**Selenium:**

Selenium is testing framework where you can do functionality test of your web applications.

Selenium is usually used if you want to test the web application frequently.

**Selenium components:**

1. Selenium IDE: In order to speed up test case creation this component was used. He re Selenium IDE is Firefox extension. It basically records the actions done on website and converts it to Java code. eclipse/intellij/visual studio

2. Selenium RC: Due to same origin policy testers had to install selenium core and web server on their local system. This was done to keep domain name same for selenium core and web application to be tested. Selenium RC is as web server which acts as HTTP Proxy. It tricks OS into believing both selenium core and website to be tested are on same domain.

3. Selenium web drivers: used to connect to browser/launch a code in browser. It is cross platform testing framework that could control the browser from the OS level. It controls the browser by directly communicating to it.

4. Selenium grid: Can run multiple test cases simultaneously can connect to multiple nodes to run multiple tasks. It enables to run parallel testing of applications on multiple machines. It can also be used to break down huge test case suit among many computers testing same application.

=====================================================================================

**Maven:**

Maven is a build automation tool used primarily for java projects.

First it builds software and second it describes its dependencies.

For example if you want selenium as a project to run you need all dependencies like jar files. So by using Maven we can specify all the dependencies needed for selenium and Maven will automatically download the dependencies required and include it in project.

Maven is used to download dependencies for the software program. Dependencies to be downloaded are included in POM.xml file. Once the dependencies are added to POM file and save the project, all the dependencies will be downloaded automatically.

=====================================================================================

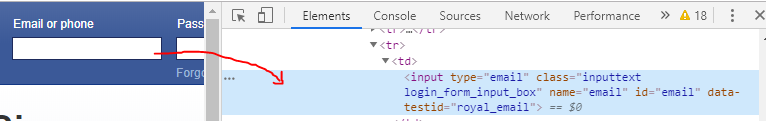
**There are 3 steps to perform web test:**

1. Find element on browser.
2. Perform an action on found elements.
3. Test and create Test report with Results.

So finding element and performing action is done by selenium web driver. Test and create test report is done by TestNG.

1. An element can be found on a webpage using following selectors:

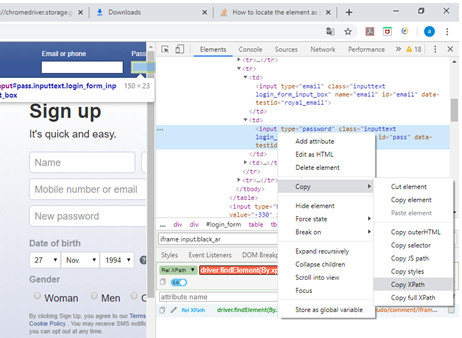
**ID, Name, Class Name, Tag Name, Link Text, Partial Link Text, XPATH**.



In above screenshot element Email or phone can be found using these selectors.

XPATH is location on webpage which is formatted in the form of xml path. In order to get XPATH there should be **chropath** extension added to chrome browser.

It can be found by just selecting the element and right click and copy xpath. Copy it in notepad.



1. Performing Action on Element:

Click()

sendKeys()

Clear()

Submit()

**package** test.test;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.testng.annotations.Test;

/\*\*

\* Hello world!

\*

\*/

**public** **class** App

{

WebDriver driver;

@Test

**public** **void** test() {

System.*setProperty*("webdriver.chrome.driver", "C:\\Users\\alokv\\Downloads\\chromedriver.exe");

driver = **new** ChromeDriver();

driver.get("http://www.facebook.com");

driver.findElement(By.*name*("email")).sendKeys("alokchinnu87@gmail.com");

driver.findElement(By.*xpath*("//\*[@id=\"pass\"]")).sendKeys("kanakeri123");

driver.findElement(By.*id*("loginbutton")).click();

}

}

Testing and Reporting in Selenium using TestNG:

**TestNG**: Next Generation

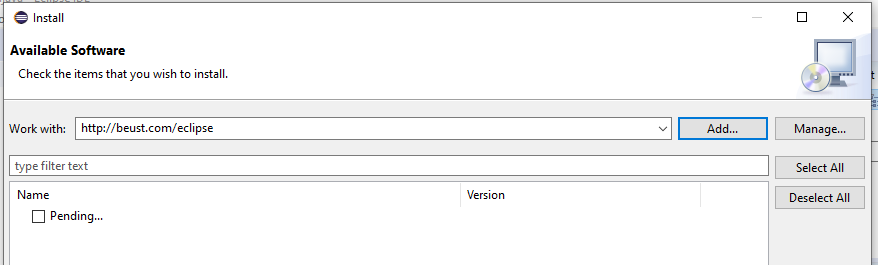
It is automated testing framework which basically helps us to perform:

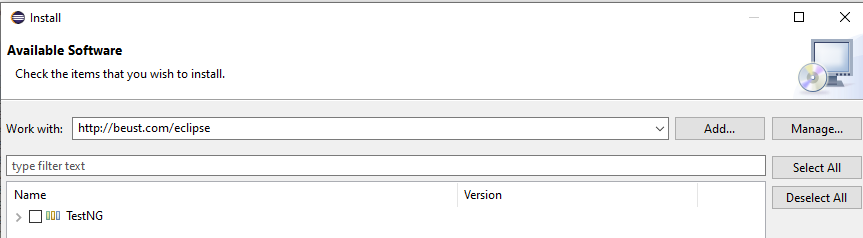
* Control flow of program
* Helps us creating HTML reports.

**Features of TestNG:**

* TestNG helps us in controlling flow of program using TestNG Annotations.
* Test cases can be grouped and prioritized.
* Parallel test execution using Grid.
* Can readily be integrated with other tools and plug-in like Eclipse IDE, build tools Ant and Maven.
* Generates HTML reports.

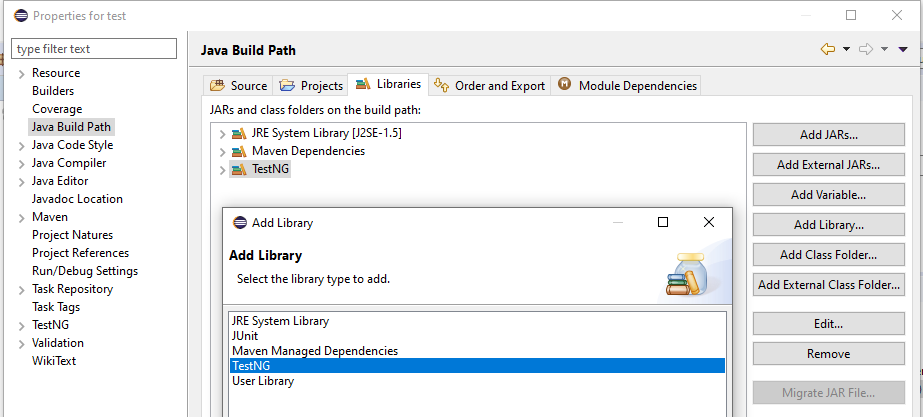
To install TestNG go to help in eclipse and Install new Software. Now new windows Available Software opens and in the work with field enter the URL [**http://beust.com/eclipse**](http://beust.com/eclipse)**.**





Now click on TestNG and click next, it will install TestNG on eclipse IDE. Once done restart eclipse IDE.

Now right click on test and go to properties 🡪Java build path and in Libraries Tab go to Add Library and select TestNG.



**Annotations in TestNG:**

**These are bunch of text strings that basically dictate or direct the flow of program.**

**@BeforeSuite**: Any function which is below @BeforeSuite would be executed before whole test suite is executed.

**@BeforeClass**: will be executed only once before the first test method in the current class is invoked.

**@Before Method**: will run before each test method.

**@Test**: Marks a class or method as part of test.

**Note: Each class contains methods. In below myMethod is created in MyClass.**

**public class MyClass {**

**static void myMethod() {**

**System.out.println("Hello World!");**

**}**

**}**

Before and after method only have to be written once but they get executed before and after each and every test which is present in the test suite.

Before class which gets executed every class.

Before Suite which gets executed only once when the program is starting.

All @before annotations are used to initialize variable or declare [something.@After](mailto:something.@After) annotation can be used for garbage collections or quitting drivers etc.

If you want to get the test output then right click on the project in eclipse and refresh. Once done we will get test-output folder and in that we will be having index.html. Just click on the properties of index.html and you will get path which is in location and copy this path and paste it in browser which will give html test report.

**Sample Test case for annotation:**

**public** **class** App

{

WebDriver driver;

@BeforeTest

**private** **void** beforetest() {

System.*setProperty*("webdriver.chrome.driver" , "C:\\Users\\alokv\\Downloads\\chromedriver.exe");

driver = **new** ChromeDriver();

}

@BeforeMethod

**private** **void** open()

{

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

}

@Test (priority = 1)

**private** **void** test() {

Boolean b = driver.findElement(By.*xpath*("//\*[@id='hplogo']")).isDisplayed();

System.***out***.println("Is the image being displayed"+b);

}

@AfterClass

**private** **void** close() {

driver.quit();

}

@Test (priority = 2)

**private** **void** search() {

driver.findElement(By.*xpath*("//input[@title='Search']")).sendKeys("intellipaat");

driver.findElement(By.*xpath*("//div[@class='FPdoLc tfB0Bf']//input[@name='btnK']")).click();

}

}

**Sample Test Case:**

Below is sample test case which does following tasks:

* Open intellipaat website [www.intellipaat.com](http://www.intellipaat.com).
* Enter Devops term and click search.
* On the search page click on DevOps Certification Training Course Online.
* On the course page verify if the page is for DevOps Certification Training Course Online.

**public** **class** App

{

WebDriver driver;

@BeforeTest

**private** **void** beforetest() {

System.*setProperty*("webdriver.chrome.driver" , "C:\\Users\\alokv\\Downloads\\chromedriver.exe");

driver = **new** ChromeDriver();

}

@BeforeMethod

**private** **void** open()

{

driver.get("http://www.intellipaat.com");

}

@Test

**private** **void** test() {

driver.findElement(By.*xpath*("//input[@placeholder='Search from 150+ experts-made courses']")).sendKeys("Devops");

driver.findElement(By.*xpath*("//button[@id='frontpagesubmitsearch']")).click();

driver.findElement(By.*xpath*("//section[@id='content']//div[3]//div[1]//div[1]//a[2]//img[1]")).click();

driver.findElement(By.*xpath*("//h1[contains(text(),'DevOps Certification Training Course Online')]")).click();

}

@AfterClass

**private** **void** close() {

driver.quit();

}

}

**Running a headless test in Selenium:**

Headless mode is without opening browser and browser opens in backend and will not open on GUI and then does the test.

This is mostly used if you’re using command line interface in which you don’t have option of opening a browser also whenever your performing testing on servers which are not GUI based.

How to run selenium test without having eclipse on system:

For that we have to create jar file of selenium test.

Usually we won’t get any Launch configuration if we don’t have main method in class. So we have to add main method. Below is java code for main method,

**public** **static** **void** main(String[] args) {

TestListenerAdapter tla = **new** TestListenerAdapter();

TestNG testng = **new** TestNG();

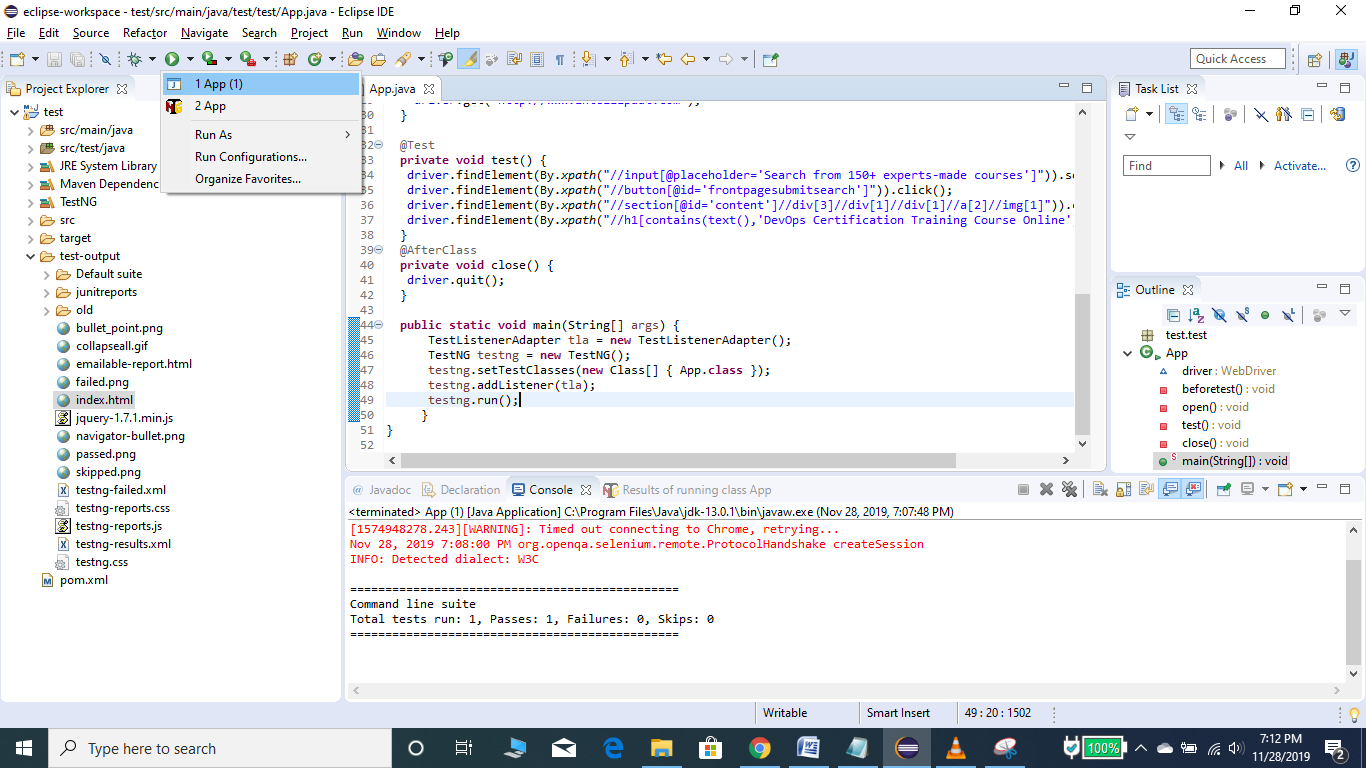
testng.setTestClasses(**new** Class[] { App.**class** });

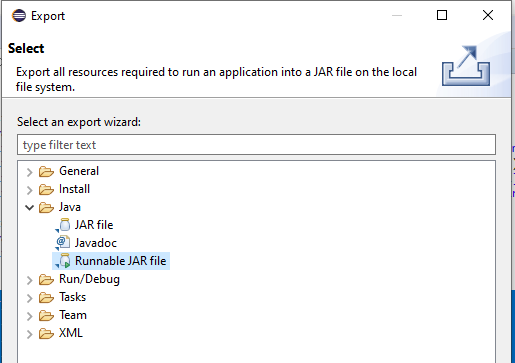
testng.addListener(tla);

testng.run();

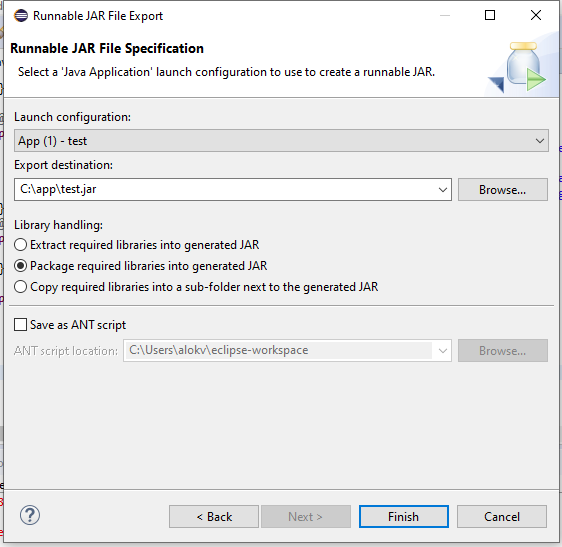
}

Once added run the App(1).Once done go to to file in eclipse and export, java, Runnable jar file.





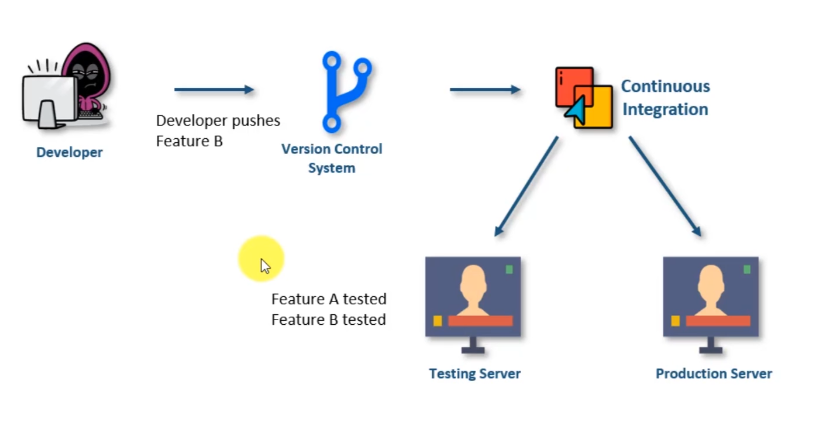
Select it and click next. Once done select the Launch Configuration and Export destination as below screen shot.



Once done got to command prompt and go the location where the jar file is located and run **java -jar test.jar.**

**Continuous testing:** Is running automated tests as part of devops lifecycle. Also it is process of executing automated tests as part of Software Delivery pipeline in order to obtain feedback on the business risks associated with software releases.

Whenever developer pushes feature A to the VCS it is picked up by CI tool and sent to testing server. If everything is fine it is sent to production. After some days if new feature B is pushed to VCS, then first feature A is tested and later feature B.

****