**Q-1 What is TestNG?**

* Full form of TestNg is “Testing Next Generation”.
* TestNg is a Testing FrameWork used for executing the unit tests in java.
* It is an automated open source testing framework that can be integrated with selenium and capable of making selenium tests easier to understand.
* It gives a lot of options, diff annotations and features to design test cases and to generate HTML report and
* It provide multiple capabilities like assertion, report generation, parallel test execution, etc.

**Q-2 What is the advantage of TestNg over junit?**

**The advantage of TestNg over Junit are :**

1. **It is a powerful tool, it is an open source ,i.e freely available in the market.**
2. It gives a lot of options, diff annotations and features to design test cases and to generate HTML report and
3. It is also available in the form of Jar file.
4. IT is also called Unit testing framework which is applicable in java
5. Test cases in TestNG can be grouped more easily
6. It provides feature to create and execute Parallel tests
7. We can define the priority and sequence in the test cases using TestNG.
8. It defines dependency in which one test case in dependent on another test cases.
9. Data provider feature is available in testing.
10. **It has different assertions that helps in checking the expected and actual results.**
11. **We can run test case using class name / group**

**Q-2)a) What is the Use Of TestNG?**

**It is a very famous framework which is used by the developers to write their unit level test cases using Junit and TestNG.and**

**The Automation engineer they also use TestNG and Integrate with selenium and they write very good test case using TestNG.**

**Q-3 What is testNG.xml file?**

**TestNg.xml file is the heart of execution. This is also called Runner class.**

**This is the main configuration file for the complete test automation. This file is used to execute all the test cases one by one.**

**Q-3/a) What is the use of testing.xml file?**

**The various uses of testing.xml file of TestNG are as follows:**

* Testing.xml file is used for configuring the whole test suite.
* All the test cases in test suite are triggered by testing.xml
* It is used to pass parameters to test scripts
* It is used to support inclusion or exclusion of tests
* It is used to create the test groups and allows to add group dependencies.
* It supports the parallel execution of test cases.

**Q-4 What are the different annotations available in TestNG?**

1. **@BeforeTest 7)@BeforeSuite**
2. **@AfterTest 8)@AfterSuite**
3. **@BeforeClass 9)@BeforeGroups**
4. **@AfterClass 10)@AfterGroups**
5. **@BeforeMethod 11)Test**
6. **@AfterMethod**

**Q-5 What is the sequence of execution of annotations in TestNG?**

**The sequence of execution of annotations is as follows -**

1. **@BeforeSuite [B🡪S.T.C.M.T- --A>M-C-T-S]**
2. **@BeforeTest**
3. **@BeforeClass**
4. **@BeforeMethod**
5. **@Test**
6. **@AfterMethod**
7. **@AfterClass**
8. **@AfterTest**
9. **@AfterSuite**

**Q-6 How to create and Run testing.xml file in TestNG?**

**To create xml file in TestNG, follow the following steps –**



**This will add testing.xml file under java project folder.**

**How to Run- Through Eclipse, Right click on testing.xml file > Click on Run as- TestNG Suite.**

**Q- 7 What is a dependency in TestNG?**

**Dependency is used for some of the methods on which many methods are dependent on.**

**For eg.:For any application, if login page do not work , then it should not test the rest of the test scenarios.**

**In this case we would be using the LoginTest method on which other tests are dependent.**

**@Test(dependsOnMethods=”LoginTest”)**

**Public void SearchPage(){**

**}**

**Since SearchPage is dependent on LoginTest Method, So if LoginTest method fails, then SearchPage method will not get executed.**

**Q-8 What is InvocationCount in TestNG?**

**InvocationCount is used to execute same test case multiple times.**

**For eg.:**

**@Test(invocationCount=10)**

**Public void Login(){**

**}**

**In this case, Login() method will be execute 10 times.**

**Q-9 What is timeOut in TestNG?**

**If we want to terminate any method in the test script which is taking too much time to execute, then we can use “timeOut” attribute in TestNG.Time is provided in miliseconds(ms).**

**For eg.:**

**@Test(timeOut=2000)**

**Public void Login(){**

**}**

**In this case, the Login() method will get terminated in 2000 ms(2 seconds) and the test case gets failed.**

Q-10 How a test can be disabled in TestNG?

To disable any test case in TestNG, we use “enabled” attribute.

For eg.:

@Test(enabled= “false”)

Public void LoginTest(){ }

In this case, LoginTest() method will get disabled.

**Q 11-What are common assertions in TestNG?**

**The common TestNg assertions are :**

* **Assert.assertEquals(String actual, String expected):**

**if both the strings are equal, then only test case will pass.**

* **Assert.assertTrue(condition):**

It accepts a Boolean value. The assertion will pass if condition is true, otherwise it will get fail.

* **Assert.assertFalse(condition):**

It accepts a Boolean value. The assertion will pass if condition is false, otherwise it will get fail.

Q-12 What is assertion and what are the types of asserts in TestNG?

Why we use Assertion?

TestNg Assertion is the verification point and also a validation point that it will check the test cases is getting passed or not(fail).

There are two types of Assertion :

if Verification fails but no stoped. Just it gives the result pass or fail.

1. HardAssert : Hard Assert throws an AssertException immediately when an assert statement fails and test suite continues with next *@Test*

It marks method as fail if assert condition gets failed and the remaining statements inside the method will be aborted. To overcome this we need to use Soft Assert.

**Assert.assertEquals(String actual, String expected)**

1. **SoftAssert: Soft Assert collects errors during *@Test*.**

**Soft Assert** does not throw an exception when an **assert fails** and would continue **with** the next step after the **assert** statement. In case of SoftAssert test case always passed. If I want to expect actual result then I have to use assertAll() in the last line of the statement.

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

**To Create a soft assert, Objects of “softAssert” class is created.**

**SoftAssert softassert= new SoftAssert();**

**Softassert.assertAll();**

Why we use Assertion? Without Assertion we should not write any test cases, otherwise everytime it will be passed or everytime it will enter username ,password and clicking on sign in button but this way we can not verify and we will not know whether it has logged successfully or not. So that’s why we have to put Assertion to verify the condition of the rest run.

What is Assert in TestNG?

Assert is a separate class of TestNG which will allow you to compare values.

There are two types of Assert -1) Hard and 2) Soft Assert.

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

**Q-13 How to set Test case priority in TestNG?**

Priority attribute is used to set the order of execution of test cases.

If priority is not set then the test scripts are executed in alphabetical order.

**For eg.:**

**package testNG;**

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

**public class SetPriority {**

**public class PriorityTestCase {**

**@Test(priority=0)**

**public void testCase1(){**

**System.*out*.println("Test case 1");**

**}**

**@Test(priority=1)**

**public void testCase2(){**

**System.*out*.println("Test case 1");}}}**

**In this case, testCase1 will be executed first and then testCase2 will get executed.**

**Q-14 a) What is parameterized testing in TestNG?**

Parameterized Tests allow developers to run the same test over and over again using different values.

There are two ways to set these parameters:

1. Using testing.xml file
2. Using DataProvider

**===========================================**

**Q-14 b)How can we pass parameter to test script using TestNG?**

**We can pass parameter to test scripts by using @Parameter annotation in test and “parameter” tag in testing.xml file.**

**============================**

**Sample testing.xml:**

**<suite name =”dlaTestSuite”>**

**<test name = “dlaTest”>**

**<parameter name =” browser” value = “browser” >**

**<classes>**

**<class name = “dlaTestFile”/>**

**</classes></test></suite>**

**======================================**

**Sample Test script:**

**Public class ParameterizedTest{**

**@Test**

**@Parameters(“browser”)**

**Public void ParameterTest(String browser){**

**If(browser.equals(“chrome”)){**

**Syso(“Open Chrome browser”)**

**} else if (browser.equals(“firefox”)){**

**Syso(“Open firefox browser”)**

**}**

**}}**

**Q -15 What is Data Provider in TestNG?** **How Data Provider Works in TestNG?**

1. Data provider is an annotation @DataProvider
2. DataProvider is a another way to achieve parameterization in TestNG.
3. Data providers are used for Data driven testing which means same test case can be run with different set of data.
4. It is very powerful feature of TestNG and effectively used during framework development.
5. There are a few points should be considered in case of Data provider—

* It marks a methods for supplying the data to other methods
* Annotated methods return an array of object,i.e Object[][].[Multidimensional Array]
* Data provider can have a name,and it will be used in other methods by using its name.
* Data provider can be implemented in the same class or different class
* A Data provider is a method annotated with @Dataprovider.

The method which is providing the data to the other method in the same test case that method will be called as data provider method.

**Q- How Data Provider Works in TestNG?**

Data provider is an annotation @DataProvider.Data provider will provide the parameterization. It will pick the data from any source then it will transfer the data into the particular test cases and then execute that test case end number of things.

**=======================================**

**Q 16 How can we create data driven framework using TestNG?**

**To create data driven framework, @DataProvider is used in which data is passed to the associated test method and multiple iteration of tests run**

**for different data values passed from @DataProvider method.**

**For eg.:**

**package testNG;**

**package testNG;**

**import org.testng.annotations.DataProvider;**

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

**public class DataProviderExample {**

**@Test(dataProvider="myData1")**

**public void testCase1(String uname, String password ){**

**System.*out*.println("Testcase 1 should now run for 4 sets of users");**

**System.*out*.println(uname);**

**System.*out*.println(password);**

**}**

**@Test(dataProvider="myData1")**

**public void testCase2(String uname, String password ){**

**System.*out*.println("Testcase 2 should now run for 4 sets of users");**

**System.*out*.println(uname);**

**System.*out*.println(password);**

**}**

**@DataProvider(name="myData1")**

**public Object myDataProvider1() {**

**Object [][] obj =new Object[4][2];**

**obj[0][0]="user1";**

**obj[0][1]="password1"; //There 4 types of USer and Each user has one specific password**

**obj[1][0]="user2";**

**obj[1][1]="password2";**

**obj[2][0]="user3";**

**obj[2][1]="password3";**

**obj[3][0]="user4";**

**obj[3][1]="password4";**

**// it will return the object**

**return obj;**

**}}**

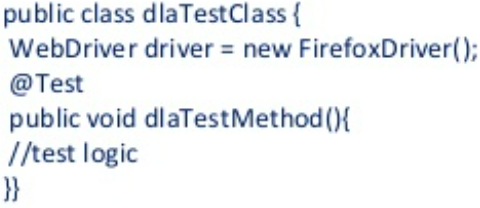
**=======**

**Q-16 ) What is the use of @Listener annotation in TestNG?**









**Q- 17 What is the difference between @Factory and @DataProvider annotation?**

**@Factory method creates instances of test class and run all the test methods in that class with different set of data.**

**@DataProvider is bound to individual test methods and run the specific methods multiple times.**

**Q- 18) How can we run test cases in parallel using TestNG?**

**To run the tests in parallel in TestNG, we have to add these two key value pairs in suite-**

* **Parallel = “{methods/tests/classes}”**
* **Thread-count =”{number of thread you want to run simultaneously}”**

**For eg.:**

**<suite name> = “DLATestSuite” parallel= “methods” thread-count=”5”>**

**Q- 19 How can we make sure a test method runs even if the test methods or groups on which it depends fail or get skipped?**

To run the test method even if test methods or groups on which it depends get fail or skipped, we use “alwaysRun” attribute of @Test annotation.

For eg.:

**@Test**

**Public void parentTest(){**

**Assert.Fail(“Failed Tesst”);**

**}**

**@Test(dependsOnMethods={“parentTest”},alwaysRun= true)**

**Public void DependentTest(){**

**Syso(“Test DLA”);**

**}**

**Q- 20 How to handle exceptions in TestNG?**

To handle exception in methods we can mention the exception in @Test annotation so that the test case does not fail.

For eg.:If a test method is expected to have “numberFormatException” exception, then the test case will fail because of this exception if no try catch block is specified.

But this can be handled in TestNG by using “expectedException” attribute:

**@Test(expectedException=numberFormatException.class)**

**After this the test case will run without failing.**

**==========================X=====================================**

**Q-21 What is Exception test in TestNG?**

TestNG provides an option for tracing the exception handling of code. You can verify whether a code throws the expected exception or not.

ExpectedException parameter is used along with the @Test annotation. If any kind of exception is coming it has to be ignored, otherwise test case will be failed.

Example**–@Test ( expectedExceptio=ns{IOException.class, NullPointerException.class })**

@Test(expectedExceptions=ElementNotFoundException.class)

Public void testPrintMessage(){

Syso(“Inside testPrintMessage”)

}

**Listener in TestNG**

listeners as a name say they keep listening to your code.

The concept of Listener in TestNG is that it keeps listening to your test cases and it gets activated when a particular condition is met.

Like a test case can start, a test case will finish ,a test case will pass and a test case will fail, So listeners will include the code .

Example if you test case is failing. Then you need a screenshot for that to get into reports or to debugger. So you will include the screenshot code in the listener section.

So listeners keep listening to your tests and they get activated

when a certain condition is met.

//Search -testNG listener -Documentation of TestNg

There are several interfaces that allow you to modify TestNG's behavior. These interfaces are broadly called "TestNG Listeners"

* IAnnotationTransformer ([doc](https://testng.org/doc/documentation-main.html#annotationtransformers), [javadoc](https://jitpack.io/com/github/cbeust/testng/master/javadoc/org/testng/IAnnotationTransformer.html))
* IAnnotationTransformer2 ([doc](https://testng.org/doc/documentation-main.html#annotationtransformers), [javadoc](https://jitpack.io/com/github/cbeust/testng/master/javadoc/org/testng/IAnnotationTransformer2.html))
* IHookable ([doc](https://testng.org/doc/documentation-main.html#ihookable), [javadoc](https://jitpack.io/com/github/cbeust/testng/master/javadoc/org/testng/IHookable.html))
* IInvokedMethodListener (doc, [javadoc](https://jitpack.io/com/github/cbeust/testng/master/javadoc/org/testng/IInvokedMethodListener.html))
* IMethodInterceptor ([doc](https://testng.org/doc/documentation-main.html#methodinterceptors), [javadoc](https://jitpack.io/com/github/cbeust/testng/master/javadoc/org/testng/IMethodInterceptor.html))
* IReporter ([doc](https://testng.org/doc/documentation-main.html#logging-reporters), [javadoc](https://jitpack.io/com/github/cbeust/testng/master/javadoc/org/testng/IReporter.html))
* ISuiteListener (doc, [javadoc](https://jitpack.io/com/github/cbeust/testng/master/javadoc/org/testng/ISuiteListener.html))
* ITestListener ([doc](https://testng.org/doc/documentation-main.html#logging-listeners), [javadoc](https://jitpack.io/com/github/cbeust/testng/master/javadoc/org/testng/ITestListener.html))

When you implement one of these interfaces, you can let TestNG know about it with either of the following ways:

* [Using -listener on the command line.](https://testng.org/doc/documentation-main.html#running-testng)
* [Using <listeners> with ant.](https://testng.org/doc/ant.html)
* Using <listeners> in your testng.xml file.
* Using the @Listeners annotation on any of your test classes.
* Using ServiceLoader.

**5.18.1 - Specifying listeners with testng.xml or in Java**

Here is how you can define listeners in your testng.xml file:

|  |  |
| --- | --- |
| <suite> | |
|  |

|  |
| --- |
| <listeners> |
| <listener class-name="com.example.MyListener" /> | |

|  |  |
| --- | --- |
| <listener class-name="com.example.MyMethodInterceptor" /> | |
| </listeners> |

|  |
| --- |
|  |
| ... | |

Or if you prefer to define these listeners in Java:

|  |  |
| --- | --- |
| @Listeners({ com.example.MyListener.class, com.example.MyMethodInterceptor.class }) | |
| public class MyTest { |

|  |  |
| --- | --- |
| // ... | |
| } |

Steps –to be considered to use ITestListener interface in java and listener in TestNG :--

* We are implementing the listener interface and we are overriding

the method and if I want to say like I want my this message when my test case passed and I want the my test case failed message when my test case is failed .

-we have created a listener main class and we implemented this interface.

-In this interface we have implemented all the methods and we have overridden two methods-- one is when my test is successful, print me that my test case passed. And another one is when my test is failed, print me that my test case is failed and

-then we created a class where we wrote a basic test wherein it gets passed and we are trying to run it from all XML file.

-In the XML file we made it clear that we are telling this file that we are implementing listeners and by implementing listener we want to execute a class called listeners class.

-then run this testing suite.