130. Why TestNG and Its Advantages

TestNG is one of the most important component when you are actually starting and developing the framework from scratch level. because you can have the control on all your test cases and the execution with the help of testNG.

So that's why this could be also called as a testing framework.

Ex:

 if you want to run a test case, “A” so before that as a prerequisite, you need to run some multiple test cases before you begin a test case “A”

So you can set and map like that with the help of testNG so that those prerequisite test cases will run first and then only it will trigger a test case “A”. so in that way you can control your test cases.

Ex :

you have some 500 test cases and your client asked you to run only 40 Test cases

which are identified as a smoke.

Now, within one single click, only those 40 test cases will be picked out of those 500 and will be triggered and you will get need results.  so like that, you can control all your test cases .

. TestNG is a testing framework and it's also one of the main building component when you are dealing with frameworks.

How can we install and configure this TestNG into our eclipse? –( TestNG.org) official website.

Eclipse->Help->eclipse market place - ( here you can install all plug ins related to the eclipse ) – search for TestNG enter – TestNG for eclipse click on install - check all the options, what it has given and click on confirm – accept agreement – click on finish – click on restart anyway – click on restart now.

Whether the TestNG is present or not in the eclipse – in eclipse top – click on windows->preferences>search TestNg if you see there then it is installed in ecplise

131. Running testcases in TestNG with out Java compiler

Eclipse – right click – new – project – java project – next – give the project name – next – finish – no.

Create one new class – give the class name – check the public static void main – finish

your test will run on the Java compiler because of this public static void main.

So now TestNG can act as a Java compiler for you. so if you can incorporate the TestNG framework and TestNG library into your test cases, so you need not depend upon the Java compiler. TestNG itself acts as a Java compiler and executes all your test cases. ( Without this(public static void main), you cannot compile or run any Java program )

So if you want to simply execute one step like system.out.println, so if you do this

outside of the public static void main, it will not work because you need this main method in Java to execute.

So here we are saying that we are using the TestNG library instead of a Java compiler.

so that doesn't mean that you need not have Java to run all your tests.

TestNG itself, again, depends upon Java in the backend, So you still need Java, but from the front end, you need not write all those Java things.

So a very simple way to execute this from TestNG is, you know, create one method first.

So TestNG needs all it's test steps' execution in a method.

So this is my simple method,

@Test

Public.void Demo()

{

System.out.println(“hello”)

}

this is a method which has a public access modifier, and "void" is written type.

So in this method, if you have defined this hello program step in particular method.

And now if you execute, it still will not run, because you have still not given enough information for TestNG to run this.

This is a test, "@Test" is one of the annotations in TestNG's library

which will recognize the method which immediately followed after this annotation

as a test case When you write this “test” whatever the method follows after this @Test annotation will be treated as a test case by TestNG, And it runs that program, even if you don't provide public static void main.

Earlier, when there is public static void main, you should always run as a Java application. As you remove that public static void main, you no more have an option of Java. Now it says as a TestNG test, because that is an annotation of TestNG

and we are running this test case as a TestNG library. TestNG, it has some custom libraries inside their framework, which will neatly provide information like "passed."

Demo is the test case name

Interview que :

1. how to run tests with TestNG. – you need to have @Test annotation, followed by method.

132. Importance of xml file in Testng configuration

 TestNG, there is a feature where you can develop/define multiple test cases in one single class file.

Whereas in Java, if you want to write one test you will define public static void main

and you print this and again if you want one more test you will create another new class.

how do you differentiate between one test and another?

So if you write a test annotation then immediately TestNG read this as a 2nd test case,

you have to wrap it in a method.That's a TestNG rule.

if you want to create a TestNG XML file for your project, simply right click on your project – TestNG->Convert to TestNG – click next – finish. It will create a ready made XML for you.

Click on the xml file. so you see it here, it says a default suite in console.

So TestNG always want a higher level as a suite where it should have a neat name,

for us it's like test suite(loan department) in general,what test suite you are running in this projectthat information first you need to provide this is starting tag and this is ending tag.

inside this suite there will be test folders.

if I want to speak in layman terms, this test folder is something where kind of module.

So now you are working in one loan project, 3 types of loan – personal,home,car

There are three different type of modules in one loan company.

So for every module have its own test cases,all test cases you can wrap it as one test name. So this test is nothing but a folder name or a module name.

Okay collection of test folders is nothing but a test suite. inside suite you have to define test folder. inside this test folder we are defining test case class where exactly all the test cases are stored.

Why classes and inside again class? Because tomorrow if there are multiple classes

which belongs to this personal loan.

 inside classes you will define multiple class files which actually comes under this suite.

(Package.classname). ( Testsuite->Test Folder(module0->Testcases )

So this all comes under a test module. A suite can have multiple modules.

how do you run all these test cases? So you have a requirement now

that go and run all your personal loan test cases.

 all your personal loan test cases you will tell to your TestNG XML file to run

right click of the xml file->run as->testNG -> all the mentioned classes in the xml file should run.

Note : All test cases, we are actually writing in this java class files.

133. Prioritizing the Testcases with TestNG

Just create multiple classes and write some simple code and add the name in the xml file ( you need to add the name like (Package.classname) ) and run

@Test

Public void webloginHomeloan()

{

//selenium

Println(“webloginHome”);

}

@Test

Public void Mobileloginhomeloan()

{

//Appium

Println(“MobileloginHome”);

}

@Test

Public Void APIHomeloan()

{

//Rest API automation

Println(“APIHome”);

}

Client wants to run only for the Home loan test cases when they ask you to run all the home loan test cases you should not go and figure out what are there

and run each and every test from here. There should be some mechanism within one single click. Only home loan test cases should run.

Go to the xml file -> you can add one more test folder <test> name = “homeloan“ </test>

Inside this we can add multiple classes.

If you run the xml file all the classes should execute.

But I want to run only the homeloan class either you could simply comment out the steps.

For example, if client told you that you need not have separate test cases for home loan

you can include that as your personal loan. So in that case you need not always define every time this test shell. If you feel that they belong to same model then add that class in the specific class only, rather than again creating one more class.

134. Include and Exclude Mechanism to control testcases

 client told that he don't want to execute mobile test cases in Car Loan application.

So run remaining things except mobile test cases in Car Loan,

Open any of the class and write few of the test cases for car loan ex ( 5 test cases )

In the 5 testcase there is one mobile test case as well

Public void Mobilelogincarloan()

{

//Appium

Println(“Mobilelogincar”);

}

you have to skip only one test case in that class file.

you are writing this method's tag as a parent of this day3 class.

So all these methods, what you write here, will refer to this day3 class only.

Note : Every method in testNG is treated as a test case. So there is inside methods again, there is one more tag called exclude provide the method name in b/w the exclude tags in xml file ( <exclude name = “Mobilelogincarloan” /> ) save the file and run the xml file.

if you want to add it back, just remove the added code, that's all, it'll come back.

So with the help of exclude tag, you can skip methods from your test.

Note : You can also get a control on running specific methods from Test case.

You can use either include or exclude.

Total test cases – 100 – if client asks exclude 98 test cases run only 2 test cases then we need to use include in the xml file

135. Executing the Testcases at Package level with regex

how to deal with regular expressions in these methods.

Client req is exclude all the mobile test cases means total test case 100 mobile test case 40.

Out of 100, if you are asked to remove only 30 mobile, you cannot just go and always write 30 mobile exclude name, exclude name, 30 times.

Now, if it is mobile, all the 30 test cases should start with mobile test word, okay?

It could be again anything mobile login, mobile sign in, mobile sign out, or mobile paying, everything. But that module should have one unique thing

in all those test case names. In framework this is most important,

You should have some naming convention which is consistent across that module.

So in that way, when you develop a framework like this you can easily write a code

to eliminate all those within a single step. Now, if you have written all those 30 mobile test cases with a keyword mobile which starts at the beginning for that test case name, okay, you can simply avoid that by writing mobile.\*, .star stands for anything after mobile. So that means here we were asking to exclude any test method

which starts with a name called mobile. So, when it starts execution, it'll find 30 test cases which starts with a mobile name as a test case, and it'll eliminate all the 30 in a single shot. Just run. ( <exclude name = “Mobile.\* /> ) - reglar expression

Ex :

<class name="test.day3">

<methods>

<exclude name="MobileloginHomeloan"></exclude>

</methods>

</class>

there is one more feature where you can run all this test cases from package level. You can even simply say, "run this package test," All test cases will run automatically.

Note : you should create this Test NG XML in your project level only.

<test name = “personal loan”>

<packages>

<package name = “test”/>

</packages>

If you added package name directly in the xml file and the classes under the package all should be run in one single shot.

when you're running regression test, you should run no matter mobile API test, REST API, or Selenium, you have to run all the test cases, So in that case, just put a package and trigger all the test cases on single shot, and they'll run that’s the power of xml file

136. TestNg Annotations part – 1

New req - when you run your automation test you will fill all the data and you submit again, you will use the same data for your next subsequent automation run.

So at that time, as data is already present in the system, it'll throw an error saying that error data exists. Now, so to overcome this issue, you want to delete all the data from the database

So you can tell to your test NG, that first clean everything by executing one particular block and then start executing all the test cases which are present in this personal loan.

@ BeforeTest – It will execute first

@BeforeTest - whatever method you define here that will execute first before any test

belonging to that test folder. So when you say before test, so before executing any class files from this test folder first, that particular (Before test )annotation will be executed.

So as I said, if you want to clean up the data or if you want to delete some records

in database before you freshly enroll in a loan application you could use that code in before test. All these belongs to one test folder you given give before test so that will executed first, no matter in which class you place.

@AfterTest – It will execute last

@AfterTest - wherever you define that inside this block, you can define anywhere,

but based upon your execution it will execute only at the end.after test annotation will be executed after all the classes completed in that specific test folder. It will not wait for other test folder or sperate module.

137. TestNG Annotations part – 2 ( @Before & after suite we will mention in the TC’s )

( Instead of @Test we need to use @BeforeSuite & @AfterSuite )

Suite – which have all whole TestNg xml file

So all these four folders come under suite and when you said @BeforeSuite it will execute before it executes any test in this XML file.

so when you say @AfterSuite, it will only execute once all the tests completed and all the test folders because suite is parent to the test folder

Earlier we said only before and after test, which belongs to that specific test folders.

This suite is generally used if you want to set some global environment variables to your framework.

If you have different url and QA environment to run your test cases for every release you will get one new enviornment so that before each and every test case starts

execution, you will make sure that you load all your environment variables which needed for the framework and then you will trigger your test cases.

similarly you have one more annotation like before method and after method. that is specific to your class file.that is not specific to Xml file.

when you said before method and after method, like each and every test is nothing but a method. every method is treated as a test case in TestNG.

when you say @BeforeMethod in that particular classfile, so before executing each and every test block that method will execute.

So if there are four test cases, so four times before method will be executed after before executing each and every block/test case.

Interview que : what is different between before the method and before the test?

 before test, it depends upon testNg xml file based upon the test folder scope before method is nothing but specific to classfile And it executes before each and every method.

Ex : caches, cookies

138. Usage of Groups functionality in TestNG

@BeforeClass annotation, it is self explanatory, so when you say before class and

the scope of it is to that specific class only, so before executing anything in that class that before class will be executed.

@BeforeClass and @BeforeMethod these two annotations are class level .

Beforetest and beforesuite is testNG xml level  based upon the folders and based upon the testsuites.

in testNG order of your test method execution depends upon the alphabetical order,

so based upon the method, alphabetical order it defines and execute the test.

New req – client came and pic 4 test cases he want on every new release build he wants to run only 4 Tc’s. he want these 4 TC’s results .

Client randomly picked 4 Tc’s in different class files . how do you run only those four?

By picking it from different classes?

so you have to give one tag to those 4 Tc’s which you feel that

you need to trigger as part of a smoke test, for example, for smoke We need to pic 4 Tc’s out of 100 TC’s.

( @Test(groups={“Smoke”}) - so you want to trigger those 4 test cases only out of everything, so in that case You have given a tag called groups.

Smoke is a tag given to those TC’s now from our testNG XML file, we will tell that go and execute all the test cases which have a tag called smoke.

it simply scans all your Tc’s. And whenever it finds smoke tag, it will execute that

xml file - before classes and between test and classes. You want to add groups.

<groups>

<run>

<include name = “Smoke.\*”/>

</run>

</groups>

139. Annotations helper attributes with examples

So if you want to run all the test cases except smoke, then you can simply say exclude this group and run the test cases. – in output except those 4 TC’s remaining all should be executed.

So in this way you can either include a group or exclude a group.

… we saw that as alphabetic order, this test methods are getting executed in this file.

we have a requirement that this specific Tc’s should execute first before executing this test.

So in TestNG you have an dependency called depend on methods attribute.

( @Test(dependsOnMethods={“weblogincarloan”}) ) -

In this way you can put interdependency between one test and another.

Similarly, you can define you can depend upon multiple methods as well.

( @Test(dependsOnMethods={“weblogincarloan”,”Mobilelogincarloan”}) ) before execute the below method first it will execute the depend methods.

All these are helper attributes to annotation depend on methods is one of the helper attribute, and at the same time there is one more helper attribute called enabled.

when you are running test, if you know that one of the application or one of the flow is not working, there is a bug in that particular execution.

So when you are signing from your mobile, there is some bug which is already reported and you are aware of it. So while executing you have to skip that particular test.

How do you do that?

( @Test(enabled=false ) )- TestNG will safely skip this test case while executing.

If you put enabled it true, it will on again.

Interview que : I know that there are few errors in my application and I don't want my framework to trigger those. What you will do then?

I will enable that to false with the help of testing helper attribute.

Similarly, there is one more helper attribute, which is timeout.

So you may feel that one of your test case is taking long time to execute.

And test cases are failing because of that.

@Test(timeout=4000) - So it will not fail till 40s of your execution.

140. Parameterising from TestNG xml file

So for web testing, the URL must be the common.

You have to land on that web page and then perform different test cases on different functionalities. So in this way you will have some global environment variables which are common across your tests.

how to drive the most common variables from XML file to your test cases and you can avoid hard coding like that. so you should not hard code like urls, API keys and package details. Those should be driven from one centralized place and that's the main rule of framework.

The main details should always come from one external file or one centralized file.

Note : 1) xml file - if you feel that those values are common across all test folders, you can define those parameters here after the suit.

<parameter name=”URL” value=”rahahulshettyacademy.com”/>

If you defining it globally, it is applicable to all your test cases

in your package.

@parameters({”URL”})

@Test

Public void weblogincarloan(String urlname )

{

Println (weblogincarloan)

Println (urlname)

}

Once you add this in any of the class you need to run the xml file not the specifc class.

we have successfully pulled out that value into our test case.

2) xml file - if you feel that those are specific to this particular test folder, then you can define inside the test folder.

<parameter name=”URL” value=”rahahulshettyacademy.com”/>

So this URL parameter will be valid only to this test folder class files only.

If you try to write something out of this, then this doesn't work.

what time I have to use test level parameterization and what level I have to

use suit level parameterization. ?

assume that you have loan department.com is your website that you can give in suite level. So in that personal loan login, you can go from that particular home page if you want to go to that personal loan login, you can write in this test.

So all these tests when try to hit that URL, it will hit that personal login.com, whatever you define in test level

141. DataProvider Annotation -Parameterizing Testcases – 1

142. DataProvider Annotation -Parameterizing Testcases-2

So not only single value, you can actually pass multiple values from your XML file.

<parameter name=”URL” value=”rahahulshettyacademy.com”/>

<parameter name=”APIKey/username” value=”123456”/>

In class – one testcase

@parameters({”URL”,” APIKey/username”})

@Test

Public void weblogincarloan(String urlname, String Key )

{

Println (weblogincarloan)

Println (urlname)

Println(key)

}

there are some tests which could be specific to that method level.

In that case, you cannot define everything in this XML file and make it too complicated.

For that you have one more annotation.

Global environment Variables can be incorporated through selenium.

 parameterizing with multiple datasets by running tests with multiple combinations.

For that, in testNG you can achieve it with the help of data provider annotation.

Data provider is the annotation.

Here you can define your class name and anything.

Just define a method.

Get data is a method which actually gives you the entire data for your test case to run.

So without writing that test three times, you can define only once and you can run your test three times with the three different data with the help of data provider.

Basically you are defining a multi dimensional object array.

3 -is how many combination we are trying( 3 is rows )

2 – how many values we are using for each combination (username password 2values)

So basically you will fill the values, all the data setup, you will give the data and simply

you will send this data to this test method to run with all combinations.

each data set can be treated as one row, and columns in the row are nothing but values

for that combination.

Now our requirement is we have to trigger this test case with first this data.

Whole test case should run with this data first.

After test case should again invoke with this data and it has to run.

And again, test case should invoke again with this data.

So that means three times your test case should run with three different set data, which we have configured

like this.

Interview que : how do you achieve parameterization?

you can do it in two ways.

You can drive the data from testing the XML file and you can also drive the data and you can parametrized it using data provider annotation.

Ex: one specific Test case

@Test(dataProvider=”getData”)

Public void mobilesignoutcarloan(String username,String password )

{

//Appium

println(“Mobile SIGNOUT”);

println(username);

Println(password);

}

@DataProvider

Public object[][] getData()

{

//1st combination -username password – good credit history – row

// 2nd username password – no credit history

//3rd fraudulent credit History

object[][] data = new object[3][2];

//1st set

data[0][0] =”firstsetusername”;

data[0][1]=”firstsetpassword”;

//columns in the row are nothing but values for that particular combination(row)

//2nd set

data[1][0] =”secondsetusername”;

data[1][1]=”secondpassword”;

//3rd set

data[2][0] =”thirdsetusername”;

data[2][1]=”thirdpassword”;

return data;

143. Listeners Interface in TestNG framework

What are testng listeners? -  there are some listeners where it will activate after your test or before your test.

New req : If any Tc fails automatically you have to take a screenshot. while writing the code you cannot write that step after each and every step because you need the

screenshot only when the test fails.

So testNG have some intelligent mechanism with the help of listeners so that it will reroute your execution to one specific block when test case is failed.

so when it reroutes to some specific block in that block, we will write screenshot code so that whenever test case fails, it always go to that block and screenshot code will execute and screenshot will be captured.

It listens to your execution results.

Create one new class.

So now to implement this laziness in your testng, there is an interface called //ITest Listener.

So I test listener is an interface which have all the methods for this test listeners.

so you need to implement that interface to use this listeners concept in your framework.

Public class Listeners implements ITestListener{

If you write like this, that means we are going to implement the all the methods present in this interface.

If you move your cursor on the above line Listeners word it says add unimplemented methods import it you will get all the methods which testNG exposes.

Or

Right click(on the Listeners class ) -> go to source-> click on overide/implement methods -> select the check boxes for the ITest listener (make sure all check box inside it should be checked )->click on oK.

Note : When you are running your test XML, you have to tell where this listeners class is located. If you don't tell, it will not go and redirect to this particular Java class. so that's mandatory.

after the suit and before the test folders

<listeners>

<listener class-name=”test.Listeners”/>

</listeners>

144. Running Tests in parallel and generating Reports

how to know for which test case we are failing?

Okay, so that particular failed test is reaching this point, but we should also know for which test case we were actually reaching this block.

If you want to know that that all the details of that test case, when it got failed, will get captured in this result object.

if you want to print name of the test case which actually got failed, then simply result dot get name.

So in this way you can actually track that particular test case name, which is failing.

Que : when you catch listeners, how do you know the test case name?

We can have a class called ITestResult, and this class actually describes the result of a test. With a ITestresult class.

running test Parallelly with testNG.?

there is a feature in testing XML file to trigger all these tests together.

If you want to run two test folders together,

<suite name=”Loan Department” parallel=”tests” thread-count=”2”> ( 2means 2 testfolders run paralleley )

Add parallel on suite level and you can trigger your test, XML file and all the results.

Whatever you saw in a sequential now will run parallelly.

And once you go inside test if you want to run this classes also parallelly. Then you can use parallel equal to classes. You can just put this parallel equals to classes and give the thread count.

<test name=”personal loan parallel=”classes” thread-count=”2”>

we have covered each and every topic which we require to develop a robust framework with the help of testNG.

if you are interested to see the results after you run your test cases.so refresh your project. And there is a test output folder here.

Once you refresh the project, you will get it. Open the test output folder and there is an index dot HTML file here. right click properties. Take the entire path of that HTML, go back to your browser and paste that URL. You will see the results for all your test case execution.

TestNG Tutorial

1. Importance of TestNG
2. Installation and configuration in eclipse
3. Testng XML file Importance
4. Controlling Testcase executions from TestNG XML files
5. Include Exclude Mechanisim
6. Understanding TestNG Annotations
7. Groups
8. Data Provider Annotation
9. Annotation helper Attributes
10. Parameterization is TestNG xml file
11. Running Tests in parallel with TestNG
12. TestNG Listeners