TestNG - unit test framework

→ developers writing their code at unit level test cases, by using JUnit or TestNG.

→ TestNG ⇒ testing New Generation framework

→ A.E also uses testNG for integrate with selenium & write good selenium test cases.

→ It is very powerful Tool, it gives lot of features & options to design test cases & to generate HTML reports.

→ purpose ⇒ Design test cases in a proper systematic way

→ It is open source, free

→ It is available in the form of Jar files

→ It is also called as Java unit testing framework.

each & every language have their own testing framework.

Adv- ① generate Html reports
② It gives diff. annotation
③ priorities (can give priorities) / Also can design sequence of test case.
④ dependency feature
↳ one test case depend on other
⑤ grouping
⑥ Data provider → execute test case multiple time with diff. set of data.
for eg. login funct ⇒ you have (want to pass same username password multiple times.

→ It is also called as TDD.
TDD → Test Driven Development
↳ Along with whatever they are writing a test cases (whatever the design a code design test cases) in Agile methodology
→ so testNG will help you to design a test cases in a systematic way what are the prerequisite is there what need to do after writing test cas.

How to Install TestNG in eclipse

→ Google → type → testNG plugin in eclipse
http://beust.com/eclipse/

→ help → click on install new s/w → paste
→ search → select last opt

→ add testNG library

TestNG - SDET

→ TestNG is a testing framework inspired from JUnit & NUnit but introducing some new functionalities that make it more powerful & easier to use.

→ TestNG is designed to cover all categories of tests: unit, functional, end-to-end, integration, etc.

Adv ⇒ ① can manages test suites & test cases

② Helps in prioritizing of tests

③ Helps in grouping of tests

④ parallel execution

⑤ Reporting

In TestNG ⇒ TestSuite → Test cases → Test steps

In JUnit → Package → Classes → methods

Package → ClassABC{}

Classes: every class contain one test case

methods: steps included in diff. type of methods

@Test → TestNG recognize that it is test method

⇒ If you want to consider method as test method then add annotation (@Test)

practical ⇒ o/p ⇒ TestNG by default execute test method by alphabetical order.

If you want to control order, you must set priority.

download → testngorg
↓
mvn repository → search testng → jar download
here maven dependency is also present.

→ TestNG XML file & report
→ Annotations
→ Prioritizing tests
→ Dependency tests
→ Grouping tests
→ Assertions
→ parameterization
→ Data Provider
→ parallel testing
→ Listeners

① What is TestNG xml file & usage (testsuit)
② How to create TestNG xml ?
③ How to run tests from xml file
④ TestNG Report.

TestNG ~~suit~~ xml ⇒ Using TestNG ~~suit~~ xml file we can execute multiple test cases as a suit.
→ we can create multiple suit & execute parallely.
→ we can also parameter from xml file to test cases.

① we need to install testNG ~~xml~~ plugin in eclipse

add jar files
write @test at method

⇓

O/P → create test-output folder

implementing this testsuit with the help of .xml file.

1 class = 1 test case → add this in xml file

if we add that all test case then we will get O/P of all test cases

Assert.fail() → fail test case

↑

import it

How to create xml file ⇒ right click on project → New → other → xml → next → give name

→ Wh

of the

java co

→ Anno

with

Executi

① Befor

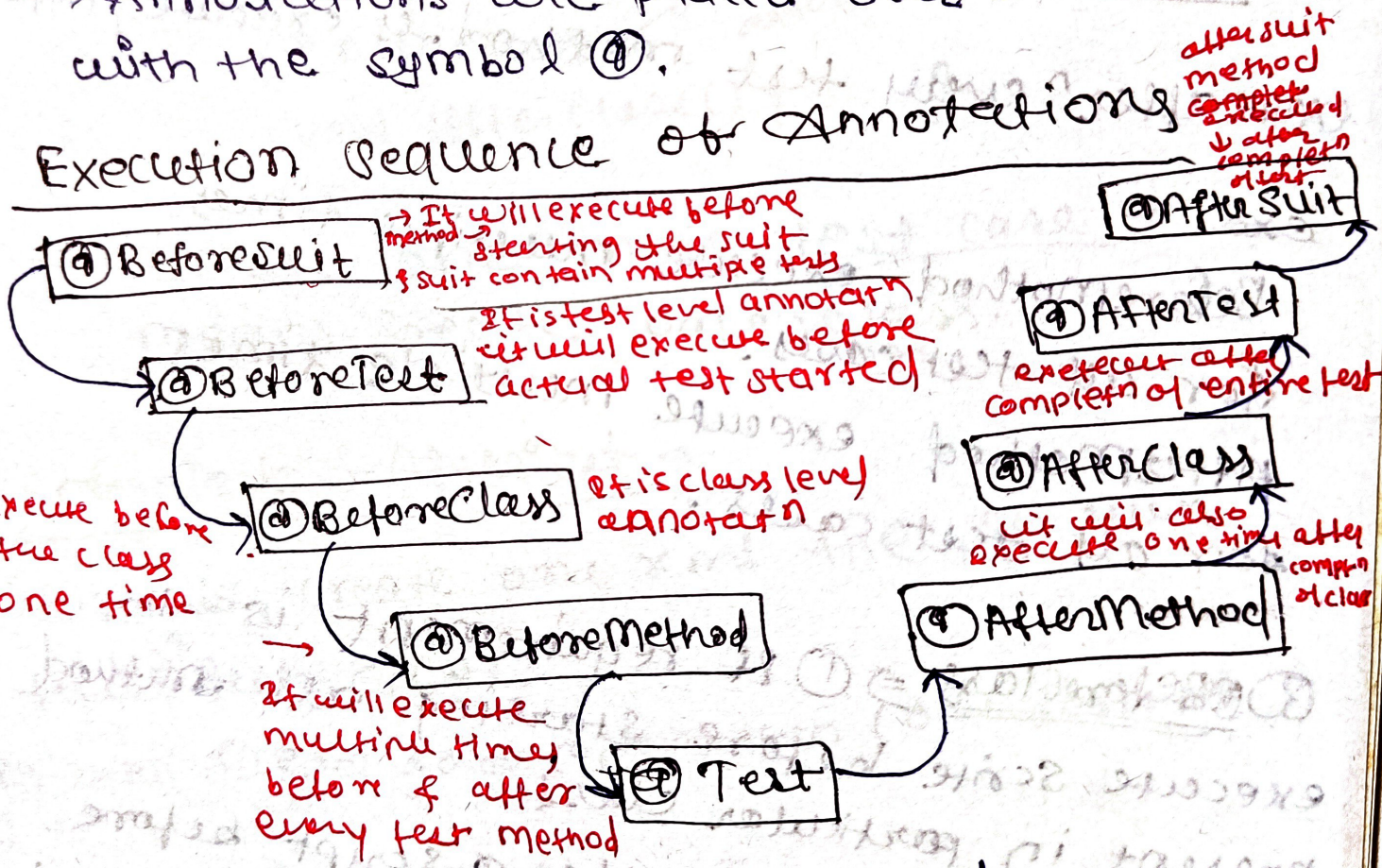
execute before the class one time

Annotations in TestNG

→ We can control the sequence & priority of the methods which allows to execute java code before and after a certain point.

→ Annotations are placed over the method with the symbol @.

Execution sequence of Annotations



→ ~~every method recognize as test~~ every test method we have to specify with @Test annotation & then testNG will recognize that method as test method.

① @BeforeMethod ⇒

~~this annot~~ method having this annotation will execute before the rest of the method (every test method)

② @AfterMethod ⇒

execute after every test method is execute / complete

eg:

BeforeMethod execute multiple times for all test cases.

AfterMethod execute multiple times for all test cases.

③ @BeforeClass ⇒ ① If requirement is to execute script before starting all method present in particular class.

If you want to execute a script before starting a particular class & also a event to execute after completion of entire class

eg:- ~~pdb~~

```
public class TC1 {
    @BeforeClass  → execute only one time before execute a class
    void BeforeClass()
    {
        P ——
    }
    @AfterClass  → execute only one time after execute a class.
    void AfterClass() { }
```

⇒ create one more test case ⇒
means create one more class.

BeforeTest & AfterTest ⇒

for this, create one xml file, To run this test cases

In xml file ⇒ go to code & then code format.

every suit contains
- tests
  - every test contain
  - multiple class
    - method

class → method

• xml file

```
<test name = "annotest" ">
    <classes>
        <class name="TC1">
            <methods>
                <include name="test1"/>
                <include name="test2"/>
            </methods>
        </class>
        <class name="TC2">
            <methods>
                <include name="test3"/>
                <include name="test4"/>
            </methods>
        </class>
    </classes>
```

⇒ If I want to execute a before actual test is started.

Before class means test → this is hierarchie

Before test

↓ this method can either create in TC1 or TC2 but in runtime this method will execute all before all the method (before class)

afterTest => execute after all the classes or test will execute

execute Before Starting the suit & after Completion the suit.

Before suit ⇒ before all test
Before test ⇒ before all class
Before class ⇒ before all method
Before method ⇒ before all test /script

can create in TC1 or TC2 but at run time (run xml file)

This annotations used for control your test execution.

Suit contain multiple test
test contain multiple class
class contain multiple method

@BeforeTest ^ Before starting the test (from xml file)
@AfterTest v After completion of test (from xml file)

suit level ⇒

If I want to execute the before starting the suit & I want to execute the after completn of suit. that can be achieved by using Before suit & after suit.

where that Method will create → either in TC1 or in TC2 → classes

@beforesuit
void beforesuit()
{ SOPL("before test suit");
}

// execute before actual (before test) test is started

aftrsuit → execut after actual test is completed. (after test)

after completn of test.

Prioritizing Tests

→ By default testng execute a test case by alphabetically order. of method name.

→ But we can prioritize them.

for eg. if there is funcn like login, logout, register

then we need to do first register then login then logout

so, we will give priority 1 to register & so on.

eg. @Test(priority=1) can give random no. also but whichever no. come first that method will execute 1st.

⇒ But I want some method then to disable

~~that~~ set ⇒ enabled = false then that method will not participate in execution. for eg. @Test (priority=3, enabled=false) by default it enabled=true.

Dependency Tests in TestNG

Sometimes, when we create multiple test methods in test case, there will be some dependency.

For eg. Some test case depends on some other test case & the o/p is depends on some other o/p

How to manage this dependency

① create class & method

```
class DependencyBy{
    @Test
    void startCar() { }
    @Test(dependsOnMethods={"startCar"})
    void DriveCar(){ }
    @Test(dependsOnMethods={"DriveCar"}
    void stopCar() { }
    @Test
    void parkCar() { }
```

Assert.fail();
→ Intentionally will fail the test case.

For eg. In above before start car we can't drive car so we need to first start car. So add (dependsOnMethods={"startCar"}) then startcar will execute only then drivecar will execute.

If 1st method fail then 2nd method will fail: because it depends on start.

If dependency method get failed still if you want to execute the method forcefully then add AlwaysRun property

eg. @Test(dependsOnMethods={".driverCon","storCon"}, alwaysRun=true)

after adding alwaysRun, it will always execute the methods even dependency get fail.

Grouping Tests

→ We have to group some of test cases comes under sanity or some of them are regression some of them are functional.

we have to categorize them in multiple groups & whenever we run set of test cases only those set of test cases

→ for achieving such type of functionality called grouping functionality.

Syntax→ @Test(groups={"sanity"})

Now if you want to run particular group of test cases then create one xml file in that class & write code there.

```
<test name="groupingtest">
    <classes>
        <class name="GroupingExample"/>
    </classes>
</test>
```

add before classes tag for adding groups / executegroups

```
<groups>
    <run>
        <include name="sanity"/>   //only sanity group test cases will execute
        <include name="regression"/>
    </run>
</groups>
```

other than sanity rest of group if you want to execute then use "exclude"

```
<groups>
    <run>
        <exclude name="sanity"/>
    </run>
</groups>
```

Now if you want to run particular group of test cases then create one xml file in that class & write code there.

```
<test name="groupingtest">
    <classes>
        <class name="GroupingExample"/>
    </classes>
</test>
```

add before classes tag for adding groups / execute groups

```
<groups>
    <run>
        <include name="sanity"/>      //only sanity group test cases will execute
        <include name="regression"/>
    </run>
</groups>
```

other than sanity test of group if you want to execute then use "exclude"

```
<groups>
    <run>
        <exclude name="sanity"/>
    </run>
</groups>
```

Assertions in TestNG

→ when Automating test cases, we need to put some verification point or object point so we can put verification point by using Assertion in TestNG.

→ There are diff kinds of Assertions we have ① Assert.assertTrue() & Assert.assertfalse

② Assert.assertEquals()

→ Assert class is available, by using that class we can use this method.

verify logo is display or not

```
public class AssertExample{
  WebDriver driver;
  @BeforeClass
  void setup()
  { System.setProperty("      ");
    driver = new ChromeDriver();
    driver.get("url");
  }
  void logotest()
  {
WebElement logo = driver.findElement(By.xpath("//  --)"));
    Assert.assertTrue(logo.isDisplayed());
```

```
void homepageTitle()
{
String title = driver.getTitle();
Assert.assertEquals("orangehrm", title);
```

Parameters in TestNG

→ It is required when you don't want to hard Code the value in code. its not good practise as well.

→ if any scenario, run with diff. set of data need to be this can be achieved by parameterization.

→ By using testNG xml file we can pass parameters to your test method.

eg.
```
class param
@Parameter({"browser", "url"})   coming from testNG xml file
void setup(String browser, String app)   Values of this parameter stored in this variable.
{
if(browser.equalsIgnoreCase(anotherString: "chrome"))
launch chrome driver
else if(browser.equalsIgnoreCase(anotherString: "firefox"))
launch firefox driver
}
```

depends on passed parameter from xml file we need to specify parameter name according to that specify the no. of variables as a parameter

xml file ⇒ create it in that class
right click → create testng.xml

→ reformat class (right click file)

→ <test name = "~~class name~~" > provide some name
<classes>
<class name = "parameterExam" />
</classes>
</test>

→ We can pass parameter at diff. level
class level, test level, suit level etc.

At test level
<parameter name = "browser" value = "chrome" />
<parameter name = "url" value = "http---" />

run xml file

Data Providers in TestNG

→ Apart from parameters, there is another way to achieve parameterization by using Data provider in testNG.

→ The method which is providing data to the other methods in a same test case will be called as Data Provider method. used annotation called as data provider annotation.

→ Apart from Parameters, there is another way to achieve parameterization which is by using DataProvider in TestNG.

→ Data Providers are used for data driven testing which means same test case can be run with different set of data. It is very powerful feature of TestNG & effectively used during framework development.

→ It marks a methods for supplying the data to other methods.

→ Annoted methods return an array of Object i.e. object[][].

→ DataProvider can have a name & it will be used in other methods by using its name.

→ DataProvider can be implemented in the same class or different class.

→ A Data Provider is a method annoted with @ DataProvider.

① create data provide method which is having return type object[][] is array

② ~~when~~ we must provide name parameter to DataProvider annotation

Because we the same name, we have to refer this data provider name in another test method.

→ for one ~~test~~ test case, we can have many no of data provider method.

```
Public class DataProvider {
@dataProvider(name="LoginDataProvider")
public object[][] getData()
{
    object[][] data={{"abc@gmail.com","abc"},
{"xyz@gmail.com","xyz"}};
return data;
}

~~public class DataProvider{~~

@Test(dataProvider="LoginDataProvider")
public void loginTest(String email, String pwd)
{
    SOP(email+" "+pwd);
```

कहाँ से data आता है वो बताता है

this method can execute multiple times
No need to write loop बनाना data रहेगा उतना बार execute होगा।

Based on no. of i/p passed by data provider method login test will execute those no. of times

* If dataprovider method present in another class :-

then add that class name

```
class DataProvider

  @Test(dataProvider = "LoginDataProvider",
add this → dataProviderClass = customDataProvider.class)
  public void loginTest(String email, String pwd)
  {
     sout("email + "  " + pwd);
  }
```

TestNG Listeners

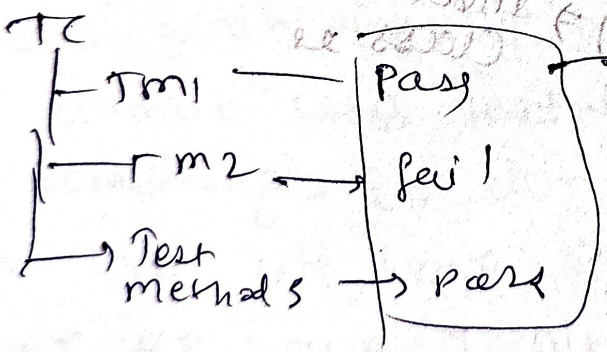
TC contains many test methods

TC
→ TM1
→ TM2

} This many test methods can Pass or fail.

Based on status of this test methods, I want to perform some certain post activity.

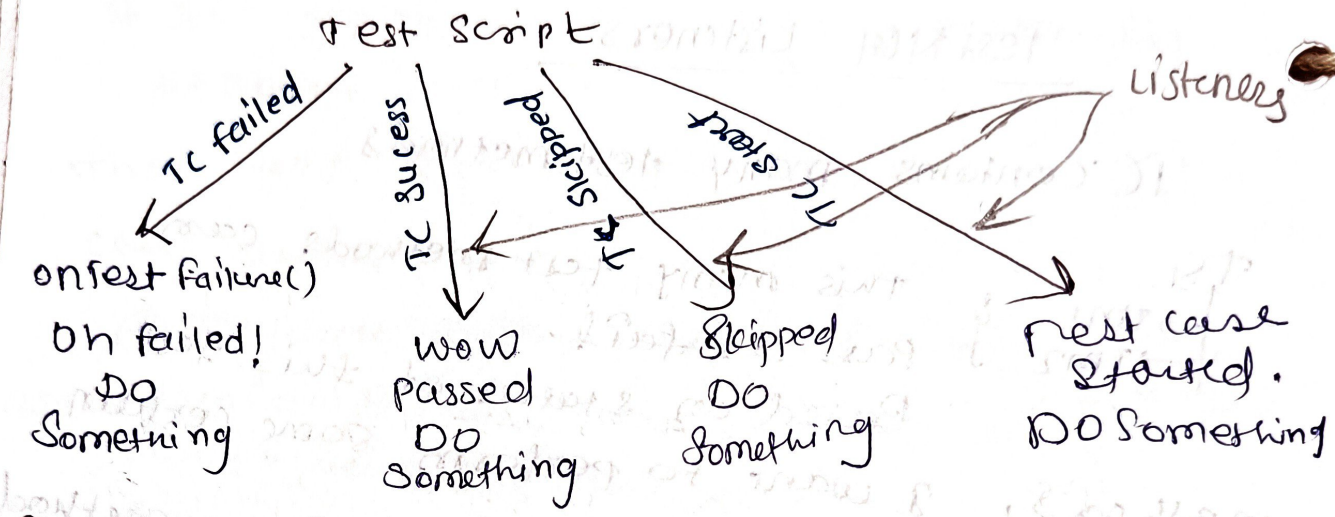
perform some action based on that test method that is post activity.



perform action on them

To implement this testNG provided Listeners.

→ Listeners is kind of Interface or set of methods & those methods can be automatically triggered based upon the result of your test method



① Types of Listeners ⇒

① ITestListeners :-

Method	Description
① onStart(ITestContext arg0)	⇒ Invoked after the test Class is

TestNG Interview questions

① What is TestNG?

TestNG is a testing framework.
The full form is test Next Generation
It is a framework which is used by developers to write their unit level test cases as well as integration level test cases, that's why we are using in selenium also because in selenium we are using Java and its dedicatedly designed for Java Java unit testing framework.
I cannot use testNG for dot net or ruby or something. In dot net we have Nunit In PHP we have PHP unit, But in Java either we have Junit or testNG unit
So testNG is a Java testing framework which is used to write your unit level test cases. In selenium you can write functional test cases as well.
So, you can write or you can design the test cases in a very systematic way. You can achieve so many things that parameterizat & lot of features are available.

But testNG is framework, available in the form of Jar files.

Que What are the advantage of TestNG?

① TestNG provides parallel execution of test methods.
In testNG.xml file you can define one parallel attribute is equal to methods or class & you can define that thread count also so we can do parallel execution.
for eg. you have 1000 test cases, you can execute with three diff. process or four diff. browsers so that we can reduce the execute time, so parallel execution we can do it allows different defined a dependency of one test method over other method case again.

② It allows to define dependency of one test method over other method.
eg. I have one login method, I have one homepage method if login method is failed then there is no point to execute your homepage method, so there is no point to execute your homepage method & registration method & search method if login itself getting failed so we can create dependency.

betn these two methods. these two test methods

⑤ It allows to assign priority to test methods. we can define priority by using priority keyword as 0, 1, 2, 3 with @ test annotatn

@ Test(priority=1)

④ It allows grouping of test methods into test groups.
With the help of group keyword, we can do grouping also.

⑤ It has support for parameterizing test cases using @ Parameters annotation.

⑥ It allows data driven testing using @DataProvider annotation.

what is purpose of data provider? or how will you perform the data driven?
We fetch the data from excel file & with the help of data provider we will execute that same test case again & again with the diff. set of data, so we can achieve parameterizatn, we can achieve data driven testing with the help of @dataprovider

⑦ It has diff. assertions that helps in checking the expected & actual results.

⑧ Detailed (HTML) reports.

It gives you index.html file as well as proper xml file also our HTML file also & for failure-cases also you can generate a seperate HTML file & it can be easily integrated with Jenkins which will give you the proper report in end up in the form of HTML You can generate your Customized report also with the help of TestNG listeners

⑨ TestNG listeners to generate a custom logs to create a custom reports

Qu ⇒ What are the annotations available in TestNG?
we had different annotations prerequisite annotatn as well as post condn annotatn
So these are prerequisite like annotatn which is starting with before it means these are prerequisites @beforetest, @beforemethod, @beforeclass, ... & post condn means @after post means later after @aftertest, @afterclass, @aftermethod

@parameter → to define parameters in xml file & we can take those parameters with the help of @ paramers

@ DataProvider

Qu ⇒ what is structure of testng.xml file.

First it will start with a suit then test then we have classes inside the class we have class then we have methods.

Before suit, we have to define xml schema i.e. xml version=1.0 & the doctype suit. this two lines are imp if you don't write this two lines it will give you a xml parsing exception.

after this suit level, if you want to define listener then you can define listerns

```
<test>
<suit>
<class>
<methods>
<classes>
```

correct order ⇒ one

```
<suit>
<test>
<classes>
<class>
<methods>
```

Qu ⇒ How to create testng.xml file?

right click on project → new → file → give this name

How to run → right click on xml file & run

before this you need to download plugins from marketplace testng properties libraries are added.

(IMP) What is importance of testNG.xml file?

It is a heart of the execution.

this is also called Runner class.

So there any project like selenium project or any API automat^n project if you are using testNG. testNG is the main configuration file for complete test automat^n that

At suit level we can do any level testing. at test level.

framework

So it is heart of execut^n, you can define all the configurat^n w.r. to your execution

In selenium TestNG project, we use testng.xml file to configure the complete test suit in a single file. Some of the features are as follows.

i) testNG.xml file allows to include or exclude the execution of test methods & test groups.

If you don't want to execute any test methods or test case then you can ignore it from a testNG class

ii) It allows to pass parameters to the test cases.

you can pass the parameter.

iii) Allows to add group dependencies

iv) Allows to add priorities to the test cases.

But we don't required to write priorities every time.

v) Allows to configure parallel execut^n of test cases.

vi) Allows to parameterize the test cases.

by using @Parameterize annotat^n we can define.

Que How to pass parameter through testNG.xml file to a test case? (imp)

We could define the parameters in the testNG.xml file & then reference those parameters in the source files.

Create a java test class, say, parameterizedTest.java & add a test method say parameterizedTest() to the test class. this method takes a string as i/p parameter. Add the annotation @parameter("browser") to this method.

```
public class ParameterizedTest{
    @Test
    @Parameters("browser")
    public void ParameterizedTest(String browser){
        if(browser.equals("firefox")){
            S.O.P("open firefox brow")
        }else if(browser.equals("chrome")){
            S.O.P("open Chrome Driver");
    }
}
```

The parameter would be passed a value from testNG.xml, which we will see in the next step. we could set the parameter using the below syntax in the testng.xml file.

```
<parameter name="browser" value="firefox"/>
```

here the name attribute represents the parameter name & value representing value of that Parameter.

Qu ⇒ what is TestNG Assert & list out common TestNG Assertions?

TestNG Asserts help us to verify the condition of the test in the middle of the test run.

Based on the TestNG Assertions, we can consider a sucessfeel test only if it is completed the test run without throwing any exception.

testNG assertion is verificatn validatn point. that it will check the test case is getting passed or ~~not~~ failed

We can put assertion anywhere like at middle of test case, at end of the test case.

without assertion we should not write any test case. otherwise everytime it will pass, it will enter password, name.

→ there are many assertions methods are available in Assert class. Assert. press ctrl+space

eg. assertEqual(String actual, String expected)
assertEqual(——— // ———, String msg)
assertEquals(boolean actual, boolean expected)
assertTrue(condition)
assertfalse(condition)

Qu ⇒ What is soft Assert ? how will you put soft asserts?

Soft Assert collects errors during @Test.

Soft Assert does not throw an exception when an assert fails & would continue with the next step after the assert statement.

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

It will collect all the errors.

I have this test case T1

```
T1 {
1 → passe → sendkeys()
2 → HA → passed → then only go in 3rd line else → terminate test case
3 → SA → passed
4 → SA → failed → jump to 5
5 → SA → passed
6 → SA → failed, program is over
}
```

SoftAssert.assertAll(); → it will sort all the assert what are diff errors are there

So, the property of Soft Assertion is that, if Soft Assert is getting passed fine it will jump to next line but soft assert is getting failed still it will jump to next line.

But the property of hard assertion is like that if it failed it will not jump to next line immediately. it will terminated. If hard assertion failed it will not jump to next line. ~~font~~(terminated)

Que ⇒ what is hard assertion?

Hard Assertion throws an AssertException immediately when an assert statement fails & test suit continues with next @Test.

Que ⇒ what is exception test in TestNG?

TestNG gives an option for tracing the Exceptn handling of code. you can verify whether a code throws the expected exception or not. The expected exception to validate while running the test case is mentioned using the expected Exceptions attribute value along with @Test annotatn.

```
@Test
ti(expectedException = ElementNotFound(Exceptn){ }
```

If you know that, that some kind of exceptn may occur or any exceptn is coming & if you have defined that yes we are expecting this then it will not mark your test a judge field.

Qu ⇒ how to set test case priority in TestNG?

We use priority attribute to the @Test annotation. In case priority is not set then the test scripts execute in alphabetical order.

Qu ⇒ what is parameterization testing in TestNG?

Parameterized tests allow developers to run the same test over & over again using diff. values.

There are two ways to set these parameters :-

(i) using testNG.xml

(ii) using Data Providers

Data provider → it will pick the data from any source & then it will transfer the data into that particular test case & then execute that particular test case & no. of things for eg. do for login, you have 10 username 10 password so no need to create 10 login methods. Single log in test case having username password then we can maintain this username password in excel file & then you can pass

all these username pass to this test case with the help of data providers.
How to use it → we have enter a data provider a notatn

Que ⇒ How can we create data driven framework using TestNG?
By using @DataProvider annotation, we can create Data Driven framework.

Que ⇒ How to define grouping?
By using @groups = {"~~group~~smoke", "functional"}

Imp → How to run test cases in parallel using TestNG?
We can use "parallel" attribute in ~~testNG~~ testNG.xml to accomplish parallel test execution in TestNG. The parallel attribute of suite tag can accept four values.
tests → All the test cases inside <test> tag of testng.xml file will run parallel
classes → All the test cases inside a java class will run parallel
methods → All the methods with @Test annotatn will execute parallel.

instances → Test cases in same instance will execute parallel but two methods of two diff instances will run in different thread.

<Suit name="softwaretestingmaterial" parallel="methods" thread-count=5>

↑ generally give methods

it means at a time you can execute 4 or can launch five diff browsers

Que ⇒ How to exclude a particular test method from a test case execution?

By adding the exclude tag in the testng.xml

```
<classes>
<class name="TestCaseName">
<methods>
<exclude name="TestMethodNameToExclude"/>
</methods>
</class>
</classes>
```

Que ⇒ How to exclude a particular test group from a test case execution?

```
<groups>
<run>
<exclude name="TestGroupNameToExclude"/>
</run>
</groups>
```

Qu ⇒ How to disable a test case in TestNG?

To disable the test case we use the parameter enabled = false to the @Test annotation.

@Test(enabled = false)

Que ⇒ How to skip a @Test method from execution in TestNG?

By using throw new SkipException()

Once SkipException() thrown, remaining part of that test method will not be executed & control will goes directly to next test method execution.

throw new SkipException("Skipping - This is not ready for testing");

Que ⇒ How to ignore test case in TestNG?

To ignore the test case we use the parameter enabled = false to the @Test annotation.

@Test(enabled = false)

Qu ⇒ How TestNG allows to state dependencies?

TestNG allows two ways to declare the dependencies.

Using attributes dependsOnMethods @Test annotation

using attributes dependsOnGroups in @Test annotation.

Que What are the diff ways to produce reports for TestNG results?

TestNG offers 2 ways to produce a report.

Listeners implement the interface:
org.testng.ITestListener & are notified in real time of when a test starts, passes, fails etc.

Reporters implement the interface
org.testng.IReporter & are notified when all the suites have been run by TestNG.

IReporter instance receives a list of objects that describes the entire test run.

we can generate very good report in testNG.

Que ⇒ what is the used of @Listener annotation in TestNG?

TestNG listeners are used to configure reports & logging. One of the most widely used listeners in testNG is ITestListener interface. It has methods like onTestStart, onTestSuccess, onTestFailure, onTestSkipped etc. We should implement the interface creating a listener class of our own.

Next we should add the listerners annotation (@Listeners) in the class which was created.

The perpose of listeners to generate the reports & to generate the logs.

Que ⇒ How to write regular expression in testNG.xml file to search @Test methods containing "smoke" keyword?

Regular expression to find @Test methods containing keyword "smoke" is as mentioned below.

```
<methods>
    <include name = ".*smoke.*"/>
</methods>
```

if there 100 methods are there having word smoke all those test cases will be executed.

Que ⇒ what is the time unit we specify in test suits & test cases?

We specify the time unit in test suites & test cases es in milliseconds.

Que ⇒ List out various ways in which TestNG can be invoked?

TestNG can be invoked in the following ways

i) Using Eclipse IDE
ii) Using ant build tool (maven)
iii) from the command line
iv) Using IntelliJ IDE

Que ⇒ How to run testNG using command prompt?

we have to use test.jar file that we need in which class you want to execute & you have to go to that particular directory

C:\cd Testfolder

C:\test

Java C:\\testng.jar test.java

@ Data Provider ⇒

To provide test data to testcase

@ Factory ⇒ execute all test methods which are available in particular test class using a seperate instance of the respective class with diff. set of data.