SELENIUM

Notes By

SURESH

Selenium with Project Lab CheckList

S.No	Tasks	Status	Comments
1	Element Locators --- name , id ,class and Xpath		
2	Element Locators --- Xpath in details		
3	Java Program for example of class		
4	Java Program for class ,Method Object		
5	Java Program for Static method		
6	Java Program for loops & cond		
7	Arrays and Switch stmt		
8	Java program for Providing the values in Runtime		
9	Oops Programs		
10	WebDrier - Login and Logout Prgram		
11	WebDriver - Verification and MouseOver Program		
12	WebDriver - Frames and Alerts		
13	WebDriver - RobotClass and Dropdown		
14	WebDriver - WaitMethod and File Upload		
15	WebDriver – Windows Handler and JSE		
16	Reading the data from WebTable and Excel		
17	Framework Implementation(2TC)		
18	TestNG		
19	WebDriver + TestNG		
20	WebDriver + TestNG + TestSuite		
21	WebDriver + TestNG + TestSuite + Log4j		
22	WebDriver + TestNG + TestSuite + Log4J + Jenkins		
23	WebDriver + TestNG + TestSuite + Log4J + Jenkins + GitHub		
24	POM FrameWork		
25	Maven + Cucumber		
26	API Testing		
27 28	Selenium Exam and Discussion on Answers Mock Interviews		

What is Automation Testing ?
Manual Testing is performed by a human sitting in front of a computer carefully executing the test steps.

Testing an application with third party Software help or Automation Tool is called Automation testing. (OR)
Converting manual test cases into Automation Scripts (in form of code) is called Automation Tesing

Retesting and Regression Testing
What is Retesting?
Retesting of the application to verify whether defects have been fixed or not.
What is Regression testing?
1.It is Re=Execution of some or all the test cases of a testing activity for each build to verify that changes or fixes made have not introduced new errors
2. Regression testing is done in three situations
i) One is after fixing the bug
ii) Second one is if a new change request will come from client
iii) Third one is when Enviromnent changes
Here we have to verify whether already existing functionality can getting defects (or) not.
The real use and purpose of automated test tools is to automate regression testing

Why to Start Test Automation?
- To Optimize the Speed & Efficiency
- To Increase the Quality and Decrease the Cost
- When there is a repetition or a need to run the test cases a lot number of times in a test cycle.
- When there are number of test cases under one test-suite
- When there is a requirement of running the test cases in a defined order
- To Increase the Test Coverage
- When you need to run the same test cases on different machines at the same time
- When you need to test single functionality with multiple data sets.
- When you need to run Regression/Sanity/Smoke Test Suite
- Generation of Detailed Reports

Benefits of Automation Testing
1. Faster feedback: Automation of testing actually comes as a relief for development and testing engineers during different phases of an app development life-cycle. It helps to better the communication among coders, designers, product owners and rectifies the potential glitches with no delay. It also increases the efficiency of the development team.
2. Saves time: Writing codes in manual testing is a tedious process especially when the testers have to write long scripts while doing regression testing. The requirement of fast bug free delivery of apps is not fulfilled and companies have to suffer business losses.
3. Early Bug Detection: Unlike manual testing, bugs can be detected early during the development phase in an automated testing which saves a lot of time during Mobile app

development life-cycle for both developers and testers.

4. Re-usability of Scripts: Testing automation makes the lives of the testers easy. The scripts can be reused with no or minimal changes in the script. These scripts can be used multiple times no matter if there are changes in the OS version of the device. The scripts and steps are stored and it helps to repeat the test without skipping or forgetting any step.

5. Running tests anytime, anywhere: Automated testing help test engineers to run their tests 24/7. If the test engineer has to leave early for the day, he can easily schedule the tests and leave the office. The test results will be ready by the time he logs in again next morning.

6. Distributed Test Execution: Automation testing cuts down complexities with its distributed test execution feature. It helps the testers to run a test script on more than one computer or shared network or servers simultaneously. So, only an automation testing tool is the requirement rather than multiple tools.

7. Robust and simpler reporting: Automation testing gives us the benefit of tracking each test script. All the test scripts executed will be visible in visual logs. The reports generated can evidently show the number of test scripts already executed, scheduled, their reported bugs or issues and the ways in which they have been fixed.

8. Testing Capabilities: Automated testing offers an unmatched and huge testing capability. The mobile app needs to be tested on multiple devices, OS versions, screen sizes etc. which can be efficiently done through automated testing and not by manual testing. In fact, it is almost impossible to get perfect results through manual testing.

9. Better Test Coverage: Test automation can easily execute thousands of different complex test cases during every test run providing coverage that is impossible with manual tests. It can only be possible through automated testing as it can run test scripts on multiple computers with varied configurations. It can look inside an application and see memory contents, data tables, file contents, and internal program states to determine if the app is functioning as expected.

10. Less Manual Effort: If an enterprise has implemented automation tools for testing, then it can accelerate the process and reduce the manual effort multi-fold. Less number of people will be required for a project and they can be utilized for different projects.

11. Improves Accuracy: We have been saying this time and again that automation testing overcomes the shortcomings of manual testing. It has improved the accuracy to a great deal by giving error-free results unlike manual testing where testing is error-prone, delays the delivery and increases the cost. It is especially a boon in stress testing where getting error free results in manual testing is almost impossible.

12. Return on Investment: One of the most important advantages of automation testing is the return on investment to the organization. Every enterprise analyses the return it would get out of its investment and then would go for creating a test automation framework. Automated testing offers immense returns in terms of faster testing, error-free results, less manual effort. If the enterprise has automated testing tool, then the testing would be more efficient and accurate and easy.

13. Volume: You can run your tests on more than 500 devices in automation testing which is impossible in manual testing.

Types of Testing Automation Tools

Open Source : Free of Cost.if we have interenet connection we can download from google and then start working with the tools

Example: Selenium , Sikuli , Sahi , Ruby with Watir etc…
Commercial : Licienced. We need to purchase the licience from that respective companies and then need to start working with tool.
Example : QTP / UFT , Test Partner , TestComplete , RFT Silk etc …

Tool Evalutation Process
Success in any test automation depends on identifying the right tool for automation. Selecting the "correct" Testing Tool for your project is one of the best ways to achieve the project target.

Automation Testing Process
In my current project, Automation tool is evaluated based on below project features:

- Multiple Browser Support
- Language support
- Ease of Use
- Multiple Operating Systems
- Ajax Support
- Web Application support
- UI tests
- Scripting
- Record and play back
- Object recognition capability
- Customization of recorded code as per our requirements
- Synchronization issues
- Test Suite Creation
- Maintenance of scripts when features modified
- Central Object Repository
- APIs availability
- Database verification
- Vendor support
- Framework Creation
- Tool Performance

Based on initial evaluation on above project scenarios, Selenium and QTP or any other automation tools are finalized after evaluation.
Clinet & management team are finalized **SELENIUM** tool to automate my Current Project.

SELENIUM	QTP
Open source	Paid Tool
Works on all OS (Windows, OS X, Linux, Solaris,Mac)	Works only on Windows
Tests only Web applications	Tests web and desktop applications
Works on almost all browsers(edge, Firefox, Chrome, Safari, Opera)	Works on Edge
Code can be made in any one of languages such as Java, C#, Ruby, Python, pearl, php	VB Script
Object Identification Options : Name ,Id , Xpath, CSS, Link text …etc	Object Identification Options : Object properties, Repository objects
Selenium dose not have such built in object repository, but object can be managed by using UI element user extension like Inspect option in All browsers	HP UFT comes with built in object repository. Object repository development and maintenance is quite easy in HP ALM
IDE sometimes does not record some events	Recording is a little reliable
Set of Libraries, around 20MB (Need to include other supporting software)	Around 1.5GB
Saucelabs.com, Element34 , Commercial Support	From HP

Selenium History

In 2004 invented by Jason Huggins and team

Originally name is JavaScript Functional Tool (JSFT)

Open source browser based integration framework (Selenium RC)built originally by Thought Works

Google, who has been a long time user of Selenium, had a developer named Simon Stewart who developed WebDriver. This tool circumvented Selenium's JavaScript sandbox to allow it to communicate with the Browser and Operating System directly using native methods

In 2008, Selenium and WebDriver merged technologies and intellectual intelligence to provide the best possible test automation framework

100% JavaScript and HTML Web Testing Tool

What is Selenium?

Selenium is a free (open source) automated testing suite for web applications across different browsers and platforms. It is quite similar to HP Quick Test Pro (QTP) only that Selenium focuses on automating web-based applications.

Selenium is not just a single tool but a suite of softwares, each catering to different testing needs of an organization. It has four components.

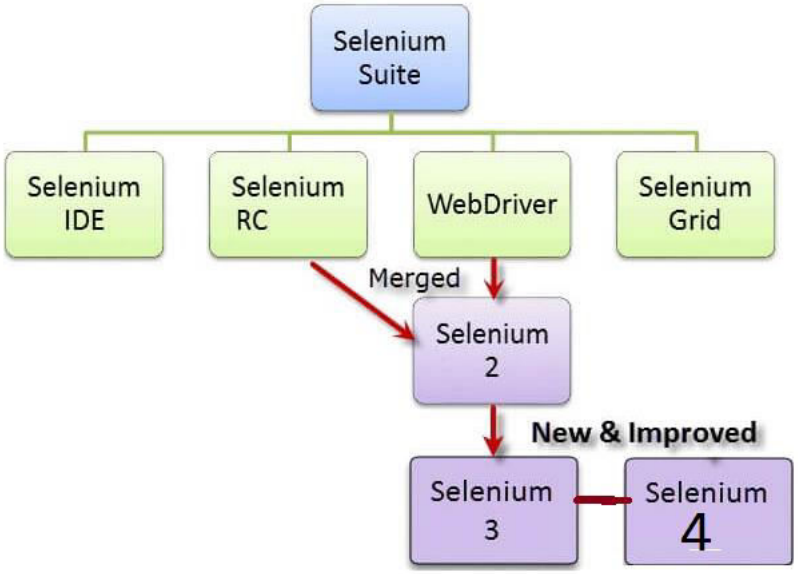
Selenium Components

Selenium IDE (Integrated Development Environment) - Record & Play back

Selenium RC- 1.0 – Server /Client (Selenium. Start /Selenium. Stop)

Selenium 2.0/3.0 / 4.0/ Web Driver /Advanced Selenium

Selenium Grid – To Execute scripts in Multiple Browsers & Systems



Selenium IDE

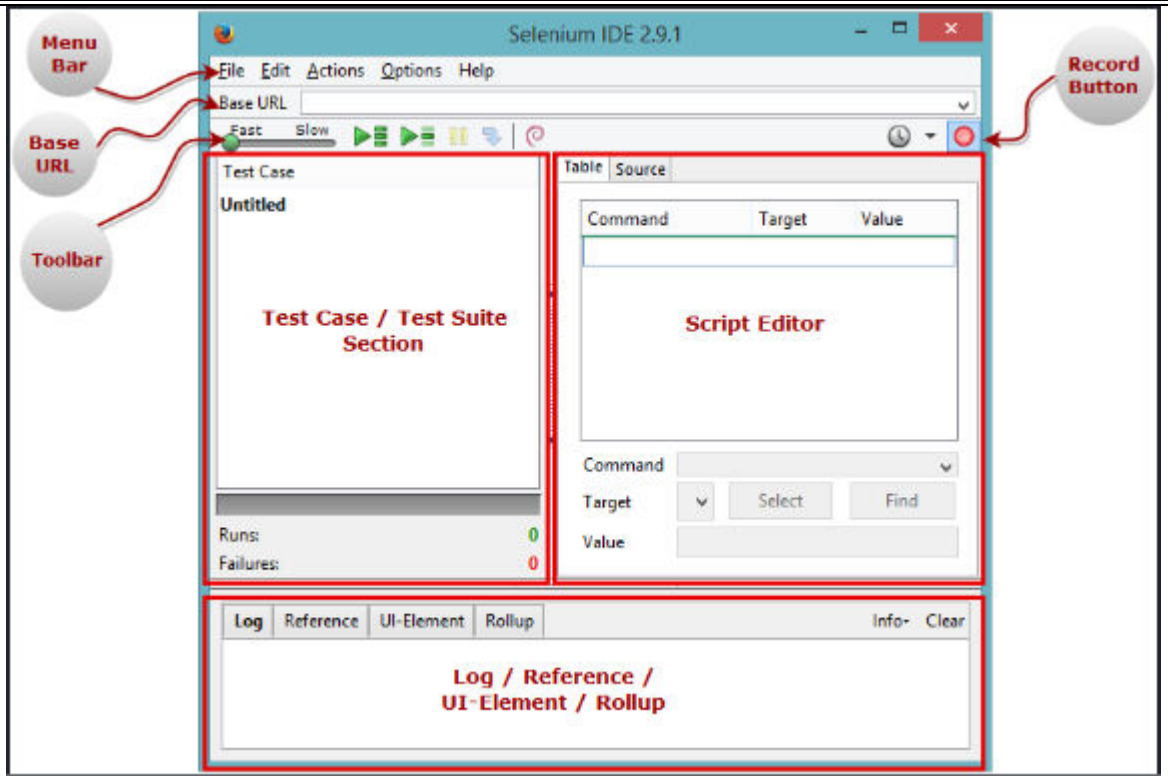
Introduction: The selenium IDE is the tool used to develop your selenium test cases by using record and play back.

Selenium IDE (Integrated Development Environment) is a tool that helps you develop your Selenium test cases. It's an easy-to-use Chrome and Firefox extension and is generally the most reliable method to develop test cases. It records users' actions in the browser for you, using the existing Selenium commands, with parameters defined by the context of the web element. This is not only a time-saver but also an excellent way of learning Selenium script syntax. Previously known as Selenium Recorder, Selenium IDE was initially created by Shinya Kasatani, of Japan and contributed to the Selenium project in 2006.

It was introduced as a Firefox plugin for faster creation of test cases. As it was a Firefox extension, it could automate the browser through a record-and-play feature providing autocompletion support and the ability to move commands around quickly.

Scripts are recorded in a special test scripting language called Selenese for Selenium. Selenese comes up with commands for carrying out actions in a web browser and restoring data from the resulting pages.

The advantage of Selenium IDE is that the tests recorded via the plugin can be exported in different programming languages like Java, Ruby, Python, etc.



Advantages

- It's an easy –to-use
- It's just Firefox plug in and is generally the most efficient way to develop test cases
- Very useful tool for beginners
- Firefox extension which allows record/Play testing paradigm
- Creates the simplest possible locator based on Selene's
- Look at various possible commands in the dropdown
- Records a test at HTML file
- We can export the test as Java /Ruby etc...

Disadvantages

- Selenium-IDE does not directly support:
- Condition statements & Iteration or looping
- Logging and reporting of test results
- Error handling, particularly unexpected errors
- Database testing
- Test case dependency
- Capture screenshots on test failures
- Results Report generations

Selenium RC (Remote Control)

Let me first tell you that Selenium Core was the first version. But with that version, testers had to install both Selenium (a JavaScript program) and the webserver containing the web application being tested on their local systems so that they would belong to the same domain.

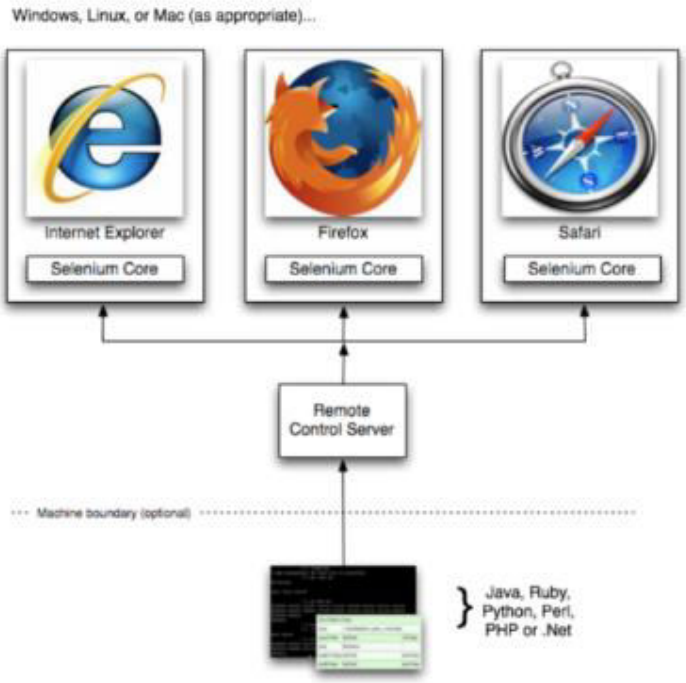
Then, another ThoughtWorks' engineer, Paul Hammant decided to create a server that will

act as an HTTP proxy to trick the browser into believing that Selenium Core and the web application being tested belong to the same domain, thus making RC a two-component tool.

- Selenium RC Server
- Selenium RC Client (*Library containing the programming language code)*

RC can support the following programming languages:

- Java
- C#
- PHP
- Python
- Perl
- Ruby



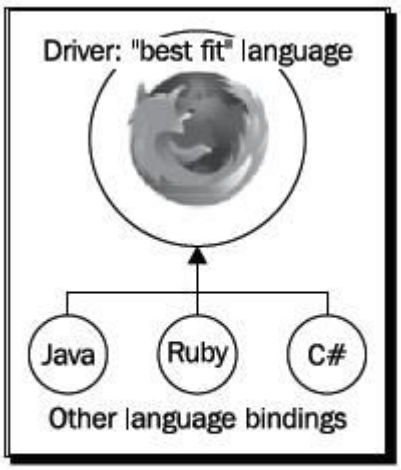
Selenium WebDriver

Founded by Simon Stewart in 2006, ThoughtWorks consultant in Australia. Selenium WebDriver was the first cross-platform testing framework that would control the browser at the OS level. Selenium WebDriver is a successor to Selenium RC. Selenium WebDriver accepts commands (sent in Selenium or via a Client API) and sends them to a browser.

This is implemented through a browser-specific driver, which sends commands to a browser and retrieves the results. Each driver launches and accesses a browser application. Different WebDrivers are:

- Firefox Driver (Gecko Driver)
- Chrome Driver
- Internet Explorer Driver

- MicroEdge
- Opera Driver
- Safari Driver
- HTML Unit Driver



Benefits of Selenium WebDriver

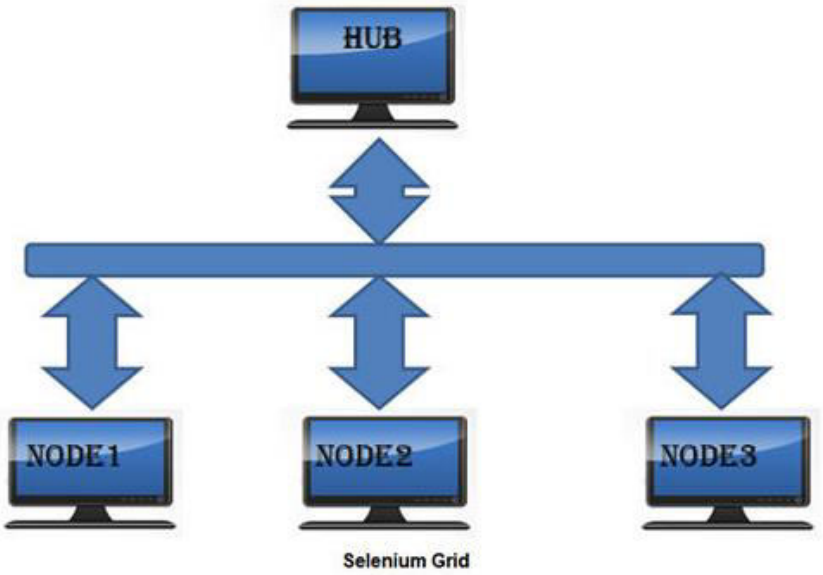
- Selenium WebDriver supports seven programming languages: Java, C#, PHP, Ruby, Perl, Python, and .Net.
- It supports cross-browser interoperability that helps you perform testing on various browsers like Firefox, Chrome, IE, Safari, etc.
- Tests can be performed on different operating systems: Windows, Mac, Linux, Android, and iOS.
- Selenium WebDriver overcomes limitations of Selenium v1 like file upload, download, pop-ups, and dialog barrier

Selenium Grid

Selenium Grid is a testing tool that lets you run your tests on various machines against different browsers. It is part of the Selenium suite that specializes in running multiple tests across different browsers, operating systems, and machines. You can connect to it with Selenium Remote Control by stating the browser version, browser, and operating system as per your choice. You will be able to specify these values through Selenium Remote Control capabilities. With Selenium Grid, one server makes a move as a hub. Tests communicate to the hub to get access to browser instances. The hub has a list of servers that provide access to browser instances (WebDriver nodes) and lets tests use these instances.

Selenium Grid allows parallel testing and also allows managing different browser versions

and browser configurations centrally (instead of in each individual test). There are multiple online platforms that provide an online Selenium Grid that you can access to run your Selenium automation scripts. For example, you can use LambdaTest. Selenium Grid has more than 2,000 browser environments over which you can run your tests and truly automate cross-browser testing.



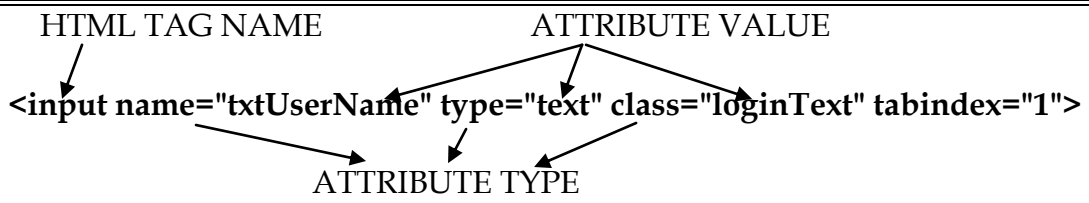
Selenium Grid

Selenium–Locators / Object Indentification

What is Locator? -- The locator can be termed as an address that identifies a web element uniquely within the webpage. Locators are the HTML properties of a web element. It is a command that tells Selenium IDE which GUI elements like – Text Box, Buttons, Check Boxes etc, it needs to operate on. Finding correct GUI elements is a prerequisite for creating an automation script but, accurate identification of GUI elements is much more difficult than it sounds. Sometimes, you might even end up working with incorrect GUI elements or no elements at all! Hence, using the right locator ensures that the tests are faster, more reliable or has lower maintenance over releases.

Selenium will identify the objects by using below locators

- **Name -** Select first element with the specified @name attribute.
- **ID -** Select element with the specified @id attribute.
- **Class Name -** Locate Element using a class Name .
- **TagName -** Locate Element using a Tag Name .
- **LinkText -** Select link (anchor tag) element which contains text matching the specified link text
- **Partial LinkText -** Select link (anchor tag) element which contains text matching the specified partial link text
- **CssSelector -** Css Select the element using css selectors. (Note : Prefered for IE Browser)
- **XPATH -** Locate an element using an XPath expression. (Note : Prefered for Any Browser)



*In any of the html code starting with **<** symbol is considered as HTML TAG name and which is having **=** symbol considered as Attributes

Locating an Element By Name:

This is an effective way to locate element with a name attribute. With this strategy, the first element with the value of the name attribute will be returned. If no element has a matching name attribute, then a NoSuchElementException will be raised.

Example:

<input name="register" class="required" type="text"/>
WebElement ele= driver.findElement(By.name("register"));

Locating an Element By ID:

The most efficient way and preferred way to locate an element on a web page is By ID. ID will be the unique on web page which can be easily identified.IDs are the safest and fastest locator option.

Example

<input id="email" class="required" type="text"/>
WebElement ele = driver.findElement(By.id("email"));

Locating an Element By Class Name:

There may be multiple elements with the same name, if we just use findElementByClassName,m make sure it is only one. If not the you need to extend using the classname and its sub elements.

Example:

<input id="email" class="required" type="text"/>
WebElement ele =driver.findElement(By.className("required"));

Locating an Element By tagName:

As the name specifies, this tagname locator in Selenium WebDriver is used to identify elements with Tag names like div tag, a tag etc. A common example of this usage could be locating all links on your homepage and verifying whether they are functional or broken.

Example:

driver.findElements(By.tagName(a));

Locating an Element By LinkText:

Finding an element with link text is very simple. But make sure, there is only one unique link on the web page. If there are multiple links with the same link text (such as repeated header and footer menu links), in such cases Selenium will perform action on the first matching element with link.

Example:

Downloads

WebElement ele = driver.findElement(By.linkText("Downloads"));

Locating an Element By PartialLinkText:

Locating element via Partial Link Text works similar to normal Link Text locator. The reason of using the Partial Link Text locator in Selenium WebDriver over the Link Text Locator is only due to the reason when you have a long linktext and you intent to use only partial text to perform further actions on it. Sometimes the intent of using this can also be to locate multiple links on a page with a common partial text.

Example:

Download selenium server

WebElement ele = driver.findElement(By.PartialLinkText("Download"));

Locating an Element By CSS:

CSS mainly used to provide style rules for the web pages and we can use for identifying one or more elements in the web page using css.We can you use Css Selectors to make sure scripts run with the same speed in IE browser. CSS selector is always the best possible way to locate complex elements in the page.

Example:

WebElement ele = driver.findElements(By.cssSelector("input[id=email']"));

Locating an Element By Xpath:

What is Xpath – group of nodes

XPath is the language used when locating XML (Extensible Markup Language) nodes. Since HTML can be thought of as an implementation of XML, we can also use XPath in locating HTML elements.

Advantage: It can access almost any element, even those without class, name, or id attributes.

Disadvantage: It is the most complicated method of identifying elements because of too many different rules and considerations.

Diffirent Types of Xpath in SLEENIUM

1.Absolute Xpath

2.Relative Xpath

Absolute XPath	Relative XPath
It starts with a single forward slash (/).	It starts with a double forward slash (//).
'/ ' instructs the XPath engine to search for the element with reference to the root node.	'//' instructs the XPath engine to search for a matching element anywhere in the DOM structure.
Element identification is faster compared to relative XPath.	Takes more time to identify the element as only a partial path is specified.
Even with the slightest change to the HTML DOM structure (e.g. adding a tag or removing one), absolute XPath would fail.	Relative XPaths are shorter and are less likely to change, making them more dependable.
E.g., /html/head/body/div[2]/form/input	E.g., //input[@name='username']

Diffirent Ways of Xpath in Selenium

We have different ways to identify the object using Xpath

1.Absolute Xpath

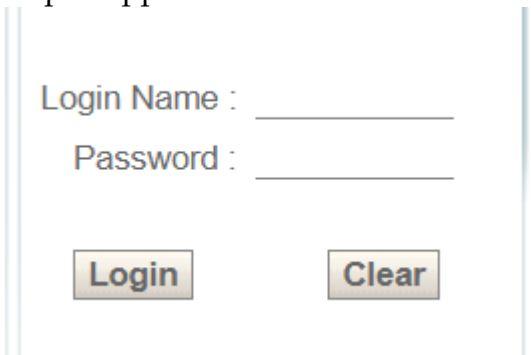
2.Relative Xpath

Absolute Xpath : Any xpath started with html tag consider as Absolute Xpath. Absolute Xpath prefixed with "/".

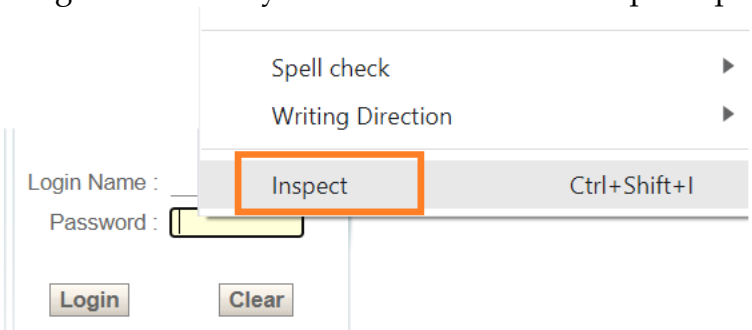
Example:

Steps to get generate Relative Xpath:

1.Open application in chrome browser

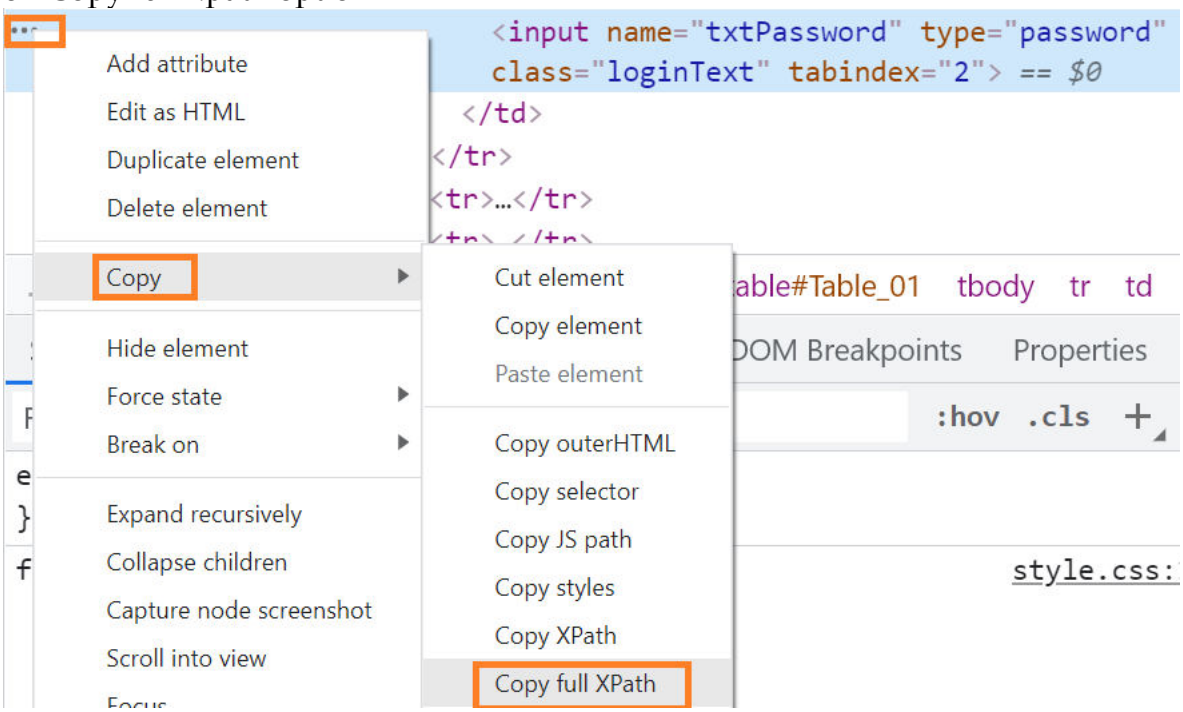


2.Right click on any element and click on Inspect option



3.click on 3 doted lines on highlited html code and Navigate to copy option and then click

on Copy full xpath option



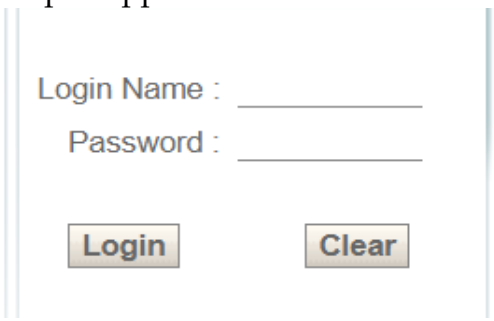
4.paste in text file ,absolute xpath will be generated.

/html/body/form/table[1]/tbody/tr/td[2]/table/tbody/tr[1]/td[2]/table/tbody/tr[3]/td[2]/input

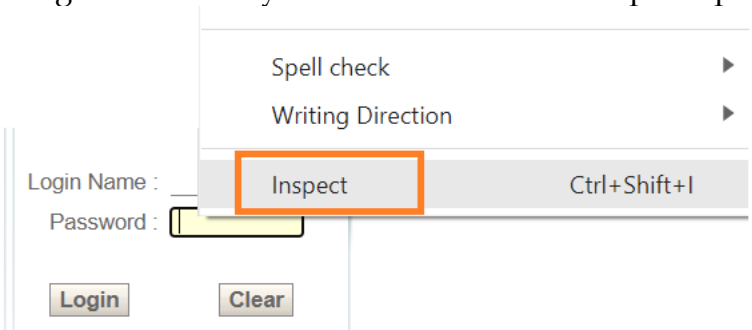
Relative Xpath: Any Xpath started with any unique attribute consider as Relative Xpath. Relative Xpath prefixed with "//".

Steps to get generate Relative Xpath:

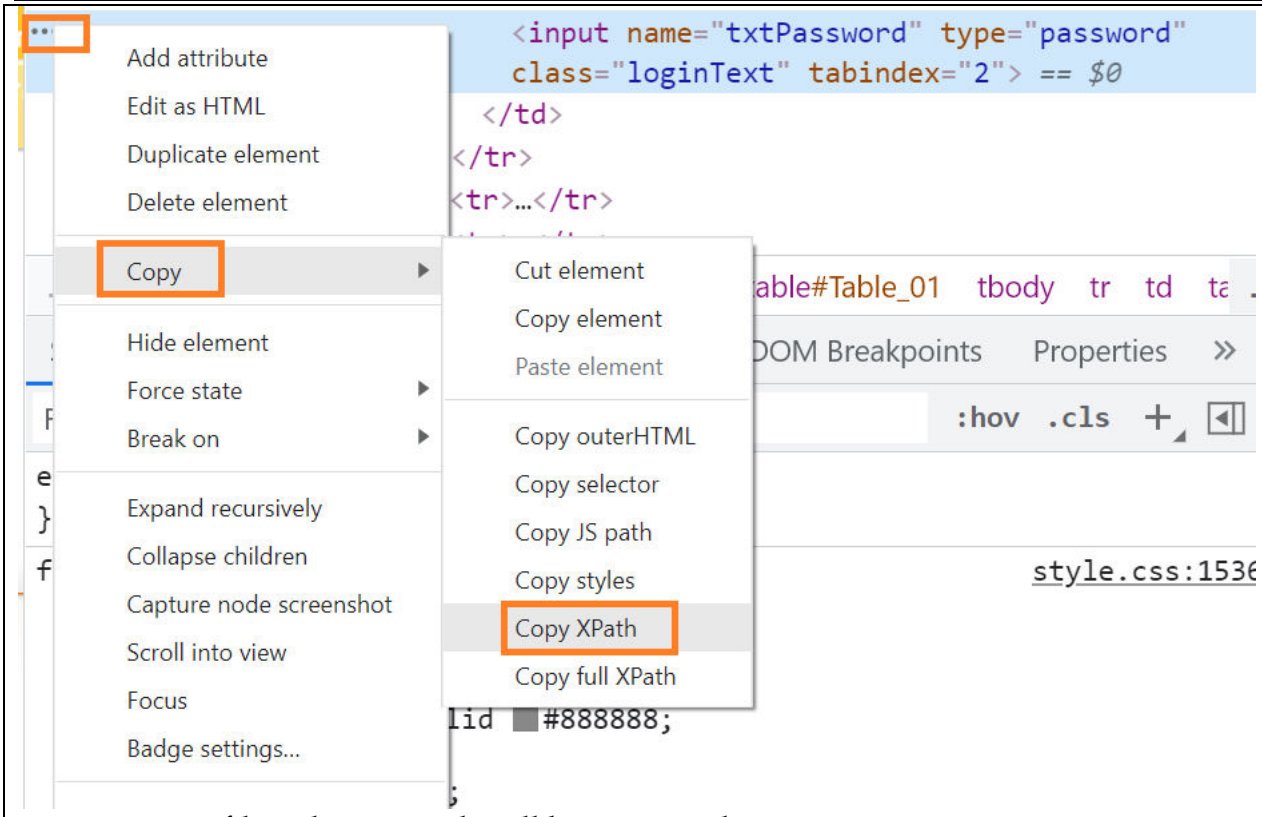
1.Open application in chrome browser



2.Right click on any element and click on Inspect option



3.click on 3 doted lines on highlited html code and Navigate to copy option and then click on Copy xpath option



4.paste in text file ,relative xpath will be generated.
//*[@id="Table_01"]/tbody/tr[1]/td[2]/table/tbody/tr[3]/td[2]/input

Or

We can write relative xpath manually by refering the html code with below syntax.

Syntax: //htmltagname[@attribute Type='attribute Value']/all nodes info:

Note :

highlited line is consider ending point to stop grouping of the nodes.

From highlited node we need to refer upper level and start grouping the nodes from any unique attribute.

In the below code **<table id="Table_01"** consider as unique attribute so started grouping the nodes from that step and ended at highlited line.

```
<td width="60%">
  <table id="Table_01" width="717" height="379" border="0" cellpadding="0"
  cellspacing="0">
    <tbody>
      <tr>
        <td rowspan="6">…</td>
        <td rowspan="5" valign="top">
          <img src="themes/orange/pictures/orange_new_13.jpg" width="167" height="18
          0">
          <table width="100%" border="0" cellspacing="0" cellpadding="3">
            <tbody>
              <tr>…</tr>
              <tr>…</tr>
              <tr>
                <td align="right" class="bodyTXT">Password : </td>
                <td>
                  <input name="txtPassword" type="password" class="loginText" tabindex=
                  "2"> == $0
                </td>
```

Example:

Relative xpath for above HTML code

//table[@id='Table_01']/tbody/tr[1]/td[2]/table/tbody/tr[2]/td[2]/input

Another approach of Xpath is Attributes.

Xpath with attributes

Xpath with attributes will use attributes to identify the objects

1.Xpath with single attribute

Syntax: //htmltagname[@attribute Type='attribute Value']

//input[@name='txtUserName']

2.Xpath with Multiple attribute

Syntax:

//htmltagname[@attributeType='attributeValue'][@attributeType='attributeValue']

//input[@name='txtUserName'][@type='text']

3.Xpath with text attribute

Syntax: //htmltagname[text()='text details']

4.Xpath with Contains

contains() can be used to find an element by specifying any partial value of the attribute. This function comes handy when attribute value changes dynamically on page reload.

Syntax: //htmltagname[contains(@attributeType,' attribute value')]

5.Xpath with Starts with

Starts-with() can be used to find an element by specifying a partial value (prefix) of the attribute. This function comes handy when attribute value changes dynamically on page reload.

Syntax: //htmltagname[starts-with(@attributeType, 'attribute value')]

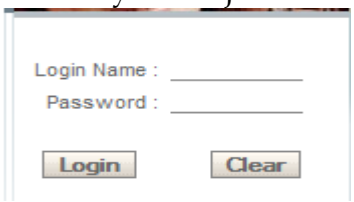
Tools for Object Indentification in Selenium

1.Chrome Browser – Inspect Option(Right click on any element and select Inspect

option) ,OSpy
2.Firefox–InspectElement(Q)-(Right click on any element and select Inspect option),Firebug ,Firepath
3.IE Browser– Developer Tools(Click on Tools and select Developer Tools option)

Assignment On - HRMS Project Object Identification

→Identify the objects for the screens with help of below html code. -- Login Page :



Username Textbox

```
▼<tr>
  ▶<td width="20%">…</td>
  ▼<td width="60%">
    ▼<table id="Table_01" width="717" height="379" border="0" cellpadding="0"
    cellspacing="0">
      ▼<tbody>
        ▼<tr>
          ▶<td rowspan="6">…</td>
          ▼<td rowspan="5" valign="top">
              <img src="themes/orange/pictures/orange_new_13.jpg" width="167" height=
              "180">
            ▼<table width="100%" border="0" cellspacing="0" cellpadding="3">
              ▼<tbody>
                ▶<tr>…</tr>
                ▼<tr>
                    <td align="right" class="bodyTXT">Login Name : </td>
                  ▼<td>
                      <input name="txtUserName" type="text" class="loginText" tabindex="1">
                    </td>
                  </tr>
```

Relative Xpath	
Xpath with Single attribute	
Xpath with Contains	
Xpath with Starts-With	
Xpath with Multiple attribute	

Password Textbox

```
▶<td width="20%">…</td>
▼<td width="60%">
  ▼<table id="Table_01" width="717" height="379" border="0" cellpadding="0" cellspacing=
  "0">
    ▼<tbody>
      ▼<tr>
        ▶<td rowspan="6">…</td>
        ▼<td rowspan="5" valign="top">
            <img src="themes/orange/pictures/orange_new_13.jpg" width="167" height="180">
          ▼<table width="100%" border="0" cellspacing="0" cellpadding="3">
            ▼<tbody>
              ▶<tr>…</tr>
              ▶<tr>…</tr>
              ▼<tr>
                  <td align="right" class="bodyTXT">Password : </td>
                ▼<td>
                    <input name="txtPassword" type="password" class="loginText" tabindex=
                    "2"> == $0
                  </td>
                </tr>
```

Relative Xpath	
Xpath with Single attribute	
Xpath with Contains	
Xpath with Starts-With	
Xpath with Multiple attribute	

Login button

```
▶<td width="20%">…</td>
▼<td width="60%">
  ▼<table id="Table_01" width="717" height="379" border="0" cellpadding="0" cellspacing=
  "0">
    ▼<tbody>
      ▼<tr>
        ▶<td rowspan="6">…</td>
        ▼<td rowspan="5" valign="top">
            <img src="themes/orange/pictures/orange_new_13.jpg" width="167" height="180">
          ▼<table width="100%" border="0" cellspacing="0" cellpadding="3">
            ▼<tbody>
              ▶<tr>…</tr>
              ▶<tr>…</tr>
              ▶<tr>…</tr>
              ▼<tr>
                ▼<td height="40" valign="bottom" align="center">
                    <input type="Submit" name="Submit" value="Login" class="button"
                    tabindex="3"> == $0
                  </td>
```

Relative Xpath	
Xpath with Single attribute	
Xpath with Contains	

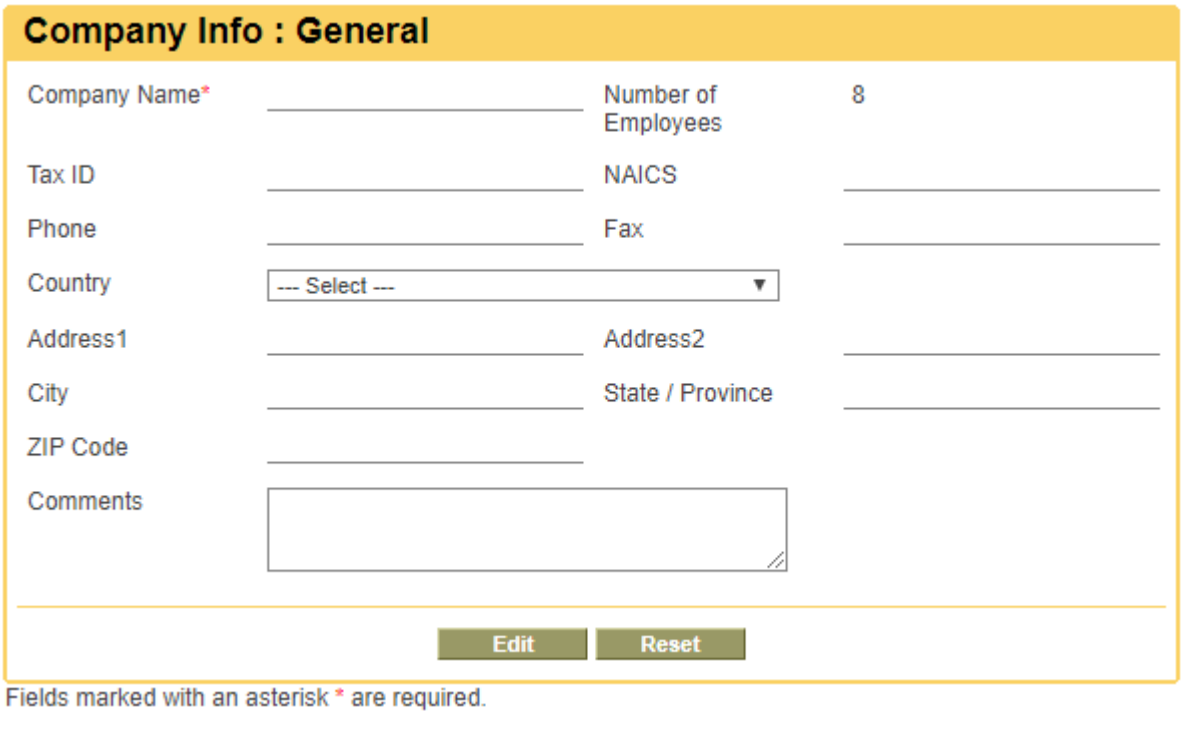
Xpath with Starts-With	
Xpath with Multiple attribute	

Clear button

```
<td width="60%">
  <table id="Table_01" width="717" height="379" border="0" cellpadding="0" cellspacing="0">
    <tbody>
      <tr>
        <td rowspan="6">…</td>
        <td rowspan="5" valign="top">
          <img src="themes/orange/pictures/orange_new_13.jpg" width="167" height="180">
          <table width="100%" border="0" cellspacing="0" cellpadding="3">
            <tbody>
              <tr>…</tr>
              <tr>…</tr>
              <tr>…</tr>
              <tr>
                <td height="40" valign="bottom" align="center">…</td>
                <td align="center" valign="bottom">
                  <input type="reset" name="clear" value="Clear" class="button" tabindex="4"> == $0
                </td>
```

Relative Xpath	
Xpath with Single attribute	
Xpath with Contains	
Xpath with Starts-With	
Xpath with Multiple attribute	

Company General Info:



Company Name Textbox

```
▶<head>…</head>
▼<body>
  ▼<div class="formpage2col">
      <div id="status"></div>
    ▼<div class="outerbox">
      ▶<div class="top">…</div>
      ▼<div class="maincontent">
        ▶<div class="mainHeading">…</div>
        ▼<form name="frmGenInfo" id="frmGenInfo" method="post" onsubmit="return
        validate()" action="/orangehrm-2.6/lib/controllers/CentralController.php?
        uniqcode=GEN">
            <input type="hidden" value="e1805f291ed654a53ba0ddd0bcd4708f" name="token">
            <input type="hidden" name="STAT" value="EDIT">
          ▶<label for="txtCompanyName">…</label>
            <input id="txtCompanyName" name="txtCompanyName" type="text" class=
            "formInputText" value maxlength="250"> == $0
            <span class="formLabel">Number of Employees</span>
```

Relative Xpath	
Xpath with Single attribute	
Xpath with Contains	
Xpath with Starts-With	
Xpath with Multiple attribute	

Number of Employees

```
▶<div class="top">…</div>
▼<div class="maincontent">
  ▶<div class="mainHeading">…</div>
  ▼<form name="frmGenInfo" id="frmGenInfo" method="post" onsubmit="return
  validate()" action="/orangehrm-2.6/lib/controllers/CentralController.php?
  uniqcode=GEN">
      <input type="hidden" value="e1805f291ed654a53ba0ddd0bcd4708f" name="token">
      <input type="hidden" name="STAT" value="EDIT">
    ▶<label for="txtCompanyName">…</label>
      <input id="txtCompanyName" name="txtCompanyName" type="text" class=
      "formInputText" value maxlength="250">
      <span class="formLabel">Number of Employees</span>
      <span class="formValue">8</span> == $0
```

Relative Xpath	
Xpath with Single attribute	
Xpath with Contains	
Xpath with Starts-With	
Xpath with Multiple attribute	

Phone number

```
▶<div class="mainHeading">…</div>
▼<form name="frmGenInfo" id="frmGenInfo" method="post" onsubmit="return
validate()" action="/orangehrm-2.6/lib/controllers/CentralController.php?
uniqcode=GEN">
    <input type="hidden" value="e1805f291ed654a53ba0ddd0bcd4708f" name="token">
    <input type="hidden" name="STAT" value="EDIT">
  ▶<label for="txtCompanyName">…</label>
    <input id="txtCompanyName" name="txtCompanyName" type="text" class=
    "formInputText" value maxlength="250">
    <span class="formLabel">Number of Employees</span>
    <span class="formValue">8</span>
    <br class="clear">
    <label for="txtTaxID">Tax ID</label>
    <input id="txtTaxID" name="txtTaxID" type="text" class="formInputText" value
    maxlength="25">
    <label for="txtNAICS">NAICS</label>
    <input id="txtNAICS" name="txtNAICS" type="text" class="formInputText" value
    maxlength="15">
    <br class="clear">
    <label for="txtPhone">Phone</label>
    <input id="txtPhone" name="txtPhone" type="text" class="formInputText" value
    maxlength="20"> == $0
    <label for="txtFax">Fax</label>
```

Relative Xpath	
Xpath with Single attribute	
Xpath with Contains	
Xpath with Starts-With	
Xpath with Multiple attribute	

Country

```
▶<div class="mainHeading">…</div>
▼<form name="frmGenInfo" id="frmGenInfo" method="post" onsubmit="return
validate()" action="/orangehrm-2.6/lib/controllers/CentralController.php?
uniqcode=GEN">
    <input type="hidden" value="e1805f291ed654a53ba0ddd0bcd4708f" name="token">
    <input type="hidden" name="STAT" value="EDIT">
  ▶<label for="txtCompanyName">…</label>
    <input id="txtCompanyName" name="txtCompanyName" type="text" class=
    "formInputText" value maxlength="250">
    <span class="formLabel">Number of Employees</span>
    <span class="formValue">8</span>
    <br class="clear">
    <label for="txtTaxID">Tax ID</label>
    <input id="txtTaxID" name="txtTaxID" type="text" class="formInputText" value
    maxlength="25">
    <label for="txtNAICS">NAICS</label>
    <input id="txtNAICS" name="txtNAICS" type="text" class="formInputText" value
    maxlength="15">
    <br class="clear">
    <label for="txtPhone">Phone</label>
    <input id="txtPhone" name="txtPhone" type="text" class="formInputText" value
    maxlength="20">
    <label for="txtFax">Fax</label>
    <input id="txtFax" name="txtFax" type="text" class="formInputText" value
    maxlength="20">
    <br class="clear">
    <label for="cmbCountry">Country</label>
  ▶<select id="cmbCountry" name="cmbCountry" class="formSelect countrySelect"
  onchange="onCountryChange(this.value);">…</select> == $0
    <br class="clear">
```

Relative Xpath	
Xpath with Single attribute	
Xpath with Contains	
Xpath with Starts-With	
Xpath with Multiple attribute	

Company Info Locations

Add button

```
▼<div class="maincontent">
  ▼<form name="standardView" method="post" action="/orangehrm-2.6/lib/controllers/
  CentralController.php?uniqcode=LOC&VIEW=MAIN&sortField=0&sortOrder0=ASC">
    ▶<div class="mainHeading">…</div>
      <input type="hidden" value="65e7076da247eca334a12f533473ea16" name="token">
      <input type="hidden" name="captureState" value>
      <input type="hidden" name="delState" value>
      <input type="hidden" name="pageNO" value="1">
    ▶<div class="searchbox">…</div>
    ▼<div class="actionbar">
      ▼<div class="actionbuttons">
          <input type="button" class="plainbtn" onclick="returnAdd();" onmouseover=
          "this.className='plainbtn plainbtnhov'" onmouseout=
          "this.className='plainbtn'" value="Add"> == $0
          <input type="button" class="plainbtn" onclick="returnDelete();" onmouseover=
```

Relative Xpath	
Xpath with Single attribute	
Xpath with Contains	
Xpath with Starts-With	
Xpath with Multiple attribute	

Delete button

```
▼<div class="maincontent">
  ▼<form name="standardView" method="post" action="/orangehrm-2.6/lib/controllers/
  CentralController.php?uniqcode=LOC&VIEW=MAIN&sortField=0&sortOrder0=ASC">
    ▶<div class="mainHeading">…</div>
      <input type="hidden" value="65e7076da247eca334a12f533473ea16" name="token">
      <input type="hidden" name="captureState" value>
      <input type="hidden" name="delState" value>
      <input type="hidden" name="pageNO" value="1">
    ▶<div class="searchbox">…</div>
    ▼<div class="actionbar">
      ▼<div class="actionbuttons">
          <input type="button" class="plainbtn" onclick="returnAdd();" onmouseover=
          "this.className='plainbtn plainbtnhov'" onmouseout=
          "this.className='plainbtn'" value="Add">
          <input type="button" class="plainbtn" onclick="returnDelete();" onmouseover=
          "this.className='plainbtn plainbtnhov'" onmouseout=
          "this.className='plainbtn'" value="Delete"> == $0
        </div>
```

Relative Xpath	
Xpath with Single attribute	
Xpath with Contains	
Xpath with Starts-With	
Xpath with Multiple attribute	

Common Interview Questions on Selenium

1. What is Automation Testing
2. What are the benefits of Automation Testing
3. On which Testing phase automation will be started
4. what are the Pre-Requisites to start automation
5. Explain Tester Roles and Responsbilities
6. Why should Selenium be selected as a test tool
7. Write Xpath for Gmail Inbox(50) but after some time it will change to Inbox(55)
8. What is Selenium? What are the different Selenium components?
9. What are the different types of locators in Selenium?
10. Explain Tester day to day activities
11. Difference between Relative Xpath and Absolute Xpath
12. What are the limitations of Selenium
13. Explain the syntax for Xpath with Contains and Startswith
14. Diffirence between Xpath with Contains and Startswith
15. Why should we hire you

Core Java

History of JAVA

JAVA is a distributed technology developed by James Gosling, Patric Naugton, etc., at Sun Micro System ((which has been acquired by Oracle Corporation)) has released lot of rules for JAVA and those rules are implemented by JavaSoft Inc, USA (which is the software division of Sun Micro System) in the year 1990. The original name of JAVA is OAK (which is a tree name). In the year 1995, OAK was revised and developed software called JAVA (which is a coffee seed name). The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

JAVA released to the market in three categories J2SE (JAVA 2 Standard Edition)[considered as Core Java], J2EE (JAVA 2 Enterprise Edition) and J2ME (JAVA 2 Micro/Mobile Edition).

Java terminology:

Before we start learning Java, let's get familiar with common java terms.

Java Development Kit(JDK)

While explaining JVM and bytecode, I have used the term JDK. Let's discuss about it. As the name suggests this is complete java development kit that includes JRE (Java Runtime Environment), compilers and various tools like JavaDoc, Java debugger etc.

In order to create, compile and run Java program you would need JDK installed on your computer.

Java Runtime Environment(JRE)

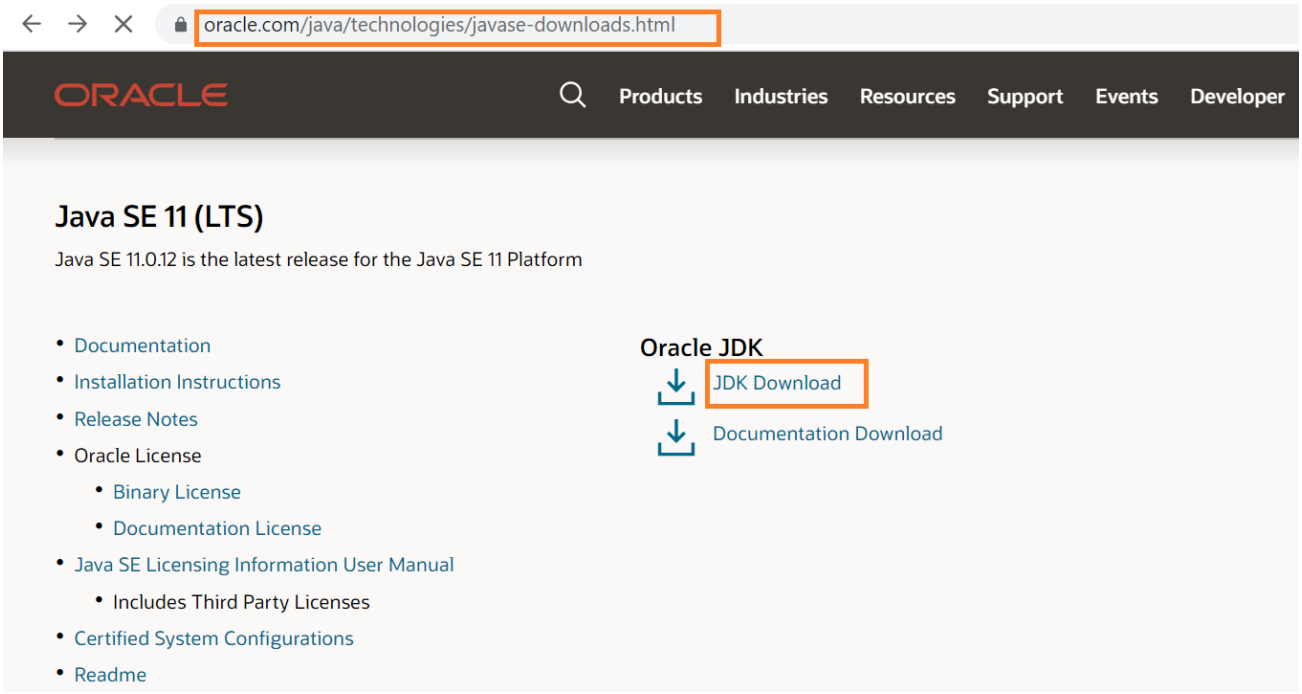
JRE is a part of JDK which means that JDK includes JRE. When you have JRE installed on your system, you can run a java program however you won't be able to compile it. JRE includes JVM, browser plugins and applets support. When you only need to run a java program on your computer, you would only need JRE.

Compiler(javac) converts source code (.java file) to the byte code(.class file). As mentioned above, JVM executes the bytecode produced by compiler. This byte code can run on any platform such as Windows, Linux, Mac OS etc. Which means a program that is compiled on windows can run on Linux and vice-versa. Each operating system has different JVM, however the output they produce after execution of bytecode is same across all operating systems. That is why we call java as platform independent language.

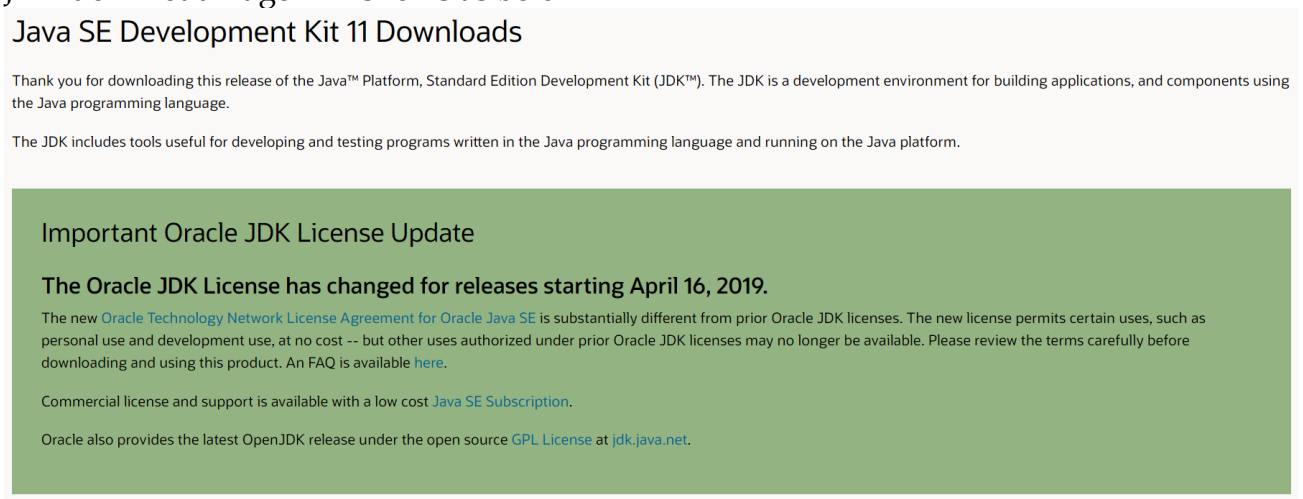
JAVA Installation Steps

Note: JRE will be installed automatically when you install JDK. You don't have to do anything extra to install JRE. Oracle has stopped providing JRE installer. Now JDK and JRE is a part of Standard Edition (SE)

Go to link - **https://www.oracle.com/java/technologies/javase-downloads.html** and click on **JDK download** based on your requirement.



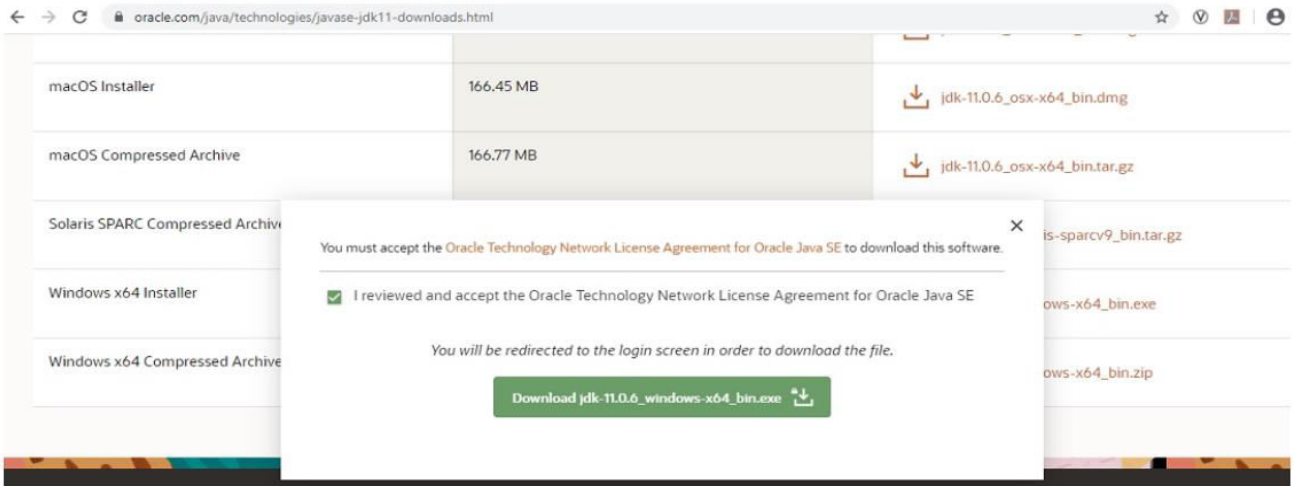
JDK download Page will shows as below



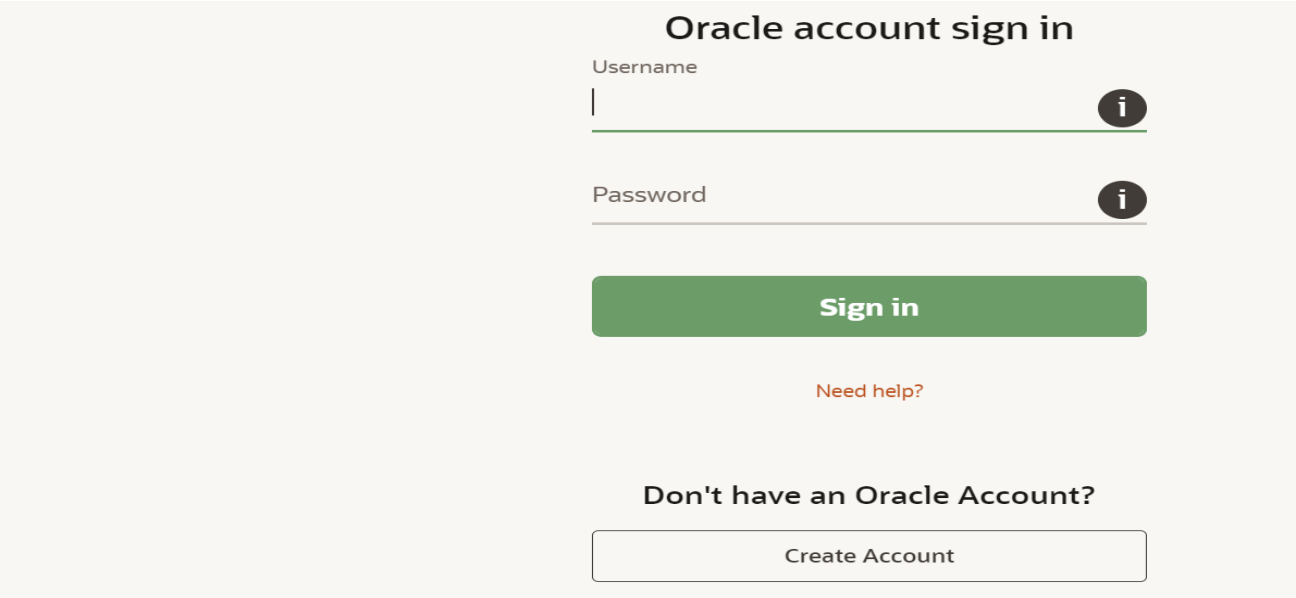
Scroll down and you will see the download options for windows which is an installer (.exe).

Linux x64 RPM Package	156.45 MB	jdk-11.0.12_linux-x64_bin.rpm
Linux x64 Compressed Archive	173.86 MB	jdk-11.0.12_linux-x64_bin.tar.gz
macOS Installer	167.69 MB	jdk-11.0.12_osx-x64_bin.dmg
macOS Compressed Archive	168.19 MB	jdk-11.0.12_osx-x64_bin.tar.gz
Solaris SPARC Compressed Archive	184.5 MB	jdk-11.0.12_solaris-sparcv9_bin.tar.gz
Windows x64 Installer	151.83 MB	jdk-11.0.12_windows-x64_bin.exe
Windows x64 Compressed Archive	171.27 MB	jdk-11.0.12_windows-x64_bin.zip

Click on required jdk based on your operating system then - you will be asked to accept a License agreement before the download begins. Accept the license to download the file.

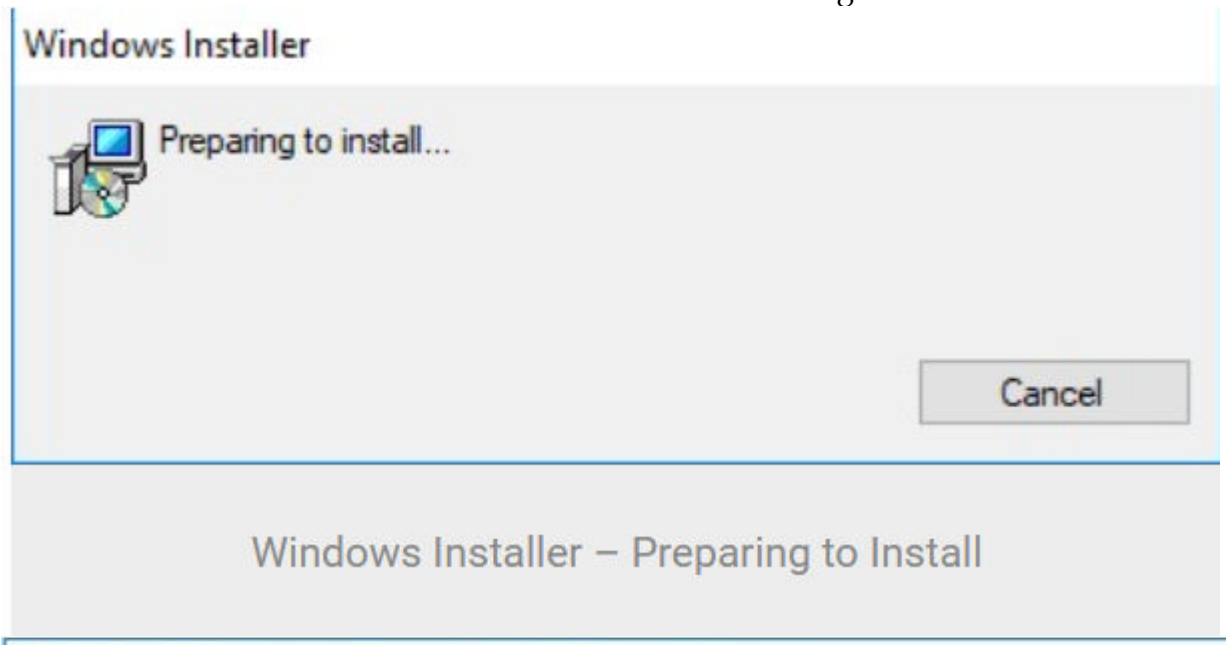


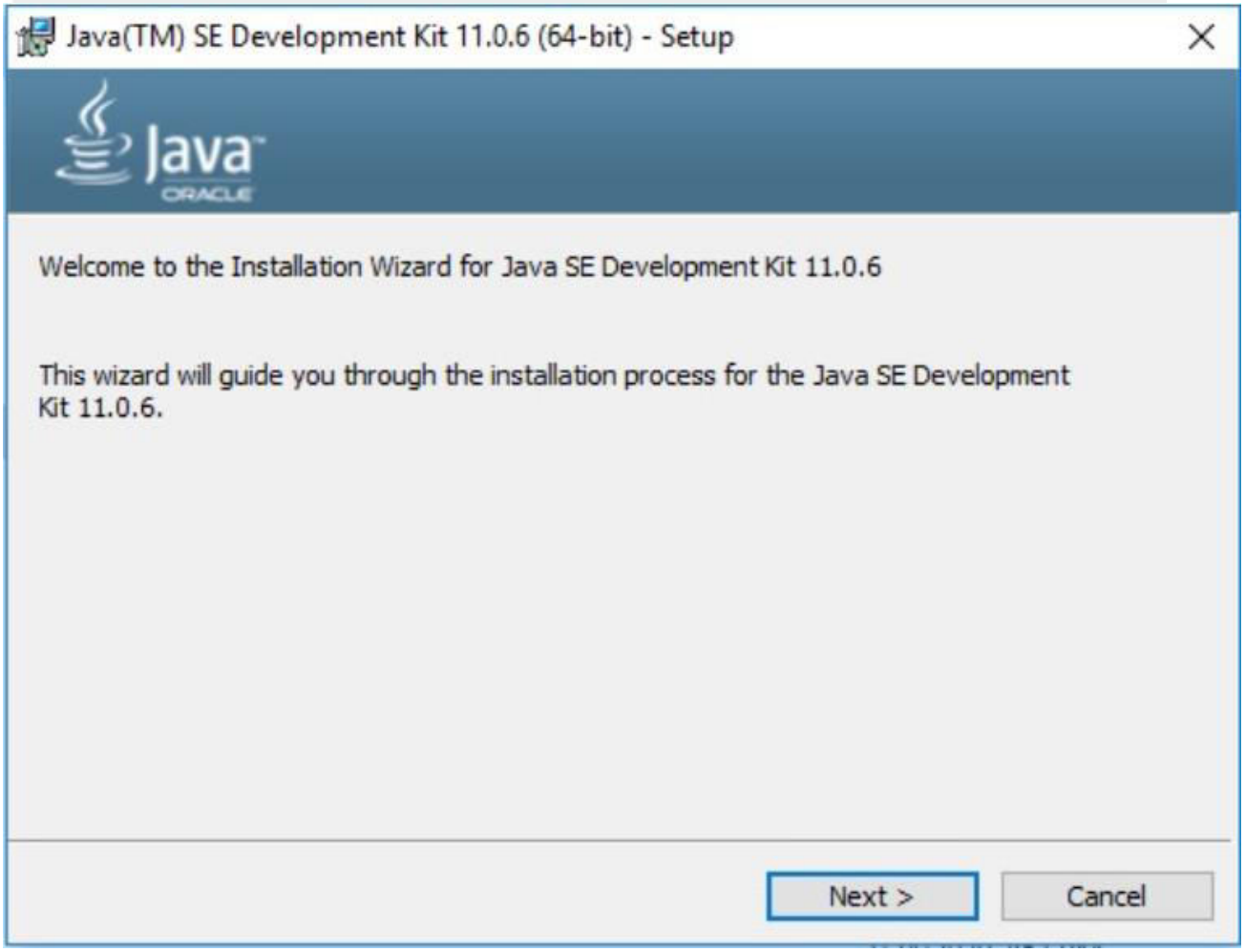
You will be asked to login to start the download. If you don't have an Oracle account, you can create here



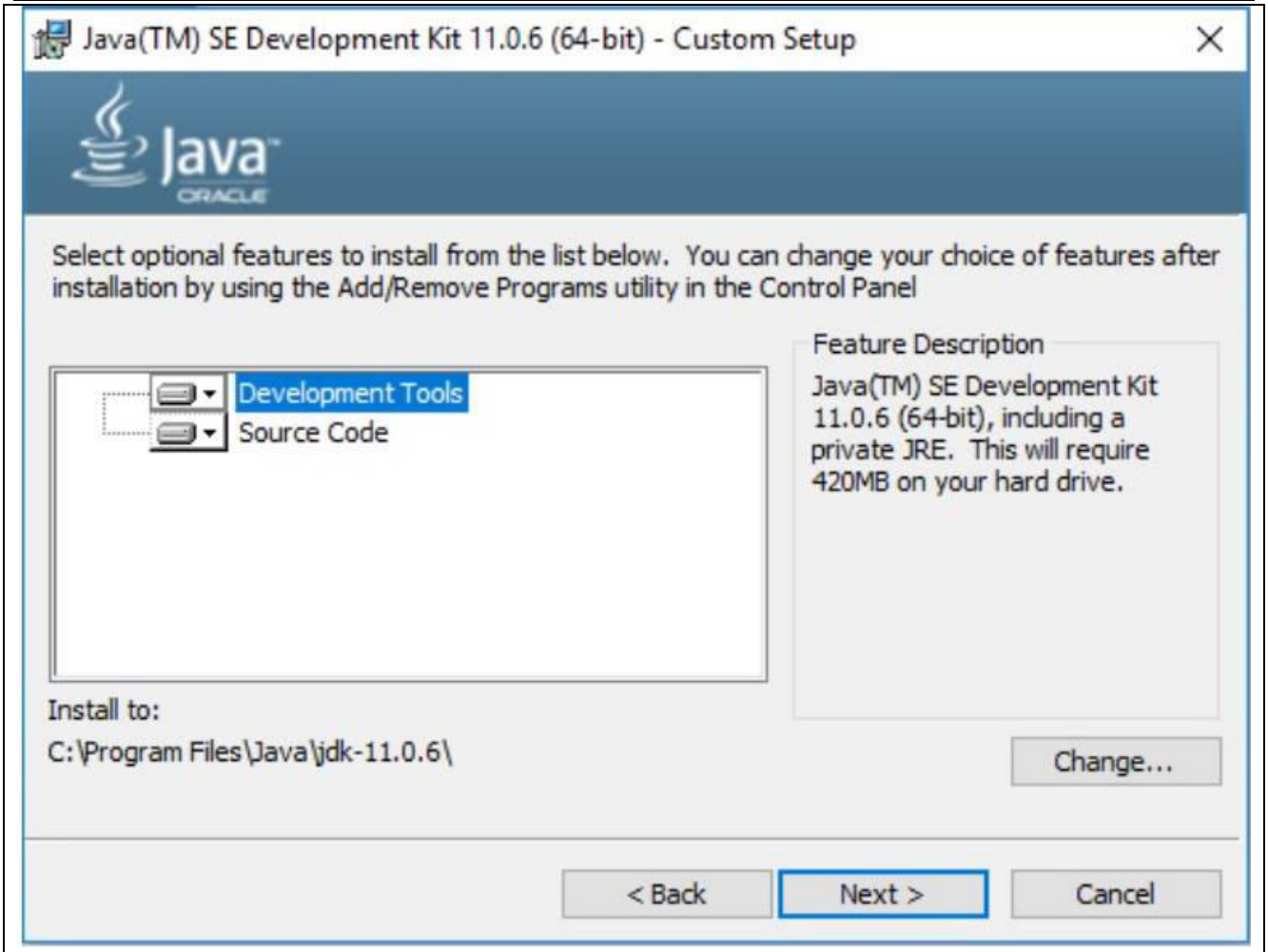
Run the installer (.exe file)

Perform double click on the .exe executable installer java file that you have downloaded from the official website.Then You will see the the following windows.

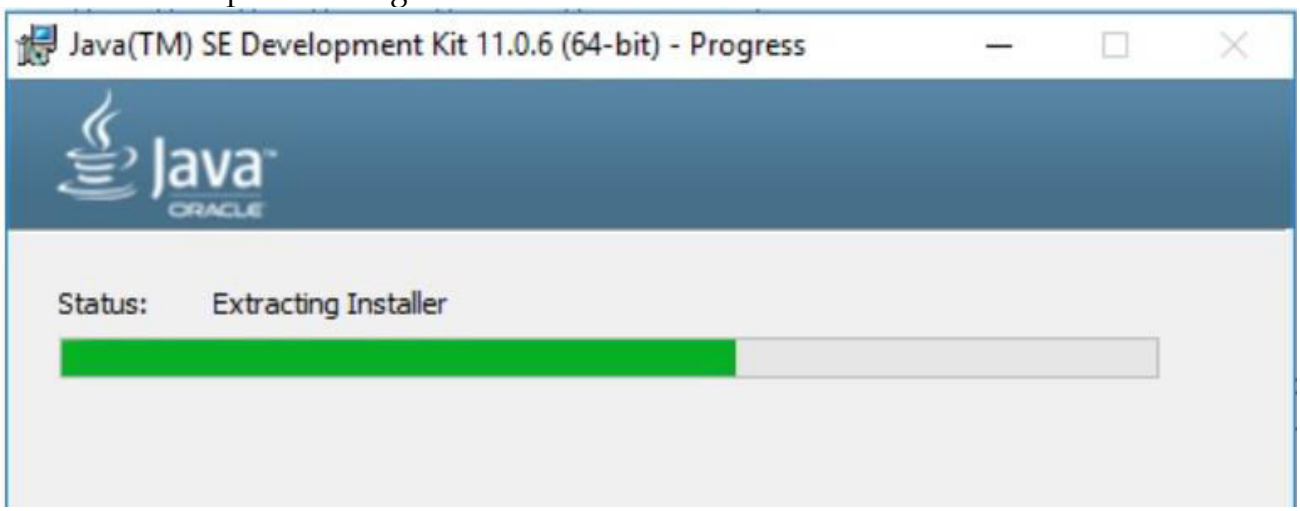




Click on next to continue and then you can change the installation directory if you want by clicking on change. Click next to continue.



Installation begins-Please wait for the process to complete.After sometime you will see installation complete message.



Click on close to complete the installation.

Note : if required perform environment varible setting based on the java path got installed.

Eclipse IDE Installation

EclipseIDE: Eclipse is an integrated development environment used in computer programming, and is the most widely used Java IDE. It contains a base workspace and an extensible plug-in system for customizing the environment and used to write the Java code.

Step 0: Complete JDK Installation-To use Eclipse, you need to first install Java Development Kit (JDK).

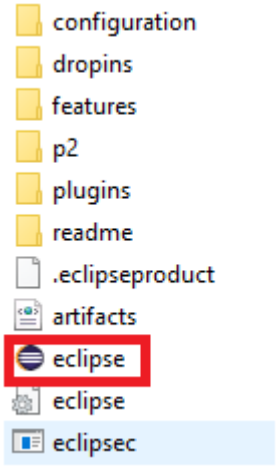
Step 1: Download Eclipse from http://www.eclipse.org/downloads/eclipse-packages/. choose the Eclipse IDE for Java EE Developers and click on 32-bit or 64-bit link

Note: Eclipse will be downloaded as Zipfile.

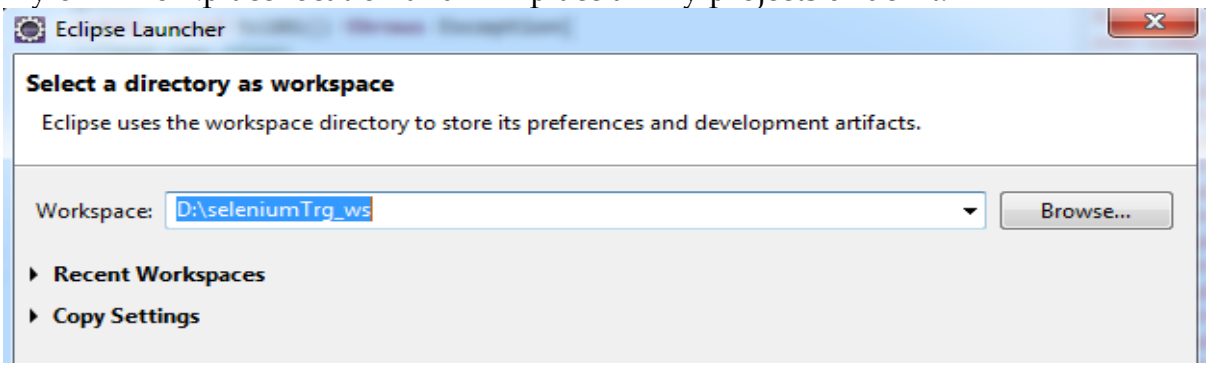
Step 2: Simply extract the downloaded file into a directory of your choice (e.g., "d:\myproject").

There is no need to run any installer. Moreover, you can simply delete the entire Eclipse directory when it is no longer needed (without running any un-installer). You are free to move or rename the directory. You can install (unzip) multiple copies

of Eclipse in the same machine.
Step 3: Since Eclipse IDE does not have any installer, there will be a file inside the Eclipse folder named eclipse.exe. You can double click on the file to run Eclipse. Note :(This step is not required, but it's strongly recommended.) Right-click the Eclipse Icon and press "Send To" -> "Desktop (Create Shortcut)." Now you will be able to launch Eclipse from your desktop.



Step 4: Create a workspace folder where you will contain all the program files you create. You can choose whatever place you want for your workspace, I like to choose my own workplace location and will place all my projects under it.



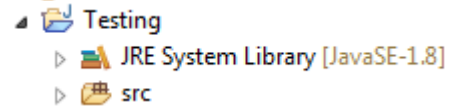
Step 5 : Now you will be able to see the welcome screen ,you can close welcome screen and then start writing the java code

About Java programs, it is very important to keep in mind the following points.
Case Sensitivity - Java is case sensitive which means identifier Hello and hello would have different meaning in Java.
Class Names - For all class names the first letter should be in Upper Case.
If several words are used to form a name of the class each inner words first letter should be in Upper Case. Example class MyFirstJavaClass
Method Names - All method names should start with a Lower Case letter.
If several words are used to form the name of the method, then each inner word's first letter should be in Upper Case. Example public void myMethodName ()
Program File Name - Name of the program file should exactly match the class name.
Example: Assume 'MyFirstJavaProgram' is the class name. Then the file should be saved as 'MyFirstJavaProgram.java'

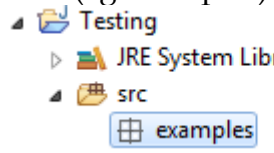
Steps To write First Java Program in Eclipse IDE :

1.Launch Eclipse

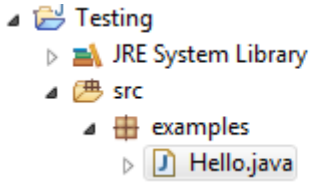
2.Create new Java Project(Navigation: File→New→Project→Select Java Folder→ Select Java Project option→Click on Next button→Provide any Project Name and click on Finish button→Project will be created in eclipse as below.)



3.Create new Package (Navigation: Perform Right click on Created project →Navigate to New and click on package option→Provide any package name(eg:examples) and click on Finish button→Package will be created as below)



4.Create new Class(Navigation : perform Right click on Created package→Navigate to New and click on Class option→Provide any class name (Hello) and click on Finish button→class will be created as below)




```
5  package examples;
6
7  public class Hello {
8
9  }
10
```

5.Create required methods /code

6. Run the program and able to see the output in eclipse console window.

Note : Any JAVA program if we want to develop then that should be developed with respective class only i.e., without class there is no java program

Class: A class can be defined as a template/ blue print that describe the behaviors/states that object of its type support. (or) "A class is a way of binding the data and associated methods in a single unit. (or) A Class is a combination of DataTypes and Data Variables

Syntax for defining a CLASS:

Class <clsname> {
Variable declaration;
Methods definition;
}

Class contains two parts namely variable declaration and method definitions. Variable

Declaration represents what type of data members which we use as a part of the class. Method definition represents the type of methods which we used as the path of the class to perform an operation.By making use of the variables, which are declared inside the class? Every operation in JAVA must be defined with in the class only i.e. outside definition is not possible.

Note : We have diffirent types of classes for now we need to understand below types

1.Instance classes (Need to create object to access methods from this class)
2.Static class (No need of to create object to access methods from this class ,We can access the method by using class name)

First Program in JAVA – To Print the statement as Welcome to JAVA

```
public class Hello{
   public static void main(String args[]) {
    System.out.println ("Welcome to JAVA "); // prints Welcome to JAVA
   }
```

Output : Welcome to JAVA

Description for above program

Public: This is access modifier keyword which tells compiler access to class. Various values of access modifiers can be public, protected, private or default (no value).

Class: This keyword used to declare class. Name of class (Hello) followed by this keyword.

"public static void main (String [ ] args)": java program processing starts from the main () method which is a mandatory part of every java program.

Its method (Function) named main with string array as argument.

public : Access Modifier

static: static is reserved keyword which means that a method is accessible and usable even though no objects of the class exist.

void: This keyword declares nothing would be returned from method. Method can return any primitive or object.

Method content inside curly braces. { }

System.out.println("Welcome to JAVA") : This statement is used to print any information

System: It is name of Java utility class.

out: It is an object which belongs to System class.

println: It is utility method name which is used to send any String to console.

Write a Java program to print stmt as – "My target to get job is 2 months"

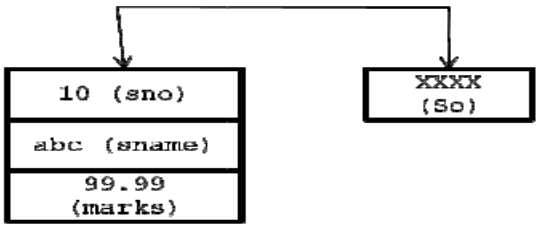
Method: A method is a program module that contains a series of statements that carry out a task.Any class can contain an unlimited number of methods, and each method can be called an unlimited number of times. The syntax to declare method is given below.

```
public void methodname(){
//statements to print;
}
Eg: public void m1(){
System.out.println("Method 1 executed");
}
```

There are two types of methods-
Builtin / Pre-Defined methods: Defined by the language
User-Defined : Defined by the user

Object - object is an instance of a class.Instance (instance is a mechanism of allocating sufficient amount of memory space for dataMembers of a class)

Syntax-1 for defining an OBJECT:
<Clsname> objname = new <clsname ()>
Student so = new student ();



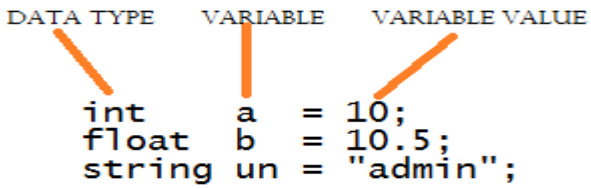

```
student So; //student object declaration//
So = new student(); //student object referencing//
```

Sample program to create Class ,Method and Object

```
public class Hello {
Public void m1() {
System.out.println ("m1 method executed");
}
public void m2() {
System.out.println ("m2 method executed");
}
public void m3() {
System.out.println ("m3 method executed");
}
public static void main (String args []) {
System.out.println("Welcome to Selenium Training");
Hello  m = new Hello();
m.m1 ();//Accessing method by using object
m.m2 ();
m.m3 ();
}
}
```

Create a class and methods to print below stmt by calling methods with object
Class name : SeleniumComponents
Method Name : seleniumIde – PerformRecord and Palyback
Method Name : seleniumWebdriver – To develop Automation Scripts
Method Name : seleniumGrid – To execute Scripts in Multipul Browsers and Systems

Data Types and Variables



Data Types :
Data type defines the values that a variable can take, for example if a variable has int data type, it can only take integer values. In java we have two categories of data types
1.Primitive Data Types
2.Reference/Object Data Types
Primitive Data Types –Byte ,Short ,int,long,float,double,boolean,char
Reference Data Types:
Reference variables are created using defined constructors of the classes. They are used to access objects. These variables are declared to be of a specific type that cannot be changed. For example, Employee, Puppy etc.
Class objects and various types of array variables come under reference data type.
Default value of any reference variable is null.
A reference variable can be used to refer to any object of the declared type or any compatible type.
Example : Animal animal = new Animal("giraffe");
Java language supports few special escape sequences for String and char literals as well.
Variables : A variable is a name which is associated with a value that can be changed. For example when I write int i=10; here variable name is i which is associated with value 10, int is a data type that represents that this variable can hold integer values.
There are three kinds of variables in Java:
In Java, all variables must be declared before they can be used
Local variables – Defined with in the method and able to access with in that method only
Instance variables – Defined outside the method and with in the class
Static/class variables –Need to use the keyword as Static

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.
Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals, or characters in these variables.

Sample program for - DataTypes , Variables and Static method

```
public class Dt_Var {
        int b=20; //instance variables
        static int empid = 101; //static variables
      public void m1()    {
            int a = 10; //local variables
            System.out.println("M1 Executed");
            System.out.println("Local varible " + a );
      }
      public void m2()    {
            System.out.println("M2 Executed");
            System.out.println("instance varible " + b);
            System.out.println("Static varible " + empid);
      }
      public void m3()    {
            System.out.println("M3 Executed");
            System.out.println("instance varible " + b);
            System.out.println("Static varible " + empid);
      }
      public static void st()       {
            System.out.println("Static method executed");
      }
public static void main(String args[]){
      System.out.println("Main methed executed");
      Dt_Var m = new Dt_Var ();
      m.m1();//Accessing method by using object
      m.m3();
      m.m2();
Dv.st();//no need of to create any object.Accessing method by using classname
   } }
```

Java Access Modifiers : The Java language has a wide variety of modifiers, including the following:

Java Access Modifiers & Non Access Modifiers

Java provides a number of access modifiers to set access levels for classes, variables, methods and constructors. The four access levels are:

Visible to the package. The default. No modifiers are needed.

Visible to the class only (private).

Visible to the world (public).

Visible to the package and all subclasses (protected).

Java provides a number of non-access modifiers to achieve many other functionality.

The static modifier for creating class methods and variables

The final modifier for finalizing the implementations of classes, methods, and variables.

The abstract modifier for creating abstract classes and methods.

The synchronized and volatile modifiers, which are used for threads.

Let's understand the access modifiers in Java by a simple table.

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

Operators : Java provides a rich set of operators to manipulate variables. We can divide all the Java operators into the following groups:

- Arithmetic Operators(- + * / % ++ --)
- Relational Operators (> < >= <= == !=)
- Bitwise Operators(& | ^ >> >>>)
- Logical Operators (&& || & |! ^)
- Assignment Operators(= , +=)
- Misc Operators (? :)

Arithmetic Operators

Operator	Name	Description	Example
+	Addition	Adds together two values	x + y
-	Subtraction	Subtracts one value from another	x - y
*	Multiplication	Multiplies two values	x * y
/	Division	Divides one value from another	x / y
%	Modulus	Returns the division remainder	x % y
++	Increment	Increases the value of a variable by 1	++x
--	Decrement	Decreases the value of a variable by 1	--x

assignment operators

Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
&=	x &= 3	x = x & 3
|=	x |= 3	x = x | 3
^=	x ^= 3	x = x ^ 3
>>=	x >>= 3	x = x >> 3
<<=	x <<= 3	x = x << 3

Comparison Operators

Operator	Name	Example
==	Equal to	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Logical operators

Operator	Name	Description	Example
&&	Logical and	Returns true if both statements are true	x < 5 && x < 10
||	Logical or	Returns true if one of the statements is true	x < 5 || x < 4
!	Logical not	Reverse the result, returns false if the result is true	!(x < 5 && x < 10)

Conditions (or) Control Flow Statements

When we need to execute a set of statements based on a condition then we need to use control flow statements. For example, if a number is greater than zero then we want to print "Positive Number" but if it is less than zero then we want to print "Negative Number". In this case we have two print statements in the program, but only one print statement executes at a time based on the input value. We will see how to write such type of conditions in the java program using control statements.

we will use four types of control statements in java programs based on the requirement:

a) if statement
b) if-else statement
c) if-else-if statement
d) nested if statement

The if Statement:

If statement consists a condition, followed by statement or a set of statements as shown below:

```
if(condition){
  Statement(s);
}
```

The statement gets executed only when the given condition is true. If the condition is false then the statements inside if statement body are completely ignored.

Example1:

```
public class Test {
   public static void main(String args[]){
      int x = 10;
      if( x < 20 ){
         System.out.print("This is if statement");
      }
   }
}
```

This would produce following result:

```
This is if statement
```

Example2:Write a program to print Student result status : studentmarks=80

if...else Statement:

Syntax :

```
if(condition) {
  Statement(s);
}
else {
  Statement(s);
}
```

The statements inside "if" would execute if the condition is true, and the statements inside "else" would execute if the condition is false.

Example: Write a program to print Student result status: totalMarks=35

```
public class ifelsestmt {
        public static void main(String[] args) {
          int totalMarks=35;
          if(totalMarks>=36){
            System.out.print("Student Passed");
          }
          else {
            System.out.print("Student failed");
          }
        }
      }
```

Example: Write a program to print status based on age details: age =10;
Example: Write a program to print voter eligibility : age =20;
Example: Write a program to print driving licience eligibility : age =20;
Example: Write a program to check a even number or Odd number : no=30;

else if Statement:

if-else-if statement is used when we need to check multiple conditions. In this statement we have only one "if" and one "else", however we can have multiple "else if". It is also known as if else if ladder. This is how it looks:

```
if(condition_1) {
  //if condition_1 is true execute this
  statement(s);
}
else if(condition_2) {
  //execute this if condition_1 is not met and condition_2 is met
  statement(s);
}
else if(condition_3) {
  // execute this if condition_1 & condition_2 are not met and condition_3 is met
  statement(s);
```

```
}
else {
  // if none of the condition is true then these statements gets executed
  statement(s);
}
```

Note: The most important point to note here is that in if-else-if statement, as soon as the condition is met, the corresponding set of statements get executed, rest gets ignored. If none of the condition is met then the statements inside "else" gets executed.

Example:

```
public class Test {
public static void main(String args[]){
int x = 30;
if( x = = 10 ){
System.out.print("Value of X is 10");
}else if( x = = 20 ){
System.out.print("Value of X is 20");
}else if( x = = 30 ){
System.out.print("Value of X is 30");
}else{
System.out.print("This is else statement");
}
}
}
```

Example: Write a program to print Grade details based on student marks: stdmarks=92

Example: Write a program to print status based on age details: age =10;

Nested if...else Statement:

When there is an if statement inside another if statement then it is called the nested if statement.

The structure of nested if looks like this:

```
if(condition_1) {
  Statement1(s);

  if(condition_2) {
    Statement2(s);
  }
}
```

Statement1 would execute if the condition_1 is true. Statement2 would only execute if both the conditions(condition_1 and condition_2) are true.

Example:

```
public class Test {
  public static void main(String args[]){
    int x = 30;            int y = 10;
    if( x = = 30 ){
      if( y = = 10 ){
      System.out.print("X = 30 and Y = 10");
    }
  }
}
```

This would produce following result: X = 30 and Y = 10

Loops:

Loops are used to execute a set of statements repeatedly until a particular condition is satisfied. In Java we have three types of basic loops:

- **while Loop**
- **do...while Loop**
- **for Loop**

As of java 5 the enhanced foreach loop was introduced. This is mainly used for Arrays.

How while Loop works?

In while loop, condition is evaluated first and if it returns true then the statements inside while loop execute. When condition returns false, the control comes out of loop and jumps to the next statement after while loop.

Note: The important point to note when using while loop is that we need to use increment or decrement statement inside while loop so that the loop variable gets changed on each iteration, and at some point condition returns false. This way we can end the execution of while loop otherwise the loop would execute indefinitely.

The syntax of a while loop is:

```
while(condition)
{
  //Statements
 //increment or decrement
}
```

Example:

```
public class Test {
  public static void main(String args[]){
    int x= 10;
    while( x < 15 ){
      System.out.println("value of x : " + x );
      x++;
    }
  }
}
```

This would produce following result:
value of x : 10
value of x : 11
value of x : 12
value of x : 13
value of x : 14
Example: write a program to print stmt for 10 times : "Java is very easy"

The do...while Loop:
do-while loop is similar to while loop, however there is a difference between them: In while loop, condition is evaluated before the execution of loop's body but in do-while loop condition is evaluated after the execution of loop's body.
Syntax:

```
do{
  //Statements
 //increment or decrement
}while(condition);
```

How do-while loop works?
First, the statements inside loop execute and then the condition gets evaluated, if the condition returns true then the control gets transferred to the "do" else it jumps to the next statement after do-while.
Example:

```
public class Test {
  public static void main(String args[]){
    int x= 10;
    do{
      System.out.println("value of x : " + x );
      x++;
     }while( x < =15 );
  }
}
```

This would produce following result:
value of x : 10
value of x : 11
value of x : 12
value of x : 13
value of x : 14
value of x : 15
Example: write a program to print stmt for 10 times : "selenium is very very easy"

The for Loop:
A for loop is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.A for loop is useful when you know how many times a task is to be repeated.

Syntax:

```
for(initialization; condition ; increment/decrement)
{
  statement(s);
}
```

For works as below:

First step: In for loop, initialization happens first and only one time, which means that the initialization part of for loop only executes once.

Second step: Condition in for loop is evaluated on each iteration, if the condition is true then the statements inside for loop body gets executed. Once the condition returns false, the statements in for loop does not execute and the control gets transferred to the next statement in the program after for loop.

Third step: After every execution of for loop's body, the increment/decrement part of for loop executes that updates the loop counter.

Fourth step: After third step, the control jumps to second step and condition is re-evaluated.

Example:

```
public class Test {
  public static void main(String args[]){
    for(int x = 10; x < 20; x ++){
      System.out.println("value of x : " + x );
    }
  }
```

Example: write a program to print numbers from 1 to 20 by increment of 2

Break and Continue Keywords in JAVA

The break Keyword:

The break statement is usually used in following two scenarios:

a) Use break statement to come out of the loop instantly. Whenever a break statement is encountered inside a loop, the control directly comes out of loop and the loop gets terminated for rest of the iterations. It is used along with if statement, whenever used inside loop so that the loop gets terminated for a particular condition.

b) It is also used in switch case control. Generally all cases in switch case are followed by a break statement so that whenever the program control jumps to a case, it doesn't execute subsequent cases. As soon as a break is encountered in switch-case block, the control comes out of the switch-case body.

Syntax: The syntax of a break is a single statement inside any loop: break;

Example:

```
public class Br {
public static void main(String args[]){
        for(int i=10; i<=15; i++)      {
                if(i==13)
                 break;
                System.out.println(i);
        }
```

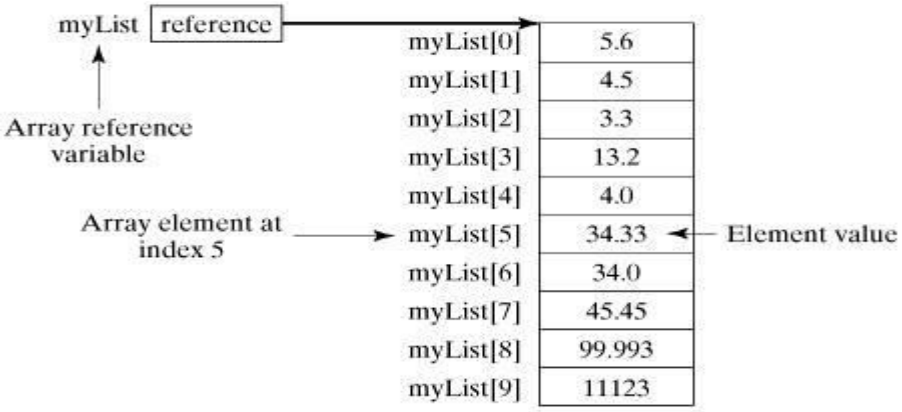
```
}
}
```
Outputshown as below
10
11
12

The continue Keyword:
Continue statement is mostly used inside loops. Whenever it is encountered inside a loop, control directly jumps to the beginning of the loop for next iteration, skipping the execution of statements inside loop's body for the current iteration. This is particularly useful when you want to continue the loop but do not want the rest of the statements (after continue statement) in loop body to execute for that particular iteration.
Syntax:
The syntax of continue is a single statement inside any loop: continue;
Example:

```
public class Cnt {
public static void main(String args[]){
        for(int i=10; i<=15; i++)      {
                if(i==13)
                continue;
                System.out.println(i);
        }
}
}
```
Outputshown as below
10
11
12
14
15

Arrays in Java
Arrays: - Group of similar elements
Java provides a data structure, the array, which stores a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.
Instead of declaring individual variables, such as number0, number1, ..., and number99, you declare one array variable such as numbers and use numbers[0], numbers[1], and ..., numbers[99] to represent individual variables.
Syntax : datatype array name = array values
Int myList[] = {5,6,4,5,3---11}
Example: Following statement declares an array variable, myList, creates an array of 10 elements of double type, and assigns its reference to myList.

double[] myList = new double[10];
Following picture represents array myList. Here myList holds ten double values and the indices are from 0 to 9.



Processing Arrays: When processing array elements, we often use either for loop or foreach loop because all of the elements in an array are of the same type and the size of the array is known.
Note: The array elements can be accessed with the help of the index,Accessing the array with an index greater than or equal to the size of the array leads to ArrayIndexOutofBound Exception.
The foreach Loops: For-each is another array traversing technique like for loop, while loop, do-while loop introduced in Java5. It starts with the keyword for like a normal for-loop.Instead of declaring and initializing a loop counter variable, you declare a variable that is the same type as the base type of the array, followed by a colon, which is then followed by the array name.In the loop body, you can use the loop variable you created rather than using an indexed array element.It's commonly used to iterate over an array or a Collections class (eg, ArrayList)

Syntax: for (type var : array) {
statements using var;
}

Sample Program for Arrays & For Each Loop

```
public class Foreach1 {
  public static void main(String[] args)  {
    int myList[] = {10, 20, 30, 40 ,50,60};
    // Print all the array elements
    for (int element: myList) {
      System.out.println(element);
    }
  }
}
```

Example : write a program to print all the elements from below array.

```
    // String data[] ={"selenium","training",”by”,"suresh”};
```

ArrayList in Java

ArrayList is a part of collection framework and is present in java.util package.ArrayList is initialized by a size; however the size can increase if collection grows or shrunk if objects are removed from the collection.

Example:

```
import java.util.ArrayList;
public class ArrayListExp {
public static void main(String args[]) {
        ArrayList<String> subjects = new ArrayList<String>();
        subjects.add("Mat");
        subjects.add("sci");
        subjects.add("eng");
        subjects.add("tel");
        System.out.println(subjects);
        subjects.add(2,"hin");
        System.out.println(subjects);
}
}
```

Example: Create an arrayList with Selenium components details and print the same.

Java User Input (Scanner)

The Scanner class is used to get user input, and it is found in the java.util package.

To use the Scanner class, create an object of the class and use any of the available methods found in the Scanner class

Program for taking the inputs from key board (or) accepting the values in run time (Example: Addition of 2 numbers)

```
Import java.util.Scanner;
 Class AddNumbers {
   Public static void main (String args [])   {
     Int x, y, z;
     System.out.println ("Enter two integers to calculate their sum ");
     Scanner in = new Scanner (System.in);
     x = in.nextInt (); //nextInt is a pre-defined function which is used to accept only integer values
     y = in.nextInt ();
     z = x + y;
     System.out.println ("Sum of entered integers = "+z);
   }
}
```

Outputshown as below:

Enter two integers to calculate their sum
10
20
Sum of entered integers = 30

Example : Write a java program to perform substract two numbers by accepting the values from keyboard

The switch Statement: Switch case statement is used when we have number of options (or choices) and we may need to perform a different task for each choice.

Syntax : switch (variable or an integer expression) {

```
    case constant:
    //Java code
    break;
    case constant:
    //Java code
    break;
    default:
    //Java code ;
}
```

Break statement is optional in switch case but you would use it almost every time you deal with switch case.

break keyword in default block :

The control would itself come out of the switch after default so I didn't use it, however if you still want to use the break after default then you can use it, there is no harm in doing that.

Few points about Switch Case

1) Case doesn't always need to have order 1, 2, 3 and so on. It can have any integer value after case keyword. Also, case doesn't need to be in an ascending order always, you can specify them in any order based on the requirement.

2) You can also use characters in switch case.

```
import java.util.Scanner;
public class SwitchExp {
public static void main(String args[]) {
        Scanner sc= new Scanner(System.in);
        System.out.println("1.Add");
        System.out.println("2.sub");
        System.out.println("3.mul");
        System.out.println("4.div");
        System.out.println("Enter first number");
        int a = sc.nextInt();
        System.out.println("Enter second number");
        int b = sc.nextInt();
        System.out.println("Enter your choice");
        int ch = sc.nextInt();
        switch(ch){
```

```
                case 1:
                        System.out.println(a+b);
                        break;
                case 2:
                        System.out.println(a-b);
                        break;
                case 3:
                        System.out.println(a*b);
                        break;
                case 4:
                        System.out.println(a/b);
                        break;
                default:
                        System.out.println("Invalid choice");
        }
}
}
```

Sample Program: Reverse of the String –

```
public class StringRev {
        public static void main(String[] args)
   {
      String text = "suresh";
      String reverse = "";
      for(int i=text.length()-1; i>=0; i--) {
         reverse = reverse + text.charAt(i);
      }
      System.out.println("Reversed string is:" + reverse );
      System.out.println("reverse completed");
   }
}
```

Sample program : multiplication table of given number

```
import java.util.Scanner;
public class Table {
         public static void main(String args[])        {
            int n, c;
           System.out.println("Enter an integer to print it's multiplication table");
            Scanner in = new Scanner(System.in);
            n = in.nextInt();
            System.out.println("Multiplication table of " + n + " is : ");
            for (c = 1; c <= 10; c++)
              System.out.println(n + "*" + c + " = " + (n*c));
          }
        }
```

What is OOP?

OOP stands for Object-Oriented Programming.

Procedural programming is about writing procedures or methods that perform operations on the data, while object-oriented programming is about creating objects that contain both data and methods.

Object-oriented programming has several advantages over procedural programming:

- OOP is faster and easier to execute
- OOP provides a clear structure for the programs
- OOP helps to keep the Java code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
- OOP makes it possible to create full reusable applications with less code and shorter development time

Inheritance

The process by which one class acquires the properties (data members) and functionalities (methods) of another class is called inheritance. The aim of inheritance is to provide the reusability of code so that a class has to write only the unique features and rest of the common properties and functionalities can be extended from the another class.

Child Class: The class that extends the features of another class is known as child class, sub class or derived class.

Parent Class: The class whose properties and functionalities are used(inherited) by another class is known as parent class, super class or Base class.

Note: The biggest advantage of Inheritance is that the code that is already present in base class need not be rewritten in the child class.This means that the data members(instance variables) and methods of the parent class can be used in the child class as.

Syntax: Inheritance in Java

To inherit a class we use extends keyword. Here class B is child class and class A is parent class. The class B is inheriting the properties and methods of A class.

```
class B extends A
{


}
```

The following kinds of inheritance are there in java.

Single level Inheritance

Multilevel Inheritance

multiple Inheritance(Java does not support)

Single Inheritance

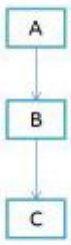
When a class extends another one class only then we call it a single inheritance. The below flow diagram shows that class B extends only one class which is A. Here A is a parent class of B and B would be a child class of A.

(a) Single Inheritance

Example:

```
class A {
        public void test ()    {
        System.out.println ("Hai...");
        System.out.println("parent class");
                }
        }
public class B extends A {
                public static void main (String [] args){
                        B s= new B ();
                        s.test ();
                }
        }
```

Multilevel Inheritance : Multilevel inheritance refers - where one can inherit from a derived class, thereby making this derived class the base class for the new class. As you can see in below flow diagram C is subclass or child class of B and B is a child class of A, Multilevel inheritance can go up to any number of levels.



(d) Multilevel Inheritance

```
class A {
int a=10;
int b=20;
public void selIDE(){
        System.out.println("IDE");
}

}
class B extends A{
        int x=30;
        int y=40;
        public void selWD(){
                System.out.println("WD");
                System.out.println(a+b); //accessing from class A
        }
```

```
}
public class C extends B{
        public void selRC(){
                System.out.println("RC");
                System.out.println(x+y); //accessing from class B
                System.out.println(a+b); //accessing from class A
        }
public static void main(String args[]){
        C obj = new C();
obj.selIDE(); //accessing methods of class A without creating object for class A
obj.selWD(); //accessing methods of class A without creating object for class B
obj.selRC();
}
}
```

Polymorphism in JAVA :

Polymorphism: It means one name with many forms.

These are 2 types:

Method Over loading

Method Over riding

Method Overloading

Writing two or more methods in the same class in such way that each method has same name but with different method signatures

Method Overriding

Writing two or more methods in super and sub classes such that the methods have same name and same signature

Method Overloading (Writing two or more methods in the same class in such way that each method has same name but with different method signatures)

As we know that a method has its own signature which is known by method's name and the parameter types. Java has a powerful feature which is known as method overloading. With the help of this feature we can define two methods of same name with different parameters. It allows the facility to define that how we perform the same action on different type of parameter.

Example:

```
public class OverLoad {
public void add(int a,int b){
        System.out.println(a+b);
}
public void add(int a,int b ,int c){
        System.out.println(a+b+c);
}
public static void main(String args[]){
        OverLoad obj = new OverLoad();
        obj.add(10, 20);
        obj.add(10, 20, 30);
}
}
```

Overriding (Writing two or more methods in super and sub classes such that the methods have same name and same signature)

Method overriding in java means a subclass method overriding a super class method. Super class method should be non-static. Subclass uses extends keyword to extend the super class. In overriding methods of both subclass and super class possess same signatures. Overriding is used in modifying the methods of the super class.

Example:

```
public class OverRide  {
  public void add(int a,int b){
          System.out.println(a+b);
  }
}
//Create new class as OverRide1
public class OverRide1 extends OverRide{
          public void add(int a,int b){
                    System.out.println(a-b);
           }
          public static void main(String args[]){
                    OverRide1 obj1 = new OverRide1();
                    obj1.add(10, 20);
                    OverRide obj = new OverRide();
                    obj.add(10, 20);
          }
}
```

Encapsulation & Abstraction in JAVA

Encapsulation – Data Bind

Encapsulation is a process of binding or wrapping the data and the codes that operates on the data into a single entity. Eg: class

Example:

```
class Person{
          //variable - data
          private String name = "Suresh";
          private int age = 26;
          //method
          public void talk()     {
                    System.out.println("Hello ,Iam"+name);
                    System.out.println("My age is"+age);
          }         public static void main(String args[]){
                    Person p = new Person();
                    p.talk();
          }
}
```

Abstraction: Data Hide

A class that is declared using "abstract" keyword is known as abstract class. It can have abstract methods (methods without body) as well as concrete methods (regular methods with body). A normal class (non-abstract class) cannot have abstract methods.

"Data abstraction is a mechanism of retrieving the essential details without dealing with background details".

•To use use abstraction we need use abstract keyword in the class

•For abstracts method implementation will be available in other class

•We can't create any object for abstract class.To get the access for abstract methods will use inherit those abstract classes.

Summary of Abstract classes and Abstract methods :

- An abstract class is a class that is declared with abstract keyword.
- An abstract method is a method that is declared without an implementation.
- An abstract class may or may not have all abstract methods. Some of them can be concrete methods
- A method defined abstract must always be redefined in the subclass,thus making overriding compulsory OR either make subclass itself abstract.
- Any class that contains one or more abstract methods must also be declared with abstract keyword.
- There can be no object of an abstract class.That is, an abstract class can not be directly instantiated with the new operator.
- An abstract class can have parametrized constructors and default constructor is always present in an abstract class.

Advantages of Abstraction

- It reduces the complexity of viewing the things.
- Avoids code duplication and increases reusability.
- Helps to increase security of an application or program as only important details are provided to the user.

Example

```
abstract class Bank {
        abstract void credit();
        abstract void debit();
}
class HDFC extends Bank{
        void credit() {
                System.out.println("Amount credit from HDFC");
        }
        void debit() {
                System.out.println("Amount debited from HDFC");
        }
}
class ICICI extends Bank{
        void credit() {
                System.out.println("Amount credit from ICICI");
```

```
	}
	void debit() {
		System.out.println("Amount debited from ICICI");
	}
}
public class TestBank{
	public static void main(String args[]) {
		HDFC h = new HDFC();
		h.credit();
		h.debit();
		ICICI i = new ICICI();
		i.credit();
		i.debit();
	}
}
```

Interface in JAVA

Interface : Interface looks like a class but it is not a class. An interface can have methods and variables just like the class but the methods declared in interface are by default abstract (only method signature, no body) the variables declared in an interface are public, static & final by default.

An interface in java is a blueprint of a class. It has static constants and abstract methods only. The interface in java is a mechanism to achieve fully abstraction. There can be only abstract methods in the java interface not method body. It is used to achieve fully abstraction and multiple inheritance in Java.

Interface definition begins with a keyword interface.

An interface like that of an abstract class cannot be instantiated.

What is the use of interface in Java?

Interface is used for full abstraction. Since methods in interfaces do not have body, they have to be implemented by the class before you can access them. The class that implements interface must implement all the methods of that interface. Also, java programming language does not allow you to extend more than one class, However you can implement more than one interfaces in your class.

Example :

```
interface WebDriver{
	public void openApplication();
	public void closeApplication();
}
class FirefoxDriver implements WebDriver{
	public void openApplication() {
		System.out.println("Firefox Open");
	}
	public void closeApplication() {
		System.out.println("Firefox Close");
	}
```

```
}
public class ChromeDriver implements WebDriver {
  public void openApplication() {
        System.out.println("Chrome Open");
  }
  public void closeApplication() {
        System.out.println("Chrome Close");
  }
  public static void main(String args[]) {
        //Creating object for ChromeDriver class directly
        ChromeDriver ch = new ChromeDriver();
        ch.openApplication();
        ch.closeApplication();
//Creating object for WebDriver(interface) indirectly - with the reference of
ChromeDriver class
        WebDriver driver = new ChromeDriver();
        driver.openApplication();
        driver.closeApplication();
       //Creating object for FirefoxDriver class directly
        FirefoxDriver ff = new FirefoxDriver();
        ff.openApplication();
        ff.closeApplication();
//Creating object for WebDriver(interface) indirectly - with the reference of
FirefoxDriver class
        WebDriver driver = new FirefoxDriver();
        driver.openApplication();
        driver.closeApplication();
  }
}
```

Exception Handling in JAVA

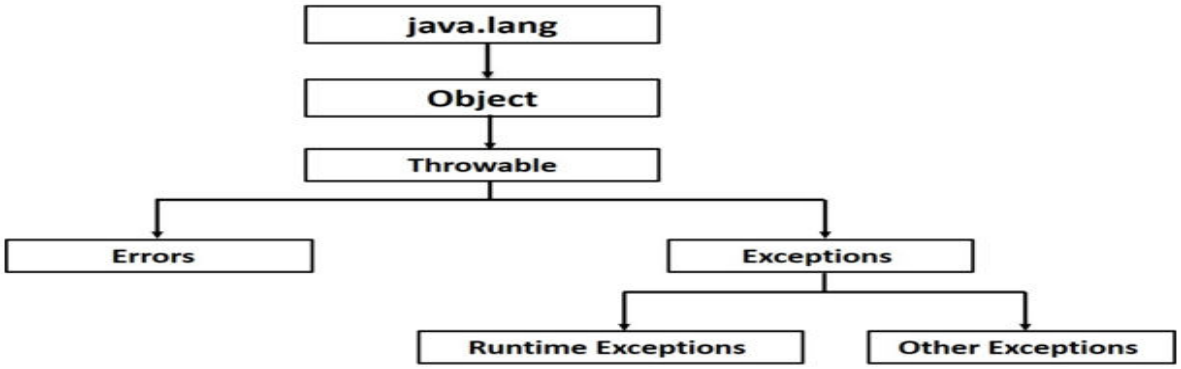
What is an exception?

An Exception is an unwanted event that interrupts the normal flow of the program. When an exception occurs program execution gets terminated. In such cases we get a system generated error message. The good thing about exceptions is that they can be handled in Java. By handling the exceptions we can provide a meaningful message to the user about the issue rather than a system generated message, which may not be understandable to a user.

Why an exception occurs?

There can be several reasons that can cause a program to throw exception. For example: Opening a non-existing file in your program, Network connection problem, bad input data provided by user etc.

Exception Hierarchy: All exception classes are subtypes of the java.lang.Exception class.



Example for Try & Catch Block

```
public class ExcepTest{
  public static void main(String args[])  {
    try     {
          int b=10/0;
          System.out.println(b);
          }
    catch(Exception e)     {
      System.out.println( "Exception thrown  :" + e);
    }
    System.out.println("Out of the block");
  }
}
```

Example for Try & Catch & Finally Block

```
public class Final{
  public static void main(String args[]){
           int a[] = {10,20,30,40};
    try     {
      System.out.println("Access element three :" + a[2]);
      System.out.println("Testing");
          }
    catch(Exception e)     {
      System.out.println("Exception thrown123  :" + e);
    }
  finally     {
      System.out.println("First element value: " +a[1]);
      System.out.println("The finally statement is executed");
    }
  }
}
```

Some More Details information on Exception Handling

An exception is a problem that arises during the execution of a program. An exception can occur for many different reasons, including the following:

- A user has entered invalid data.
- A file that needs to be opened cannot be found.
- A network connection has been lost in the middle of communications, or the JVM has run out of memory.

Some of these exceptions are caused by user error, others by programmer error, and others by physical resources that have failed in some manner.
To understand how exception handling works in Java, you need to understand the three categories of exceptions:
Checked exceptions: A checked exception is an exception that is typically a user error or a problem that cannot be foreseen by the programmer. For example, if a file is to be opened, but the file cannot be found, an exception occurs. These exceptions cannot simply be ignored at the time of compilation.
Runtime exceptions: A runtime exception is an exception that occurs that probably could have been avoided by the programmer. As opposed to checked exceptions, runtime exceptions are ignored at the time of compliation.
Errors: These are not exceptions at all, but problems that arise beyond the control of the user or the programmer. Errors are typically ignored in your code because you can rarely do anything about an error. For example, if a stack overflow occurs, an error will arise. They are also ignored at the time of compilation.
Exception Hierarchy:
All exception classes are subtypes of the java.lang.Exception class. The exception class is a subclass of the Throwable class. Other than the exception class there is another subclass called Error which is derived from the Throwable class.
Errors are not normally trapped form the Java programs. These conditions normally happen in case of severe failures, which are not handled by the java programs.
Errors are generated to indicate errors generated by the runtime environment.
Example: JVM is out of Memory. Normally programs cannot recover from errors.
The Exception class has two main subclasses: IOException class and RuntimeException Class.
Catching Exceptions: A method catches an exception using a combination of the try and catch keywords. A try/catch block is placed around the code that might generate an exception. Code within a try/catch block is referred to as protected code, and the syntax for using try/catch looks like the following:

```
try{
  //Protected code
}catch(ExceptionName e1){
  //Catch block
}
```

A catch statement involves declaring the type of exception you are trying to catch. If an exception occurs in protected code, the catch block (or blocks) that follow the try is checked. If the type of exception that occurred is listed in a catch block, the exception is passed to the catch block much as an argument is passed into a method parameter.
Example: The following is an array is declared with 2 elements. Then the code tries to access the 3rd element of the array which throws an exception.

```
// File Name : ExcepTest.java
import java.io.*;
public class ExcepTest{
  public static void main(String args[]){
    try{
```

```
      int a[] = new int[2];
      System.out.println("Access element three :" + a[3]);
    }catch(ArrayIndexOutOfBoundsException e){
      System.out.println("Exception thrown  :" + e);
    }
    System.out.println("Out of the block");
  }
}
```

This would produce following result:

```
Exception thrown  :java.lang.ArrayIndexOutOfBoundsException: 3
Out of the block
```

Multiple catch Blocks:A try block can be followed by multiple catch blocks. The syntax for multiple catch blocks looks like the following:

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

The previous statements demonstrate three catch blocks, but you can have any number of them after a single try. If an exception occurs in the protected code, the exception is thrown to the first catch block in the list. If the data type of the exception thrown matches ExceptionType1, it gets caught there. If not, the exception passes down to the second catch statement. This continues until the exception either is caught or falls through all catches, in which case the current method stops execution and the exception is thrown down to the previous method on the call stack.

Example: Here is code segment showing how to use multiple try/catch statements.

```
class ex{
        public static void main(String[] args)      {
                try                {
                        //open the files
                        System.out.println("open files");
                        //do some processing
                        int n = 0;
                        //System.out.println("n= "+ n);
                        int a = 45/n;
                        System.out.println("a= "+ a);
                        int b[] = {10,20,30};
                        b[50] = 100;
                }
        catch (ArithmeticException ae)              {
                //display the exception details
```

```
                System.out.println(ae);
                //display any message to the user
        System.out.println("Please pass data while running this program");
                }
        catch(ArrayIndexOutOfBoundsException aie)          {
                //diaplay exception details
                aie.printStackTrace();
                //display a message to user
System.out.println("please see that the array index is within the range");
                }

        } }
```

The throws/throw Keywords: If a method does not handle a checked exception, the method must declare it using the throws keyword. The throws keyword appears at the end of a method's signature.

You can throw an exception, either a newly instantiated one or an exception that you just caught, by using the throw keyword. Try to understand the different in throws and throw keywords.

The following method declares that it throws a Remote Exception:

```
import java.io.*;
public class className{
  public void deposit(double amount) throws RemoteException  {
    // Method implementation
    throw new RemoteException();
  }
  //Remainder of class definition
}
```

A method can declare that it throws more than one exception, in which case the exceptions are declared in a list separated by commas. For example, the following method declares that it throws a Remote Exception and an InsufficientFundsException:

```
Import java.io.*;
public class className{
  public void withdraw(double amount) throws RemoteException,
                    InsufficientFundsException  {
     // Method implementation
  }
  //Remainder of class definition
}
```

The finally Keyword The finally keyword is used to create a block of code that follows a try block. A finally block of code always executes, whether or not an exception has occurred.

Using a finally block allows you to run any cleanup-type statements that you want to execute, no matter what happens in the protected code.

A finally block appears at the end of the catch blocks and has the following syntax:

```
try{
```

```
  //Protected code
}catch(ExceptionType1 e1) {
  //Catch block
}catch(ExceptionType2 e2) {
  //Catch block
}catch(ExceptionType3 e3) {
  //Catch block
}finally {
  //The finally block always executes.
}
Example:  public class Final{
  public static void main(String args[]){
    int a[] = new int[2];
    try{
      System.out.println("Access element three :" + a[3]);
    }catch(ArrayIndexOutOfBoundsException e){
      System.out.println("Exception thrown  :" + e);
    }
    finally{
      a[0] = 6;
      System.out.println("First element value: " +a[0]);
      System.out.println("The finally statement is executed");
    }
  }
}
```

This would produce following result:

```
Exception thrown  :java.lang.ArrayIndexOutOfBoundsException: 3
First element value: 6
The finally statement is executed
```

Note the followings:

- A catch clause cannot exist without a try statement.
- It is not compulsory to have finally clauses whenever a try/catch block is present.
- The try block cannot be present without either catch clause or finally clause.
- Any code cannot be present in between the try, catch, finally blocks.

Common Scenarios of Java Exceptions

1) A scenario where ArithmeticException occurs

If we divide any number by zero, there occurs an ArithmeticException.

```
int a=50/0;//ArithmeticException
```

2) A scenario where NullPointerException occurs

If we have a null value in any variable, performing any operation on the variable throws a NullPointerException.

```
String s=null;
System.out.println(s.length());//NullPointerException
```

3) A scenario where NumberFormatException occurs

The wrong formatting of any value may occur NumberFormatException. Suppose I have a string variable that has characters, converting this variable into digit will occur NumberFormatException.

```
String s="abc";
int i=Integer.parseInt(s);//NumberFormatException
```

4) A scenario where ArrayIndexOutOfBoundsException occurs

If you are inserting any value in the wrong index, it would result in ArrayIndexOutOfBoundsException as shown below:

```
int a[]=new int[5];
a[10]=50; //ArrayIndexOutOfBoundsException
```

JAVA Assignment

Write a sample java program to Print the statement as – " *** I have not failed. I've just found 10,000 ways that won't work *** "
Create One Class, three Methods and print below stmt. Method1 - Failure is the key to success; each mistake teaches us something. Method2 - Coming together is a beginning; keeping together is progress; working together is success. Method3 - In a day, when you don't come across any problems - you can be sure that you are travelling in a wrong path
Create One Class, three static Methods and print below stmt. Method1 – Daily I will practice selenium for 2 hours. Method2 – Daily I will sleep only for 6 hours. Method3 – Daily I will wake up at 6 clock
Write a Java program to find given number is - odd or even
Write a program to swap the two numbers
Write a program to print reverse of the number
Print below format using for loop * ** *** **** *****
Write a program to print the each element from the below array using for each loop int arr[]={12,13,14,44}; String s1[]={"Suresh","selenium","project","training"}
Write a program to add list of element to array list and print the same- Selenium ,Training ,By ,Suresh.

Write a program to Choose Day of the Week using switch stmt
Write any example for single level and multilevel inheritance
Write any example for Method Overload and Method Overriding
Write any example for abstraction
Write any example for encapsulation
Write any example for interface
Write any example for Exception handling

Selenium 4 - WebDriver

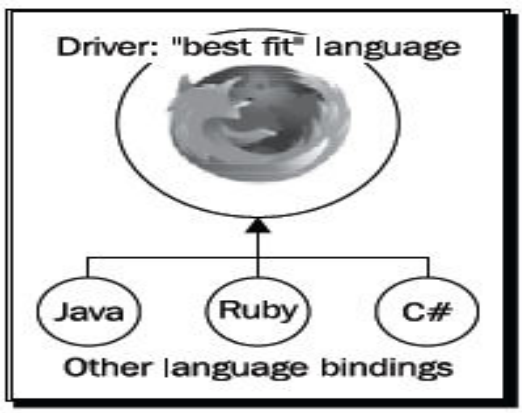
About Selenium WebDriver

Selenium WebDriver tool is used to automate web application testing to verify that it works as expected. It supports many browsers such as Firefox, Chrome, IE, and Safari. However, using the Selenium WebDriver, we can automate testing for web applications only. It does not qualify for window-based applications. It also supports different programming languages such as C#, Java, Perl, PHP and Ruby for writing test scripts. Selenium Webdriver is platform-independent since the same code can be used on different Operating Systems like Microsoft Windows, Apple OS and Linux. It is one of the components of the selenium family, which also includes Selenium IDE, Selenium Client API, Selenium Remote Control and Selenium Grid.

Selenium WebDriver does not handle window component, but this limitation can be overcome by using external tools such as AUTO IT tool, Sikuli etc. It has different location strategies as well such as ID, Name, Link text, Partial link text, Class name, CSS selector and Xpath. It also has better support dynamic web pages like Ajax, where elements of the web page may change without the page itself being reloaded. By using different jar files, we can also test API, Database Test etc. using Selenium WebDriver.

WebDriver – Introduction & Architecture

Web Driver is designed to providing a simpler, more concise programming interface along with addressing some limitations in the Selenium-RC API.
Developed to better support dynamic web pages where elements of a page may change without the page itself being reloaded
Makes direct calls to the browser using each browser's native support for automation.



WebDriver is one Interface and its had below implementing classes
Implementing Classes
ChromeDriver,
EdgeDriver,
EventFiringWebDriver,
FirefoxDriver,
HtmlUnitDriver,
InternetExplorerDriver,
MarionetteDriver,
OperaDriver, RemoteWebDriver, SafariDriver

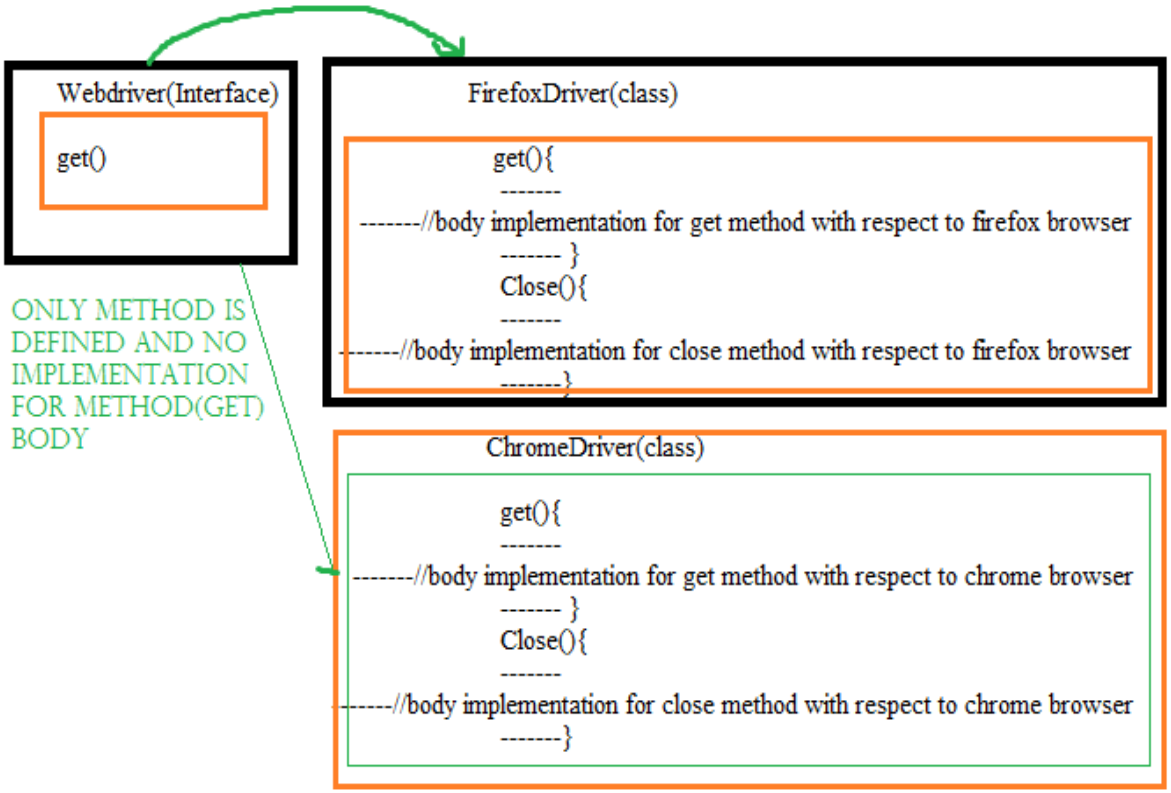
Diffirence between SeleniumRC and WebDriver

Selenium RC	**WebDriver**
Workson almost all browsers.Does not work on latest version of Firefox/IE	Works on latest versions of almost all browsers - Firefox, IE(6,7,8), Opera, Chrome
No Record and run	No Record and run
Server is required to start	No server required to start
Core engine is JavaScript based	Interacts natively with browser application
Its a simple and small API	Complex and a bit large API as compared to RC
Less Object oriented API	Purely Object oriented API
Cannot move mouse with it	Can move mouse cursor
Full xpaths have to be appended with 'xapth=\\' syntax	No need to append 'xpath=\\'
No Listeners	Implementation of Listeners is provided
Selenium RC Cannot be used to Test iPhone or Android Apps	WebDriver can be used to Test iPhone or Android Apps

General Example for SeleniumRC

WebDriver

Design of WebDriver interface:

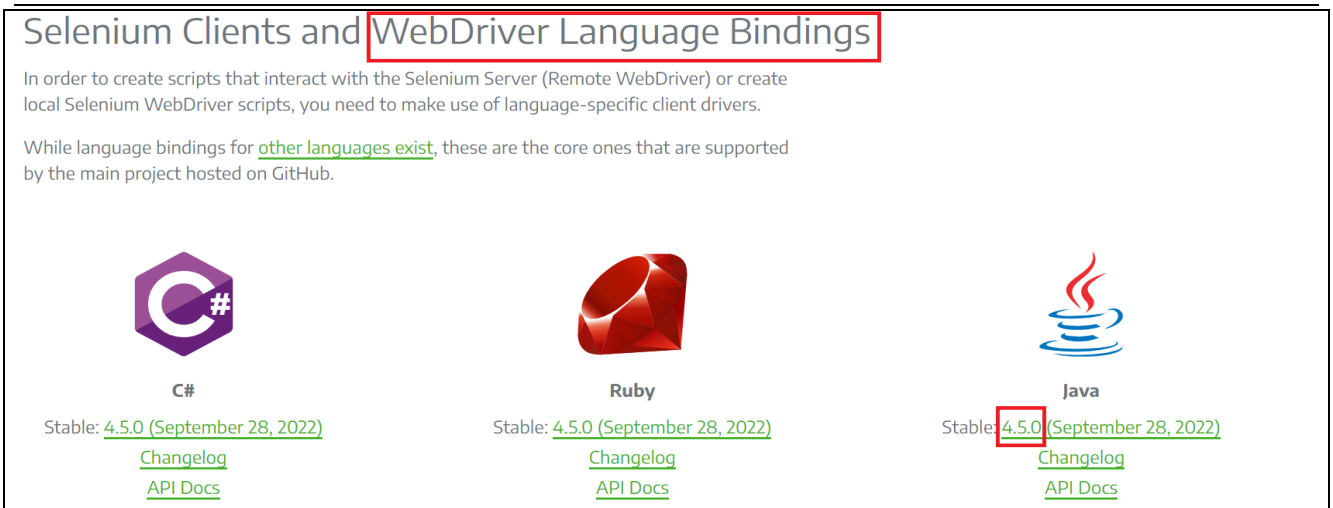


WebDriver driver = new FirefoxDriver();
driver.get();
FirefoxDriver driver1 = new FirefoxDriver();
driver1.get();
From the above diagram driver object is related to webdriver interface and driver1 object related to FirefoxDriver class. But both objects can run method bodies in FirefoxDriver class, because ChromeDriver is Concrete class for WebDriver interface.
Because of this webdriver interface facility in feature if any new things need to get added that will be introduced as one more new Driver class.

Selenium WebDriver installation

Download the latest version of Selenium WebDriver and Extract

1. Open **https://www.selenium.dev/downloads/**
2. Find the 'Selenium Client & WebDriver Language Bindings' section on the page
3. In the 'Selenium Client & WebDriver Language Bindings' section, click on the 'Download' link that is related to Java Programming language (As we are writing our tests in Java language) as shown below:



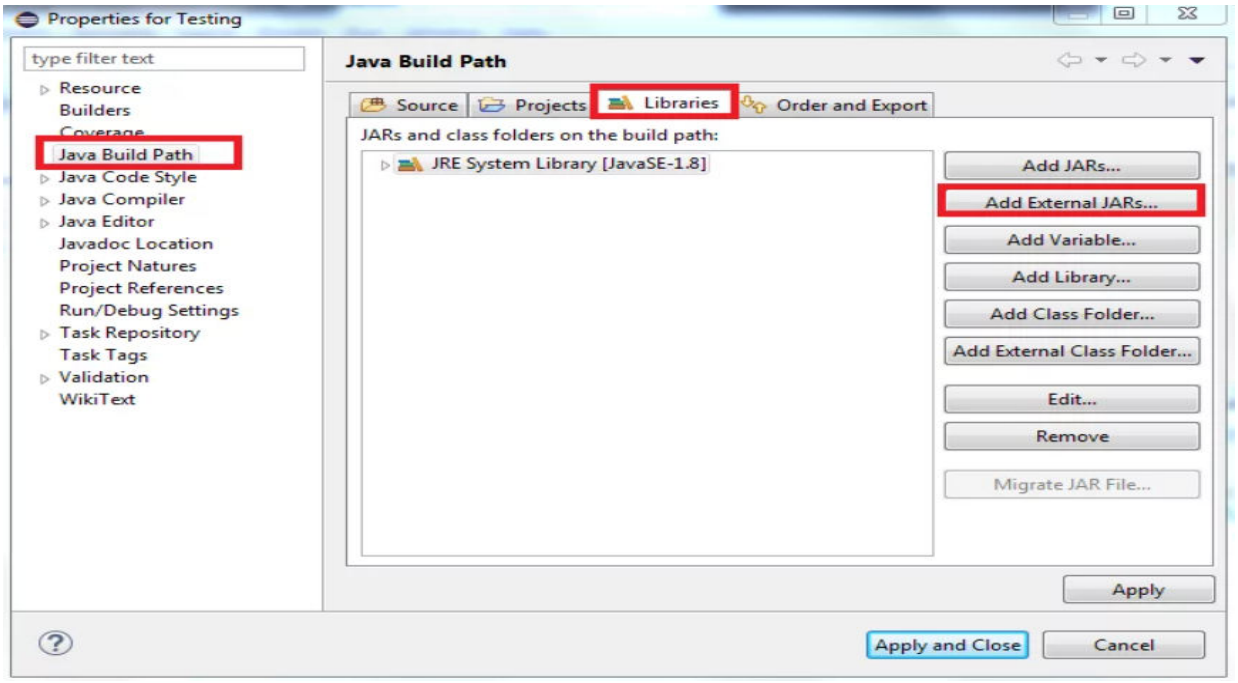
4. Ensure that **selenium-java-X.XX.X.zip** file got downloaded as shown below: (chances of version will be changing from date to date)
5. Extract the downloaded **selenium-java-X.XX.X.zip** file as shown below: (perform right click on folder and click on Extract All option)
6. Ensure that the zip file got extracted
7. Open the extracted folder and ensure that all the jar files shown got extracted:

Note : Before writing webdriver program all jar files[inside of lib folder jarfiles & outside of lib folder jarfiles] need to added for java project.

Note: Before writing webdriver program makes sure that you added webdriver jar files to project.

Steps to add Jar files to project

- Right click on created project
- Select Build path option and click on Configure build path option
- Click on libraries Tab

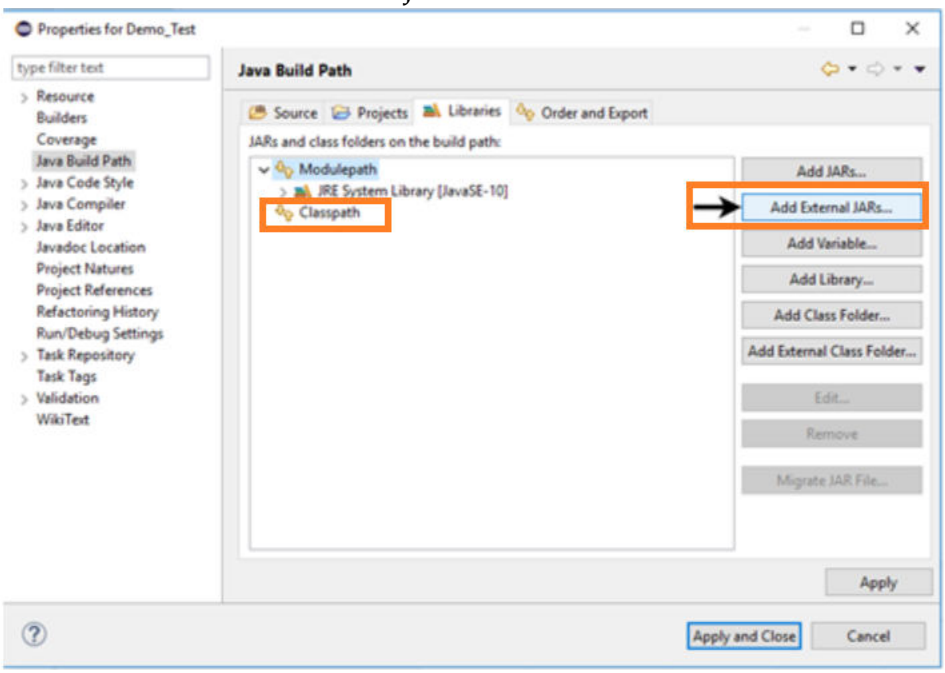


- Click on Add external jar file button
- Locate the directory where you have downloaded the Selenium jar files, select the respective jars and click on "Open" button.
- Repeat the same steps for the jars which are present under the "libs" folder.
- Open "libs" folder, select all of the respective jar files and click on "Open" button.
- Once you get all the Selenium jar files in your Libraries tab, click on Apply and Close button.

[or]

Note : in case of Java is latest version in your system follow below steps to add jar files.

- Right click on created project
- Select Build path option and click on Configure build path option
- Click on libraries Tab
- Click on **classpath** option under libraries tab
- Click on Addextennal Jars button



- Locate the directory where you have downloaded the Selenium jar files, select the respective jars and click on "Open" button.
- Repeat the same steps for the jars which are present under the "libs" folder.
- Open "libs" folder, select all of the respective jar files and click on "Open" button.
- Once you get all the Selenium jar files in your Libraries tab, click on Apply and Close button.

By this we had successfully configured Selenium WebDriver with Eclipse IDE. Now, we are ready to write our test scripts in Eclipse and run it .

HRMS – Project Test Cases :

Company Logo	Project History					
	Project ID					
	Project Name					
	Test Case History					
	Created By		Date Created			
	Reviewed By		Date Reviewed			
	Approved By		Date Last Updated			
	Test Information					
	Test case Name		Test case Objective			
	Test Executed By		Date Executed			
	Version		Build			

TC ID	TestSteps	Test Design	Input Data	Expected Result	Pass/Fail	Comments
TC_101_VerifyLogin	1	Open Application	application url			
	2	add any wait stmt	3000ms			
	3	Verify HRMS title				
	4	Enter Username	admin			
	5	Enter Password	admin			
	6	Click on login button				
	7	Verify Title				
	8	Verify Welcome text				
	9	Click on Logout				
	10	close application				

TC-Example for FirefoxDriver(firefox browser)

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.By;
public class TC001_Login_Logout {
public static void main(String args[]) throws Exception{
//To Launch Browser
WebDriver driver = new FirefoxDriver();
                    //TestCase steps
//To Enter URL
driver.navigate.to("https://sureshitacademy.in/login.php");
//To Enter Data in Text Box
driver.findElement(By.name("txtUserName")).sendKeys("sureshit");
driver.findElement(By.xpath("//input[@name='txtPassword']")).sendKeys("sureshit");
//To perform click on Action on button
```

```
driver.findElement(By.name("Submit")).click();
//This statement is used in Selenium Automation to pause the execution for the specified time.
Thread.sleep(3000);
        System.out.println("Login completed");
//To perform click on Action on Link
driver.findElement(By.linkText("Logout")).click();
        System.out.println("Logout completed");
//To Close Browser
driver.close();
}
}
```

TC-Example for ChromeDriver(chrome browser)

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class TC_101 {
        public static void main(String args[])throws Exception{
                WebDriver driver=new ChromeDriver();
                driver.navigate.to("https://sureshitacademy.in/login.php");
                driver.findElement(By.name("txtUserName")).sendKeys("sureshit");
                driver.findElement(By.name("txtPassword")).sendKeys("sureshit");
                driver.findElement(By.name("Submit")).click();
                Thread.sleep(2000);
                driver.findElement (By.linkText ("Logout")).click();
                driver.close();
        }
}
```

TC-Example for EdgeDriver(edge browser)

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.edge.EdgeDriver;
public class TC_101 {
        public static void main(String args[])throws Exception{
                WebDriver driver=new EdgeDriver();
                driver.navigate.to("https://sureshitacademy.in/login.php");
                driver.findElement(By.name("txtUserName")).sendKeys("sureshit");
                driver.findElement(By.name("txtPassword")).sendKeys("sureshit");
                driver.findElement(By.name("Submit")).click();
                Thread.sleep(2000);
                driver.findElement (By.linkText ("Logout")).click();
                driver.close();
        }
}
```

Example to Verify the application title and calling test data by using variables

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.By;
public class TC_Verify {
        public static url = “https://sureshitacademy.in/login.php”;
        public static String un = "sureshit";
        public static String pw = "sureshit";
        public static String title1 = “OrangeHRM - New Level of HR Management”;

public static void main(String args[]) throws Exception{
WebDriver driver = new FirefoxDriver();
        //Test Case steps
        driver.navigate().to(url);
//getTitle()—Pre-defined method in selenium to retrieve Current Page Title
//equals()--- its Java Pre-Defined method to compare expected result and actual
result
        if(driver.getTitle().equals(title1)) {
                System.out.println("Title matched");
        }
        else    {
System.out.println("Title not matched and expected title is "+driver.getTitle());
        }
        driver.findElement(By.xpath("//input[@name='txtUserName']")).sendKeys(un);
        driver.findElement(By.xpath("//input[@name='txtPassword']")).sendKeys(pw);
        driver.findElement(By.name("Submit")).click();
        Thread.sleep(3000);
        System.out.println("Login completed");
        driver.findElement(By.linkText("Logout")).click();
        System.out.println("logout completed");
        driver.close();
   }
}
```

Mouse Over Actions in WebDriver

What is Mouse Hover Action?

A mouse hover is also called as hover. Mouse hover action is basically an action where a user places a mouse over a designated area like a hyperlink. It can cause some event to get triggered.

For Example, moving the mouse over an element on web page displays some pop-up windows or maybe description boxes.

To perform mouseover, first we identify the element to be hovered in the web page and then we perform the action of movetoelement using Actions class provided in

WebDriver
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.

Syntax :

```
import org.openqa.selenium.interactions.Actions;
WebElement element = driver.findElement(By.linkText("PIM"));
Actions action = new Actions(driver);
action.moveToElement(element).build().perform();
```

The 'build()' method is used to compile all the list of actions into a single step and ready to be performed.

// Example to perform mouseover on PIM main menu and click on AddEmployee link

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.interactions.Actions;
public class mainmenu_MouseOver{
public static void main(String args[])     {
WebDriver driver =new FirefoxDriver();
driver.navigate().to("https://sureshitacademy.in/login.php");
driver.manage().window().maximize();
Thread.sleep(3000L);
driver.findElement(By.name("txtUserName")).sendKeys("suresh");
driver.findElement(By.name("txtPassword")).sendKeys("suresh123");
driver.findElement(By.name("Submit")).click();
Thread.sleep(3000);
//mouseover on pim mainmenu
/* We have created an instance of Actions class i.e. “action” to access methods of
Actions class and used WebElement to locating “PIM - menu” link on the homepage
with the help of “linkText” locator and storing  it in a WebElement “element”.  */
// moveToElement(toElement) --- Moves the mouse to the middle of the element.
Actions action = new Actions(driver);
// We need use the perform() method when executing the Action object
action.moveToElement(driver.findElement(By.linkText("PIM"))).perform();
Thread.sleep(3000);
//clicking on addemployee submenu link
driver.findElement(By.linkText("Add Employee")).click();
Thread.sleep(3000);
System.out.println("Clicked on submenu");
driver.close();
 }
}
```

For practice Test case steps:

TC ID	TestSteps	Test Design	Input Data	Expected Result	Pass/Fail	Comments
TC_101_Verification and mouseover	1	Open Application	url=urlname			
	2	add any wait stmt	3000ms			
	3	Verify HRMS title	title =title name			
	4	Enter Username by calling variable	un=admin			
	5	Enter Password by calling varible	pw=admin			
	6	Click on login button				
	7	Verify Title				
	8	Navigate to admin main menu				
	9	Navigate to DataImport/Export submenu				
	10	click on export option				
	11	add any wait stmt				
	12	click on logout				
	13	close application				

What is Iframe?

A web page which is embedded in another web page or an HTML document embedded inside another HTML document is known as a frame.

The IFrame is often used to insert content from another source, such as an advertisement, into a Web page. The <iframe> tag specifies an inline frame.

How to identify the frame ?

Perform inspect Element and Search with the 'frame'/ 'iframe', if you can find any tag name with the 'frame'/ 'iframe' then it is meaning to say the page consisting an frame/ iframe.

Methods to handle frames

To Enter – driver.switchTo().frame("framename/frameid/index")

To Exit - driver.switchTo().defaultContent()

switchTo().frame(index)

- This method allows users to switch to a particular frame using the frame index.The frame index is a zero-based value which means the first frame of the web page has the index 0, the second frame has the index 1, and the third frame has the index 3 and so on.
- This method throws NoSuchFrameException when the required frame is not found on the current web page.

switchTo().frame(name)

- This method allows users to switch to a particular frame using the developer-defined name of the frame.
- Frame name needs to be enclosed within double quotes for it to be considered as a String parameter.
- This method throws NoSuchFrameException when the required frame is not found on the current web page.

switchTo().frame(id)

- This method allows users to switch to a particular frame using the developer-defined id of the frame.
- Frame name needs to be enclosed within double quotes for it to be considered as a String parameter.
- This method throws NoSuchFrameException when the required frame is not found on the current web page.

```
//Example for Frames
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import static org.testng.Assert.assertTrue;
public class AddEmp {
   public static void main(String[] args) throws InterruptedException {
WebDriver driver=new FirefoxDriver();
      driver.navigate().to("https://sureshitacademy.in/login.php");
      driver.findElement(By.xpath("//input[@type='text']")).sendKeys("suresh");
 driver.findElement(By.xpath("//input[@type='password']")).sendKeys("suresh123");
      driver.findElement(By.xpath("//input[@type='Submit']")).click();
      Thread.sleep(5000L);
      //Entering into frame
      driver.switchTo().frame("rightMenu"); //Enter into Frame
      //Clicking on Add Button
driver.findElement(By.xpath("//*[@id='standardView']/div[3]/div[1]/input[1]")).click();
     Thread.sleep(2000L);
     driver.findElement(By.xpath("//*[@id='txtEmpLastName']")).sendKeys("suresh");
      Thread.sleep(2000L);
driver.findElement(By.xpath("//*[@name='txtEmpFirstName']")).sendKeys("selenium");
      driver.findElement(By.xpath("//*[@id='btnEdit']")).click();
      Thread.sleep(2000L);
      System.out.println("New Employee Added");
      driver.switchTo().defaultContent(); //Exit from Frame
      driver.findElement(By.xpath("//*[@id='option-menu']/li[3]/a")).click();
      driver.close();
 }
}
```

Alerts in WebDriver

What is Alert?

Alert is a small message box which displays on-screen notification to give the user some kind of information or ask for permission to perform certain kind of operation. It may be also used for warning purpose.

In General There are two types of alerts:

- Windows-based alert pop-ups
- Web-based alert pop-ups

Selenium can't handle window based alerts.To handle those kind of scenarios need to use some third-party tools like Sikuli or Auto-it.

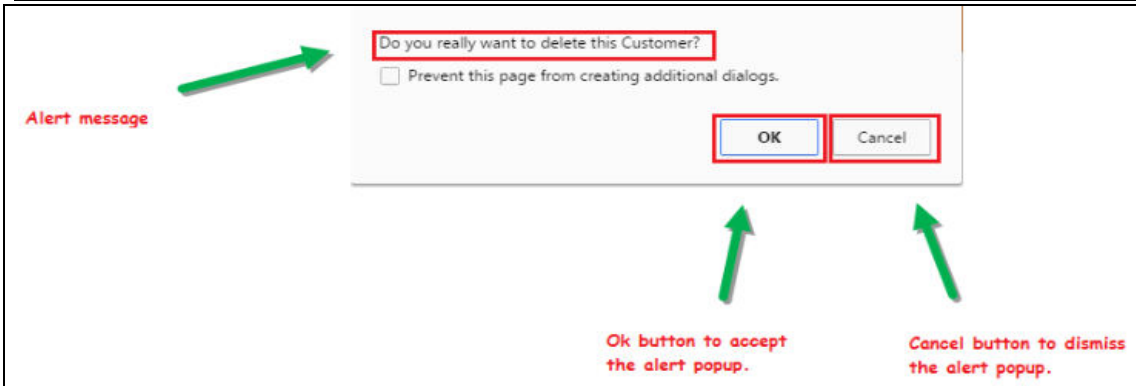
In this example we will understand handling webbased alerts using selenium.

Here are few alert scenarios

1)Simple Alert --This simple alert displays some information or warning on the screen.

2) Prompt Alert--This Prompt Alert asks some input from the user and selenium webdriver can enter the text using sendkeys(" text to enter").

3) Confirmation Alert--This confirmation alert asks permission to do some type of operation.



Syntax :

```
import org.openqa.selenium.Alert;
Alert alert = driver.switchTo().alert();
```

Alert is an interface. below are the methods that are used to handle the alerts

To Click on OK button. - alert.accept();

To click on Cancel button - alert.dismiss()

To get the text which is present on the Alert. - alert.getText();

To pass the text to the prompt popup - alert.sendkeys();

Example for alerts

```
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
public class PopUp {
        public static void main(String args[])throws Exception{
WebDriver driver =new FirefoxDriver ();
                driver.get ("https://sureshitacademy.in/login.php");
                 System.out.println (driver.getTitle ());
                 driver.findElement (By.name ("txtUserName")).sendKeys ("suresh");
                 driver.findElement (By.name ("Submit")).click ();
                 Thread.sleep (2000L);
                  Alert a= driver.switchTo ().alert ();
//To Retrive data from Alert
System.out.println (a.getText ());
//To click on Ok button on Alert
a.accept ();
            driver.findElement (By.name ("txtPassword")).sendKeys ("suresh123");
                 driver.findElement (By.name ("Submit")).click ();
                 Thread.sleep (2000);
                 System.out.println ("Login completed");
                 driver.findElement (By.linkText ("Logout")).click ();
                 driver.close ();
        }
}
```

TC ID	TestSteps	Test Design	Input Data	Expected Result	Pass/Fail	Comments
TC_101_Frames&Alerts	1	Open Application	url=urlname			
	2	add any wait stmt	3000ms			
	3	Verify title using if stmt	title =title name			
	4	Verify username textbox label using assert stmt				
	5	Enter Username by calling variable	un=admin			
	6	Click on login button				
	7	print text from popup				
	8	click on OK btn in popup				
	9	click on clear btn				
	10	Type username and password by calling variable				
	11	Click on login button				
	12	add any wait stmt				
	13	Verify title using if stmt				
	14	navigate to PIM mainmenu				
	15	Click on AddEmployee submenu				
	16	Type Employee Firstname and Lastname	firstname = selenium lastname = Suresh			
	17	print emp code				
	18	click on save button				
	19	click on logout				
	20	close application				

Keyboard Actions – Robot Class

Robot Class in Selenium:

In certain Selenium Automation Tests, there is a need to control keyboard or mouse to interact with OS windows like Download pop-up, Alerts, Print Pop-ups, etc. or native Operation System applications like Notepad, Skype, Calculator, etc.

Selenium Webdriver cannot handle these OS pop-ups/applications.In genereal while working with selenium to handle these kind of scenarios need to take the help of third party tools like Sikuli or Auto-it.

Integrating these tools will be an extrs activity for the tester,So reather going with third party tools we can go-ahead using Robot Class.

Advantages of Robot Class

- Robot Class can simulate Keyboard and Mouse Event
- Robot class can suppour webbased application and Windows based application
- Robot Class can help in upload/download of files when using selenium web driver
- No need of to identify any object to perform repective action
- Robot Class can easily be integrated with automation framework

Java.awt.Robot class is used to take the control of mouse and keyboard. Once you get

the control, you can do any type of operation related to mouse and keyboard through your java code. This class is used generally for test automation.

KeyPress(): This method is called when you want to press any key

KeyRelease(): This method is used to release the pressed key on the keyboard

Eg:

```
import java.awt.Robot;
import java.awt.event.KeyEvent;
        Robot r = new Robot();
To press TAB key from keyboard
        r.keyPress(KeyEvent.VK_TAB);
        r.keyRelease(KeyEvent.VK_TAB);
To Press ENTER Key from Keyboard
      r.keyPress(KeyEvent.VK_ENTER);
      r.keyRelease(KeyEvent.VK_ENTER);
```

//Example to perform Keyboard activities using Robot Class

```
import java.awt.Robot;
import java.awt.event.KeyEvent;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
public class TC_Robotclass {
      public static void main(String args[]) throws Exception{
            //Test case steps
      WebDriver driver = new FirefoxDriver();
      driver.get("https://sureshitacademy.in/login.php");
            System.out.println("Application Opened");
            driver.findElement(By.name("txtUserName")).sendKeys("sureshit");
driver.findElement(By.xpath("//input[@name='txtPassword']")).sendKeys("sureshit");
            //Perform TAB & Enter using KeyBoard actions
            Robot r = new Robot();
            r.keyPress(KeyEvent.VK_TAB);
            r.keyRelease(KeyEvent.VK_TAB);

            r.keyPress(KeyEvent.VK_ENTER);
            r.keyRelease(KeyEvent.VK_ENTER);
            Thread.sleep(3000L);
            System.out.println("Login completed");
            driver.findElement(By.linkText("Logout")).click();
            System.out.println("Logout completed");
            driver.close();
      }
}
```

//Example to perform keyboard activities using WebDriver and Auto Complete Feature

Keys: keys is a pre-defined class available in Selenium ,which is used to perform keyboard action. Keys will support only webbased scenarios not able to handle windows related scenaios.

```
import org.openqa.selenium.By;
import org.openqa.selenium.Keys;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
public class AutoComplete{
        public static void main(String args[]) throws Exception        {
WebDriver driver = new FirefoxDriver();
        driver.navigate().to("https://www.google.co.in");
Thread.sleep(3000);
      driver.findElement(By.name("q")).sendKeys("selenium suresh");
      Thread.sleep(5000);
      driver.findElement(By.name("q")).sendKeys(Keys.ARROW_DOWN);
      Thread.sleep(3000);
      System.out.println("Firsttime down arrow pressed");
                driver.findElement(By.name("q")).sendKeys(Keys.ARROW_DOWN);
                Thread.sleep(3000);
                System.out.println("Second time down arrow pressed");
                driver.findElement(By.name("q")).sendKeys(Keys.ENTER);
                Thread.sleep(3000);
                System.out.println("clicked on Enter btn");
                driver.close();
        }
}
```

Dropdown , Navigate and checkbox Methods

DropDown & Multiple Select Operations works together and almost the same way. To perform any action, the first task is to identify the element group.as DropDown /Multiple Select is not a single element. They always have a single name or id but and they contains one or more than one elements in them.

Select Class in Selenium

WebDriver's support classes called "Select", which provides useful methods for interacting with select options. User can perform operations on a select dropdown and also de-select operation using the below methods.

```
import org.openqa.selenium.support.ui.Select;
Select lstbox = new Select(driver.findElement(By.id("dropdownid")));
Method Name: selectByVisibleText
 lstbox.selectByVisibleText("Text");
Method Name: selectByIndex
 lstbox.selectByIndex(index);
Method Name: selectByValue
 lstbox.selectByValue(value);
Method Name: deselectByIndex
```

lstbox.deselectByIndex(index);

Navigate Methods

To move back a single "item" in the web browser's history. And it will not perform any action if you are on the first page viewed.

navigate().back()

To move a single "item" forward in the web browser's history. And it will not perform any action if we are on the latest page viewed.

navigate().forword()

This methods Load a new web page in the current browser window. This is done using an HTTP GET operation, and the method will block until the load is complete.

URL – It should be a fully qualified URL.

navigate().to(url)

It refreshes the current web page

navigate().refresh()

Checkbox

check box on/off is also done using the **click()** method

//Example to select single value & Multipul values from dropdown

```
package webDriverExamples;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.Select;

public class TC_DropDown {
public static void main(String args[]) throws Exception{
        WebDriver driver = new FirefoxDriver();
        driver.navigate().to("https://sureshitacademy.in/login.php");
        System.out.println(driver.getTitle());
        driver.findElement(By.name("txtUserName")).sendKeys("sureshit");
        driver.findElement(By.name("txtPassword")).sendKeys("sureshit");
        driver.findElement(By.name("Submit")).click();
        Thread.sleep(4000);
        System.out.println("Login completed");
        //Enter frame
        driver.switchTo().frame("rightMenu");
//Select the value from search by dropdown
  Select st = new Select(driver.findElement(By.name("loc_code")));
  st.selectByVisibleText("Emp. ID");
  Thread.sleep(4000);
  driver.findElement(By.name("loc_name")).sendKeys("0071");
  Thread.sleep(4000);
  driver.findElement(By.xpath("//input[@value='Search']")).click();
        Thread.sleep(4000);
//Clicking on checkbox
        driver.findElement(By.name("chkLocID[]")).click();
```

```
        Thread.sleep(4000);
        driver.findElement(By.xpath("//input[@value='Delete']")).click();
        Thread.sleep(4000);
        driver.switchTo().defaultContent();
        driver.findElement(By.linkText("Logout")).click();
        System.out.println("Logout competed");
        driver.close();
}
}
```

TC ID	TestSteps	Test Design	Input Data	Expected Result	Pass/Fail	Comments
TC_101_RobotClass & Dropdown	1	Open Application	url=urlname			
	2	add any wait stmt	3000ms			
	3	Verify title using if stmt	title =title name			
	4	Verify password textbox label using assert stmt				
	5	Enter password by calling variable	un=admin			
	6	Click on login button				
	7	print text from popup				
	8	click on OK btn in popup				
	9	click on clear btn				
	10	Type username	admin			
	11	Type password by using robot class	admin			
	12	Click on login button using robot class				
	13	add any wait stmt				
	14	navigate to admin main menu				
	15	navigate to DataImport / Export Submenu				
	16	click on Export option				
	17	select any value from dropdown				
	18	click on Export option				
	19	select Save radio btn from download popup				use robot class
	20	click on Ok btn from download popup				use robot class

21.click on logout
22.close application

WaitMethods (or) Synchronization

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

Why do users need Selenium Wait?

Most web applications are developed with Ajax and Javascript. When a page loads on a

browser, the various web elements that someone wants to interact with may load at various time intervals.
This obviously creates difficulty in identifying any element. On top of that, if an element is not located then the "ElementNotVisibleException" appears. Selenium Wait commands help resolve this issue.
Selenium WebDriver provides wait in tests.

- Implicit Wait
- Explicit Wait

Thread.sleep()
In this we just specify timeout value only. We will make the tool to wait until certain amount of time and then proceed further.
Implicit Wait: WebDriver waits for an element if they are not immediately available. So, WebDriver does not throw NoSuchElementException immediately.
Implicit Wait directs the Selenium WebDriver to wait for a certain measure of time before throwing an exception. Once this time is set, WebDriver will wait for the element before the exception occurs.
To add implicit waits in test scripts, import the following package.
import java.util.concurrent.TimeUnit;
Syntax : driver.manage.TimeOuts.implicitwait(10,Duration.ofSeconds(3));
Here we wait for 10 seconds, after that it gives NoSuchElementException. If the element present in 5 second then it should not wait for another 10 seconds.
Explicit Wait: Using Explicit Wait, we can tell WebDriver to wait for a certain condition to occur before proceeding further in the execution. We can use some of the prebuilt ExpectedConditions to wait for elements to become clickable, visible, invisible, etc.
Setting Explicit Wait is important in cases where there are certain elements that naturally take more time to load. If one sets an implicit wait command, then the browser will wait for the same time frame before loading every web element. This causes an unnecessary delay in executing the test script.
Explicit wait is more intelligent, but can only be applied for specified elements. However, it is an improvement on implicit wait since it allows the program to pause for dynamically loaded Ajax elements.
In order to declare explicit wait, one has to use "**ExpectedConditions**".
To use Explicit Wait in test scripts, import the following packages into the script.
import org.openqa.selenium.support.ui.ExpectedConditions
import org.openqa.selenium.support.ui.WebDriverWait
Then**, Initialize A Wait Object using WebDriverWait Class.**
WebDriverWait wait = new WebDriverWait(driver,30);
Here, the reference variable is named <wait> for the <WebDriverWait> class. It is instantiated using the WebDriver instance. The maximum wait time must be set for the execution to layoff. Note that the wait time is measured in seconds.
Syntax : WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(3));
wait.until(ExpectedConditions.elementToBeClickable(By.name("name")));
Difference between Implicit and Explicit Wait

Implicit Wait	Explicit Wait
Applies to all elements in a test script.	Applies only to specific elements as intended

	by the user.
No need to specify "ExpectedConditions" on the element to be located	Must always specify "ExpectedConditions" on the element to be located
Most effective when used in a test case in which the elements are located with the time frame specified in implicit wait	Most effective when used when the elements are taking a long time to load. Also useful for verifying property of the element such as visibilityOfElementLocated, elementToBeClickable,elementToBeSelected

//Example for wait method in webdriver

```
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
public class Explicitwait{
   public static void main(String[] args) throws InterruptedException {
WebDriver driver=new FirefoxDriver();
      driver.navigate().to("https://sureshitacademy.in/login.php");
      driver.findElement(By.xpath("//input[@type='text']")).sendKeys("suresh");
driver.findElement(By.xpath("//input[@type='password']")).sendKeys("suresh123");
   //Explicit Wait for element to be clickable
      WebDriverWait wait = new WebDriverWait(driver, 15);
wait.until(ExpectedConditions.elementToBeClickable(By.xpath("//input[@type='Submit']"
)));
      driver.findElement(By.xpath("//input[@type='Submit']")).click();
      //Implicit wait
      driver.manage().timeouts().implicitlyWait(3, TimeUnit.SECONDS);
      System.out.println("Login completed");
      driver.findElement(By.xpath("//*[@id='option-menu']/li[3]/a")).click();
      driver.close();
   } }
```

File upload using WebDriver

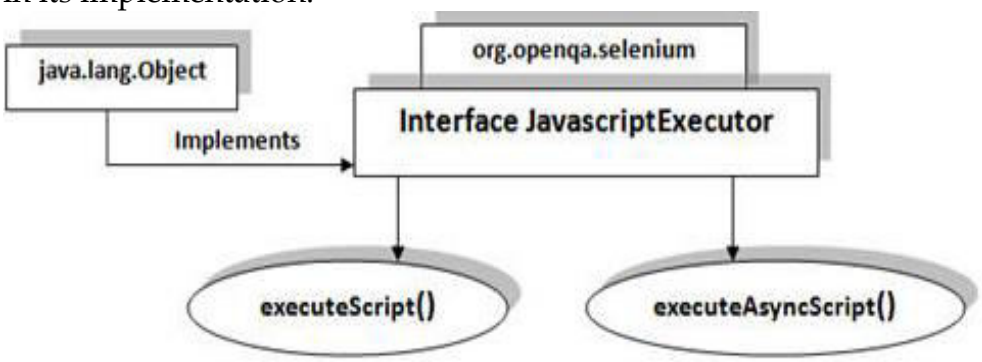
```
//Example to upload an image for AddNewEmployee Test case
WebElement fileInput =
driver.findElement(By.xpath("//input[@type='file'][@name='photofile']"));
fileInput.sendKeys("C:\\Users\\Public\\Pictures\\Sample Pictures\\Desert.jpg");
   Thread.sleep(5000);
   System.out.println("File uploaded successfully");
```

TC ID	TestSteps	Test Design	Input Data	Expected Result	Pass/Fail	Comments
TC_101_Wait&FileUpload	1	Open Application	url=urlname			
	2	add any wait stmt	3000ms			
	3	Verify title using if stmt	title =title name			
	4	Type username	un=admin			
	5	Type password	pw=admin			
	6	add explicit wait for Login btn				
	7	Click on login button				
	8	add implicit wait stmt				
	9	click on Add button				
	10	add implicit wait stmt				
	11	print Emp id				
	12	Type Employee Firstname & lastName	firstname=selenium lastname=suresh			
	13	Upload employee photo				
	14	click on save btn				
	15	click on logout				
	16	close application				

JavaScript Executer

What are JavaScript Executors?

JavascriptExecutor interface is a part of org.openqa.selenium and implements java.lang.Object class. JavascriptExecutor presents the capabilities to execute JavaScript directly within the web-browser. To be able to execute the JavaScript, certain mechanisms in the form of methods along with a specific set of parameters are provided in its implementation.



JavaScript Executors

While automating a test scenario, there are certain actions those become an inherent part of test scripts.

These actions may be:

•Clicking a button, hyperlink etc.

•Typing in a text box

•Scrolling Vertically or Horizontally until the desired object is brought into view

•And many more

But what if the selenium commands don't work?

Yes, it is absolutely possible that the very basic and elementary Selenium Commands

don't work in certain situations.
That said, to be able to troubleshoot such situation, we shoulder JavaScript executors into the picture.
Methods
ExecuteScript (String script, args)
As the method name suggests, it executes the JavaScript within the current window, alert, frame etc (the window that the WebDriver instance is currently focusing on)
ExecuteAsyncScript (String script, args)
As the method name suggests, it executes the JavaScript within the current window, alert, frame etc (the window that the WebDriver instance is currently focusing on)
The parameters and import statement are common to both the executor methods.
Parameters
Script – the script to be executed
Argument – the parameters that the script requires for its execution (if any)
Import statement
To be able to use JavascriptExecutors in our test scripts, we need to import the package using the following syntax:

```
import org.openqa.selenium.JavascriptExecutor;
```

Sample Code
#1) clicking a web element

```
/ Locating the web element using id
WebElement element = driver.findElement (By.id ("id of the webelement"));
// Instantiating JavascriptExecutor
JavascriptExecutor js = (JavascriptExecutor) driver;
// clicking the web element
js.executeScript ("arguments [0].click ();", element);
```

#2) Typing in a Text Box

```
//Instantiating JavascriptExecutor
JavascriptExecutor js = (JavascriptExecutor) driver;
// Typing the test data into Textbox
js.executeScript("document.getElementById('id of the element').value='test data';");
```

#3) Scrolling down until the web element is in the view

```
WebElement element=driver.findElement(By.xpath("//input[contains(@value,'Save')]"));
// Instantiating the javascriptExecutor and scrolling into the view in the single test step
((JavascriptExecutor)driver).executeScript("arguments[0].scrollIntoView(true);",element);
```

Sample Program

```
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.By;
public class JavaScriptExe {
public static void main(String args[]) throws Exception{
WebDriver driver = new FirefoxDriver();
//test casse steps
   driver.get("https://sureshitacademy.in/login.php");
```

```
   System.out.println(driver.getTitle());

driver.findElement(By.xpath("//input[@name='txtUserName']")).sendKeys("sureshit");
  driver.findElement(By.name("txtPassword")).sendKeys("sureshit");
 //using javascriptExecuter to click on login btn
   WebElement element = driver.findElement(By.name("Submit"));
   JavascriptExecutor executor = (JavascriptExecutor)driver;
   executor.executeScript("arguments[0].click();", element);
   Thread.sleep(3000);
   System.out.println("Login completed");
 // using javascriptExecuter to click on logout
   WebElement logout = driver.findElement(By.linkText("Logout"));
   JavascriptExecutor executor1 = (JavascriptExecutor)driver;
   executor1.executeScript("arguments[0].click();", logout);
   System.out.println("Logout completed");
   driver.close();
  }
}
```

Windows Handelers :

Get Window Handles. The Get Window Handles command of the WebDriver API returns a list of all WebWindow s. Each tab or window, depending on whether you are using a tabbed browser, is associated by a window handle that is used as a reference when switching to the window

Note : for the html file refer htmlfiles folder in onedrive.

```
import java.util.ArrayList;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
public class WindowHandels {
public static void main(String args[]) throws Exception{
        WebDriver driver = new FirefoxDriver();
        driver.get("file:///D:/Suresh_Selenium/HtmlFiles/multiplewindows.html");
        driver.findElement(By.id("btn1")).click();
        Thread.sleep(3000);
        driver.findElement(By.id("btn2")).click();
        ArrayList<String> wind=new
ArrayList<String>(driver.getWindowHandles());
        driver.switchTo().window(wind.get(0));
        Thread.sleep(3000);
        driver.quit();      }
}
```

Example for WebTable /HTML Table:

S#	Course Name	Instructor Name	Start Date
1	C++	James	1/2/2009
2	Pascal	John	2/2/2009
3	Cobol	Raja	3/3/2009
4	Selenium	Kangs	4/4/2009
5	Perl	Keith	5/5/2009
6	Python	Michell	6/6/2009

Now in above html table need to retrieve the row and coloumn count after that retrieve the data from particular cell and whole webtable.

Note : for the html file refer htmlfiles folder in onedrive.

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
public class Table {
        public static void main(String[] args)throws Exception          {
WebDriver driver=new FirefoxDriver ();
        driver.get("url");
//Count Details
int row =driver.findElements(By.xpath("//table[@id='idCourse']/tbody/tr")).size();
int col
=driver.findElements(By.xpath("//table[@id='idCourse']/tbody/tr[1]/td")).size();
int rowcol
=driver.findElements(By.xpath("//table[@id='idCourse']/tbody/tr/td")).size();
        System.out.println(row);
        System.out.println(col);
        System.out.println(rowcol);
//To get Data from particular Cell
String data1 = driver.findElement(By.xpath("//table [@id='idCourse']/tbody/tr
[2]/td[2]")).getText();
        System.out.println (data1);
//To get Data from all rows
        for (int i=1;i<=row;i++)      {
                //for (int j=1;j<=col;j++)              {
String data = driver.findElement(By.xpath("//table [@id='idCourse']/tbody/tr
["+i+"]")).getText ();
            System.out.println(data);
                }
        }
        driver.close();
        }
}
```

Examples for Excel Activities

Example for Excel Sheets using JXL.jar file

Reading the single row from the excel sheet

Note: To work with JXL we need to add JXL.jar file to the project

ExcelFileName : 12345.xls

EmpID	EmpName	EmpSal
101	aaa	10000
102	bbb	20000
103	ccc	15000

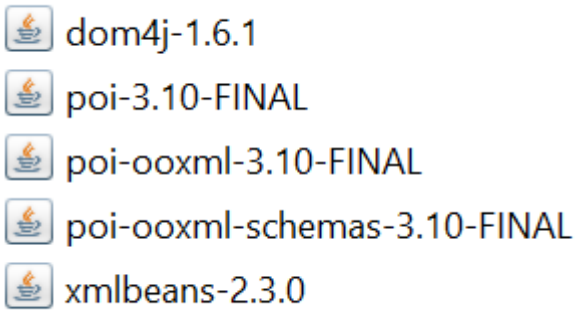
```
import java.io.FileInputStream;
import jxl.*;
public class Excel  {
public static void main(String args[])throws Exception{
        FileInputStream f1=new FileInputStream ("E:\\Selenium\\ReadExcel.xls");
        Workbook w1=Workbook.getWorkbook(f1);
        Sheet s1=w1.getSheet("Sheet1");
        System.out.println(s1.getName());
        int i=2; //reading data from one particular row
        String EmpID=s1.getCell(0, i).getContents ();
        String EmpName=s1.getCell(1, i).getContents ();
        String EmpSal=s1.getCell(2, i).getContents ();
        System.out.println(EmpID);
        System.out.println(EmpName);
        System.out.println(EmpSal);      }
}
```

Reading all the rows in Excel sheet

```
import java.io.FileInputStream;
import jxl.*;
public class Excelloop {
public static void main(String args[]) throws Exception{
        FileInputStream f1=new FileInputStream("E:\\Selenium\\ReadExcel.xls");
        Workbook objwb=Workbook.getWorkbook (f1);
        Sheet s1=objwb.getSheet(0);
        int rows = s1.getRows();     //to get row count
        System.out.println(rows);
        for (int i=1;i<rows;i++)      {
                String EmpID=s1.getCell(0, i).getContents ();
                String EmpName=s1.getCell(1, i).getContents ();
                String EmpSal=s1.getCell(2, i).getContents ();
                System.out.println(EmpID);
                System.out.println(EmpName);
                System.out.println(EmpSal);                }
    }
}
```

Example for Excel Sheets using POI.jar file

Reading the data from the excel sheet and writing the data in particular cell
Note: To Work with POI we need to add below POI.jar files to the project.



Excel File name: 123.xlsx

EmpNo	EmpName
101	Hari
102	Naveen/Test123
103	suresh

2ndRow,1stcol

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
import org.apache.poi.xssf.usermodel.XSSFCell;
import org.apache.poi.xssf.usermodel.XSSFRow;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
public class WriteExcelData {
public static void main(String args[]) throws Exception{
FileInputStream fis = new
FileInputStream("D:\\Suresh_Selenium\\WriteData.xlsx");
   XSSFWorkbook workbook = new XSSFWorkbook(fis);
   XSSFSheet sheet = workbook.getSheet("test");
   System.out.println(sheet.getSheetName());
   System.out.println(sheet.getLastRowNum());
   System.out.println("Before updating Cell Data "+sheet.getRow(2).getCell(1));
   // Write the data to excel file
   XSSFCell cell =  sheet.getRow(2).getCell(1);
   cell.setCellValue("Test123");
   fis.close();
FileOutputStream fileOut=new
FileOutputStream("D:\\Suresh_Selenium\\WriteData.xlsx");
   workbook.write(fileOut);
   System.out.println("Updated data after write is done " + cell.getStringCellValue());
   fileOut.close();
}
}
```

Example for - Taking screensot on failure in webDriver

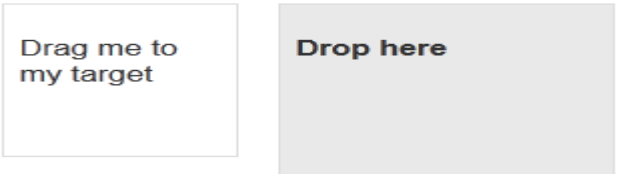
Note: Before writing the this program download and Add **commons.io.jar** file to project for the purpose of getting access for FileUtils class

```
import java.io.File;
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.By;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.interactions.Actions;

public class ScreenShot {
public static void main(String args[]) throws Exception {
        WebDriver driver = new FirefoxDriver();
        try             {
                driver.navigate().to("https://sureshitacademy.in/login.php");
                driver.findElement(By.name("txtUserName")).sendKeys("sureshit");
                driver.findElement(By.name("txtPassword")).sendKeys("sureshit");
                driver.findElement(By.name("Submit")).click();
                 Thread.sleep(3000);
                WebElement element = driver.findElement(By.linkText("PIM"));
                Actions action = new Actions(driver);
                action.moveToElement(element).perform();
                Thread.sleep(3000L);
        driver.findElement(By.linkText("Add Employee123")).click();
                Thread.sleep(4000);
                System.out.println("Clicked on submenu");
        driver.findElement(By.linkText("Logout")).click();
        }
        catch(Exception e)          {
                File f1 = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
          FileUtils.copyFile(f1, new File("g:\\TestResults.png"));
         }
        driver.quit();
}
}
```

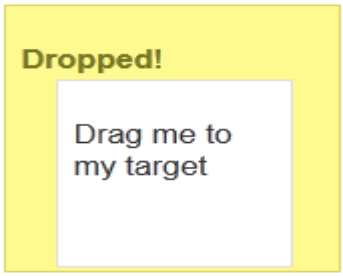
Example for - Drag and Drop in WebDriver

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.interactions.Actions;
// Note: Add testing.jar file for accessing assert statement
import static org.testng.Assert.assertEquals;
public class TC_DragandDrop {
public static void main(String args[]) throws Exception{
WebDriver driver = new FirefoxDriver();
driver.navigate().to("https://jqueryui.com/droppable/");
Thread.sleep(4000);
//Verify the title using assert statement
/*Verifying title using assert statement. The common functionality of assert statement is ,in case of condition is true then it will continue the execution , in case of condition is failed then it will stop the execution*/
assertEquals("Droppable | jQuery UI",driver.getTitle());
System.out.println("Title Matched");
//Enter into Frame
driver.switchTo().frame(0);
Actions ac = new Actions(driver);
ac.dragAndDrop(driver.findElement(By.id("draggable")),
                    driver.findElement(By.id("droppable"))).perform();
Thread.sleep(5000);
System.out.println("DragandDropCompleted");
//Exit from frame
driver.switchTo().defaultContent();
driver.close();
}
}
```

Before Executing the program



After Executing the program



1. Diffirence between get() & navigate.to() method
2. Diffirence between close() & quit() method
3. Write a syntax to type the data in Textbox
4. Write a syntax to perform click activity in Webdriver
5. Write a syntax to verify the Title in webdriver
6. Write a code to execute the programs in Chrome/IE browsers
7. Write a code to Verify Text in webdriver using Assert stmt
8. Explain Frames in WebDriver
9. Write a code to retrieve the data from TextBox
10. Write a code to Handle Alerts in WebDriver
11. Explain Wait methods in Webdriver
12. How to Perfrom Click action with out using Click method
13. Diffirence between close() and quit() methods
14. What are the Technical challenges you faced while working with Selenium Automation
15. List the common errors you faced while working with Selenium
16. Write a code to perform MoverOver
17. Write a code to Select single value & Multipul values from dropdown
18. Write a code to read the data from Excel
19. Write a code to get Row count & Coloum count from WebTable & Write a code TO retrive data from particular cell
20. Write a code to Click on Yes/OK btn in Alert.

Automation Frameworks

What an Automation Framework is?

A test automation framework is a set of assumptions, concepts and tools that provide support for automated software testing. The main advantage of such a framework is the low cost for maintenance. If there is change to any test case then only the test case file needs to be updated and the Driver Script and Startup script will remain the same. Ideally, there is no need to update the scripts in case of changes to the application.

Utility of Test Automation Framework

Provides an Outline of overall Test Structure
Ensures Consistency of Testing
Minimizes the Amount of Code for Development - thereby Less Maintenance
Maximizes Reusability
Reduces Exposure of Non-Technical Testers to Code
Enables Test Automation using Data

How Many Types of Automation Frameworks are there?

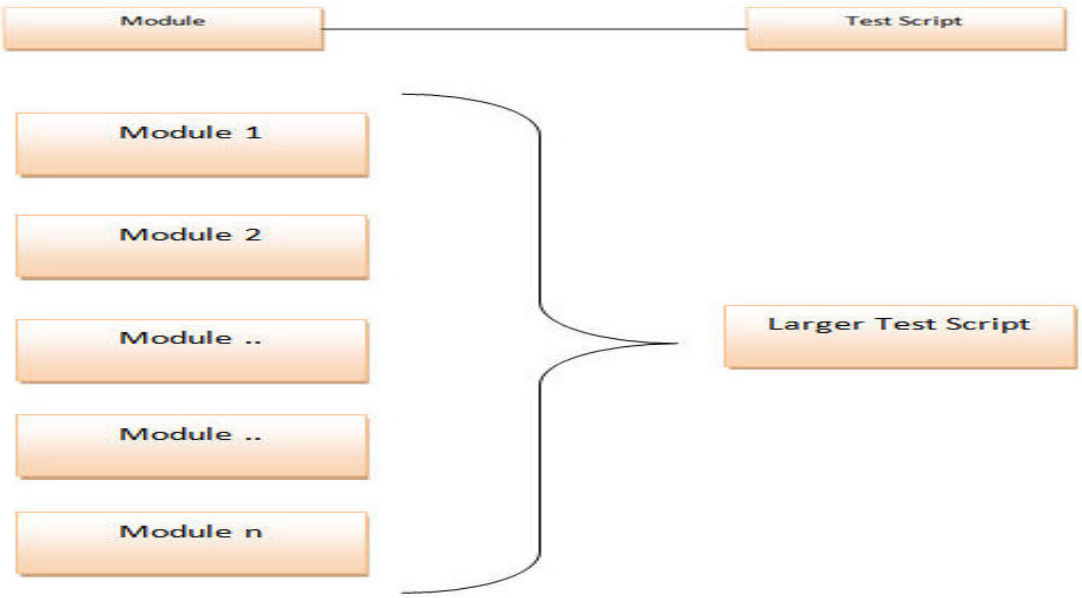
Generally there are 4 Types:

Data Driven Automation Framework
Keyword Driven Automation Framework
Modular Automation Framework
Hybrid Automation Framework

Framework

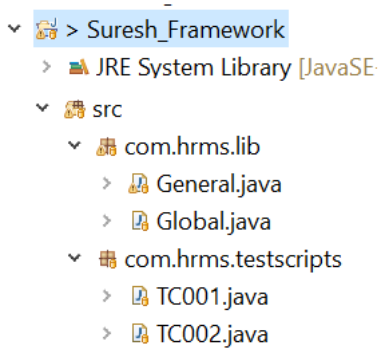
Framework is the approach where all the test cases are first analyzed to find out the

reusable flows. Then while scripting, all these reusable flows are created as functions and stored in external files and called in the test scripts wherever required.
Enables creation of Small, Independent Scripts representing
Modules & Functions of the Application under Test (AUT)
Test Library Architecture Framework:
Enables creation of Library Files representing Modules & Functions of the Application under Test (AUT)
Objects defined by parameterized code (i.e., regular expressions)
Custom functions used to enhance workflow capabilities



Advantages of Modular Framework
•Test Scripts can be created in relatively less time as the reusable functions need to be created only once.
• Effort required to create test cases is also lesser due to code reuse.
•If there are any changes in the reusable functions, the changes need to be done in only a single place. Hence script maintenance is easier.
Disadvantages of Modular Framework
•Since data is still hardcoded in the script, the same test case cannot be run for multiple data values without changing data after each run.
•Additional time is spent in analyzing the test cases to identify with reusable flows.
•Good programming knowledge is required to create and maintain function libraries.
Packages- Steps to Design the Modular FrameWork from the scratch.
A package is a group of classes.If we create on package in eclipse it will be considered as one folder in your eclipse workspace.

Design the Framework in eclipse as below.



Write the below code in each and every file as mentioned.

Project	Suresh_Framework	
Package	com.hrms.LIB	
	General.java	Maintains all the reusable fundtions related to your project. Eg: openBrowser() closeBrowser() login() logout() addemp() delemp() etc….
	Global.java	Maintains all the varibles & objects related your project Eg: WebDriver driver, application url, UserName, Password, etc … ==========Objects========== txt_UserName = "txtUserName" btn_Login = "Submit" link_logout = "Logout" etc…
Package	com.hrms.testscripts	All the actual test cases need to written in this package only
	TC_HRMS_101	
	TC_HRMS_102	
	TC_HRMS_103	

=====create com.hrms.LIB package and in that create **Global.java** file=========

```
package com.hrms.lib;
import org.openqa.selenium.WebDriver;
public class Global  {
//====================Varibles info=======================
```

```
public WebDriver driver;
public String url = "https://sureshitacademy.in/login.php";
public String un = " sureshit ";
public String pw = " sureshit ";
//======================Objects====================
public String txt_loginname = "txtUserName";
public String txt_password = "txtPassword";
public String btn_login = "Submit";
public String link_logout = "Logout";
}
```

==========create **General.java** file in same package=========

```
package com.hrms.lib;
import org.openqa.selenium.By;
import org.openqa.selenium.firefox.FirefoxDriver;
public class general extends Global  {
public void openapplication(){
driver = new FirefoxDriver();
        driver.navigate().to(url); }
public void closebrowser(){
        driver.quit();
}
public void login() throws Exception{
        driver.findElement(By.name(txt_loginname)).sendKeys(un);
        driver.findElement(By.name(txt_password)).sendKeys(pw);
        driver.findElement(By.name(btn_login)).click();
        Thread.sleep(3000);
}
public void logout(){
        driver.findElement(By.linkText(link_logout)).click(); }
public void addemp(){
        System.out.println("adding new emp"); }
public void delmp(){
        System.out.println("delete emp"); }
}
```

=======create all automation **test scripts** under the package of – com.hrms.testscripts=======

```
package com.hrms.testscripts;
import com.hrms.lib.*;
public class TC_hrms_101 {
public static void main(String args[]) throws Exception{
        general g = new general();
    //test case steps
        g.openapplication();
        g.login();
        g.addemp();
        g.delmp();
```

```
        g.logout();
        g.closebrowser();
}
}
```

Automate Below Test cases based on FrameWork:

	Project History					
	Project ID					
Company Logo	Project Name					
	Test Case History					
	Created By		Date Created			
	Reviewed By		Date Reviewed			
	Approved By		Date Last Updated			
	Test Information					
	Test case Name		Test case Objective			
	Test Executed By		Date Executed			
	Version		Build			

TC ID	TestSteps	Test Design	Input Data	Expected Result	Pass/Fail	Comments
TC_101_VerifyLogin	1	Open Application	application url			
	2	add any wait stmt	3000ms			
	3	Verify HRMS title				
	4	Enter Username	admin			
	5	Enter Password	admin			
	6	Click on login button				
	7	Verify Title				
	8	Verify Welcome text				
	9	Click on Logout				
	10	close application				

TC ID	TestSteps	Test Design	Input Data	Expected Result	Pass/Fail	Comments

TC_102_Add_NewEmp	1	Open Application	application url			
	2	add any wait stmt				
	3	Verify title				
	4	Enter Username	admin			
	5	Enter Password	admin			
	6	Click on login button				
	7	Verify Title				
	8	Verify Welcome text				
	9	navigate to pim mainmenu				
	10	click on addemployee submenu				
	11	enter employee firstname & last name	firstname=selenium lastname=suresh			
	12	click on save button				
	13	add wait stmt				
	14	click on logout				
	15	close application				

TC ID	TestSteps	Test Design	Input Data	Expected Result	Pass/Fail	Comments
TC_103_DeleteEmp						
	1	Open Application	application url			
	2	add any wait stmt				
	3	Verify title				
	4	Enter Username	admin			
	5	Enter Password	admin			
	6	Click on login button				
	7	navigate to pim mainmenu				
	8	click on EmployeeList submenu				
	9	select Emp.ID option from search By dropdown				
	10	Enter EmpID in SearchFor textbox				
	11	click on Search button				
	12	click on checkbox to delete employee				
	13	click on delete btn				
	14	click on logout				
	15	close application				

Crate a testsuite (By using xml) to execute end to end flow and then run above 3 test test cases.

1.TC_101_VerifyLogin
2.TC_102_AddNewEmp
3.TC_103_DelEmp

Note: Create automation scripts for below scenarios.(Company Location Test cases)

1.TC_101_VerifyLogin
2.TC_102_Add New Company Location

3.TC_103_Search For newly Added company Location
4.TC_104_Delete company Location
Note: Create automation scripts for below scenarios.(Company Property Test cases)
1.TC_101_VerifyLogin
2.TC_102_Add New Company property
3.TC_103_Delete company property

TestNG (http://testng.org)

Introduction

TestNG is a testing framework that overcomes the limitations of another popular testing framework called JUnit. The "NG" means "Next Generation". Most Selenium users use this more than JUnit because of its advantages. There are so many features of TestNG, but we will only focus on the most important ones that we can use in Selenium.

Advantages of TestNG over JUnit

There are three major advantages of TestNG over JUnit:
Annotations are easier to understand
Test cases can be grouped more easily
Parallel testing is possible

Annotations in TestNG are lines of code that can control how the method below them will be executed. They are always preceded by the @ symbol.

Writing a test is typically a three-step process:

Write the business logic of your test and insert TestNG annotations in your code.
Add the information about your test (e.g. the class name, the groups you wish to run, etc...) in a testng.xml file or in build.xml.
Run TestNG

Some Information on TestNG as follows:

A suite is represented by one XML file. It can contain one or more tests and is defined by the <suite> tag.
A test is represented by <test> and can contain one or more TestNG classes.
A TestNG class is a Java class that contains at least one TestNG annotation. It is represented by the <class> tag and can contain one or more test methods.
A test method is a Java method annotated by @Test in your source.
A TestNG test can be configured by @BeforeXXX and @AfterXXX annotations which allows to perform some Java logic before and after a certain point, these points being either of the items listed above.

Below are the List of TestNG Annotations

@BeforeSuite @AfterSuite @BeforeTest @AfterTest @BeforeGroups @AfterGroups	Configuration information for a TestNG class: @BeforeSuite: The annotated method will be run before all tests in this suite have run. @AfterSuite: The annotated method will be run after all tests in this suite have run.

@BeforeClass @AfterClass @BeforeMethod @AfterMethod	@BeforeTest: The annotated method will be run before any test method belonging to the classes inside the <test> tag is run. @AfterTest: The annotated method will be run after all the test methods belonging to the classes inside the <test> tag have run. @BeforeGroups: The list of groups that this configuration method will run before. This method is guaranteed to run shortly before the first test method that belongs to any of these groups is invoked. @AfterGroups: The list of groups that this configuration method will run after. This method is guaranteed to run shortly after the last test method that belongs to any of these groups is invoked. @BeforeClass: The annotated method will be run before the first test method in the current class is invoked. @AfterClass: The annotated method will be run after all the test methods in the current class have been run. @BeforeMethod: The annotated method will be run before each test method. @AfterMethod: The annotated method will be run after each test method.
alwaysRun	For before methods (beforeSuite, beforeTest, beforeTestClass and beforeTestMethod, but not beforeGroups): If set to true, this configuration method will be run regardless of what groups it belongs to. For after methods (afterSuite, afterClass, ...): If set to true, this configuration method will be run even if one or more methods invoked previously failed or was skipped.
dependsOnGroups	The list of groups this method depends on.
dependsOnMethods	The list of methods this method depends on.
enabled	Whether methods on this class/method are enabled.
groups	The list of groups this class/method belongs to.
inheritGroups	If true, this method will belong to groups specified in the @Test annotation at the class level.
@DataProvider	Marks a method as supplying data for a test method. The annotated method must return an Object[][] where each Object[] can be assigned the parameter list of the test method. The @Test method that wants to receive data from this DataProvider needs to use a dataProvider name equals to the name of this annotation.
name	The name of this data provider. If it's not supplied, the

	name of this data provider will automatically be set to the name of the method.
parallel	If set to true, tests generated using this data provider are run in parallel. Default value is false.
@Factory	Marks a method as a factory that returns objects that will be used by TestNG as Test classes. The method must return Object[].
@Listeners	Defines listeners on a test class.
value	An array of classes that extend org.testng.ITestNGListener.
@Parameters	Describes how to pass parameters to a @Test method.
value	The list of variables used to fill the parameters of this method.
@Test	Marks a class or a method as part of the test.
alwaysRun	If set to true, this test method will always be run even if it depends on a method that failed.
dataProvider	The name of the data provider for this test method.
dataProviderClass	The class where to look for the data provider. If not specified, the data provider will be looked on the class of the current test method or one of its base classes. If this attribute is specified, the data provider method needs to be static on the specified class.
dependsOnGroups	The list of groups this method depends on.
dependsOnMethods	The list of methods this method depends on.
description	The description for this method.
enabled	Whether methods on this class/method are enabled.
expectedExceptions	The list of exceptions that a test method is expected to throw. If no exception or a different than one on this list is thrown, this test will be marked a failure.
groups	The list of groups this class/method belongs to.
invocationCount	The number of times this method should be invoked.
invocationTimeOut	The maximum number of milliseconds this test should take for the cumulated time of all the invocationcounts. This attribute will be ignored if invocationCount is not specified.
priority	The priority for this test method. Lower priorities will be scheduled first.
successPercentage	The percentage of success expected from this method

singleThreaded	If set to true, all the methods on this test class are guaranteed to run in the same thread, even if the tests are currently being run with parallel="methods". This attribute can only be used at the class level and it will be ignored if used at the method level. Note: this attribute used to be called sequential (now deprecated).
timeOut	The maximum number of milliseconds this test should take.
threadPoolSize	The size of the thread pool for this method. The method will be invoked from multiple threads as specified by invocationCount. Note: this attribute is ignored if invocationCount is not specified

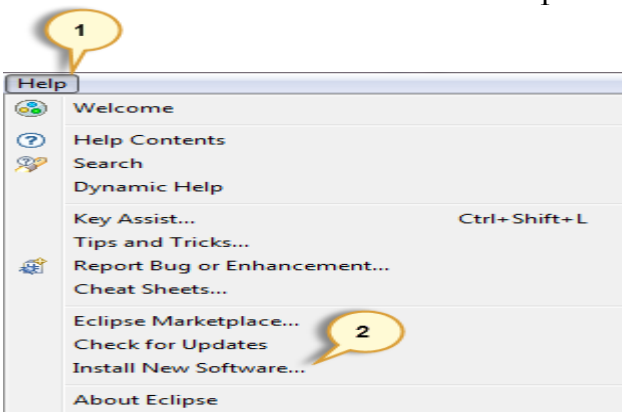
Installing TestNG in Eclipse

Step 1

Launch Eclipse.

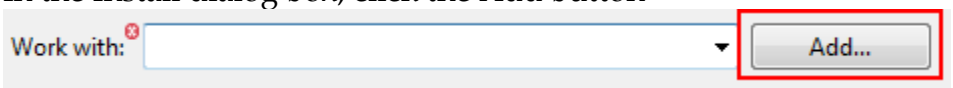
On the menu bar, click Help.

Choose the "Install New Software..." option.



Step 2

In the Install dialog box, click the Add button

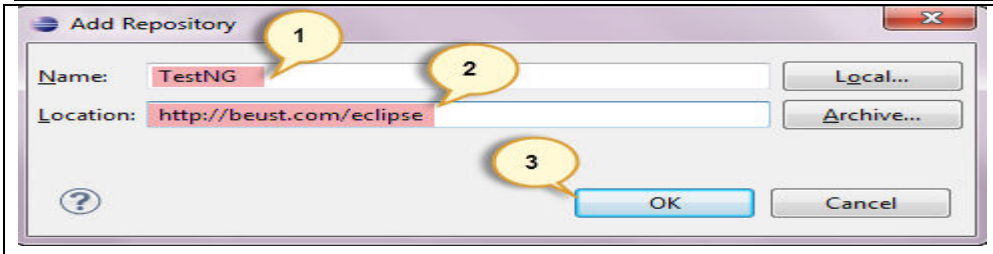


Step 3

In "Name", type TestNG.

In "Location", type http://beust.com/eclipse.

Click OK

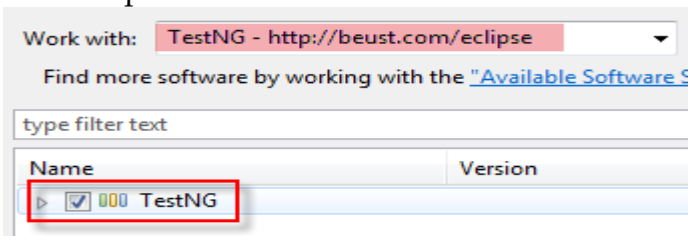


Step 4

Notice that "TestNG - http://beust.com/eclipse" was populated onto the "Work with:" textbox.

Check the "TestNG" check box as shown below, then click Next.

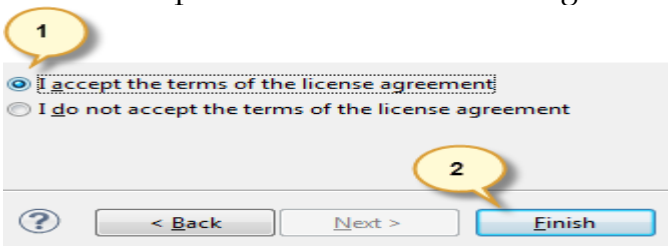
Note: In the latest Eclipse (Kepler) you don't have a checkbox for TestNG, instead you click on question mark (help) icon which will open up the form, and you can select all and installation will continue as for the remaining instructions. Thanks Jana for the tip!



Step 5

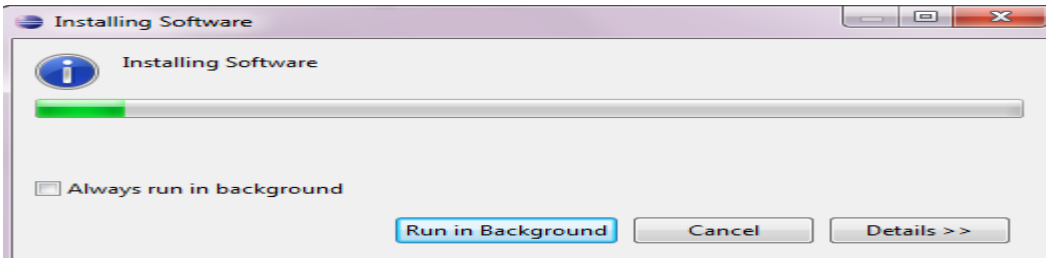
Click next again on the succeeding dialog box until you reach the License Agreement dialog.

Click "I accept the terms of the license agreement" then click Finish.

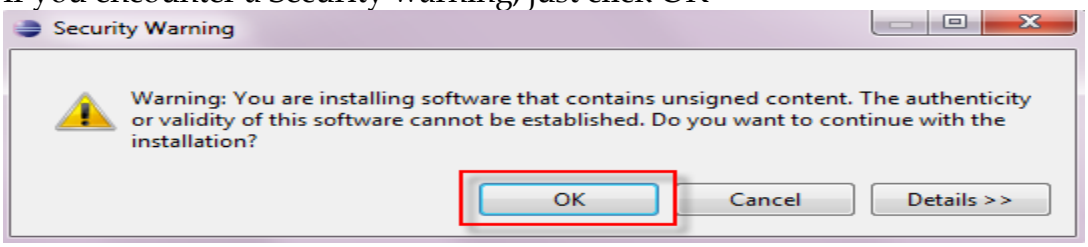


Step 6

Wait for the installation to finish

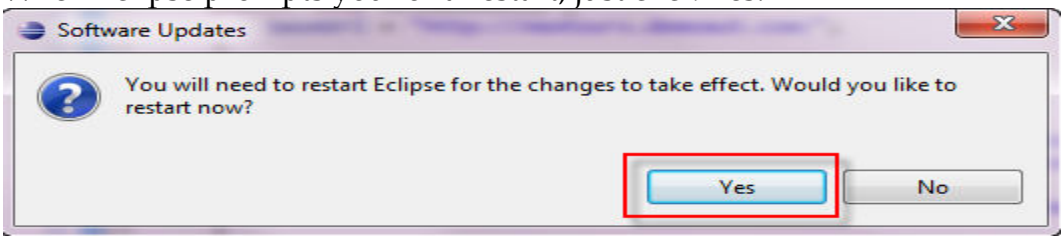


If you encounter a Security warning, just click OK



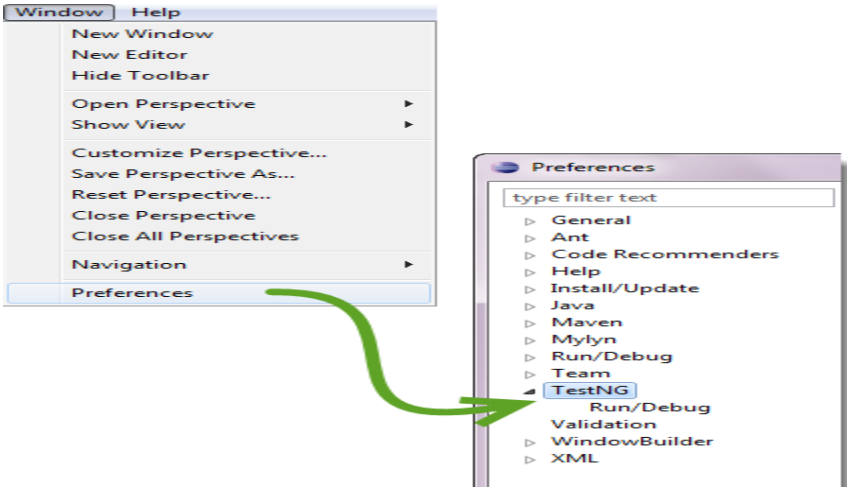
Step 7

When Eclipse prompts you for a restart, just click Yes.



Step 8

After the restart, verify if TestNG was indeed successfully installed. Click Window > Preferences and see if TestNG is included on the Preferences list.



OR

Installation:

Select help menu option in eclipse
Select eclipse market place option
Search for TestNG plugin
Click on install btutton
Click on Next
Accept license
Click on finish btn
Restart eclipse

Step 2: download and Add Testng .Jar file to project

Go to TestNG.org site
Click on downloads
Click on "You can download the current release version of TestNG here".
Add the .jar files to the project

Example:.

```
import org.testng.annotations.Test;
public class TC_TestNG {
        @Test
  public void login(){
          System.out.println("login completed");
  }
        @Test(dependsOnmethods="login")
```

```
  public void logout(){
        System.out.println("Logout completed");
  }
}
```

Note: in above example logout method will get executed only in case of login method got passed other it will skip the logout method

Example for TestNG annotations

```
import org.testng.annotations.AfterClass;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;
public class ExampleForTestNgAnnotations {
        @BeforeClass / @BeforeMethod
public void login(){
        System.out.println("login completed");
}
        @AfterClass / @AfterMethod
public void logout(){
        System.out.println("logout completed");
}
        @Test(priority=2)
        public void addemp()        {
                System.out.println("Adding new emp");
        }
        @Test(priority=1)
        public void delemp()        {
                System.out.println("delete emp");
        }
}
```

Output for @BeforeClass & @AfterClass :Note- @BeforeClass & @AfterClass will executed only one time for the whole program

```
login completed
delete emp
Adding new emp
logout completed
```

Output for @BeforeMethod & @AfterMethod: Note-@BeforeMethod & @AfterMethod will executed for every @Test method

```
login completed
delete emp
logout completed
login completed
Adding new emp
```

logout completed

Example for - Webdriver with Testng

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.Reporter;
import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;
public class WebDriverTestng {
private WebDriver driver;
 @BeforeClass
 public void Startup(){
driver = new FirefoxDriver();
 }
  @Test (description="OrangeHRM Login")
 public void Login() throws Exception{
   Reporter.log("Test case steps");
 driver.get("https://sureshitacademy.in/login.php");
 Reporter.log("1.Application opened");
 driver.findElement(By.name("txtUserName")).sendKeys("suresh");
 Reporter.log("2.typing user name");
 driver.findElement(By.name("txtPassword")).sendKeys("suresh123");
 Reporter.log("3.Typing password");
 driver.findElement(By.name("Submit")).click();
 Thread.sleep(3000);
 Reporter.log("4.login completed");
driver.findElement(By.linkText("Logout")).click();
 }
 @AfterClass
 public void teardown(){
   driver.quit();
 }
}
```

TestNG Sample Report

Test	# Passed	# Skipped	# Failed	Time (ms)	Included Groups	Excluded Groups
Default suite						
Default test	1	0	0	33,780		

Class	Method	Start	Time (ms)
Default suite			
Default test — passed			
TestNGExamples.DragandDrop	testDragnDrop	1411408109316	11095

Default test

TestNGExamples.DragandDrop#testDragnDrop

Example for – Excel + Webdriver +TestNG

Reading the username and password from excelsheet and inputing these values in application using webdriver program and generating the Report with TestNG.

LoginExcel.xls

UserName	Password
sureshit	sureshit

```
import java.io.FileInputStream;
import jxl.Sheet;
import jxl.Workbook;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.Reporter;
import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;
public class TC_Excel_WD_TestNG {
  public WebDriver driver;
@BeforeClass
public void Startup(){
driver = new FirefoxDriver();
}
@AfterClass
public void teardown(){
        driver.quit();
}
@Test
public void login() throws Exception{
        //Reading username and password from excel and assigning to variables
        FileInputStream f1 = new FileInputStream("E:\\Selenium\\LoginExcel.xls");
        Workbook w = Workbook.getWorkbook(f1);
        Sheet s = w.getSheet(0);
        String un = s.getCell(0,1).getContents();
        String pw = s.getCell(1,1).getContents();
        //Typing username and password from Excel file
```

```
        driver.navigate().to("https://sureshitacademy.in/login.php");
        driver.findElement(By.name("txtUserName")).sendKeys(un);
        driver.findElement(By.name("txtPassword")).sendKeys(pw);
        driver.findElement(By.name("Submit")).click();
        Thread.sleep(3000);
        System.out.println("Login completed");
        Reporter.log("Login completed");
        driver.findElement(By.linkText("Logout")).click();
        Reporter.log("Logout completed");
}
}
```

Parallel Execution : Running the testcases with multipul browsers using WebDriver with TestNG

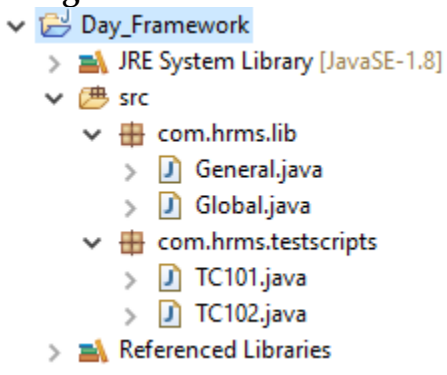
This program will open the two browsers - one is Chrome and another is Firefox and it will execute the test scrips parallel.

```
============= TC_101 =============
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.ie.InternetExplorerDriver;
import org.testng.Reporter;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Parameters;
import org.testng.annotations.Test;
public class TC_101 {
        WebDriver driver;
        @Test
        public void openFF() throws Exception  {
                driver.navigate().to("https://sureshitacademy.in/login.php");
                Reporter.log("Opened HRMS application in Firefox Browser");
                Thread.sleep(5000L);
        }
        @Parameters({"browser"})
        @BeforeMethod
        public void setUp(String browser)          {
                if(browser.equals("Firefox"))                  {
                        driver=new FirefoxDriver();
                }
                else if(browser.equals("IE"))                  {
                driver=new InternetExplorerDriver();
                }
                else if(browser.equals("Chrome"))                      {
                        driver=new ChromeDriver();
```

```
		}
	}
	@AfterMethod
	public void tearDown()	{
		driver.quit();
	}
}
============= TC_102 =============
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.ie.InternetExplorerDriver;
import org.testng.Reporter;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Parameters;
import org.testng.annotations.Test;
public class TC_102 {
WebDriver driver;
	@Test
	public void openChrome() throws Exception	{
		driver.navigate().to("http://www.google.com");
		Thread.sleep(20000);
		Reporter.log("Opened google page in Chrome Browser");
		Thread.sleep(20000);
	}
	@Parameters({"browser"})
	@BeforeMethod
	public void setUp(String browser)		{
		if(browser.equals("Firefox")) {
		driver=new FirefoxDriver();  }
		else if(browser.equals("IE"))		{
		driver=new InternetExplorerDriver();  }
		else if(browser.equals("Chrome")) {
		driver=new ChromeDriver();	}
 }
	@AfterMethod
	public void tearDown()	{
		driver.quit();
	}
}
======browser.xml============
<?xml version="1.0" encoding="UTF-8"?>
<suite name="parallel Suites" parallel="tests">
  <test name="Test in GC">
 <parameter name="browser"  value="Chrome"/>
```

```
  <classes>
    <class name="com.hrms.parallel.TC_102"/>
  </classes>
 </test>
<test name="Test in FF">
 <parameter name="browser"  value="Firefox"/>
  <classes>
    <class name="com.hrms.parallel.TC_101"/>
  </classes>
 </test>
</suite>
```

WebDriver with TestNG Framework:
Steps to Integreate TestNg with our Existing Framework.(WebDriver + TestNG)
Design Framework structure as below in Eclipse



1.First make sure Framework is working fine with WebDriver
2.Intall TestNg in Eclispe
3.Add testng annotation in Actual Automation Sctips
4.Exeute Test Scripts using TestNg by that we need to need be able to see testNg/Html reports to be generated

Framework code need to write as below.
Global.java (Not required any changes in this file)

```
package com.hrms.lib;
import org.openqa.selenium.WebDriver;
public class Global {
//var
   public WebDriver driver;
   public String url = "https://sureshitacademy.in/login.php";
   public String un= " sureshit";
   public String pw= " sureshit";
```

```
//obj
   public String txt_loginname= "txtUserName";
   public String txt_password = "//input[@name='txtPassword']";
   public String btn_login = "Submit";
   public String link_logout = "Logout";
}
```

General.java(Not required any changes in this file)

```
package com.hrms.lib;

import org.openqa.selenium.By;
import org.openqa.selenium.firefox.FirefoxDriver;

public class General extends Global{
//re-fun
   public void openApplication() {
       driver = new FirefoxDriver();
      driver.navigate().to(url);
      System.out.println("Application Opened");
   }
   public void closeApplication() {
      driver.quit();
      System.out.println("Application closed");
   }
   public void login() throws Exception{
      driver.findElement(By.name(txt_loginname)).sendKeys(un);
      driver.findElement(By.xpath(txt_password)).sendKeys(pw);
      driver.findElement(By.name(btn_login)).click();
      Thread.sleep(3000);
      System.out.println("Login completed");
   }
   public void logout() {
      driver.findElement(By.linkText(link_logout)).click();
      System.out.println("Logout completed");
   }
   public void addEmp() {
      System.out.println("Adding new emp");
   }
   public void delEMp() {
      System.out.println("Delete emp");
   }
}
```

TC_101(Remove main method and in place of that write any other method and add @Test to invoke testNG)

```
package com.hrms.testscripts;
import com.hrms.lib.*;
public class TC_101 {
```

```
//public static void main(String args[]) throws Exception{
@Test
Public void tc101() throws Exception{
   //Test case steps
   General obj = new General();
   obj.openApplication();
   obj.login();
   obj.logout();
   obj.closeApplication();
}
}
```

TC_102(Remove main method and in place of that write any other method and add @Test to invoke testNG)

```
package com.hrms.testscripts;
import com.hrms.lib.*;
public class TC_102 {
//public static void main(String args[]) throws Exception{
@Test
Public void tc102() throws Exception{
//test case steps
   General obj = new General();
   obj.openApplication();
   obj.login();
   obj.addEmp();
   obj.delEMp();
   obj.logout();
   obj.closeApplication();
}
}
```

Sample html report shown as below:

Test	Methods Passed	Scenarios Passed	# skipped	# failed	Total Time	Included Groups	Excluded Groups
Default test	1	1	0	0	46.6 seconds		

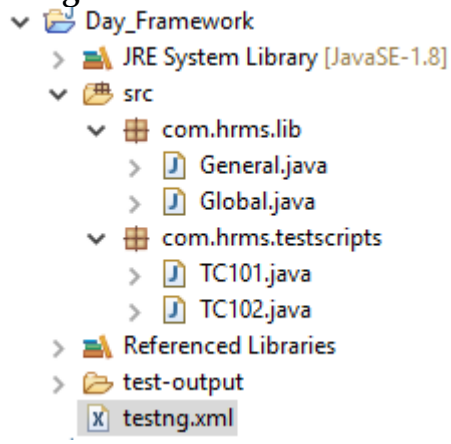
Class	Method	# of Scenarios	Start	Time (ms)
Default test — passed				
com.hrms.testscripts.TC101	tc101	1	1569874518021	46608

Steps to Integreate TestSuite with our Existing Framework.
(WebDriver + TestNG+TestSuite)

1.Make Sure existing framework is working fine with WebDriver and TestNG
2.Create **.xml** for project(xml file name we can provide any thing for now I mentioned as

testng.xml)
3.Write required xml code in **testng.xml** file based on test case execution requirement
4.Right click on **testng.xml** and select the option as – **Run as Testng Suite** option to start the test suite execution.

Design Framework structure as belwo in Eclipse



Framework code need to write as below

Global.java(Not required any changes in this file)

```
package com.hrms.lib;
import org.openqa.selenium.WebDriver;
public class Global {
//var
   public WebDriver driver;
   public String url = "https://sureshitacademy.in/login.php";
   public String un = "sureshit";
   public String pw = " sureshit";
//obj
   public String txt_loginname = "txtUserName";
   public String txt_password = "//input[@name='txtPassword']";
   public String btn_login = "Submit";
   public String link_logout = "Logout";
}
```

General.java - (Not required any changes in this file)

```
package com.hrms.lib;

import org.openqa.selenium.By;
import org.openqa.selenium.firefox.FirefoxDriver;

public class General extends Global{
//re-fun
   public void openApplication() {
      driver = new FirefoxDriver();
      driver.navigate().to(url);
      System.out.println("Application Opened");
   }
   public void closeApplication() {
```

```
        driver.quit();
        System.out.println("Application closed");
    }
    public void login() throws Exception{
        driver.findElement(By.name(txt_loginname)).sendKeys(un);
        driver.findElement(By.xpath(txt_password)).sendKeys(pw);
        driver.findElement(By.name(btn_login)).click();
        Thread.sleep(3000);
        System.out.println("Login completed");
    }
    public void logout() {
        driver.findElement(By.linkText(link_logout)).click();
        System.out.println("Logout completed");
    }
    public void addEmp() {
        System.out.println("Adding new emp");
    }
    public void delEMp() {
        System.out.println("Delete emp");
    }
}
```

TC_101 - (Not required any changes in this file)

```
package com.hrms.testscripts;
import org.testng.annotations.Test;
import com.hrms.lib.*;
public class TC_101 {
//public static void main(String args[]) throws Exception{
    @Test
    public void tc101() throws Exception
    {
    //Test case steps
    General obj = new General();
    obj.openApplication();
    obj.login();
    obj.logout();
    obj.closeApplication();
}
}
```

TC_102 - (Not required any changes in this file)

```
package com.hrms.testscripts;
import org.testng.annotations.Test;
import com.hrms.lib.*;
public class TC_102 {
//public static void main(String args[]) throws Exception{
    @Test
    public void tc102() throws Exception{
```

```
   //test case steps
   General obj = new General();
   obj.openApplication();
   obj.login();
   obj.addEmp();
   obj.delEMp();
   obj.logout();
   obj.closeApplication();
}
}
```

Testng.xml – code -- <classes code need to modify based on your execution >

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Default suite">
 <test verbose="2" name="Default test">
  <classes>
   <class name="com.hrms.testscripts.TC_101"/>
   <class name="com.hrms.testscripts.TC_102"/>
  </classes>
 </test> <!-- Default test -->
</suite> <!-- Default suite -->
```

Sample TestNg results shown as below:

Test	Methods Passed	Scenarios Passed	# skipped	# failed	Total Time	Included Groups	Excluded Groups
Default test	2	2	0	0	53.7 seconds		

Class	Method	# of Scenarios	Start	Time (ms)
Default test — passed				
com.hrms.testscripts.TC_101	tc101	1	1562080139291	28042
com.hrms.testscripts.TC_102	tc102	1	1562080167372	25607

Hybrid Framework

Hybrid Framework is a framework that is created by combining different features of any of the frameworks. Based upon your requirements, you can combine the features of any of the frameworks to come up with your own version of Hybrid Framework.

It is a Combination of the Three Types of Frameworks described before

It has an Ability of Evolving Itself Over a Passage of Time and Over Many Projects

Advantages

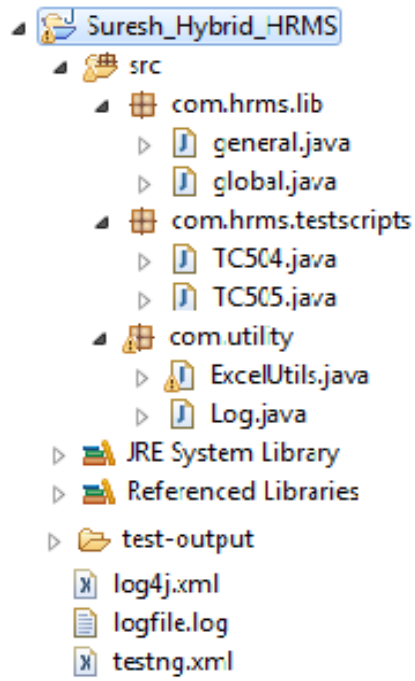
The main advantage of this approach is that you can use the best features from all the types of frameworks to create your own.

Disadvantages

The framework is highly complex and needs very good programming expertise and experience to build a Hybrid Framework from scratch.

Design Project structure as below in Eclipse:

(WebDriver + TestNG+TestSuite+Log4j+Jenkins+Github)



Note: While working with Selenium we had an option to implement Log4j also .In below will discuss in detail to implement Log4j with our Framework.

Log4j-Logger

Selenium with Logs (Log4j)

Sometimes logging is considered to be an overhead upon the existing script creation mechanism but experts considers it to be one of the best practices if used in the accurate proportion because of the following advantages:

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

Logs can also be reviewed to ascertain the application's health by the stakeholders

Log4j – A Java based Logging API

Moving on to the technical details about logging, let us discuss the origin of the API that we would be using throughout the log4j to generate logs. Log4j was a result of collaborative efforts of people at Secure Electronic Marketplace for Europe to develop a utility that would help us generating logs and hence the log4j came into limelight in the year 1996. Log4j is an open source tool and licensed under IBM Public License.

There are three main components that constitute the implementation of log4j. These components represent the details about the log level, formats of the log message in which they would be rendered and their saving mechanisms.

Constituents of Log4j

Loggers

Appenders

Layouts
#1) Loggers
The following steps need to done in order to implement loggers in the project.
Step 1: Creating an instance of Logger class
Step 2: Defining the log level
Logger Class – It is a java based utility that has got all the generic methods already implemented so that we are enabled to use log4j.
Log levels – Log levels are popularly known as printing methods. These are used for printing the log messages. There are primarily five kinds of log levels.
error()
warn()
info()
debug()
log()
Thus, to be able to generate logs, all we need to do is to call any of the printing method over the logger instance. We will have a broader look into it during the implementation phase.
#2) Appenders
Now that we know how to generate these logs, the next thing that should pop up into our minds is that where do I get to view the logs? The answer to this question lies in the definition of "Appenders".
Appenders are consistently used to specify the data source/medium where the logs should be generated. The scope of data sources stretches from various external mediums like console, GUI, text files etc.
#3) Layouts
At times, user wishes certain information to be pre – pended or appended with each log statement. For example I wish to a print a timestamp along with my log statement. Thus, such requirements can be accomplished by "Layouts".
Layouts are a utility that allows the user to opt for a desired format in which the logs would be rendered. Appenders and Layout have a tight coupling between them. Thus, we are required to map each of the appender with a specific layout.
Take a note that user is leveraged to define multiple appenders, each mapped with a distinct layout

Follow below Steps to Implement Log4j in our TestNG framework
Note: Make sure your project is working fine upto test suite and then implement log4j by adding below steps.
1.Create one New Package as **com.hrms.utility**
2.Create one New Class with Class Name as **Log.java**
3.Create **info()** method in Log.java
4.Download **log4j.jar (http://logging.apache.org/)** file and add log4j.jar file to our Project
5.Create one new Xml file and Name as **log4j.xml**
6.Copy the required the xml code to log4j.xml
7.Call Log.info() method in required functions on General.java file
8.Finally call and configure log4j.xml in Each and every TestCase

Log.java – write code as below.

```
package com.hrms.utility;
import org.apache.log4j.Logger;
public class Log {
   //Initialize Log4j logs
   private static Logger Log = Logger.getLogger(Log.class.getName());
   // Need to create these methods, so that they can be called
   public static void info(String message) {
     Log.info(message);
     }
   public static void error(String message) {
        Log.error(message);
        }
   }
```

Log4j.xml --- copy below code in log4j.xml and not required any changes in .xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/" debug="false">
<appender name="fileAppender" class="org.apache.log4j.FileAppender">
<param name="Threshold" value="INFO" />
<param name="Append" value="true" />
<param name="File" value="logfile.log"/>
<layout class="org.apache.log4j.PatternLayout">
<param name="ConversionPattern" value="%d %-5p [%c{1}] %m %n" />
</layout>
</appender>
<root>
<level value="INFO"/>
<appender-ref ref="fileAppender"/>
</root>
</log4j:configuration>
```

Global.java – not required any changes in this file

```
package com.hrms.lib;
import org.openqa.selenium.WebDriver;
public class Global {
//var
   public WebDriver driver;
   public String url = "https://sureshitacademy.in/login.php";
   public String un = "sureshit";
   public String pw = "sureshit";
//obj
   public String txt_loginname = "txtUserName";
   public String txt_password = "//input[@name='txtPassword']";
   public String btn_login = "Submit";
```

```
    public String link_logout = "Logout";
}
```

General.java –Add Log.info() method in all required functions

```
package com.hrms.lib;
import org.openqa.selenium.By;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.Reporter;
import com.hrms.utility.Log;
public class General extends Global{
//re-fun
    public void openApplication() {
        driver = new FirefoxDriver();
        driver.navigate().to(url);
        System.out.println("Application Opened");
        Reporter.log("Application Opened");
        Log.info("Application Opened");
    }
    public void closeApplication() {
        driver.quit();
        System.out.println("Application closed");
        Log.info("Application closed");
    }
    public void login() throws Exception{
        driver.findElement(By.name(txt_loginname)).sendKeys(un);
        driver.findElement(By.xpath(txt_password)).sendKeys(pw);
        driver.findElement(By.name(btn_login)).click();
        Thread.sleep(3000);
        System.out.println("Login completed");
        Log.info("Login completed");
    }
    public void logout() {
        driver.findElement(By.linkText(link_logout)).click();
        System.out.println("Logout completed");
        Log.info("Logout completed");
    }
    public void addEmp() {
        System.out.println("Adding new emp");
    Log.info("Add new emp");
    }
    public void delEMp() {
        System.out.println("Delete emp");
        Log.info("Del emp");
    }
}
```

TC_101 – configure log4j.xml in each and every test script

```
package com.hrms.testscripts;
import org.apache.log4j.xml.DOMConfigurator;
import org.testng.annotations.Test;

import com.hrms.lib.*;
public class TC_101 {
//public static void main(String args[]) throws Exception{
   @Test
   public void tc101() throws Exception
   {
   //Test case steps
   DOMConfigurator.configure("log4j.xml");
   General obj = new General();
   obj.openApplication();
   obj.login();
   obj.logout();
   obj.closeApplication();
}
}
```

TC_102 – configure log4j.xml

```
package com.hrms.testscripts;
import org.apache.log4j.xml.DOMConfigurator;
import org.testng.annotations.Test;

import com.hrms.lib.*;
public class TC_102 {
//public static void main(String args[]) throws Exception{
   @Test
   public void tc102() throws Exception{
   //test case steps
      DOMConfigurator.configure("log4j.xml");
      General obj = new General();
   obj.openApplication();
   obj.login();
   obj.addEmp();
   obj.delEMp();
   obj.logout();
   obj.closeApplication();
}
}
```

testng.xml --- update <classes code based on test case execution requirement>

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
```

```
<suite name="Default suite">
  <test verbose="2" name="Default test">
    <classes>
      <class name="com.hrms.testscripts.TC_101"/>
      <class name="com.hrms.testscripts.TC_102"/>
    </classes>
  </test> <!-- Default test -->
</suite> <!-- Default suite -->
```

Now execute the test case, once execution had completed its need be generate logfile as below.

```
2023-06-12 08:18:38,469 INFO  [Log] Application opened
2023-06-12 08:18:39,704 INFO  [Log] Login completed
2023-06-12 08:18:40,324 INFO  [Log]  logout completed
2023-06-12 08:18:42,527 INFO  [Log] Application closed
```

Jenkins

Jenkins - History

2005 - Hudson was first release by Kohsuke Kawaguchi of Sun Microsystems
2010 – Oracle bought Sun Microsystems
Due to a naming dispute, Hudson was renamed to Jenkins
Oracle continued development of Hudson (as a branch of the original)

About Jenkins

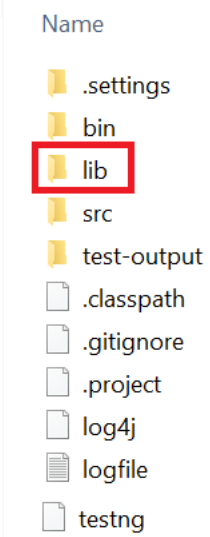
Jenkins is an open source tool written in Java.
Jenkins is CI (Continuous Integration) tool which will help you to run test in easy manner.
Jenkins is a self-contained, open source automation server which can be used to automate all sorts of tasks related to building, testing, and delivering or deploying software.

Steps to configure Jenkins with Selenium

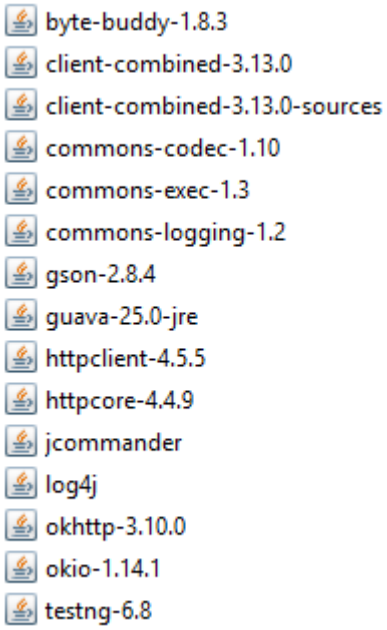
1.Generate batch file
2.Download Jenkins .exe file and Install the same
3.Configure pre-defined settings(JDK path and Email Configuration)
4.Create a job Schedule

Steps to Generate bacth file

Create lib folder in project root directory



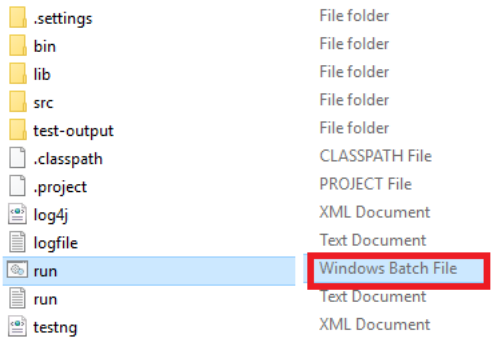
Copy all the required jar files to run the project in to created **lib** folder



Create new notepad file and provide the code as below.(Note: project location will be your project path of workspace)

```
set projectLocation=D:\seleniumTrg_ws\Day_MFW_105
cd %projectLocation%
set classpath=%projectLocation%\bin;%projectLocation%\lib\*
java org.testng.TestNG %projectLocation%\testng.xml
```

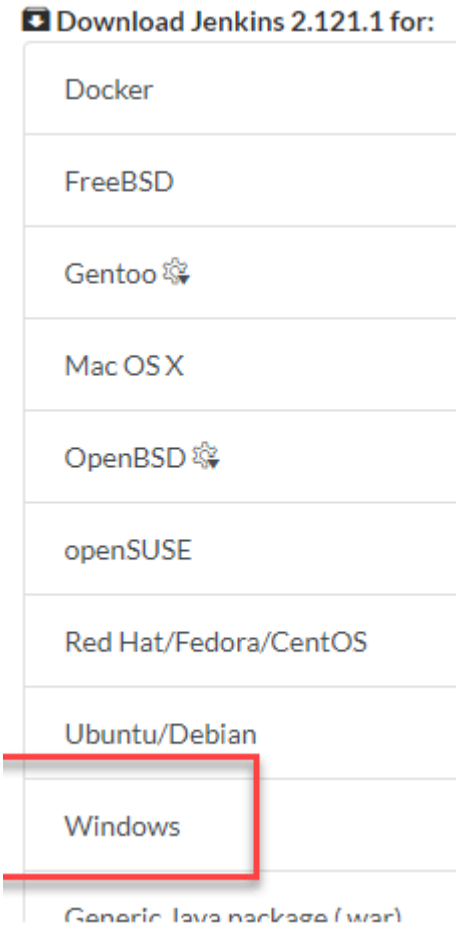
Save the file as run.bat ,by that batch file will be created



Perform double click on run.bat , by that you should be able to see that scripts are running. Once scripts are running then we can confirm batch file created successfully.

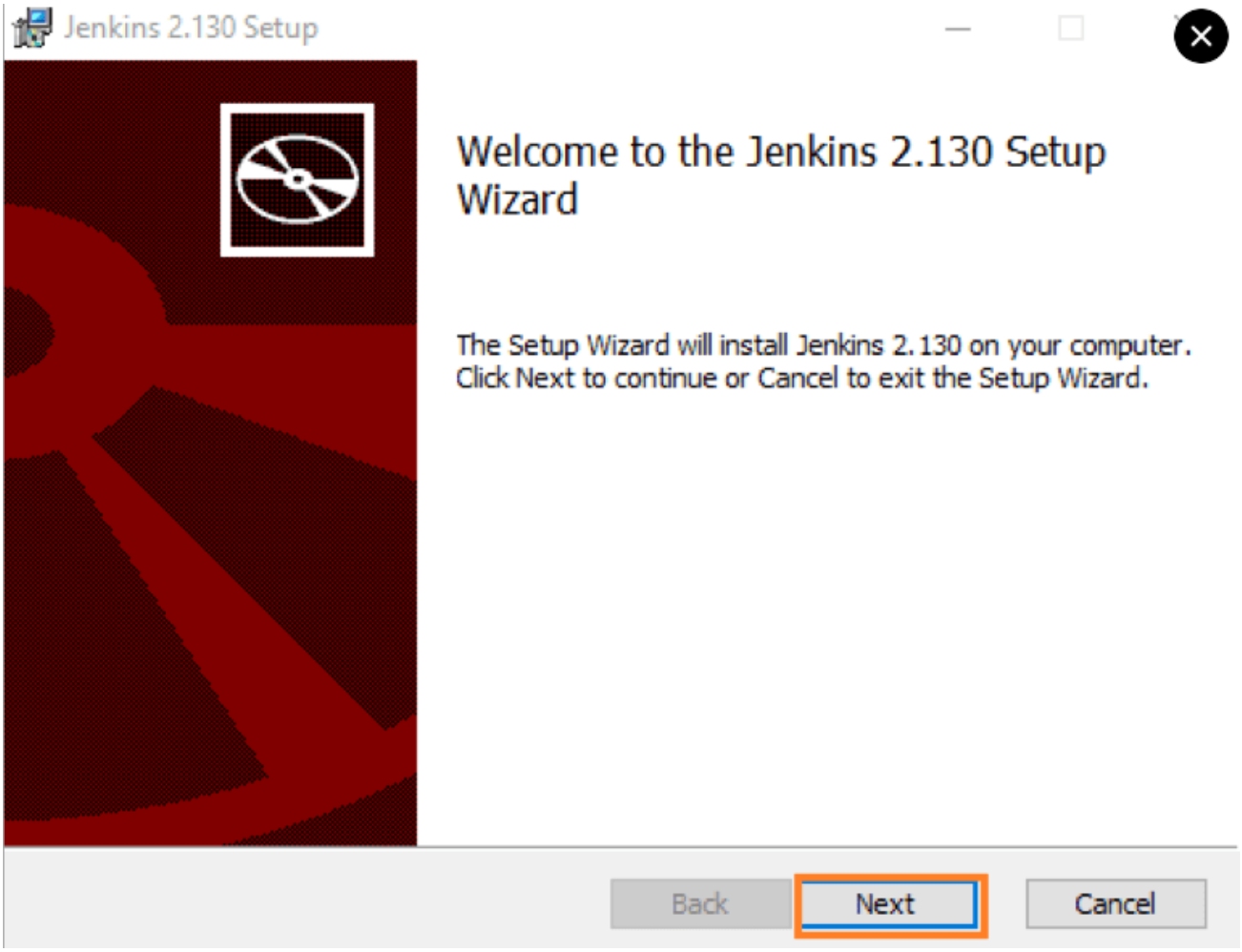
Steps to Install Jenkins

Got to https://jenkins.io/download/ and select the platform. In my system working with Windows

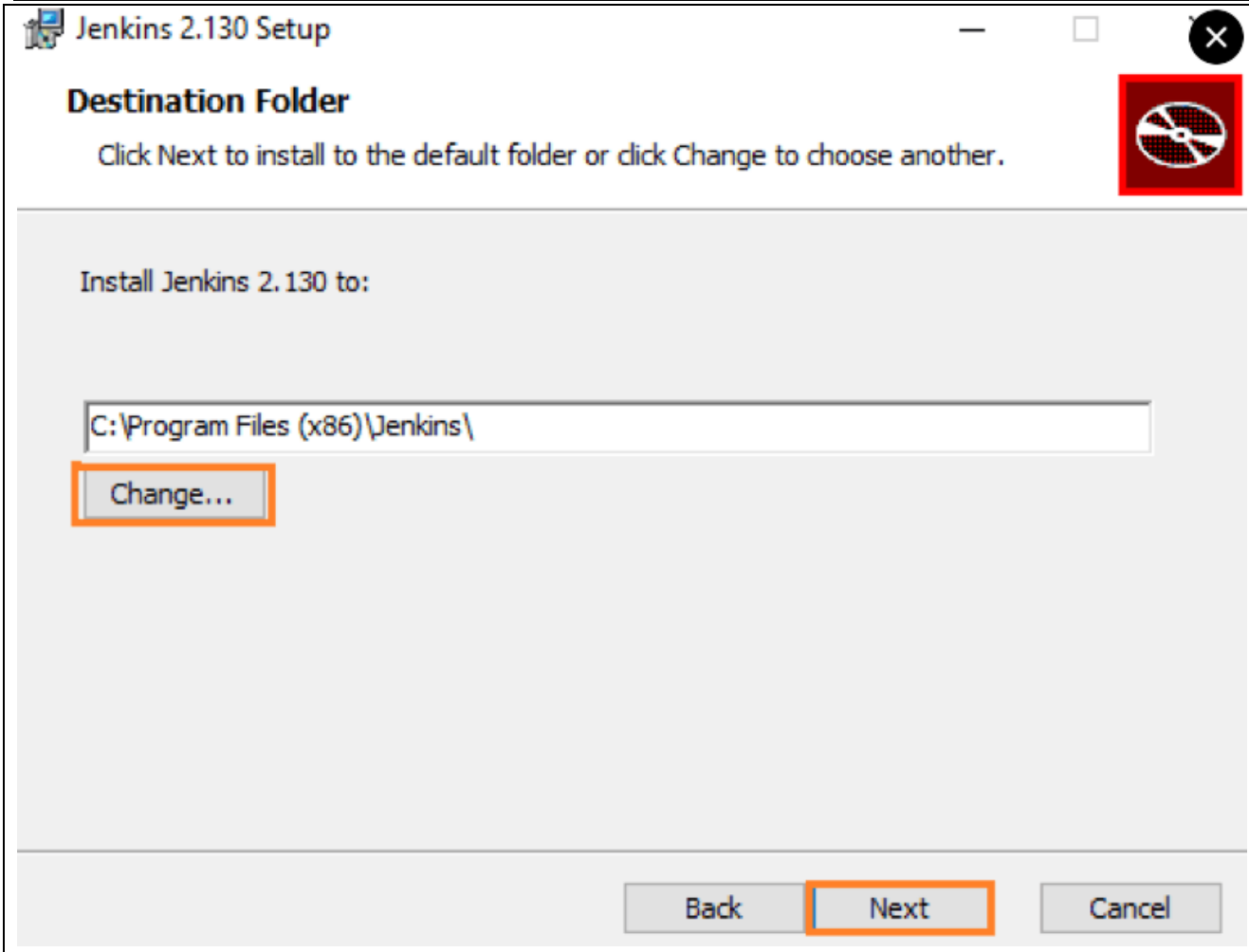


Unzip the file to a folder and double click on the Jenkins.exe file

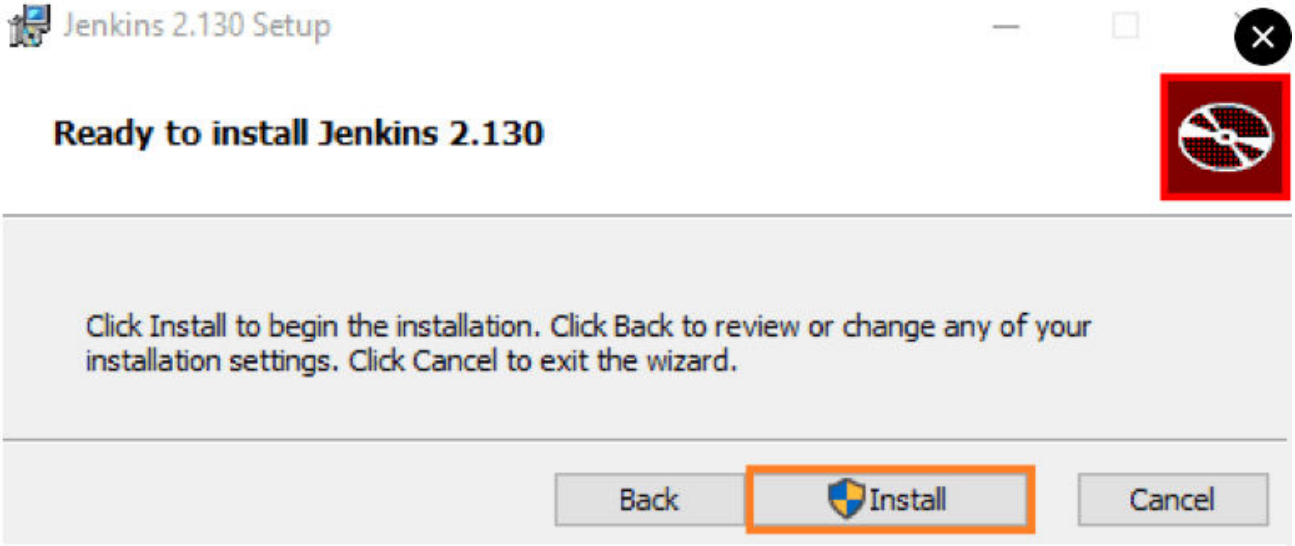
Click "Next" to start the installation.



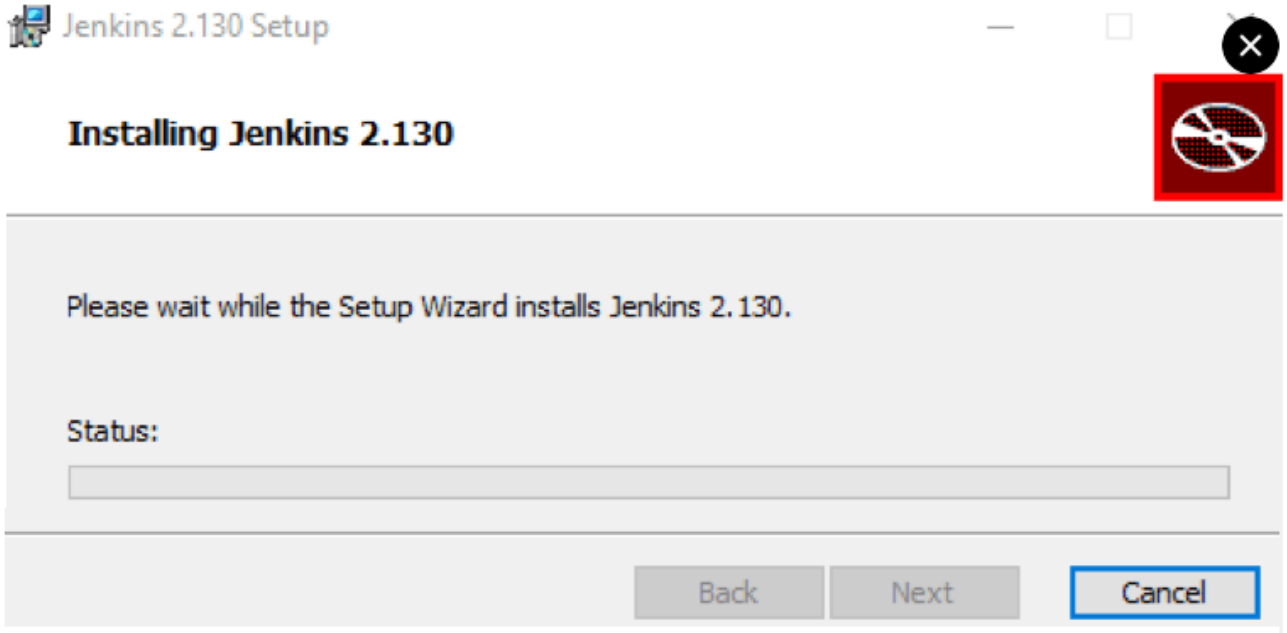
Click the "Change…" button if you want to install Jenkins in another folder. In this example I will keep the default option and click on the "Next" button.



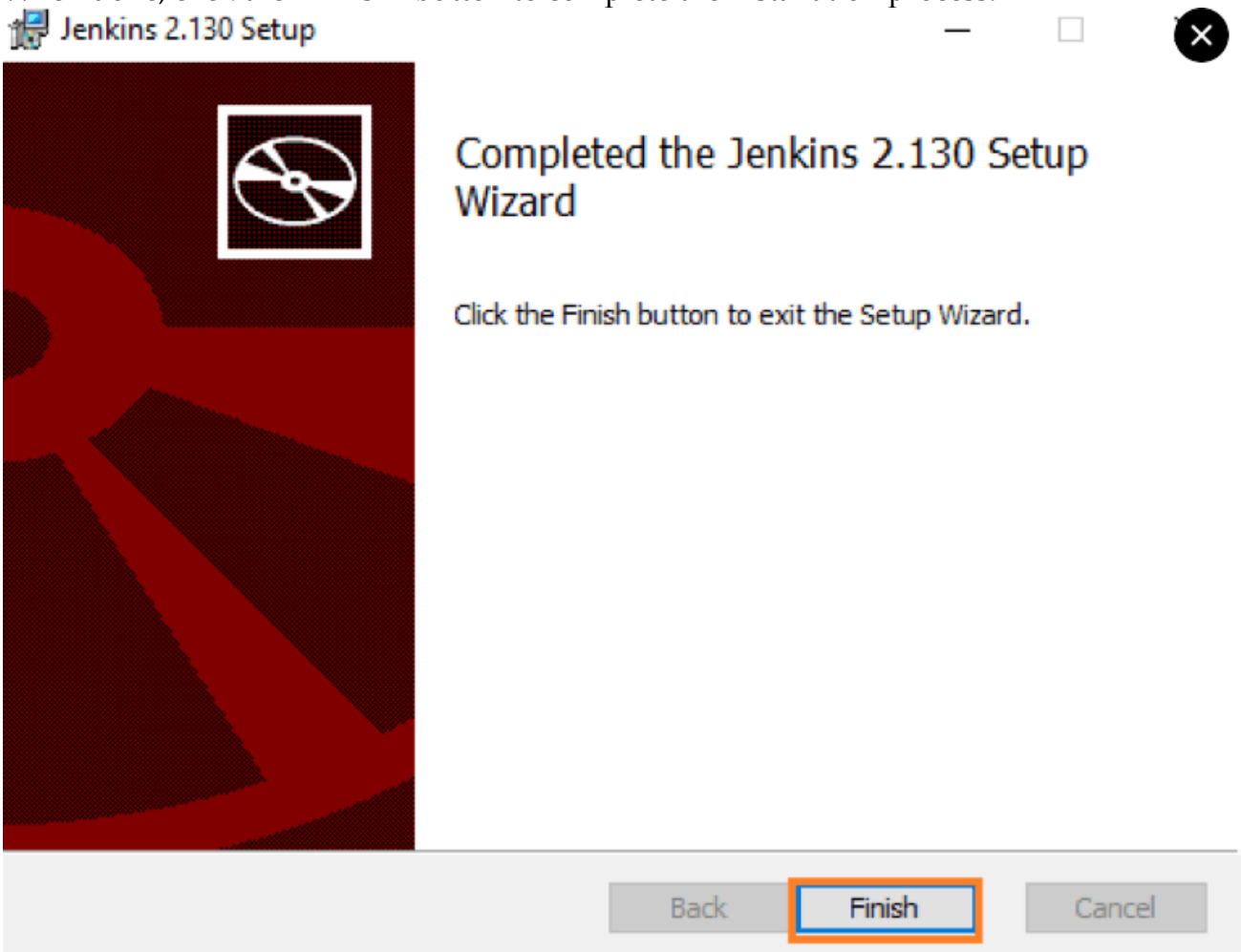
Click the "Install" button to start the installation process.



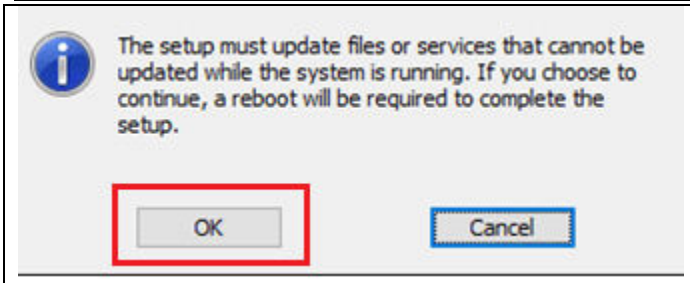
The installation is processing.



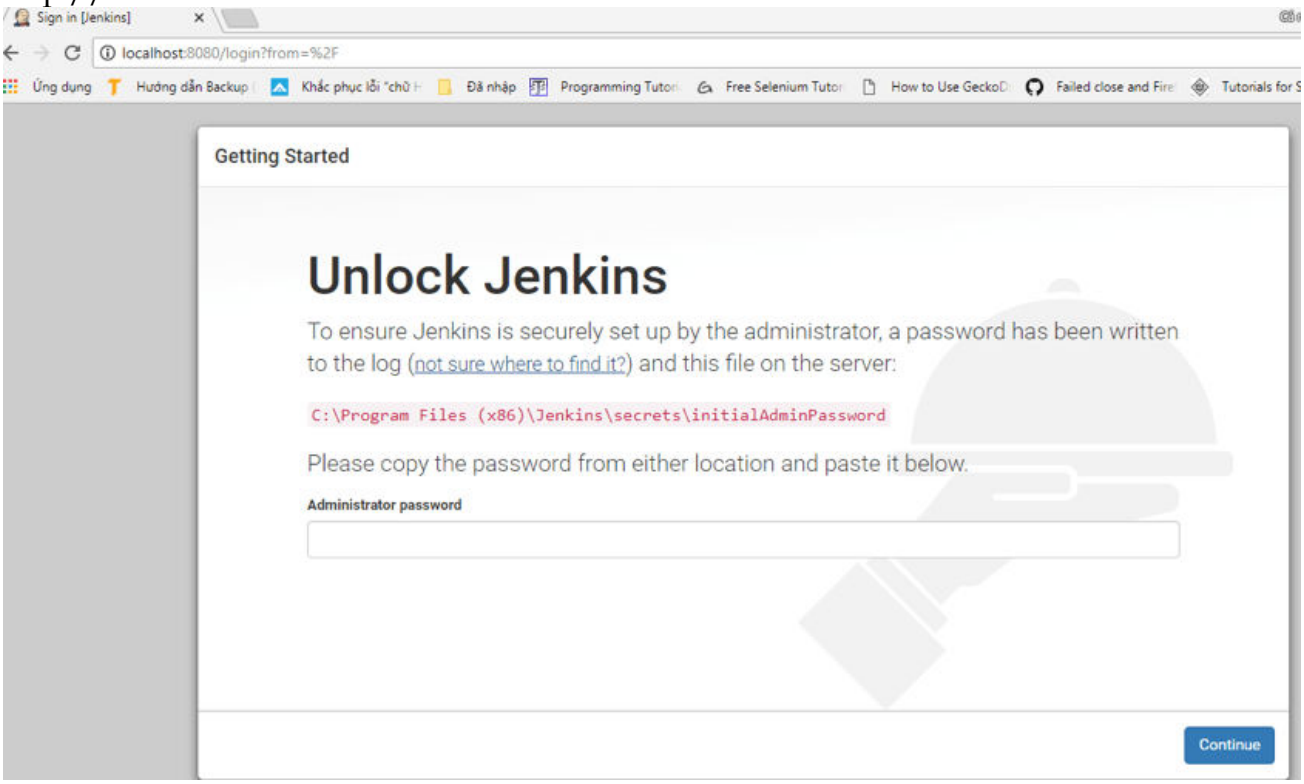
When done, click the "Finish" button to complete the installation process.



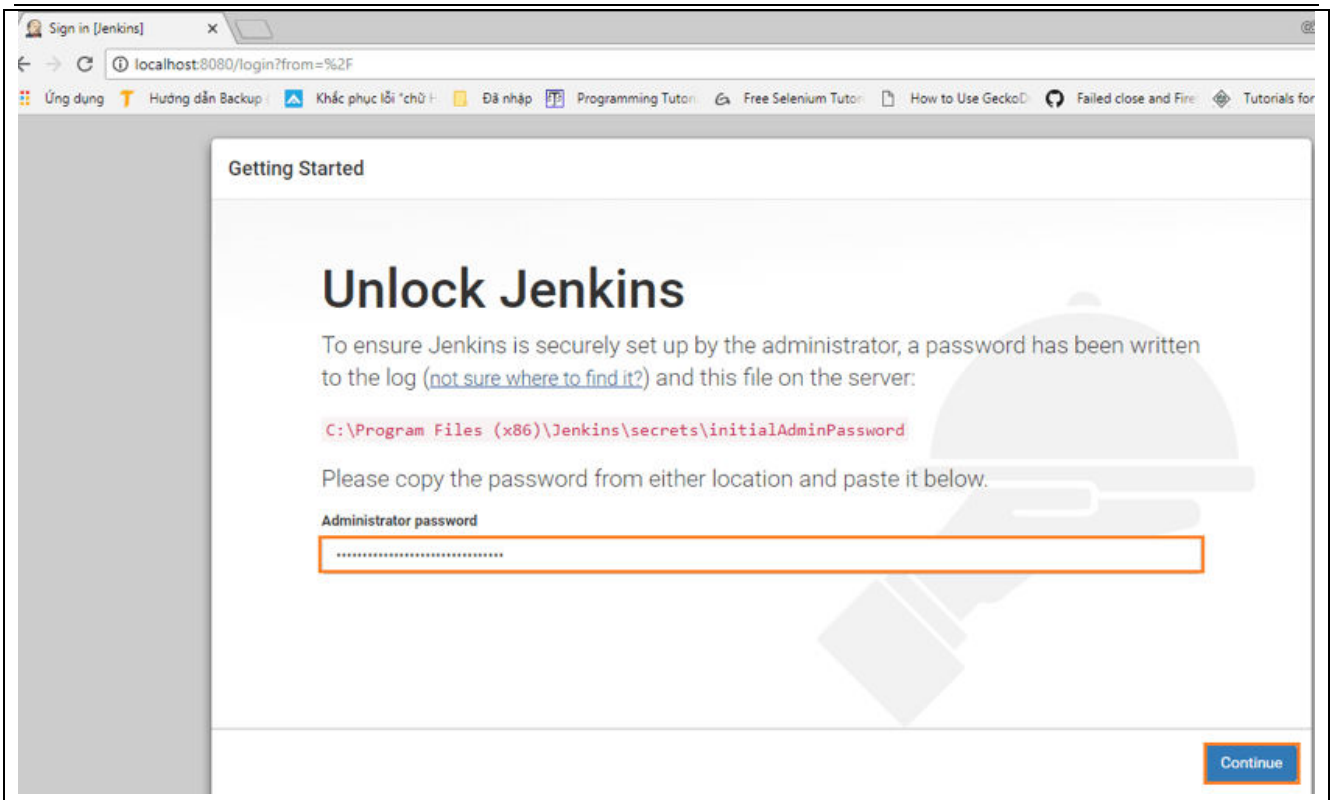
During the installation process an info panel may pop-up to inform the user that for a complete setup, the system should be rebooted at the end of the current installation. Click on OK button when the Info panel is popping-up:



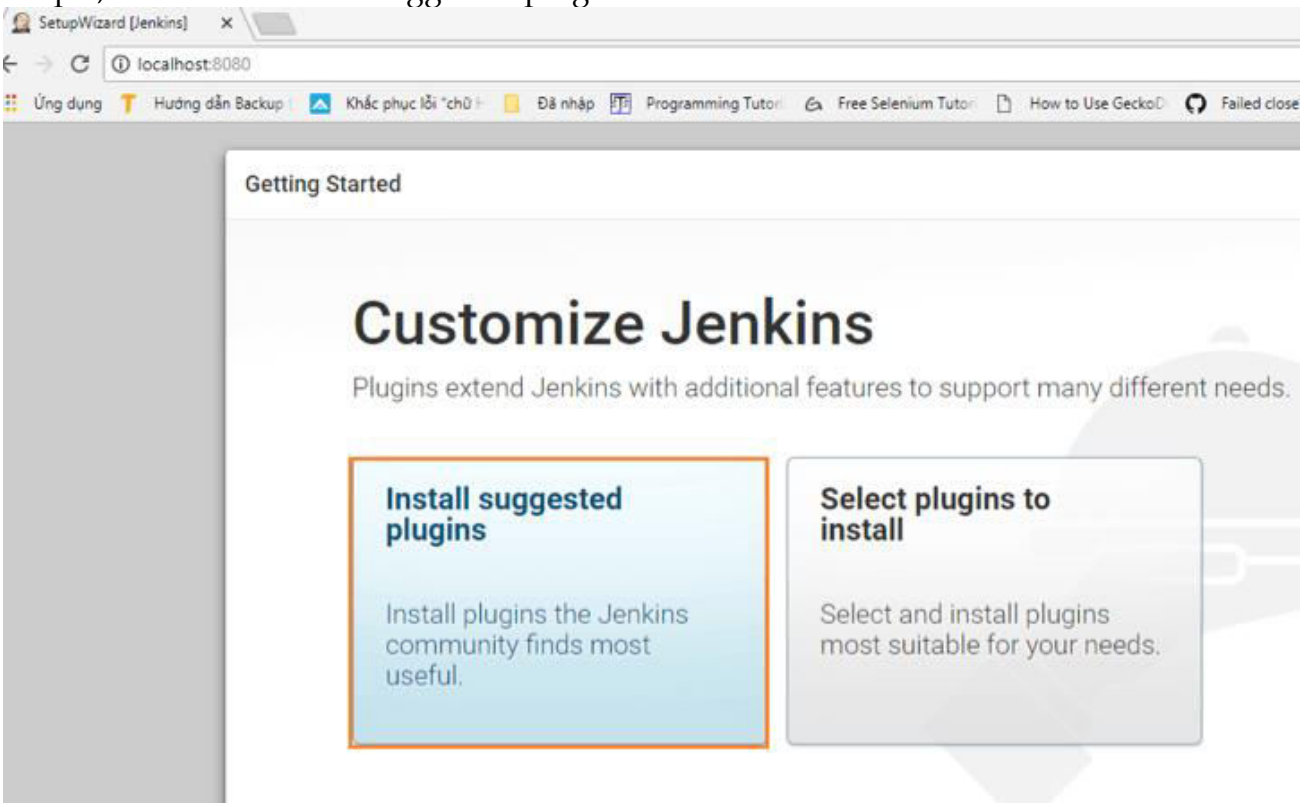
You will automatically be redirected to a local Jenkins page, or you can paste the URL http://localhost:8080 in a browser.



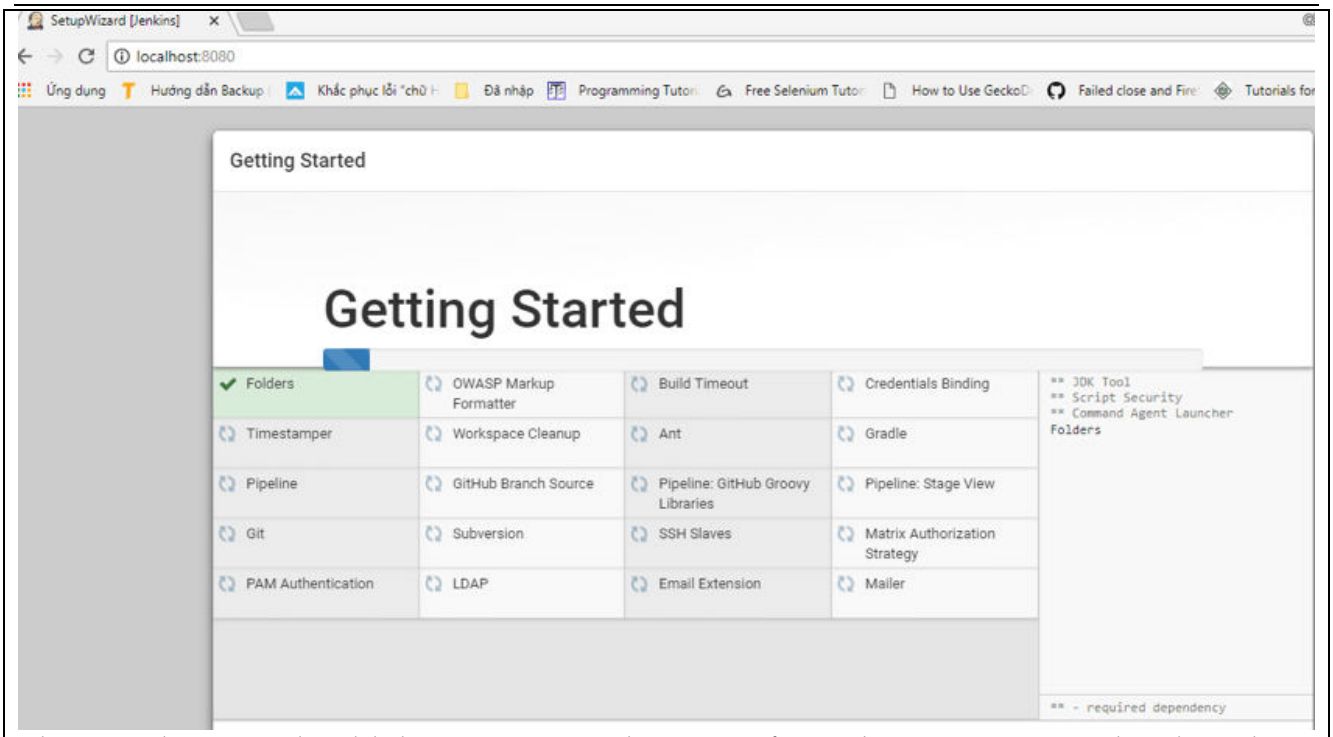
To unlock Jenkins, copy the password from the file at C:\Program Files (x86)\Jenkins\secrets\initialAdminPassword and paste it in the Administrator password field. Then, click the "Continue" button.



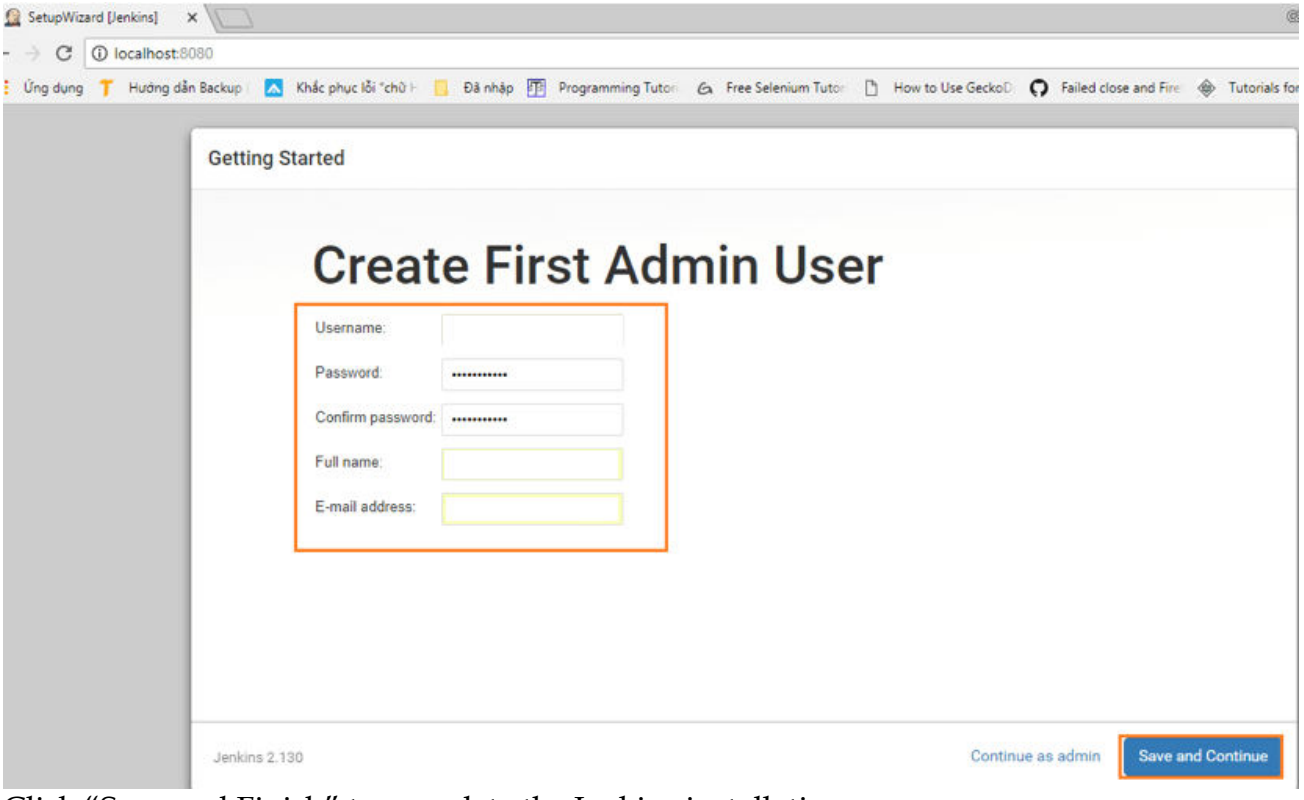
You can install either the suggested plugins or selected plugins you choose. To keep it simple, we will install the suggested plugins.



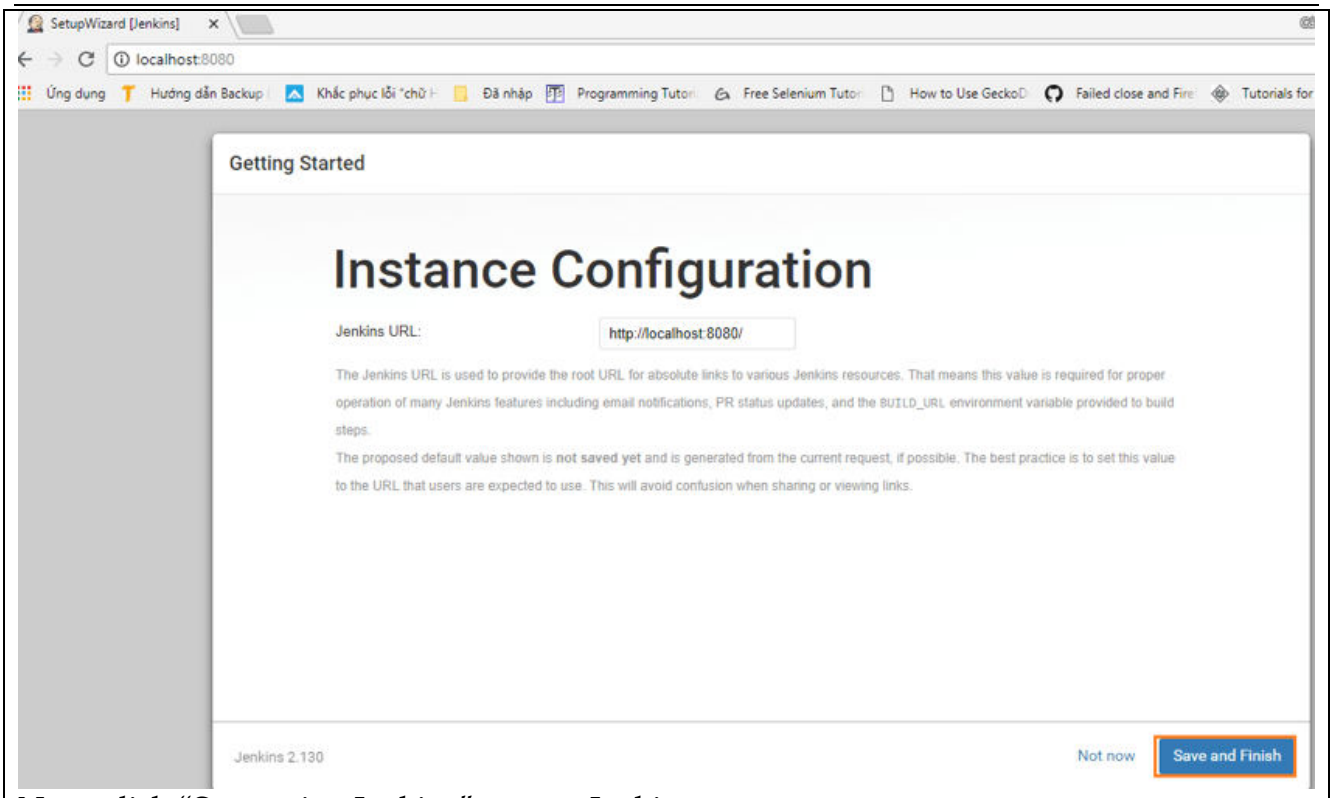
Wait until the plugins are completely installed.



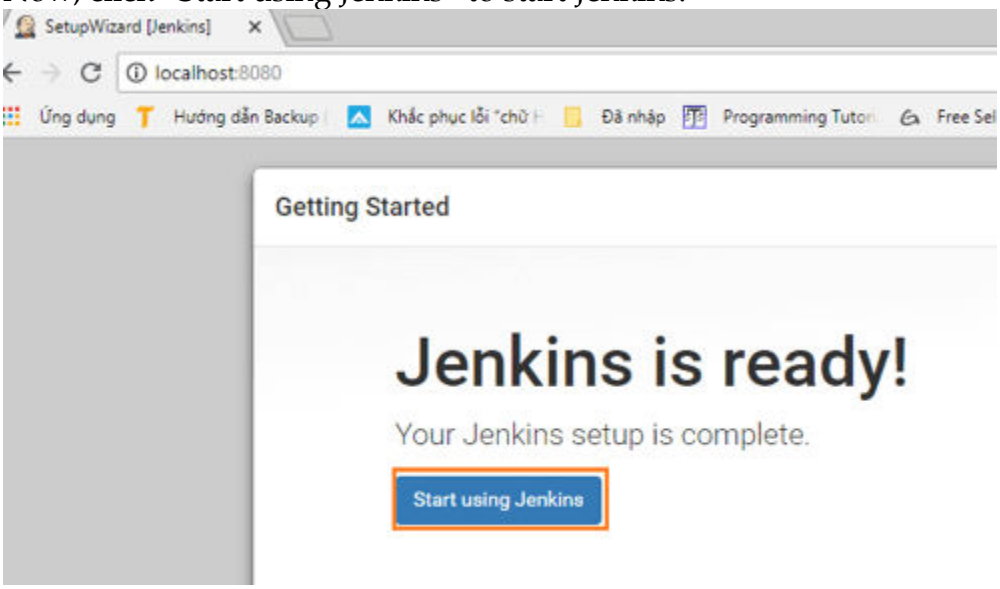
The next thing we should do is create an admin user for Jenkins. Put in your details and click "Save and Continue".



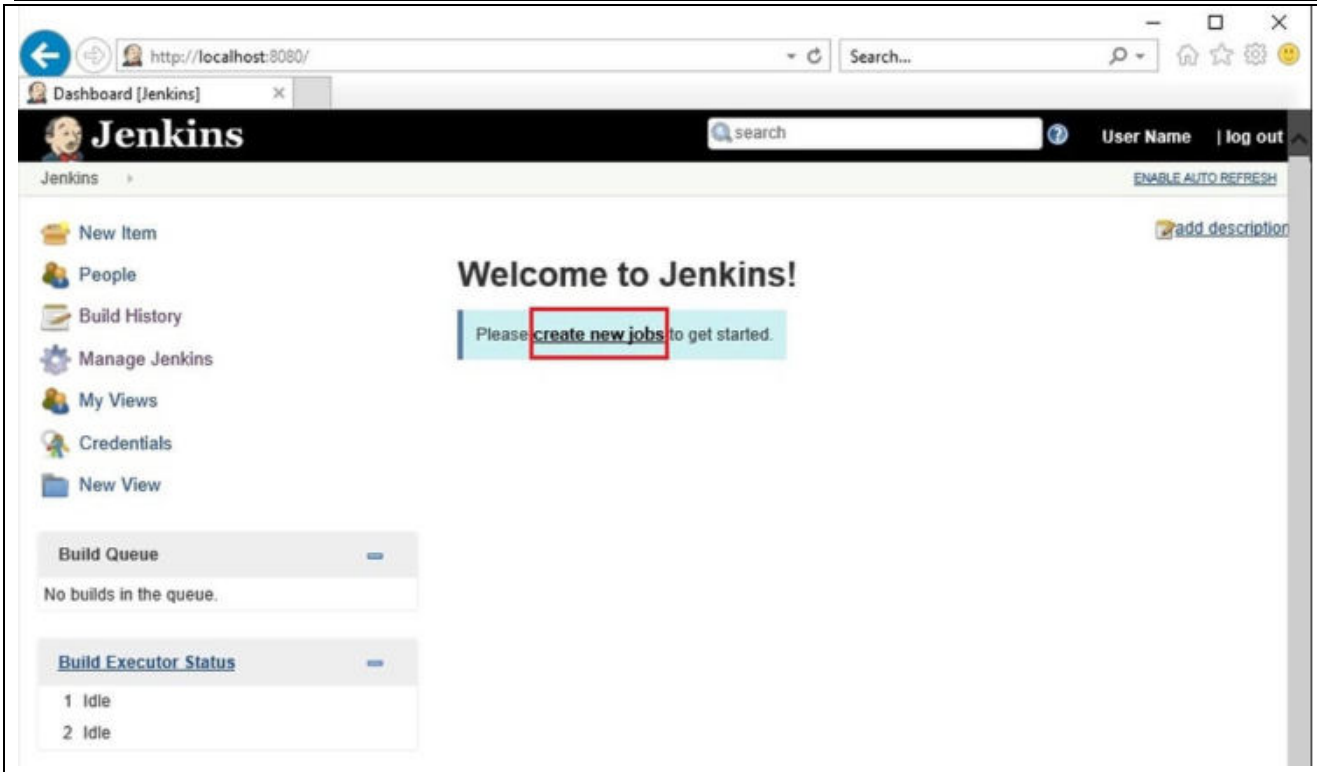
Click "Save and Finish" to complete the Jenkins installation.



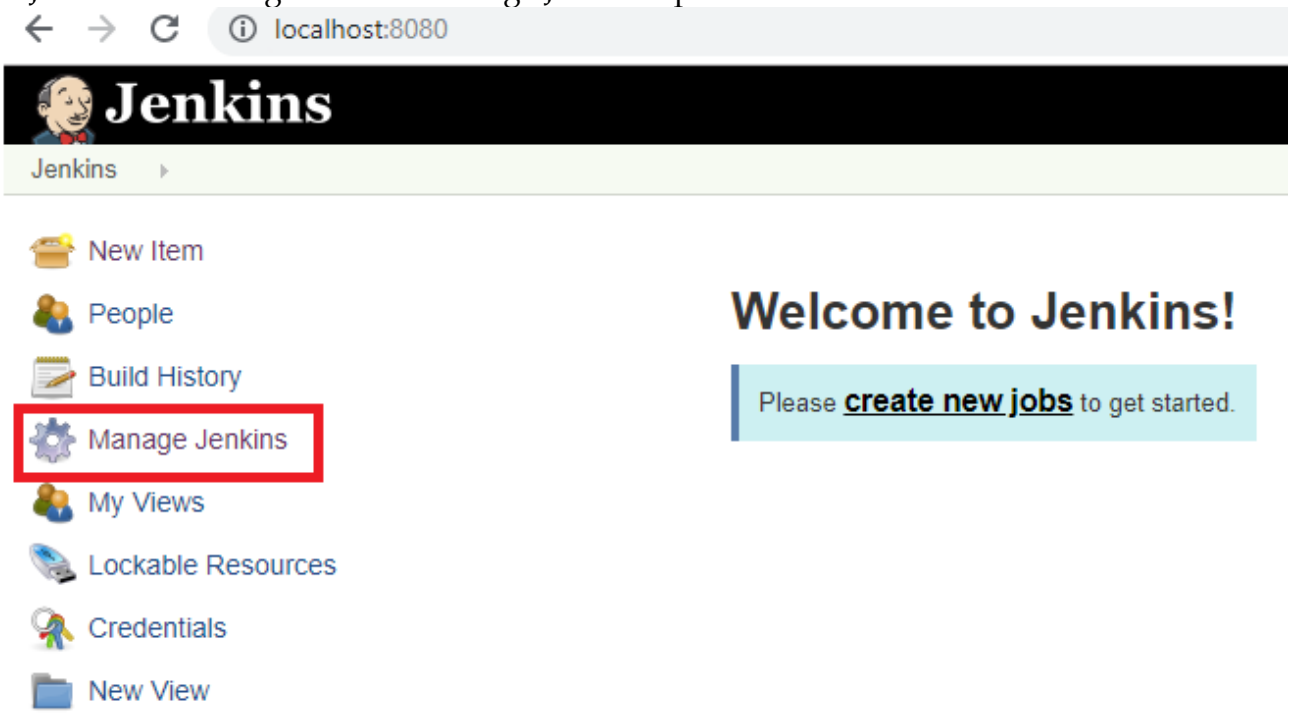
Now, click "Start using Jenkins" to start Jenkins.



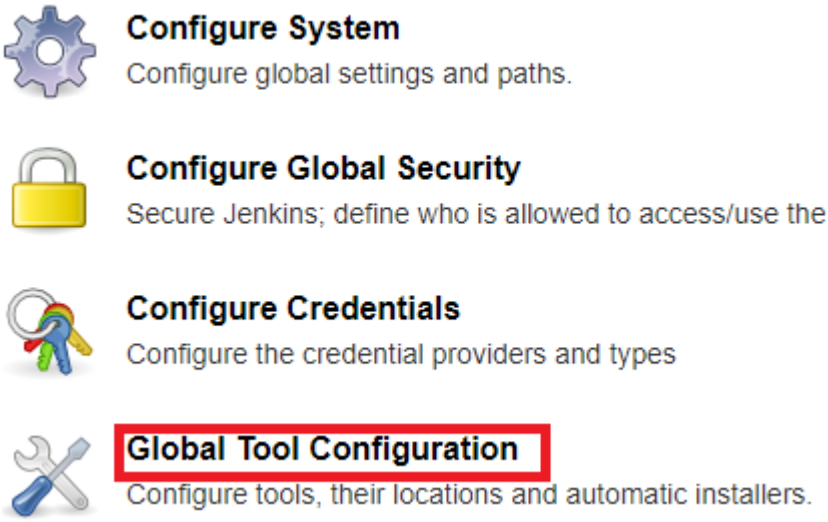
Finally, Jenkins page shown as below.



Configure pre-defined settings(**JDK path and Email Configuration**)
In Jenkins HomePage click on Manage Jenkins option

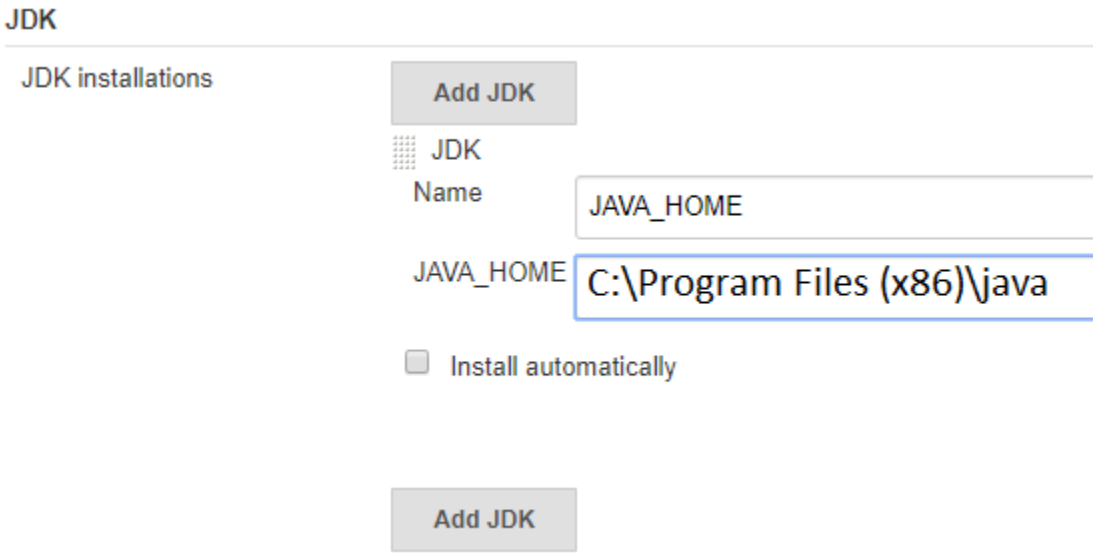


Now click on Global Tool Configuration option



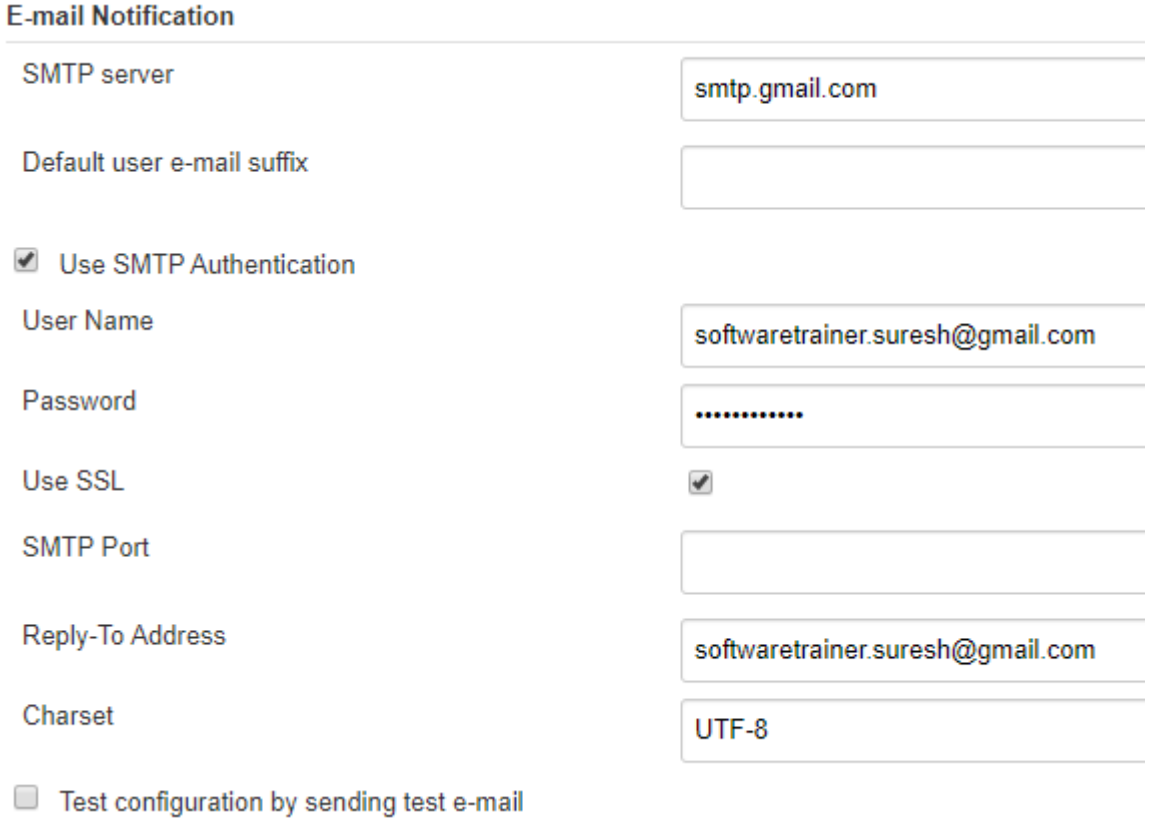
Now, set the JDK path. – Click on JDK installation button or Add JDK button.

Provide JDK Name as JAVA_HOME and JAVA_HOME (JDK path installed in your system) ,make sure you Uncheck the install automatically check box



Click on Apply and Save buton
Next again Goto Manage Jenkins page and **click on Configure System** option to configure E-mail notification details

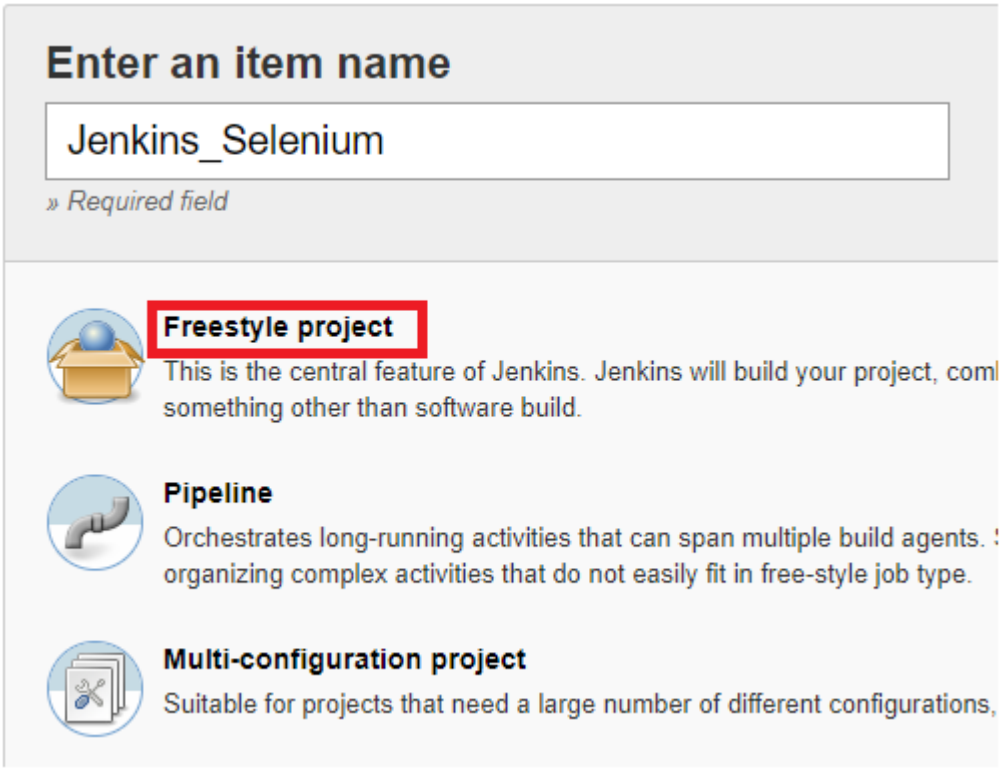
Provide required details as below,username and password will be yours gmail id and that gmail password need to be entered.



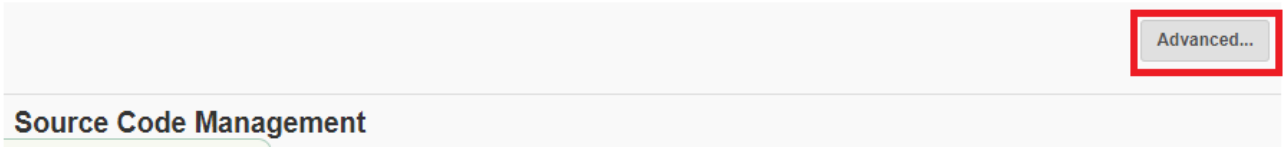
Next click on SAVE and APPLY button

Create a job Schedule – To Execute Selenium TestScripts with Jenkins

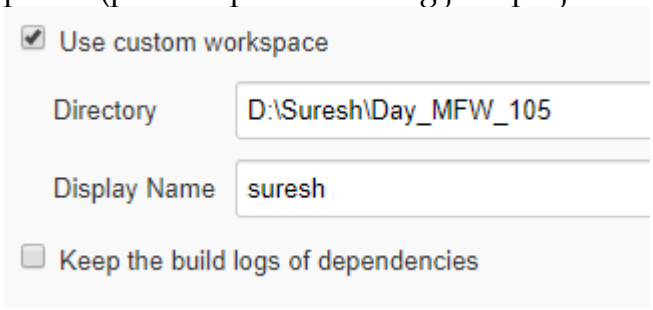
In Jenkins HomePage click on New Item
- Select FreeStyle Project, as selected in the image
- Provide any job a name by writing it in Item Name text box
- Now, click on OK



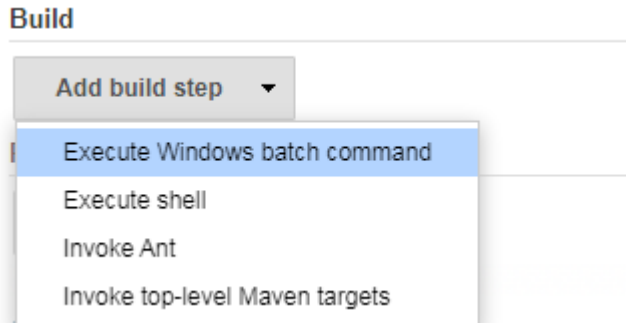
Click on Advanced button



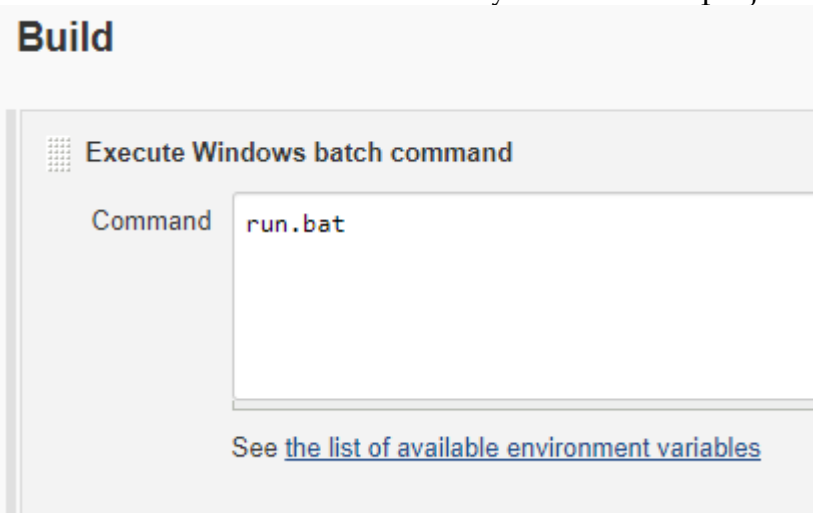
Click on use custom workspace checkbox and give your Selenium script project workspace path – (provide path including java project name)



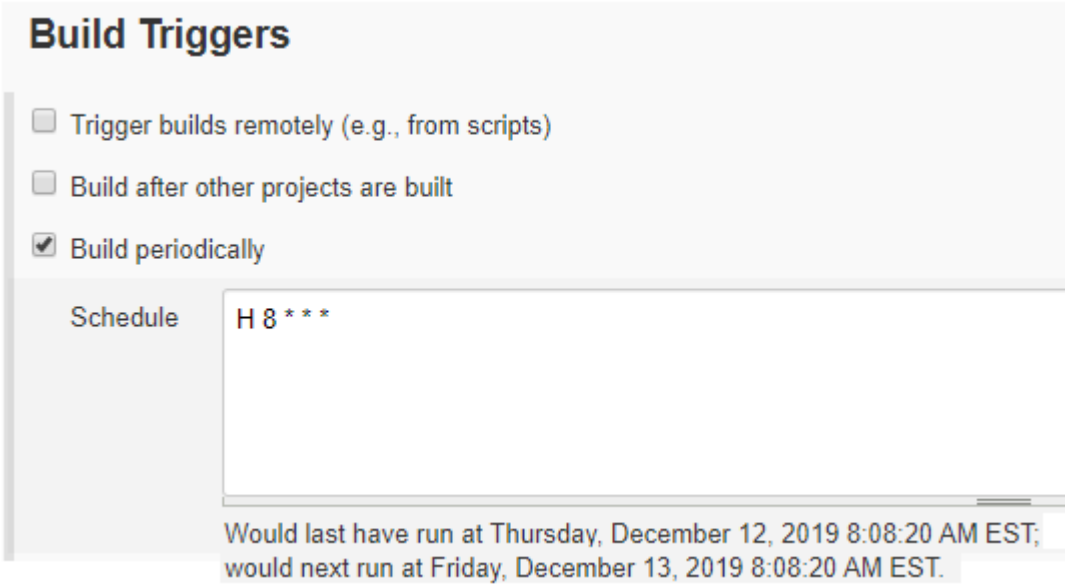
Then go to Build and Select – Execute Windows batch command option from the drop-down box.



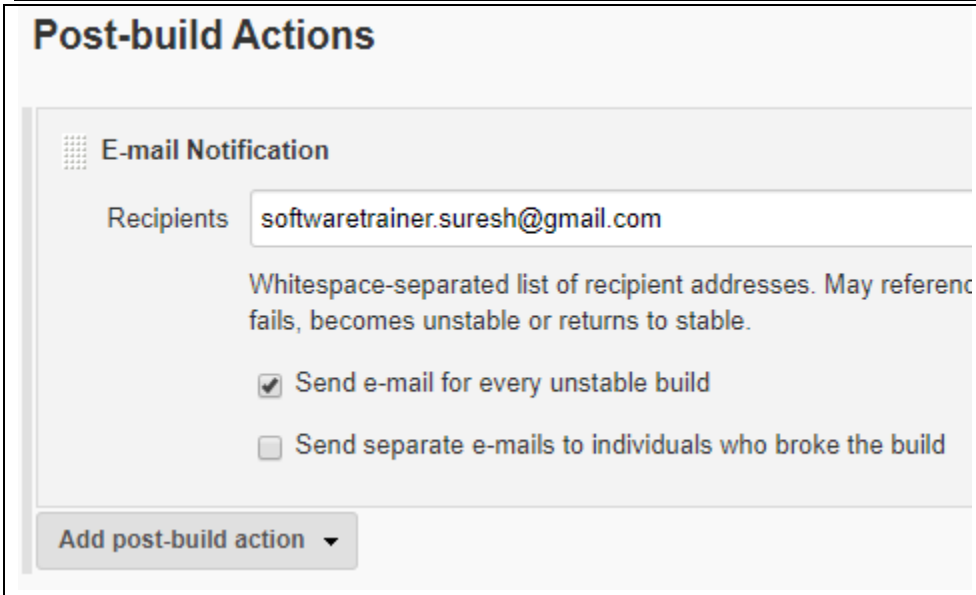
Provide the batch file name which you created in project workspace



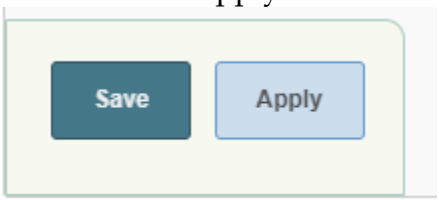
To run build automatically provide schedule details by clicking on Build periodically check box opion[This setting is optional]



Select E-mail notification option from Post-build Actions option(Note : selecting this option is Optional) ,Provide email id to whom jekins need to be send an e-mail notification of build status

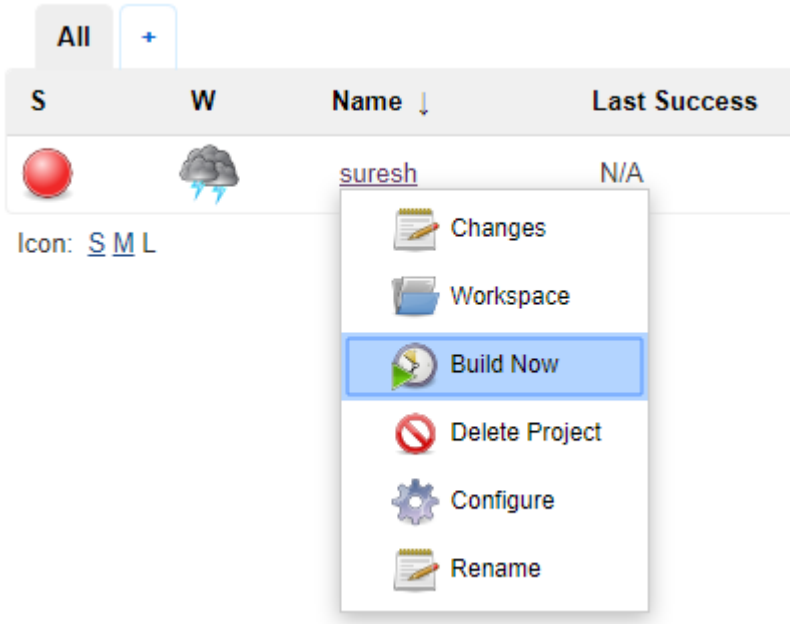


Now click on Apply button after that click on save button.

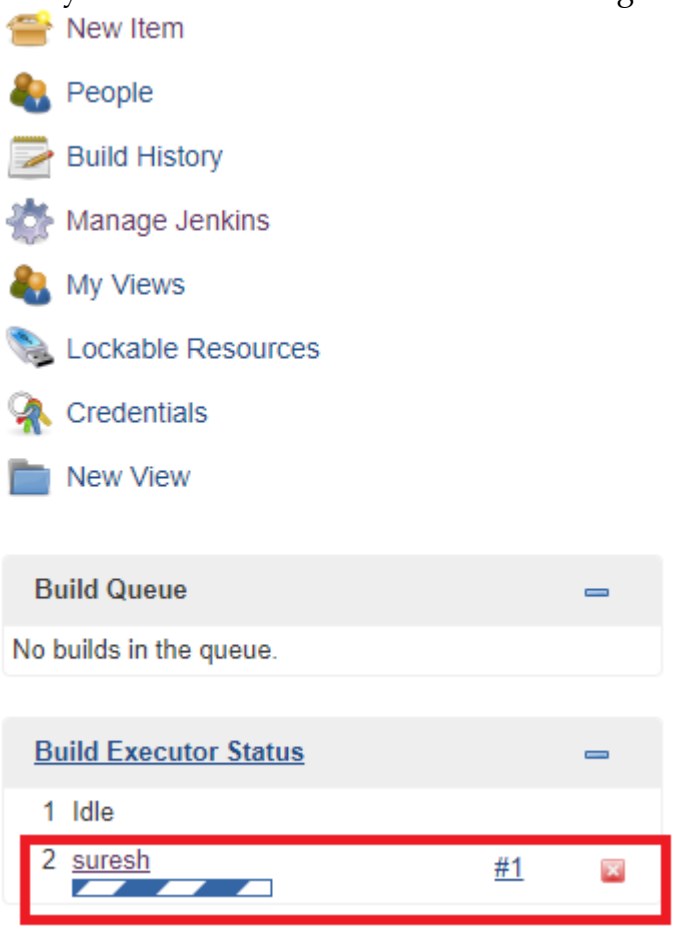


Then goto homepage - created job will be shown

To execute the script from Jenkins(manually) now click on created job and select BuildNow option by that it will execute the program will provide the results



Now you will be able to see build is executing and status shown as below



Click on build which u excuted to see the execution log information and then click on Console output option, then it will shows the execution log information

```
===============================================
    Default test
    Tests run: 1, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 1, Failures: 0, Skips: 0
===============================================
```

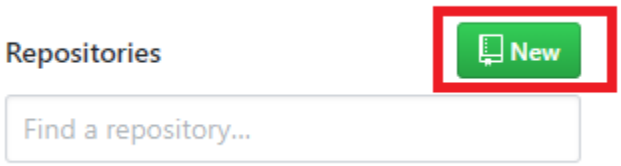
Git HUB(https://github.com/)

Github is a repository on web, which support all the feature of revision control and source code management

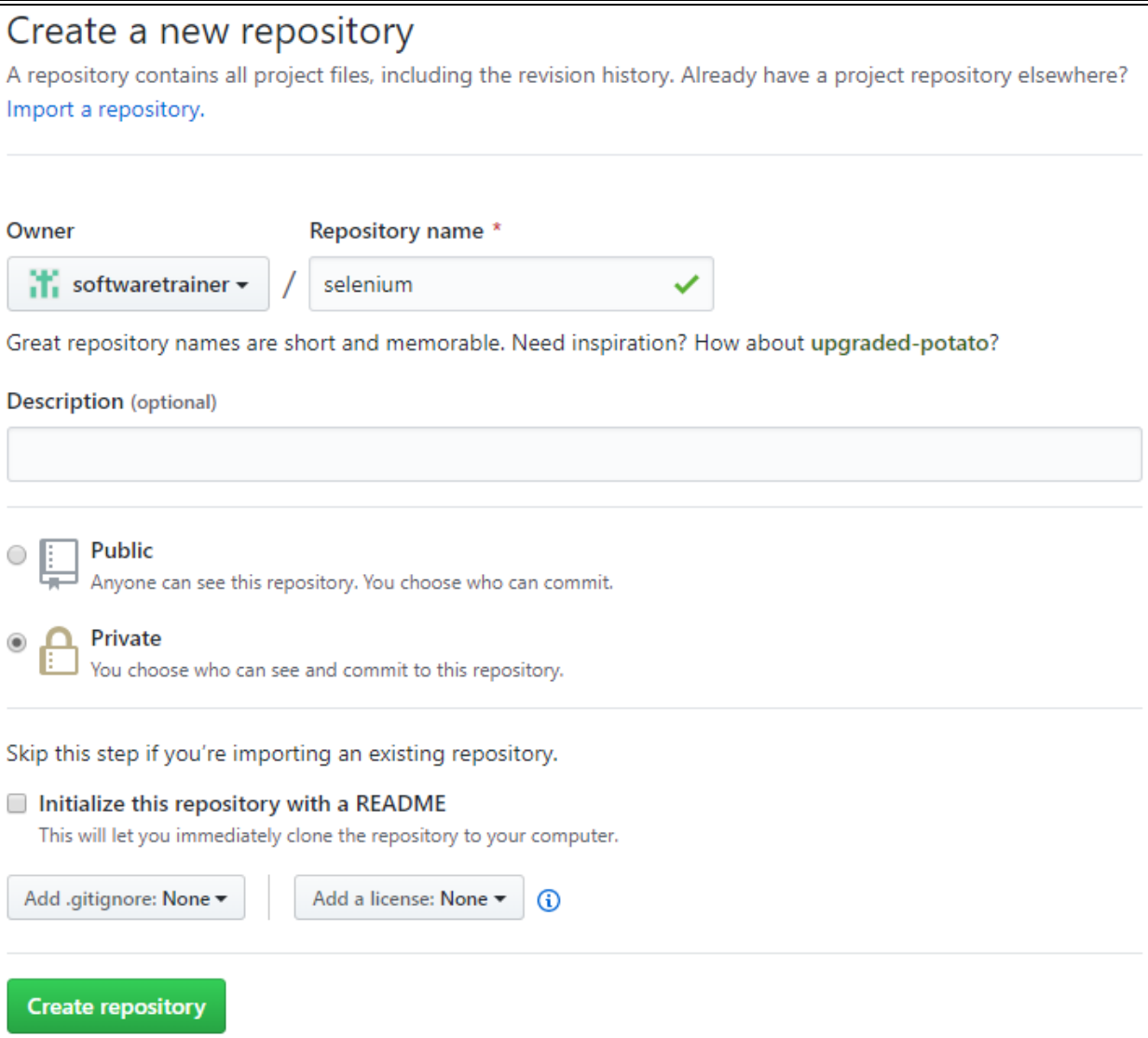
Steps to Integreate GitHub with Selenium Project in Eclipse:

Create an account in github with your valid email and other information.

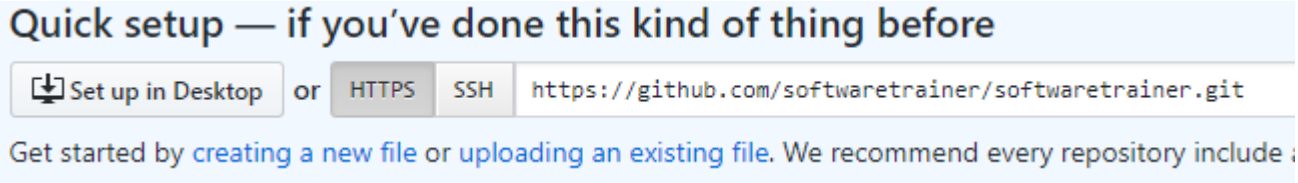
Login to github account and create new repository



Specify the name of the repository, description and click on create repository.

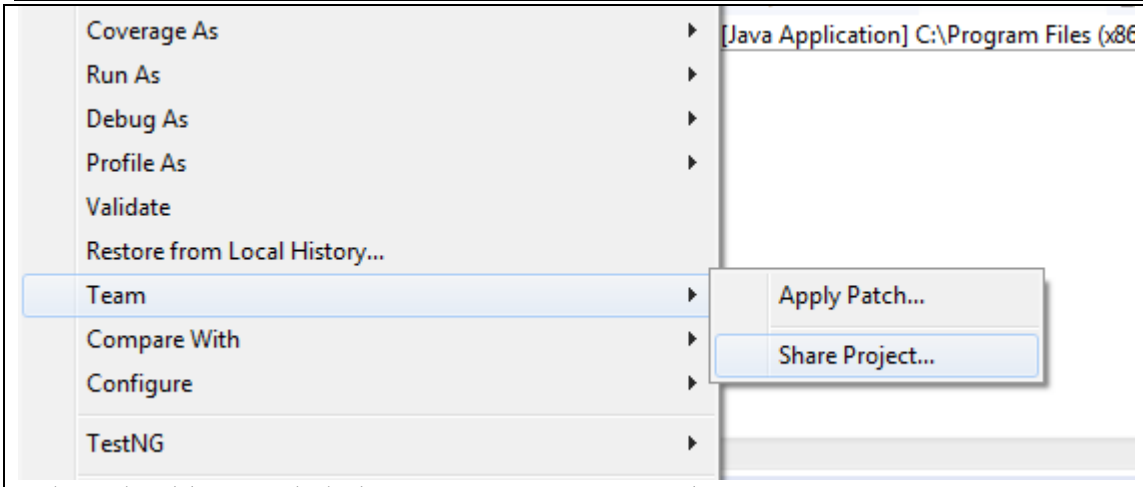


Copy new generated URL of your repository and save in textfile to use in futher steps

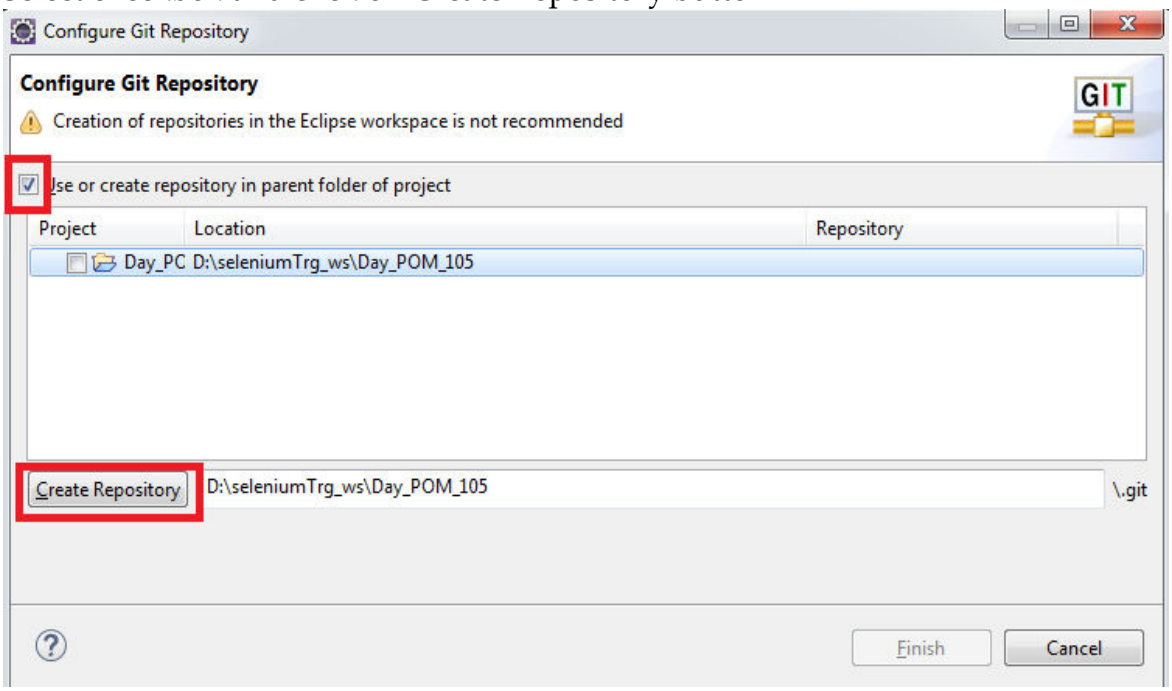


Note : Create github password token and Save the password in text file to use in further steps
https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token

Now open Eclipse and Select project which we want to upload on github.
Perform right click on project and Go to team section and Select share project.



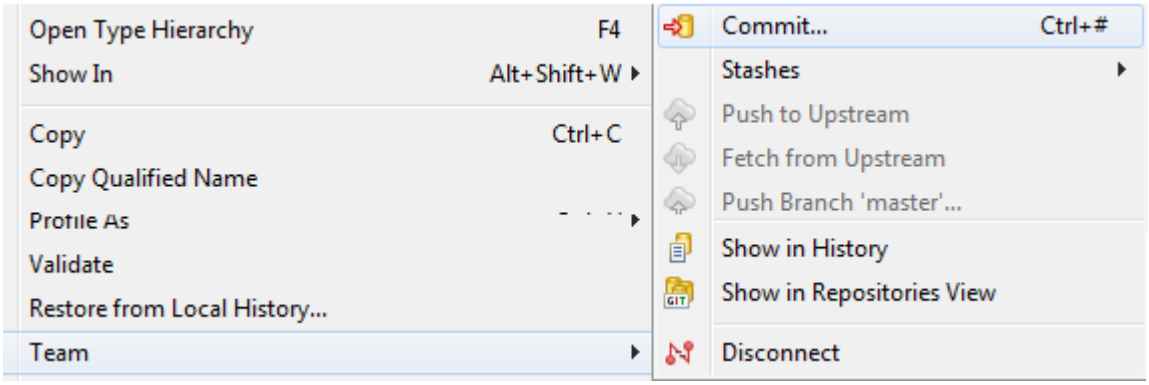
Select checkbox and click on Create Repository button



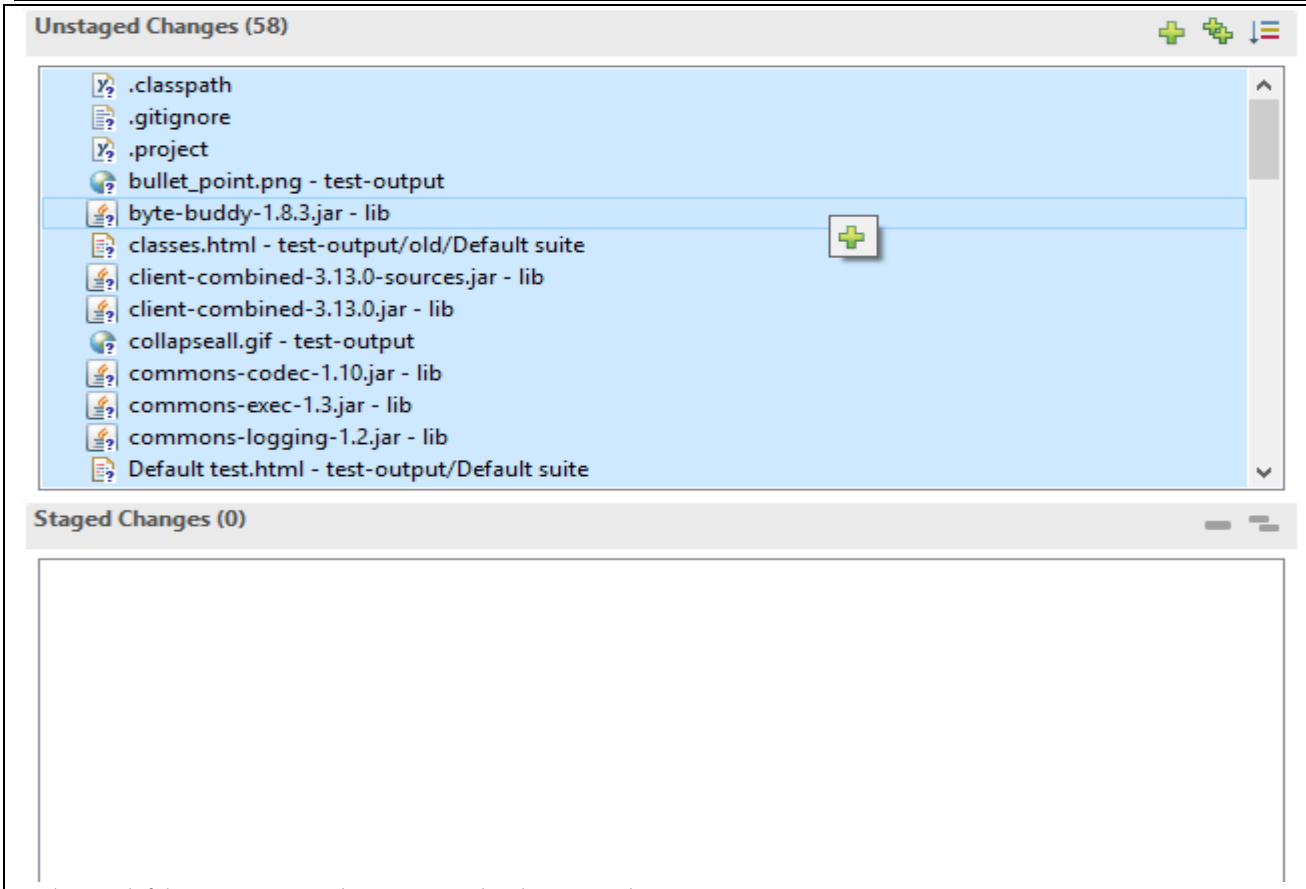
Click on Finish button

Perform right click on created project.

Navigate to Team option and click on Commit option

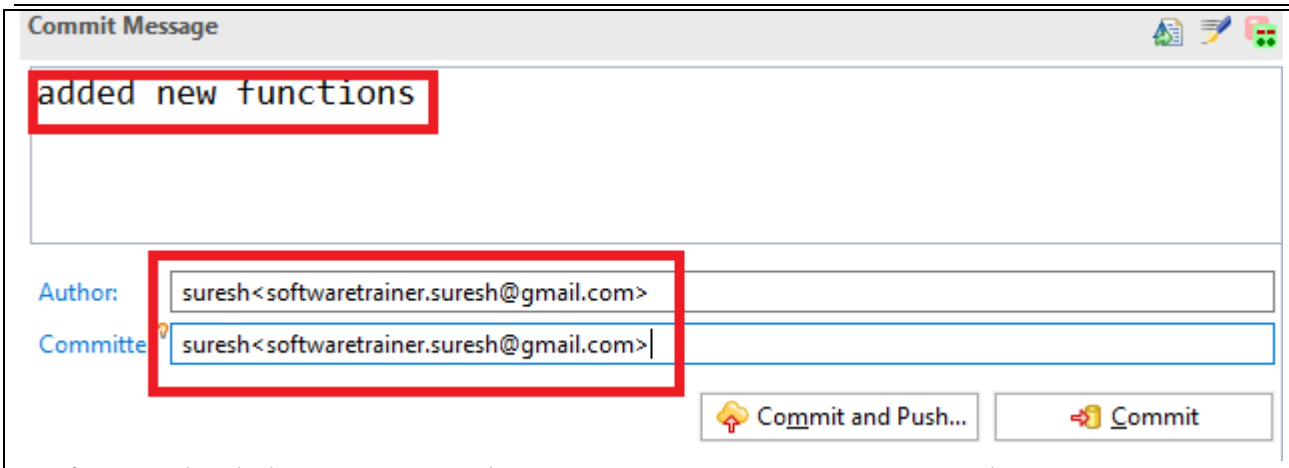


Select the required files to upload from Unstaged Changes box and click on + symbol, by that those files will be moved to Staged Chnages box.

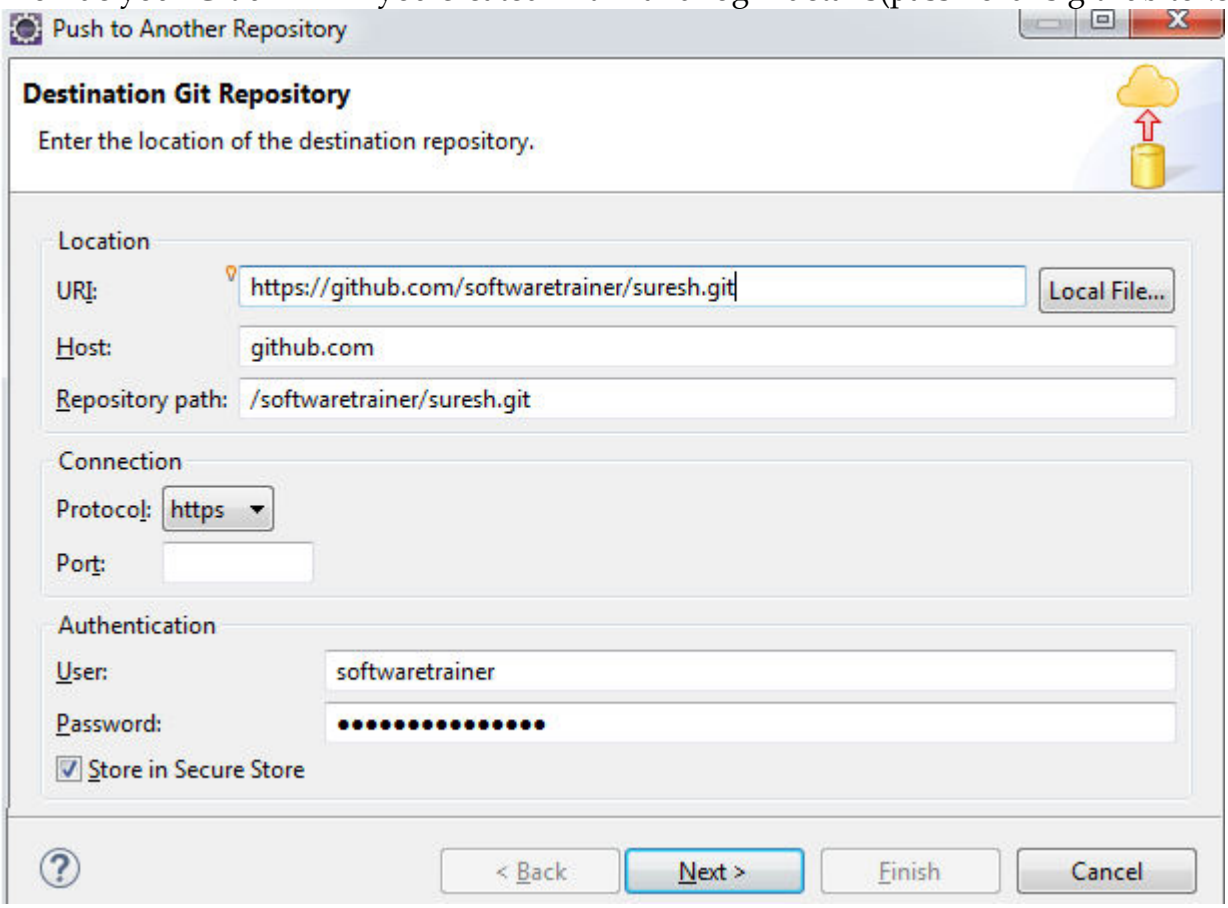


Selected file got moved to Staged Changes box

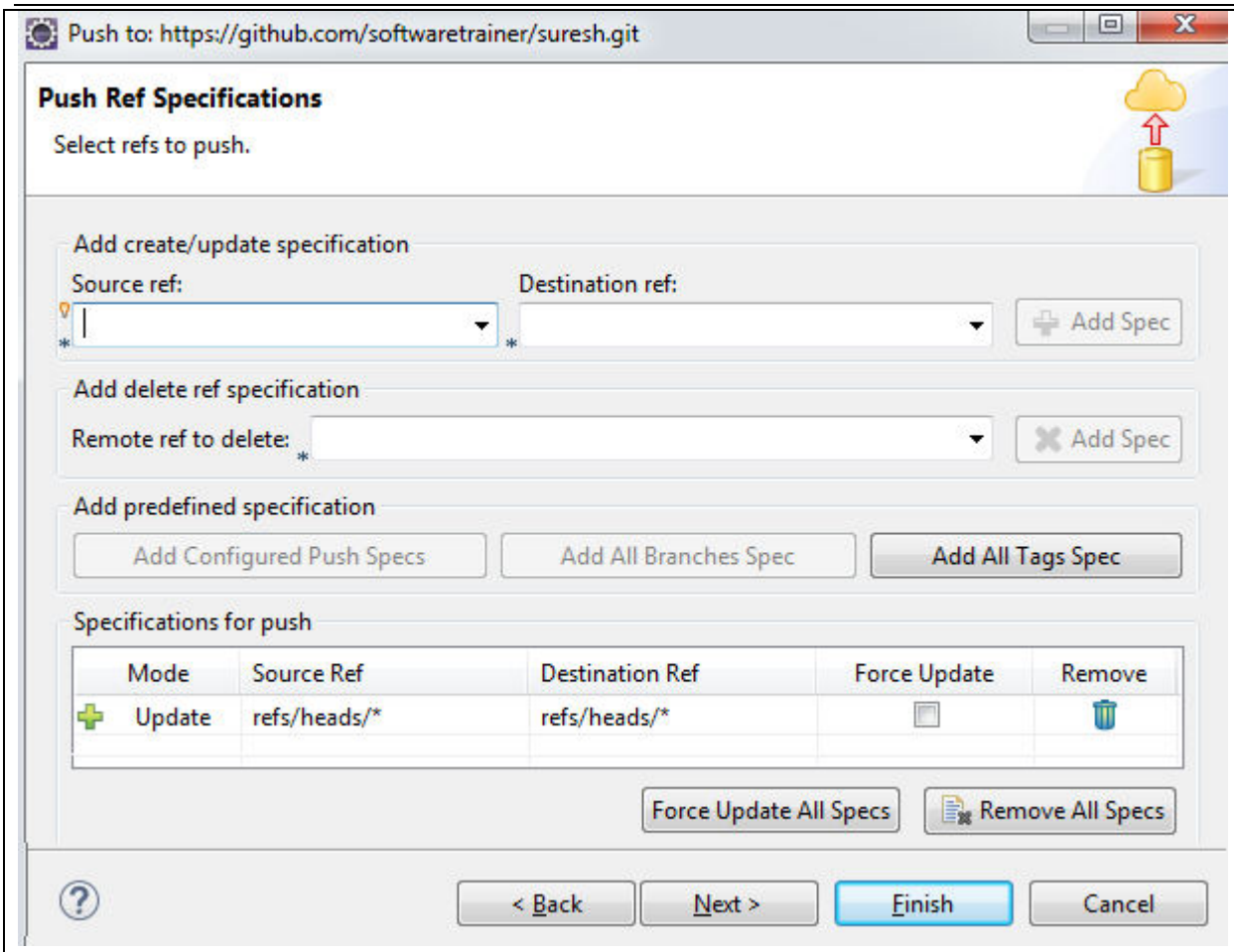
Provide commit message ,author and committer details and click on commit button



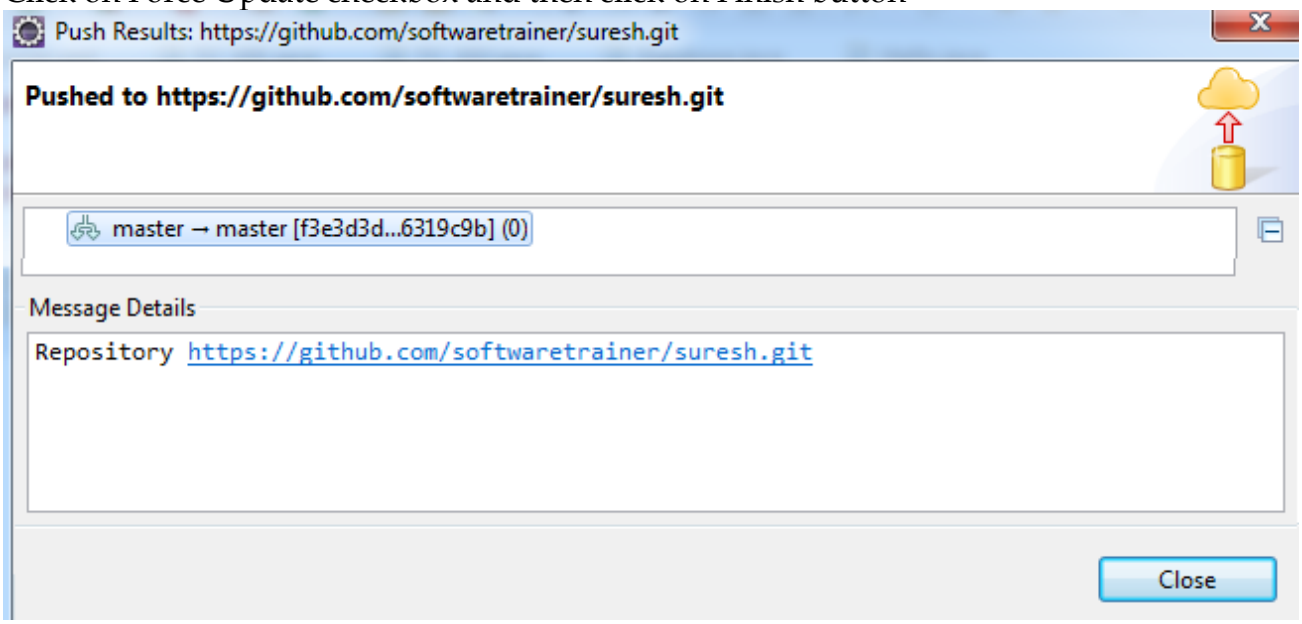
Perform right click on project and navigate to Team →Remote →Push
Provide your Git url whih you created with valid login details(password is github token)



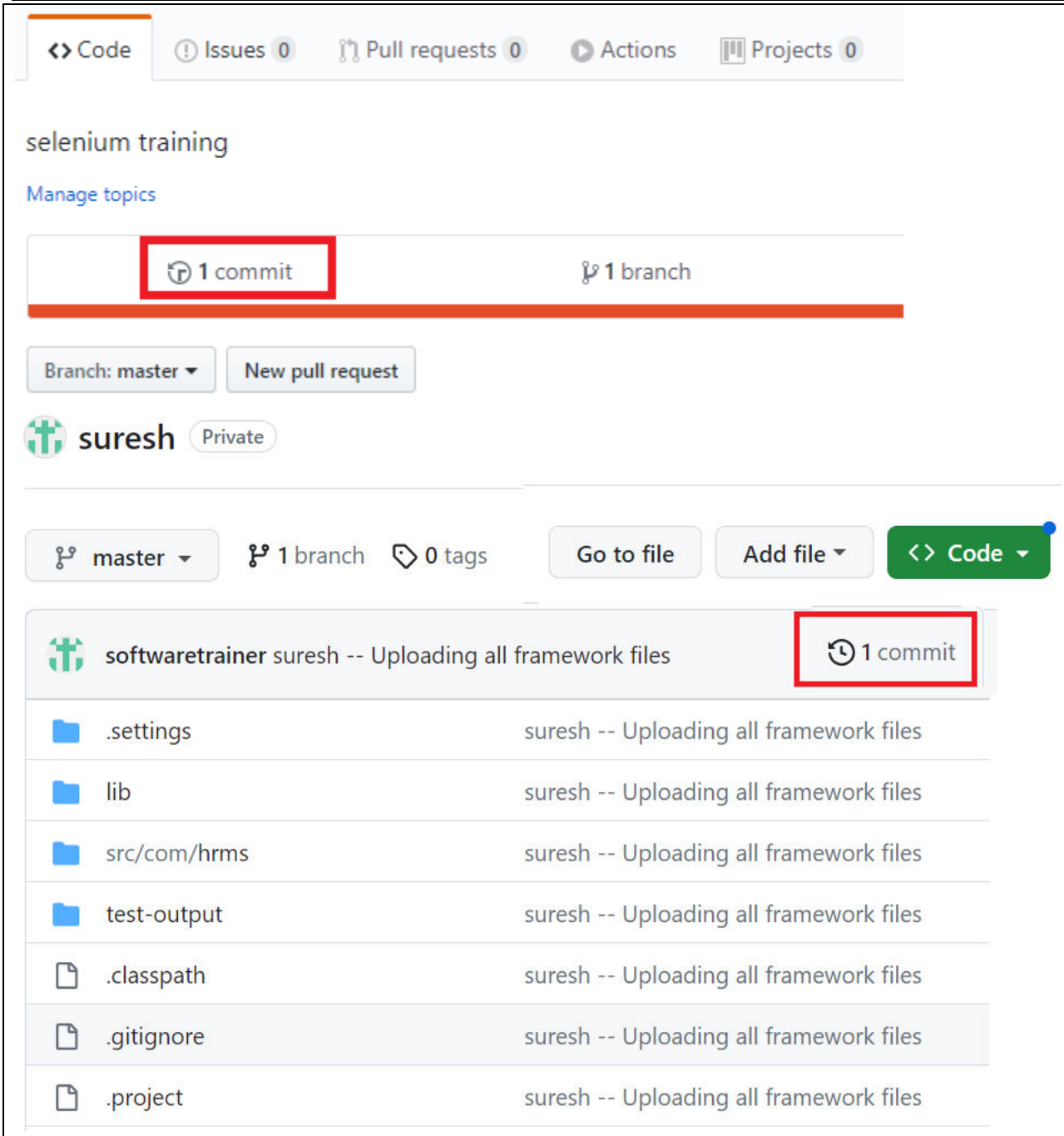
Click on Next button and then click on Add all branches Spec button



Click on Force Update checkbox and then click on Finish button



Once we are getting this window we completed to push the code to git reposity.
To confirm we can goto git repository and check either code is moved to repository or not



Note: if required we can download this code into another team member system and then we can make required modifications.

POM - PageObjectModel

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

What is POM?

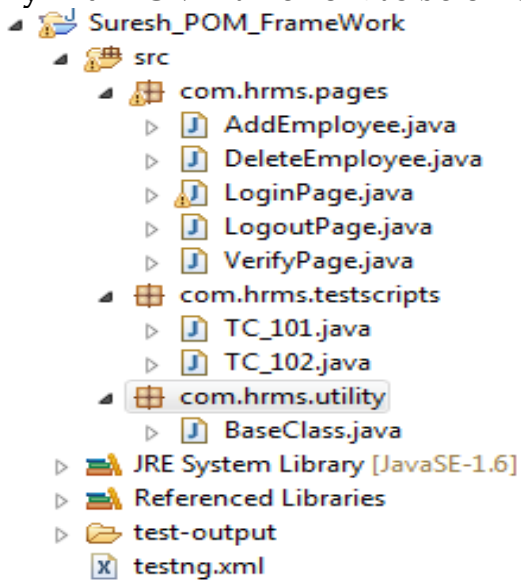
Page Object Model is a design pattern to create Object Repository for web UI elements.

Under this model, for each web page in the application there should be corresponding page class.
This Page class will find the WebElements of that web page and also contains Page methods which perform operations on those WebElements.
Name of these methods should be given as per the task they are performing

Advantages of POM :

Page Object Patten says operations and flows in the UI should be separated from verification. This concept makes our code cleaner and easy to understand.
Second benefit is the object repository is independent of testcases, so we can use the same object repository for a different purpose with different tools. For example, we can integrate POM with TestNG/JUnit for functional testing and at the same time with JBehave/Cucumber for acceptance testing.
Code becomes less and optimized because of the reusable page methods in the POM classes.
Methods get more realistic names which can be easily mapped with the operation happening in UI.

Try with POM framework as below structure –



Write the code as below :

BaseClass.java –

```
package com.hrms.utility;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.Reporter;
```

```
public class BaseClass {
public static WebDriver driver;
public static void openApplication() {
   driver = new FirefoxDriver();
   driver.navigate().to("https://sureshitacademy.in/login.php");
   Reporter.log("Application Opened");
}
public static void closeApplication() {
   driver.quit();
   Reporter.log("Application closed");
}
}
```

LoginPage.java--

```
package com.hrms.pages;

import org.openqa.selenium.By;
import org.testng.Reporter;

import com.hrms.utility.*;
public class LoginPage extends BaseClass{
//obj
   static By txt_loginname = By.name("txtUserName");
   static By txt_password = By.name("txtPassword");
   static By btn_login = By.name("Submit");
//fun
   public static void login(String un,String pw) throws Exception{
      driver.findElement(txt_loginname).sendKeys(un);
      driver.findElement(txt_password).sendKeys(pw);
      driver.findElement(btn_login).click();
      Thread.sleep(3000);
      Reporter.log("Login completed");
   }
}
```

LogoutPage.java –

```
package com.hrms.pages;

import org.openqa.selenium.By;
import org.testng.Reporter;

import com.hrms.utility.*;
public class LogoutPage extends BaseClass{
//obj
   static By link_logout = By.linkText("Logout");
//fun
```

```
    public static void logout() {
        driver.findElement(link_logout).click();
        Reporter.log("Logout completed");
    }
}
```

VerifyPage.java

```
package com.hrms.pages;
import org.testng.Reporter;

import com.hrms.utility.*;
public class VerifyPage extends BaseClass{
public static void verifyTitle(String title) {
    if(driver.getTitle().equals(title)) {
        Reporter.log("Title matched");
    }
    else {
        Reporter.log("Title not matched and expected title is " + driver.getTitle());
    }
}
}
```

TC_101.java –

```
package com.hrms.testscripts;
import org.testng.annotations.Test;
import com.hrms.pages.LoginPage;
import com.hrms.pages.LogoutPage;
import com.hrms.pages.VerifyPage;
import com.hrms.utility.BaseClass;

public class TC_101 {
//test case steps
    @Test
    public static void tc101() throws Exception{
        BaseClass.openApplication();
        VerifyPage.verifyTitle("HRMS");
        LoginPage.login("sureshit", "sureshit");
        VerifyPage.verifyTitle("OrangeHRM");
        LogoutPage.logout();
        BaseClass.closeApplication();
    }
}
```

Based on project requiremt we can continue implementing in remaining features too.

Apache-MAVEN

MAVEN INTRODUCTION

Apache Maven, an open source build framework that can be used to build projects and it provides developers a complete build life-cycle framework.

Originally Maven was designed to simplify building processes in Jakarta Turbine project.

The Main Objectives of Maven are:

•It follows the best practices and standards which helps new developers coming into a project

•It provides quality information of the project like test reports, dependency list etc.

•It provides a uniform build system with its project object Model.

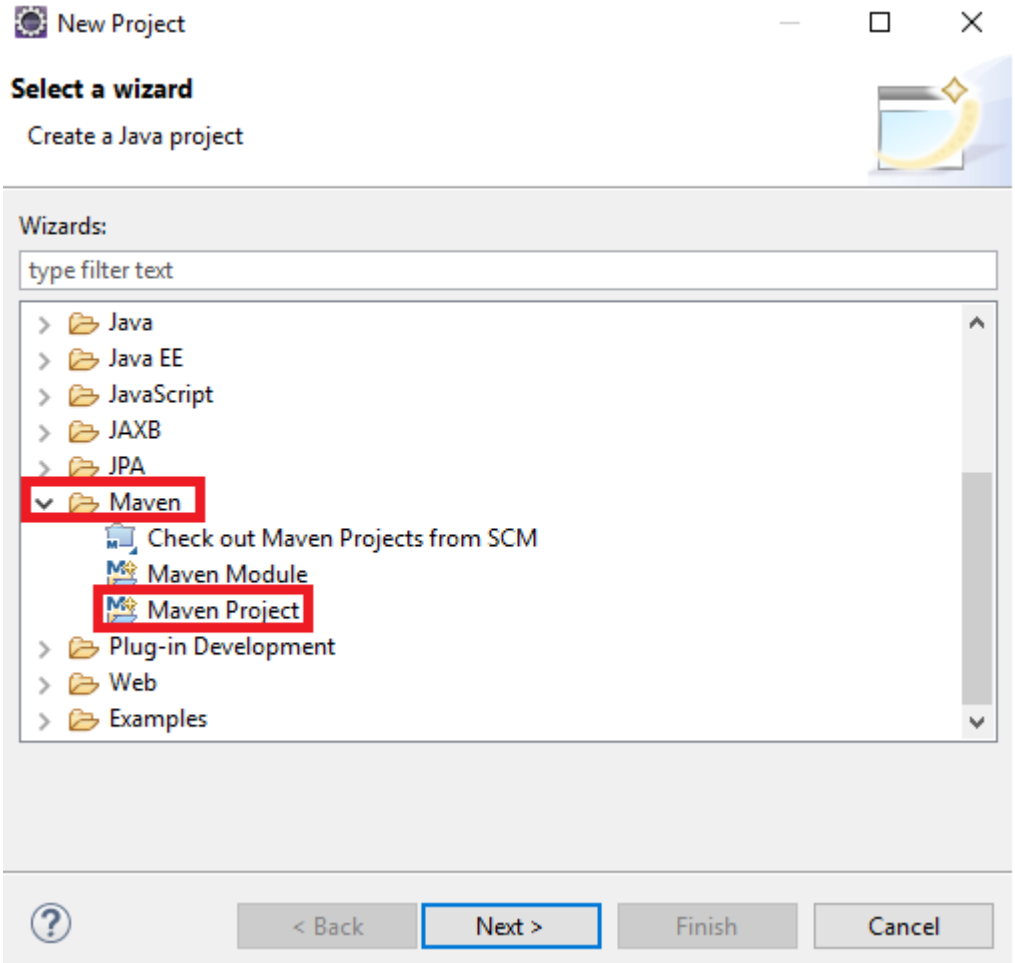
Maven is a project management tool, provides concept of a project object model (POM) file to manage project's build, dependency and documentation. The most powerful feature is able to download the project dependency libraries automatically

Let see how to configure eclipse with maven for selenium: (assuming eclipse is already there or you can go to http://www.eclipse.org/downloads/packages/release/juno/sr2)

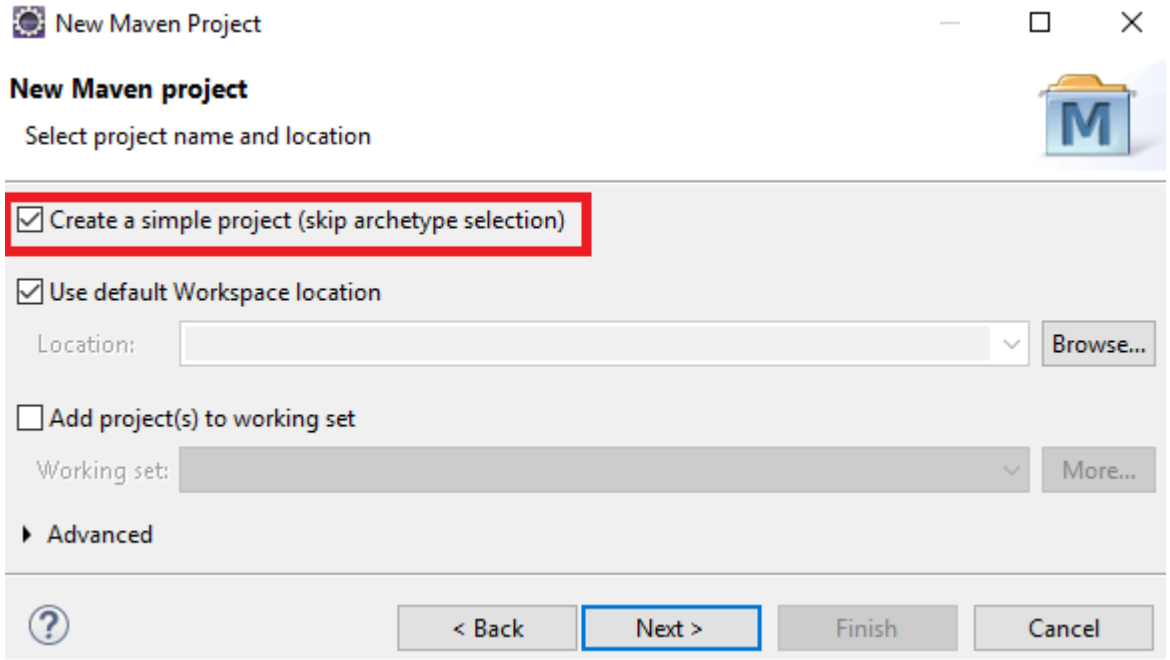
Launch the Eclipse IDE.

Create a new project by selecting File |New | Other from Eclipse Main Menu.

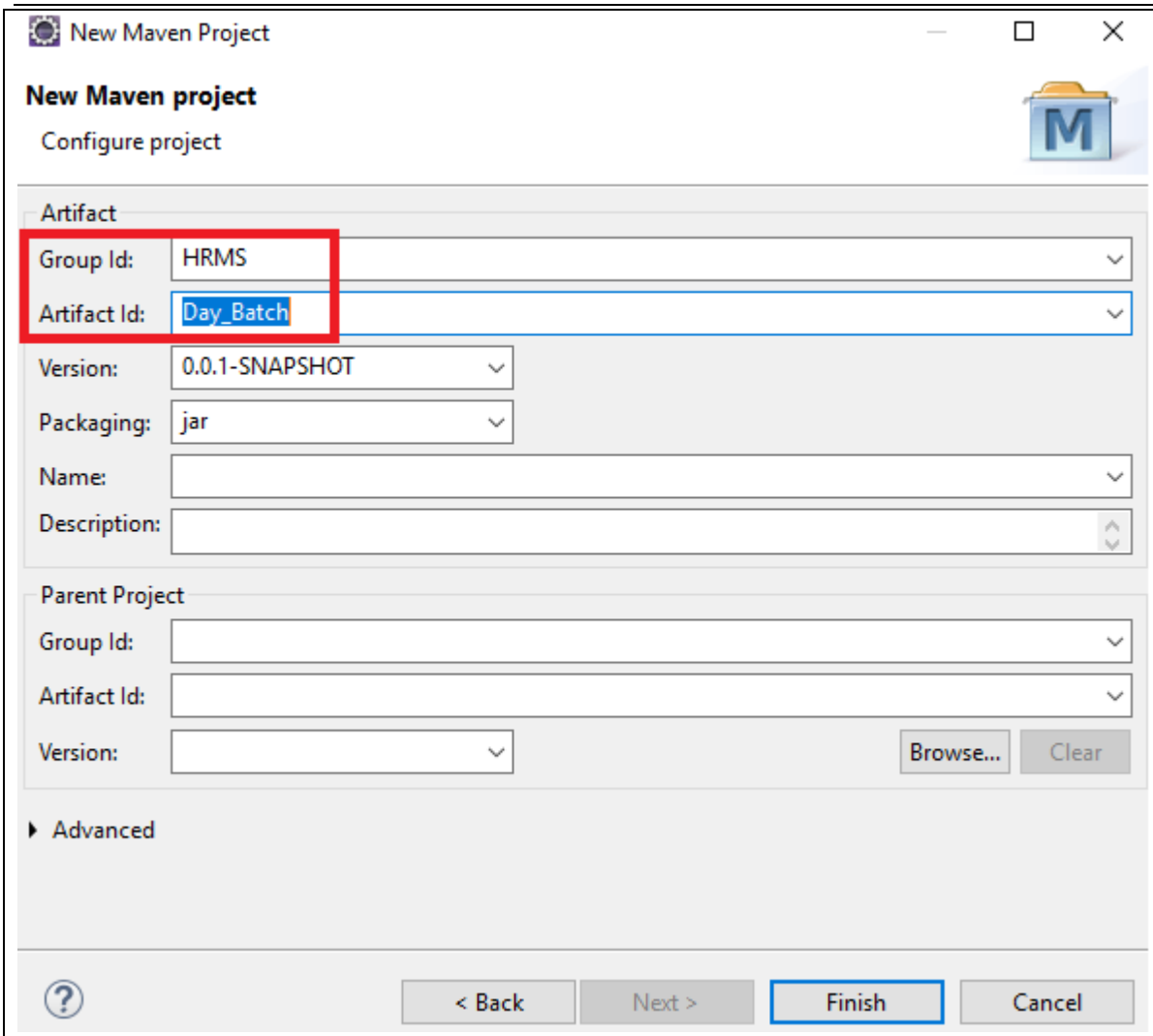
On the New dialog, select Maven |Maven Project as shown in the following screenshot:



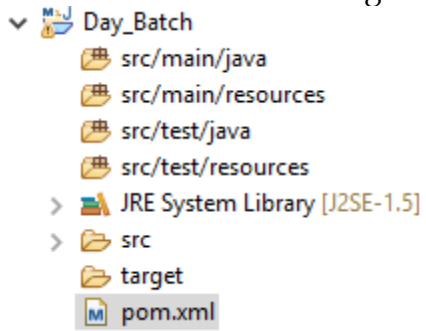
Next, the New Maven Project dialog will be displayed. Select the Create a simple project (skip archetype selection) check-box and set everything to default and click on the Next button as shown in the following screenshot:



On the New Maven Project dialog box, enter base package name (like com.myproject.app) in Group Id and project name (like myproject) in Artifact Id textboxes. You can also add a name and description. Set everything to default and click on the Finish button as shown in the following screenshot:



Eclipse will create the project with a structure (in Package Explorer) similar to the one shown in the following screenshot:

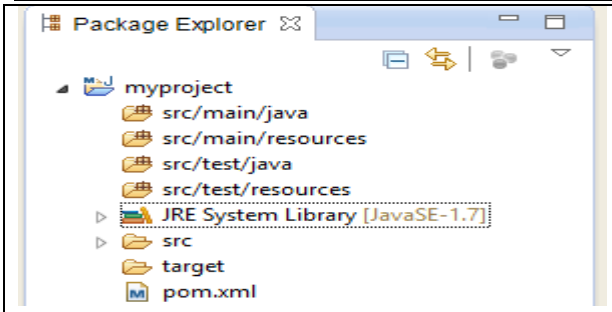


Right-click on JRE System Library [J2SE-1.5] and select the Properties option from the menu.

On the Properties for JRE System Library [J2SE-1.5] dialog box, make sure Workspace default JRE (jre7) is selected. If this option is not selected by default, select this option.

Note: The JRE version might change based on the Java version installed on your machine.

Click on the OK button to save the change as shown in the following screenshot:



Select pom.xml from Package Explorer. This will open the pom.xml file in the editor area with the Overview tab open. Select the pom.xml tab instead.
Add the WebDriver and testng dependencies highlighted in the following code snippet, to pom.xml in the <project> node:

```
---------------------------------------------------------------------
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>
<groupId>com.myproject.app</groupId>
<artifactId>myproject</artifactId>
<version>0.0.1-SNAPSHOT</version>
<dependencies>
  <dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>4.10.0</version>
  </dependency>
<dependency>
 <groupId>org.testng</groupId>
 <artifactId>testng</artifactId>
 <version>6.8.0</version>
 <scope>test</scope>
</dependency>
</dependencies>
</project>
```

Cucumber

What are the benefits?

1. It is helpful to involve business stakeholders who can't easily read code
2. Cucumber focuses on end-user experience
3. Style of writing tests allow for easier reuse of code in the tests
4. Quick and easy set up and execution
5. Efficient tool for testing

Introduction

Cucumber introduces the notion of "features" which describe the behavior you wish to test. The Feature is then broken down into a number of different "scenarios" which comprise the test you wish to execute which will subsequently validate the

feature. Each scenario is further broken down into a number of "steps" which describe the execution path of each scenario. Typically, these follow a strict "given-when-then" format which aids consistency and provides a clear Template for writing acceptance tests.

Testing with Cucumber

Cucumber is a testing framework that helps to bridge the gap between software developers and business managers. Tests are written in plain language based on the behavior-driven development (BDD) style of Given, When, Then, which any layperson can understand. Test cases are then placed into feature files that cover one or more test scenarios. Cucumber interprets the tests into the specified programming language and uses Selenium to drive the test cases in a browser. Our tests are translated into Java code.

The Given, When, Then syntax is designed to be intuitive. Consider the syntax elements:

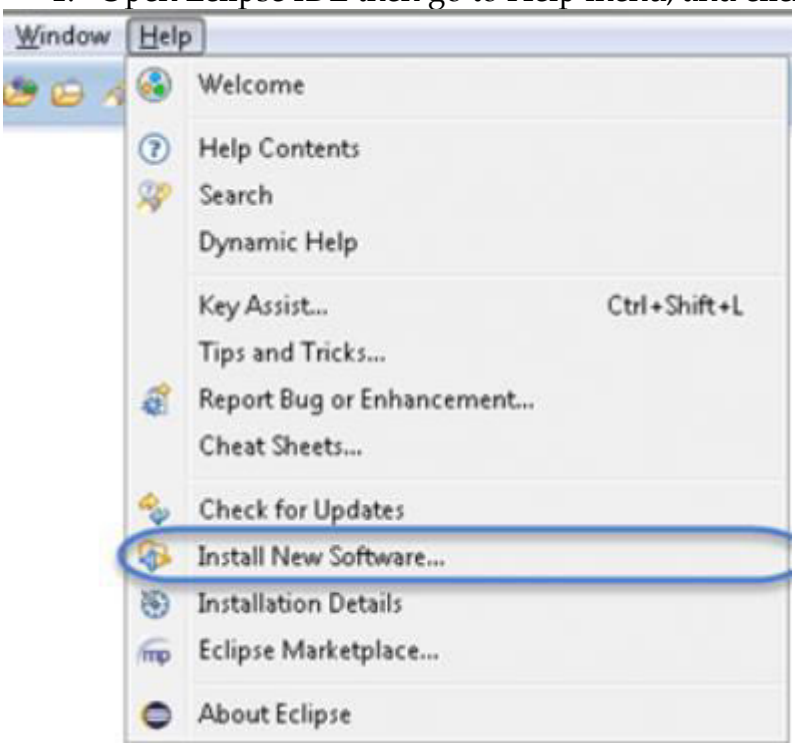
Given provides context for the test scenario about to be executed, such as the point in your application that the test occurs as well as any prerequisite data.

When specifies the set of actions that triggers the test, such as user or subsystem actions.

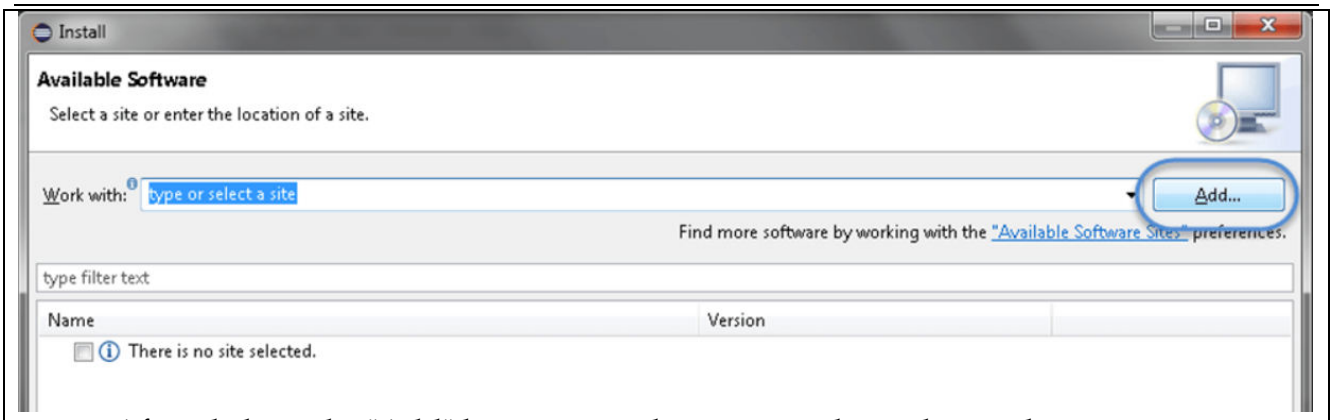
Then specifies the expected result of the test.

Steps to install Cucumber plugin installation in Eclipse :

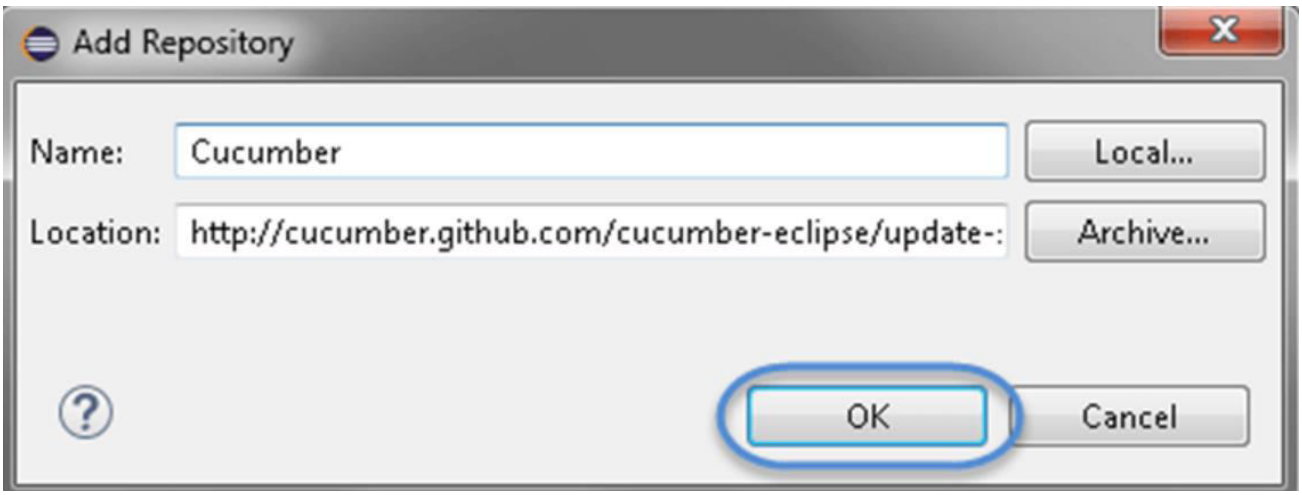
1. Open Eclipse IDE then go to Help menu, and click "Install New Software".



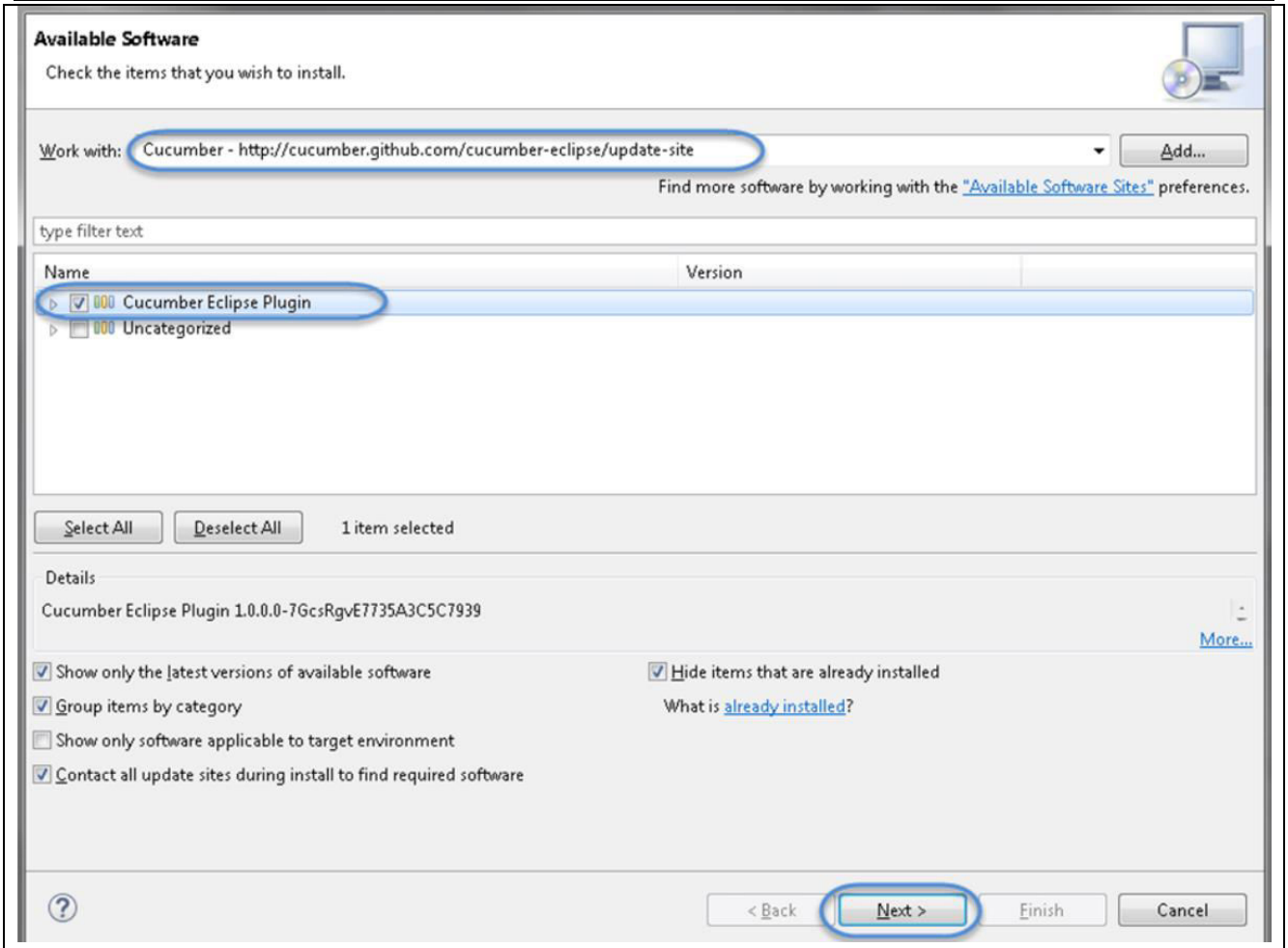
2. After clicking "Install New Software", a window will be prompted, on this window, click the "Add" button.



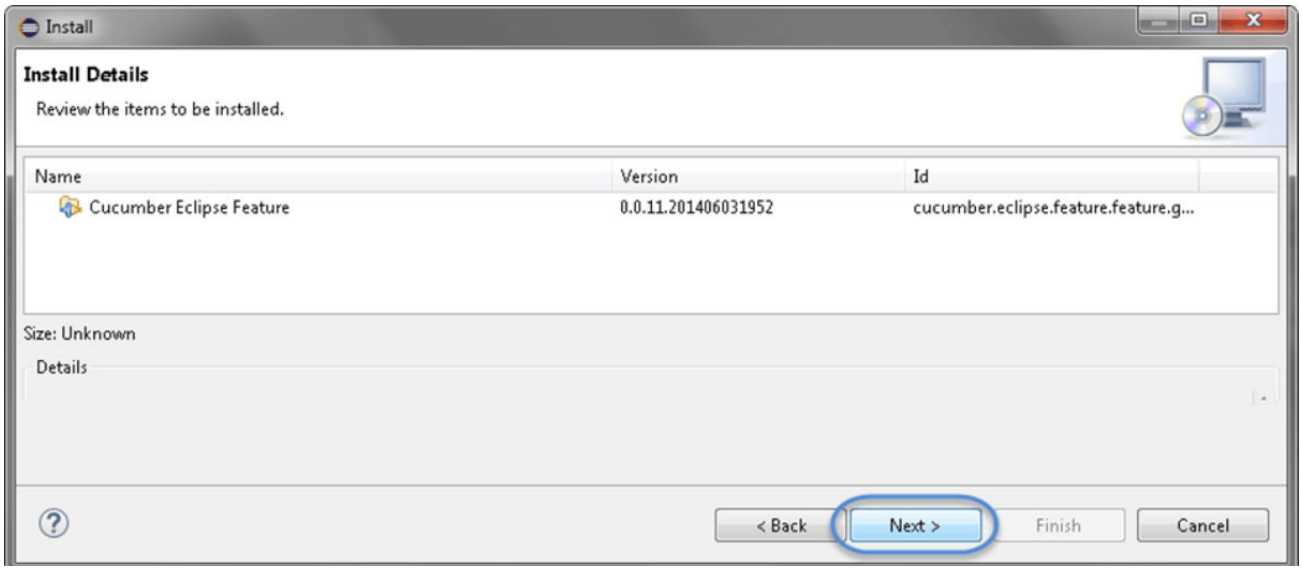
3. After clicking the "Add" button, give the Name in the in the text box as per your choice. We provided "Cucumber".
   Now, in the Location text box type "**http://cucumber.github.com/cucumber-eclipse/update-site**" [OR] "**http://cucumber.github.io/cucumber-eclipse/update-site**"as location and then click OK.



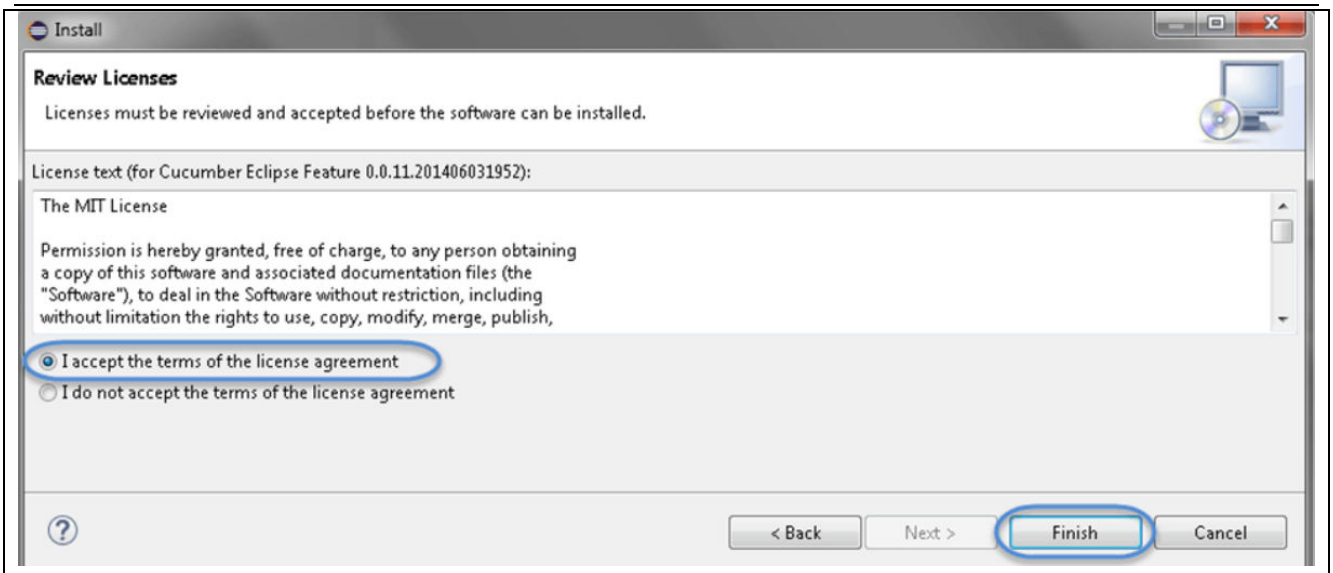
4. Now, you will come back on the previous window, but this time you will see "Cucumber Eclipse Plugin" in the software list. Just click "Check Box" and then the "Next" button.



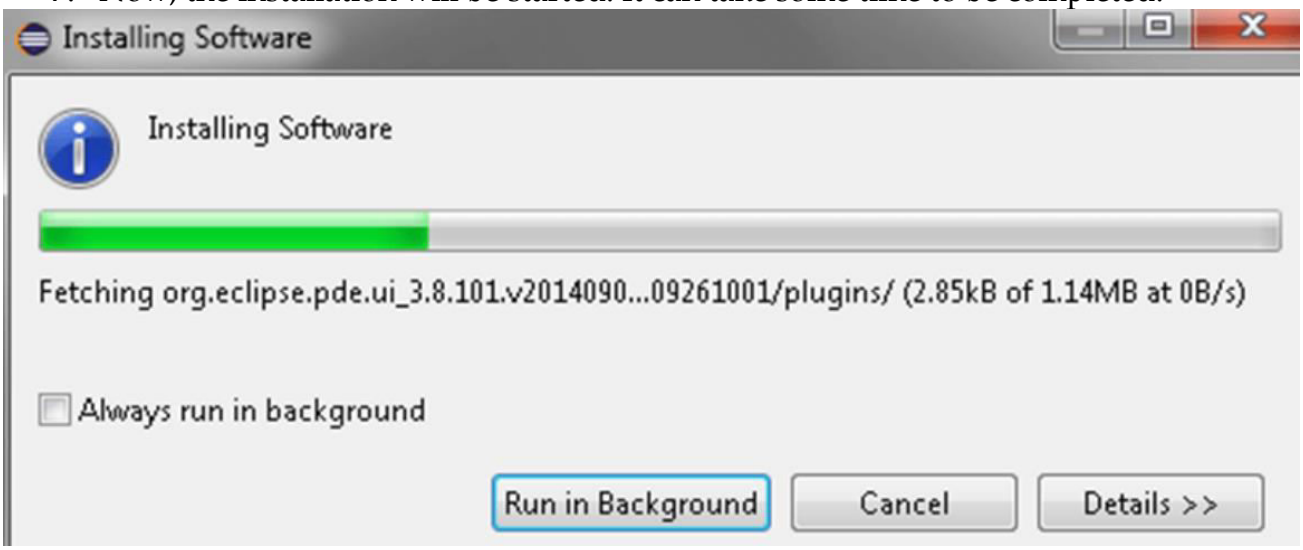
5. Now, click on the "Next" button.



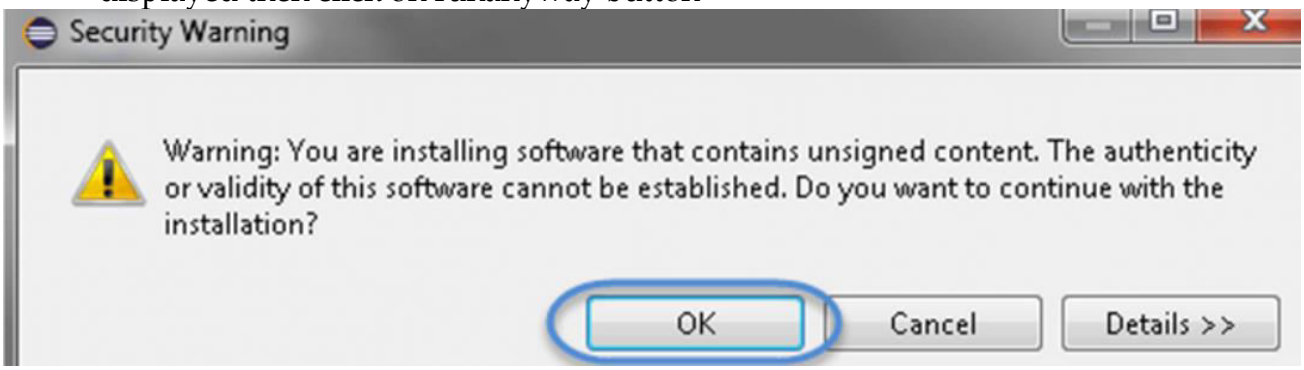
6. Click the check box "I accept the terms of the license agreement" on the license window then click Finish.



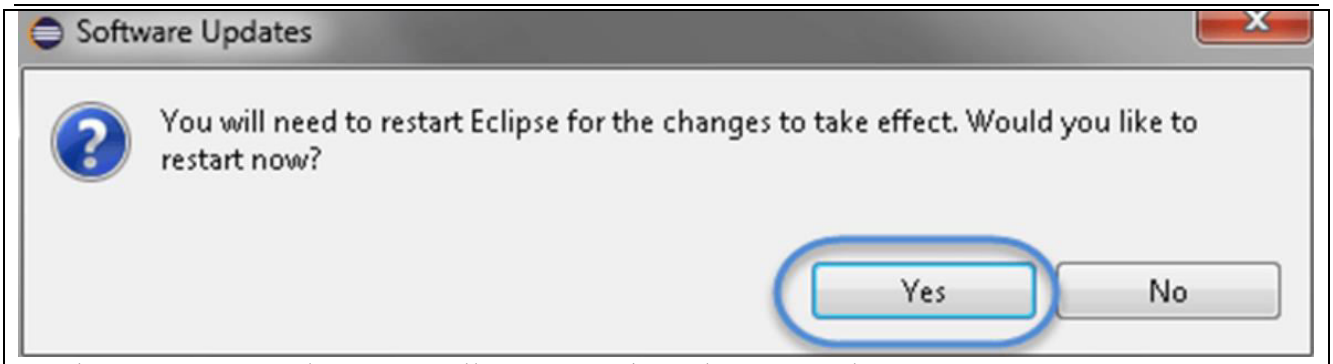
7. Now, the installation will be started. It can take some time to be completed.



8. If you encounter a Security warning, just click OK. Or In case of runanyway button displayed then click on runanyway button



9. The installation has been completed, now just click "Yes" button.



By this we are completing installing cucumber plugin in eclipse.

Steps to implement Selenium with Cucumber

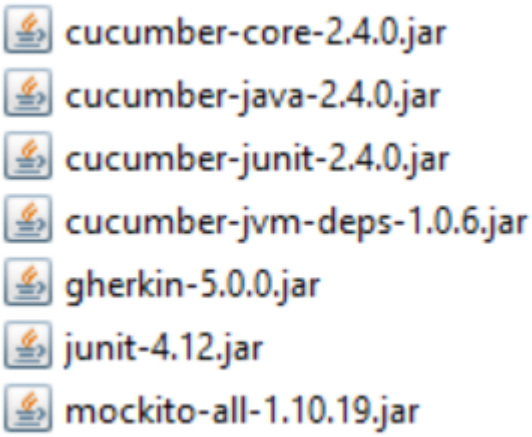
1. Getting started - Open Eclipse and create a new Maven project

2. Add Maven dependencies

In your pom.xml, add the following dependencies:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>hrms</groupId>
  <artifactId>Cuc_hrms</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <dependencies>
  <dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-java</artifactId>
        <version>4.11.0</version>
    </dependency>

<dependency>
    <groupId>info.cukes</groupId>
    <artifactId>cucumber-java</artifactId>
    <version>1.2.5</version>
    <scope>compile</scope>
</dependency>

<dependency>
    <groupId>info.cukes</groupId>
    <artifactId>cucumber-junit</artifactId>
    <version>1.2.5</version>
</dependency>

 </dependencies>
</project>
```

Note : In case of JAVA Project download below jarfiles & Add to project & Make sure that core, java and junit files all need to be the same file version



3. Create a feature file as "hrms.feature" . This file contain below scenario.

```
@Application
Feature: Verify Title

Scenario: Verify Title
Given Open Application
When  Verify Title
Then  Close Application
```

4. Now need to create step definition file for above scenario and implement webdriver automation code. So Create a java file "TC001.java" and write below code:

```
package com.hrms;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import cucumber.api.java.en.Given;
import cucumber.api.java.en.Then;
import cucumber.api.java.en.When;

public class TC001 {
        public WebDriver driver;
        @Given("^Open Application$")
        public void openbrowser(){
                driver = new FirefoxDriver();
                driver.navigate().to("https://sureshitacademy.in/login.php");
    System.out.println("Application opened");
        }
        @When("^Verify Title$")
        public void verifyTitle()    {
                System.out.println("Verifying the Title");
        }

        @Then("^Close Application$")
        public void closebrowser() {
                driver.quit();
```

```
                System.out.println("Login page should be shown");
        }
}
```

5.Now create "TestRun.java" which defines cucumber-jvm configuration and write below code

```
package com.hrms;

import org.junit.runner.RunWith;
import cucumber.api.CucumberOptions;
import cucumber.api.junit.Cucumber;

@RunWith(Cucumber.class)
@CucumberOptions(format={"pretty","html:target/cucumber-html-
report"},tags={"@Application"})

public class TestRun {


}
```

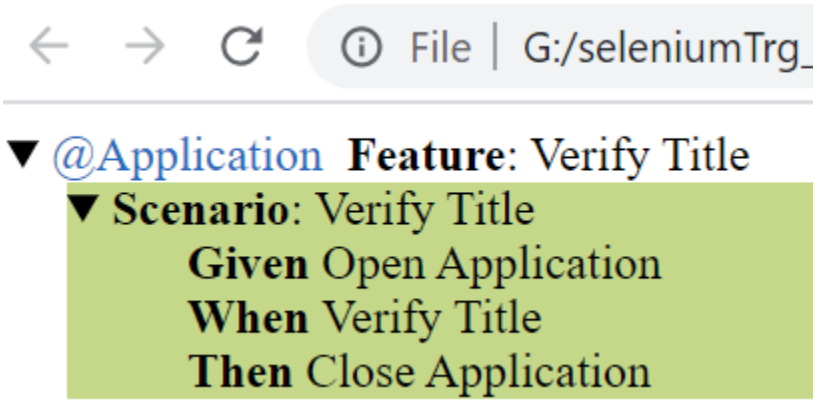
6. **Now you can exeute the program by using two option one is feature file and another one is TestRun**

Rightclick on hrms.feature file and Select Run as Cucumbe feature. **Or** Right click on **TestRun** file and Select Run as JunitTest

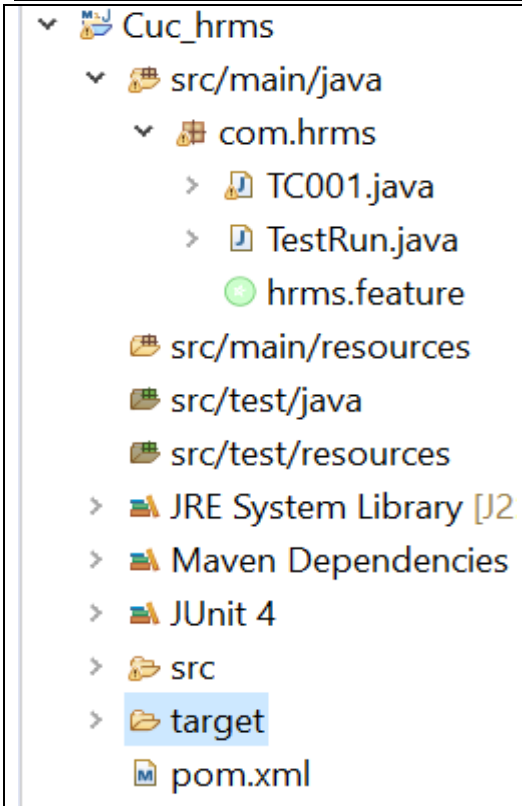
Results log need to shown as below

```
  Scenario: Verify Title   [90m# com/hrms/hrms.feature:4[0m
    [32mGiven [0m[32mOpen Application[0m [90m# TC001.openbrowser()[0m
    [32mWhen [0m[32mVerify Title[0m      [90m# TC001.verifyTitle()[0m
    [32mThen [0m[32mClose Application[0m [90m# TC001.closebrowser()[0m

1 Scenarios ([32m1 passed[0m)
3 Steps ([32m3 passed[0m)
0m23.103s
```

Html Results shown in the target folder as below as below.



Project structure shown as below.



Steps to execute Group of Scenarios using Cucumber

1. Getting started - Open Eclipse and create a new Maven project

2. Add Maven dependencies

In your pom.xml, add the following dependencies:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>hrms</groupId>
  <artifactId>Cuc_hrms</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <dependencies>
  <dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-java</artifactId>
        <version>4.11.0</version>
    </dependency>
<dependency>
    <groupId>info.cukes</groupId>
    <artifactId>cucumber-java</artifactId>
    <version>1.2.5</version>
    <scope>compile</scope>
</dependency>
<dependency>
    <groupId>info.cukes</groupId>
    <artifactId>cucumber-junit</artifactId>
    <version>1.2.5</version>
</dependency>
 </dependencies>
</project>
```

3. Create a feature file as "hrms.feature" and provide scenarios details as below.

```
@Application
Feature: Verify Title

Scenario: Verify Title
Given Open Application
When  Verify Title
Then  Close Application

@LoginLogout
Scenario: LoginandLogout
Given Open Application
When  Type username and password and click on Login button
When  Click on Logout
Then  Close Application
```

4. Now need to create step definition file for above scenarios and implement webdriver automation code. So Create a java file "TC001.java" and write below code:

```
package com.hrms;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import cucumber.api.java.en.Given;
import cucumber.api.java.en.Then;
import cucumber.api.java.en.When;

public class TC001 {
        public WebDriver driver;
        @Given("^Open Application$")
        public void openbrowser(){
                driver = new FirefoxDriver();
                driver.navigate().to("https://sureshitacademy.in/login.php");
      System.out.println("Application opened");
        }
        @When("^Verify Title$")
        public void verifyTitle()     {
                System.out.println("Verifying the Title");
        }
        @When("^Type username and password and click on Login button$")
        public void login()   {
                driver.findElement(By.name("txtUserName")).sendKeys("sureshit");
                driver.findElement(By.name("txtPassword")).sendKeys("sureshit");
                driver.findElement(By.name("Submit")).click();
                System.out.println("Login completed");
        }
        @When("^Click on Logout$")
        public void logout() {
                driver.findElement(By.linkText("Logout")).click();
                System.out.println("Logout Completed");
        }
```

```
	@Then("^Close Application$")
	public void closebrowser() {
		driver.quit();
		System.out.println("Login page should be shown");
	}
}
```

5.Now create "TestRun.java" which defines cucumber-jvm configuration and write below code

```
package com.hrms;
import org.junit.runner.RunWith;
import cucumber.api.CucumberOptions;
import cucumber.api.junit.Cucumber;

@RunWith(Cucumber.class)
@CucumberOptions(format={"pretty","html:target/cucumber-html-
report"},tags={"@Application,@LoginLogout"})

public class TestRun {


}
```

6. Now you can exeute the program by using TestRun file

Rightclick on hrms.feature file and Select Run as Cucumbe feature. **Or** Right click on **TestRun** file and Select Run as JunitTest

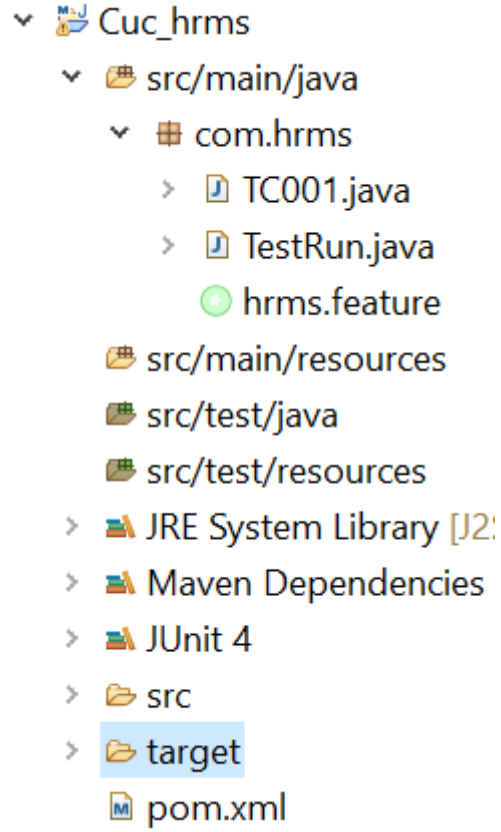
Results log need to shown as below

```
 Scenario: Verify Title   [90m# com/hrms/hrms.feature:4[0m
   [32mGiven [0m[32mOpen Application[0m [90m# TC001.openbrowser()[0m
   [32mWhen [0m[32mVerify Title[0m      [90m# TC001.verifyTitle()[0m
   [32mThen [0m[32mClose Application[0m [90m# TC001.closebrowser()[0m
Login page should be shown

  @LoginLogout
  Scenario: LoginandLogout                                  [90m# com/hrms/hrms.feature:10[0m
    [32mGiven [0m[32mOpen Application[0m                                    [90m# TC001.openbrowser()[0m
    [32mWhen [0m[32mType username and password and click on Login button[0m [90m# TC001.login()[0m
    [32mWhen [0m[32mClick on Logout[0m                                      [90m# TC001.logout()[0m
    [32mThen [0m[32mClose Application[0m                                    [90m# TC001.closebrowser()[0m

2 Scenarios ([32m2 passed[0m)
7 Steps ([32m7 passed[0m)
0m23.571s
```

Html Results shown in the target folder as below as below.

```
▼ @Application  Feature: Verify Title
  ▼ Scenario: Verify Title
        Given Open Application
        When Verify Title
        Then Close Application
  ▼ @LoginLogout  Scenario: LoginandLogout
        Given Open Application
        When Type username and password and click on Login button
        When Click on Logout
        Then Close Application
```

-In Eclipse Project Structure shown as below.



Sikuli Automation Tool

Introduction

Sikuli is a tool to automate graphical user interfaces (GUI) using "Visual Image Match" method. In Sikuli, all the web elements should be taken as an image and stored inside the project. Sikuli will trigger GUI interactions based on the image visual match, the image which we have passed as the parameter along with all methods.

Sikuli can be very much useful to automate flash objects (which do not have ID or name). It can be useful in the situation, where we have a stable GUI (i.e. GUI components not changing).

Even Window based applications can also be automated using Sikuli. Sikuli provides very friendly Sikuli-script.jar, which can be easily used together with Selenium WebDriver. We can even automate Adobe Video/Audio player, Flash Games on website using Sikuli. With simple API, it makes coding easier.
Practical Uses

Sikuli can be used to automate Flash Objects / Flash Websites.
It can be useful to automate Window based application. We can automate what we are seeing on the screen.
It provides, simple API. i.e. all methods can be accessed using screen class object.
It can be easily integrated with Selenium and all other tools.
Using Sikuli we can automate desktop applications.
Most of the automation testing tools will not support flash object automation (E.g. Selenium). Sikuli provides extensive support to automate flash objects.
It uses powerful "Visual Match" mechanism to automate desktop & flash objects.
Benefits

Open source Tool.
One of the biggest advantages of Sikuli is that, it can easily automate Flash objects.
It makes easy to automate windows application.
When you're testing an application under development and you don't know the ID/name of the elements, then you can go with Sikuli. It will check the appearance of the image and if match found, it will interact with the image accordingly.

Prerequisites:
Before getting started, we need to download and install the following software:
Any screenshot capturing tool (E.g. DuckCapture, or qSnap or Snipping Tool)

DownLoad and Add Sikuli Jarfile to java project and continue writing the sikuli program.

Example for - SIKULI + SELENIUM WEBDRIVER

```
import org.junit.Test;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.sikuli.script.App;
import org.sikuli.script.FindFailed;
import org.sikuli.script.Pattern;
import org.sikuli.script.Screen;
public class sikuliFirstTest {
@Test
public void functionName() throws FindFailed {
// Create a new instance of the Firefox driver
```

```
WebDriver driver = new FirefoxDriver();
// And now use this to visit Google
driver.get("http://www.google.com");
//Create and initialize an instance of Screen object
Screen screen = new Screen();
//Add image path
Pattern image = new Pattern("C:\\searchButton.png");
 //Wait 10ms for image
screen.wait(image, 10);
 //Click on the image
screen.click(image);
  }
}
```

Example for - TC_adding new emp –Clicking on browse button and selecting an file to upload an image using WebDriver with Sikuli

```
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.sikuli.script.Pattern;
import org.sikuli.script.Screen;
public class WinPopup {
   public static void main(String[] args) throws InterruptedException {
WebDriver driver=new FirefoxDriver();
      driver.navigate().to("https://sureshitacademy.in/login.php");
      driver.findElement(By.xpath("//input[@type='text']")).sendKeys("sureshit");
driver.findElement(By.xpath("//input[@type='password']")).sendKeys("sureshit");
      driver.findElement(By.xpath("//input[@type='Submit']")).click();
      Thread.sleep(5000L);
      //Selecting the frame
      driver.switchTo().frame("rightMenu");
      //Clicking on Add Button
driver.findElement(By.xpath("//*[@id='standardView']/div[3]/div[1]/input[1]")).click();
      driver.manage().timeouts().implicitlyWait(2, TimeUnit.SECONDS);
      driver.findElement(By.xpath("//*[@id='txtEmpLastName']")).sendKeys("aaa");
      driver.findElement(By.xpath("//*[@name='txtEmpFirstName']")).sendKeys("bbb");
      try      {
         //Create and initialize an instance of Screen object
         Screen screen = new Screen();
         //Add image path
 Pattern browse = new Pattern("C:\\Images\\Browse.png");
 Pattern pictures = new Pattern("C:\\images\\pictures.png");
 Pattern samplepictures = new Pattern("C:\\images\\samplepictures.png");
  Pattern Desertimage = new Pattern("C:\\images\\Desertimage.png");
  Pattern Openbtn = new Pattern("C:\\images\\Openbtn.png");
```

```
        //Wait 10ms for image
        screen.wait(browse, 10);
        //Click on the image
        screen.click(browse);
      screen.wait(pictures, 10);
        screen.click(pictures);
        screen.wait(samplepictures, 10);
        screen.click(samplepictures);
        screen.wait(Desertimage, 10);
        screen.click(Desertimage);
        screen.wait(Openbtn, 10);
        screen.click(Openbtn);
       }
      catch(Exception e)      {
        System.out.println(e);
      }
      driver.findElement(By.xpath("//*[@id='btnEdit']")).click();
      Thread.sleep(2000L);
      System.out.println("New Employee Added");
      driver.switchTo().defaultContent();
      driver.findElement(By.xpath("//*[@id='option-menu']/li[3]/a")).click();
      driver.quit();
   }
}
```

Selenium GRID

What is need of Selenium Grid?

Selenium Grid– A distributed test execution environment to speed up the execution of a test pass.

Here are few problems with such a setup:

What if you want to execute your test cases for different Operating Systems?

How to run your test cases in different version of same browser?

How to run your test cases for multiple browsers?

Why a scenario should wait for execution of other test cases even if it does not depend upon any test cases?

All these problems are addressed in Selenium GRID.

How we can overcome to these problems

Basically Grid architecture is based on master slave architecture. Master machine distributes test cases to different slave machines.

There are 2 versions of Grid available. Selenium Grid 2.0 is the latest from Selenium. Selenium 1.0 was the earlier version. Most of the Selenium experts prefer using Selenium Grid 2.0 as it is packed with new features. Selenium Grid 2.0 supports both Selenium RC and Selenium WebDriver scripts.

Benefits of Selenium Grid:

Selenium Grid gives the flexibility to distribute your test cases for execution.

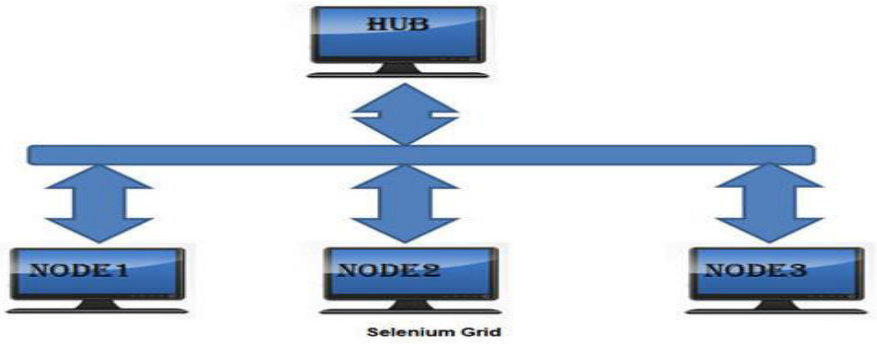
Reduces batch processing time.

Can perform multi browser testing.
Can perform multi OS testing.

Basic terminology of Selenium Grid:

Hub: Hub is central point to the entire GRID Architecture which receives all requests. There is only one hub in the selenium grid. Hub distributes the test cases across each node.

Node: There can be multiple nodes in Grid. Tests will run in nodes. Each node communicates with the Hub and performs test assigned to it.



Install Selenium GRID

Step 1: Download Selenium Server jar file from Selenium's official website which is formerly known as Selenium RC Server and save it at any location on the local disk.
URL of selenium HQ: http://www.seleniumhq.org/download/

Step 2: Open command prompt and navigate to folder where the server is located. Run the server by using below command

java -jar selenium-server-standalone-3.12.0.jar -role hub

http://localhost:4444/grid/console --- To check the node status

The hub will use the port 4444 by default. This port can be changed by passing the different port number in command prompt provided the port is open and has not been assigned a task.

Status can be checked by using the web interface:

Step 3: Go to the other machine where you intend to setup Nodes. Open the command prompt and and navigate to folder where the server is located and then run the below line.

java -jar selenium-server-standalone-3.12.0.jar -role node -hub http://localhost:4444/grid/register -port 5556

{OR}

java -jar selenium-server-standalone-3.12.0.jar -role node -hub http://localhost:4444/grid/register -port 5556

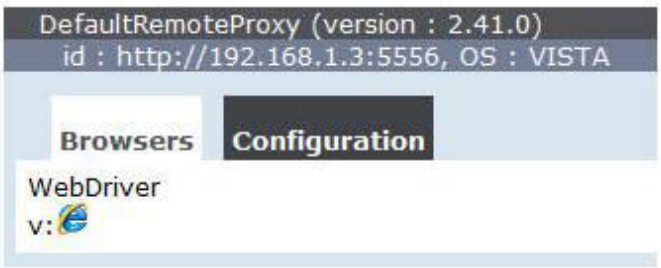
Note : Once hub and node started successfully then you can start the executing webdriver program.

Sample program for Grid

```
package com.hrms.parallel;
import java.net.URL;
```

```
import org.openqa.selenium.By;
import org.openqa.selenium.Platform;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.remote.RemoteWebDriver;
public class TC_Grid {
	public static void main(String[] args) throws Exception {
		DesiredCapabilities caps = DesiredCapabilities.firefox();
		//FirefoxOptions ff = new FIREFOX_OPTIONS().
		 //caps.setVersion("20");
		caps.setPlatform(Platform.WINDOWS);
		URL urlHub = null;
		urlHub = new URL("http://localhost:5556/wd/hub");
		RemoteWebDriver driver = new RemoteWebDriver(urlHub, caps);
		driver.navigate().to("https://sureshitacademy.in/login.php ");
		//driver.navigate().to("http://www.google.com/");
		Thread.sleep(2000);
		System.out.println(driver.getTitle());
		System.out.println("Application opned");
		driver.findElement(By.name("txtUserName")).sendKeys("sureshit");
driver.findElement(By.xpath("//input[@name='txtPassword']")).sendKeys("sureshit");
		driver.findElement(By.name("Submit")).click();
		Thread.sleep(3000);
		System.out.println("Login completed");
		driver.findElement(By.linkText("Logout")).click();
		System.out.println("Logout completed");
		driver.close();

		}
}
```

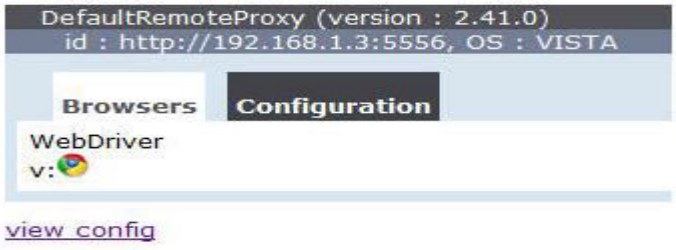
Some other examples - For Instance if you want to use only IE you can start the node by using below command:
java -jar selenium-server-standalone-2.41.0.jar -role webdriver -hub http://localhost:4444/grid/register -port 5556 -browser browserName=iexplore
Verify the browser Type along with other details in GRID Console by clicking on view config.



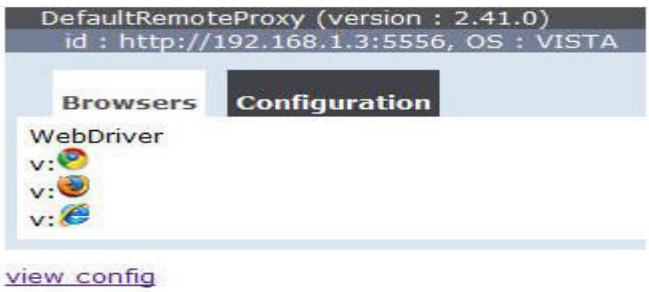
Similarly for Firefox:
java -jar selenium-server-standalone-2.41.0.jar -role webdriver -hub

http://localhost:4444/grid/register -port 5556 -browser browserName=firefox

For Chrome:
java -jar selenium-server-standalone-2.41.0.jar -role webdriver -hub http://localhost:4444/grid/register -port 5556 -browser browserName=chrome



There are few scenarios where you may need browser from each type i.e.: IE, Chrome and Firefox.
For instance you may need to use 1 IE and 1 Firefox and 1 Chrome browser
java -jar selenium-server-standalone-2.41.0.jar -role webdriver -hub http://localhost:4444/grid/register -port 5556 -browser browserName=iexplore -browser browserName=firefox -browser browserName=chrome



Grid Program Example – for WebDriver with TestNG

Prerequisite: Create Hub and nodes as explained earlier and TestNG should be configured in eclipse.

```
public class GridExample {
	@Test
	public void mailTest() throws MalformedURLException{
		DesiredCapabilities dr=null;
		if(browserType.equals("firefox")){
		dr=DesiredCapabilities.firefox();
		dr.setBrowserName("firefox");
		dr.setPlatform(Platform.WINDOWS);
		}else{
			dr=DesiredCapabilities.internetExplorer();
```

```
                dr.setBrowserName("iexplore");
                dr.setPlatform(Platform.WINDOWS);
            }
RemoteWebDriver driver=new RemoteWebDriver(new
URL("http://localhost:4444/wd/hub"),dr);
            driver.navigate().to("http://gmail.com");
    driver.findElement(By.xpath("//input[@id='Email']")) .sendKeys("username");
    driver.findElement(By.xpath("//input[@id='Passwd']")) .sendKeys("password");
            driver.close();
}
```

As in the example you have to use RemoteWebDriver if you are using GRID and you have to provide capabilities to the browser. You have to set the browser and platform as above.

In this example I have used platform as WINDOWS. You can use any platform as per your requirement.

Version of browser can also be set by using dr.setVersion("version")

Serially Execution in multiple browsers

For Instance you need to run this test serially in multiple browsers you have to configure your testng.xml .Below is the testng.xml suite for above test to run your test serially.

```
<?xml version="1.0" encoding="UTF-8"?>
<suite name="GRID SAMPLE TEST" thread-count="2">
  <test name="GRID TEST WITH SERIAL EXECTION WITH BROWSER IE">
  <parameter name ="browserType" value="IE"/>
    <classes>
      <class name ="GridExample"/>
    </classes>
  </test>
   <test name="GRID TEST WITH SERIAL EXECTION WITH BROWSER FF ">
  <parameter name ="browserType" value="firefox"/>
    <classes>
      <class name ="GridExample"/>
    </classes>
  </test>
</suite>
```

To run the test parallel, you have to change your testng.xml like below.

```
<?xml version="1.0" encoding="UTF-8"?>
<suite name="GRID SAMPLE TEST" parllel="tests" thread-count="3">
  <test name="GRID TEST WITH SERIAL EXECTION WITH BROWSER FF">
  <parameter name ="browserType" value="firefox"/>
    <classes>
      <class name ="GridExample"/>
    </classes>
  </test>
  <test name="GRID TEST WITH SERIAL EXECTION WITH BROWSER IE">
```

```
  <parameter name ="browserType" value="IE"/>
    <classes>
      <class name ="GridExample"/>
    </classes>
  </test>
</suite>
```

Here in the testng.xml you have to specify parameter asparllel="tests" and thread-count="3" describes the maximum number of threads to be executed in parallel.

Frequently Selenium interview questions and answers:

1) What is test automation or automation testing?

Automation testing uses automation tools to write and execute test cases, no manual involvement is necessary for executing an automated test suite. Testers prefer automation tools to write test scripts and test cases and then group into test suites.

Automation testing enables the use of specialized tools to automate the execution of manually designed test cases without any human intervention. Automation testing tools can access the test data, controls the execution of tests and compares the actual result against the expected result. Consequently, generating detailed test reports of the system under test.

2) What are the benefits of Automation Testing?

Benefits of Automation testing are as follows.

It allows execution of repeated test cases
It enables parallel execution
Automation Testing encourages unattended execution
It improves accuracy. Thus, it reduces human-generated errors
It saves time and money.

3. What is Selenium?

Ans. Selenium is a robust test automation suite that is used for automating web-based applications. It supports multiple browsers, programming languages, and platforms.

4. What are the different forms of Selenium?

Ans. Selenium comes in four forms-

Selenium WebDriver – Selenium WebDriver is used to automate web applications using the browser's native methods.
Selenium IDE – A firefox plugin that works on record and playback principle.
Selenium RC – Selenium Remote Control(RC) is officially deprecated by selenium and it used to work on javascript to automate the web applications.
Selenium Grid – Allows selenium tests to run in parallel across multiple machines.

5. What are some advantages of Selenium?

Ans. Following are the advantages of Selenium-

Selenium is open source and free to use without any licensing cost.
It supports multiple languages like java, ruby, python, etc.
It supports multi-browser testing.

It has a good amount of resources and helping community over the internet.
Using Selenium IDE component, non-programmers can also write automation scripts
Using the selenium grid component, distributed testing can be carried out on remote machines.

6. What are some limitations of Selenium?

Ans. Following are the limitations of Selenium-

We cannot test desktop applications using Selenium.
We cannot test web services using Selenium.
For creating robust scripts in Selenium Webdriver, programming language knowledge is required.
We have to rely on external libraries and tools for performing tasks like – logging(log4J), testing framework-(TestNG, JUnit), reading from external files(POI for excels), etc.

7. Which browsers/drivers are supported by Selenium Webdriver?

Ans. Some commonly used browsers supported by Selenium are-

Google Chrome – ChromeDriver
Firefox – FireFoxDriver
Internet Explorer – InternetExplorerDriver
Safari – SafariDriver
HtmlUnit (Headless browser) – HtmlUnitDriver
Android – Selendroid/Appium
IOS – ios-driver/Appium

8. Can we test APIs or web services using Selenium Webdriver?

Ans. No Selenium WebDriver uses the browser's native method to automate the web applications. Since web services are headless, so we cannot automate web services using selenium WebDriver.

9. What is the testing type supported by Selenium WebDriver?

Ans. Selenium Webdriver can be used for performing automated functional and regression testing.

10. What are the various ways of locating an element in Selenium?

Ans. The different locators in Selenium are-

- Id
- XPath
- CSS selector
- className
- tagName
- name
- link text
- partialLinkText

11.What is an XPath?

Ans. Xpath or XML path is a query language for selecting nodes from XML documents. XPath is one of the locators supported by Selenium Webdriver.

12. What is an absolute XPath?

Ans. An absolute XPath is a way of locating an element using an XML expression

beginning from root node i.e. HTML node in case of web pages. The main disadvantage of absolute XPath is that even with the slightest change in the UI or any element the whole absolute XPath fails.
Example – html/body/div/div[2]/div/div/div/div[1]/div/input

13. What is a relative XPath?

Ans. A relative XPath is a way of locating an element using an XML expression beginning from anywhere in the HTML document. There are different ways of creating relative XPaths which are used for creating robust XPaths (unaffected by changes in other UI elements).
Example – //input[@id='username']

14. What is the difference between single slash(/) and double slash(//) in XPath?

Ans. In XPath a single slash is used for creating XPaths with absolute paths beginning from the root node.
Whereas double slash is used for creating relative XPaths.

15) How many types of WebDriver API's are available in Selenium?

The list of WebDriver API's which are used to automate browser include:

- AndroidDriver
- ChromeDriver
- EventFiringWebDriver
- FirefoxDriver
- HtmlUnitDriver
- InternetExplorerDriver
- iPhoneDriver
- iPhoneSimulatorDriver
- RemoteWebDriver

16) Explain the difference between assert and verify commands?

Assert: Assert command checks if the given condition is true or false. If the condition is true, the program control will execute the next phase of testing, and if the condition is false, execution will stop, and nothing will be executed.

Verify: Verify command also checks if the given condition is true or false. It doesn't halt program execution, i.e., any failure during verification would not stop the execution, and all the test phases would be executed.

17) What is the difference between findElement() and findElements()?

findElement(): It is used to find the first element within the current page using the given "locating mechanism". It returns a single WebElement.

findElements(): It uses the given "locating mechanism" to find all the elements within the current page. It returns a list of web elements.
How can we launch different browsers in Selenium WebDriver?
Ans. By creating an instance of driver of a particular browser-

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

18. How can we clear a text written in a textbox?

Ans. Using a clear() method we can delete the text written in a textbox.

driver.findElement(By.id("elementLocator")).clear();

19. Explain the difference between close and quit command.

Ans. driver.close() – Used to close the current browser having a focus
driver.quit() – Used to close all the browser instances

20. How to switch between multiple windows in selenium?

Ans. Selenium has driver.getWindowHandles() and driver.switchTo().window("{windowHandleName}") commands to work with multiple windows. The getWindowHandles() command returns a list of ids corresponding to each window and on passing a particular window handle to driver.switchTo().window("{windowHandleName}") command we can switch control/focus to that particular window.

```
for (String windowHandle : driver.getWindowHandles()) {
    driver.switchTo().window(handle);
}
```

21. What is the difference between driver.getWindowHandle() and driver.getWindowHandles() in selenium?

Ans. driver.getWindowHandle() returns a handle of the current page (a unique identifier)
Whereas driver.getWindowHandles() returns a set of handles of the all the pages available.

22. How can we move to a particular frame in selenium?

Ans. The driver.switchTo() commands can be used for switching to frames.

driver.switchTo().frame("{frameIndex/frameId/frameName}");
For locating a frame we can either use the index (starting from 0), its name or Id.

23.Can we move back and forward in the browser using selenium?

Ans. Yes, using driver.navigate().back() and driver.navigate().forward() commands we can move backward and forward in a browser.

24. Is there a way to refresh the browser using selenium?

Ans. There a multiple ways to refresh a page in selenium-

Using driver.navigate().refresh() command
Using sendKeys(Keys.F5) on any textbox on the webpage
Using driver.get("URL") on the current URL or using driver.getCurrentUrl()
Using driver.navigate().to("URL") on the current URL or driver.navigate().to(driver.getCurrentUrl());

25. How can we maximize the browser window in selenium?

Ans. We can maximize browser window in selenium using the following command-

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

26. How to delete cookies in selenium?

Ans. Using deleteAllCookies() method-

driver.manage().deleteAllCookies();

27. What is an implicit wait in Selenium?

Ans. An implicit wait is a type of wait that waits for a specified time while locating an element before throwing NoSuchElementException. By default, selenium tries to find elements immediately when required without any wait. So, it is good to use implicit wait. This wait is applied to all the elements of the current driver instance.

```
driver.manage().timeouts().implicitlyWait(5, TimeUnit.SECONDS);
```

28. What is an explicit wait in Selenium?

Ans. An explicit wait is a type of wait that is applied to a particular web element until the expected condition specified is met.

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

WebElement element =
wait.until(ExpectedConditions.elementToBeClickble(By.id("elementId")));
```

29. Write the code to double click an element in selenium?

Ans. Code to double click an element in selenium-

```
Actions action = new Actions(driver);
WebElement element=driver.findElement(By.id("elementId"));
action.doubleClick(element).perform();
```

30. Write the code to right-click an element in selenium?

Code to right-click an element in selenium-

```
Actions action = new Actions(driver);
WebElement element=driver.findElement(By.id("elementId"));
action.contextClick(element).perform();
```

31. How to mouse hover an element in selenium?

Ans. Code to mouse hover over an element in selenium-

```
Actions action = new Actions(driver);
WebElement element=driver.findElement(By.id("elementId"));
action.moveToElement(element).perform();
```

32. How to fetch the current page URL in selenium?

Ans. Using getCurrentURL() command we can fetch the current page URL-

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

33. How can we find all the links on a web page?

Ans. All the links are of anchor tag 'a'. So by locating elements of tagName 'a' we can find all the links on a webpage.

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

34. What are some commonly encountered exceptions in selenium?

Ans. Some of the commonly seen exceptions in Selenium are-

NoSuchElementException – When no element could be located from the locator provided.

ElementNotVisibleException – When an element is present in the dom but is not visible.
NoAlertPresentException – When we try to switch to an alert but the targetted alert is not present.
NoSuchFrameException – When we try to switch to a frame but the targetted frame is not present.
NoSuchWindowException – When we try to switch to a window but the targetted window is not present.
UnexpectedAlertPresentException – When an unexpected alert blocks the normal interaction of the driver.
TimeoutException – When a command execution gets a timeout.
InvalidElementStateException – When the state of an element is not appropriate for the desired action.
NoSuchAttributeException – When we are trying to fetch an attribute's value but the attribute is not correct
WebDriverException – When there is some issue with driver instance preventing it from getting launched.

35.How can we capture screenshots in selenium?

Ans. Using the getScreenshotAs method of TakesScreenshot interface, we can take the screenshots in selenium.

```
File scrFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
FileUtils.copyFile(scrFile, new File("D:\\testScreenShot.jpg"));
```

36. How to handle dropdowns in selenium?

Ans. Using Select class-

```
Select countriesDropDown = new Select(driver.findElement(By.id("countries")));
dropdown.selectByVisibleText("India");

//or using index of the option starting from 0
dropdown.selectByIndex(1);

//or using its value attribute
dropdown.selectByValue("Ind");
```

37. How can we check if an element is getting displayed on a web page?

Ans. Using the isDisplayed() method we can check if an element is getting displayed on a web page.

```
driver.findElement(By locator).isDisplayed();
```

38. What is Robot API?

Ans. Robot API is used for handling Keyboard or mouse events. It is generally used to upload files to the server in selenium automation.

```
Robot robot = new Robot();

//Simulate enter key action
```

robot.keyPress(KeyEvent.VK_ENTER);

39. How to do file upload in selenium?

Ans. File upload action can be performed in multiple ways-

Using element.sendKeys("path of file") on the webElement of input tag and type file i.e. the elements should be like –
<input type="file" name="fileUpload">
Using Robot API.
Using the AutoIT API.
How to do drag and drop in Selenium?
Using Action class, drag and drop can be performed in selenium. Sample code-

```
Actions builder = new Actions(driver);
Action dragAndDrop = builder.clickAndHold(SourceElement)
.moveToElement(TargetElement)
.release(TargetElement)
.build();
dragAndDrop.perform();
```

40. How to handle alerts in Selenium?

Ans. In order to accept or dismiss an alert box, the alert class is used. This requires first switching to the alert box and then using accept() or dismiss() command as the case may be.

```
Alert alert = driver.switchTo().alert();
//To accept the alert
alert.accept();

Alert alert = driver.switchTo().alert();
//To cancel the alert box
alert.dismiss();
```

41. What is Selenium Grid?

Ans. Selenium grid is a tool that helps in the distributed running of test scripts across different machines having different browsers, browser versions, platforms, etc in parallel. In the selenium grid, there is a hub that is a central server managing all the distributed machines known as nodes.

42. What are some advantages of the Selenium grid?

Ans. The advantages of the Selenium grid are-

It allows running test cases in parallel thereby saving test execution time.
Multi-browser testing is possible using the Selenium grid by running the test on machines having different browsers.
It allows multi-platform testing by configuring nodes having different operating systems.

43. What is TestNG?

Ans. TestNG(NG for Next Generation) is a testing framework that can be integrated with selenium or any other automation tool to provide multiple capabilities like

assertions, reporting, parallel test execution, etc.

44. What are some advantages of TestNG?

Ans. Following are the advantages of TestNG-

TestNG provides different assertions that help in checking the expected and actual results.
It provides parallel execution of test methods.
We can define the dependency of one test method over others in TestNG.
We can assign priority to test methods in selenium.
It allows grouping of test methods into test groups.
It allows data-driven testing using @DataProvider annotation.
It has inherent support for reporting.
It has support for parameterizing test cases using @Parameters annotation.

45. What is the use of the testng.xml file?

Ans. testng.xml file is used for configuring the whole test suite. In the testng.xml file, we can create a test suite, create test groups, mark tests for parallel execution, add listeners and pass parameters to test scripts. Later this testng.xml file can be used for triggering the test suite.

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

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

The benefits of using POM are as follows.
It facilitates with separate operations and flows in the UI from Verification - improves code readability
Multiple tests can use the same Object Repository because the Object Repository is independent of Test Cases.
Reusability of code

47. What is Page Factory?

Page Factory gives an optimized way to implement Page Object Model. When we say it is optimized, it refers to the fact that the memory utilization is very good and also the implementation is done in an object oriented manner.

Page Factory is used to initialize the elements of the Page Object or instantiate the Page Objects itself. Annotations for elements can also be created (and recommended) as the describing properties may not always be descriptive enough to differentiate one object from the other.

The concept of separating the Page Object Repository and Test Methods is followed here also. Instead of having to use 'FindElements', we use annotations like: @FindBy to find WebElement, and initElements method to initialize web elements from the Page Factory class. @FindBy can accept tagName, partialLinkText, name, linkText, id, css, className & xpath as attributes.

48.What are the Selenium Tools, and Testing framework that you are using in your Current Project?
Ans: We are using,
Selenium WebDriver for Creating Test Cases.
Java Programming for Enhancing Test cases.
TestNG Testing Framework for Grouping Test cases, executing Test batches and generating Test reports. Also used,
Inscept , Firebug and Firepath for inspecting elements.
IE Browser driver, Chrome browser driver for Cross Browser testing.
49.What is your project domain and Application Environment?
Ans: Our project is Banking Application, Our AUT (Application Under Test) developed using Java Technology and Database is Oracle.
50.What are the major challenges in Functional Test Automation?
Ans:Object Identification.
wait statement
Debugging Issues. etc…
51.How you conducted Batch Testing in your project?
Ans: We conducted Batch Testing using TestNG Testing Framework.
52.How you conducted Cross-browser testing in your project?
Ans: Using Mozilla Firefox, IE and Google Chrome browsers(Downloaded IE and Google Chrome Browser drivers). we executed Test cases.
53.How you handled duplicate Elements in your project?
Ans: Using the index property of Elements we handled duplicate objects.
54.How many Test cases you wrote for your Project/Module?
Ans: I prepared around 120 Test cases in my Module.
55.How many defects you detected and give one example?
Ans: I detected nearly 20 Defects in which 5 defects are Severe defects.
56.How you selected Test cases for Regression Testing?
Ans: We selected Test cases for Regression Testing based on Defect effected Test cases and from defect dependent Test cases.
57.Did you create any reusable components?
Ans: Yes, We created some reusable components in our project for Login Functionality, Registration Functionality, etc.
58.Did you find any Test Scenarios that not to be automated in your project using Selenium?
Ans: Yes, We find some Some Test Scenarios in our Current project, Functionalities that require more user interaction, Functionalities that require Dynamic test data submission.
59.What Defect management / Test management tool you used in your project?
Ans: We are using Jira Test Management tool in our project with Selenium.
60.Did you involve in the Selenium Test environment Setup?
Ans: Yes, I involved in Selenium Environment Setup in my Current Project. As per my project, we selected Java, Selenium WebDriver and TestNG Framework.

We Downloaded Eclipse IDE and Extracted.
Downloaded Java Software and Installed.
Environment Variable path setup.

Downloaded Selenium WebDriver Java language bindings(jar files) and added to Java Project in Eclipse.
Downloaded and Installed TestNG Framework from Eclipse IDE.

61. How do you handle Ajax dropdowns?

With help of Selenium Sync commands like ImplicitWait, WebDriverWait or FluentWait.

62. What is the default port for Selenium Grid?

4444

63. How to find broken images in a page using Selenium Web driver.

Get xpath and then using tag name 'a'; get all the links in the page
Use HttpURLConnector class and sent method GET
Get the response code for each link and verify if it is 404/500

```
List links = driver.findElements(By.tagName("a"));
for (int i = 0; i < links.size(); i++) {
WebElement element = links.get(i);
// By using "href" attribute, we could get the url of the requried link
String url = element.getAttribute("href");
//System.out.println(url);
URL link = new URL(url);
// Create a connection using URL object (i.e., link)
HttpURLConnection httpConn = (HttpURLConnection) link.openConnection();
// Set the timeout for 2 seconds
httpConn.setConnectTimeout(2000);
// connect using connect method
httpConn.connect();
// use getResponseCode() to get the response code.
if (httpConn.getResponseCode() >= 400) {
   System.out.println(url + " - " + "is Broken Link");
   } else {
      System.out.println(url + " - " + "is valid Link");
   }
```

64. Write down scenarios which we can't automate?

Barcode Reader, Captcha etc.

65. How do you manage the code versions in your project?

Using SVN, GitHub or other versioning tools

66. How to count total no of hyperlinks in a page?

Listalllinks=driver.finElements(By.tagname("a");
System.out.println(alllinks.size());

67. What type of tests have you automated?

Our main focus is to automate test cases to do Regression testing, Smoke testing, and Sanity testing. Sometimes based on the project and the test time estimation, we do focus on End to End testing.

68. How many test cases you have automated per day?

It depends on Test case scenario complexity and length.
I did automate 2-5 test scenarios per day when the complexity is limited.

Sometimes just 1 or fewer test scenarios in a day when the complexity is high.

69. When do you use Selenium Grid?

Selenium Grid can be used to execute same or different test scripts on multiple platforms and browsers concurrently so as to achieve distributed test execution.

70. What are the advantages of Selenium Grid?

- It allows running test cases in parallel thereby saving test execution time.
- It allows multi-browser testing
- It allows us to execute test cases on multi-platform

71.How would you make sure that a page is loaded using Selenium and Webdriver?

Ans. In Selenium, you can use the below lines of code to check for the successful loading of a web page. The best approach is by selecting an element from the page & stand by until it becomes clickable.

```
selenium.waitForPageToLoad("5000");
// Or
while (!(selenium.isElementPresent("any page element ")==true)) {
 selenium.setSpeed("5");
 Thread.sleep(5);
}
```

or

Below is the Webdriver specific code to achieve the same objective.

```
WebDriverWait check = new WebDriverWait(driver, 100);
check.until(ExpectedConditions.anyElement(By.id(id)));
```

72.How would you fill a text field without calling the sendKeys()?

Ans. It's a bit slower than the sendKeys() method but we do have means to type in a text field. See the Java code given below.

```
JavascriptExecutor JS = (JavascriptExecutor)webdriver;
JS.executeScript("document.getElementById(User').value='admin@testmail.com'");
JS.executeScript("document.getElementById('Pass').value='######'");
```

73.How can you check the state of a checkbox/radio button?

Ans. We can call the isSelected() method to test the status of these elements.

Example Code:

```
boolean test = driver.findElement(By.xpath("checkbox/radio button
XPath")).isSelected();
```

74.How would you simulate the right click operation in WebDriver?

Ans. You can make use of the Actions class features.

```
Actions test = new Actions(driver); // Here, driver is the object of WebDriver class.
test.moveToElement(element).perform();
test.contextClick().perform();
```

75.What is the best way to check for the highlighted text on a web page?

Ans. Use the below code to verify the highlighted text for an element on the web page.

```
String clr =
driver.findElement(By.xpath("//a[text()='TechBeamers']")).getCssValue("color");
String bkclr =
driver.findElement(By.xpath("//a[text()='TechBeamers']")).getCssValue("background-
color");
System.out.println(clr);
```

System.out.println(bkclr);

76. What is an 'element'?

Every single object that is present on the webpage is called an element.

77.What's Junit?

Junit is a Java-based framework designed for unit testing.

78.How to skip a test method in TestNG?

If you want to skip a certain test method within TestNG, you'll have to set that test's parameter to "false" in the annotation area.

79.Briefly explain what the following snippet of Java code does:

WebElement sample = driver.findElement(By.xpath("//*[contains(text(), 'data')]"));

It defines a variable sample of type WebElement, and uses an XPath search to initialize it with a reference to an element that contains the text value "data".

80.Is it possible to automate captcha?

Yes, we can automate the captcha, but there is a limitation that we can automate our own captcha but not others'

For example, a company has captcha on their website, so somebody has to check it, but at the same time, it is not possible for a manual tester to check all the captcha's. So we have to automate the captcha in the dev/QA environment by getting the captcha answer in some attribute of the element, so based on that attribute, we can enter the value to the text bar, which accepts captcha value.

We should remove this attribute while pushing the code to the Live environment.

81. What are the annotations used in TestNG?

@Test, @BeforeSuite, @AfterSuite, @BeforeTest, @AfterTest, @BeforeClass, @AfterClass, @BeforeMethod, @AfterMethod

82 How do you read data from excel?

FileInputStream fis = new FileInputStream ("path of excel file");
Workbook wb = WorkbookFactory.create (fis);
Sheet s = wb.getSheet ("sheetName")
String value = s.getRow (rowNum).getCell (cellNum).getStringCellValue ();

83. What is the use of xpath?

It is used to find the WebElement in web page. It is very useful to identify the dynamic web elements.

84. What is the alternate way to click on login button?

Use submit () method but it can be used only when attribute type=submit.

85. How do you verify if the checkbox/radio is checked or not?

We can use isSelected () method.

Syntax – driver.findElement (By.xpath ("xpath of the checkbox/radio button")).isSelected ();

If the return value of this method is true then it is checked else it is not.

86. Give the example for method overload in WebDriver.

Frame (string), frame (int), and frame (WebElement).

87. How do you simulate browser back and forward?

Driver. Navigate ().back ();
Driver. Navigate ().forward ();

88. How do you get the current page URL?

driver.getCurrentUrl ();

89. What is the difference between '/' and '//'?

// It is used to search in the entire structure.

/ It is used to identify the immediate child.

90. How do you clear the contents of a textbox in selenium?

Use clear () method.

driver.findElement (By.xpath ("xpath of box")).clear ();

91. What are the prerequisites to run selenium WebDriver?

JDK, Eclipse, WebDriver (selenium standalone jar file), browser, application to be tested.

92. What are the advantages of selenium WebDriver?

a) It supports with most of the browsers like Firefox, IE, Chrome, Safari, Opera etc.

b) It supports with most of the language like Java, Python, Ruby, C# etc.

b) Doesn't require to start server before executing the test script.

c) It has actual core API which has binding in a range of languages.

d) It supports of moving mouse cursors.

93. How to get the number of frames on a page?

List < WebElement> frames List = driver.findElements (By.xpath ("//iframe"));

int numOfFrames = frameList.size ();

94. What is the use of get Options () method?

get Options () is used to get the selected option from the dropdown list.

95. What is the use of deselect All () method?

It is used to deselect all the options which have been selected from the dropdown list.

96. Is WebElement an interface or a class?

WebDriver is an Interface.

97. Firefox Driver is class or an interface and from where is it inherited ?

Firefox Driver is a class. It implements all the methods of WebDriver interface.

98.What are the benefits of using TestNG?

a) TestNG allows us to execute of test cases based on group.

b) In TestNG Annotations are easy to understand.

c) Parallel execution of Selenium test cases is possible in TestNG.

d) Three kinds of report generated

e) Order of execution can be changed

f) Failed test cases can be executed

g) Without having main function we can execute the test method.

h) An xml file can be generated to execute the entire test suite. In that xml file we can rearrange our execution order and we can also skip the execution of particular test case.

99.While explaining the framework, what are points which should be covered?

a) What is the frame work.

b) Which frame work you are using.

c) Why This Frame work.

d) Architecture.

e) Explanation of every component of frame work.

f) Process followed in frame work.

g) How & when u execute the frame work.

h) Code (u must write code and explain).
i) Result and reporting.

100.How to press Shift+Tab?

String press = Keys.chord(Keys.SHIFT,Keys.ENTER);
webelement.sendKeys(press);

Syntax for WebDriver Methods

1. Creating New Instance Of Firefox Driver – this will open an new Empty browser

WebDriver driver = new FirefoxDriver();

2. Command To Open URL In Browser

driver.get("http://selenium-suresh.blogspot.com");

This syntax will open specified URL of software web application in web browser.

3. Clicking on any element or button of webpage

driver.findElement(By.id("id of any element or button")).click();

4. **Store text of targeted element in variable** - This will retrieve text from targeted element of software web application page and will store it in variable = suresh

String suresh = driver.findElement(By.tagName("select")).getText();

5. Typing text in text box or text area.

driver.findElement(By.name("txtboxname")).sendKeys("My First Name");

6. **Applying Implicit wait in webdriver** - This syntax will force webdriver to wait for 15 second if element not found on page of software web application.

driver.manage().timeouts().implicitlyWait(15, TimeUnit.SECONDS);

7. Applying Explicit wait in webdriver - This will wait for till 15 seconds for expected text "Time left: 7 seconds" to be appear on targeted element.

WebDriverWait wait = new WebDriverWait(driver, 15);
wait.until(ExpectedConditions.textToBePresentInElementLocated(By.xpath("xpathofelement"), "Time left: 7 seconds"));

8. Get page title in selenium webdriver

driver.getTitle();

9. Get Current Page URL In Selenium WebDriver- It will retrieve current page URL and you can use it to compare with your expected URL.

driver.getCurrentUrl();

10. Get domain name using java script executor - This will retrieve your software application's domain name using webdriver's java script executor interface and store it in to variable.

JavascriptExecutor javascript = (JavascriptExecutor) driver;
String CurrentURLUsingJS=(String)javascript.executeScript("return document.domain");

11. Generate alert using webdriver's java script executor interface -- It will generate alert during your selenium webdriver test case execution.

JavascriptExecutor javascript = (JavascriptExecutor) driver;
javascript.executeScript("alert('Test Case Execution Is started Now..');");

12. Selecting or Deselecting value from drop down in selenium webdriver.

Select By Visible Text-- It will select value from drop down list using visible text value = "Audi".

Select mydrpdwn = new Select(driver.findElement(By.id("Carlist")));
mydrpdwn.selectByVisibleText("Audi");

Select By Value- It will select value by value = "Italy".
Select listbox = new Select(driver.findElement(By.xpath("//select[@name='FromLB']")));
listbox.selectByValue("Italy");
Select By Index- It will select value by index= 0(First option).
Select listbox = new Select(driver.findElement(By.xpath("//select[@name='FromLB']")));
listbox.selectByIndex(0);
Deselect by Visible Text- It will deselect option by visible text = Russia from list box.
Select listbox = new Select(driver.findElement(By.xpath("//select[@name='FromLB']")));
listbox.deselectByVisibleText("Russia");
Deselect by Value- It will deselect option by value = Mexico from list box.
Select listbox = new Select(driver.findElement(By.xpath("//select[@name='FromLB']")));
listbox.deselectByValue("Mexico");
Deselect by Index- It will deselect option by value = Mexico from list box.
Select listbox = new Select(driver.findElement(By.xpath("//select[@name='FromLB']")));
listbox.deselectByIndex(5);
It will deselect option by Index = 5 from list box.
Deselect All - It will remove all selections from list box of software application's page.
Select listbox = new Select(driver.findElement(By.xpath("//select[@name='FromLB']")));
listbox.deselectAll();
isMultiple()-It will return true if select box is multiselect else it will return false
Select listbox = new Select(driver.findElement(By.xpath("//select[@name='FromLB']")));
boolean value = listbox.isMultiple();
13. Navigate to URL or Back or Forward in Selenium Webdriver - 1st command will navigate to specific URL, 2nd will navigate one step back and 3rd command will navigate one step forward.
driver.navigate().to("http://selenium-suresh.blogspot.com");
driver.navigate().back();
driver.navigate().forward();
14. Verify Element Present in Selenium WebDriver -- It will return true if element is present on page, else it will return false in variable iselementpresent.
Boolean iselementpresent =driver.findElements(By.xpath("//input[@id='text2']")).size()!= 0;
15. Capturing entire page screenshot in Selenium WebDriver - It will capture page screenshot and store it in your D: drive.
File screenshot = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
FileUtils.copyFile(screenshot, new File("D:\\screenshot.jpg"));
16. Generating Mouse Hover Event In WebDriver- This will move mouse on targeted element.
Actions actions = new Actions(driver);
WebElement moveonmenu = driver.findElement(By.xpath("//div[@id='menu1']/div"));
actions.moveToElement(moveonmenu).build().perform();
17. Handling Multiple Windows In Selenium WebDriver.
Get All Window Handles.—This will give you to get window handle and then how to switch from one window to another window.
Set<String> AllWindowHandles = driver.getWindowHandles();
Extract parent and child window handle from all window handles.

String window1 = (String) AllWindowHandles.toArray()[0];
String window2 = (String) AllWindowHandles.toArray()[1];
Use window handle to switch from one window to other window.
driver.switchTo().window(window2);
18. Check Whether Element is Enabled Or Disabled In Selenium Web driver- This will verify that element (text box) fname is enabled or not. You can use it for any input element.
boolean fname = driver.findElement(By.xpath("//input[@name='fname']")).isEnabled();
System.out.print(fname);
19. Selenium WebDriver Assertions With TestNG Framework
assertEquals
Assert.assertEquals(actual, expected);
assertEquals assertion helps you to assert actual and expected equal values.
assertNotEquals
Assert.assertNotEquals(actual, expected);
assertNotEquals assertion is useful to assert not equal values.
assertTrue
Assert.assertTrue(condition);
assertTrue assertion works for boolean value true assertion.
assertFalse
Assert.assertFalse(condition);
assertFalse assertion works for boolean value false assertion.
20. **Submit() method to submit form**
driver.findElement(By.xpath("//input[@name='Company']")).submit(); It will submit the form.
21. Handling Alert, Confirmation and Prompts Popups
String myalert = driver.switchTo().alert().getText(); To store alert text.
driver.switchTo().alert().accept(); To accept alert.
driver.switchTo().alert().dismiss(); To dismiss confirmation.
driver.switchTo().alert().sendKeys("This Is John"); To type text In text box of prompt popup.
22.Handling DRAG and DROP
WebDriver d = new FirefoxDriver();
Actions a=new Actions(d);
a.dragAndDrop(d.findElement(By.id("draggable")),d.findElement(By.id("droppable"))).build().perform;
23.Handling the frames in Webdriver
To Enter/Select the Frame – driver.switchTo().frame("frameid/name / index")
To Exit from Frame - driver.switchTo().defaultContent()
24. CALENDAR popups -/*IRCTC calendar*/
driver.findElement(By.id("calendar_icon1")).click();
driver.findElement(By.xpath("//div[@id='CalendarControl']/table[tbody[tr[td[text()='October 2012']]]]/descendant::a[text()='5']")).click();
25. Context Click (Right Click)
WebElement parentMenu = driver.findElement(By.linkText("Tourist Trains")); Actions act = new Actions(driver); //Create Action object for Driver
act.contextClick(parentMenu).build().perform(); //Context Click

```
act.sendKeys(Keys.ARROW_RIGHT).build().perform(); Thread.sleep(1000);
act.sendKeys(Keys.ARROW_DOWN).build().perform(); Thread.sleep(1000);
act.sendKeys(Keys.ENTER).build().perform();
```

26. Other Browser (Internet Explorer)

```
WebDriver driver =new InternetExplorerDriver();
driver.get("http://www.google.com");
```

27. Other Browser (Chrome)

```
WebDriver driver = new ChromeDriver();
driver.get("http://www.google.com");
```

28. Using Auto-IT tool -To handle windows authentication and calling that code in Selenium

--Auto-IT code

```
WinWaitActive("Authentication Required")
Send("admin")
Send("{TAB} admin{TAB} {ENTER}")
```

Save the file as default save.(Authentication1.exe)

Calling AutoIT .exe file in selenium

```
Process P = Runtime.getRuntime().exec("D:\\AUTOIT\\Authentication1.exe");
```

29.File download using RobotClass

```
Robot robot = new Robot();
//it clicks on SaveFile Radio btn
robot.keyPress(KeyEvent.VK_ALT);
robot.keyPress(KeyEvent.VK_S);
robot.keyRelease(KeyEvent.VK_ALT);
robot.keyRelease(KeyEvent.VK_S);
Thread.sleep(2000);
//Clicks on OK btn
robot.keyPress(KeyEvent.VK_ENTER);
robot.keyRelease(KeyEvent.VK_ENTER);
Thread.sleep(2000);
System.out.println("File downloaded successfully");
```

30.File Uplad using WebDriver

```
WebElement fileInput =
driver.findElement(By.xpath("//input[@type='file'][@name='photofile']"));
fileInput.sendKeys("C:\\Users\\Public\\Pictures\\Sample Pictures\\Desert.jpg");
   Thread.sleep(5000);
   System.out.println("File uploaded successfully");
```

How to Explain Project in Interview

1. Overview of Client & Project Introduction:
2. Modules description:
3. Main functionality of your application:
4. Tools, Technologies, and Platform used:
5. Personal contribution and your role in the project:

Example: I worked on ABC Project. This project has total 4 modules. I worked on B module. Explain the functionality flow. I was involved in writing test cases/designing test

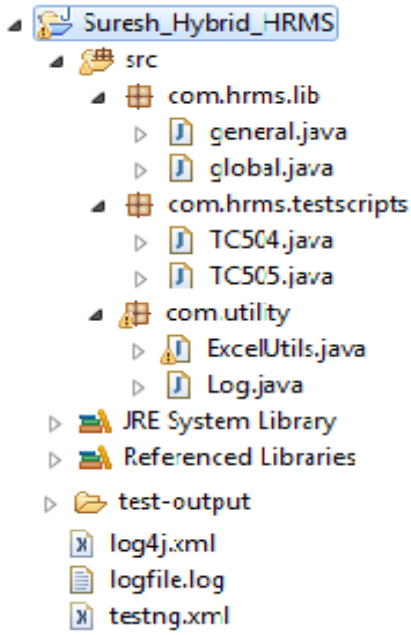
scripts for which we used <tool name>, we executed the test cases/test scripts<tool name>and while testing, we found bugs. For bugs, we used <toolname>. You can mention the framework being used during automation testing and elaborate more upon asking.
Note: You have to give the answer within 2-3 minutes so that interviewer can ask you further questions. Do not spend 5 minutes or more in explaining the project. You will miss the opportunity. Do not bore the interviewer.

How to Explain Automation Framework in Interview

This answer will vary from person to person in general we can explain as below.
In current project I am working on Hybrid Framework which is having the features of **– (WebDriver + TestNG + TestSuite+Log4J).**

- WebDriver had used to develop automation scripts
- TestNg had used to Generate Html Reports
- TestSuite to Execute Group of testCases
- Log4j to Generate log file.

You need to draw the folder strucuter for the interview and explain what is the purpose of each and every file including navigation followed by one sample test script.



General.Java : All re-usable functions to perform particular action based on Manual test case steps.
Global.java : considered like object repository which are maintaing all varibles and objects information
Com.hrms.testscripts : all automation scripts need to be created in this folder
TC504.java : Accessing all required methods from general.java based on manual test case steps.
ExcelUtils.java : All re-usable functions to perform excel related activities.
Log.Java : All re-usable functions to generate logfile
Test-ouput : html reports or TestNG reports
Log4j.xml : To Generate log file
Logfile.log : printing test case step execution details in text file or logfile
Testng.xml : Test Suite to execute group of testcases.