1) What is API?

**API** (*Application Programming Interface*) helps in communication and data exchange between two software systems. API act as an interface between two applications and allows the two software systems communicate with one another. API is a collection of functions which can be executed by another software program.

API works as; it takes a request from the source, takes that request to the database, fetches the request data from the database and returns a response to the source. API takes the requests from the user and gives the response without exposing the internal details. API acts as Abstraction.

### 2) What is API testing?

**API testing** is a type of software testing that involves testing APIs directly. API is a part of integration testing to check whether the API meets expectations in terms of functionality, reliability, performance, and security of applications. Multiple API system can performed API testing. In API testing, our primary focus is on Business Logic Layer of the software architecture.

### 3) What are the types of API testing?

API testing involves the following types of testing:

* Unit Testing
* Functional Testing
* Load Testing
* Runtime/Error Detection
* Security Testing
* UI Testing
* Interoperability and WS compliance Testing
* Penetration Testing
* Fuzz Testing

4) What are the protocols used in API Testing?

Protocols used in API testing are:

* HTTP
* REST
* SOAP
* JMS
* UDDI

### 6) What is API test environment?

For API the test environment is a quite complex method where the configuration of server and database is done as per the requirement of the software application. API testing does not involve graphical user interface (GUI).

API is checked for its proper functioning after installation.

### 7) What is API framework?

API framework is described by the config. File which consist of the list of all APIs that are required to be activated and are activated for any particular program run. This is essential as every test run does not require all APIs.

### 8) What are the limits of API usage?

Many APIs have certain limit set up by the provider. Hence, try to estimate our usage and understand how that will impact the overall cost of the offering.

### 9) What are the advantages of API testing?

Advantages of API testing are:

* **Test for core functionality:** API testing provides access to the application without the user interface. The core functionality of the application will be tested before the GUI tests. This will help to detect the minor issue which can become bigger during the GUI testing.
* **Time effective:** API testing is less time consuming than GUI testing. Particularly, API test requires less code so it can provide better and faster test coverage compare to GUI test automation. This will reduce the cost for the testing project.
* **Language Independent:** In API testing data is exchange using XML or JSON. These transfer mode are completely language-independent, which allows users to select any code language when adopting automation test service for the project.
* **Easy Integration with GUI:** API tests provide highly integrable tests which is useful to perform functional GUI tests after GUI tests. Simple integration would allow new user accounts to be created within the application before GUI started.

### 10) What are the principles of an API test design?

Here, are the seven principles of API test design.

1. **Exhaustive Testing:** Exhaustive testing is not possible. Instead we need optimal amount of testing which is based on the risk assessment of the application.
2. **Defect Clustering:** Defect Clustering states that a small number of modules contain the most of the defect detected. Approximately 80% of the defect found in 20% of the modules. By experience we can identify such risky modules. But this approach has its own problems. If the same tests are repeated over and over again, eventually the same test case will no longer find new bugs.
3. **Pesticide Paradox:** Testers cannot depend on existing technique. They must have to look continually to improve the existing method to make testing more effective. But even all these hard work in testing we can never claim our product is bug free. To overcome this, test cases need to be regularly reviewed and revised add new and different test cases to help find more defects.
4. **Testing shows presence of defects:** Testing principle states that- testing talks about the presence of defects not about the absence of defect. Software testing reduces the probability of undiscovered defects remaining in the software but even if no defects found, it is not a proof of correctness.

But if we work hard, taking all precautions and make our software products 99% bug free. The software does not meet the needs and requirements of the client.

1. **Absence of error -fallacy:** This can be possible the software which is 99% bug free is still unusable. The case can be if the system is tested for the wrong requirement. Software testing is not finding the defects but also to check that software addresses the business needs. The absence of error is fallacy i.e. finding and fixing defects does not help if the system build is unusable and doesn't fulfill the user's needs and requirements.
2. **Early Testing:** Testing should start as soon as possible in the software development lifecycle. So that defects in the requirement or design phase captured in the early stages. It is cheaper to fix defect in the early stages of testing. We should start finding the bug at the moment the requirements are defined.
3. **Testing is context dependent:** Testing is context dependent that we test an e-commerce site will be different from the way we test the commercial. All the developed software's are not identical. We will use different methodology; techniques and type of testing depend on the application type.

### 11) What is API framework?

A framework or software framework is a platform for developing software applications. API framework is a foundation on which software developer can build applications for a specific platform.

**Example:** A framework can include predefined classes and functions that can be used to process input, manage hardware devices and interact with system software.

Framework is similar to an Application Programming Interface, technically framework includes API. Framework serves foundation for programming while API provides access to the elements supported by the framework. Framework also includes code libraries, compiler and other programs used in the software development process.

API framework is defined by configuration file which consists the list of all APIs that is required to be activated and activated for a particular program run.

### 12) What are the common tests that performed on API?

Here, are the common tests that performed on API are as:

1. Response of the API should be verified based on the request. We will verify that the return value is based on request.
2. When API is updating any data structure we should verify the system is authenticating the outcome.
3. We will verify whether the API is trigger other event or request another API.
4. We will verify the behavior of the API when no value is return.

### 13) What exactly needs to verify in API testing?

In API testing, we send a request to API with the known data and then analysis the response.

1. We will verify the accuracy of the data.
2. Will see the HTTP status code.
3. We will see the response time.
4. Error codes in case API returns any errors.
5. Authorization would be check.
6. Non-Functional testing such as performance testing, security testing.

### 14) What are the differences between API and Web Services?

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **API** | **Web Services** |
| **1.** | API may or may not need network for its operations. | Web Services always need network for its operation. |
| **2.** | API can be communicated through SOAP, REST, XML-RPC and CURL calls as well. API can also be exposed in number of ways like JAR, DLL, XML over HTTP, JSON over HTTP etc. | Web service can be communicated through SOAP, REST, AND RPC. |
| **3.** | API can perform all the operations which web service can't perform. | Web service can't perform all the operations like API. |
| **4.** | All APIs are not web service. | All web services are API |

### 17) What are the types of bug that can be found during API testing?

API testing helps us to find many types of bugs which are:

* Stress
* Security
* Duplicate or missing functionality
* Reliability
* Unused flags
* Performance
* Incompatible error handling
* Multi-threaded issue
* Improper errors

### 18) What are the difference between API testing and UI testing?

UI (User Interface) testing means the testing of the graphical user interface. The focus of UI testing is on the look and feel of the application. In user interface testing the main focus is on how users can interact with app elements such as images, fonts, layout etc. are checked.

API testing allows the communication between two software systems. API testing works on backend also known as backend testing.

19) What is SOAP?

SOAP (Simple Object Access Control) . It is an XML based protocol that helps in exchanging information among computers.

20) What is REST API?

**REST API** is a set of function helps the developers performing requests when the response is receiving. Through HTTP protocol interaction is made in REST API.

REST is defined as Representational state transfer. It is an effective standard for API creation.

21) What are the differences between SOAP and REST API?

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **SOAP API** | **REST API** |
| **1.** | **SOAP** stands as Simple Object Access Protocol. | **REST** stands as Representational State Transfer. |
| **2.** | **SOAP** is a protocol. | **REST** is an architectural pattern. |
| **3.** | **SOAP** can work with XML format. In SOAP all the data passed in XML format. | **REST** permit different data format such as Plain text, HTML, XML, JSON etc. But the most preferred format for transferring data is in JSON. |

### 24) What is a RESTFUL web services?

There are two kinds of web services

1. SOAP Web Services
2. RESTFUL Web Services

**1. SOAP (Simple Object Access Protocol) -** SOAP is a XML based method which is used in Web Services.

**2. RESTFUL Web Services -** To implement the concept of REST architecture HTTP method is used. RESTFUL Web Services defines URI (Uniform Resource Identifier), and also provides resource representation like JSON and a set of HTTP method.

### 30) What are the components of an HTTP request?

An HTTP request have five components. These are:

1. **Action showing HTTP method** like GET, PUT, POST, DELETE.
2. **Uniform Resource Identifier (URI):** URI is the identifier for the resource on the server.
3. **HTTP version:** Indicate the HTTP version like- HTTP V1.1.
4. **Request Header:** Request Header carries metadata for the HTTP request message. Metadata could be a client type, format supported by the client, format of a message body, cache setting etc.
5. **Request Body:** Resource body indicates message content or resource representation.

### 31) What is the HTTP protocol supported by REST?

**GET:** GET is used to request data from the specified resource.

GET request can be cached and bookmark. It remains in the browser history and has length restriction. When dealing with sensitive data GET requests should not be used.

**POST:** POST is used to send data to server for creation or updating the resources.

POST requests are never cached or bookmark.

**PUT:** PUT replaces the current representation of the target resource with the request payload.

**DELETE:** DELETE removes the specified resource.

**OPTIONS:** OPTION is used to describe the communication option for the target resources.

**HEAD:** HEAD asks for response which is identical to GET requests, but without the response body.

### 32) Can we use GET request instead of PUT to create a resource?

PUT or POST method is used create a resource. GET is only used to request the resources.

### 4. What are some architectural styles for creating a Web API?

This is one of the fundamental Web API interview questions. Bellows are four common Web API architectural styles:

* HTTP for client-server communication
* XML/JSON as formatting language
* Simple URI as the address for the services
* Stateless communication

### 9. What is the test environment of API?

Setting up the API’s test environment is not an easy task, so you should have a ready answer if your API testing interview is coming. The test environment of API is a bit complete and requires the configuration of the database and server, depending on the software requirements. No GUI (Graphical User Interface) is available in this test form.

When the installation process is complete, API is verified for the proper operation. Throughout the process, the API called from the original environment is set up with different parameters to study the test results.

### 10. What are principles of an API test design?

The five most important principles of an API test design are:

* Setup: Create objects, start services, initialize data, etc
* Execution: Steps to apply API or the scenario, including logging
* Verification: Oracles to evaluate the result of the execution
* Reporting: Pass, failed or blocked
* Clean up: Pre-test state

### 12. What is the procedure to perform API testing?

1. Choose the suite to add the API test case
2. Choose the test development mode
3. Demand the development of test cases for the required API methods
4. Configure the control parameters of the application and then test conditions
5. Configure method validation
6. Execute the API test
7. Check test reports and filter API test cases
8. Arrange all API test cases

### 35. What are the most commonly used HTTP methods supported by REST?

* GET is only used to request data from a specified resource. Get requests can be cached and bookmarked. It remains in the browser history and haS length restrictions. GET requests should never be used when dealing with sensitive data.
* POST is used to send data to a server to create/update a resource. POST requests are never cached and bookmarked and do not remain in the browser history.
* PUT replaces all current representations of the target resource with the request payload.
* DELETE removes the specified resource.
* OPTIONS is used to describe the communication options for the target resource.
* HEAD asks for a response identical to that of a GET request, but without the response body.

### 38. Which purpose does the OPTIONS method serve for the RESTful Web services?

The OPTIONS Method lists down all the operations of a web service supports. It creates read-only requests to the server.

### 40. What is payload in RESTFul Web services?

The “payload” is the data you are interested in transporting. This is differentiated from the things that wrap the data for transport like the HTTP/S Request/Response headers, authentication, etc.

### 61. What are the factors that help to decide which style of Web services – SOAP or REST – to use?

Generally, REST is preferred due to its simplicity, performance, scalability, and support for multiple data formats.

However, SOAP is favorable to use where service requires an advanced level of security and transactional reliability.

But you can read the following facts before opting for any of the styles.

* **Does the service expose data or business logic?** REST is commonly used for exposing data while SOAP for logic.
* **The requirement from clients or providers for a formal contract**. SOAP can provide contract via WSDL.
* **Support multiple data formats**.
* **Support for AJAX calls.** REST can apply the XMLHttpRequest.
* **Synchronous and asynchronous calls.**SOAP enables both synchronous/ asynchronous operations whereas REST has built-in support for synchronous.
* **Stateless or Stateful calls.** REST is suited for stateless operations.
* **Security.** SOAP provides a high level of security.
* **Transaction support.** SOAP is good at transaction management.
* **Limited bandwidth**. SOAP has a lot of overhead when sending/receiving packets since it’s XML based, requires a SOAP header. However, REST requires less bandwidth to send requests to the server. Its messages are mostly built using JSON.
* **Ease of use**. REST based application is easy to implement, test, and maintain.

SELENIUM:

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

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

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

#### 2) What is Selenium 2.0 ?

Web[Testing](https://www.guru99.com/software-testing.html)tools Selenium RC and WebDriver are consolidated in single tool in Selenium 2.0



#### 4) How will you find an element using Selenium?

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

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

#### 6) Explain what is assertion in Selenium and what are the types of assertion?

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

#### 7) Mention what is the use of X-path?

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

#### 8) Explain the difference between single and double slash in X-path?

Single slash ‘/ ’

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

Double Slash ‘// ’

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

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

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

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

#### 13) While using click command can you use screen coordinate?

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

#### 16) What are the four parameter you have to pass in Selenium?

Four parameters that you have to pass in Selenium are

* Host
* Port Number
* Browser
* URL

#### 17) What is the difference between setSpeed() and sleep() methods?

Both will delay the speed of execution.

Thread.sleep () :  It will stop the current (java) thread for the specified period of time.  Its done only once

* It takes a single argument in integer format

Ex: thread.sleep(2000)- It will wait for 2 seconds

* It waits only once at the command given at sleep

SetSpeed () :  For specific amount of time it will stop the execution for every selenium command.

* It takes a single argument in integer format

Ex: selenium.setSpeed(“2000”)- It will wait for 2 seconds

* Runs each command  after setSpeed delay by the number of milliseconds mentioned in set Speed

#### 18) What is same origin policy? How you can avoid same origin policy?

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

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

#### 22) Mention what is the difference between Implicit wait and Explicit wait?

Implicit Wait: Sets a timeout for all successive Web Element searches. For the specified amount of time it will try looking for element again and again before throwing a NoSuchElementException.  It waits for elements to show up.

Explicit Wait :  It is a one-timer, used for a particular search.

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

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

#### 24) Explain what is the difference between find elements () and find element () ?

find element ():

It finds the first element within the current page using the given “locating mechanism”.  It returns a single WebElement

findElements () : Using the given “locating mechanism” find all the elements within the current page.  It returns a list of web elements.

#### 26) Explain what is Datadriven framework and Keyword driven?

**Datadriven framework:**  In this framework, the test data is separated and kept outside the Test Scripts, while[Test Case](https://www.guru99.com/test-case.html)logic resides in Test Scripts.  Test data is read from the external files ( Excel Files) and are loaded into the variables inside the Test Script.  Variables are used for both for input values and for verification values.

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

#### 30) What is Object Repository ?

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

#### 36)  Explain how you can find broken images in a page using Selenium Web driver ?

To find the broken images in a page using Selenium web driver is

* Get XPath and get all the links in the page using tag name
* In the page click on each and every link
* Look for 404/500 in the target page title

#### 39)  Explain how you can switch between frames?

To switch between frames webdrivers **[ driver.switchTo().frame() ]** method takes one of the three possible arguments

* A number:  It selects the number by its (zero-based) index
* A name or ID: Select a frame by its name or ID
* Previously found WebElement: Using its previously located WebElement select a frame

#### 40)  Mention 5 different exceptions you had in Selenium web driver?

The 5 different exceptions you had in Selenium web drivers are

* WebDriverException
* NoAlertPresentException
* NoSuchWindowException
* NoSuchElementException
* TimeoutException

#### 41)  Explain using Webdriver how you can perform double click ?

You can perform double click by using

* **Syntax- Actions act = new Actions (driver);**
* **act.doubleClick(webelement);**

#### 45) What is the difference between getWindowhandles() and getwindowhandle() ?

getwindowhandles(): It is used to get the address of all the open browser and its return type is Set<String>

getwindowhandle(): It is used to get the address of the current browser where the control is and return type is string

#### 46) Explain how you can switch back from a frame?

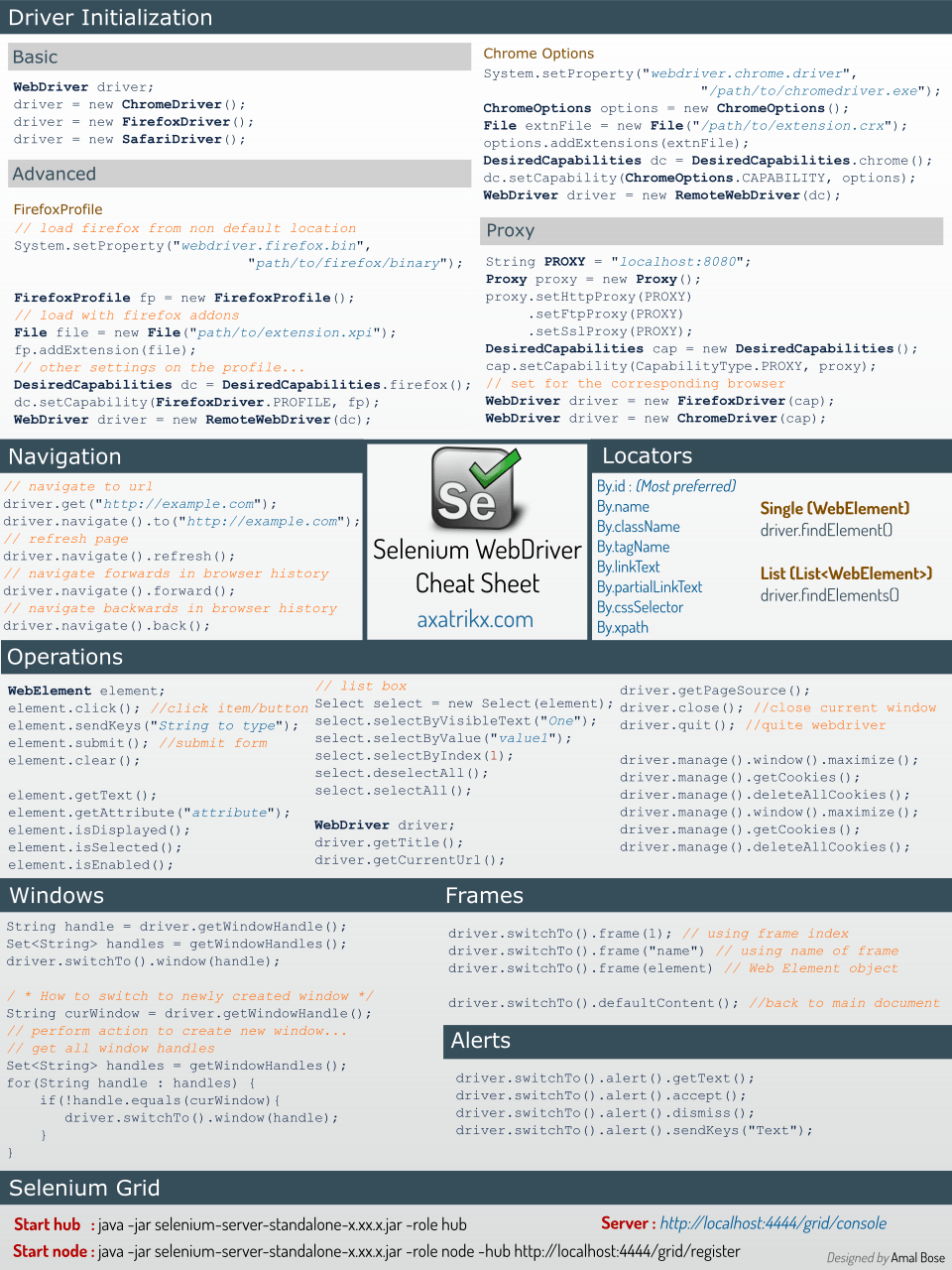
To switch back from a frame use method defaultContent()

Syntax-driver.switchTo().defaultContent();

#### 47) List out different types of locators?

Different types of locators are

* By.id()
* By.name()
* By.tagName()
* By.className()
* By.linkText()
* By.partialLinkText()
* By.xpath
* By.cssSelector()



### 11) What do you mean by Selenese?

Selenium commands, also known as "Selenese" are the set of commands used in Selenium that run your tests. For example, command - open (URL); launches the desired URL in the specified browser and it accept both relative and absolute URLs.

A sequence of Selenium commands (Selenese) together is known as a test script.

### 33) Write a code snippet to perform right-click an element in WebDriver.

We will use **Action class** to generate user event like right-click an element in WebDriver.

1. Actions action = newActions(driver);
2. WebElement element = driver.findElement(By.id("elementId"));
3. action.contextClick(element).perform();

### 34) Write a code snippet to perform mouse hover in WebDriver.

1. Actions action = newActions(driver);
2. WebElement element = driver.findElement(By.id("elementId"));
3. action.moveToElement(element).perform();

### 35) How do you perform drag and drop operation in WebDriver?

Code snippet to perform drag and drop operation:

1. //WebElement on which drag and drop operation needs to be performed
2. WebElementfromWebElement = driver.findElement(By Locator of fromWebElement);
4. //WebElement to which the above object is dropped
5. WebElementtoWebElement = driver.findElement(By Locator of toWebElement);
7. //Creating object of Actions class to build composite actions
8. Actions builder = newActions(driver);
10. //Building a drag and drop action
11. Action dragAndDrop = builder.clickAndHold(fromWebElement)
12. .moveToElement(toWebElement)
13. .release(toWebElement)
14. .build();
16. //Performing the drag and drop action
17. dragAndDrop.perform();

### 36) What are the different methods to refresh a web page in WebDriver?

There are multiple ways of refreshing a page in Webdriver.

1. Using driver.navigate command -

1. driver.navigate().refresh();

2. Using driver.getCurrentUrl() with driver.get() command -

1. driver.get(driver.getCurrentUrl());

3. Using driver.getCurrentUrl() with driver.navigate() command -

1. driver.navigate().to(driver.getCurrentUrl());

4. Pressing an F5 key on any textbox using the sendKeys command -

1. driver.findElement(By textboxLocator).sendKeys(Keys.F5);

5. Passing ascii value of the F5 key, i.e., "\uE035" using the sendKeys command -

1. driver.findElement(By textboxLocator).sendKeys("\uE035");

### 46) What is POM (Page Object Model)? What are its advantages?

Page Object Model is a design pattern for creating an Object directory for web UI elements. Each web page is required to have its page class. The page class is responsible for finding the WebElements in web pages and then perform operations on WebElements.

The benefits of using POM are as follows.

* It facilitates with separate operations and flows in the UI from Verification - improves code readability
* Multiple tests can use the same Object Repository because the Object Repository is independent of Test Cases.
* Reusability of code

### How to capture screenshot in WebDriver?

public class TakeScreenshot {

WebDriver drv;

@Before

public void setUp() throws Exception {

driver = new FirefoxDriver();

drv.get("https://google.com");

}

@After

public void tearDown() throws Exception {

drv.quit();

}

@Test

public void test() throws IOException {

//capture the screenshot

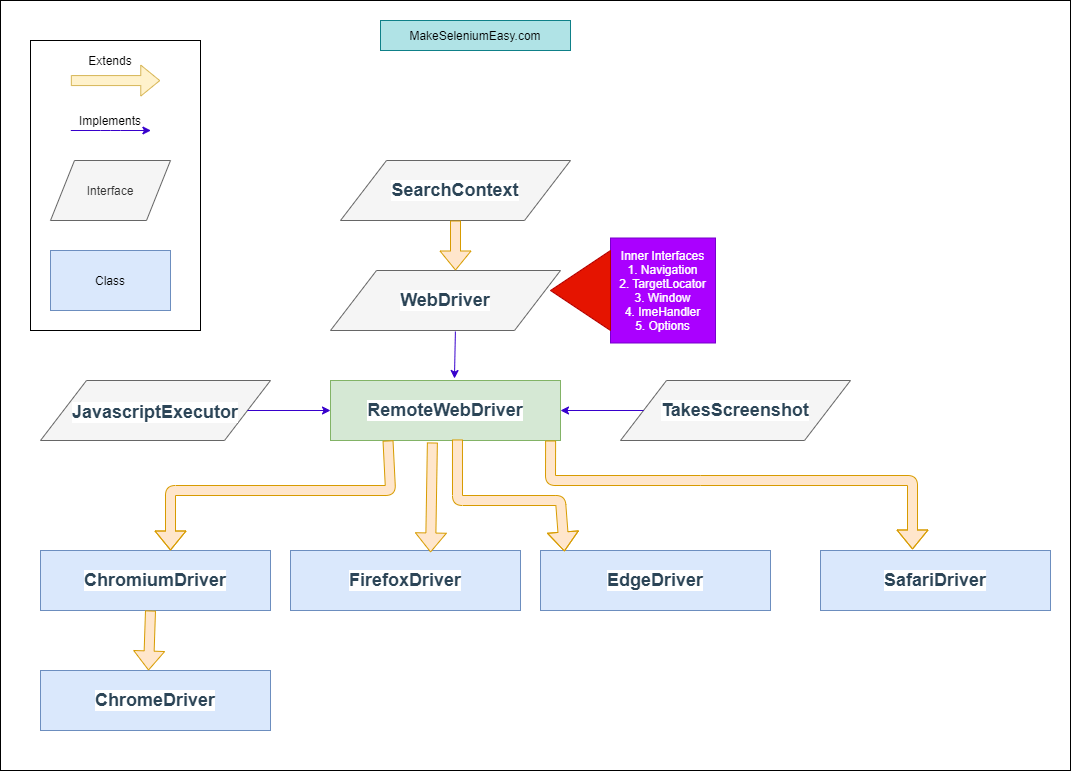
File scrFile = ((TakeScreenshot)drv).getScreenshotAs(OutputType.FILE);

// paste the screenshot in the desired location

FileUtils.copyFile(scrFile, new File("C:\\Screenshot\\Scr.jpg"))

}

}



#### Interface SearchContext

**SearchContext***is the top most interface in WebDriver hierarchy. This interface consists of two methods****findElement*​(By by)***and***findElements(By by)***. This interface is extended by both WebDriver and WebElement interfaces.*

#### Interface WebDriver

**WebDriver***is an interface which extends***SearchContext***interface. As per*[*official document*](https://www.javadoc.io/doc/org.seleniumhq.selenium/selenium-api/latest/org/openqa/selenium/WebDriver.html)*:- WebDriver is a remote control interface that enables introspection and control of user agents (browsers). The methods in this interface fall into three categories:*

* Control of the browser itself
* Selection of WebElement and WebElements
* Debugging aids

*WebDriver interface has multiple inner interfaces which contains methods related o specific events.*

1. **ImeHandler** – An interface for managing input methods.
2. **Navigation**– An interface for provide mechanism to access browser history.
3. **Options**– An interface for managing stuff you would do in a browser menu
4. **TargetLocator**– Used to locate a given frame or window.
5. **Timeouts**– An interface for managing timeout behavior for WebDriver instances.
6. **Window**– An interface to manage browser window actions like maximize, minimize etc.

#### Class RemoteWebDriver

*RemoteWebDriver is the fully implemented class i.e. non -abstract class which implements WebDriver interface. All implementations of WebDriver that communicate with the browser, or a RemoteWebDriver server shall use a common wire protocol. This wire protocol defines a RESTful web service using JSON over HTTP.*

Browser’s specific classes

Then we have browser specific driver classes like ChromeDriver(), EdgeDriver(), FirefoxDriver() etc in hierarchy. These classes provides control on a browser running on the local machine. These class are provided as a convenience for easily testing the browsers. The control serves (which is started every time when we launch a browser ) which each instance communicates with will live and die with the instance.

### types of waits supported by WebDriver?

**Implicit wait** - Implicit wait commands Selenium to wait for a certain amount of time before throwing a “No such element” exception.

**driver.manage().timeouts().implicitlyWait(TimeOut, TimeUnit.SECONDS);**

**Explicit wait**- Explicit wait is used to tell the Web Driver to wait for certain conditions before throwing an "ElementNotVisibleException" exception.

**WebDriverWait wait = new WebDriverWait(WebDriver Reference, TimeOut);**

**Fluent wait** - It is used to tell the web driver to wait for a condition, as well as the frequency with which we want to check the condition before throwing an "ElementNotVisibleException" exception.

**Wait wait = new FluentWait(WebDriver reference).withTimeout(timeout, SECONDS).pollingEvery(timeout, SECONDS).ignoring(Exception.class);**

### scroll down a page using JavaScript?

**scrollBy()** method is used to scroll down the webpage

General syntax:

**executeScript("window.scrollBy(x-pixels,y-pixels)");**

First, create a JavaScript object

**JavascriptExecutor js = (JavascriptExecutor) driver;**

Launch the desired application

**driver.get(“https://www.amazon.com”);**

Scroll down to the desired location

**js.executeScript("window.scrollBy(0,1000)");**

The window is not scrolled vertically by 1000 pixels

### assert the title of a webpage?

Get the title of the webpage and store in a variable

**String actualTitle = driver.getTitle();**

Type in the expected title

**String expectedTitle = “abcdefgh";**

Verify if both of them are equal

**if(actualTitle.equalsIgnoreCase(expectedTitle))**

 **System.out.println("Title Matched");**

 **else**

 **System.out.println("Title didn't match");**

Alternatively,

**Assert.assertEquals(actualTitle, expectedTitle);**

### 15. How to mouse hover over a web element?

**Actions class** utility is used to hover over a web element in Selenium WebDriver

Instantiate Actions class.

**Actions action = new Actions(driver);**

In this scenario, we hover over**search box** of a website

**actions.moveToElement(driver.findElement(By.id("id of the searchbox"))).perform();**

### select a value from a dropdown in Selenium WebDriver?

**Select**class in WebDriver is used for **selecting and deselecting** options in a dropdown.

The objects of Select type can be initialized by passing the dropdown webElement as a parameter to its constructor.

**WebElement testDrop = driver.findElement(By.id("testingDropdown"));**

**Select dropdown = new Select(testDrop);**

WebDriver offers three ways to select from a dropdown:

**selectByIndex**: Selection based on index starting from 0

**dropdown.selectByIndex(5);**

**selectByValue:** Selection based on value

**dropdown.selectByValue(“Books”);**

**selectByVisibleText**: Selection of option that displays text matching the given argument

**dropdown.selectByVisibleText(“The Alchemist”);**

### 23. What does the switchTo() command do?

**switchTo()**command is used to switch between windows, frames or pop-ups within the application. Every window instantiated by the WebDriver is given a unique alphanumeric value called **“Window Handle”**.

Get the window handle of the window you wish to switch to

**String  handle= driver.getWindowHandle();**

Switch to the desired window

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

Alternatively

**for(String handle= driver.getWindowHandles())**

 **{ driver.switchTo().window(handle); }**

### 24. How to upload a file in Selenium WebDriver?

You can achieve this by using **sendkeys()** or**Robot class** method. Locate the text box and set the file path using **sendkeys()** and click on submit button

Locate the browse button

**WebElement browse =driver.findElement(By.id("uploadfile"));**

Pass the path of the file to be uploaded using sendKeys method

**browse.sendKeys("D:\\SeleniumInterview\\UploadFile.txt");**

### 25. How to set browser window size in Selenium?

The window size can be maximized, set or resized

To maximize the window

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

To set the window size

**Dimension d = new Dimension(400,600);**

**driver.manage().window().setSize(d);**

Alternatively,

The window size can be reset using JavaScriptExecutor

**((JavascriptExecutor)driver).executeScript("window.resizeTo(1024, 768)");**

### 28. How to login to any site if it is showing an Authentication Pop-Up for Username and Password?

To handle authentication pop-ups, verify its **appearance and then handle**them using an explicit**wait command**.

Use the explicit wait command

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

Alert class is used to verify the alert

**Alert alert = wait.until(ExpectedConditions.alertIsPresent());**

Once verified, provide the credentials

**alert.authenticateUsing(new UserAndPassword(<username>, <password>));**

Navigate to the interested webpage for e.g. **www.amazon.com**

Collect all the links from the webpage. All the links are associated with the Tag ‘a‘

**List<WebElement> links = driver.findElements(By.tagName("a"));**

Create a list of type WebElement to store all the Link elements in it.

**for(int i=0; i<links.size(); i++) {**

 **WebElement element = links.get(i);**

 **String url=element.getAttribute("href");**

 **verifyLink(url);  }**

Now Create a Connection using URL object( i.e ., link)

**URL link = new URL(urlLink);**

Connect using Connect Method

**HttpURLConnection httpConn =(HttpURLConnection)link.openConnection();**

Use **getResponseCode ()** to get response code

**if(httpConn.getResponseCode()!== 200)**

Through exception, if any error occurred

**System.out.println(“Broken Link”);**

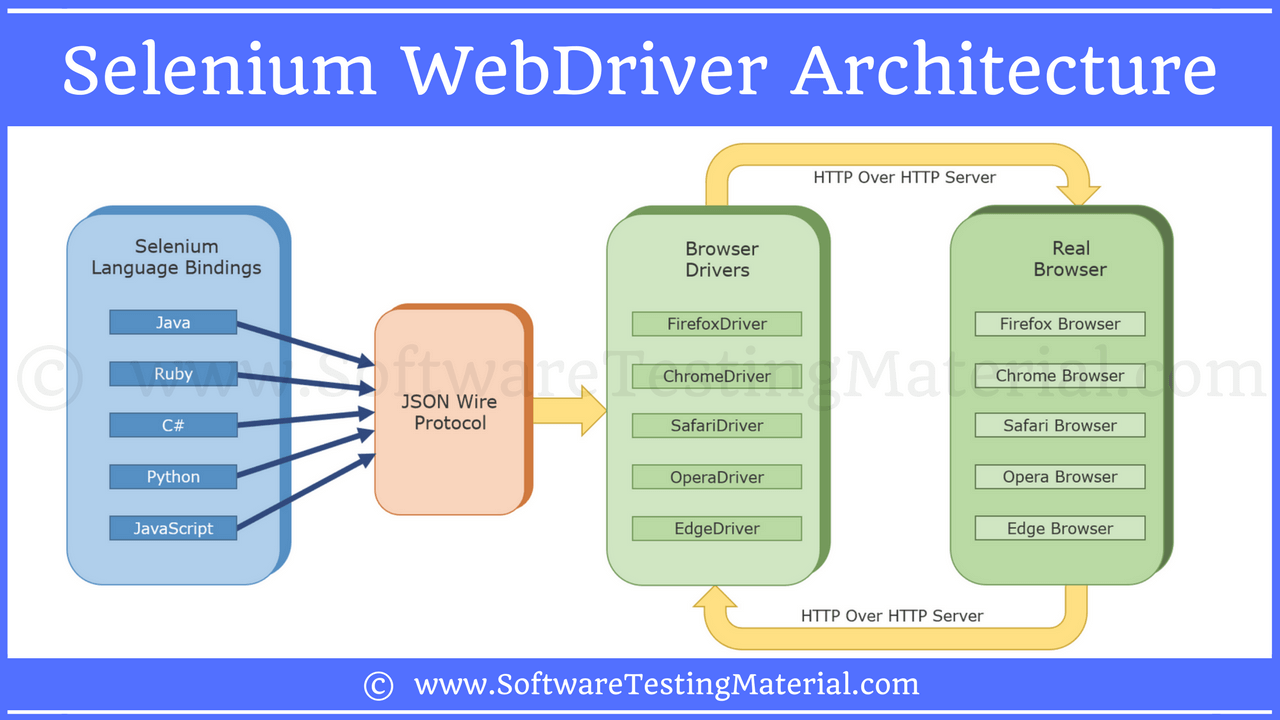
**WebDriver driver = new WebDriver();**

We cannot write our code like this because we cannot create Object of an Interface. WebDriver is an interface.

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

We can create Object of a class FirefoxDriver by taking reference of an interface (WebDriver). In this case, we can call implemented methods of WebDriver interface.

WebDriver is an interface and all the methods which are declared in Webdriver interface are implemented by respective driver class. But if we do upcasting,we can run the scripts in any browser .i.e running the same automation scripts in different browsers to achieve [Runtime Polymorphism](https://www.softwaretestingmaterial.com/polymorphism-in-java/).



### ****53. What happen if you mix both implicit wait and explicit wait in a Selenium Script?****

As per the official Selenium documentation, it is suggested not to mix both Implicit waits and Explicit Waits. Mixing both of them can cause unpredictable wait times.

Implicit wait is defined only once in the code. It will remain same throughout the driver object instance.

Explicit wait is defined whenever it is necessary in the code. This wait will call at the time of execution. It is a conditional wait.

Explicit wait will overwrite the implicit wait where ever explicit wait is applied. So, Explicit Wait gets first preference then Implicit Wait.

MANUAL TESTING:

### ****3. What do verification and validation mean in software testing?****

In software testing, verification is a process to confirm that product development is taking place as per the specifications and using the standard development procedures. The process comprises the following activities:

* Inspections
* Reviews
* Walk-throughs
* Demos

### Validation is a means to confirm that the developed product doesn’t have any bugs and is working as expected. It comprises the following activities:

* Functional testing
* Non-functional testing

### ****4. What is static testing? When does it start and what does it cover?****

Static testing is a white-box testing technique that directs developers to verify their code with the help of a checklist to find errors in it. Developers can start the static testing without actually finalizing the application or program. Static testing is more cost-effective than dynamic testing as it more areas than dynamic testing in a shorter time.

### ****5. Define Black-box testing.****

It is a standard software testing approach that requires testers to assess the functionality of the software as per the business requirements. The software is treated as a black box and validated as per the end user’s point of view.

### ****6. What is a test plan and what does it include?****

A test plan stores all possible testing activities to ensure a quality product. It gathers data from the product description, requirement, and use case documents.

The test plan document includes the following:

* Testing objectives
* Test scope
* Testing the frame
* Environment
* Reason for testing
* Criteria for entrance and exit
* Deliverables
* Risk factors

### ****7. What is meant by test coverage?****

Test coverage is a quality metric to represent the amount (in percentage) of testing completed for a product. It is relevant for both functional and non-functional testing activities. This metric is used to add missing test cases.

### ****11. Mention the different types of software testing.****

**Various testing types used by manual testers are as follows:**

* Unit testing
* Integration testing
* Regression testing
* Shakeout testing
* Smoke testing
* Functional testing
* Performance testing
  + Load testing
  + Stress testing
  + Endurance testing
* White-box and Black-box testing
* Alpha and Beta testing
* System testing

### ****12. What is the difference between a test driver and a test stub?****

The **test driver** is a section of code that calls a software component under test. It is useful in testing that follows the bottom-up approach.

The **test stub** is a dummy program that integrates with an application to complete its functionality. It is relevant for testing that uses the top-down approach.

For example:

1. Let’s assume a scenario where we have to test the interface between Modules A and B. We have developed only Module A. Here, we can test Module A if we have the real Module B or a dummy module for it. In this case, we call Module B as the test stub.
2. Now, Module B can’t send or receive data directly from Module A. In such a scenario, we’ve to move data from one module to another using some external features called test driver.

A bug has its life cycle from the point when the bug is logged in to the point the bug is closed. Bug undergoes the following states:

* New
* Assigned
* Open
* Fixed
* Retesting
* Reopen
* Verified
* Closed

### ****difference between retesting and regression testing?****

Possible differences between retesting and regression testing are as follows:

* We perform **retesting** to verify the defect fixes. But, the regression testing assures that the bug fix does not break other parts of the application.
* **Regression**test cases verify the functionality of some or all modules.
* **Regression** testing ensures the re-execution of passed test cases. Whereas, **retesting** involves the execution of test cases that are in a failed state.
* **Retesting** has a higher priority over **regression**. But in some cases, both get executed in parallel.

### ****the different types of functional testing?****

Functional testing covers the following types of validation techniques:

* Unit testing
* Smoke testing
* UAT
* Sanity testing
* Interface testing
* Integration testing
* System testing
* Regression testing

### ****functional test cases and non-functional test cases?****

* Functional testing: It is testing the ‘functionality’ of a software or an application under test. It tests the behavior of the software under test. Based on the requirement of the client, a document called a software specification or requirement specification is used as a guide to test the application.
* Non-functional testing: In software terms, when an application works as per the user’s expectation, smoothly and efficiently under any condition, then it is stated as a reliable application. Based on quality, it is very critical to test these parameters. This type of testing is called non-functional testing.

### ****26. What do you understand by STLC?****

Software testing life cycle (STLC) proposes the test execution in a planned and systematic manner. In the STLC model, many activities occur to improve the quality of the product.

**The STLC model lays down the following steps:**

1. Requirement Analysis
2. Test Planning
3. Test Case Development
4. Environment Setup
5. Test Execution
6. Test Cycle Closure

### ****28. Difference between Bug, Defect, and Error.****

A slip in coding is indicated as an error. The error spotted by a manual tester becomes a defect. The defect which the development team admits is known as a bug. If a built code misses on the requirements, then it is a functional failure.

### ****29. How do severity and priority relate to each other?****

**Severity:** It represents the gravity/depth of a bug. It describes the application point of view.

**Priority:** It specifies which bug should get fixed first. It defines the user’s point of view.

### ****30. List the different types of severity.****

The criticality of a bug can be low, medium, or high depending on the context.

* User interface defects – Low
* Boundary related defects – Medium
* Error handling defects – Medium
* Calculation defects – High
* Misinterpreted data – High
* Hardware failures – High
* Compatibility issues – High
* Control flow defects – High
* Load conditions – High

### ****35. Is there any difference between quality assurance, quality control, and software testing. If so, what is it?****

Quality Assurance (QA) refers to the planned and systematic way of monitoring the quality of the process which is followed to produce a quality product. QA tracks the test reports and modifies the process to meet the expectation.

Quality Control (QC) is relevant to the quality of the product. QC not only finds the defects but suggests improvements too. Thus, a process that is set by QA is implemented by QC. QC is the responsibility of the testing team.

Software testing is the process of ensuring that the product which is developed by developers meets the users’ requirements. The aim of performing testing is to find bugs and make sure that they get fixed. Thus, it helps to maintain the quality of the product to be delivered to the customer.

**6. What is white box testing and list the types of white box testing?**

White box testing technique involves selection of test cases based on an analysis of the internal structure (Code coverage, branches coverage, paths coverage, condition coverage, etc.) of a component or system. It is also known as Code-Based testing or Structural testing. Different types of white box testing are

1. Statement Coverage
2. Decision Coverage

**8. What is black box testing? What are the different black box testing techniques?**

Black box testing is the software testing method which is used to test the software without knowing the internal structure of code or program. This testing is usually done to check the functionality of an application. The different black box testing techniques are

1. Equivalence Partitioning
2. Boundary value analysis
3. Cause-effect graphing

**9. What is the difference between static and dynamic testing?**

Static testing: During Static testing method, the code is not executed, and it is performed using the software documentation.

Dynamic testing: To perform this testing the code is required to be in an executable form.

**How can we group test cases like separate test cases for Sanity suite, Regression suite, etc?**  
Ans. Using **groups** attribute in TestNG, we can assign the test methods to different groups.

//Test method belonging to sanity suite only

@Test(groups = {"sanitySuite"})

public void testMethod1() {

//Test logic

}

//Test method belonging to both sanity and regression suite

@Test(groups = {"sanitySuite", "regressionSuite"})

public void testMethod2() {

//Test logic

}

**Ques.6. How can we exclude a Test method from getting executed via testng.xml file?**  
Ans. Using the **exclude** tag in testng.xml file, we can exclude a particular test method from getting executed.

<suite name="Test Suite" verbose="1">

<test name="TestName">

<classes>

<class name="TestClassName">

<methods>

<exclude name="testMethodToBeExcluded"/>

</methods>

</class>

</classes>

</test>

</suite>

**Ques.7. What are some commonly used TestNG annotations?**  
Ans. The commonly used TestNG annotations are-

* @Test – @Test annotation marks a method as a Test method.
* @BeforeSuite – The annotated method will run only once before all tests in this suite have run.
* @AfterSuite – The annotated method will run only once after all tests in this suite have run.
* @BeforeClass – The annotated method will run only once before the first test method in the current class is invoked.
* @AfterClass – The annotated method will run only once after all the test methods in the current class have been run.
* @BeforeTest – The annotated method will run before any test method belonging to the classes inside the <test> tag is run.
* @AfterTest – The annotated method will run after all the test methods belonging to the classes inside the <test> tag have run.
* **@**BeforeMethod – The annotated method will run before each test method marked by @Test annotation.
* **@**AfterMethod – The annotated method will run after each test method marked by @Test annotation.
* @DataProvider – The @DataProvider annotation is used to pass test data to the test method. The test method will run as per the number of rows of data passed via the data provider method.

What are the categories of annotations in TestNG?

TestNG annotations divide into three categories:

Precondition Annotations: The annotations under this category execute before the test. It consists of the following annotations:

@BeforeMethod

@BeforeClass

@BeforeSuite

@BeforeTest

Test Annotations: The annotations under this category are defined just before the test methods. Moreover, it consists of the following annotations:

@Test

Postcondition Annotations: The annotations under this category execute after the test methods. Additionally, it consists of the following annotations:

@AfterMethod

@AfterClass

@AfterTest

@AfterSuite

In TestNG, parameterization runs a test method multiple times with different values. Another name for this process is data-driven testing in TestNG. We can acquire Parameterization in TestNG in two ways:

* *Firstly, we can achieve it through the XML file.*
* *Secondly, we can achieve it through the dataproviders in TestNG.*
* public class Params
* {
* @Test
* @Parameters ({"val1", "val2"})
* public void Sum(int v1, int v2) {
* int finalsum = v1 + v2;
* System.out.println("The final sum of the given values is " + finalsum);
* }
* }

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">

<suite name="TestNG Parameters Suite">

<test name="Params">

<parameter name="val1" value="2" />

<parameter name="val2" value="3" />

<classes>

<class name="Params" />

</classes>

</test>

</suite>

**Ques.9. What are some common assertions provided by TestNG?**  
Ans. Some of the common assertions provided by testNG are-

1. assertEquals(String actual, String expected, String message) and other overloaded data types in parameter
2. assertNotEquals(double data1, double data2, String message) and other overloaded data types in parameter
3. assertFalse(boolean condition, String message)
4. assertTrue(boolean condition, String message)
5. assertNotNull(Object object)
6. fail(boolean condition, String message)
7. true(String message)

**Ques.10. How can we disable or prevent a test case from running?**  
Ans. By setting the “enabled” attribute as false, we can disable a test method from running.

**Ques.11. How can we make one test method dependent on others using TestNG?**  
Ans. Using dependsOnMethods parameter inside @Test annotation in TestNG we can make one test method run only after the successful execution of the dependent test method.

@Test(dependsOnMethods = { "preTests" })

**Ques.12. How can we set the priority of test cases in TestNG?**  
Ans. We can define the priority of test cases using the “priority” parameter in @Test annotation. The tests with lower priority value will get executed first. Example-

@Test(priority=1)

**Ques.14. How can we run a Test method multiple times in a loop(without using any data provider)?**  
Ans. Using invocationCount parameter and setting its value to an integer value, makes the test method to run n number of times in a loop.

**Ques.15. What is threadPoolSize? How can we use it?**  
Ans. The **threadPoolSize** attribute specifies the number of threads to be assigned to the test method. This is used in conjunction with **invocationCount** attribute. The number of threads will get divided with the number of iterations of the test method specified in the invocationCount attribute.

@Test(threadPoolSize = 5, invocationCount = 10)

public void threadPoolTest(){

//Test logic

}

**Ques.16. What is the difference between soft assertion and hard assertion in TestNG?**  
Ans. Soft assertions (SoftAssert) allows us to have multiple assertions within a test method, even when an assertion fails the test method continues with the remaining test execution. The result of all the assertions can be collated at the end using softAssert.assertAll() method.

@Test

**Ques.17. How to fail a testNG test if it doesn’t get executed within a specified time?**  
Ans. We can use the **timeOut** attribute of @Test annotation. The value assigned to this timeOut attribute will act as an upperbound, if test doesn’t get executed within this time frame then it will fail with timeOut exception.

**Ques.18. How can we skip a test case conditionally?**  
Ans. Using SkipException, we can conditionally skip a test case. On throwing the skipException, the test method is marked as skipped in the test execution report and any statement after throwing the exception will not get executed.

**Ques.19. How can we make sure a test method runs even if the test methods or groups on which it depends fail or get skipped?**  
Ans. Using “alwaysRun” attribute of @Test annotation, we can make sure the test method will run even if the test methods or groups on which it depends fail or get skipped.

@Test

public void parentTest() {

Assert.fail("Failed test");

}

@Test(dependsOnMethods={"parentTest"}, alwaysRun=true)

public void dependentTest() {

System.out.println("Running even if parent test failed");

}

Here, even though the parentTest failed, the dependentTest will not get skipped instead it will executed because of “alwaysRun=true”. In case, we remove the “alwaysRun=true” attribute from @Test then the report will show one failure and one skipped test, without trying to run the dependentTest method.

**Ques.20. How can we pass the parameter to test script using TestNG?**  
Ans. Using @Parameter annotation and ‘parameter’ tag in testng.xml we can pass parameters to test scripts.  
Sample testng.xml –

<suite name="sampleTestSuite">

<test name="sampleTest">

<parameter name="sampleParamName" value="sampleParamValue"/>

<classes>

<class name="TestFile" />

</classes>

</test>

</suite>

**Ques.21. How can we create data driven framework using TestNG?**  
Ans. Using @DataProvider we can create a data-driven framework in which data is passed to the associated test method and multiple iterations of the test run for the different test data values passed from the @DataProvider method. The method annotated with @DataProvider annotation return a 2D array of object.

//Data provider returning 2D array of 3\*2 matrix

@DataProvider(name = "dataProvider1")

public Object[][] dataProviderMethod1() {

return new Object[][] {{"kuldeep","rana"}, {"k1","r1"},{"k2","r2"}};

}

//This method is bound to the above data provider returning 2D array of 3\*2 matrix

//The test case will run 3 times with different set of values

@Test(dataProvider = "dataProvider1")

public void sampleTest(String s1, String s2) {

System.out.println(s1 + " " + s2);

}

**Ques.23. What is the use of @Factory annotation in TestNG?**  
Ans. @Factory annotation helps in the dynamic execution of test cases. Using @Factory annotation we can pass parameters to the whole test class at run time. The parameters passed can be used by one or more test methods of that class.  
Example – there are two classes TestClass and the TestFactory class. Because of the @Factory annotation, the test methods in class TestClass will run twice with the data “k1” and “k2”

public class TestClass{

private String str;

//Constructor

public TestClass(String str) {

this.str = str;

}

@Test

public void TestMethod() {

System.out.println(str);

}

}

public class TestFactory{

//The test methods in class TestClass will run twice with data "k1" and "k2"

@Factory

public Object[] factoryMethod() {

return new Object[] { new TestClass("K1"), new TestClass("k2") };

}

}

**Ques.24. What is the difference between @Factory and @DataProvider annotation?**  
Ans. @Factory method creates instances of test class and run all the test methods in that class with a different set of data.  
Whereas, @DataProvider is bound to individual test methods and run the specific methods multiple times