Why Continuous Integration is important for Agile?

Continuous Integration is important for Agile for following reasons

- It helps to maintain release schedule on time by detecting bugs or integration errors
- Due to frequent agile code delivery usually every sprint of 2-3 weeks, stable quality of build is a must and continuous integration ensures that
- In helps to maintain the quality and bug free state of code-base

- Continuous integration helps to check the impact of work on branches to the main trunk if development work is going on branches using automatic building and merging function

20) What testing is done during Agile?

The primary testing activities during Agile is automated unit testing and explora tory testing.

Though, depending on project requirements, a tester may execute Functional and Non - functional tests on the Application Under Test (AUT).

21) Explain what is Velocity in Agile?

Velocity is a metric that is calculated by addition of all efforts estimates related with user stories completed in an iteration. It figures out how much work Agile can complete in a sprint and how much time will it need to finish a project.

22) What are the qualities of a good Agile tester should have?

A good Agile tester should have following qualities

- It should be able to understand the requirements quickly
- Agile tester should know Agile princ ipals and concepts well
- As requirements keep changing, tester should understand the risk involve in it
- Based on the requirements Agile tester should be able to prioritize the work
- Continue communication between bus iness associates, developers and tester is must

23) Who are all involved in the Agile team?

In agile the two main leads are

- **Scrum Masters**: It coordinates most of the inputs and outputs required for an agile program
- **Development Managers** : They hire right people and develop them with the team

24) Mention in detail what are the role's of Scrum Master?

Scrum Master key responsibilities involves

- Understand the requirements and turn them into working software
- Monitoring and Tracking
- Reporting and Communication

- Process Check Master
- Quality Master
- Resolve Impediments
- Resolve Conflicts
- Shield the team and performance feedback
- Lead all the meetings and resolve obstacles

25) Mention what are the Agile quality strategies?

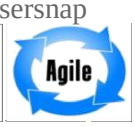
Agile quality strategies are

- Re-factoring
- Non-solo development
- Static and dynamic code analysis
- Reviews and Inspection
- Iteration/sprint demos
- All hands demo
- Light weight milestone reviews
- Short feedback cycles
- Standards and guidelines

26) Mention what are the Tools that can be useful for screenshots while working on Agile projects?

While working on Agile projects you can use tools like

- BugDigger
- BugShooting
- qTrace
- Snagit
- Bonfire
- Usersnap



27) Mention what are the advantages of maintaining consistent iteration length throughout the project?

The advantages are

- It helps team to objectively measure progress
- It provides a consistent means of measuring team veloc ity
- It helps to establish a consistent pattern of delivery

28) If a timebox plan needs to be reprioritized who should re -prioritise it?

If a timebox plan needs to be reprioritized it should include whole team, product owner, and developers.

29) Mention what should a burndown chart should highlight?

The burn-down chart shows the remaining work to complete before the timebox (iteration) ends.

30) Mention what is the difference between Scrum and Agile?

- **Scrum** : In the scrum, a sprint is a basic unit of development. Each sprint is followed by a planning meeting, where the tasks for the sprint are identified and estimated. During each sprint, the team creates finished portion of a product
- **Agile**: In Agile, each iteration involves a team working through a full software development cycle, including planning, design, coding, requirement analys is, unit testing, and acceptance testing when a product is demonstrated to stakeholders

In simple words, Agile is the practice and scrum is the process to following this practice.

31) Mention what are the challenges involved in AGILE software development?

Challenges involved in Agile Software development includes

- It requires more testing and customers involvement
- It impacts management more than developers
- Each feature needs to be completed before moving on to the next
- All the code has to work fine to ensure application is in working state
- More planning is required

32) When not to use Agile?

Before using Agile methodology, you must ask following questions

- Is functionality split-able
- Is customer available
- Are requirements flexible
- Is it really time constrained
- Is team skilled enough

33) Explain how can you implement scrum in an easy way to your project?

These are the tips which can be helpful to implement scrum in your project

- Get your backlog in order
- Get an idea of the size of your product backlog items
- Clarify sprint requirement and duration to complete the sprint backlog
- Calculate the team sprint budget and then break requirements into tasks
- Collaborate workspace- a center of all team discussion, which inc ludes plans, roadmaps, key dates, sketches of functionality, issues, log, status reports, etc.
- Sprint- Make sure you complete one feature at a time before moving on to the next. A sprint should not be abort unless if there is no other option
- Attend a daily stand-up meeting: In meeting you need to mention, what have been achieved since the last meeting, what will they achieve before the next meeting and is anything holding up their progress
- Use burndown chart to track daily progress. From the burndown chart, you can estimate whether you are on track, or you are running behind
- Complete each features well before moving on to the next
- At the end of the sprint- hold a sprint review meeting, mention what is achieved or delivered in the sprint.

34) Explain what does it mean by product roadmap?

A product roadmap is referred for the holistic view of product features that create the product vision.

Difference bwtween Desktop application and Web Based Application DesktopApplication

A native application that executes on a user's local machine. This application may or may not have a network component, although most desktops have some kind of network component these days, even if it's just to update itself online. To keep things simple, let's put it this way: if you need to update the application, an update needs to be downloaded locally.

The biggest d/f b/w Desktop and web application is-Desktop App (DA) is the machine independent, hence every change has only reflects at the machine level.Where as Web App (WA) is the Internet dependent program,hence any change in the program reflects at every where,where it becomes use.
EX.....

Suppose there are 5 machines in DA, 5 time installed individually at every machine and if there is any change made in DA then at every machine change has to be made. In WA where the program or Application at the Server or at the one common machine, then if changes made at only central or server or common machine all the changes get reflected at every clientmachine

.

An example of a desktop app would be MS Word, Adobe PhotoShop, a web browser.

Desktop:

- Normally a stable and relatively fast internet connection, but that's not a guarantee, so you need to test how the app handles online and offline states.
- Different OSes. If you build a Mac app, how far back are you supporting? Does the installer prevent installation on unsupported OSes?
- Different hardware configurations: this can range from having enough RAM, to enough storage space, to differing screen resolutions. What does the top of the line hardware config look like? What does the bottom of the line look like?
- Software conflicts! This is one of the hardest, and generally where the most problems/edge cases creep up. Is that obscure firewall that your end user is using blocking required network access? Is that spyware that's installed and trying to record all outgoing and incoming packets messing up your app's connection?
- Security. If an individual's machine is exploited, does your app store sensitive information locally in an easily read format? An example of this is I've seen several Mac Apps that store passwords in either plain text or or an easily decrypt - able format in a .plist file.

- Permissions. If your app assumes that it can write to /var/log, but some user has done something funky with their system, will it crash when it can't? Don't have your installer change permissions and then assume they'll stay that way!

Webapplication s:

Applications that run 100% within a browser. This is where the waters get muddy, because there are a group of apps on both the desktop and on mobile that are just web apps, running within an app-specific browser. An example of one of these would have been the early versions of the Facebook app -- those were just a bunch of web views running within a browser window. Sometimes those are hard to identify, but a surefire tell is if your app updates without you needing to do anything from the App Store/Play Store.

Now that the longest pre-amble I've ever written is complete, let's get on with answering the question! **Web:**

- Browser version/supported features. Why, oh why won't some corporate IT environments upgrade their users past IE7? Le sigh.
- You can assume that a network connection will be present, but what happens when that's lost? Hey, outages happen -- will your users lose data if one does?
- To some degree, hardware. Responsive design has minimized the impact of differences in screen size, but if somebody is hitting your JS-heavy site using a really slow machine, what's that experience going to be like for the m? Thankfully, you can use analytics to determine whether this should be a concern or not.
- APIs. A lot of web apps these days have an API -- gotta cover that too.
- External services -- a lot of web apps will rely on external apps and infrastructure. What happens if (for example) I'm hosting all of my files with S3, and Amazon has an outage?

Security! Chances are, with a web app, you've got more than one person using the app. How do you prevent one individual's info from being accessible by others? How do you prevent your app from being completely compromised and used to infect your users' computers with virii?

Difference between throw and throws in Java

There are many differences between throw and throws keywords. A list of differences between throw and throws are given below:

No.	throw	throws

1)	Java throw keyword is used to explicitly throw an exception.	Java throws keyword is used to declare an exception.
2)	Checked exception cannot be propagated using throw only.	Checked exception can be propagated with throws.
3)	Throw is followed by an instance.	Throws is followed by class.
4)	Throw is used within the method.	Throws is used with the method signature.
5)	You cannot throw multiple exceptions.	You can declare multiple exceptions e.g. public void method()throws IOException,SQLException.

SQL

6. What is a primary key?

A primary key is a combination of fields which uniquely specify a row. This is a special kind of unique key, and it has implicit NOT NULL constraint. It means, Primary key values cannot be NULL.

7. What is a unique key?

A Unique key constraint uniquely identified each record in the database. This provides uniqueness for the column or set of columns.

A Primary key constraint has automatic unique constraint defined on it. But not, in the case of Unique Key.

There can be many unique constraint defined per table, but only one Primary key constraint defined per table.

8. What is a foreign key?

A foreign key is one table which can be related to the primary key of another table. Relationship needs to be created between two tables by referencing foreign key with the primary key of another table.

9. What is a join?

This is a keyword used to query data from more tables based on the relationship between the fields of the tables. Keys play a major role when JOINs are used.

10. What are the types of join and explain each?

There are various types of join which can be used to retrieve data and it depends on the relationship between tables.

Inner join.

Inner join return rows when there is at least one match of rows between the tables.

Right Join.

Right join return rows which are common between the tables and all rows of Right hand side table. Simply, it returns all the rows from the right hand side table even though there are no matches in the left hand side table.

Left Join.

Left join return rows which are common between the tables and all rows of Left hand side table. Simply, it returns all the rows from Left hand side table even though there are no matches in the Right hand side table.

Full Join.

Full join return rows when there are matching rows in any one of the tables. This means, it returns all the rows from the left hand side table and all the rows from the right hand side table.

14. What is a View?

A view is a virtual table which consists of a subset of data contained in a table. Views are not virtually present, and it takes less space to store. View can have data of one or more tables combined, and it is depending on the relationship.

18. What is a relationship and what are they?

Database Relationship is defined as the connection between the tables in a database. There are various data basing relationships, and they are as follows:.

- One to One Relationship.
- One to Many Relationship.
- Many to One Relationship.
- Self- Referenc ing Relationship.

19. What is a query?

A DB query is a code written in order to get the information back from the database. Query can be designed in such a way that it matched with our expectation of the result set. Simply, a question to the Database.

20. What is subquery?

A subquery is a query within another query. The outer query is called as main query, and inner query is called subquery. SubQuery is always executed first, and the result of subquery is passed on to the main query.

21. What are the types of subquery?

There are two types of subquery – Correlated and Non-Correlated.

A correlated subquery cannot be considered as independent query, but it can refer the column in a table listed in the FROM the list of the main query.

A Non-Correlated sub query can be considered as independent query and the output of subquery are substituted in the main query.

Tell me something about yourself

I am Manas Kumar Jha I was born and bought up in bihar and did my 10 th and 12th from campus public school Pusa Samastipur Bihar that I did my Bachelor of engineering from