**How to run TestNG?**

1. You need to have @Test annotation followed by a method ()
2. We can define multiple tests from a single class
3. We can modularize the test cases based upon our functionality and trigger them accordingly
4. TestNG Annotations:
   1. @BeforeTest: at start of test shell execution
   2. @AfterTest: at end of test shell execution
   3. @BeforeSuite: at start of test framework/suite execution
   4. @AfterSuite: at end of test framework/suite execution
   5. @BeforeMethod: This will execute this method first out of all methods in the class
   6. @AfterMethod: This will execute this method at last out of all methods in the class
5. Groups: Groups can be used to run specific test cases (exclude/include command) out of hundreds of cases. We have to mention group after *@Test(groups={“GroupName”})*

<**groups**>

<!-- Used to run a particular group of test cases that belong to mentioned tag

We can trigger execution of selected test cases out of our test suite -->

<**run**>

<**exclude** name=*"Smoke"*/>

</**run**>

</**groups**>

1. dependsOnMethods:   
     
   *@Test*(dependsOnMethods= {"WebLoginCar"})  
     
   public class Day5 {

*@Test*

public void WebLoginCar()

{

System.***out***.println("Web Login Car");

}

*@Test*(dependsOnMethods= {"WebLoginCar"})

public void MobileLoginCar()

{

System.***out***.println("Mobile Login Car");

}

*@Test*(dependsOnMethods= {"WebLoginCar"})

public void APILoginCar()

{

System.***out***.println("API Login Car");

}

*@Test*(dependsOnMethods= {"WebLoginCar"})

public void APILoginCar2()

{

System.***out***.println("API Login Car2");

}

*@Test*(dependsOnMethods= {"WebLoginCar"})

public void APILoginCar3()

{

System.***out***.println("API Login Car3");

}

*@Test*(dependsOnMethods= {"WebLoginCar"})

public void APILoginCar4()

{

System.***out***.println("API Login Car4");

}

}

1. enabled  
   *@Test*(enabled=false) //TestNG will safely skip this case while executing. This could be due to known bug in the module

public void APILoginCar3()

{

System.***out***.println("API Login Car3");

}

1. timeOut: Can be used when the particular execution is taking time. So, we say hold on for mentioned time

*@Test*(timeOut=4000) //Will not fail until 40seconds before throwing and error

//Can be used when the particular execution is taking time. So, we say hold on for mentioned time

public void APILoginCar4()

{

System.***out***.println("API Login Car4");

}

1. Parameters:   
   Java code:

*@Parameters*({"URL"}) //Value from TestNG XML file lands here

*@Test*

public void WebLoginCar(String urlname) //Value that has landed above in parameters will be caught here

//Should be executed from XML file

{

System.***out***.println("Web Login Car");

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

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

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

}  
XML File:

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

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

<**suite** name=*"Suite"*>

<!-- In one suite we can have multiple projects or test shell base on our requirement -->

<**parameter** name=*"URL"* value =*"leadsquared.com"*/>

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

<!-- One test shell is like in loan department- car, home will be different test

So, in one test shell we can have multiple class, where each class can be the .java file

we have to pass package.filename as parameter

-->

<**classes**>

A parameter can be at:

1. Suite level
2. Test Level

Note: A suite/test can pass one or more parameter

Parameterization cab happen from the XML file or from java class

From the class it happens with help of @Dateprovider

*@Test*(dataProvider="getData")

public void day2(String username, String password)

{

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

System.***out***.println("Data provider");

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

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

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

}

*@DataProvider*

public Object[][] getData()

{

//1st combination - user name and password - good credit history

//2nd - user name and password -no credit history

//2nd - user name and password -false credit history

//Creating multi-dimensional array

Object[][] data = new Object[3][2]; //Array with 3 rows for 3 combination and 2 columns for 2 parameters used

data[0][0]="firstrowusername";

data[0][1]="password";

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

//2nd set

data[1][0]="secondrowusername";

data[1][1]="secondpassword";

//3nd set

data[2][0]="secondrowusername";

data[2][1]="secondpassword";

return data; //must return data

}

1. Parallel:  
   <**suite** name=*"Suite"* parallel=*"tests"* thread-count=*"2"*>

<!-- parallel test should be run, how many is mentioned in thread count -->

1. <**test** name=*"Test"* parallel=*"classes"* thread-count=*"2"*> <!-- Here classes will run parallel -->