**Locators:**

ID

Name

Class name

link Text

Xpath

Css selector

Html nodes:

1. Tagname
2. Attribute name and value
3. LinkText(Optional)

Example:

**(node Starts)--------**

**<div --------(**Tag name**)**

**class="r-1862ga2 r-1loqt21 r-1enofrn r-tceitz r-u8s1d css-76zvg2" -------(Attribute name = “Value”)**

**dir="auto" -------(Attribute name = “Value”)**

**data-focusable="true" -------(Attribute name = “Value”)**

**tabindex="0" -------(Attribute name = “Value”)**

**style="font-family: inherit; top: 8px; width: calc(100% - 32px);" -------(Attribute name = “Value”)**

**>From< ------------ (Link Text)**

**/div> ---------(node ends)**

**<input**

**autocapitalize="sentences"**

**autocomplete="on"**

**autocorrect="on"**

**class="css-1cwyjr8 r-homxoj r-ubezar r-10paoce r-13qz1uu"**

**dir="auto"**

**spellcheck="false" type="text"**

**data-focusable="true"**

**value=""**

**style="font-family: inherit; text-overflow: ellipsis;">**

**Relative Xpath:**

1. **Using Attribute –**

**//tagname[@Attributename = “value”]**

**//input[@class=”css-1cwyjr8 r-homxoj r-ubezar r-10paoce r-13qz1uu”]**

1. **Using linktext**

**//tagname[text()=’lintext’]**

**//div[text()='Departure Date']**

**Code(selenium/eclipse ide) 🡪 Browser (not possible)**

**Code(selenium/eclipse ide) 🡪 chrome driver (driver )🡪 Browser (possible)**

JavaScript:

1. Date picker
2. Scroll test
3. Highlight

There are two types of web application

Static = applications whose attribute values does not change with refresh

Dynamic = applications whose attribute values do change with refresh (at least one attribute value changes with refresh)

**Regular expressions in xpath:**

1. **contains**
2. **Starts-with**
3. **Ends-with**

id="u\_0\_b\_qR"

id="u\_0\_b\_5p"  
id="u\_0\_b\_QN"

**Contains syntax: //input[contains(@id, 'u\_0\_b\_')]**

**//tagname[@Attributename = “value”]**

//tagname[contains(@attriuteName, ‘PartialValue’)]

**//input[@id=’**u\_0\_b\_qR**’] (fails because dynamic application)**

**//input[contains(@id, ‘**u\_0\_b\_**’)]**

**With text:**

**//tagname[text()=’lintext’]**

**//div[text()='Departure Date']**

**//tagname[contains(text(), ‘Partial LinkText’)]**

**//div[contains(text(), 'Create a new')]**

Create a new account1 **//input[text()=’**Create a new account1**']**

Create a new account4

Create a new account8

Starts-with:

**//input[starts-with(@id, 'u\_0\_b\_')]\**

**//input[contains(@id, '\_0\_b\_')]**

**//div[starts-with(text(), 'Create a new')]**

id="u\_0\_b\_qR"

id="t\_0\_b\_5p"  
id="z\_0\_b\_QN"

ends-with:

***ends-with function is available for Xpath 2.0 and usually browsers lib uses xpath 1.0***

//tagname[ends-with(@attriuteName, ‘PartialValue’)]

//tagname[contains(text(), ‘PartialLinkTextValue’)]

id="u\_0\_b\_qR"

id="r\_1\_b\_qR"

id="z\_6\_b\_qR"

id="j\_p\_b\_qR"

//input[ends-with(@id, ‘\_b\_qR’)]

//input[ends-with(text(), ‘a new account’)]

**’**Create1 a new account**'**

**’**Create2 a new account**'**

**’**Create3 a new account**'**

//input[ends-with(text(), ‘a new account’)]

**Parent**

**//button[@id="search\_btn"]/parent::div/parent::section/parent::div**

**//button[@id="search\_btn"]/../../..**

**Descendant**

**//div[@id="search\_div"]/descendant::span[@id="togglebtn"]**

**//div[@id="search\_div"]/section/div/span**

**23-08-2022**

**Close : It closes the active tab only**

**Quit : It ends the entire session or closes the whole browser.**

**Assertions:**

**Done**

**Dynamic password:**

**String split:**

**24-08-2022**

**Cross browser testing**

**Navigate method – forward, back, refresh**

**Get – driver will wait until all the components of the page are loaded**

**Naviagte.to()- it will not wait for all components. As soon as the next line component appears. It will perform it’s action.**

**getTitle();**

**getCurrentUrl();**

**Important UI components:**

**Dropdowns:**

**Static and dynamic**

**25-08-2022**

custom\_date\_pic required home-date-pick valid

custom\_date\_pic required home-date-pick

style="display: block; opacity: 1;"

style="display: block; opacity: 0.5;"

#### "Cucumber", "Beetroot"

#### Cucumber - 0 index

#### – 1 kfg - 1st index

Synchronization :

**Implicit wait** default polling period(500ms or half second)

**Explicit wait:**

* WebDriverWait default polling period(500ms or half second)
* Fluent wait

Thread.sleep (Selenium method)

Implicit wait

d.manage().timeouts().implicitlyWait(Duration.*ofSeconds*(5));

//no such element: Unable to locate element

features:

It is a global wait

It will act as soon as the element is available

Declaring once is enough

Explicit wait:

Hyderabad = 100 hotels

Bangalore = 500

Mumbai = 1000

Feature :

Is applicable only for one element

Element take maximum time

**Difference between Web Driver wait and fluent wait**

Fluent wait find the web element repeatedly at regular intervals of time until the object is available or until time out.

Webdriver wait find the web element repeatedly at default rate/default polling period(500ms or half second)

Unlike webdriver wait we can build customized wait methods based on requirement

**Actions:**

Mouse hovering

Capital letters

Rightclick

Double click etc

Handling HTTPS certification:

Screen shots

**Broken Links in a webpage** - With the help of HttpURLConnection java clss we will establish connection to the link without directly opening it. And we will get the response code. Finallly if the response code is more than 400, it is broken link;

Eg: 404 = page not found.

**Hard assert:**

Assert.*assertTrue*(true)

Assert.*assertTrue*(false) //fails

If the condition becomes true execution will flow till the end

But if the condition fails, execution ends at that step.

**Soft assert:**

Though the condition fails, execution will run till the end and finally it will display all the results

//softAssert

SoftAssert sa = new SoftAssert();

sa.assertTrue(respCode < 400, "Link is broken = " + url + " because of response code = " + respCode)

Invoking Multiple tabs or Windows

Taking screenshot of a webElement

Finding dimensions(UX)(length x Width) of an element

**TestNG FrameWork:**

Importance of testNG

Installing TestNG plugin

We created two sample test and run the individually

Methods are treated as testcases in testNG Framework

**Lets see about XML file:**

**Test Suite > Test Folders > Test cases**

**Suite > test>class >methods**

**Eg: Loan >(PL, HL, CL) >Test cases**

**We can create separate test depending on requirement and run the required java classes**

**Include tag will help you in running on the mentioned test case**

Exclude: testcase/method mentioned in this tag will not run

We can also run a package lone with package tag

TestNG Annotations:

@Test – It says the method written is a testcase

@BeforeTest – Method below this annotation will be executed first before all the testcases/methods from that test folder.

@AfterTest - Method below this annotation will be executed at last after all the testcases/methods from that test folder.

@Before Suite – This method will execute at first or before all the testcases from the suite

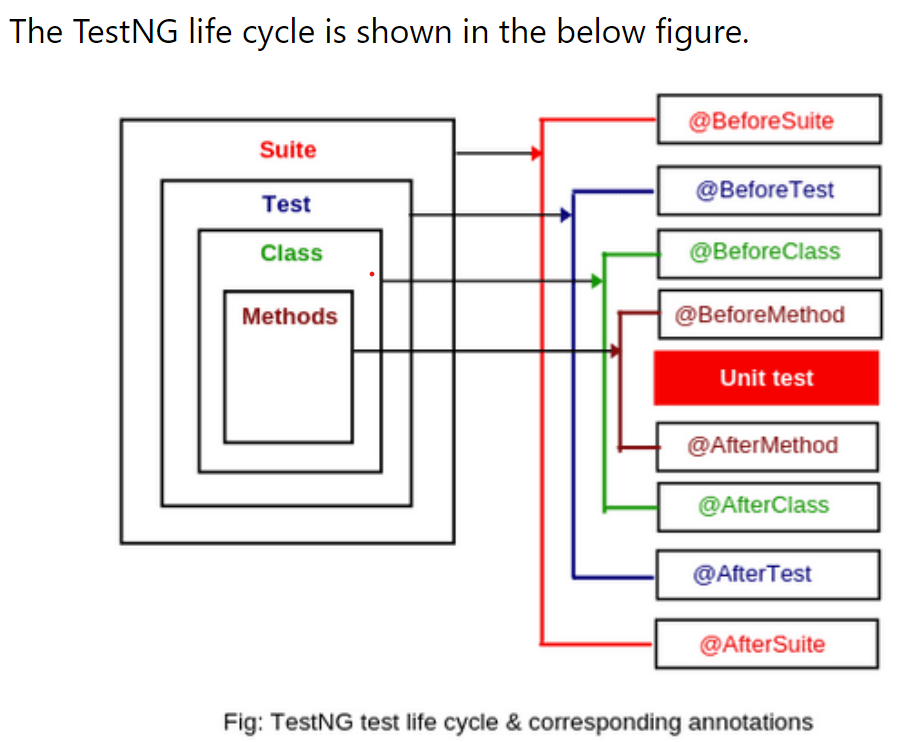
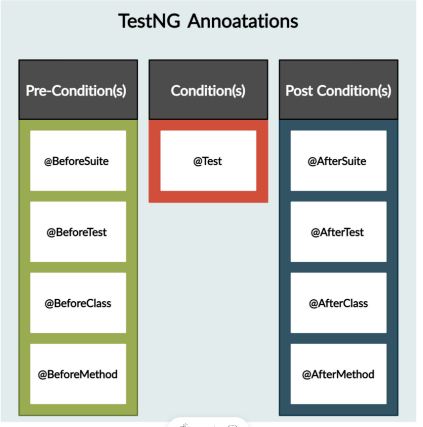
@AfterSuite - This method will execute at last or after all the testcases from the suite

@Before class - This method will be executed before all the testcases(methods) from that class

@AfterClass - This method will be executed after all the testcases(methods) from that class

@Before Method – This method will execute before each testcase/test methods present in that class

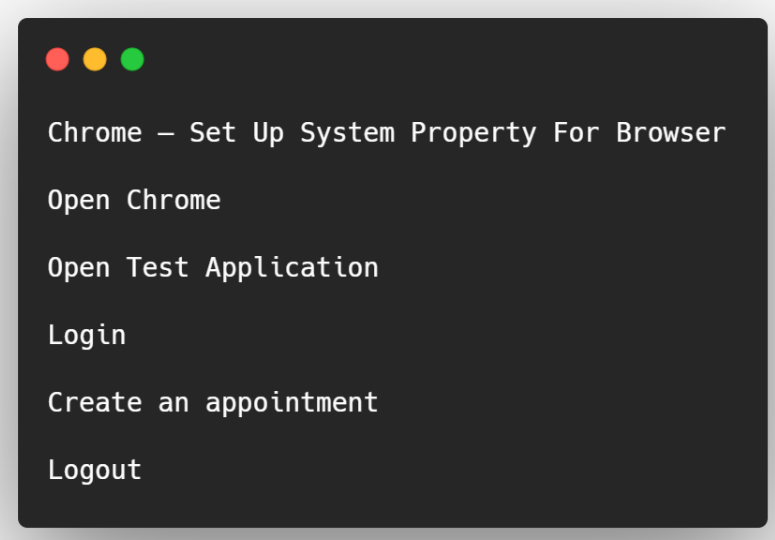
@AfterMethod - This method will execute after each testcase/test methods present in that class

**Example:**

****

**Output:**

****

**What is grouping in testNG?**

**Swapnith = completed**

**Raja Rathod = chromeoptions, ,missed calender concept, screenshot error**

**Prashanth = failed to get text, syntax errors in screenshot code, 5 and 6 line**

**Madhu sudhan= redo**

**Tarun – 0 percent in hospital**

**Venkat kalyan-**

**Gangaraju -**

**Swapnith – 10**

**Raja Rathod – 7**

**Prashanth – 8**

**Madhu sudhan= redo**

**Tarun – 0 percent in hospital**

**Venkat kalyan- redo**

**Gangaraju - redo**

**22-09-2022**

**DependsOnMethods**

**Prioritization**

**Enable –** If you want to skip a testcase from execution **use Enabled = false**

**timeOut -** The maximum number of milliseconds this test should take. If it hasn't returned after this time, it will be marked as a FAIL.

**Parametrization**

**Data providers**

Testng Listeners:

**TestNG:**

**What is testNG**

**TestNG Setup (install plug in, add dependency)**

**TestNG XML File**

**Controlled test execution from xml**

**Include, exclude mechanism**

**Testng Annotation**

**Groups**

**DependsOnMethods**

**Prioritization**

**Enable**

**timeOut**

**Parametrization**

**Data providers**

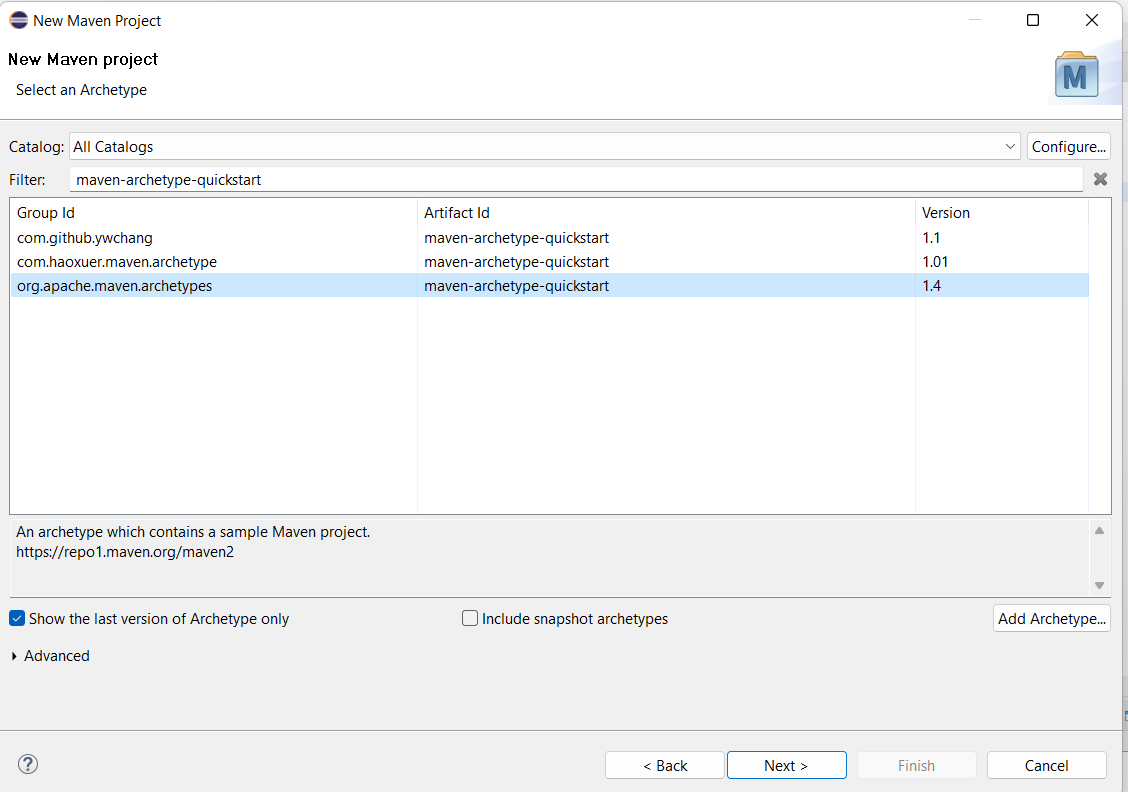
**TestNG Listeners**

If you want to track the test flow status(success failure etc.,)

To print any user defined statement at start, at end, after success, before success

**Framework Development:**

**Webdriver manager**

****

**Page Object Model:**

**Creating page class for each webpage and writing all the xpath or locators related that page in its page class and using them in test class.**

**This way maintenance is reduced and usage becomes user friendly.**

**You should always write the assertions in the test class only(i.e., you should not write any assert method in page class or common methods class)**

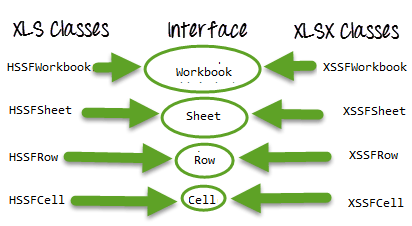
Please use temporary password 'rahulshettyacademy' to Login.

**Please use temporary password - 0**

**rahulshettyacademy - 1**

**to Login. – 2**

## Classes and Interfaces in POI:

Classes and Interfaces in Apache POI

Following is a list of different Java Interfaces and classes in **POI** for reading **XLS** and **XLSX** file-

* **Workbook**: XSSFWorkbook and HSSFWorkbook classes implement this interface.
* **XSSFWorkbook**: Is a class representation of XLSX file.
* **HSSFWorkbook**: Is a class representation of XLS file.
* **Sheet**: XSSFSheet and HSSFSheet classes implement this interface.
* **XSSFSheet**: Is a class representing a sheet in an XLSX file.
* **HSSFSheet**: Is a class representing a sheet in an XLS file.
* **Row**: XSSFRow and HSSFRow classes implement this interface.
* **XSSFRow**: Is a class representing a row in the sheet of XLSX file.
* **HSSFRow**: Is a class representing a row in the sheet of XLS file.
* **Cell**: XSSFCell and HSSFCell classes implement this interface.
* **XSSFCell**: Is a class representing a cell in a row of XLSX file.
* **HSSFCell:** Is a class representing a cell in a row of XLS file.