

1. What is Selenium and what are the different components and versions of Selenium?

Answer: Selenium is a free (open source) automated testing suite for web applications across different browsers and platforms. Selenium is not just a single tool but a suite of software's, each catering to different testing needs of an organization. It has four components.

- **Selenium Integrated Development Environment (IDE)**
- **Selenium Remote Control (RC)**
- **WebDriver**
- **Selenium Grid**

Selenium Core

The story starts in 2004 at ThoughtWorks in Chicago, with Jason Huggins building the Core mode as "JavaScriptTestRunner". Its JavaScript program that would automatically control the browser's actions.

"JavaScriptTestRunner" was later named as "Selenium Core" and released into the market as an Open Source tool. This Open Source tool started gaining demand in the market and people started using it for automating the repeated tasks in their Web Applications.

Selenium Remote Control

Unfortunately; testers using Selenium Core had to install the whole application under test and the web server on their own local computers because of the restrictions imposed by the same origin policy. To resolve this another ThoughtWork's engineer, Paul Hammant created system (in 2007) known as the Selenium Remote Control or Selenium 1.

Selenium 1 = Selenium IDE + Selenium RC + Selenium Grid

Selenium Grid

Patrick Lightbody to address the need of minimizing test execution times as much as possible, So he created Selenium Grid. Basically grid is for parallel execution and execute your test scripts on multiple environments.

Using "Selenium Grid", testers were able to distribute the tests across multiple machines and get them executed them on different machines over their network to reduce or minimize the time taken for overall execution of tests.

Selenium IDE

"Shinya Kasatani", who developed a Firefox extension named as "Selenium IDE".

"Selenium IDE" using its record and playback feature, records the automation tests like recording a video and executes the recorded tests like playing the recorded videos.

Selenium WebDriver

Earlier Selenium 1 used to be the major project of Selenium.

Selenium 1 = Selenium IDE + Selenium RC + Selenium Grid

Later Selenium Team has decided to merge both Selenium WebDriver and Selenium RC to form a more powerful Selenium tool.

They both got merged to form "Selenium 2"

"Selenium WebDriver" was the core of "Selenium 2" and "Selenium RC" used to run in maintenance mode.

Hence Selenium 2 = Selenium IDE + Selenium WebDriver 2.x + Selenium Grid.

"Selenium 2" released on July 8, 2011. Selenium team has decided to completely remove the dependency for Selenium RC. After 5 years, "Selenium 3" was released on October 13, 2016 with a major change, which is the original Selenium Core implementation and replacing it with one backed by WebDriver and lot more improvements.

Hence Selenium 3 = Selenium IDE + Selenium WebDriver 3.x + Selenium Grid.

After 3 years from it's a major release, now Selenium has put out its first alpha version of Selenium 4 on Apr 24, 2019. Still, there is no official announcement about the release date of Selenium 4, but we are expecting it around October 2019. Till that there can be several alpha or beta versions released time to time with stabilization.

2. What is the latest Selenium tool?

Answer: Selenium WebDriver is the successor to Selenium RC which sends commands directly to the browser and retrieves results. Selenium Grid is a tool used to run parallel tests across different machines and different browsers simultaneously which results in minimized execution time.

3. What is the difference between Manual and Automation Testing?

Answer: Differences are given below:

Automation Testing	Manual Testing
Automated testing is more reliable. It performs same operation each time. It eliminates the risk of human errors.	Manual testing is less reliable. Due to human error, manual testing is not accurate all the time.
Automation Testing uses automation tools to execute test cases.	In manual testing, test cases are executed by a human tester and software.
Initial investment of automation testing is higher. Investment is required for testing tools. In the long run it is less expensive than manual. ROI is higher in the long run compared to Manual testing.	Initial investment of manual testing is less than automation. Investment is required for human resources. ROI is lower in the long run compared to Automation testing.
Automation testing is a practical option when we do regressions testing.	Manual testing is a practical option where the test cases are not run repeatedly and only needs to run once or twice.
Automation does not allow random testing	Exploratory testing is possible in Manual Testing
Automated testing is a reliable method, as it is performed by tools and scripts. There is no testing Fatigue. Manual testing is not as accurate because of the possibility of the human errors.	Manual testing is not as accurate because of the possibility of the human errors.
Automated testing does not involve human consideration. So it can never give assurance of user-friendliness and positive customer experience.	The manual testing method allows human observation, which may be useful to offer user-friendly system.
Performance Tests like Load Testing, Stress Testing, Spike Testing, etc. have to be tested by an automation tool compulsorily.	Performance Testing is not feasible manually
This testing can be executed on different operating platforms in parallel and reduce test execution time.	Manual tests can be executed in parallel but would need to increase your human resource which is expensive
Automation testing uses frameworks like Data Drive, Keyword, Hybrid to accelerate the automation process.	Manual Testing does not use frameworks but may use guidelines, checklists, stringent processes to draft certain test cases.

4. What are the benefits of Automation Testing?

Answer: Following are benefits of automated testing:

- 70% faster than the manual testing.
- Wider test coverage of application features.
- Reliable in results.
- Ensure Consistency.

- Saves Time and Cost.
- Improves accuracy.
- Human Intervention is not required while execution.
- Increases Efficiency.

5. What are the popular test automation tools for functional testing?

Answer: Please follow below link from “softwaretestinghelp.com” . It is very good article on different testing tools.

<https://www.softwaretestinghelp.com/tools/top-30-functional-testing-tools/>

6. What is the main purpose of Automation Testing?

Answer: Automated software testing can increase the depth and scope of tests to help improve software quality. Lengthy tests that are often avoided during manual testing can be run unattended. They can even be run on multiple computers with different configurations. Test Automation demands considerable investments of money and resources.

Successive development cycles will require execution of same test suite repeatedly. Using a test automation tool, it's possible to record this test suite and re-play it as required. Once the test suite is automated, no human intervention is required. This improved ROI of Test Automation. The goal of Automation is to reduce the number of test cases to be run manually and not to eliminate Manual Testing altogether.

7. What is the goal of Automation Testing?

Answer: The goal of Automation is to reduce the number of test cases to be run manually and not to eliminate Manual Testing altogether.

8. Why Selenium should be selected as a Test tool?

Answer: Benefits of selecting Selenium:

It is completely open source - So you can easily download it for FREE. Selenium supports different programming languages such as Java, Python, C#, Ruby, Groovy, Javascript etc - So you can easily use Selenium. Selenium works in multiple operating systems - Hence no worries about using it.

9. What is Same Origin Policy and how it can be handled? How to overcome same origin policy through web driver?

Answer: Same Origin policy prohibits JavaScript code from accessing elements from a domain that is different from where it was launched. Example, the HTML code in www.google.com uses a JavaScript program "testScript.js". The same origin policy will only allow testScript.js to access pages within google.com such as google.com/mail, google.com/login, or google.com/signup. However, it cannot access pages from different sites such as yahoo.com/search or fbk.com because they belong to different domains.

10. What are the testing types that can be supported by Selenium?

Answer: Using Selenium type of testing can be done are:

- Functional Testing.
- Regression Testing.
- Sanity Testing.
- Smoke Testing.
- Responsive Testing.

- Cross Browser Testing.
- UI testing (black box)
- Integration Testing.

11. What are the limitations of Selenium?

Answer: Challenges and limitations of Selenium WebDriver.

- We cannot test windows application.
- We cannot test mobile apps.
- Limited reporting.
- Handling dynamic Elements.
- Handling page load.
- Handling pop up windows.
- Handling captcha.

12. Which Test cases needs to be automated?

Answer: test cases can (and should be automated) if:

- Tests are used repeatedly.
- Tests involve a lot of data entry.
- Tests clearly pass or fail.
- Tests deliver an exact result.
- Tests use consistent UI and regular controls.
- Tests are only to do what they're told — not check anything else.

13. What is Selenese?

Answer: Selenese is the set of selenium commands which are used to test your web application. Tester can test the broken links, existence of some object on the UI, Ajax functionality, Alerts, window, list options and lot more using selenese. Selenium command tells selenium automation engine to perform certain tasks.

14. Which automation tools could be used for post-release validation with continuous integration?

Answer:

1) **Experitest** - Experitest is the continuous testing platform for mobile and web apps. It integrates seamlessly with your development, testing, and continuous integration ecosystem, and is fully compatible with Appium, Selenium, Jenkins and other tools.

2) **Selenium**

Selenium is open-source software testing tool. It supports all the leading browsers like Firefox, Chrome, IE, and Safari. Selenium WebDriver is used to automate web application testing.

3) **QuerySurge**

QuerySurge is the smart data testing solution that is the first-of-its-kind full DevOps solution for continuous data testing.

Key features include detailed data intelligence & data analytics, seamless integration into the DevOps pipeline for continuous testing.

4) **Jenkins**

Jenkins is a Continuous Integration tool which is written using Java language. This tool can be configured via GUI interface or console commands.

5) **Travis**

Travis is continuous testing tool hosted on the GitHub. It offers hosted and on-premises variants. It provides a variety of different languages and a good documentation.

15. Does the latest version of Selenium WebDriver support Mobile Testing?

Answer: No. Selenium cannot automate Mobile Applications. you should use Appium for mobile automation.

16. Explain your project, roles & Experience summary?

Answer: Following are the responsibilities according to the experience level:

1 to 2 Years

=====

- Understanding Test Requirements and analyzing the Application under Test(AUT).
- Generating Test Cases (Test Scripts) using Selenium Element locators and WebDriver API Commands.
- Enhancing Test cases using Java Programming.
- Debugging Test Cases and Fixing Errors.
- Executing/Running Test Cases
- Defect Reporting & Tracking
- Test Reporting

2 to 4 years of Experience in Test Automation using Selenium

=====

- Understanding and Analyzing the Application Under Test in terms of Object Identification.
- Creating Test scenarios and Collecting Test Data.
- Creating Test Automation Resources (Function Libraries etc...).
- Implementing JUnit or TestNG Test Automation framework and developing automation infrastructure.
- Creating and enhancing Test Cases (Test Scripts) using Element locators, WebDriver methods, Java programming concepts and TestNG Annotations.
- Creating reusable components.
- Handling duplicate objects and dynamic objects using index property and Regular expressions.
- Collecting Test Data for Data Driven Testing.
- Creating Test Cases (Test Scripts) using Selenium Webdriver, Java and TestNG Annotations.
- Parameterization, Synchronization and define Test results.
- Error Handling, Adding comments.
- Creating Data driven Tests and Running through framework.
- Debugging and Running Tests
- Analyzing Test Results
- Defect Reporting and Tracking using any Defect Management Tool.
- Test Reporting
- Modifying Tests and performing Re & Regression Testing.
- Tracking Defects and Select Test cases for Re & Regression Testing.
- Modifying Test Automation Resources and Maintenance of Resources.

4+ Years

=====

- Selecting or Identifying Areas/Test cases for Automation.
- Designing & Implementing Test Automation Strategy.

- Creating Automation Test Plan and getting approvals.
- Choose selenium tools and Configuring Selenium Test Environment (Ex: Eclipse IDE, Java, Selenium WebDriver and TestNG etc...).
- Involvement in Selenium Environment Setup...
- Automation Framework Design and Implementation.
- Creating, Organizing, and managing Test Automation Resources.
- Creating, Enhancing, debugging and Running Test Cases.
- Organizing, monitoring defect management process.
- Handling changes and conducting Regression Testing.
- Finding solutions for Object Identification issues and error handling issues.
- Co-coordinating Test team members and Development team in order to resolve the issues.
- Interacting with client side people to solve issues and update status.

17. What are your roles and responsibilities as part of automation in your project?

Answer : Please refer above answer

18. What are the major challenges in Functional Test Automation?

Answer:

1. Scripting expertise—a high bar for testing talent
2. When you can't find a trace of tractability
3. Quickly scaling test environments is a challenge
4. Too many UI tests can break testing
5. A lack of transparency can inhibit automated software testing

19. What are the difficulties you have faced in Object Identification?

Answer:

1. Handling windows based Pop-Up Windows
2. Identifying Dynamic Elements

Many web apps or websites often have web elements that are dynamic in nature, which are not visible when you visit the site for the first time. This means that the web pages are user-specific and display different data for different users based on their requirements; new data appears on the web page after a certain period of time or when a user clicks something on the page. For example, if the ID of an element is changing on every page load, then it's not easy to handle this situation in a normal way.

3. Timeout or Sync Issue

Whether you call it a timeout or sync issue, it is one of the most common challenges in Selenium test automation. If you don't handle this issue carefully, most of your testing script might fail. It is even proved many times that around 80% of scripts fail due to improper sync while executing automation testing.

4. Page Loading

As mentioned earlier, some of the web pages in a web app are user-specific and load different elements depending on the user. Some features even appear based on the user's previous activity. For example, if you have a drop-down menu for Italian food, then food items related to that category will appear in the food dropdown. During the runtime, the Selenium script might not be able to identify the element. Therefore, to overcome this issue, you can use explicit waits to provide elements enough time to load and to discover the element.

20. How you organized your Test Automation resources in your Project ?

Answer:

1. Plan Your Test Cases & Test Suites
2. Centralize Your Test Assets
3. Differentiate Test Objects
4. Validate & Remove Outdated Test Cases
5. Separate Test Architecture

21. Did you use any build management tools in your project?

Answer: Yes

Build tools are programs that automate the creation of executable applications from source code. Building incorporates compiling, linking and packaging the code into a usable or executable form. ... Using an automation tool allows the build process to be more consistent. Some of most popular tools in this category "Java Build Tools" are Gradle, Apache Maven, Sonatype Nexus, Apache Ant, and Pants are the.

22. How you handled errors in your Test Scripts?

Answer: Please refer below link

<https://www.automationtestinginsider.com/2019/10/selenium-questions-part5-selenium.html>

Common Exceptions in Selenium

1. **ElementNotVisibleException** - This type of Selenium exception occurs when an existing element in DOM has a feature set as hidden.
2. **ElementNotSelectableException** - This Selenium exception occurs when an element is presented in the DOM, but you can be able to select. Therefore, it is not possible to interact.
3. **NoSuchElementException** - This Exception occurs if an element could not be found.
4. **NoSuchFrameException** - This Exception occurs if the frame target to be switched to does not exist.
5. **NoAlertPresentException** - This Exception occurs when you switch to no presented alert.
6. **NoSuchWindowException** - This Exception occurs if the window target to be switch does not exist.
7. **StaleElementReferenceException** - This Selenium exception occurs when the web element is detached from the current DOM.
8. **SessionNotFoundException** - The WebDriver is acting after you quit the browser.
9. **TimeoutException** - Thrown when there is not enough time for a command to be completed. For Example, the element searched wasn't found in the specified time.
10. **WebDriverException** - This Exception takes place when the WebDriver is acting right after you close the browser.
11. **ConnectionClosedException** - This type of Exception takes place when there is a disconnection in the driver.
12. **ElementNotInteractableException** - This Selenium exception is thrown when any element is presented in the DOM. However, it is impossible to interact with such an element.
13. **InvalidArgumentException** - It occurs when an argument does not belong to the expected type.
14. **SessionNotCreatedException** - It happens when a new session could not be successfully created.
15. **ElementNotVisibleException** - If selenium tries to find an element but the element is not visible within the page
16. **NoSuchAttributeException** - While trying to get attribute value but the attribute is not available in DOM.

23. Did you create any reusable components while automation any of your projects?

Answer: Yes.

As an automation test engineer, you might have noticed that some of our test steps get repeated very often in multiple tests. In such cases, designing the tests in a such a way that it could be reused in multiple workflow is very important. Re-usability allows us to be more efficient & to write better and clean code.

Lets see how we could design reusable tests.

Lets consider a sample application which has below workflows.

- An user can register himself in the application by ordering a product.
- An existing registered user can also order a new product.
- Once a product is ordered, it can be viewed.
- User can logout.

Based on the above requirements, we can come up with different workflows as shown in below.

- A new user enters details -> orders a product -> logout
- A new user enters details -> orders a product -> views product -> logout
- An existing user -> orders a product -> views product -> logout
- An existing user -> orders a product -> logout
- An existing user -> views product -> logout

We might be interested in testing all these above workflows. The same workflow should also be tested for different input parameters. For ex: Different types of products, different payment methods like Credit card, promocodes etc. So writing reusable tests is very important to maintain these tests in future to avoid duplication in your code.

24. Did you find any test scenarios that cannot be automated in your projects using Selenium?

Answer: Below are some scenarios I think are not beneficial to automate using Selenium.

CAPTCHA scenarios: Well, CAPTCHAS are there for purpose. To bypass automation. Best way to handle it is to tell your dev team to disable it or make it static.

Video streaming scenarios: More often that not, Selenium won't be able to recognise video controls. JavaScript Executor and flex-ui-selenium will work to some extent, but they are not entirely reliable.

Code reading scenarios: If your web app has a functionality which reads barcodes or QR codes, it's not beneficial to automate it. There may be some tools available for them but I'm not sure how effective they are.

Crash recovery scenarios: You might want to test your application's crash recovery. This is a scenario best tested manually. I am not saying you won't be able to test it using Selenium. You may be. But I don't know how feasible and beneficial it would be.

Performance testing: It can be automated but it's best not to automate performance testing using Selenium.

25. Explain Automation Life Cycle (ATLC) ?

Answer:

1- Automation feasibility analysis: The main objective of this phase will be to check feasibility of automation.

So your main focus will be on below points.

- Which test case can be automated and how we can automate them?
- Which module of your application can be tested and which can not be automated
- Which tools we can use for our application (like Selenium,QTP,Sahi,OATS, Telrik etc) and which tools will be best of our application
- Take following factors into consideration like Team size, Effort and cost involved for tools which we will use.

2- Test Plan/Test Design:

This phase plays very important role in Automation test life cycle. In this phase you have to create a Test plan by considering below point into considerations.

- Fetch all the manual test case from test management tool that which TC has to automate.
- Which framework to use and what will be advantage and disadvantage of the framework which we will use.
- Create a test suite for Automation test case in Test Management tool.
- In test plan you can mention background, limitation, risk and dependency between application and tools.
- Approval from client/ Stack holders.

3- Environment Setup: By name itself you can understand that we need to setup machine or remote machine where our test case will execute.

- In this section you can mention how many machine you want.
- What should be the configuration in terms of hardware and software.

4-Test Script development/ Automation testcase development:

In this phase you have to start develop automation script and make sure all test script is running fine and should be stable enough.

- Start creating test script based on your requirement
- Create some common method or function that you can reuse throughout your script
- Make your script easy, reusable, well structured and well documented so if third person check your script then he/she can understand your scripts easily.
- Use better reporting so in case of failing you can trace your code
- Finally review your script and your script should be ready before consumption.

5-Test script execution: In this phase you have to execute all your test script.

Some points to remember while execution

- Your script should cover all the functional requirement as per test case.
- Your script should be stable so it should run in multiple environment and multiple browsers (depends on your requirement)
- You can do batch execution also if possible so it will save time and effort.
- In case of failure your script should take screen shots.
- If test case is failing due to functionality, you have to raise a bug/defect

6- Generate test result / Analyses of result:

This is the last phase of Automation test life cycle in which we will gather test result and will share with team/client/stack holders.

- Analyze the output and calculate how much time it take to complete the testcase.
- You should have good report generation like Extent Report ,XSLT report, TestNG report, ReporterNG etc.

26. Does manual bring more ROI or automation brings more ROI?

Answer: The goal of automated testing is to improve software quality while testing faster and reducing costs, and there is more to the ROI of automation than accounting for manual

and regression tests. ... Without proper parallel testing and the coverage it can provide, you risk encountering defects further downstream.

27. Why did you choose Selenium in your project, when there are so many tools?

Answer: Selenium gained popularity because of one single reason: it is free while other testing tools such as HP QTP are insanely priced. Although QTP is better in terms of easier to learn, better support, and have cool features such as Object Repository, their pricing is unjustifiable.

In an organisation, there would be many projects running and buying licenses for paid automation tool for each projects would burn a very big hole in the company's wallet. Second, Selenium is cross platform. You can execute Selenium scripts in Linux and Mac OS where as other tools are tied to Windows Platform. These two are the primary reasons why Selenium is a popular test automation tool.

28. Criteria for selecting test cases for automation

Answer: Following are the criteria:

1. Criteria for selecting test cases for automation
2. Tests that use multiple data values for the same actions (data driven tests)
3. Complex and time consuming tests
4. Tests involving many simple, repetitive steps
5. Testing needed on multiple combinations of OS, DBMS & Browsers
6. Test Cases that are very tedious or difficult to perform manually

29. Which type of test cases exclude for automation

Answer: Test Cases that are newly designed and not executed manually at-least once
Test Cases for which the requirements are changing frequently
Test cases which are executed on ad-hoc basis.

30. Main stages in automation testing life cycle?

Answer: Below are the main states in ATLC

- Determining The Scope Of Test Automation
- Selecting The Right Tool For Automation
- Test Plan + Test Design + Test Strategy
- Setting Up The Test Environment
- Automation Test Script Development + Execution
- Analysis + Generation Of Test Reports

31. What are the main task during planning phase of automation testing?

Answer: During the test planning phase, the testing team decides the test procedure creation standards and guidelines; hardware; software and network to support test environment; a preliminary test schedule; test data requirements; defect tracking procedure and associated tracking tool and a procedure to control test.

32. Principal features of good automation tool.

Answer: A good automation tool should have the following characteristics.

- Quick and easy test environment setup.
- Cross-platform support.
- Good debugging/logging support.
- Robust object identification.
- Object and image testing abilities.
- Cross browser testing support.
- Database integration and validation.

33. Different approach for designing automation solution

Answer: These approaches include:

- Keyword Driven Testing
- Page Object Model
- Behavior Driven Development

34. How to measure success of automation testing?

Answer: As you look to adopt an automated testing process to meet the rising demand for faster delivery cycles and bug-free releases, it's vital to assess whether the return on investment (ROI) is worth the change. Before executing, or even thinking of building out an automation strategy, you'll want to calculate the net gain you'll see from transitioning. Divide this by the net investment needed to transition (i.e., the tools and resources you use), and you'll get your ROI for automated testing.

This equation will look like the following: $(\text{Gain} - \text{Investment}) / \text{Investment}$

The key question that arises is, "what defines the gains and the investments?" Calculate the 6 measurements outlined in this white paper to estimate the long and short term monetary value you will receive from investing in automation. Before we dive into the benefits of transitioning and the investments in tools and resources you'll have to make, let's dive into the common pitfalls in calculating the ROI.

1. Only accounting for creating, developing, and maintaining automated tests versus manual tests. Manual testing will always be important. While automation is on your mind, there are scenarios that will always require manually executed test cases.
2. Not accounting for the percentage of tests that need to stay manual. Redundant or repetitive test steps are great candidates for automation, as having to run multiple of the same test type can be tedious and ultimately prone to human error.
3. Not syncing your automation tool stack with organizational capabilities. To implement an automation strategy, you'll need to have both automation knowledge and product knowledge.
4. Not accounting for ROI over a period of time. When building out a business case to transition to automation, you'll not only want to gauge the short-term benefits of investing, but also how it will impact your team and organization in the long run.

35. If sprint is of 2 weeks, what about automation cycle?

Answer: Answer will be provided soon

36. How do you estimate and how to track your automation test cases ?

Answer: Answer will be provided soon

37. How to calculate ROI?

Answer: Please refer answer from question 34

38. How to calculate automation efforts?

Answer: Please refer below link

https://www.stepinforum.org/Test_Automation/TA_finaltalk/PDFs/Papers/TEST%20AUTOMATION%20EFFORT%20ESTIMATION.pdf

39. How to optimize the execution time?

Answer: Answer will be provided soon

1. What are the different types of Exceptions in Selenium?

Answer: During automation in Selenium WebDriver, we come across various exceptions & we need to deal with them. Below is the list of various exceptions occur in selenium webdriver.

There are main three types of Exceptions in Selenium WebDriver –

Checked Exceptions – These Exceptions can be handled during compile time. If they are not handled, it gives compile time error. Example- FileNotFoundException, IOException etc.

Unchecked Exceptions – These exceptions can not be handled during compile time & they got caught at run time. Example – ArrayIndexOutOfBoundsException.

Error – Errors which can not be handled by using even try catch block. Example -Assertion Error.

Common Exceptions in Selenium

1. ElementNotVisibleException - This type of Selenium exception occurs when an existing element in DOM has a feature set as hidden.
2. ElementNotSelectableException - This Selenium exception occurs when an element is presented in the DOM, but you can be able to select. Therefore, it is not possible to interact.
3. NoSuchElementException - This Exception occurs if an element could not be found in DOM.
4. NoSuchFrameException - This Exception occurs if the frame target to be switched to does not exist.
5. NoAlertPresentException - This Exception occurs when you switch to no presented alert.
6. NoSuchWindowException - This Exception occurs if the window target to be switch does not exist.
7. StaleElementReferenceException - This Selenium exception occurs when the web element is detached from the current DOM.
8. SessionNotFoundException - The WebDriver is acting after you quit the browser.
9. TimeoutException - Thrown when there is not enough time for a command to be completed. For Example, the element searched wasn't found in the specified time.
10. WebDriverException - This Exception takes place when the WebDriver is acting right after you close the browser.
11. ConnectionClosedException - This type of Exception takes place when there is a disconnection in the driver.
12. ElementNotInteractableException - This Selenium exception is thrown when any element is presented in the DOM. However, it is impossible to interact with such an element.
13. InvalidArgumentException - It occurs when an argument does not belong to the expected type.
14. SessionNotCreatedException - It happens when a new session could not be successfully created.

15. `ElementNotVisibleException` - If selenium tries to find an element but the element is not visible within the page

16. `NoSuchAttributeException` - While trying to get attribute value but the attribute is not available in DOM.

Read About Java Exceptions

here: <https://www.automationtestinginsider.com/2019/08/exception-handling-in-java.html>

2. How to handle Selenium WebDriver Exceptions?

Answer: Handling Exceptions In Selenium WebDriver

Following are a few standard ways using which one can handle Exceptions in Selenium WebDriver:

Try-catch: This method can catch Exceptions by using a combination of the try and catch keywords. Try indicates the start of the block, and Catch is placed at the end of the try block to handle or resolve the Exception. The code that is written within the Try/Catch block is referred to as “protected code.” The following code represents the syntax of Try/Catch block

```
-
try
{
    // Some code
}
catch(Exception e)
{
    // Code for Handling the exception
}
```

Multiple catch blocks: There are various types of Exceptions, and one can expect more than one exception from a single block of code. Multiple catch blocks are used to handle every kind of Exception separately with a separate block of code. One can use more than two catch blocks, and there is no limitation on the number of catch blocks. The code below represents the syntax of multiple catch blocks –

```
try
{
    //Some code
}
catch(ExceptionType1 e1)
{
    //Code for Handling the Exception 1
}
catch(ExceptionType2 e2)
{
    //Code for Handling the Exception 2
}
```

Throw/Throws: When a programmer wants to generate an Exception explicitly, the Throw keyword is used to throw Exception to runtime to handle it. When a programmer is throwing an Exception without handling it, then he/she needs to use Throws keyword in the method signature to enable the caller program to understand the exceptions that might be thrown by the method. The syntax for Throws is as follows:

```
public static void anyFunction() throws Exception
{
try
{
    // write your code here
}
catch (Exception e)
{

```

```

    // Do whatever you wish to do here

    // Now throw the exception back to the system
    throw(e);
}
}

```

Multiple Exceptions: One can mention various Exceptions in the throws clause. Refer to the example below:

```

public static void anyFunction() throws ExceptionType1, ExceptionType2
{
    try
    {
        // write your code here
    }
    catch (ExceptionType1 e1)
    {
        // Code to handle exception 1
    }
    catch (ExceptionType1 e2)
    {
        // Code to handle exception 2
    }
}

```

Finally: The Final keyword is used to create a block of code under the try block. This final code block always executes irrespective of the occurrence of an exception

```

try
{
    //Protected code
}
catch(ExceptionType1 e1)
{
    //Catch block
}
catch(ExceptionType2 e2)
{
    //Catch block
}
catch(ExceptionType3 e3)
{
    //Catch block
}

finally
{
    //The finally block always executes.
}

```

One can also use the following methods to display Exception Information:

printStackTrace(): It prints the stack trace, name of the exception, and other useful description.

toString(): It returns a text message describing the exception name and description.

getMessage(): It displays the description of the exception

3. What are the different types of exceptions you have faced in Selenium WebDriver?

Answer: Please refer question#1

4. What are the types of exceptions which will appear while finding elements?

Answer:

1. `ElementNotVisibleException` - This type of Selenium exception occurs when an existing element in DOM has a feature set as hidden.
2. `ElementNotSelectableException` - This Selenium exception occurs when an element is presented in the DOM, but you can be able to select. Therefore, it is not possible to interact.
3. `NoSuchElementException` - This Exception occurs if an element could not be found in DOM.
4. `StaleElementReferenceException` - This Selenium exception occurs when the web element is detached from the current DOM.
5. `ElementNotInteractableException` - This Selenium exception is thrown when any element is presented in the DOM. However, it is impossible to interact with such an element.
6. `ElementNotVisibleException` - If selenium tries to find an element but the element is not visible within the page
7. `NoSuchAttributeException` - While trying to get attribute value but the attribute is not available in DOM.

5. What is a null pointer exception ? Is it checked or not?

Answer: `NullPointerException` is a unchecked exception(Run time).`NullPointerException` (usually) occurs because there is some bug in your code. If you expect a `NullPointerException` to be thrown, the correct solution is to fix the bug rather than to handle the exception.
`NullPointerException` doesn't force us to use catch block to handle it.

6. What is StaleElementException? When does it occur? How do you handle it?

Answer: Stale means old, decayed, no longer fresh. Stale Element means an old element or no longer available element. Assume there is an element that is found on a web page referenced as a `WebElement` in `WebDriver`. If the DOM changes then the `WebElement` goes stale. If we try to interact with an element which is staled then the `StaleElementReferenceException` is thrown.
The two reasons for Stale element reference are

The element has been deleted entirely.
The element is no longer attached to the DOM.

Solution 1:

The Most way to handle this is to refresh the page, On refreshing it, most of the time driver found the element, But it's not the perfect solution.
`Driver.navigate().refresh();`
`Driver.findElement(By.name("name")).click();`
`Driver.navigate().refresh();`
`Driver.findElement(By.name("name")).click();`

Solution 2:

If an element is not attached to DOM then you could try using 'try-catch block' within 'for loop'

// Using for loop, it tries for 3 times.

// If the element is located for the first time then it breaks from the for loop and comes out of the loop

```
for(int i=0; i<=2;i++){
    try{
        driver.findElement(By.xpath("xpath")).click();
        break;
    }
    catch(Exception e){
        Sysout(e.getMessage());
    }
}
```

Solution 3:

Wait for the element till it gets available

```
wait.until(ExpectedConditions.presenceOfElementLocated(By.id("id")));
```

Solution 4:

We can handle Stale Element Reference Exception by using POM.

We could avoid StaleElementException using POM. In POM, we use `initElements()` method which loads the element but it won't initialize elements. `initElements()` takes latest address. It initializes during run time when we try to perform any action on an element. This process is also known as Lazy Initialization.

7. NoSuchElementException and ElementNotVisibleException what is difference?

Answer:

1. NoSuchElementException - This Exception occurs if an element could not be found in DOM.
2. ElementNotVisibleException - This type of Selenium exception occurs when an existing element in DOM has a feature set as hidden.

8. Why we are getting class not found exception?

Answer: The `ClassNotFoundException` is thrown when the Java Virtual Machine (JVM) tries to load a particular class and the specified class cannot be found in the classpath. The `ClassNotFoundException` is a checked exception and thus, must be declared in a method or constructor's throws clause.

The following example tries to load a class using the `forName` method. However, the specified class name cannot be found and thus, a `ClassNotFoundException` is thrown.

How to deal with the ClassNotFoundException

Verify that the name of the requested class is correct and that the appropriate .jar file exists in your classpath. If not, you must explicitly add it to your application's classpath.

In case the specified .jar file exists in your classpath then, your application's classpath is getting overridden and you must find the exact classpath used by your application.

In case the exception is caused by a third party class, you must identify the class that throws the exception and then, add the missing .jar files in your classpath.

9. Diff between classnotfound exception and nodeffoundexception?

Answer: ClassNotFoundException is an exception that occurs when you try to load a class at run time using Class.forName() or loadClass() methods and mentioned classes are not found in the classpath. NoClassDefFoundError is an error that occurs when a particular class is present at compile time, but was missing at run time.

ClassNotFoundException is raised in below program as class "ATI" is not found in classpath.
// ClassNotFoundException

Output

java.lang.ClassNotFoundException: ATI

Below program will be successfully compiled and generate two classes Example1.class and Test.class

Now remove Example1.class file and run Test.class.

At Java runtime NoClassDefFoundError will be thrown.

// NoClassDefFoundError

1. What do we mean by Selenium 1, Selenium 2 and Selenium 3?

Answer: Earlier Selenium 1 used to be the major project of Selenium.

Selenium 1 = Selenium IDE + Selenium RC + Selenium Grid

Later Selenium Team has decided to merge both Selenium WebDriver and Selenium RC to form a more powerful Selenium tool.

They both got merged to form "Selenium 2"

"Selenium WebDriver" was the core of "Selenium 2" and "Selenium RC" used to run in maintenance mode.

Hence Selenium 2 = Selenium IDE + Selenium WebDriver 2.x + Selenium Grid.

"Selenium 2" released on July 8, 2011.

Selenium team has decided to completely remove the dependency for Selenium RC.

After 5 years, "Selenium 3" was released on October 13, 2016 with a major change, which is the original Selenium Core implementation and replacing it with one backed by WebDriver and lot more improvements.

Hence Selenium 3 = Selenium IDE + Selenium WebDriver 3.x + Selenium Grid.

After 3 years from it's a major release, now Selenium has put out its first alpha version of Selenium 4 on Apr 24, 2019. Still, there is no official announcement about the release date of Selenium 4, but we are expecting it around October 2019. Till that there can be several alpha or beta versions released time to time with stabilization.

2. When should I use Selenium Grid?

Answer: Selenium Grid is a part of the Selenium Suite that specializes in running multiple tests across different browsers, operating systems, and machines in parallel.

You should use Selenium Grid when you want to do either one or both of following:

1. For Parallel Testing - Run your tests against different browsers, operating systems, and machines all at the same time. This will ensure that the application you are Testing is fully compatible with a wide range of browser-O.S combinations.

2. Save time in the execution of your test suites. If you set up Selenium Grid to run, say, 4 tests at a time, then you would be able to finish the whole suite around 4 times faster.

3. What are the different types of drivers available in WebDriver?

Answer: Drivers are required for different types of browsers.

Here is the list of different types of drivers available in Selenium WebDriver.

- FirefoxDriver
- InternetExplorerDriver
- ChromeDriver

- SafariDriver
- OperaDriver
- AndroidDriver
- IPHONEDriver
- HTMLUnitDriver

4. Is WebDriver a class or interface?

Answer: WebDriver is an Interface, and we are defining a reference variable (driver) whose type is an interface. Now any object we assign to it must be an instance of a class (FirefoxDriver) that implements the interface. Whenever we use Selenium for automation, the first line of our Program starts with the code to invoke Fire Fox Driver
 WebDriver driver = new FirefoxDriver();

5. What is the super interface of WebDriver ?

Answer: SearchContext is the super most interface in selenium, which is extended by another interface called WebDriver. All the abstract methods of SearchContext and WebDriver interfaces are implemented in RemoteWebDriver class.

6. Is FirefoxDriver a class or interface?

Answer: FirefoxDriver is a class that has been written specifically for the Firefox browser. It has methods that are implemented and it can be instantiated. It can perform all functions (or methods) on the Firefox browser as defined in the interface WebDriver.

7. Why do we create a reference variable 'driver' of type WebDriver and what is the purpose of its creation?

Answer: Having a reference variable of type WebDriver allows us to assign the driver object to different browser specific drivers. Thus allowing multi-browser testing by assigning the driver object to any of the desired browser.

8. Explain the line of code WebDriver driver = new FirefoxDriver();?

Answer: 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. As per the above statement, we are creating an instance of the WebDriver interface and casting it to FirefoxDriver Class. All other Browser Drivers like ChromeDriver, InternetExplorerDriver, PhantomJSdriver, SafariDriver etc implemented the WebDriver interface (actually the RemoteWebDriver class implements WebDriver Interface and the Browser Drivers extends RemoteWebDriver). Based on this statement, you can assign Firefox driver and run the script in Firefox browser (any browser depends on your choice).

9. What are the different types of navigation commands in WebDriver?

Answer: Browser Navigation Commands:
 WebDriver provides some basic Browser Navigation Commands that allows the browser to move backwards or forwards in the browser's history.

1. Navigate To:

Method - String to(String argo)

In WebDriver, this method loads a new web page in the existing browser window. It accepts String as parameter and returns void. The respective command to load/navigate a new web page can be written as:

Example:

```
driver.navigate().to("www.automationtestinginsider.com");
```

2. Backward:

Method - void back()

This method enables the web browser to click on the back button in the existing browser window. It neither accepts anything nor returns anything. The respective command that takes you back by one page on the browser's history can be written as:

Example:

```
driver.navigate().back();
```

3. Forward:

Method - void forward()

This method enables the web browser to click on the forward button in the existing browser window. It neither accepts anything nor returns anything. The respective command that takes you forward by one page on the browser's history can be written as:

Example:

```
driver.navigate().forward();
```

4. Refresh:

Method - void refresh()

In WebDriver, this method refresh/reloads the current web page in the existing browser window. It neither accepts anything nor returns anything. The respective command that takes you back by one page on the browser's history can be written as:

Example:

```
driver.navigate().refresh();
```

Complete article: <https://www.automationtestinginsider.com/2019/10/webdriver-commands.html>

10. What are the different types of waits available in WebDriver?

Answer: Please refer complete article here

<https://www.automationtestinginsider.com/2020/02/waits-in-selenium-webdriver.html>

11. What is the difference between driver.close() and driver.quit() commands?

Answer: driver. quit() is used to exit the browser, end the session, tabs, pop-ups etc. But the when you driver. close(), only the window that has focus is closed.

12. Can Selenium Automate Desktop Applications?

Answer: Selenium does not have the capability to automate the desktop applications. It cannot recognize the objects in a desktop application. Selenium drives the testing using the driver object that identifies the elements on screen using id, cssselector, xpath etc. which are not present in a desktop app.

13. What is the difference between Assert and Verify commands?

Answer: In case of the “Assert” command, as soon as the validation fails the execution of that particular test method is stopped and the test method is marked as failed. Whereas, in case of “Verify”, the test method continues execution even after the failure of an assertion statement.

Assert

We use Assert when we have to validate critical functionality, failing of which makes the execution of further statements irrelevant. Hence, the test method is aborted as soon as failure occurs.

Verify

At times, we might require the test method to continue execution even after the failure of the assertion statements. In TestNG, Verify is implemented using SoftAssert class. In case of SoftAssert, all the statements in the test method are executed (including multiple assertions).

14. Can Captcha be automated using Selenium?

Answer: No. CAPTCHA can be automated if you are able to decode the image using OCR (Optical Character Recognition). For that, one will need to write complex algorithm to sort out the image pattern and has to be an expert in image pattern mapping as well. But images with the passage of time have become progressively more unreadable, therefore reducing the very chances of CAPTCHA automation.

15. What is Object Repository and how can we create an Object Repository in Selenium?

Answer: An object repository is a common storage location for all objects. In Selenium WebDriver context, objects would typically be the locators used to uniquely identify web elements.

The major advantage of using object repository is the segregation of objects from test cases. If the locator value of one webelement changes, only the object repository needs to be changed rather than making changes in all test cases in which the locator has been used. Maintaining an object repository increases the modularity of framework implementation. An object repository is a common storage location for all objects. In Selenium WebDriver context, objects would typically be the locators used to uniquely identify web elements. The major advantage of using object repository is the segregation of objects from test cases. If the locator value of one webelement changes, only the object repository needs to be changed rather than making changes in all test cases in which the locator has been used. Maintaining an object repository increases the modularity of framework implementation. A property file stores information in a Key-Value pair. Key value pair is represented by two string values separated by the equal to sign.

There are two ways to create object repository in Selenium:

1. any companies also use Page Object Model to store all locators which also make the test in a readable format. Depends on your current framework style you can adopt any of these. Personally, I use Page Object Model using PageFactory which make my test more readable and in terms of performance as well.
2. Using Properties file, Ex: ObjectRepo.properties

16. What are the types of WebDriver API's that are supported/available in Selenium?

Answer: Please refer answer of question#108

17. Which WebDriver implementation claims to be the fastest?

Answer: The fastest implementation of WebDriver is the HTMLUnitDriver. It is because the HTMLUnitDriver does not execute tests in the browser and also called as a Headless browser.

18. What is the difference between Soft Assert and Hard Assert in Selenium?

Answer: Please refer answer of question#13

19. What are the verification points available in Selenium?

Answer: Here some examples of verification

To check if element is enabled or not

```
driver.findElement(By.id("<ID>")).isEnabled()
```

To check if element is displayed or not

```
driver.findElement(By.id("<ID>")).isDisplayed()
```

To check if element is selected or not

```
driver.findElement(By.id("<ID>")).isSelected()
```

To check if element is currently active or not on the page

```
driver.findElement(By.id("<ID>")).equals(driver.switchTo().activeElement());
```

20. Is Selenium Server needed to run Selenium WebDriver scripts?

Answer: In case of Selenium WebDriver, it does not require to start Selenium Server for executing test scripts. Selenium WebDriver makes the calls between browser & automation script. Selenium WebDriver has native support for each browser to supports Test Automation; on the same machine (both Selenium WebDriver Automation tests & browsers are on same machine).

21. What are the different ways for refreshing the page using Selenium WebDriver?

Answer: 1) **Refresh command:** The most commonly used and simple command for refreshing a webpage.

```
driver.get("https://www.google.co.in");
```

```
driver.navigate().refresh();
```

2) **SendKeys command:** Second most commonly used command for refreshing a webpage. As it is using a send keys method, we must use this on any Text box on a webpage.

```
driver.get("https://www.google.co.in");
```

```
// Element "q" is a Search Text box on google website
```

```
driver.findElement(By.name("q")).sendKeys(Keys.F5);
```

3) **Get command:** This is a tricky one, as it is using another command as an argument to it. If you look carefully, it is just feeding get command with a page URL.

```
driver.get("https://www.google.co.in");
```

```
driver.get(driver.getCurrentUrl());
```

4) **To command:** This command is again using the same above concept. `navigate().to()` is feeding with a page URL and an argument.

```
driver.get("https://www.google.co.in");
```

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

5) **SendKeys command:** This is the same SendKeys command but instead of using Key, it is using ASCII code.

```
driver.get("https://www.google.co.in");
```

```
driver.findElement(By.name("s")).sendKeys("\uE035");
```

22. What is the difference between `driver.getWindowHandle()` and `driver.getWindowHandles()` in Selenium WebDriver and their return type?

Answer: The main difference between `driver.getWindowHandle()` and

`driver.getWindowHandles()` are :-

`driver.getWindowHandle()` is used to handle single window i.e. main window and

`driver.getWindowHandles()` is used to handle multiple windows.

`driver.getWindowHandle()` return type is string and `driver.getWindowHandles()` return type is `Set<string>`.

23. What is JavascriptExecutor and in which case JavascriptExecutor will help in Selenium automation?

Answer: JavaScriptExecutor is used when Selenium Webdriver fails to click on any element due to some issue. JavaScriptExecutor provides two methods "executescript" & "executeAsyncScript" to handle. Executed the JavaScript using Selenium Webdriver.
JavascriptExecutor js = (JavascriptExecutor) driver;
js.executeScript(Script,Arguments);

24. List some scenarios which we cannot automate using Selenium WebDriver?

Answer: 1.Pop-up Windows

When a simple, prompt, or confirmation alert pops-up, it can be difficult to automate it to either accept or close.

2.Dynamic Content

A web page that has dynamically loaded content has elements that may not be at first visible when you visit the site.

3.Flakiness

Sometimes Selenium will give you flaky tests, which means they'll come back as positive or negative when they actually are the opposite.

4. Mobile Testing

While Selenium WebDriver can test on any operating system and browser on desktops, it's limited when it comes to mobile testing in that it cannot run on native operating systems like iOS and Android.

5. Limited Reporting

While Selenium will exponentially increase your automated testing capabilities, because it's an open source tool it is limited and features and does not support much reporting on its own.

25. How can use Recovery Scenario in Selenium WebDriver?

Answer: Recovery scenarios depends upon the programming language you use. If you are using Java then you can use exception handling to overcome same. By using "Try Catch Block" within your Selenium WebDriver Java tests.

26. Have you used any cross browser testing tool to run Selenium Scripts on cloud?

Answer: We can perform cross browser using TestNG framework.

Some other popular tools for cross browser testing are:

1) LambdaTest

LambdaTest is a cloud based platform that helps you perform cross browser compatibility testing of your web app or websites. You can run automated selenium scripts on LambdaTest's scalable cloud grid, or can even perform live interactive testing on real browser environments.

Key Features:

Run Selenium automation tests on a scalable Selenium grid having 2000+ browser environments

Execute automated screenshot and responsive testing of your website

Test your locally or privately hosted website using SSH Tunnel

One click bug logging to your favorite bug tracking tool like Asana, BitBucket, GitHub, JIRA, Microsoft VSTS, Slack, Trello etc.

2) CrossBrowser Testing - Crossbrowser testing tool has wide range of different browsers and their versions. It is available for multiple OS. It supports over 1000 combinations of different browsers and O.S including mobile browsers.

3) Browser-Stack- With browser stack it is possible to do web based browser testing on desktop and mobile browser. It is cloud based and so it does not require any installation, and

the pre-installed developer tools are useful for quick cross-browser testing and debugging. With browser-stack you can set up a comprehensive testing environment with support for proxies, firewalls and Active Directory. It supports opera mobile, Android, Windows (XP, 7 and 8), iOS, OSX snow leopard, lion and mountain lion and so on. Browser stack allows you to test your pages remotely.

4) Sauce Labs- It is the leading cloud based web and mobile app testing platform. It allows you to run tests in the cloud on more than 260 different browser platform and devices. There is no VM set up or maintenance required.

27. What are the DesiredCapabilities in Selenium WebDriver and their use?

Answer: Desired Capabilities is a class used to declare a set of basic requirements such as combinations of browsers, operating systems, browser versions, etc. to perform automated cross browser testing of a web application.

When we try to automate our test scripts through Selenium automation testing, we need to consider these combinations to declare a specific test environment over which our website or web application should render seamlessly.

Desired Capabilities class is a component of the `org.openqa.selenium.remote.DesiredCapabilities` package. It helps Selenium WebDriver set the properties for the browsers. So using different capabilities from Desired Capabilities class we can set the properties of browsers. For example, the name of the browser, the version of the browser, etc. We use these capabilities as key-value pairs to set them for browsers.

28. While injecting capabilities in WebDriver to perform tests on a browser (which is not supported by a webdriver), what is the limitation that one can come across?

Answer: Major limitation of injecting capabilities is that “findElement” command may not work as expected.

29. List out the test types that are supported by Selenium?

Answer: Using Selenium type of testing can be done are:

- Functional Testing.
- Regression Testing.
- Sanity Testing.
- Smoke Testing.
- Responsive Testing.
- Cross Browser Testing.
- UI testing (black box)
- Integration Testing.

30. Explain what is assertion in Selenium and what are the different types of assertions?

Answer: Please refer answer of question#13

31. List out the technical challenges with Selenium?

Answer: The Most Common Selenium Challenges

Pop-up Windows. When a simple, prompt, or confirmation alert pops-up, it can be difficult to automate it to either accept or close. ...

- Dynamic Content.
- Flakiness.
- Mobile Testing.
- Limited Reporting.
- Multi-tab Testing.
- Manual Testing.
- Scalability.

32. What is the difference between type keys and type commands?

Answer: 1. The sendKeys" command does not replace the existing text content in the text box whereas the "type" command replaces the existing text content of the text box. 2. It will send explicit key events like a user pressing a key on the keyboard.

33. While using click() command, can you use screen coordinates?

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

34. What are the four parameters you have to pass in Selenium?

Answer: In total, there are four conditions (parameters) for Selenium to pass a test. These are as follows: URL, host, browser and port number.

35. What is the difference between setSpeed() and sleep() methods?

Answer: Both sleep() and setSpeed() are used to delay the speed of execution. The main difference between them is that: setSpeed sets a speed that will apply a delay time before every Selenium operation. ... sleep() will set up wait only for once when called.Both sleep() and setSpeed() are used to delay the speed of execution. The main difference between them is that: setSpeed sets a speed that will apply a delay time before every Selenium operation. ... sleep() will set up wait only for once when called.

36. What is heightened privileged browsers?

Answer: The purpose of heightened privileges is similar to Proxy Injection, allows websites to do something that are not commonly permitted.

The key difference is that the browsers are launched in a special mode called heightened privileges. By using these browser mode, Selenium core can open the AUT directly and also read/write its content without passing the whole AUT through the Selenium RC server.

37. Which attribute you should consider throughout the script in frame for (if no frame id as well as no frame name")?

Answer: You can use.....driver.findElements(By.xpath("//iframe"))....
This will return list of frames.

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

38. List the advantages of Selenium WebDriver over Selenium Server?

Answer: WebDriver is faster than Selenium RC because of its simpler architecture. WebDriver directly talks to the browser while Selenium RC needs the help of the RC Server in order to do so. WebDriver's API is more concise than Selenium RC's. WebDriver can support HtmlUnit while Selenium RC cannot.

40. What is IntelliJ and how it is different from Eclipse IDE?

Answer:

IntelliJ IDEA is the most powerful, popular, and fully-featured IDE for Java Developers, which was released for the public in 2001. It is developed and maintained by Jet Brains Company. It is licensed by Apache 2.0.

Eclipse

Eclipse is an open-source IDE for developing applications using the Java, Python, Ruby, C, C++, etc. The IBM released it in 2001 under the Eclipse Public License (EPL). It became popular soon for developing free and commercial projects. Today, it became the most popular Java IDE.

System Requirements

We can install IntelliJ Idea on Windows, macOS and Linux with the following hardware:

2 GB RAM minimum, 4 GB RAM recommended

1.5 GB hard disk space + at least 1 MB for caches

1024×768 minimum screen resolution

We can run Eclipse IDE on any platform that supports JVM including Windows, macOS, Linux and Solaris. It demands the following hardware:

0.5 GB RAM minimum, 1+ GB RAM recommended

300 MB hard disk space minimum, 1+ GB recommended

Processor speed of 800 MHz minimum, 1.5 GHz or faster recommended.

41. What are Selenium WebDriver Listeners?

Answer: Listeners “listen” to the event defined in the selenium script and behave accordingly. The main purpose of using listeners is to create logs and reports. There are many types of listeners such as WebDriver Event Listeners and TestNG Listeners.

We need to know the following class and interface when we talk about listeners in Selenium.

WebDriverEventListener: This WebDriver Event Listener interface allows us to implement the methods

Once the script is executed, Selenium WebDriver does perform activities such as Type, Click, Navigate etc., To keep track of these activities we use WebDriver Event Listeners interface.

EventFiringWebDriver: This EventFiringWebDriver class actually fire WebDriver event
Let's see how to implement Listeners in Selenium WebDriver Script.

42. What are the different types of Listeners in TestNG?

Answer: What is Listeners in TestNG?

Listener is defined as interface that modifies the default TestNG's behavior. As the name suggests Listeners "listen" to the event defined in the selenium script and behave accordingly. It is used in selenium by implementing Listeners Interface. It allows customizing TestNG reports or logs. There are many types of TestNG listeners available. There are many types of listeners which allows you to change the TestNG's behavior. Below are the few TestNG listeners:

- IAnnotationTransformer ,
- IAnnotationTransformer2 ,
- IConfigurable ,
- IConfigurationListener ,
- IExecutionListener,
- IHookable ,
- IInvokedMethodListener ,
- IInvokedMethodListener2 ,

- IMethodInterceptor ,
- IReporter,
- ISuiteListener,
- ITestListener .

ITestListener is one of the most important listener. ITestListener has following methods

OnStart- OnStart method is called when any Test starts.

onTestSuccess- onTestSuccess method is called on the success of any Test.

onTestFailure- onTestFailure method is called on the failure of any Test.

onTestSkipped- onTestSkipped method is called on skipped of any Test.

onTestFailedButWithinSuccessPercentage- method is called each time Test fails but is within success percentage.

onFinish- onFinish method is called after all Tests are executed.

43. What is the API that is required for implementing Database Testing using Selenium WebDriver?

Answer: Selenium WebDriver alone is ineligible to perform database testing but this can be done using Java Database Connectivity API (JDBC). The API lets the user connect and interact with the data source and fetch the data with the help of automated queries.

44. When to use AutoIt?

Answer: AutoIt v3 is also freeware. It uses a combination of mouse movement, keystrokes and window control manipulation to automate a task which is not possible by selenium webdriver.

Why Use AutoIt?

Selenium is an open source tool that is designed to automate web-based applications on different browsers but to handle window GUI and non HTML popups in application. AutoIT is required as these window based activity are not handled by Selenium.

45. Why do we need Session Handling while using Selenium WebDriver?

Answer: During test execution, the Selenium WebDriver has to interact with the browser all the time to execute given commands. This can be achieved using Session Handling in Selenium.

46. What is exception test in Selenium?

Answer: TestNG provides an option of tracing the exception handling of code. You can test whether a code throws a desired exception or not. Here the expectedExceptions parameter is used along with the @Test annotation.

47. How do you achieve synchronization in WebDriver?

Answer: Please refer below article on waits

<https://www.automationtestinginsider.com/2020/02/waits-in-selenium-webdriver.html>

48. Can Bar Code Reader be automated using Selenium?

Answer: Selenium has limitation to automate Bar code but by using third party API we can automate Bar codes. So, ZXing is one the third party API will be used to automate Bar Codes.

49. What is Robot API? What methods of Robot class do you know?

Answer: As per the class description, this class is used to generate native system input events. This class uses native system events to control the mouse and keyboard. java.awt.Robot class provides various methods needed for controlling mouse and keyboard. Following are some of the methods commonly used in browser test automation:

Keyboard methods:

keyPress(int keycode): This method presses a given key. For Example,

keyPress(KeyEvent.VK_SHIFT) method presses "SHIFT" key

keyRelease(int keycode): This method releases a given key. For Example,

keyRelease(KeyEvent.VK_SHIFT) method releases "SHIFT" key

Mouse Methods:

mousePress(int buttons): This method presses one or more mouse buttons. For Example,

mousePress(InputEvent.BUTTON1_DOWN_MASK) method is used left click mouse button

mouseRelease(int buttons): This method releases one or more mouse buttons. For Example,

mouseRelease(InputEvent.BUTTON1_DOWN_MASK) method is used to release the left

mouse button click

mouseMove(int x, int y): This method moves the mouse pointer to given screen coordinates specified by x and y values. For Example, mouseMove(100, 50) will move the mouse pointer to the x coordinate 100 and y coordinate 50 on the screen.

50. Which package can be imported while working with WebDriver?

Answer: org.openqa.selenium

51. What is the purpose of deselectAll() method?

Answer: deselectAll() Method

deselectAll() method is useful to remove selection from all selected options of select box. It will work with multiple select box when you need to remove all selections. Syntax for deselectAll() method is as below.

```
Select listBox = new Select(driver.findElement(By.xpath("//select[@name='FromLB']")));  
listBox.deselectAll();
```

52. What is the purpose of getOptions() method?

Answer: getOptions() is used to get all options from the dropdown list. This gets the all options belonging to the Select tag. It takes no parameter and returns List<WebElement>. Sometimes you may like to count the element in the dropdown and multiple select box, so that you can use the loop on Select element. Syntax for get method is:

dropdownElement.getOptions();

Example : Select optionSelect = new Select(driver.findElement(By.id("dropdown_cities")));

List <WebElement> elementCount = optionSelect.getOptions();

System.out.println(elementCount.size());

53. What could be the cause for Selenium WebDriver test to fail?

Answer: 1. Asynchronous websites

Selenium tests are developed keeping a particular order in mind and when there are websites that are built using asynchronous architecture. The order of response cannot be fixed at all times. In such scenarios, sometimes the responses from the website are not in order. As a result, the test case fails. Such failures are tough to debug as well because they are not reproducible at all times.

2. Dynamic websites that change without refresh

Now websites are built that do a page reload or refresh rarely and, all the changes happen dynamically using JavaScript on the same page. In some scenarios, if there is a small error in

one step, the whole test case fails because there is no opportunity to reload or refresh the page.

3. Alerts, Pop-ups, Nested IFrames

Selenium Users faced problems while automating alerts, pop-ups and Nested IFrames related functionalities on a website but with the latest version of Selenium, this problem has been handled quite well.

4. Selenium WebDriver version mismatch

Selenium releases WebDrivers for supported browsers to execute tests on them. If the installed browser on the system does not match the versions supported by the web drivers, an exception like this is reported: “This version of ChromeDriver only supports Chrome version 74”.

5. Selectors move on screen

Probably the single greatest cause of Selenium failures is when a selector moves or subtly changes on screen.

54. Can we test APIs or web services using Selenium WebDriver?

Answer: To do API testing, one should only use – Postman/Rest Assured (which is a Java based library which possess inbuilt API methods)/Jmeter & Soap Ui. Selenium is an API for automation of browsers. The API testing can be included in Selenium framework. In postman, code for any API request can be generated.

55. What are some expected conditions that can be used in Explicit Waits?

Answer: In order to declare explicit wait, one has to use “ExpectedConditions”. The following Expected Conditions can be used in Explicit Wait.

- `alertIsPresent()`
- `elementSelectionModeToBe()`
- `elementToBeClickable()`
- `elementToBeSelected()`
- `frameToBeAvaliableAndSwitchToIt()`
- `invisibilityOfTheElementLocated()`
- `invisibilityOfElementWithText()`
- `presenceOfAllElementsLocatedBy()`
- `presenceOfElementLocated()`
- `textToBePresentInElement()`
- `textToBePresentInElementLocated()`
- `textToBePresentInElementValue()`
- `titleIs()`
- `titleContains()`
- `visibilityOf()`
- `visibilityOfAllElements()`
- `visibilityOfAllElementsLocatedBy()`
- `visibilityOfElementLocated()`

Syntax:

```
WebDriverWait wait= new WebDriverWait(driver, 5);  
WebElement  
ele=wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("xpath")));
```

56. What is HtmlUnitDriver?

Answer: HtmlUnit is a headless web browser written in Java. It allows high-level manipulation of websites from other Java code, including filling and submitting forms and clicking hyperlinks. It also provides access to the structure and the details within received web pages.

A headless browser is a web-browser without a graphical user interface. This program will behave just like a browser but will not show any GUI.

Some of the examples of Headless Drivers include

- HtmlUnit
- Ghost
- PhantomJS
- ZombieJS

Watir-webdriver

57. Name an API used for logging in Java?

Answer: util. logging. Logger is the class used to log application messages in java logging API

58. What is the use of logging in Automation?

Answer: Advantages of Logging in Selenium Scripts:

Grants a complete understanding of test suites execution. Log messages can be stored in external files for post-execution scrutiny. Logs are an exceptional assistant in debugging the program execution issues and failures.

59. Can Selenium Test an application on Android Browser?

Answer: Selenium should be able to handle Android browser. There is a Selenium Android Driver for running tests in Android browser.

You can use Selendroid or Appium framework to test native apps or web apps in Android browser

60. Give the example for method overload in Selenium?

Answer: Method overloading example

You want to have your own Listbox class to interact with dropdown lists.

The Listbox class is just a wrapper around the Select class.

You want it to have simpler method names for selecting list options:

```
public class Listbox
{
    Select list;
    public Listbox(Select list) {
        this.list = list;
    }
    public void select(int i) {
        this.list.selectByIndex(i);
    }
    public void select(String text) {
        this.list.selectByVisibleText(text);
    }
    public void deSelect(int i) {
        this.list.deselectByIndex(i);
    }
}
```

```

}
public void deSelect(String text) {
this.list.deselectByVisibleText(text);
}
//other methods }

```

61. What is WebDriverBackedSelenium?

Answer: WebDriverBackedSelenium is an implementation of the Selenium-RC API by Selenium Webdriver, which is primarily provided for backwards compatibility. It allows to test existing test suites using the Selenium-RC API by using WebDriver under the covers. WebDriver Backed Selenium is used for migrating Selenium 1.0(Selenium RC) to Selenium WebDriver tests.

62. What is the use of contextClick()?

Answer: This method is from Actions class. ContextClick Method. Right-clicks the mouse at the last known mouse coordinates. Right-clicks the mouse on the specified element.

Please refer below article:

<https://www.automationtestinginsider.com/2020/01/perform-double-click-and-right-click.html>

63. How do you accommodate project specific method in your framework?

Answer: 1st go through all the manual test cases and identify the steps which are repeating. Note down such steps and make them as methods and write into ProjectSpecificLibrary.

64. What is Actions class in WebDriver and its methods?

Answer: Handling special keyboard and mouse events are done using the Advanced User Interactions API. It contains the Actions and the Action classes that are needed when executing these events.

Some Examples: Mouse double click, Drag and Drop, Handling tooltip, Mouse hover, entering uppercase letters in the textbook using Shift+Letters .

Actions Class: Actions class is an API for performing complex user web interactions like double-click, right-click, etc. and it is the only choice for emulating Keyboard and Mouse interactions.

Please refer below article:

<https://www.automationtestinginsider.com/2020/01/actions-class-in-selenium.html>

65. What are different versions of Selenium available you have used and what are the additional features you have seen from the previous versions?

Answer:

Version	Version	Comparison
Selenium 1	Selenium RC	Essentially the same thing. Selenium 1 has never been an official name, but is commonly used in order to distinguish between versions.
Selenium 2	Selenium WebDriver	Essentially the same thing.

		The term "Selenium WebDriver" is now more commonly used.
Selenium RC	Selenium WebDriver	Selenium RC is the predecessor of Selenium WebDriver. It has been deprecated and now released inside Selenium WebDriver for backward compatibility.
Selenium IDE	Selenium RC/WebDriver	Selenium IDE is a recording tool for automating Firefox, with the ability to generate simple RC/WebDriver code. Selenium RC/WebDriver are frameworks to automate browsers programmatically.
Selenium Grid	Selenium WebDriver	Selenium Grid is a tool to execute Selenium tests in parallel on different machines. Selenium WebDriver is the core library to drive web browsers on a single machine.

66. What are the challenges have you faced with Selenium and how did you overcome them?

Answer: Some of the challenges given below:

- We cannot test windows application.
- We cannot test mobile apps.
- Limited reporting.
- Handling dynamic Elements.
- Handling page load.
- Handling pop up windows.
- Handling captcha.

67. Which of the WebDriver APIs is the fastest and why?

Answer: Selenium is a GUI automated testing tool therefore execution speed depends on how fast a particular browser can respond to action events. PhantomJS or HtmlUnit for that matter would be the fastest in terms of execution speed as both are headless browsers.

68. Give different examples for method overloading and overriding in Selenium Project?

Answer: Please refer last section in below article

<https://www.automationtestinginsider.com/2020/02/selenium-data-driven-framework-with-pom.html>

69. How do you debug your automation code when it is not working as expected?

Answer: Please refer below article

<http://total-qa.com/selenium-webdriver-debugging-code-breakpoints-java-debugging-eclipse/>

70. What are the end methods you use for verifying whether the end result is achieved by our Selenium automation scripts?

Answer: Answer will be provided soon

71. What is the use of property file in Selenium?

Answer: properties' files are mainly used in Java programs to maintain project configuration data, database config or project settings etc. Each parameter in properties file are stored as a pair of strings, in key and value format, where each key is on one line. It can also be used as object repository.

72. How will you ensure that the page has been loaded completely?

Answer: Example for using the above specified JavaScript code in Selenium for ensuring the completeness of page loading: The below reusable user defined method can be called anytime you want to check the completeness of page loading:

```
public void waitForPageLoad(WebDriver driver, int timeout) {
    ExpectedCondition<Boolean> pageLoadCondition = new ExpectedCondition<Boolean>() {
        public Boolean apply(WebDriver driver) {
            return ((JavascriptExecutor)driver).executeScript("return
document.readyState").equals("complete");
        }
    };
    WebDriverWait wait = new WebDriverWait(driver, timeout);
    wait.until(pageLoadCondition);
}
```

We can call the above method using the following method calling statement:

```
waitForPageLoad(driver, 30); //waits till page load gets completed by setting a time limit of 30 seconds
```

73. Class.forName(X). What will you write in place of X?

Answer: Class.forName("X") loads the class if it not already loaded. The JVM keeps track of all the classes that have been previously loaded. This method uses the classloader of the class that invokes it. The "X" is the fully qualified name of the desired class.

74. Explain about typecasting?

Answer: Type casting in Java is to cast one type, a class or interface, into another type i.e. another class or interface. Since Java is an Object oriented programming language and supports both Inheritance and Polymorphism, It's easy that Super class reference variable is pointing to SubClass object but the catch here is that there is no way for Java compiler to know that a Superclass variable is pointing to SubClass object. Which means you can not call a method which is declared in the subclass. In order to do that, you first need to cast the Object back into its original type. This is called type casting in Java.

75. What is the default timeout of Selenium WebDriver?

Answer: The default WebDriver setting for timeouts is never. WebDriver will sit there forever waiting for the page to load. Apparently there is a timeout. It is 30 minutes long.

76. What is the difference between isDisplayed() and isEnabled() functions in Selenium WebDriver?

Answer: isDisplayed() is capable to check for the presence of all kinds of web elements available. isEnabled() is the method used to verify if the web element is enabled or disabled within the webpage. isEnabled() is primarily used with buttons.

77. Can you test flash images in Selenium?

Answer: Yes

Why flash object capturing is difficult? How is it resolved?

Flash is an outdated technology. It is difficult to capture a flash object as it is different from HTML. Also, Flash is an embedded SWF file (Small Web Format). It is also difficult to access Flash object on a mobile device.

You use to write a script using any automation tool like Selenium, SoapUI, TestComplete, etc. and execute the script.

Difference between the Flash and other element.

As mentioned above, the main difference between flash and other element is that Flash is embedded in SWF files, while other elements are embedded in HTML files. That's why HTML is easy to capture compared to flash.

Below are the requirements in order to test the flash application

Flash Application.

Support web browser.

Adobe Flash player plugins.

78. How to verify whether an object is present on the multiple pages?

Answer: answer will be provided soon.

79. Which driver implementation will allow headless mode?

Answer: HTML UnitDriver is the most light weight and fastest implementation headless browser for of WebDriver. It is based on HtmlUnit. It is known as Headless Browser Driver. It is same as Chrome, IE, or FireFox driver, but it does not have GUI so one cannot see the test execution on screen.

80. What are important Classes in WebDriver?

Answer: The major implementation classes of WebDriver interface are ChromeDriver, EdgeDriver, FirefoxDriver, InternetExplorerDriver etc. Each driver class corresponds to a browser. We simply create the object of the driver classes and work with them. It helps you to execute Selenium Scripts on Chrome browser.

81. What is Sikuli, its purpose and explain more about it?

Answer: Sikuli is a scripting language which helps us to automate Software testing of Graphical User Interface(GUI). ... It basically uses image recognition technology as it helps us to identify and control GUI components. It is robust and powerful GUI automation tool.

82. What is Robot Class and explain more about it?

Answer: Answer given in question#49.

83. What is AutoIt and explain more about it?

Answer: AutoIt is an open source automation language for Windows operating system. AutoIt uses the combination of simulated keystrokes, mouse movement, and window manipulation in order to perform user interface testing.

Article: <https://www.automationtestinginsider.com/2020/01/autoit-tool-and-two-ways-to-handle.html>

84. What is the difference between AutoIt, Sikuli and Robot Class?

Answer: answer will be provided soon.

85. Which is best in AutoIt, Sikuli and Robot Class along with reasons?

Answer: answer will be provided soon.

86. What classes you will use for reading the PDF files?

Answer: We will use PDFBox API to read PDF file using Java code.

87. What is the use of pdfstripper class?

Answer: Class PDFTextStripper. This class will take a pdf document and strip out all of the text and ignore the formatting and such. Please note; it is up to clients of this class to verify that a specific user has the correct permissions to extract text from the PDF document.

88. What is verbose?

Answer: TestNG - Verbose Attribute [Selenium Users] Verbose Attribute lets you obtain clear reports through IDE console. This attribute will be placed inside the <Suite> tag of testng.xml as shown below: <suite name="Suite" parallel="tests" verbose="2">

89. What is thread count?

Answer: It's the number of tests that are run at the same time. It takes up more slots in the grid because it uses one grid slot per test run. It's like saying how many cars in a fleet that you want on the highway at the same time. If you say you want them to take up three lanes, then they will take up no more than three lanes, but it will take longer to get all cars through. If you say five lanes, then they will take up no more than five lanes, but it will take less time to get all cars through.

90. What is the difference between build and perform methods in Actions Class?

Answer: build() method in Actions class is used to create chain of action or operation you want to perform. perform() this method in Actions Class is used to execute chain of action which are build using Action build method.

91. How will you install ReportNG in your project?

Answer: ReportNG is a simple plug-in for the TestNG unit-testing framework to generate HTML reports as a replacement for the default TestNG HTML reports. You can also customize html report with the help of TestNG listeners.

To use ReportNG reports we need to follow the below three steps:

Step 1: Add the below Jars Files to your project.

reportng-1.1.4.jar
velocity-dep-1.4.jar
guice-3.0.jar

Step 2: To make sure reportNG reports, we need to disable the default TestNG Listeners.

It can be done by following the below steps:

1. Right Click on Properties
2. Click on TestNG
3. You will find an option as "Disable default listeners", check the checkbox
4. Click on "Apply" button, it will show as message as "Project preferences are saved".
5. Now Click on "OK" button.

Step 3: We need to add the below two listeners to testng.xml file.

```
<listeners>
  <listener class-name="org.uncommons.reportng.HTMLReporter"/>
  <listener class-name="org.uncommons.reportng.JUnitXMLReporter"/>
</listeners>
```

Finally testng.xml file should look as the below for the given example :

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
<suite name="Suite1" verbose="1" >
<listeners>
  <listener class-name="org.uncommons.reportng.HTMLReporter"/>
  <listener class-name="org.uncommons.reportng.JUnitXMLReporter"/>
</listeners>
<test name="Regression Test Suite" >
  <packages>
    <package name="packOne" />
    <package name="packTwo" />
  </packages>
</test>
</suite>
```

92. Why do we go for Apache POI API? What is its purpose?

Answer: Apache POI (Poor Obfuscation Implementation) is an API written in Java to support read and write operations – modifying office files. This is the most common API used for Selenium data driven tests to read/write excel file.

93. Why do we provide “//” in java while fetching a path of excel?

Answer: In a string a single backslash is a so-called 'escape' character. This is used to include special characters like tab (\t) or a new line (\n). In order to specify a backslash in the case of a path you will have to 'escape' the single slash using the escape character, which means that you will have to write a double backslash.

94. What is an object array and why do we use it for data provider?

Answer: An array of array of objects (Object[][]) where the first dimension's size is the number of times the test method will be invoked and the second dimension size contains an array of objects that must be compatible with the parameter types of the test method. The reason is quite simple as we get test data in forms of rows and columns so that we need 2D object (object type because we are free to pass any type data String, int etc.) array. An important feature provided by TestNG is the testng DataProvider feature. It helps you to write data-driven tests which essentially means that same test method can be run multiple times with different data-sets.

A Data Provider is a method on your class that returns an array of array of objects.
 //This method will provide data to any test method that declares that its Data Provider
 //is named "test1"
 @DataProvider(name = "test1")
 public Object[][] createData1() {
 return new Object[][] {
 { "Cedric", new Integer(36) },
 { "Anne", new Integer(37)},
 };
 }
 //This test method declares that its data should be supplied by the Data Provider
 //named "test1"
 @Test(dataProvider = "test1")
 public void verifyData1(String n1, Integer n2) {
 System.out.println(n1 + " " + n2);
 }

95. In how many ways, can I take keyboard inputs?

Answer: There are many ways to read data from the keyboard in java:

- InputStreamReader Class.
- Scanner Class.
- Using Command Line Arguments.
- Console Class.

96. Is it possible to compare two images with Sikuli?

Answer: answer will be provided soon.

97. What is the difference between “type” and “typeAndWait” command?

Answer: "type" command is used to type keyboard key values into the text box of software web application. It can also be used for selecting values of combo box whereas "typeAndWait" command is used when your typing is completed and software web page start reloading. This command will wait for software application page to reload. If there is no page reload event on typing, you have to use a simple "type" command.

98. How can you run Selenium Server other than the default port 4444?

Answer: The Hub will listen to port 4444 by default. You can view the status of the hub by opening a browser window and navigating to <http://localhost:4444/grid/console>. To change the default port, you can add the optional -port flag with an integer representing the port to listen to when you run the command.

99. Explain how you can capture server side log Selenium Server?

Answer: Selenium RC Server Logging has two options: Server-Side Logs and Browser-Side Logs.

Server-Side Logs:

When launching Selenium server with the -log option, the server can record valuable debugging information reported by the Selenium Server to a text file.

For example: this command below will create a file named "selenium.log" under c:\SeleniumTestCase folder.

```
C:\selenium-java-2.39.0>java -jar selenium-server-standalone-2.39.0.jar -log  
c:\SeleniumTestCase\selenium.log
```

Another example: this example below creates a debug.log file with debug information and nnn.log file

```
C:\selenium-java-2.39.0>java -jar selenium-server-standalone-2.39.0.jar -debug -log  
c:\SeleniumTestCase\debug.log -DSelenium.LOGGER=nnn.log
```

Setting system property Selenium.LOGGER to nnn.log

Browser-Side Logs:

JavaScript on the browser side also logs important messages. To access browser-side logs, pass the `-browserSideLog` argument to the Selenium Server.

For example: this command below creates a log on the browser side under `c:\SeleniumTestCase` folder. If `-log` option is not defined, the server console will display all the debug information.

```
C:\selenium-java-2.39.0>java -jar selenium-server-standalone-2.39.0.jar -browserSideLog -log c:\SeleniumTestCase\browserside.log
```

100. Can you handle flash using web driver?

Answer: Answer given in question#77

101. What is the difference between `dragAndDrop()` and `dragAndDropBy()`?

Answer: The Actions class has two methods that support Drag and Drop.

`Actions.dragAndDrop(SourceLocator, DestinationLocator)`

In `dragAndDrop` method, we pass the two parameters -

First parameter "SourceLocator" is the element which we need to drag

Second parameter "DestinationLocator" is the element on which we need to drop the first element

`Actions.dragAndDropBy(SourceLocator, x-axis pixel of DestinationLocator, y-axis pixel of DestinationLocator)`

`dragAndDropBy` method we pass the 3 parameters -

First parameter "SourceLocator" is the element which we need to drag

The second parameter is x-axis pixel value of the 2nd element on which we need to drop the first element.

The third parameter is y-axis pixel value of the 2nd element on which we need to drop the first element.

<https://www.automationtestinginsider.com/2020/01/how-to-handle-drag-and-drop-in-selenium.html>

102. What is the use of `getPageSource()`?

Answer: There is a method called `getPageSource()` in selenium webdriver. It returns string, so you can either store it in a file or can print it in the console.

103. What is bitmap Comparison? Why we use it in Selenium WebDriver?

Answer: Please refer below link

<https://www.telerik.com/forums/bitmap-comparison>

104. Tell me some of the tools name which is used to store the script in common place?

Answer: SVN, GitHub

105. What is an assertion? What is its drawback? How to overcome it?

Answer: Assertions verify that the state of the application is same to what we are expecting. Selenium Assertions can be of three types: “assert”, “verify”, and “waitFor”. When an “assert” fails, the test is aborted. When a “verify” fails, the test will continue execution, logging the failure.

Please refer detail answer in question#13

106. What is a headless browser?

Answer: Please refer answer from question#17

107. Have you ever done profiling of a web page?

Answer: profile is the collection of settings, customization, add-ons and other personalization settings that can be done on the Firefox Browser. You can customize profile to suit your Selenium automation requirement.

108. What are the different methods available in selenium webdriver?

Answer: 1. get ().

It is used to open specified url browser in windows.

Syntax:

```
//to launch the browser  
driver.get(http://google.com);
```

2. getCurrentUrl().

Its Returns title of the Browser

Syntax:

```
//to launch the browser  
Driver.get(http://google.com);  
String url=driver.getCurrentUrl();  
System.out.println(url);
```

3. getTitle().

It is used to get the title of current web page

Syntax:

```
//to launch the browser  
driver.get("http://www.google.com");  
String title=driver.getTitle();
```

4. getPageSource().

It is used to get the source of current load page

Syntax:

```
//to launch the browser  
driver.get("http://www.google.com");  
String pagesource=driver.getPageSource();  
System.out.println(pagesource);
```

5. findElement().

It is used to find the first WebElement using the given method.

Syntax:

```
//to launch the browser  
driver.get("http://www.gmail.com");  
WebElement gmailink=driver.findElement(By.id());  
System.out.println(gmailink.getText());
```

6. findElements().

It is used to find all elements within the current page

Syntax:

```
//to launch the browser
driver.get("http://www.facebook.com");
//to findelements
List links=driver.findElements(By.TagName("a"));
//Counting no of links in result page
System.out.println(links.size());
```

7. close().

Close the current window, if there are multiple windows, it will close the current window which is active and quits the browser if it's the last window opened currently.

Syntax:

```
driver.get("http://www.etestinghub.com");
driver.close();
```

8. quit().

It is used to close every associated window which is opened.

Syntax:

```
driver.get("http://www.etestinghub.com");
driver.quit();
```

9. getWindowHandle().

Whenever the web driver launches the browser it assigns the unique id to that browser which is called as window handler. This can be captured through the method.

Syntax: driver.getWindowhandle().

10. getWindowHandles().

Whenever multiple windows are opened by webdriver and we want to capture all their ids. We use this method.

Syntax: getWindowHandles().

11. switchTo().

Used to switch from one window to another window (or) window to a frame (or) frame to a window (or) window to an alert

Syntax:

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

12. navigate().

The driver to access the browser's history and to navigate to a given URL&Refresh page.

Syntax:

```
driver.get("http://gmail.com");
//navigate to page
driver.navigate().to("http://etestinghub.com ");
//navigate to back
driver.navigate().back();
//navigate to forward
driver.navigate().forward();
//navigate to refresh page
driver.navigate().refresh();
```

13. manage().

This is used to perform maximize the size of the window.

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

Web operations on web Elements

1.click()

This is used to click on webelements like link, button, radio group, checkbox, images...etc.

2.sendKeys()

Purpose: This is used to sending inputs into text fields and text areas, and also used to select value from the drop down box.

3.clear()

Purpose: This is used to clear the input from existing data.

4.getText()

Purpose: This is used to capture text of the webElement.

5. getTagName()

Purpose: This is used to capture html tag of the webElement.

6.getLocation()

This is used to capture X and Y co-ordinates of webelement in the application.

7.isSelected()

This is used to check, is the check-box is currently checked or unchecked to checked Radio buttons are selected or not.

8. isDisplayed()

This is a Boolean condition. It is used to either an element is visible or not.

If an element is displayed it gives true and an element is not displayed it gives false.

9. isEnabled()

This is a Boolean condition. It is used to either an element is enable or not.

If an element is enable it gives true and an element is disable it gives false.

10.getAttribute ()

This is used to capture the attributes which are present in web applications.

109. Which time unit we provide In time test? minutes? seconds? milliseconds? or hours? Give Example?

Answer: SECONDS

110. Can we use implicitly wait() and explicitly wait() together in the test case?

Answer: Implicit wait destroys meaning of using explicit wait when using together. So it is advised not to use implicit wait and explicit wait together. Actually when we use both waits together, both waits will be applied at the same time and it get messed up.

Now we will see reason behind these using below scenarios:

1. Explicit Wait= Implicit Wait (Say 10 seconds)

Both waits get activated at same time to locate element. Explicit wait keeps searching for an element till it is found and implicit wait allows webdriver to search till timeout. When explicit wait starts and looks for element, because of implicit wait it needs to wait for 10 seconds because element is not found. So both waits completes 10 seconds wait time.

2. Implicit wait(20) > Explicit Wait(10)

When explicit wait stars looking for element, it needs to wait for 20 seconds because of implicit wait time.

3. Implicit wait(10) < Explicit Wait(20)

When explicit wait starts looking for element, it needs to wait for 10 seconds because of implicit wait. After that implicit wait throws exception because of not able to locate element. Exception stops explicit wait to search further and does not allow to reach its timeout.

Few more points to remember:

1. The most widely used waits are implicit and explicit waits. Fluent waits are not preferable for real-time projects.

2. We use FluentWait commands mainly when we have web elements which sometimes visible in few seconds and some times take more time than usual. Mostly in Ajax applications.

3. We use mostly explicit wait because it is for specific condition/element and we have more flexibility in using explicit rather than implicit.

111. Does selenium support https protocols ?

Answer: Yes

112. What is object repository and explain page factory technique?

Answer: Page Object Model is an Object Repository design pattern in Selenium WebDriver. POM creates our testing code maintainable, reusable. Page Factory is an optimized way to create object repository in POM concept.

Page Object Model is an Object Repository design pattern in Selenium WebDriver. POM creates our testing code maintainable, reusable. Page Factory is an optimized way to create object repository in POM concept.

What is Page Factory?

Page Factory is an inbuilt Page Object Model concept for Selenium WebDriver but it is very optimized.

we follow the concept of separation of Page Object Repository and Test Methods.

Additionally, with the help of PageFactory class, we use annotations @FindBy to find WebElement. We use initElements method to initialize web elements

@FindBy can accept tagName, partialLinkText, name, linkText, id, css, className, xpath as attributes.

Let's look at the below example using Page Factory.

```
package PageFactory;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
public class HomePage {
    WebDriver driver;
    @FindBy(xpath="//table//tr[@class='TestUsername']")
    WebElement homePageUserName;
    public Guru99HomePage(WebDriver driver){
        this.driver = driver;
        //This initElements method will create all WebElements
        PageFactory.initElements(driver, this);
    }
    //Get the User name from Home Page
    public String getHomePageDashboardUserName(){
        return homePageUserName.getText();
    }
}
```

113. What is log4j and how did you use in web driver?

Answer:

Log4j is an open source logging framework. With log4j – logging behavior can be controlled by editing a configuration file only without touching the application binary and can be used to store the Selenium Automation flow logs. It equips the user with detailed context for application failures.

log4j has 3 major components:

Loggers – It is used for logging information. To use loggers we need to take care of things mentioned below:

Create object of logger class: Logger class is a Java-based utility which has all the generic methods in it to use log4j (To use Logger class we need to Import org.apache.log4j.Logger)

Define the log level: We can define the log levels in multiple forms and levels available are:

All – This level of logging will log everything, it is intended to turn on all logging.

DEBUG – It saves the debugging information and is most helpful to debug an application.

INFO – It prints informational message that highlights the progress of the application.
WARN – It designates potentially harmful situations.
ERROR – It designates error events that might still allow the application to continue running.
FATAL – It designates very severe error events that will presumably lead the application to crash
OFF – It is intended to turn off logging.
Appenders – In log4j, an output destination is called an appender. It allows the destination where the logs would get saved. It supports the following types of appenders:-
ConsoleAppender – It logs to some standard output.
File appender – It prints logs to some file at a particular destination.
Rolling file appender – It is used to for a log file with maximum size.
Layouts – It is used to format the logging information in different style.
Link - <https://logging.apache.org/log4j/1.2/manual.html>

114. Why we use selenium grid and jenkins and advantages of it?

Answer: Selenium Grid is a part of the Selenium Suite that specializes in running multiple tests across different browsers, operating systems, and machines in parallel.

Selenium Grid has 2 versions - the older Grid 1 and the newer Grid 2. We will only focus on Grid 2 because Grid 1 is gradually being deprecated by the Selenium Team.

Selenium Grid uses a hub-node concept where you only run the test on a single machine called a hub, but the execution will be done by different machines called nodes.

When to Use Selenium Grid?

You should use Selenium Grid when you want to do either one or both of following:

Run your tests against different browsers, operating systems, and machines all at the same time. This will ensure that the application you are Testing is fully compatible with a wide range of browser-O.S combinations.

Save time in the execution of your test suites. If you set up Selenium Grid to run, say, 4 tests at a time, then you would be able to finish the whole suite around 4 times faster.

Jenkins

Jenkins is an open source automation tool written in Java with plugins built for Continuous Integration purpose. Jenkins is used to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build.

115. How many types of reports you will generate in your project?

Answer: What are the essential qualities of a good test report?

Brevity –

A report should be short and concise.

It should reveal the total no. of successes as well as failures.

Trackability –

Captures all the footprints that could lead to the root cause of a failure.

Traceability –

It must provide the ability to review the following.

Historical data for test cases and failures

Age of a particular defect

Sharable –

It should support a format that you can share through email or integrate with CI tools like Jenkins/Bamboo.

Test coverage –

It should highlight the test coverage for the following.

Test coverage of the module under test.

Test coverage of the application under test.

Selenium Webdriver doesn't have a built-in reporting feature, but there are plugins like the TestNG and JUnit which can add this functionality.

TestNG HTML Report Generation.

Generating Extent HTML Reports.

116. Which is fastest web browser

Answer: Opera is the fastest browser in a recent test. PC World, one of the world's leading technology publications, recently took a more in-depth look at the top 5 web browsers for computers: Opera, Firefox, Chrome, Microsoft Edge and Internet Explorer

117. For a simple single page how to start automation testing?

Answer: answer will be provided soon

118. Diff between findby, findbys, findAll

Answer: @FindBy will return the elements depending upon how @FindBy specified inside it. to put it in simple words, @FindBy has AND conditional relationship among the @FindBy whereas @FindAll has OR conditional relationship.

119. Challenges in Selenium and how to overcome?

Answer: Please refer answer from question#31, 66

120. what are static members in selenium?

Answer: answer will be provided soon

121. which version of selenium?

Answer: Mention the version of selenium you are using. Like Selenium 3.

122. which version of Eclipse?

Answer: Mention the version of eclipse you are using

123. How do you use action class?

Answer: Refer below link

<https://www.automationtestinginsider.com/2020/01/actions-class-in-selenium.html>

124. what is assert class?

Answer: The assert methods are provided by the class org. junit. Assert which extends java. ... Object class. There are various types of assertions like Boolean, Null, Identical etc.

125. Difference between pom.xml and testng.xml

Answer: testng. xml is the configuration for TestNG testing framework (e.g. defining test suites, test listeners, etc.) pom. xml is the configuration for Maven build tool (e.g. defining build plugins, compile and test dependencies, build profiles, etc.)

126. Can we give priority as -ve

Answer: Yes

Priority is an element applicable only for @Test annotated methods. Priority should be an integer value. It can be negative, zero or positive number. ... TestNG will execute test methods from lowest to highest priority.

127. Diff between selenium 1, 2, 3, 4?

Answer: Please refer below article

<https://www.automationtestinginsider.com/2019/07/selenium-webdriver-part1-selenium-and.html>

128. When to use explicit wait?

Answer: Please refer below article

<https://www.automationtestinginsider.com/2020/02/waits-in-selenium-webdriver.html>

129. What is By and what it returns?

Answer: By is a class and Mechanism used to locate elements within a document. In order to create your own locating mechanisms, it is possible to subclass this class and override the protected methods as required, though it is expected that all subclasses rely on the basic finding mechanisms provided through static methods of this class.

130. Diff among implicit, explicit and fluent wait

Answer: Please refer below article

<https://www.automationtestinginsider.com/2020/02/waits-in-selenium-webdriver.html>

131. How to handle null pointer exception after importing a project in eclipse

Answer: answer will be provided soon

132. When to use soft assertion

Answer: When an assert fails the test script stops execution unless handled in some form. We call general assert as Hard Assert

Hard Assert – Hard Assert throws an AssertionError immediately when an assert statement fails and test suite continues with next @Test

The disadvantage of Hard Assert – It marks method as fail if assert condition gets failed and the remaining statements inside the method will be aborted.

To overcome this we need to use Soft Assert. Let's see what is Soft Assert.

Soft Assert – Soft Assert collects errors during @Test. Soft Assert does not throw an exception when an assert fails and would continue with the next step after the assert statement.

If there is any exception and you want to throw it then you need to use assertAll() method as a last statement in the @Test and test suite again continue with next @Test as it is.

133. How to login any application using 5 different users at the same time?

Answer: Using parallel testing by passing parameters from testng.xml
Here I have given example of two user login:

Test Class:

134. driver.get or driver.navigate.get which one is faster ?

Answer: main difference between get() and navigate() is, both are performing the same task but with the use of navigate() you can move back() or forward() in your session's history. navigate() is faster than get() because navigate() does not wait for the page to load fully or completely.

135. How to do Parellel testing and cross browser testing

Answer: answer will be provided soon

136. Diff between seleniumatandalone and selenium-jar files

Answer: Selenium server standalone jar is a library that provides you the classes of Selenium automation framework.

Just for additional info, selenium-standalone-server.jar is a bundled jar that contains both API and selenium server.

Selenium server is required to run older selenium RC tests or to run WebDriver tests in remote machines through selenium Grid.

137. Diff B/W java script click and normal click?

Answer: click is a function on HTML elements you can call to trigger their click handlers: element. ... onclick is a property that reflects the onclick attribute and allows you to attach a "DOMo" handler to the element for when clicks occur: element.

138. What is the diff between profiles vs options vs desiredcapabilities

Answer: Capabilities are options that you can use to customize and configure a ChromeDriver session. This page documents all ChromeDriver supported capabilities and how to use them.

There are two ways to specify capabilities.

The first is to use the ChromeOptions class.

If your client library does not have a ChromeOptions class (like the selenium ruby client), you can specify the capabilities directly as part of the DesiredCapabilities.

139. What are the new things in geckodriver

Answer: answer will be provided soon

140. What is the diff b/w POM with and without page factory?

Answer: Page Object Model (POM) and Page Factory has following differences:

A Page Object Model is a test design pattern which says organize page objects as per pages in such a way that scripts and page objects can be differentiated easily. A Page Factory is one way of implementing PageObject Model which is inbuilt in selenium.

In POM, you define locators using 'By' while in Page Factory, you use FindBy annotation to define page objects.

Page Object Model is a design approach while PageFactory is a class which provides implementation of Page Object Model design approach.

POM is not optimal as it does not provide lazy initialization while Page Factory provides lazy initialization.

Plain POM will not help in StaleElementReferecneException while Page Factory takes care of this exception by relocating web element every time whenever it is used.

In plain page object model, you need to initialize every page object individually otherwise you will encounter NullPointerException while In PageFactory all page objects are initialized (Lazily) by using initElements() method.

141. Diff between pause and thread.sleep?

Answer: The major difference is that wait() releases the lock or monitor while sleep() does

142. What is the usage of remote web driver?

Answer: RemoteWebDriver is an implementation class of the WebDriver interface that a test script developer can use to execute their test scripts via the RemoteWebDriver server on a remote machine.

143. Is selenium supports angular jar?

Answer: There is a library called ngWebDriver that is designed to automate AngularJS and Angular Web Applications using Selenium with Java. ... No need to write extra JavaScript for Angular Requests Waiting. It provides new locating techniques to use Angular Specific Attributes.

144. Diff between isDisplayed and isPresent?

Answer: There is a major difference between isDisplayed() and isPresent(). isDisplayed() - Your element is present on the page but it is displayed. isPresent() - Your element is present in entire DOM of the page. Probably it can be hidden or not disabled, but present.

1. There is a scenario whenever “Assert.assertEquals()” function fails automatically it has to take screenshot. How can you achieve this?

Answer: Please refer below link

1. By using @AfterMethod annotation method in test class.
2. By using TestNG listener ITestListener.

Let's have a look at the first method:

1. By using @AfterMethod annotation method in test class.

Utility Class:

TestClass

@AfterMethod

```
public void tearDown(ITestResult result) {  
    if(result.FAILURE==result.getStatus())  
    {  
        TakeScreenShot.screenShot(driver, result.getName());  
    }  
    driver.close();  
}
```

2. By using TestNG listener ITestListener.

BaseClass: We have created baseclass where we kept setup method to launch the browser and application and screenShot method to capture the screenshots

```
public static WebDriver driver;
```

```
public void setup() {
    System.setProperty("webdriver.chrome.driver",
        "C:\\Users\\Hitendra\\Downloads\\chromedriver_win32\\chromedriver.exe");
    driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://www.google.com/");
}
public void screenShot(String filename) {
    String dateName = new SimpleDateFormat("yyyyMMddhhmmss").format(new Date());
    TakesScreenshot takesScreenshot = (TakesScreenshot) driver;
    File source = takesScreenshot.getScreenshotAs(OutputType.FILE);
    try {
        FileUtils.copyFile(source, new
            File(System.getProperty("user.dir")+"\\ScreenShot\\"+filename+"_"+dateName+".png"));
    } catch (Exception e) {
        // TODO Auto-generated catch block
        e.getMessage();
    }
}
}
```

Utility Class: In ListenerClass we have classed screenShot method from base class. This class implements ITestListener

```
public class ListenerClass extends BaseClass implements ITestListener {
```

```
    public void onTestStart(ITestResult result) {
        // TODO Auto-generated method stub
    }

    public void onTestSuccess(ITestResult result) {
        // TODO Auto-generated method stub
    }

    public void onTestFailure(ITestResult result) {
        screenShot(result.getName());
    }

    public void onTestSkipped(ITestResult result) {
        // TODO Auto-generated method stub
    }

    public void onTestFailedButWithinSuccessPercentage(ITestResult result) {
        // TODO Auto-generated method stub
    }

    public void onStart(ITestContext context) {
```

```
// TODO Auto-generated method stub
}

public void onFinish(ITestContext context) {
// TODO Auto-generated method stub

}
}
```

2. Open a browser in memory, means whenever it will try to open a browser the browser page must not come and can perform the operation internally?

Answer:

use HtmlUnitDriver. Please refer below program.

ex-

MAVEN

1. What is Maven and explain its life cycle phases?

Answer: Maven is a build / project management tool, based on the concept of a project object model (POM) contains information of project and configuration information for the maven to build the project such as dependencies, build directory, source directory, test source directory, plugin, goals etc.

The default life cycle comprises of the following phases:

- **Validate** - validate the project is correct and all necessary information is available
- **Compile** - compile the source code of the project
- **Test** - test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed
- **Package** - take the compiled code and package it in its distributed format, such as a JAR.
- **Verify** - run any checks on results of integration tests to ensure quality criteria are met
- **Install** - install the package into the local repository, for use as a dependency in other projects locally
- **Deploy** - done in the build environment, copies the final package to the remote repository for sharing with other developers and projects.

Maven provides a common platform to perform these activities which makes programmer's life easier while handling the huge project.

Maven Build Life Cycle:

Basic maven phases are used as below.

- **clean:** deletes all artifacts and targets which are created already.
- **compile:** used to compile the source code of the project.
- **test:** test the compiled code and these tests do not require to be packaged or deployed.
- **package:** package is used to convert your project into a jar or war etc.
- **install:** install the package into the local repository for use of another project.

2. What is Maven and its advantages of using it in your Selenium Project?

Answer:

Advantages:

The benefits of Maven

- Better dependency management
- More powerful builds
- Better debugging
- Better collaboration
- More componentized builds
- Reduced duplication
- More consistent project structure

Uses in Selenium:

We can create Maven project for writing script and create dependency-using POM.xml once dependency is set Maven will download all the dependent jar files automatically and in

future if any update comes from Selenium or TestNG side it will simply update all the required changes.

3. Why maven preferred on other?

Answer: The benefits of Maven

- Better dependency management
- More powerful builds
- Better debugging
- Better collaboration
- More componentized builds
- Reduced duplication
- More consistent project structure

4. What plug-in you used in maven?

What are Maven Plugins?

Maven is actually a plugin execution framework where every task is actually done by plugins.

Maven Plugins are generally used to –

- create jar file
- create war file
- compile code files
- unit testing of code
- create project documentation
- create project reports

A plugin generally provides a set of goals, which can be executed using the following syntax –

mvn [plugin-name]:[goal-name]

For example, a Java project can be compiled with the maven-compiler-plugin's compile-goal by running the following command.

```
mvn compiler:compile
```

Plugin Types

Maven provided the following two types of Plugins –

Build plugins- They execute during the build process and should be configured in the <build/> element of pom.xml.

Reporting plugins- They execute during the site generation process and they should be configured in the <reporting/> element of the pom.xml.

Plugin

- **clean**-clean up after build.
- **compiler**-compiles java source code.
- **deploy**-deploys the artifact to the remote repository.
- **failsafe**-runs the JUnit integration tests in an isolated classloader.
- **install**-installs the built artifact into the local repository.
- **resources-copies** the resources to the output directory for including in the JAR.
- **site**-generates a site for the current project.
- **surefire**-runs the JUnit unit tests in an isolated classloader.
- **verifier**-verifies the existence of certain conditions. It is useful for integration tests.
-

5. How do you define dependencies in your Maven Project?

Answer: We define all the dependencies inside <dependencies> tag

Please refer the below sample pom.xml file for reference:

6. What is the name of maven folder which contains all the libraries?

Answer: The local repository of Maven is a folder location on the developer's machine, where all the project artifacts are stored locally. When maven build is executed, Maven automatically downloads all the dependency jars into the local repository. Usually this folder is named .m2.

Here's where the default path to this folder is – based on OS:

- Windows: C:\Users\<User_Name>\.m2
- Linux: /home/<User_Name>/.m2
- Mac: /Users/<user_name>/.m2
- And of course, for both on Linux or Mac:
- Linux/Mac: ~/.m2

7. Without internet can we work with maven repository?

Answer: You need an internet connection. Maven isn't initially self-sufficient. It needs to download a bunch of plugins along with their dependencies and the dependencies of your own project.

8. What is the difference between Maven and TestNG?

Answer: TestNG is a testing framework. It also generates testing reports. Maven is a software project management and comprehension tool. It manages all dependencies and different flows for building a project.

9. in Maven, from where the jar files will get downloaded?

Answer: When you add a dependency ("Dependency" here means what your project depends on e.g., if you want to use Log4j in your project, then you depend on that library) to the pom.xml, your IDE (Eclipse) will use Maven to download the jars and store them in your local repository (%HOME%/.m2 folder) so that you can compile your project and run it. Maven also allows manages transitive dependencies for you (libraries that your dependencies rely on).

10. In Maven, do we have to manually download and configure/update the required jar files?

Answer: No

11. What are the different plugins used for Maven? And it's use?

Answer: Covered on question#4

12. What's the difference between a Maven project and a Java project?

Answer: In Normal Java Project, if you want to work on any third party / API applications then you have to associate those jar files and associate/configure those jar files to your project manually, whereas in Maven project provide the third party/API applications dependency in POM file and then click on Maven install then automatically those respective libraries automatically to your project.

13. In Maven what are the two setting files called and what are their location?

Answer: In Maven, the setting files are called settings.xml, and the two setting files are located at

Maven installation directory: \$M2_Home/conf/settings.xml

User's home directory: \${user.home}/.m2 / settings.xml

14. What is POM?

Answer: POM stands for Project Object Model. In Maven, it is a fundamental Unit of Work and it is an XML file. You can find it in the base directory of the project. It consists of information about the project and various configuration details used by Maven to build the project(s).

15. What is the command to build your Maven site?

Answer: Type the command – mvn site

16. What would the command mvn clean do?

Answer: This command deletes the target directory with all the build data before starting the build process.

17. Tell me the command to install JAR file in local repository.

Answer: mvn install

18. What is Archetype?

Answer: An archetype is a Maven plugin whose task is to create a project structure as per its template.

19. What is the command to create a new project based on an archetype?

Answer: Type the following command – mvn archetype:generate

20. What is SNAPSHOT in Maven?

Answer: SNAPSHOT can be defined as a special version that indicates a current development copy.

TESTNG

1. What is TestNG?

Answer: Please refer the below points about TestNG-

- TestNG is an automation testing framework.
- NG stands for "Next Generation".
- Java unit testing framework.
- TestNG is an advance framework designed in a way to leverage the benefits by both the developers and testers.
- TestNG is a testing framework inspired from JUnit and NUnit but introducing some new functionalities that make it more powerful and easier to use.

Please refer link: <https://www.automationtestinginsider.com/2020/03/testng-part1-introduction-to-testng.html>

2. What are Annotations and what are the different annotations available in TestNG?

Answer: TestNG Annotation is a piece of code which is inserted inside a program or business logic used to control the flow of execution of test methods.

List of TestNG Annotations

Below are the different annotations used in TestNG

- **@BeforeSuite:** The annotated method will be run only once before all tests in this suite have run.
- **@AfterSuite:** The annotated method will be run only once after all tests in this suite have run.
- **@BeforeTest:** The annotated method will be run before any test method belonging to the classes inside the <test> tag is run.
- **@AfterTest:** The annotated method will be run after all the test methods belonging to the classes inside the <test> tag have run.
- **@BeforeClass:** The annotated method will be run only once before the first test method in the current class is invoked.
- **@AfterClass:** The annotated method will be run only once after all the test methods in the current class have run.
- **@BeforeMethod:** The annotated method will be run before each test method.
- **@AfterMethod:** The annotated method will be run after each test method.
- **@BeforeGroups -** The @BeforeGroups annotated method run only once for a group before the execution of all test cases belonging to that group.
- **@AfterGroups -** The @AfterGroups annotated method run only once for a group after the execution of all test cases belonging to that group.

Please refer link: <https://www.automationtestinginsider.com/2020/03/testng-annotations-and-their-execution.html>

3. What is JUnit?

Answer: JUnit is a unit testing framework for the Java programming language. JUnit has been important in the development of test-driven development, and is one of a family of unit testing frameworks which are collectively known as xUnit that originated with sUnit.

4. What are JUnit annotations?

Answer: Below are the annotations in JUnit

- **@Test:** This annotation is a replacement of `org.junit.TestCase` which indicates that public void method to which it is attached can be executed as a test Case.
- **@Before:** This annotation is used if you want to execute some statement such as preconditions before each test case.
- **@BeforeClass:** This annotation is used if you want to execute some statements before all the test cases for e.g. test connection must be executed before all the test cases.
- **@After:** This annotation can be used if you want to execute some statements after each Test Case for e.g. resetting variables, deleting temporary files, variables, etc.
- **@AfterClass:** This annotation can be used if you want to execute some statements after all test cases for e.g. Releasing resources after executing all test cases.
- **@Ignore:** This annotation can be used if you want to ignore some statements during test execution for e.g. disabling some test cases during test execution.
- **@Test(timeout=500):** This annotation can be used if you want to set some timeout during test execution for e.g. if you are working under some SLA (Service level agreement), and tests need to be completed within some specified time.
- **@Test(expected=IllegalArgumentException.class):** This annotation can be used if you want to handle some exception during test execution. For, e.g., if you want to check whether a particular method is throwing specified exception or not.

5. How is TestNG better than JUnit?

Answer: please refer Advantages of TestNG over Junit in below link –

<https://www.automationtestinginsider.com/2020/03/testng-part1-introduction-to-testng.html>

What is TestNG?

- TestNG is an automation testing framework.
- NG stands for "Next Generation".
- Java unit testing framework.
- TestNG is an advance framework designed in a way to leverage the benefits by both the developers and testers.
- TestNG is a testing framework inspired from JUnit and NUnit but introducing some new functionalities that make it more powerful and easier to use.

TestNG Features:

- TestNG simplifies the way the tests are coded. There is no more need for a static main method in our tests.
- Support for annotations (@).
- Using TestNG you can generate a proper report and can see passed, failed, skipped test results.
- Test case Prioritization.

- Support for Data Driven Testing using Data providers.
- Multiple test cases can be grouped more easily by converting them into testng.xml file.
- The same test case can be executed multiple times without loops just by using keyword called 'invocation count.'
- Cross browser testing.
- Parallel Testing is possible using TestNG.

Advantages of TestNG over Junit:

- There are three major advantages of TestNG over Junit.
- More Annotations than JUnit and Annotations are easier to understand
- Test cases can be grouped more easily.
- Parallel testing is possible.
- The testing framework can be easily integrated with tools like Maven, Jenkins, etc.

Install TestNG step by step

There are two ways you can install TestNG in Eclipse.

First Way on installing Eclipse is using "Install new software" option.

Second way is using "Eclipse Market Place". - This option will be available in new versions of eclipse.

Lets have a look at the first version:

Step 1:

In Eclipse, on top menu bar, Under Help Menu, Click on "Install new Software" in help window.

Step 2:

Enter the URL (<http://dl.bintray.com/testng-team/testng-eclipse-release/>) at Work With field and click on "Add" button. Add the URL and later click on next.

Step 3:

It will check for the requirement and dependencies before starting the installation. Once the above step is done, it will ask you to review the installation details. If your are ready or OK to install TestNG, click on "Next" to continue..Later Accept the Terms of the license agreement and Click on "Finish" button. If there is any problem with the requirements/dependencies, it will ask you to install them first before continuing with TestNG. Most of the cases it will successfully get installed nothing to worry about it.

It will take few minutes to get installed.

Verify TestNG installed or not?

Step 4: In Eclipse, on top menu bar, Under Help Menu, Click on "Install new Software" in help window.

Step 5: Click on already installed?

Step 6: You can verify if the TestNG is available as per the below screen shot.

What is TestNG Annotation?

TestNG Annotation is a piece of code which is inserted inside a program or business logic used to control the flow of execution of test methods.

List of TestNG Annotations

Below are the different annotations used in TestNG

@BeforeSuite: The annotated method will be run only once before all tests in this suite have run.

@AfterSuite: The annotated method will be run only once after all tests in this suite have run.

@BeforeTest: The annotated method will be run before any test method belonging to the classes inside the <test> tag is run.

@AfterTest: The annotated method will be run after all the test methods belonging to the classes inside the <test> tag have run.

@BeforeClass: The annotated method will be run only once before the first test method in the current class is invoked.

@AfterClass: The annotated method will be run only once after all the test methods in the current class have run.

@BeforeMethod: The annotated method will be run before each test method.

@AfterMethod: The annotated method will be run after each test method.

@BeforeGroups - The @BeforeGroups annotated method run only once for a group before the execution of all test cases belonging to that group.

@AfterGroups - The @AfterGroups annotated method run only once for a group after the execution of all test cases belonging to that group.

Hierarchy of the TestNG Annotations:

Hierarchy of the TestNG Annotations

@BeforeSuite

@BeforeTest

@BeforeClass

@BeforeMethod

@Test

@AfterMethod

@AfterClass

@AfterTest

@AfterSuite

Benefits of using TestNG Annotations:

- TestNG Annotations made the life of testers very easy. Based on your requirements, you can access the test methods, i.e., it has no predefined pattern or format.
- You can pass the additional parameters to TestNG annotations.
- In the case of TestNG annotations, you do not need to extend any test classes.
- TestNG Annotations are strongly typed, i.e., errors are detected at the compile time.

6. How to set test case priority in TestNG?

Answer: TestNG is a Testing framework that covers different types of test designs like a unit test, functional test, end to end test, UI test and integration test. You can run a single or multiple test cases in your Testng code.

You can set test case priority in TestNG by using priority attribute to the @Test annotations. In case priority is not set then the test scripts execute in alphabetical order. Following code snippet prioritizes the test cases:

If test priority is not defined while, running multiple test cases, TestNG assigns all @Test a priority as zero(0).

Now, while running; lower priorities will be scheduled first.

7. How to ignore test cases?

Answer: When executing TestNG tests, there may be some scenarios where you may have to disable a particular test or a set of tests from getting executed.

For example, consider a scenario where a serious bug exists in a feature due to certain tests belonging to certain scenarios that cannot be executed. As the issue has already been identified we may need to disable the said test scenarios from being executed.

Ways to Skip / Disable / Ignore Tests in TestNG:

1. Suite Level

Using “exclude” parameter in testng.xml.

TestNg provides an option to include or exclude for Groups, Test Methods, Classes and Packages using include and exclude tags by defining in testng.xml.

Only include methods will be Execute / Run other methods will be skip / ignore.

Example:

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

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

    <test name="Method Test Cases" >

        <classes>

            <class name="com.easy.entry.AddTestCase">

                <methods>

                    <include name="addLocationTestCase" />

                    <include name="addDepartmentTestCase" />

                    <exclude name="addEmployeeTestCase" />

                </methods>

            </class>

        </classes>

    </test>

</suite>
```

2. TestClass Level

To ignore all the tests in class, you need to use @Ignore annotation at the class level.

@Ignore

```
public class IgoneTestClassExample {  
  
    @Test  
  
    public void testCase1() {  
  
        System.out.println("testCase1");  
  
    }  
  
    @Test  
  
    public void testCase2() {  
  
        System.out.println("testCase2");  
  
    }  
  
}
```

3. Test Method Level

Using @Ignore at test level

```
public class IgoneTestClassExample {  
  
    @Ignore("not yet ready , Please ignore.")  
  
    @Test  
  
    public void testCase1() {  
  
        System.out.println("testCase1");  
  
    }  
  
    @Test  
  
    public void testCase2() {
```

```

        System.out.println("testCase2");
    }
}

```

Using enabled = false at test level

@Test(enabled = false)

```

public void testCase2() {

    System.out.println("testCase2");

}

```

Use of depends on in testng?

Answer: Dependency is a feature in TestNG that allows a test method to depend on a single or a group of test methods. This will help in executing a set of tests to be executed before a test method.

The dependency on multiple test methods is configured for a test by providing comma separated dependent test method names to the attribute dependsOnMethods while using the Test annotation.

+9. How do you generate reports using testng?

Answer: Report generation is very important when you are doing the Automation Testing as well as for Manual Testing. By looking at the result, you can easily identify how many test cases are passed, failed and skipped.

By looking at the report, you will come to know what the status of the project is.

Selenium web driver is used for automating the web-application, but it won't generate any reports.

There are two ways we can generate reports in testng:

Using emailable-report.html - The TestNG will generate the default report.

When you execute testng.xml file, and refresh the project. You will get test-output folder in that folder.

Right click on the emailable-report.html and select the option. Open with the web browser.

Using index.html - When you execute testng.xml file, and refresh the project. You will get test-output folder in that folder.

Right click on the index.html and select the option. Open with the web browser.

10. What are the threads in testng , uses of it?

Answer: we can utilize TestNG to perform load testing by creating multiple threads. We can even re-execute a test the no. of times we want.

TestNG is indeed a quite resourceful test automation framework. And integrating it with Webdriver only makes it better to perform automated testing. In this post, we are going to expose two powerful attributes of @Test annotation of TestNG.

These are <invocationCount> and <threadPoolSize>. The first attribute specifies the exact no. of times a test method will get called. And the latter sets the total no. of threads that will run to call the test method.

1. TestNG example using @Test(invocationCount=?) attribute.

The below code will show how does the <invocationCount> attribute call a method multiple times

```
@Test(invocationCount = 2)

    public void testCase1() {

        System.out.println("testCase1");

    }
```

2. TestNG example using @Test(invocationCount=?, threadPoolSize=?) attribute.

```
@Test(invocationCount = 2,threadPoolSize = 2)

    public void testCase1() {

        System.out.printf("Thread Id : %s is
started!\n", Thread.currentThread().getId());

        System.out.println("testCase1");

    }
```

In this TestNG example, we are using both the attributes together while keeping their value same. It'll let each test method call run on a separate thread.

We've added thread ID and timestamp in the log messages so that you can identify the thread and the execution time of the test running.

12. How to prepare customized html reports using testng?

Answer: We will learn to create a custom emailable report.

We need to implement an IReporter interface to create a custom TestNG Report. So, if you implement IReporter by any Java class then you need to override the unimplemented method as per the requirement to display data in the custom report, which is as below:

```

package Test;

import java.util.List;

import org.testng.IReporter;

import org.testng.ISuite;

import org.testng.xml.XmlSuite;

public class TestReporterClass implements IReporter{

    public void generateReport(List<XmlSuite> xmlSuites, List<ISuite> suites, String
outputDirectory) {

        // TODO Auto-generated method stub

    }

}

```

STEPS TO CREATE CUSTOM TESTNG HTML REPORT IN SELENIUM

Here are the steps which you need to implement to create a custom HTML report in TestNG.

STEP# 1: IMPLEMENT IREPORTER AND OVERRIDE THE UNIMPLEMENTED METHOD

Above sample code has the unimplemented method. We will first take its second argument, that is, List<ISuite> suites. We will iterate through the ISuite which is basically iteration over the entire test suite.

Once we defined the iteration through for each loop then we put all the test results (consisting passed and failed tests) of the single suite inside the Map. We further get the key from the result map. This key will help further to find failed and passed

```
Map<String, ISuiteResult> resultMap = ist.getResults();
```

```
Set<String> key = resultMap.keySet();
```

We further use iteration over the keys and we identify the Context objects of the result.

```
ITestContext cntx = resultMap.get(k).getTestContext();
```

Context help us further to get the map for failed or passed tests, which further gives their respective methods name.

```
IResultMap failedTest = cntx.getFailedTests();
```

```
Collection<ITestNGMethod> failedMethods = failedTest.getAllMethods();
```

Finally, we get custom details of the execution in the console and custom TestNG report as well.

16. When to use dataprovider testng anotation?

Answer: TestNG @DataProvider – Test parameters example. An important features provided by TestNG is the testng DataProvider feature. It helps you to write data-driven tests which essentially mean that same test method can be run multiple times with different data-sets.

17. What is TestNG Assert and list out some common assertions supported by TestNG?

Answer: Assertions in TestNG are a way to verify that the expected result and the actual result matched or not. If we could decide the outcome on different small methods using assertions in our test case, we can determine whether our test failed or passed overall. An example of assertion can be logging into the website, checking the title of the webpage, verifying the functionality of an input box that takes only integers, etc.

We should remember that an assertion in TestNG is successful only if there are no exceptions thrown during the test case execution. TestNG asserts (or assertions) popularly validate the results in TestNG using selenium.

Please refer the below link

<https://www.automationtestinginsider.com/2020/03/assertions-in-testng.html>

18. How to create and run TestNG.xml?

Answer: What is testng.xml?

testng. xml file is a configuration file in TestNG.

It is used to define test suites and tests. It provides different options to include packages, classes and independent test methods in our test suite. It also allows us to configure multiple tests in a single test suite and run them in multi threaded environment.

Importance of testng.xml

In TestNG, you can define multiple test cases in a single class whereas, in Java, you can define only one test in a single class in the main() method. In Java, if you want to create one more test, then you need to create another java file and define the test in the main() method.

Instead of creating test cases in different classes, we recommend you to use TestNG framework that allows you to create multiple test cases in a single class.

Please refer complete details about testing.xml here:

<https://www.automationtestinginsider.com/2020/03/in-this-post-we-will-discuss-about.html>

19. What is parameterized testing in TestNG?

Answer: Parameterization [Data driven test] is an execution strategy, which allows us to run a test case automatically, multiple times with different input values.

To pass multiple data to the application at runtime, we need to parameterize our test scripts.

There are two ways by which we can achieve parameterization in TestNG

1. With the help of Parameters annotation and TestNG XML file.
2. with the help of DataProvider annotation.

Please refer the below link for complete details and program

<https://www.automationtestinginsider.com/2020/03/parameterization-using-testng.html>

20. How to run a group of test cases using TestNG?

Answer: TestNG allows you to perform ordered groupings of test methods. You can not only declare that methods belong to groups, but you can also specify groups that contain other groups. Then TestNG can be invoked and asked to include a certain set of groups while excluding another set. This gives you maximum flexibility in how you partition your tests and doesn't require you to recompile anything if you want to run two different sets of tests back to back.

Groups are specified in your testng.xml file and can be found either under the <test> or <suite> tag. Groups specified in the <suite> tag apply to all the <test> tags underneath.

```
@Test(groups = { "smokeTest", "functionalTest" })
```

```
public void loginTest(){
```

```
System.out.println("Logged in successfully");
```

```
}
```

21. What is the use of @Listener annotation in TestNG?

Answer:

What is Listeners in TestNG?

Listener is defined as interface that modifies the default TestNG's behavior. As the name suggests Listeners "listen" to the event defined in the selenium script and behave accordingly. It is used in selenium by implementing Listeners Interface. It allows customizing TestNG reports or logs. There are many types of TestNG listeners available.

There are many types of listeners which allows you to change the TestNG's behavior.

Below are the few TestNG listeners:

- IAnnotationTransformer ,
- IAnnotationTransformer2 ,
- IConfigurable ,
- IConfigurationListener ,
- IExecutionListener,

- IHookable ,
- IInvokedMethodListener ,
- IInvokedMethodListener2 ,
- IMethodInterceptor ,
- IReporter,
- ISuiteListener,
- ITestListener

Above Interface are called TestNG Listeners. These interfaces are used in selenium to generate logs or customize the TestNG reports.

22. How can we create a data driven framework using TestNG?

Answer: Please refer answer of question #19

23. How to pass parameters through testng.xml to a test case?

Answer: Please refer answer of question #19

24. Explain DataProviders in TestNG using an example

Answer: **Answer:** Please refer answer of question #19

Please refer the below links as well:

<https://www.automationtestinginsider.com/2020/06/data-driven-framework-part-1.html>

<https://www.automationtestinginsider.com/2020/06/data-driven-framework-part-2.html>

25. Can I call a single data provider method for multiple functions and classes?

Answer: Yes, the same DataProvider can be used in multiple functions and classes by declaring DataProvider in separate class and then reusing it in multiple classes.

26. How to skip a @Test method or a code block in TestNG?

Answer:

There are two ways you can skip Test Method

1. Using @Ignore at test level

```
public class IgoneTestClassExample {

    @Ignore("not yet ready , Please ignore.")

    @Test

    public void testCase1() {

        System.out.println("testCase1");
    }
}
```

```

    }

    @Test

    public void testCase2() {

        System.out.println("testCase2");

    }

}

```

Output:

testCase2

2. Using enabled = false at test level

```

public class SkipTestClassExample
{

    @Test

    public void testCase1() {

        System.out.println("testCase1");

    }

    @Test(enabled = false)

    public void testCase2() {

        System.out.println("testCase2");

    }

}

```

Output:

testCase1

27. What is Soft Assertion in selenium and how can you mark a test case as failed by using soft assertion?

Answer: Soft Assertions (Verify):

It is a custom assert mechanism supported by TestNG's "org.testng.asserts.Softassert" package. We use it when a test has to continue execution even after an assertion fails in the sequence.

Soft Assert does not throw an exception when an assert fails and would continue with the next step after the assert statement.

If there is any exception and you want to throw it then you need to use `assertAll()` method as a last statement in the `@Test` and test suite again continue with next `@Test` as it is.

28. Explain what is a Group Test in TestNG?

Answer: TestNG Groups allow you to perform groupings of different test methods. Grouping of test methods is required when you want to access the test methods of different classes.

Not only you can declare the methods within a specified group, you can also declare another group within a specified group. Thus, TestNG can be asked to include a certain set of groups while excluding another set of groups.

It provides you maximum flexibility by partitioning your test methods in groups and does not require recompilation of test cases if you run your two different sets of test cases back to back.

Groups are specified in the `testng.xml` file with `<groups>` tag. Groups can be specified either in the `<suite>` tag or `<test>` tag. If the `<groups>` tag is specified inside the `<suite>` tag, then it is applied to all the `<test>` tags of XML file. If the `<groups>` tag is specified within a particular `<test>` folder, then it is applied to that particular `<test>` tag only.

First case: When `<groups>` tag is defined inside the `<suite>` tag.

```
<suite name="test_suite">

<groups>

<run>

<include name="SmokeTest"/>

</run>

</groups>
```

Second case: When `<groups>` tag is defined inside the `<test>` tag.

```
<suite name="test_suite">

<test name="Loan">

<groups>

<run>

<include name="SmokeTest"/>

</run>
```

```

</groups>

<classes>

<class name="com.javatpoint.Personal_loan"/>

<class name="com.javatpoint.Home_loan"/>

<class name="com.javatpoint.Car_loan"/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

```

29. What is the difference between @Factory and @DataProvider annotation?

Answer: The annotation like @Factory and @DataProvider are mainly used to reiterate the same test class with different test data. These annotations will help the user to use the same class seamlessly without duplicating the test class code.

@DataProvider Annotation

A test method that uses DataProvider will be executed multiple numbers of times based on the data provided by the DataProvider. That means this annotation parametrizes the particular test method and executes the test number of times based on the data provided by the DataProvider method.

The condition that needs to be met here is that the method marked as @DataProvider must return a 2D Object array (Object[][]) where each Object[] will be used as the input parameter to an iteration of the test method which uses the data provider.

The Test method will be executed using the same instance of the test class to which the test method belongs.

@Factory Annotation

It can be used to execute all the test methods present inside a test class with multiple sets of data, using the separate instance of the class.

Using this, we can instantiate a class multiple times rather than just a method.

The factory method should return an Object[]. This can be an array of Method calls or class objects.

The Test method will be executed using the separate instance of the respective class.

Example:

Using @DataProvider

```

@Test(dataProvider = "dp1")

    public void testMethod(String param) {

        System.out.println("The parameter value is: " + param);

    }

@DataProvider(name = "dp1")

    public Object[][] dataMethod() {

        return new Object[][] { { "one" }, { "two" } };

    }

}

```

Using @Factory

The SimpleTest class contains the testMethod() and beforeClass() methods.

The constructor of the test class takes a String argument value. Both beforeClass() and testMethod() print a message onto console.

```

public class SimpleTest

{

    private String param = "";

    public SimpleTest(String param) {

        this.param = param;

    }

    @BeforeClass

    public void beforeClass() {

        System.out.println("Before SimpleTest class executed.");

    }

    @Test

    public void testMethod() {

```

```

        System.out.println("testMethod parameter value is: " +
param);

    }

}

public class SimpleTestFactory

{

    @Factory

    public Object[] factoryMethod() {

        return new Object[] {

                                new SimpleTest("one"),

                                new SimpleTest("two")

                                };

    }

}

```

30. What is the difference between @BeforeMethod and @BeforeClass ?

Answer: @BeforeMethod: The annotated method will be run before each test method.
 @BeforeClass: The annotated method will be run before the first test method in the current class is invoked. @BeforeTest: The annotated method will be run before any test method belonging to the classes inside the <test> tag is run.

31. What are the different attributes for @Test annotation?

Answer: Please refer below link to understand different attributes in testing

<https://www.automationtestinginsider.com/2020/03/different-attributes-in-testng.html>

32. How can we run test cases in parallel using TestNG?

Answer: Please refer below link to understand to run test cases in parallel in testing

<https://www.automationtestinginsider.com/2020/03/parallel-testing-in-testng.html>

33. Suppose I want to check a particular exception in TestNG. How will you check?

Answer: TestNG provides an option of tracing the exception handling of code. You can test whether a code throws a desired exception or not. Here the `expectedExceptions` parameter is used along with the `@Test` annotation. Now, let's see `@Test(expectedExceptions)` in action.

```
public class MessageUtil {

    private String message;

    //Constructor

    //@param message to be printed

    public MessageUtil(String message) {

        this.message = message;

    }

    // prints the message

    public void printMessage() {

        System.out.println(message);

        int a = 0;

        int b = 1/a;

    }

}

import org.testng.annotations.Test;

public class ExpectedExceptionTest {

    String message = "ExceptionTest";

    MessageUtil messageUtil = new MessageUtil(message);

    @Test(expectedExceptions = ArithmeticException.class)

    public void testPrintMessage() {

        System.out.println("Inside testPrintMessage()");

        messageUtil.printMessage();

    }

}
```

34. How Cross Browser testing is handled in Selenium?

What is Cross Browser Testing?

Cross Browser Testing is a type of functional test to check that your web application works as expected in different browsers?

Why do we need?

A web application can be opened in any browser by the end user. For example, some people prefer to open in Firefox, some in chrome, ie etc. We need to ensure that the web application will work as expected in all popular browsers so that more people can access it and use it.

This motive can be fulfilled with Cross Browser testing of the product.

What is Cross Browser Testing?

Cross Browser Testing is a type of functional test to check that your web application works as expected in different browsers..

Why do we need?

A web application can be opened in any browser by the end user. For example, some people prefer to open in Firefox, some in chrome, ie etc. We need to ensure that the web application will work as expected in all popular browsers so that more people can access it and use it.

This motive can be fulfilled with Cross Browser Testing of the product.

Some Cross Browser Issues

- Font size mismatch in different browsers.
- JavaScript implementation can be different.
- CSS,HTML validation difference can be there.
- Some browser still not supporting HTML5.
- Page alignment and div size.
- Image orientation.
- Browser incompatibility with OS.

How to Achieve Cross Browser in Selenium?

To execute test cases with different browsers in the same machine at the same time we can integrate TestNG framework with Selenium WebDriver.

Example: We have created a class CrossBrowserDemo and we have a test method launchApp in it. Using @Parameters annotation we are passing browser name(as string) to test method. And based on the browser parameter our test method launchApp will be executed on particular browser.

Our xml looks like below: Inside <suite> tag we have <parameter name="browser" value="Firefox"></parameter>

Running test on single browser

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite" parallel="tests">
  <test name="FireFoxTest">
    <parameter name="browser" value="Firefox"></parameter>
```



```

    <classes>
      <class name="com.CrossBrowserTest.CrossBrowserDemo"></class>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->

```

Running test on multiple browser

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite" parallel="tests">
  <test name="FireFoxTest">
    <parameter name="browser" value="Firefox"></parameter>
    <classes>
      <class name="com.CrossBrowserTest.CrossBrowserDemo"></class>
    </classes>
  </test> <!-- Test -->
  <test name="ChromeTest">
    <parameter name="browser" value="Chrome"></parameter>
    <classes>
      <class name="com.CrossBrowserTest.CrossBrowserDemo"></class>
    </classes>
  </test> <!-- Test -->
  <test name="IETest">
    <parameter name="browser" value="IE"></parameter>
    <classes>
      <class name="com.CrossBrowserTest.CrossBrowserDemo"></class>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->

```

Test Class

```

public class CrossBrowserDemo {

    WebDriver driver;

    @Parameters("browser")

    @Test

    public void launchApp(String browser) throws InterruptedException {
        if (browser.equalsIgnoreCase("Chrome")) {

            System.setProperty("webdriver.chrome.driver",

"C:\\Users\\Hitendra\\Downloads\\chromedriver win32\\chromedriver.exe");

            driver = new ChromeDriver();

        } else if (browser.equalsIgnoreCase("Firefox")) {

```

```

        System.setProperty("webdriver.gecko.driver",

            "C:\\Users\\Hitendra\\Downloads\\geckodriver-v0.26.0-
win64\\geckodriver.exe");

        driver = new FirefoxDriver();

    } else if (browser.equalsIgnoreCase("IE")) {

        System.setProperty("webdriver.ie.driver",

            "C:\\Users\\Hitendra\\Downloads\\IEDriverServer_x64_3.14.0\\IEDriverServer.
exe");

        driver = new InternetExplorerDriver();

    }

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

    driver.get("https://www.automationtestinginsider.com/");

    Thread.sleep(2000);

    driver.close();

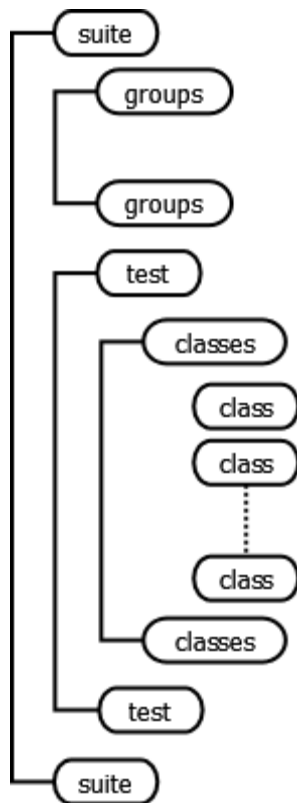
}

}

```

35. Explain the structure of testng.xml file?

Answer: please refer the below image for basic structure of testng.xml



36. What are the different methods of Assert?

Answer: All the assertions are in the Assert class.

```
public class Assert extends java.lang.Object
```

This class provides a set of assertion methods, useful for writing tests. Only failed assertions are recorded. Some of the important methods of Assert class are as follows –

- void assertEquals(boolean expected, boolean actual)- Checks that two primitives/objects are equal.
- void assertTrue(boolean condition)- Checks that a condition is true.
- void assertFalse(boolean condition) - Checks that a condition is false.
- void assertNotNull(Object object) - Checks that an object isn't null.
- void assertNull(Object object) - Checks that an object is null.
- void assertSame(object1, object2) - The assertSame() method tests if two object references point to the same object.
- void assertNotSame(object1, object2) - The assertNotSame() method tests if two object references do not point to the same object.
- void assertEquals(expectedArray, resultArray); - The assertEquals() method will test whether two arrays are equal to each other.

37. What the difference between include and exclude in TestNG?

Answer: TestNg provides an option to include or exclude Groups, Test Methods, Classes and Packages using include and exclude tags by defining in testng.xml.

We will create a Class with three Test Methods. In that we will include two test methods and try to exclude one test method.

```
import org.testng.annotations.Test;

public class AddTestCase {

    @Test

    public void addLocationTestCase() {

        System.out.println("Im in add
location test case");

    }

    @Test

    public void addDepartmentTestCase() {

        System.out.println("Im in add
department test case");

    }

    @Test

    public void addEmployeeTestCase() {

        System.out.println("Im in add
employee test case");

    }

}
```

In the above class example, we have created three test methods, 'addLocationTestCase', 'addDepartmentTestCase', and

'addEmployeeTestCase'.

In the below testng.xml file we will exclude 'addEmployeeTestCase' and try to execute the program. We need to first add the class name and in that class, we need to define the methods which needs to be included and excluded

```
!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >

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

    <test name="Method Test Cases" >

        <classes>
```

```

<class name="com.easy.entry.AddTestCase">

    <methods>

        <include name="addLocationTestCase" />

        <include name="addDepartmentTestCase" />

        <exclude name="addEmployeeTestCase" />

    </methods>

</class>

</classes>

</test>

</suite>

```

After running the above testng.xml file, we will get the output as It will just run the test methods which are included in the class. And will not execute the test methods which are in excluded.

38. How to execute the single selected method in TestNG?

Answer: To include only a particular @Test method to run like below:

```

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

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

<suite name="Suite">

    <test name="Test">

        <classes>

            <class name = "Full path to the Test class" />

            <methods>

                <include name="testMethodName" />

            </methods>

        </classes>

    </test>

```

</suite> <!-- Suite -->

40. Tell me a login page script in TestNG?

Answer:

```
import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.Assert;

import org.testng.annotations.Test;

public class LoginUsingSelenium {

    @Test public void login() {

        // TODO Auto-generated method stub

        System.setProperty("webdriver.chrome.driver", "path of driver");

        WebDriver driver=new ChromeDriver(); driver.manage().window().maximize();

        driver.get("https://www.linkedin.com/login");

        WebElement username=driver.findElement(By.id("username"));

        WebElement password=driver.findElement(By.id("password"));

        WebElement login=driver.findElement(By.xpath("//button[text()='Sign in']"));

        username.sendKeys("example@gmail.com"); password.sendKeys("password");

        login.click(); String actualUrl="https://www.linkedin.com/feed/";

        String expectedUrl= driver.getCurrentUrl();

        Assert.assertEquals(expectedUrl,actualUrl);

    }

}
```

41. What is the difference between @Parameters and @DataProviders in TestNG?

Answer: DataProviders pass the different parameters on a single test in a single execution, whereas parameters pass the parameters just once per execution in TestNG.

42. How to create Suites in TestNG?

Answer: Please refer **Different ways to setup testng.xml** section in below link:

<https://www.automationtestinginsider.com/2020/03/in-this-post-we-will-discuss-about.html>

43. How to prioritize the tests in TestNG at Class level and Suite level?

Answer:

1. at Test Method Level:

Here you have to consider 3 points:

If you are not using any priority in your test method then TestNG assign by default priority=0 to the Test Method and test method will be executed in alphabetical order

```
public class ClassA {  
  
    @Test  
  
    public void testC() {  
  
        System.out.println("ClassC");  
  
    }  
  
    @Test  
  
    public void testA() {  
  
        System.out.println("ClassA");  
  
    }  
  
    @Test  
  
    public void testB() {  
  
        System.out.println("ClassB");  
  
    }  
}
```

```
}
```

Output:

ClassA

ClassB

ClassC

If there is same priority assign to test methods then execution order will be alphabetically.

```
public class ClassA {  
  
    @Test(priority=1)  
  
    public void testC() {  
  
        System.out.println("ClassC");  
  
    }  
  
    @Test(priority=1)  
  
    public void testA() {  
  
        System.out.println("ClassA");  
  
    }  
  
    @Test  
  
    public void testB() {  
  
        System.out.println("ClassB");  
  
    }  
  
}
```

Output:

ClassB

ClassA

ClassC

In Case of different priority, execution will be from minimum to maximum priority:

```
public class ClassA {  
  
    @Test(priority=-1)  
  
    public void testC() {  
  
        System.out.println("ClassC");  
  
    }  
  
    @Test(priority=1)  
  
    public void testA() {  
  
        System.out.println("ClassA");  
  
    }  
  
    @Test(priority=0)  
  
    public void testB() {  
  
        System.out.println("ClassB");  
  
    }  
  
}
```

Output:

ClassC

ClassB

ClassA

2. At Suite Level: classes will be executed in the order as they defined

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

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

```
<suite name="Suite">
```

```

<test thread-count="5" name="Test">

    <classes>

        <class name="testNG.ClassB"/>

        <class name="testNG.ClassA"/>

    </classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

```

In the above example ClassB will be executed first and then ClassA

44. How to create Group of Groups in TestNG?

Answer: We have below test class which has three test methods, which belongs to different groups.

```

import org.testng.annotations.Test;

public class Groups {

    @Test(groups = { "Sanity" })

    public void testcase1() {

        System.out.println("Test case belonging to
Sanity");

    }

    @Test(groups = { "Sanity", "Smoke" })

    public void testcase2() {

        System.out.println("Test case belonging to
both Sanity and Smoke");

    }

    @Test(groups = { "Regression" })

    public void testcase3() {

        System.out.println("Test case belonging to
Regression");

    }
}

```

```
}
```

Now let's say if we want to execute all the groups at once. Refer below testing.xml

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

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

<suite name="Suite">

    <test thread-count="5" name="Test">

        <groups>

            <run>

                <include
name="Sanity"></include>

                <include
name="Smoke"></include>

                <include
name="Regression"></include>

            </run>

        </groups>

        <classes>

            <class name="testNG.Groups"

        />

        </classes>

    </test> <!-- Test -->

</suite> <!-- Suite -->
```

Output:

Test case belonging to Sanity

Test case belonging to both Sanity and Smoke

Test case belonging to Regression

45. How to exclude a particular group from a test case execution using TestNG?

Answer: we can take the help of above example. Suppose we want to remove **Regression** group.

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

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

<suite name="Suite">

    <test thread-count="5" name="Test">

        <groups>

            <run>

                <include

name="Sanity"></include>

                <include

name="Smoke"></include>

                <exclude

name="Regression"></exclude>

            </run>

        </groups>

        <classes>

            <class name="testNG.Groups"

/>

        </classes>

    </test> <!-- Test -->

</suite> <!-- Suite -->
```

Output:

Test case belonging to Sanity

Test case belonging to both Sanity and Smoke

46. How to write regular expressions in testng.xml file to search @Test methods containing “smoke” keyword?

Answer: Regular expression to find @Test methods containing keyword “smoke” is as mentioned below.

```
<methods>
```

```
<include name=".*smoke.*"/>

</methods>
```

47. What is the time unit we specify in test suites and test cases?

Answer: We specify the time unit in test suites and test cases is in milliseconds.

48. List out various ways in which TestNG can be invoked?

Answer: TestNG can be invoked in the following ways

- Using Eclipse IDE
- Using ant build tool
- From the command line
- Using IntelliJ's IDEA

49. How to run TestNG using command prompt?

Answer:

Step 1: Open notepad

Step 2: Paste the below lines of code - You may need to add your project location. In the example, project location is set as 'F:\Selenium\TestNGBatchExample'.

Step 3: Save the file as 'testNGBatchFile.bat' in location that you want to save.

```
set projectLocation=F:\Selenium\TestNGBatchExample
```

```
cd %projectLocation%
```

```
set classpath=%projectLocation%\bin;%projectLocation%\lib\*
```

```
java org.testng.TestNG %projectLocation%\testng.xml
```

```
pause
```

50. What is the use of @Test(invocationCount=x)?

Answer: The invocationcount attribute tells how many times TestNG should run a test method

```
@Test(invocationCount = 5)
```

```
public void testCase1(){
```

```
}
```

51. What is the use of @Test(threadPoolSize=x)?

Answer: The threadPoolSize attribute tells to form a thread pool to run the test method through multiple threads.

Note: This attribute is ignored if invocationCount is not specified

```
@Test(threadPoolSize = 3, invocationCount = 10)

public void testCase1(){

}
```

In this example, the method testCase1 will be invoked from three different threads

52. What does the test timeout mean in TestNG?

Answer: The maximum number of milliseconds a test case should take.

```
@Test(threadPoolSize = 3, invocationCount = 10, timeOut = 10000)

public void testCase1(){

}
```

In this example, the function testCase1 will be invoked ten times from three different threads.

Additionally, a time-out of ten seconds guarantees that none of the threads will block on this thread forever.

53. Is it possible to pass test data through testng.xml file, if yes how?

Answer: TestNG allows the user to pass values to test methods as arguments by using parameter annotations through testng. xml file. Sometimes it may be required for us to pass values to test methods during run time. ... The @Parameters annotation can be placed on any method that has a @Test, @Before/After or @Factory annotation.

54. Do you run test cases in parallel with TestNG? If yes how many threads and does it cause any problem?

Answer: TestNG provides multiple ways to execute tests in separate threads. In testng.xml, if we set 'parallel' attribute on the tag to 'tests', testNG will run all the '@Test' methods in tag in the same thread, but each tag will be in a separate thread.

If we want to run methods/classes in separate threads, we need to set 'parallel' attribute on the tag to 'methods' / 'classes'

This helps us to run test methods / classes / tests in parallel. By using parallel execution, we can reduce the 'execution time' as tests are 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.

55. What's TestNG Listener Class & why do we use it?

Answer: TestNG Listeners also allows you to customize the tests logs or report according to your project requirements.

TestNG Listeners in Selenium WebDriver are modules that listens to certain events and keep track of test execution while performing some action at every stage of test execution.

TestNG Listeners in Selenium WebDriver can be implemented at two levels:

Class level: In this, you implement listeners for each particular class no matter how much test cases it includes.

Suite level: In this, you implement listeners for a particular suite which includes several classes as test cases.

There are numerous TestNG listeners in Selenium WebDriver, some of them are used very frequently by the testing community & some are almost forgotten. In this TestNG tutorial, I will demonstrate the most popular TestNG listeners with examples but before that, let me enlist the various TestNG listeners in Selenium WebDriver.

- ITestListener
- IAnnotationTransformer
- IInvokedMethodListener
- ISuiteListener
- IReporter
- IConfigurable
- IExecutionListener
- IHookable
- IMethodInterceptor
- IConfigurationListener

56. How will you get the browser values from testng?

Answer: Please refer below link

<https://www.automationtestinginsider.com/2020/03/cross-browser-testing-in-testng.html>

57. Can we run testNG class code without using any TestNg annotation?

Answer: annotations belong to what is called reflection and meta-programming. Also, it's not necessary to have main() method in your tests, but you can use main() method to run the TestNg tests if you want.

58. How to Install TestNG In Eclipse? How do you verify that TestNg Is Installed properly In Eclipse?

Answer: Please refer below link

<https://www.automationtestinginsider.com/2020/03/testng-part1-introduction-to-testng.html>

59. What is Error Collector in TestNG? What is its use?

Answer: This class allows the collection of errors during the process of retrieving the test data for the test method parameters.

60. Detail about TestNG Test Output folder.?

Answer: By default the report files (HTML & XML) are written to a folder named test-output under your workspace. Netbeans however overrides this location. It places output to build/test/results folder. Please re-run the TestNG test suite and watch results folder. All required files will be generated there.

61. What is the difference between WebDriver Listeners and TestNG Listeners?

Answer: Listeners are those which will be listening to any state change (event).

So in case of WebDriver, whenever anyone clicks on a button on a web page, then automatically a method will be called (actionPerformed() in ActionListener case).

But in case of TestNG, there are the actual "listeners" which listen (here the method gets called automatically) to the test execution events. A few examples are: onStart(), beforeStart(), afterFinish(), and onFinish() etc. Mostly, the TestNG automation teams implement their own custom listeners for custom Logging and reporting.

WebDriver listeners too do a similar job...of logging and/or reporting. But then both of them work on different event sets. WebDriver works on different automation events whereas TestNG works on different test's related events. The point to note is that, with WebDriver listeners, "Logging" happens before/after event occurrence.

62. Explain how does TestNG allow you to state dependencies with an example?

Answer: Sometimes, you may need to invoke methods in a test case in a certain order. Here comes TestNG Dependencies into the picture. TestNG allows you to specify dependencies either with annotations or in XML.

TestNG allows you to specify dependencies either with:

Using attribute dependsOnMethods in @Test annotations, OR.

Using attribute dependsOnGroups in @Test annotations.

1.Let's see first dependsOnMethods in @Test annotations

```
public class DependsOnMethodsTestCase {
```

```
@Test
```

```
public void testCase1(){
```

```
System.out.println("Test Case 1");
```

```
}
```

```
@Test
```

```
public void testCase2(){
```

```
System.out.println("Test Case 2");
```


}

}

testng.xml

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

```
<suite name="testdependsOn">
```

```
<test name="testngTest">
```

```
<classes>
```

```
<class name="com.test.DependsOnMethodsTestCase" />
```

```
</classes>
```

```
</test>
```

```
</suite>
```

Output

Test Case 1

Test Case 2

Now we add the dependsOnMethods attribute to the @Test Annotations and execute the same program.

```
public class DependsOnMethodsTestCase {
```

```
@Test(dependsOnMethods = {"testCase2"})
```

```
public void testCase1(){
```

```
System.out.println("Test Case 1");
```

```
}
```

```
@Test
```

```
public void testCase2(){
```

```
System.out.println("Test Case 2");
```

```
}
```

```
}
```

Execute the same testng.xml which was placed above and see the difference in Console Output

Output

Test Case 2

Test Case 1

2. Let's see Dependencies with XML:

```
public class DependsOnMethodsTestCase {  
  
    @Test(groups = {"FirstGroup"})  
  
    public void testCase1(){  
  
        System.out.println("Test Case 1");  
  
    }  
  
    @Test(groups = {"SecondGroup"})  
  
    public void testCase2(){  
  
        System.out.println("Test Case 2");  
  
    }  
  
}
```

testng.xml

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >  
  
<suite name=" testdependsOn ">  
  
    <test name="testngTest">  
  
        <groups>  
  
            <dependencies>  
  
                <group name="FirstGroup" depends-on="SecondGroup"></group>  
  
            </dependencies>  
  
        </groups>  
  
        <classes>  
  
            <class name="com.test.DependsOnMethodsTestCase" />
```

</classes>

</test>

</suite>

Output

Test Case 2

Test Case 1

63. Sequence of execution of below annotations: @Test @BeforeGroups @AfterGroups @BeforeSuite @AfterSuite @BeforeMethod @AfterMethod @BeforeClass @AfterClass?

Answer: Please refer below link

<https://www.automationtestinginsider.com/2020/03/testng-annotations-and-their-execution.html>

64. Explain execution seq-- @Test(priority=1) @Test(priority=2) @Test(priority=0) @Test(priority=-1)

Answer: priority sequence would be: -1, 0, 1, 2.

65. Explain execution seq-- @Test(priority=1) @Test(priority=2) @Test()

Answer: If priority not assigned then automatically it will assigned 0 priority, hence the sequence would be: 0, 1, 2.

At times, test cases may fail while running automated test scripts. The reason may be anything (say, Network issue, System issue or browser issue) but as an automation tester, you need to analyze the result and need to execute the test scripts again. Here is a solution to run failed test cases using TestNG in Selenium.

Why Test fails:

The reason for the failures can be anything

- Application Failure – not getting expected output
- Network issue – wifi, LAN issue etc.
- Server is not responding
- Scripting issue – Locator change due to new functionality
- Application is down
- Browser/browser driver issue

How to execute failed test cases?

We can execute failed test cases using two methods

- By running “testng-failed.xml”
- By Implementing TestNG IRetryAnalyzer interface

Lets have a look the first method:

1. By running “testng-failed.xml”

We have created two classes ClassA and ClassB. Each class has two test methods of which we are deliberately failing one of the tests.

```
public class ClassA {

    @Test

    public void testCase1() {

        System.out.println("this is testcase1");

        Assert.assertTrue(false);

    }

    @Test

    public void testCase2() {

        System.out.println("this is testcase2");

        Assert.assertTrue(true);

    }

}

public class ClassB {

    @Test

    public void testCase3() {

        System.out.println("this is testcase3");

        Assert.assertTrue(true);

    }

    @Test

    public void testCase4() {

        System.out.println("this is testcase4");

        Assert.assertTrue(false);

    }

}
```

testng.xml

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

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

<suite name="Smoke Suite">
```

```

<test name="Test">

    <classes>

        <class name="Com.runFailedTestCases1.ClassA"/>

        <class name="Com.runFailedTestCases1.ClassB"/>

    </classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

```

Output

```

this is testcase1
this is testcase2
this is testcase3
this is testcase4

```

```

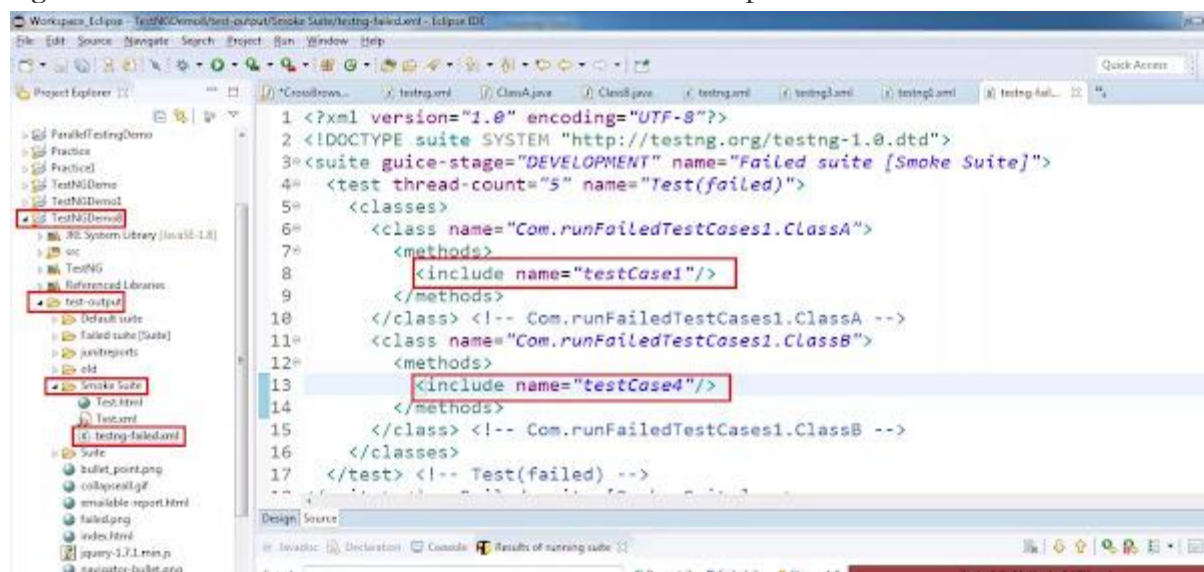
=====
Suite
Total tests  run: 4, Failures: 2, Skips: 0
=====

```

If you want to execute only failed test cases through the Eclipse, then first refresh the project. Now navigate to test-output folder and then navigate to Suite folder (in our case suite name is Smoke Suite) Expand that, you will be able to see testng-failed.xml

Open the xml and you will see failed test cases in xml file.

Right-click on this file and click on run as and select the option called "testNG suite".



testng-failed.xml

Suppose if you have three test cases if all the test cases are executed successfully means you are not able to see this folder under the test-output folder. This folder will appear only when

one of the test case is failed. Then run this file, it will going to run only failed test cases.

You can create Runner file to execute your failed test cases through script. Please refer below program.

```
public class FailTestRunner {

    @AfterTest

    public void runFailTestCases() {

        TestNG obj= new TestNG();

        List<String> list= new ArrayList<String>();

        list.add("D:\\Workspace_Eclipse\\TestNGDemo8\\test-output\\Suite\\testng-
failed.xml");

        obj.setTestSuites(list);

        obj.run();
    }
}
```

2. By Implementing TestNG IRetryAnalyzer interface

Create a class to implement IRetryAnalyzer. Here I am creating a class (say, RetryAnalyzerTest) and implementing IRetryAnalyzer.

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

public class RetryAnalyzerTest implements IRetryAnalyzer {

    int count=0;

    int maxCount=2;

    public boolean retry(ITestResult result) {

        if(count<maxCount) {

            System.out.println("Retrying " +result.getName()

+" again and count is " +(count+1));

            count++;

            return true;
        }
        return false;
    }
}
```

```
}
```

Now create another class 'RetryListnerClass' by Implementing 'IAnnotationTransaformer' interface. transform method is called for every test during test run. A simple implementation of this 'IAnnotationTransformer' interface can help us set the 'setRetryAnalyzer' for 'ITestAnnotation'. Add the above class name (RetryAnalyzerTest.class) in the below program. This interface does its work in run time by adding annotation to the test methods.

```
import java.lang.reflect.Constructor;
import java.lang.reflect.Method;
import org.testng.IAnnotationTransformer;
import org.testng.annotations.ITestAnnotation;

public class RetryListnerClass implements IAnnotationTransformer {

    public void transform(ITestAnnotation annotation, Class testClass,

        Constructor testConstructor, Method testMethod) {

        annotation.setRetryAnalyzer(RetryAnalyzerTest.class);
    }
}
```

Let us see the example by executing simple tests below

```
public class ClassA {

    @Test()

    public void testCase1() {
        System.out.println("this is testcase1");
        Assert.assertTrue(true);
    }

    @Test
    public void testCase2() {
        System.out.println("this is testcase2");
        Assert.assertTrue(false);
    }
}
```

We have two test cases in above program and one of the test case (testCase2) is getting failed. First lets include below mentioned Listener to testng.xml file. Below mentioned syntax is to add Listener for RetryListnereClass.

```
<listeners>
<listener class-name="method2.RetryListnerClass"></listener>
</listeners>
```

Final testng.xml file should looks like below:

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

```

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

<suite name="Suite">

<listeners>

<listener class-name="method2.RetryListnerClass"></listener>

</listeners>

  <test name="Test">

    <classes>

      <class name="method2.ClassA"/>

    </classes>

  </test> <!-- Test -->

</suite> <!-- Suite -->

```

Execute the testng.xml. Here is the output which I got. You could see in the below mentioned result that the Test 2 is executed three times as we have mentioned 'maxCount = 2'. Even though we have just 2 tests, we could find total test runs are 4 in the result.

Output:

```

this is testCase1
this is testCase2
Retrying testCase2 again and count is 1
this is testCase2
Retrying testCase2 again and count is 2
this is testCase2
=====
Suite

Total tests run: 4, Failures: 1, Skips: 2
=====

```


GIT

1. What is Git? What is the difference between Git and GitHub?

Answer: Git is a version control system that lets you manage and keep track of your source code history. GitHub is a cloud-based hosting service that lets you manage Git repositories. If you have open-source projects that use Git, then GitHub is designed to help you better manage them.

2. What is the advantage of using GitHub for Selenium?

Answer: GitHub is a cloud-based hosting service that lets you manage Git repositories; it helps to have a backup code in case of physical failures.

- github supports branching, so we can have multiple versions of code.
- github supports project cloning, so it helps in easily distribution of project across multiple teams and multiple locations
- github supports code pull so anyone with access rights can pull code in local machine. This can also be integrated with jenkins.
- github supports code push so anyone with access rights can checkin code in github central repository.

3. How to handle git conflicts?

Answer: Git can handle on its own most merges by using its automatic merging features. There arises a conflict when two separate branches have made edits to the same line in a file, or when a file has been deleted in one branch but edited in the other. Conflicts are most likely to happen when working in a team environment.

- Identify the files that have caused the conflict.
- Make the necessary changes in the files so that conflict does not arise again.
- Add these files by the command git add.
- Finally to commit the changed file using the command git commit

Please refer below YouTube video link

<https://www.youtube.com/watch?v=CQqdTDBlogp&feature=youtu.be>

4. Explain different Git commands?

Answer: Below are the most git commands

1. Initialize a repo

Create an empty git repo or re-initialize an existing one

```
$ git init [repository path]
```

2. git remote add [variable name] [Remote Server Link]

This command is used to connect your local repository to the remote server.

Example:

```
git remote add origin https://github.com/hverma22/Test5
```

3. git clone

This command is used to obtain a repository from an existing URL.

```
git clone [url]
```

Example: navigate to your repo path where you want to clone and write below command in cmd

```
git clone https://github.com/hverma22/Test2
```

4. How to Create a New Branch in Git

To create a new branch use:

```
$ git checkout -b <new_branch_name>
```

5. How to List Branches in Git

```
$ git branch
```

Example output:

```
develop
```

```
my_feature
```

```
master
```

6. git log

```
git log
```

This command is used to list the version history for the current branch.

Example:

```
git log --oneline
```

7. git merge

```
git merge [branch name]
```

This command merges the specified branch's history into the current branch.

8. How to Switch Branches in Git

When you create a new branch then Git automatically switches to the new branch.

If you have multiple branches, then you can easily switch between branches with git checkout:

```
$ git checkout master
```

```
$ git checkout develop
```

```
$ git checkout my_feature
```

You can get the specific previous version as well.

Command: git checkout <ChangeID> <filePath with extension>

Example:

```
git checkout 6475fgh5 pom.xml
```

9. How to Delete Branches in Git

To delete a local branch:

```
$ git branch -d <local_branch>
```

To delete a remote branch on origin:

```
$ git push origin :<remote_branch>
```

10. Git Stage Files

To stage or simply add files, you need to use git add command. You can stage individual files:

```
$ git add foo.js
```

or all files at once:

```
$ git add .
```

11. git diff

This command shows the file differences which are not yet staged.

Example:

```
git diff --staged
```

```
git diff [first branch] [second branch]
```

12. Git Unstage Changes

If you want to remove a certain file from the stage:

```
$ git reset HEAD foo.js
```

Or remove all staged files:

```
$ git reset HEAD .
```

13. Git Status

If you want to see what files have been created, modified or deleted, Git status will show you a report.

```
$ git status
```

14. git rm

This command deletes the file from your working directory and stages the deletion.

```
git rm [file]
```

15.git commit

```
git commit -m "[commit message]"
```

This command records or snapshots the file permanently in the version history.

Example:

```
git commit -m "First Commit"
```

16. git show

```
git show [commit]
```

This command shows the metadata and content changes of the specified commit.

Command: git show <ChangeID>:<FilePath>

Example:

```
git show 45dhfg56:/src/test/newtest.xml
```

17. Undoing Commits

The following command will undo your most recent commit and put those changes back into staging, so you don't lose any work:

```
$ git reset --soft HEAD~1
```

To completely delete the commit and throw away any changes use:

```
$ git reset --hard HEAD~1
```

18. git push - After you have committed your changes, next is to push to a remote repository.

```
git push [variable name] master
```

This command sends the committed changes of master branch to your remote repository.

Example:

Push a local branch for the first time:

```
git push origin master
```

```
git push origin master --force
```

After that, then you can just use

```
$ git push
```

19. To push a local branch to a different remote branch, you can use:

```
$ git push origin <local_branch>:<remote_branch>
```

20. Undo Last Push

If you have to undo your last push, you can use:

```
$ git reset --hard HEAD~1 && git push -f origin master
```

21.git config

This command sets the author name and email address respectively to be used with your commits.

```
git config --global user.name "[name]"
```

```
git config --global user.email "[email address]"
```

Example:

```
git config user.name "Hitendra Kuamar Verma"
```

```
git config user.email "Hitendra@Hitendra-PC"
```

22. git pull

```
git pull [Repository Link]
```

This command fetches and merges changes on the remote server to your working directory.

Example:

```
git pull https://github.com/hverma22/Test2.git
```

5. What is the version control tool you are using and tell me the steps what you follow and how will you resolve conflicts?

Answer: Please refer below YouTube video link

<https://www.youtube.com/watch?v=CQqdTDBlog&feature=youtu.be>

6. What is the difference between SVN & GIT?

Answer: Below are the differences -

- Git is a distributed VCS; SVN is a non-distributed VCS.
- Git uses multiple repositories including a centralized repository and server, as well as some local repositories; SVN is a centralized version control system.
- The content in Git is stored as metadata; SVN stores files of content.
- Git branches are easier to work with than SVN branches.
- Git does not have the global revision number feature like SVN has.
- Git has better content protection than SVN.
- Git was developed for Linux kernel by Linus Torvalds; SVN was developed by CollabNet, Inc.
- Git belongs to the 3rd generation of Version Control tools; SVN belongs to the 2nd generation of Version Control tools

Selenium Integration with Git and GitHub

What is CM and what are diff SCM tool?

- Configuration management(CM) is managing the configuration of all of the project's key products and assets.
- SCM stands for Source Code Management is an integral part of any project in the IT world.
- Important component in DeveOps culture.
- Source Code Management or Version Control Systems in any project ensure all the members of a team stay on top of the source code changes.
- SCM practices include revision control and the establishment of baselines.

Top SCM Tools:

- MS Team Foundation Server (TFS):
- Kallithea - Open Source
- GitLab - Continuous Integration (CI), Continuous Delivery (CD) is an integral part of GitLab
- Bitbucket Server:
- Subversion (SVN):
- Git and GitHub

Why do we use version control System?

- **Collaboration**- Without a SCM in place, you're probably working together in a shared folder on the same set of files. It's extremely error prone, someone will overwrite someone else's changes. With a SCM, everybody on the team is able to work absolutely freely - on any file at any time.
- **Storing Versions**
- **Restoring Previous Versions**

- **Understanding What Happened:** Every time you save a new version of your project, your SCM requires you to provide a short description of what was changed. This helps you understand how your project evolved between versions.

- **Backup**

Git and GitHub

- Git – initially developed by Linus Torvalds is a version control system.
- Git is a version control system that lets you manage and keep track of your source code history.
- GitHub is a cloud-based hosting service that lets you manage Git repositories.

Different Terminologies with GitHub

Repository: You can simply, treat it as a storage area of your workplace that contains all your documentation files and the history of changes.

Clone: Clones are literally clones (copies) of a repository that sit on the developer's computer instead of a server elsewhere.

Commit: Whatever the changes you make in your files will come under commit. Every change is saved under a particular name or ID which is also called "revision".

Push: Pushing refers to sending your committed changes to a remote repository such as GitHub.com.

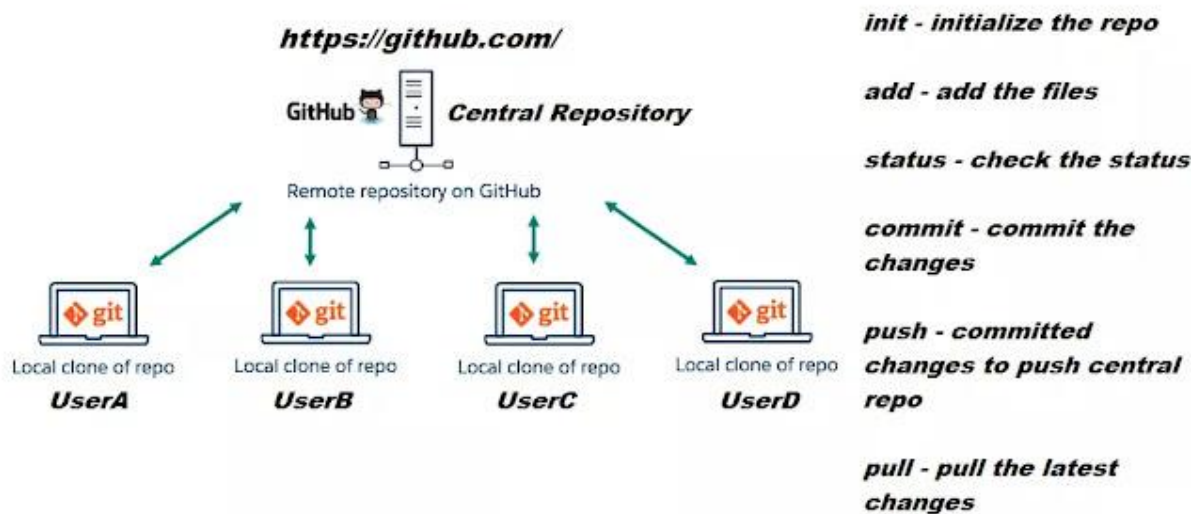
Pull Request: If you have made changes in code/script, to show the other collaborators you send a pull request.

Fork: It is a copy of other's repository in your account in which you can make changes and it won't affect the original code.

Branching: When you extract a portion /section of code from the main or remote track of your software, then it is called 'branch' and the process is known as Branching.

Fetch: Fetching refers to getting the latest changes from an online repository (like GitHub.com) without merging them in.

Merge: Merging takes the changes from one branch (in the same repository or from a fork), and applies them into another.



•

Steps to work with Git and GitHub

- **Download and Install Git** - <https://git-scm.com/download/win> , <https://git-scm.com/download/mac> (Please refer below YouTube video at 27:00 minute for download/installation of Git)
- Create a GitHub Account
- Using Command Prompt - by using different git commands
- Using Eclipse - we have inbuilt Git plugin in eclipse

1. git config

This command sets the author name **and** email address respectively to be used **with** your commits.

```
git config --global user.name "[name]"
```

```
git config --global user.email "[email address]"
```

Example:

```
git config user.name "Hitendra Kuamar Verma"
```

```
git config user.email "Hitendra@Hitendra-PC"
```

2. git init

This command **is** used to start a new repository.

```
git init [repository path]
```

Example: navigate to repo path **and**

```
git init
```

3. git clone

This command **is** used to obtain a repository **from an existing URL**.

```
git clone [url]
```

Example: navigate to your repo path where you want to clone **and** write below command **in** cmd

```
git clone https://github.com/hverma22/Test2
```

4. git add

This command adds a file to the staging area.

```
git add [file]
```

Example:

```
git add [file]
```

```
git add *
```

```
git add .
```

5. git commit

```
git commit -m "[commit message]"
```

This command records **or** snapshots the file permanently **in** the version history.

Example:

```
git commit -m "First Commit"
```

6. git diff

This command shows the file differences which are **not** yet staged.

Example:

`git diff --staged`
`git diff [first branch] [second branch]`

7. git reset

This command unstages the file, but it preserves the file contents.

`git reset [file]`
`git reset [commit]`
`git reset --hard [commit]`

8. git status

`git status`
This command lists all the files that have to be committed.

9. git rm

This command deletes the file **from your working directory and stages the deletion.**

`git rm [file]`

10. git branch

`git branch`
This command lists all the local branches **in** the current repository.
Example:
`git master`

11. git log

`git log`
This command **is** used to list the version history **for** the current branch.
Example:
`git log --online`

12. git merge

`git merge [branch name]`
This command merges the specified branch's history into the current branch.

13. git remote add [variable name] [Remote Server Link]

This command **is** used to connect your local repository to the remote server.
Example:
`git remote add origin https://github.com/hverma22/Test5`

14. git push

`git push [variable name] master`
This command sends the committed changes of master branch to your remote repository.
Example:
`git push origin master`
`git push origin master --force`

15. git pull

`git pull [Repository Link]`
This command fetches **and** merges changes on the remote server to your working directory.

Example:

```
git pull https://github.com/hverma22/Test2.git
```

16. git show

```
git show [commit]
```

This command shows the metadata **and** content changes of the specified commit.

Command: `git show <ChangeID>:<FilePath>`

Example:

```
git show 45dhfg56:/src/test/newtest.xml
```

17. Checkout

```
git checkout [branch name]
```

This command **is** used to switch **from one branch to another or You can get the specific previous version.**

Command: `git checkout <ChangeID> <filePath with extension>`

Example:

```
git checkout 6475fgh5 pom.xml
```

8. Suppose there are 10 classes & I want to push only 5 classes, how do you do that?

Answer: Normally we commit to git, all files are going to git but in your scenario push only single file to git. For this, you have to run specific command to push the only single file to git.

```
$ git commit -m "Message goes here" filename
```

Example to push to single file to git

```
$ git commit -m "Pushing Only Single file to git" config/file1.txt
```

Let's take look how to push one or two or three files to git in a single commit.

```
$ git commit -m "Message goes here" file1 file2 file3
```

Example to push to three files to git

```
$ git commit -m "Pushing Only three files to git" config/file1.txt config/file2.txt config/file3.txt
```

9. Mention the various Git repository hosting functions.

Answer:

- Github
- Gitlab
- Bitbucket
- SourceForge
- GitEnterprise

10. In Git how do you revert a commit that has already been pushed and made public?

Answer: There can be two approaches to tackle this question and make sure that you include both because any of the below options can be used depending on the situation:

Remove or fix the bad file in a new commit and then push it to the remote repository. This is the most obvious way to fix an error. Once you have made necessary changes to the file, then commit it to the remote repository using the command: `git commit -m "commit message"`

Also, you can create a new commit that undoes all changes that were made in the bad commit. To do this use the command

```
git revert <name of bad commit>
```

11. What is the difference between git pull and git fetch?

Answer: Git pull command pulls new changes or commits from a particular branch from your central repository and updates your target branch in your local repository.

Git fetch is also used for the same purpose but it works in a slightly different way. When you perform a git fetch, it pulls all new commits from the desired branch and stores it in a new branch in your local repository. If you want to reflect these changes in your target branch, git fetch must be followed with a git merge. Your target branch will only be updated after merging the target branch and fetched branch. Just to make it easy for you, remember the equation below:

Git pull = git fetch + git merge

12. What is git stash?

Answer: Often, when you've been working on part of your project, things are in a messy state and you want to switch branches for some time to work on something else. The problem is, you don't want to do a commit of half-done work just so you can get back to this point later. The answer to this issue is Git stash.

Stashing takes your working directory that is, your modified tracked files and staged changes and saves it on a stack of unfinished changes that you can reapply at any time.

```
$ git status
```

```
modified: index.php
```

```
modified: css/styles.css
```

Apply Git Stash

```
$ git stash
```

Saved working directory and index state WIP on master:

2dfe283 Implement the new login box

HEAD is now at 2dfe283 Implement the new login box

Git's Stash is meant as a temporary storage. When you're ready to continue where you left off, you can restore the saved state easily:

```
$ git stash pop
```

13. What is the function of 'git stash apply'?

Answer: If you want to continue working where you had left your work then 'git stash apply' command is used to bring back the saved changes onto your current working directory.

```
git stash apply n
```

To get list of stashes:

```
git stash list
```

14. What is the function of 'git config'?

Answer: Git uses your username to associate commits with an identity. The git config command can be used to change your Git configuration, including your username.

Now explain with an example.

Suppose you want to give a username and email id to associate a commit with an identity so that you can know who has made a particular commit. For that I will use:

```
git config --global user.name "Your Name": This command will add a username.
```

```
git config --global user.email "Your E-mail Address": This command will add an email id.
```

15. How will you know in Git if a branch has already been merged into master?

Answer: To know if a branch has been merged into master or not you can use the below commands:

```
git branch --merged – It lists the branches that have been merged into the current branch.
```

```
git branch --no-merged – It lists the branches that have not been merged.
```

16. Can you explain the Gitflow workflow?

Answer: To record the history of the project, Gitflow workflow employs two parallel long-running branches – master and develop:

Master – this branch is always ready to be released on LIVE, with everything fully tested and approved (production-ready).

Hotfix – these branches are used to quickly patch production releases. These branches are a lot like release branches and feature branches except they're based on master instead of develop.

Develop – this is the branch to which all feature branches are merged and where all tests are performed. Only when everything's been thoroughly checked and fixed it can be merged to the master.

Feature – each new feature should reside in its own branch, which can be pushed to the develop branch as their parent one.

Jenkins

1. What is CI?

Answer: What is Continuous Integration?

Continuous Integration is a development practice in which the developers are required to commit changes to the source code in a shared repository several times a day or more frequently. Every commit made in the repository is then built. This allows the teams to detect the problems early. Continuous Integration is the most important part of DevOps that is used to integrate various DevOps stages. Jenkins is the most famous Continuous Integration tool.

Continuous Integration comprises of:

- Development and Compilation
- Database Integration
- Unit Testing
- Production Deployment
- Code Labeling
- Functional Testing
- Generating and Analyzing Reports

Below is the list of few popular Continuous Integration tools:

- Jenkins
- TeamCity
- Travis CI
- Go CD
- Bamboo
- GitLab CI
- CircleCI
- Codeship

Continuous Integration: CI or Continuous Integration is an engineering practice in which members of a development team integrate their code at a very high frequency. Teams implementing CI aim to integrate code daily or, in some cases, even hourly.

Continuous Delivery: CD or Continuous Delivery is the practice of ensuring that code is always in a deployable state. This means that all changes to code – new features, bug fixes, experiments, configuration changes – are always ready for deployment to a production environment.

Continuous Deployment: CD can also mean Continuous Deployment – a practice in which all changes are automatically deployed into production. Unlike Continuous Delivery, there is no final manual approval step before releasing into production.

2. What is Jenkins?

Answer: Jenkins is an open source automation tool written in Java with plugins built for Continuous Integration purpose. Jenkins is used to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. It also allows you to continuously deliver your software by integrating with a large number of testing and deployment technologies. Jenkins integrates development life-cycle processes of all kinds, including build, document, test, package, stage, deploy, static analysis and much more. Jenkins achieves Continuous Integration with the help of plugins. Plugins allow the integration of various DevOps stages. If you want to integrate a particular tool, you need to install the plugins for that tool. For example: Git, Maven, HTML publisher etc.

Advantages:

Advantages of Jenkins include:

- It is an open source tool with great community support.
- It is easy to install.
- It has 1000+ plugins to ease your work. If a plugin does not exist, you can code it and share with the community.
- It is built with Java and hence, it is portable to all the major platforms.

3. What is a Jenkins Pipeline?

Answer: Jenkins Pipeline (or simply “Pipeline”) is a suite of plugins which supports implementing and integrating continuous delivery pipelines into Jenkins.

4. What is Groovy in Jenkins?

Answer: Groovy is the default scripting language that is being used in the development of JMeter Version 3.1. Currently Apache Groovy is the dynamic object-oriented programming language that is used as a scripting language for the Java platform. Apache Groovy comes with some useful features such as Java Compatibility and Development Support.

5. Diff between maven, Jenkins, ant

Answer: Maven is a build tool, in short a successor of ant. It helps in build and version control. However Jenkins is continuous integration system, where in maven is used for build. Jenkins can be used to automate the deployment process.

6. Which SCM tools Jenkins supports?

Answer: below are the Source code management tools supported by Jenkins:

- AccuRev
- CVS
- Subversion
- Git
- Mercurial
- Perforce
- Clearcase
- RTC

7. What is the relation between Hudson and Jenkins?

Answer: There is no such difference between Hudson and Jenkins. Jenkins is actually the renamed version of Hudson

Jenkins was forked from Hudson when Sun was acquired by Oracle who aimed to develop a commercial version of the software. Since the fork, Jenkins has grown to be much more than a CI solution.

8. How to make sure that your project builds doesn't break in Jenkins?

Answer: To make sure Jenkins build isn't broken in the slightest degree we need to make sure that we tend to perform a successful clean install on the local machine with all unit tests.

Then make sure that all code changes are checked in without any issues.

Then synchronize with a repository to make sure that all needed config and changes and any variations are checked into the repository.

9. How to move or copy Jenkins from one server to another

Answer:

1. Copy all the files in your JENKINS_HOME directory over to the new server.
2. Point JENKINS_HOME on the new server at the new directory.
3. Copy the Jenkins war file (or your servlet container setup if you have one) over to the new machine and start it up.

All Jenkins settings, jobs, plugins, config, etc. live in JENKINS_HOME.

You just need a copy of it to start it elsewhere.

10. How can we create a backup and copy files in Jenkins?

Answer: to create a backup all you need to do is to periodically back up your JENKINS_HOME directory.

This contains all of your build jobs configurations, your slave node configurations, and your build history.

To create a back-up of your Jenkins setup, just copy this directory. You can also copy a job directory to clone or replicate a job or rename the directory.

You can create a cron job to do so.

Or you can use “Thin Backup” plugin in Jenkins.

11. How can you clone a git repo via Jenkins?

Answer: To create a clone repository via Jenkins you need to use your login credentials in the Jenkins System. To achieve the same you need to enter the Jenkins job directory and execute the git config command.

12. How to run smoke and sanity using Jenkins?

Answer: Please refer below YouTube video

<https://www.youtube.com/watch?v=KG47Y8uUtVI&list=PLsGOlyTzNH6f6azWCMypTpW11F-3gyDYxV&index=2>

13. How to reset Jenkins username and password

Answer: To reset the jenkins admin password, You can simply disable the security in the config.xml file.

1. If your jenkins is running on the Linux OS, edit the below file.

vi /var/lib/jenkins/config.xml file.

2. Search for the word <useSecurity>true</useSecurity>

and change the word true to false.

3. Restart the Jenkins server.

service jenkins restart

4. Now go to the Jenkins portal again and Jenkins will not ask any credentials this time. You navigate to "Manage Jenkins" to set the administrator password again.

5. Enable the security again by changing settings to <useSecurity>true</useSecurity> and restart the Jenkins again.

Note:

If your jenkins is running on Windows OS, config.xml file located in C:\Program Files (x86)\Jenkins\ folder.

14. How do you setup Maven Project in Jenkins?

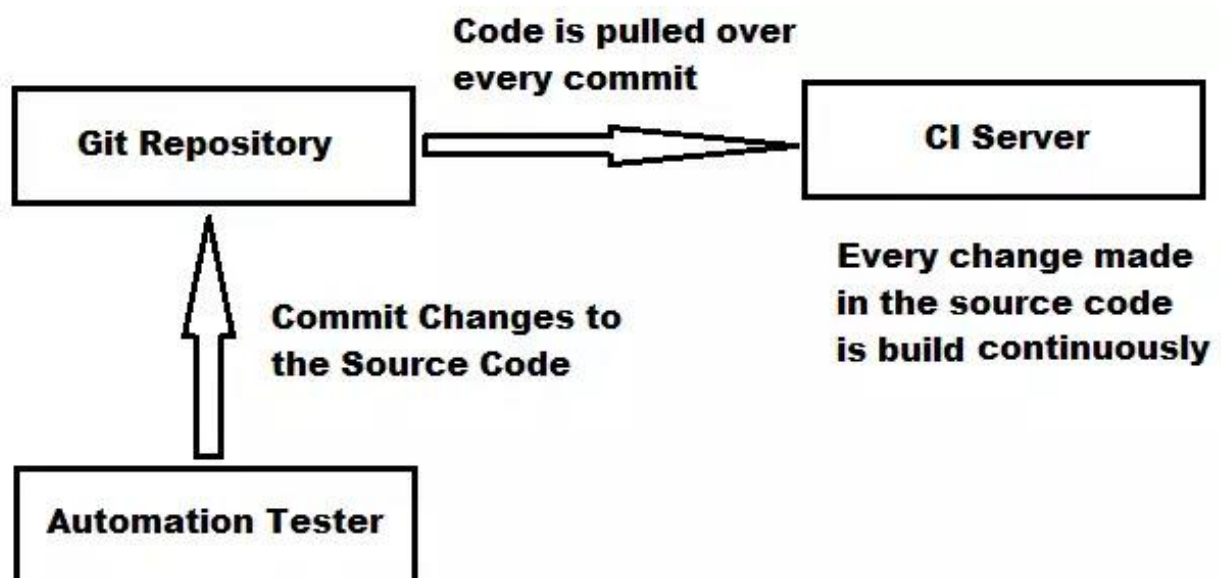
Selenium Integration with Jenkins

In this post we will discuss about following points:

- What is Continuous Integration?
- What is Jenkins
- Advantages of Jenkins
- Other CI tools
- Complete Jenkins installation Process

What is Continuous Integration?

Continuous Integration is a development practice in which the Automation Testers are required to commit changes to the source code in a shared repository several times a day or more frequently. Every commit made in the repository is then built. This allows the teams to detect the problems early. Continuous Integration is the most important part of DevOps that is used to integrate various DevOps stages. Jenkins is the most famous Continuous Integration tool.



What is Jenkins?

Jenkins is an open source automation tool written in Java with plugins built for Continuous Integration purpose. Jenkins is used to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. It also allows you to continuously deliver your software by integrating with a large number of testing and deployment technologies. Jenkins integrates development life-cycle processes of all kinds, including build, document, test, package, stage, deploy, static analysis and much more. Jenkins achieves Continuous Integration with the help of plugins. Plugins allows the integration of Various DevOps stages. If you want to integrate a particular tool, you need to install the plugins for that tool. For example: Git, Maven, HTML publisher etc.

Advantages:

- Advantages of Jenkins include:

- It is an open source tool with great community support.
- It is easy to install.
- It has 1000+ plugins to ease your work. If a plugin does not exist, you can code it and share with the community.
- It is built with Java and hence, it is portable to all the major platforms.

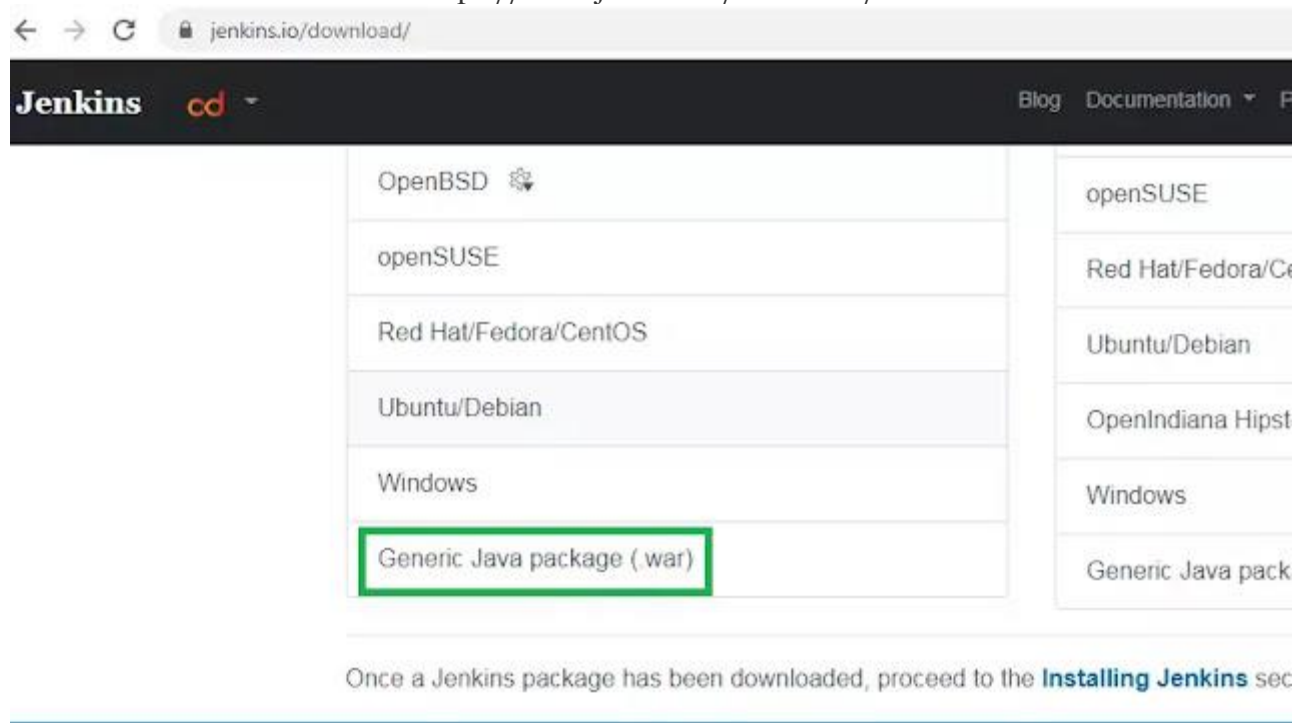
Apart from Jenkins, we have many more tools in the market such as:

- Anthill
- Bamboo
- Cruise Control
- Team City

Complete Jenkins Installation Process

URL: <https://www.jenkins.io/download/>

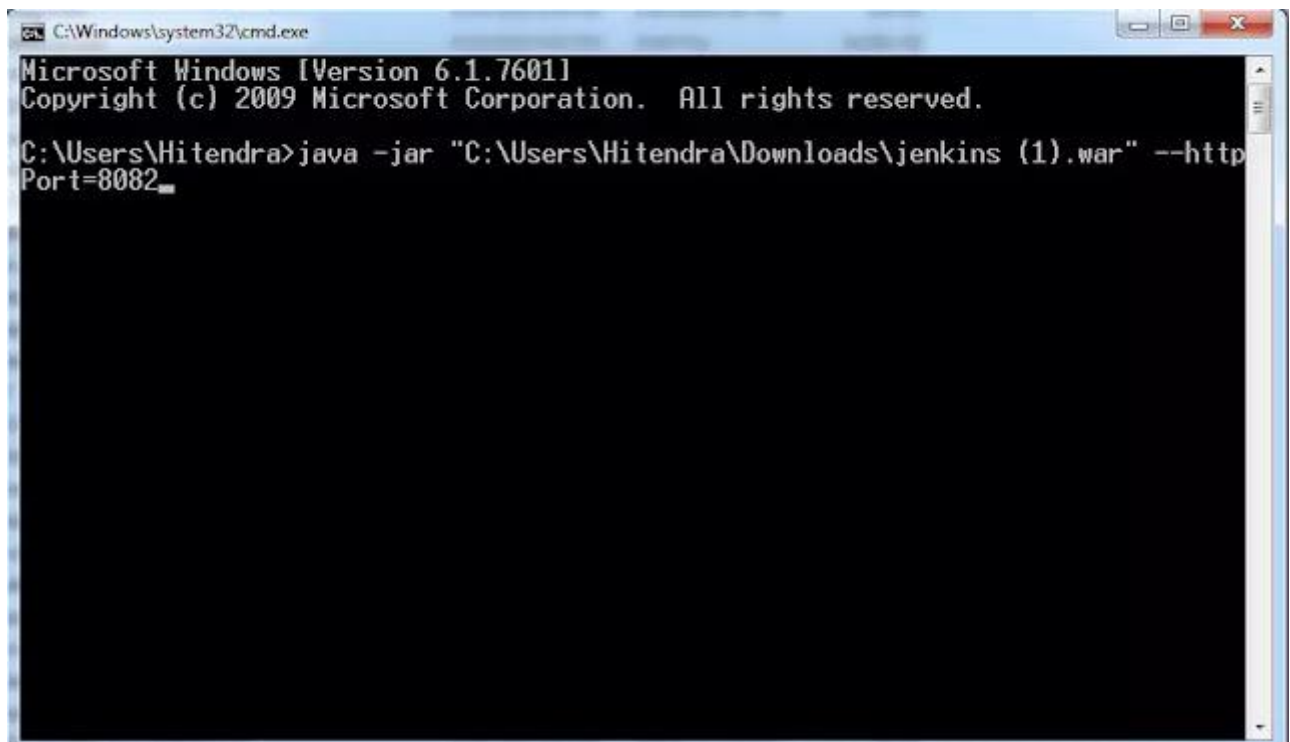
1. Download the WAR file from <https://www.jenkins.io/download/>



2. Start the Jenkins using below commands:

```
>C:\Users\Hitendra>java -jar <Path of WAR file> --httpPort=8082
```

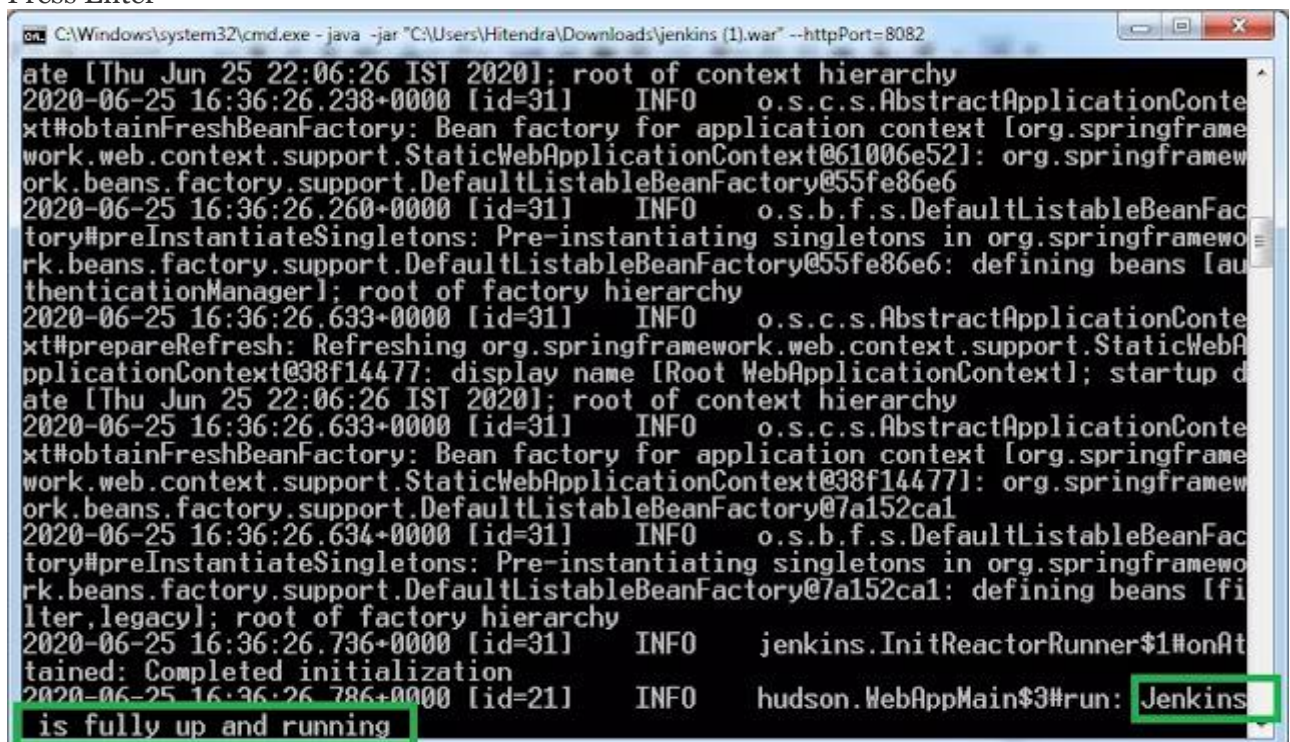
Default port is 8080, no need to specify port if your jenkins running on 8080 port



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Hitendra>java -jar "C:\Users\Hitendra\Downloads\jenkins (1).war" --httpPort=8082
```

Press Enter



```
C:\Windows\system32\cmd.exe - java -jar "C:\Users\Hitendra\Downloads\jenkins (1).war" --httpPort=8082
ate [Thu Jun 25 22:06:26 IST 2020]; root of context hierarchy
2020-06-25 16:36:26.238+0000 [id=31] INFO o.s.c.s.AbstractApplicationConte
xt#obtainFreshBeanFactory: Bean factory for application context [org.springframe
work.web.context.support.StaticWebApplicationContext@61006e52]: org.springframe
work.beans.factory.support.DefaultListableBeanFactory@55fe86e6
2020-06-25 16:36:26.260+0000 [id=31] INFO o.s.b.f.s.DefaultListableBeanFac
tory#preInstantiateSingletons: Pre-instantiating singletons in org.springframewo
rk.beans.factory.support.DefaultListableBeanFactory@55fe86e6: defining beans [au
thenticationManager]; root of factory hierarchy
2020-06-25 16:36:26.633+0000 [id=31] INFO o.s.c.s.AbstractApplicationConte
xt#prepareRefresh: Refreshing org.springframework.web.context.support.StaticWebA
pplicationContext@38f14477: display name [Root WebApplicationContext]; startup d
ate [Thu Jun 25 22:06:26 IST 2020]; root of context hierarchy
2020-06-25 16:36:26.633+0000 [id=31] INFO o.s.c.s.AbstractApplicationConte
xt#obtainFreshBeanFactory: Bean factory for application context [org.springframe
work.web.context.support.StaticWebApplicationContext@38f14477]: org.springframe
work.beans.factory.support.DefaultListableBeanFactory@7a152ca1
2020-06-25 16:36:26.634+0000 [id=31] INFO o.s.b.f.s.DefaultListableBeanFac
tory#preInstantiateSingletons: Pre-instantiating singletons in org.springframewo
rk.beans.factory.support.DefaultListableBeanFactory@7a152ca1: defining beans [fi
lter,legacy]; root of factory hierarchy
2020-06-25 16:36:26.736+0000 [id=31] INFO jenkins.InitReactorRunner$1#onAt
tained: Completed initialization
2020-06-25 16:36:26.786+0000 [id=21] INFO hudson.WebAppMain$3#run: Jenkins
is fully up and running
```

3. Browse to <http://localhost:8080> (or whichever port you configured for Jenkins when installing it) and wait until the Unlock Jenkins page appears.

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

```
/var/jenkins_home/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

Continue

4. From the Jenkins console log output, copy the automatically-generated alphanumeric password from the cmd or from given path.

```
INFO: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultList
beans [filter,legacy]: root of factory hierarchy
Sep 30, 2017 7:18:39 AM jenkins.install.SetupWizard init
INFO:

*****
*****
*****

Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:

2f064d3663814887964b682940572567

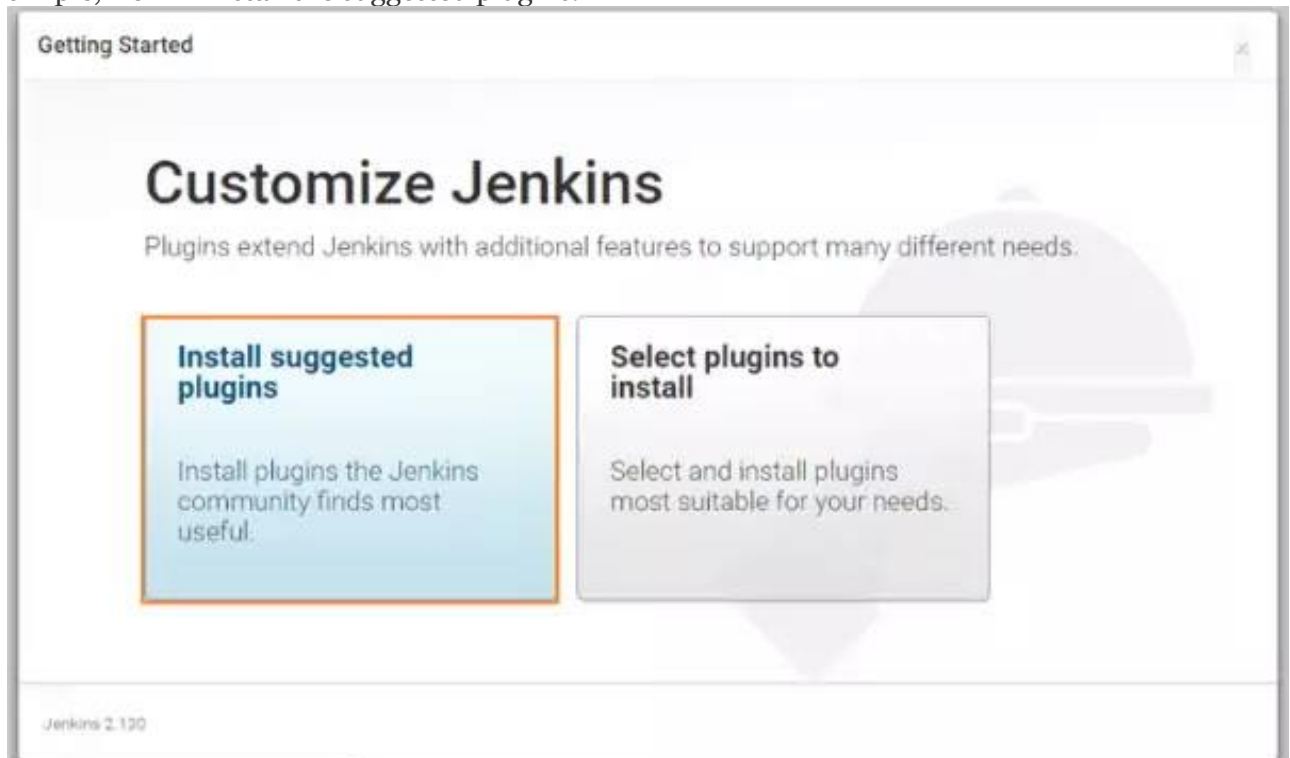
This may also be found at: /var/jenkins_home/secrets/initialAdminPassword

*****
*****
*****

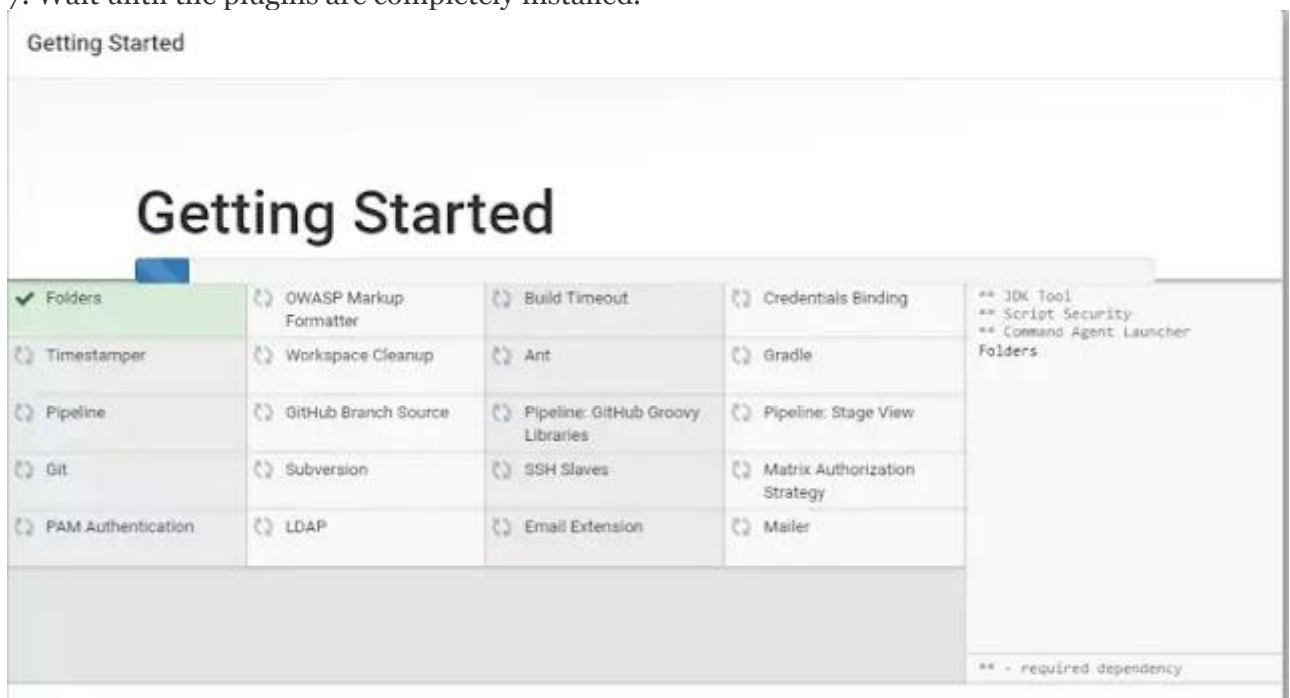
--> setting agent port for jnlp
--> setting agent port for jnlp... done
Sep 30, 2017 7:18:51 AM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Sep 30, 2017 7:18:52 AM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Sep 30, 2017 7:18:52 AM hudson.WebAppMain$3 run
INFO: Jenkins is fully up and running
```


5. On the Unlock Jenkins page, paste this password into the Administrator password field and click Continue.

6. You can install either the suggested plugins or selected plugins you choose. To keep it simple, we will install the suggested plugins.



7. Wait until the plugins are completely installed.



8. The next thing we should do is create an admin user for Jenkins. Put in your details and click “Save and Continue”.

Getting Started

Create First Admin User

Username:

Password:

Confirm password:

Full name:

E-mail address:

9. Click “Save and Finish” to complete the Jenkins installation.

Getting Started

Instance Configuration

Jenkins URL:

http://localhost:8080/

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD_URL environment variable provided to build steps.

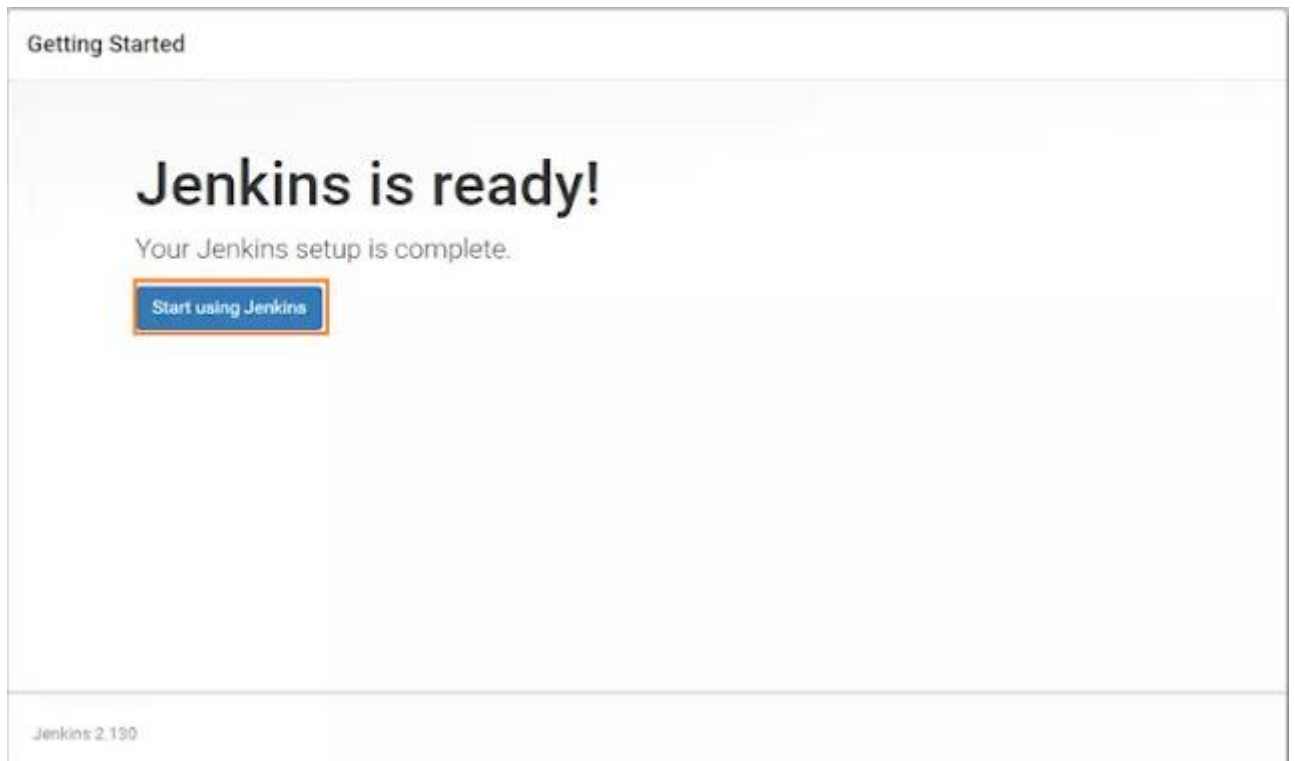
The proposed default value shown is not saved yet and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.130

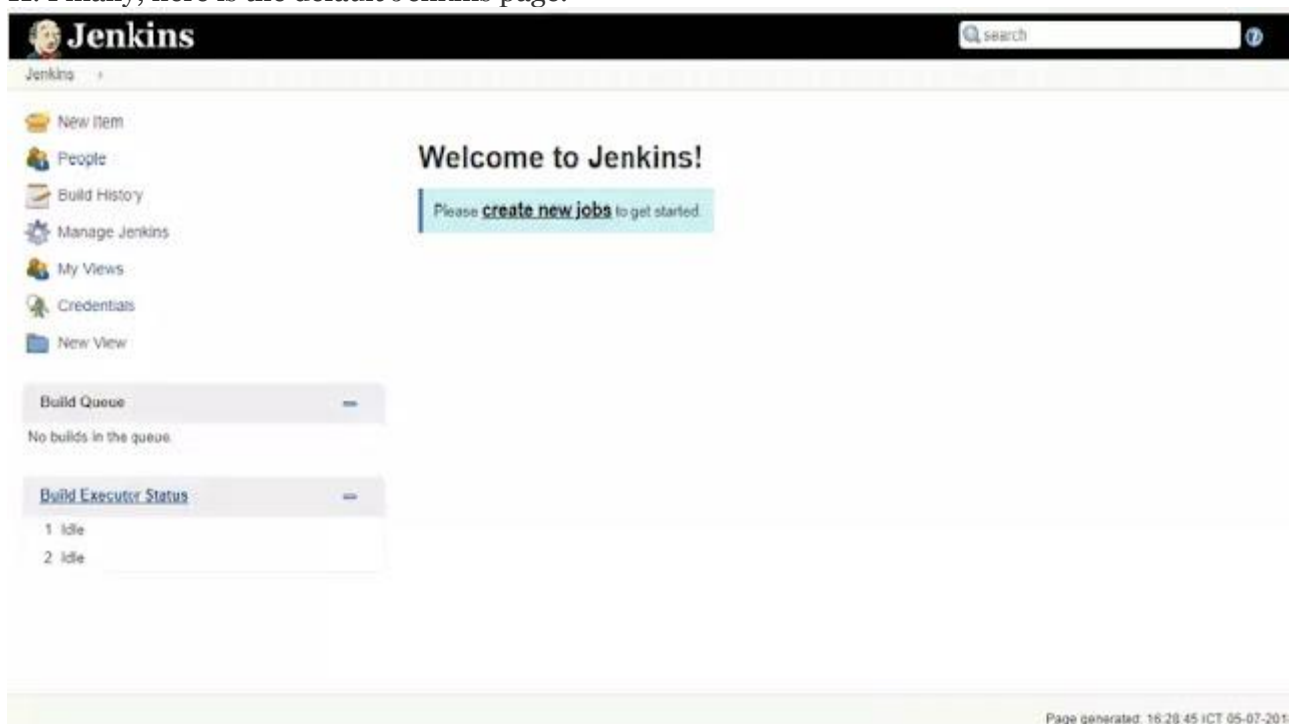
Not now

Save and Finish

10. Now, click “Start using Jenkins” to start Jenkins.



11. Finally, here is the default Jenkins page.



Please refer below YouTube videos to understand the Different Configurations on Jenkins

- **Demo 1:** Create a batch file and run on Jenkins (Integration of Eclipse Java Project + Jenkins)

- Steps to create the batch file

```
set projectLocation=D:\Workspace_Eclipse\TestNGJenkinsDemo
cd %projectLocation%
set classpath=%projectLocation%\bin;%projectLocation%\lib\*
java org.testng.TestNG %projectLocation%\testng.xml
pause
```

Save this file with extension (.bat).
Sample Email Body Template:

```
Hi Team,
<br/>
<br/>
Please find the below Automation Testing Results for Project:
<b>${PROJECT_NAME}</b>.
<br/>
<br/>
<b>Project Name:</b> ${PROJECT_NAME}
<br/>
<b>Build#</b>: ${BUILD_NUMBER}
<br/>
<b>Build Status</b>: ${BUILD_STATUS}.<br/>
<br/>
Check <a href='http://localhost:8082/job/Demo4/ws/ExtentDemo/test-
output/ExtentReport/MyReport.html'>Extent Report</a> to view full
results.<br/>
<br/>
Please find below <b>Colsole Logs</b> for your reference<br/>
<br/>
--LOG-BEGIN--<br/>
<pre style='line-height: 22px; display: block; color: #333; font-
family: Monaco,Menlo,Consolas,"Courier New",monospace; padding:
10.5px; margin: 0 0 11px; font-size: 13px; word-break: break-all;
word-wrap: break-word; white-space: pre-wrap; background-color:
#f5f5f5; border: 1px solid #ccc; border: 1px solid rgba(0,0,0,.15);
-webkit-border-radius: 4px; -moz-border-radius: 4px; border-radius:
4px;'>
${BUILD_LOG, maxLines=500, escapeHtml=true}
</pre>
--LOG-END--
<br/>
<br/>
Thank You
KARTHIC
```

15. How do you start Jenkins?

Answer: To start Jenkins from command line

Open command prompt.

Go to the directory where your war file is placed and run the following command: java -jar jenkins.war

16. How to create a job in Jenkins?

Answer: Few of the steps given below

Step 1 – Go to the Jenkins dashboard and Click on New Item.

Step 2 – in the next screen, enter the Item name, in this case we have named it Helloworld.

Step 3 – the following screen will come up in which you can specify the details of the job.

Step 4 – we need to specify the location of files which need to be built.

17. What are the basic plugins you used in Jenkins?

Answer:

Maven Integration - This plug-in provides, for better and for worse, a deep integration of Jenkins and Maven.

Git plugin - This plugin integrates Git with Jenkins.

TestNG Results Plugin - This plugin integrates TestNG test reports to Jenkins.

HTML Publisher plugin - This plugin publishes HTML reports.

Email Extension - This plugin is a replacement for Jenkins's email publisher. It allows to configure every aspect of email notifications: when an email is sent, who should receive it and what the email says.

Email Extension Template Plugin - This plugin allows administrators to create global templates for the Extended Email Publisher.

External Monitor Job Type Plugin - Adds the ability to monitor the result of externally executed jobs

Green Balls- Because green is better than blue! For color blind support configure user property.

18. On which platform JENKINS will work?

Answer: It is built with Java and hence, it is portable to all the major platforms.

19. Mention what are the commands you can use to start Jenkins manually?

Answer: To start Jenkins manually, you can use either of the following

(Jenkins_url)/restart: Forces a restart without waiting for builds to complete

(Jenkin_url)/safeRestart: Allows all running builds to complete

20. Mention what are the two components Jenkins is mainly integrated with?

Answer: Jenkins is mainly integrated with two components

- Version Control system like GIT, SVN

- And build tools like Apache Maven.

21. What are Triggers?

Answer: Trigger in Jenkins defines the way in which the pipeline should be executed frequently. PollSCM, Cron, etc are the currently available Triggers.