**What is Framework?**

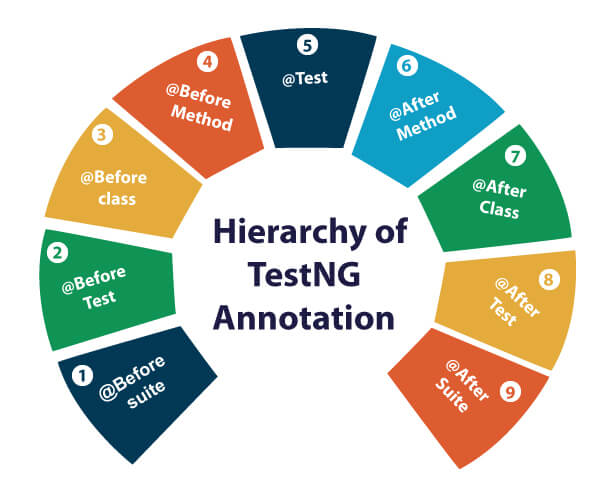
In any Real time project whenever Automation scripts are developed, One should come up with an Execution system called framework to run and maintain Automated tests

**TestNG**

* It is an open source automated Testing framework
* TestNG, NG stands for "Next Generation".
* It is used in Functional,unit ,integration and end to end testing
* TestNG provides you full control over the test cases and the execution of the test cases. Due to this reason, TestNG is also known as a testing framework.

|  |  |
| --- | --- |
| **TestNG Annotation** | **Description** |
| [@BeforeSuite](https://www.javatpoint.com/testng-beforesuite-annotation) | The @BeforeSuite annotated method will run before the execution of all the test methods in the suite. |
| [@AfterSuite](https://www.javatpoint.com/testng-aftersuite-annotation) | The @AfterSuite annotated method will run after the execution of all the test methods in the suite. |
| [@BeforeTest](https://www.javatpoint.com/testng-beforetest-annotation) | The @BeforeTest annotated method will be executed before the execution of all the test methods of available classes belonging to that folder. |
| [@AfterTest](https://www.javatpoint.com/testng-aftertest-annotation) | The @AfterTest annotated method will be executed after the execution of all the test methods of available classes belonging to that folder. |
| [@BeforeClass](https://www.javatpoint.com/testng-beforeclass-annotation) | The @BeforeClass annotated method will be executed before the first method of the current class is invoked. |
| [@AfterClass](https://www.javatpoint.com/testng-afterclass-annotation) | The @AfterClass annotated method will be invoked after the execution of all the test methods of the current class. |
| [@BeforeMethod](https://www.javatpoint.com/testng-beforemethod-annotation) | The @BeforeMethod annotated method will be executed before each test method will run. |
| [@AfterMethod](https://www.javatpoint.com/testng-aftermethod-annotation) | The @AfterMethod annotated method will run after the execution of each test method. |
| [@BeforeGroups](https://www.javatpoint.com/testng-beforegroups-annotation) | The @BeforeGroups annotated method run only once for a group before the execution of all test cases belonging to that group. |
| [@AfterGroups](https://www.javatpoint.com/testng-aftergroups-annotation) | The @AfterGroups annotated method run only once for a group after the execution of all test cases belonging to that group. |

## Hierarchy of the TestNG Annotations:



* @BeforeSuite
* @BeforeTest
* @BeforeClass
* @BeforeMethod
* @Test
* @AfterMethod
* @AfterClass
* @AfterTest
* @AfterSuite

Sample Testcase with @Test annotation

1. **public** **class** test
2. {
3. @Test
4. **public** **void** test1() // First test case.
5. {
6. System.out.println("test1");
7. }
8. @Test
9. **public** **void** test2() // Second test case.
10. {
11. System.out.println("test2");
12. }}

**Priority:**

* Priority is an attribute that helps the **users define the order** in which they want the test cases to be executed.
* @Test(priority =0)

PublicclassPrioritylearn {  
  
  
@Test(priority=0)  
public void startthecar() {  
System.out.println("Start the car");  
}  
  
@Test(priority = 1)  
public void firstgear() {  
System.out.println("firstgear");  
}  
  
@Test(priority = 2)  
public void secondgear() {  
System.out.println("secondgear");  
}  
@Test(priority = 3)  
public void thirdgear() {  
System.out.println("thirdgear");  
}

**Dependencies:**

public class Dependecies {

@Test(enabled = true)

public void highschool()

{

System.out.println("High School");

}

@Test(dependsOnMethods ="highschool")

public void highersecondary()

{

System.out.println("Higher secondary");

}

@Test (dependsOnMethods = "highersecondary")

public void engineering()

{

System.out.println("Engineering");

}

**Dependency in TestNG:**

* It allows a test method (@Test) to depends on another methods.
* Dependency allows us to make one test method dependent on one or multiple other test methods.
* We define by giving @Test(dependsOnMethods ="highschool")

**How to write Test suites (Before suite and after suite)?**

@BeforeSuite

public void launchbrowser ()

{

starttime = System.currentTimeMillis();

System.setProperty("webdriver.chrome.driver", "C:\\Users\\NAZEER\\Desktop\\chromedriver.exe");

driver = new ChromeDriver();

}

@Test

public void openchrome()

{

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

}

@Test

public void bing()

{

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

}

@Test

public void openyahoo()

{

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

}

@AfterSuite

public void closebrowser()

{

driver.quit();

endtime = System.currentTimeMillis();

long totaltime=endtime -starttime;

System.out.println("Totaltime" + totaltime);

}

**How to skip or exclude test cases::**

* **We can skip test cases by using** @Test(enabled =false )

public class Skiptestcase {

@Test(priority=0)

public void startthecar()

{

System.out.println("Car started");

}

@Test(priority=4,enabled =false )

public void turnonthemusic()

{

System.out.println("start music");

}

@Test(priority=1)

public void firstgear()

{

System.out.println("1st gear");

}

@Test(priority=2)

public void secondgear()

{

System.out.println("2nd gear");

}

@Test(priority=3)

public void thirdgear()

{

System.out.println("thirdgear");

}

**Why we need a test suite in XML?**

* Imagine if we have 200 test cases ,we cannot run each class file (which is test cases) for 200 classes
* To make it simpler,we are creating a suite and we are adding test cases in test suite

**How to create XML file?**

New > XML or other then XML> XML File

Name the file name as **TestNG.xml**

1.<suite>

</suite>

Then inside suite add test

<test>

</test>

Then inside test add classes

Add individual class inside the classes

<class>

</class>

**It will be like**

**<suite>**

**<test>**

**<classes>**

**<class>**

**</class>**

**</classes>**

**</test>**

**</suite>**

**How to add class name in xml?**

Packagename.classname like below

<classes>

<class name="testng.Groupsexample"></class>

</classes>

Note – If u select CTRL + on classname it will open as link

**<suite name = “ Test Suite”>**

**<test name = “Test case”>**

**<classes>**

<class name="testng.Groupsexample">

</class>

**</classes>**

**</test>**

**</suite>**

**Practical code:**

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

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

**<suite name ="Testsuite" verbose ="2" parallel="methods" thread-count="2" >**

**<listeners>**

**<listener class-name="testng.Listernersexample"></listener>**

**</listeners>**

**<test name = "Test cases">**

**<parameter name="Name" value="Nazeer"></parameter>**

**<groups>**

**<run>**

**<include name="Moto"></include>**

**<include name="Apple"></include>**

**</run>**

**</groups>**

**<classes>**

**<class name="testng.Groupsexample"></class>**

**</classes>**

**</test>**

**</suite>**

**Assertion:**

public class Assertexample {

String name = "Nazeer";

boolean value = false;

@Test

public void checkequal()

{

//System.out.println("Nazeer");

Assert.assertEquals(name, "Nazeer");

}

@Test

public void checknotequal()

{

//System.out.println("Nazeer");

Assert.assertNotEquals(name, "Nazeers");

}

@Test

public void Asserttrue()

{

//System.out.println("Nazeer");

Assert.assertTrue(value, "This is true");

}

@Test

public void Assertfalse()

{

//System.out.println("Nazeer");

Assert.assertFalse(value,"This is true");

}

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

A screenshot of a computer program

Description automatically generated

< suite name=”My suite” parallel = “tests” thread count=”5”

**How to run parallel tests in TestNG?**

* TestNG provides inbuild support to run TestNG
* We can run parallel tests in suite,test,classes and methods

Example – if u want to run parallel tests in test level means simply put

Parallel = ‘test’

**Listeners in selenium:**

* In Selenium, a listener is an object that “listens” for certain events which occur during the execution of a test script, such as the start or completion of a test case, or the detection of an error

**Types of Listeners in Selenium**

In Selenium, there are two types of listeners.

1. TestNG listeners.
2. WebDriver listeners.

**TestNG listeners**

* TestNG listeners allow to listen to event before and after we execute a test method.
* We can provide additional functionality to TestNG tests, such as capturing screenshots or logging test results.

**Built-in TestNG Listeners**

**IAnnotationTransformer:** It is used to modify test method annotations.

**IExecutionListener:** It is invoked at the beginning and end of a test run.

**ISuiteListener:** This listener is invoked before and after a suite is run.

**ITestListener**: It is invoked before and after a test is run and on test success or failure.

**Use Cases of TestNG Listeners**

* Taking screenshots of test failure.
* Logging test results.
* Capturing performance metrics.
* Analyzing browser logs.

**WebDriver listeners**

* WebDriver listeners **allow you to listen to certain events emitted by the WebDriver**
* Events,such **as navigating to a new URL or clicking on an element asre listened**

**WebDriverEventListener:** This is an interface that contains methods for listening to WebDriver events.

**AbstractWebDriverEventListener:** It is an abstract class that you can subclass to implement the WebDriverEventListener interface.

**EventFiringWebDriver:** This class wraps up a WebDriver instance and enables the firing of WebDriver events.

**Use Cases of WebDriver Listeners**

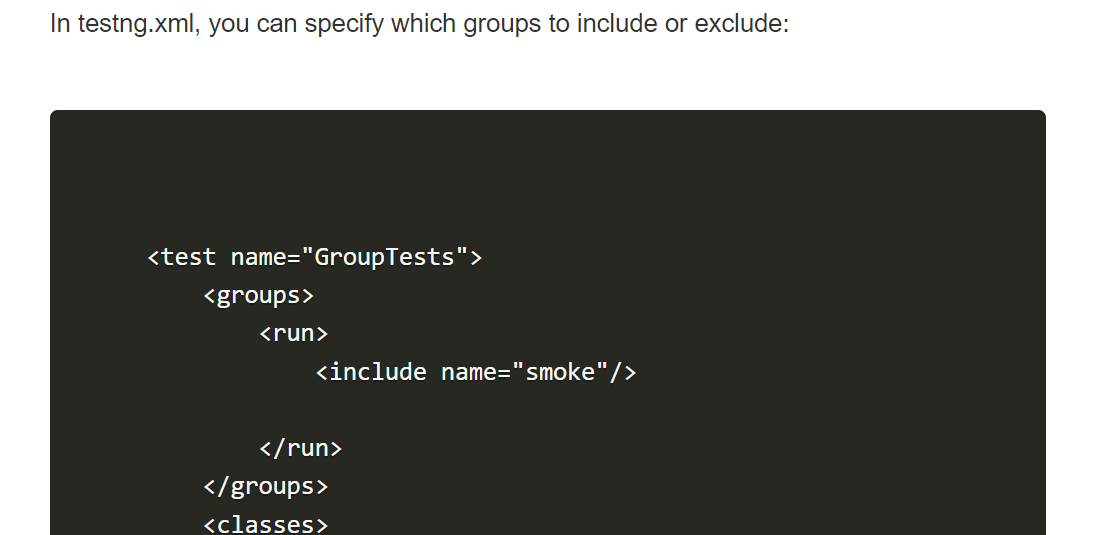
* Taking screenshots of test failure.
* Logging test results.
* Capturing performance metrics.
* Analyzing browser logs.

A close up of a message

Description automatically generated

A screenshot of a computer

Description automatically generated



A close up of text

Description automatically generatedA screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A computer screen shot of a computer code

Description automatically generated

A screen shot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

**Factory Annotations Uses or purpose?**

* Using @Factory annotation ,We can run the same class multiples times By Creating multiple instances of the class

**Why we need Factory annotation?**

* Sometimes we may need to run a same class multiple times with different data values.
* To achieve this we need to add the class inside a suite in the testng XML
* The problem with this approach is that if we get an extra set of data, we will need to add more tests.

**Explain the @Optional annotation.**  
-> @Optional is used to specify a default value for a parameter in case it is not provided in the testng.xml file.  
  
**What is the use of the @BeforeSuite annotation?**  
-> @BeforeSuite is used to define a method that should run before all tests in the suite are executed.  
 **How can you create a custom annotation in TestNG?**  
-> Create a new annotation by defining it with the @interface keyword

**Explain the use of the @AfterSuite annotation.**  
-> @AfterSuite is used to define a method that should run after all tests in the suite have completed.  
  
**How do you include or exclude tests in testng.xml?**  
-> Use the <include> and <exclude> tags within the <methods> tag to specify which tests to include or exclude.

**Try to answer from the below question ,If u don’t know check above for the answers:**

What is Framework?

What is TestNG?

What are all the hierarchy of test annotations?

What is priority ? its uses?

What is dependencies in TESTNG and its uses?

How to write Test suites ?

How to skip or exclude test cases in TestNG test method?

How to skip or exclude test cases in TestNG testing.xml?

Why we need a test suite in XML?

How to create XML file in TESTNG?

How to add class name in xml?

How to run parallel tests in TestNG?

What are all listeners and its types?

What are all the purposes of data annotations in test ng?

How can you perform group wise test execution?

Explain soft assertions in testing?

How to use testng dataproviders with external data sources like csv and excel?

How to pass parameters to a testng method using testng.xml file?

what is factory annotations and its purposes?

Explain the @Optional annotation?

What is the use of the @BeforeSuite annotation?

How can you create a custom annotation in TestNG?

Explain the use of the @AfterSuite annotation.

How do you include or exclude tests in testng.xml?