1. In TestNG there are many annotations and its sequence is

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

1. @Test = number of Test cases
2. For every test (@BeforeMethod, @AfterMethod) get executed, if there are two test cases, @beforemethod - @afterMethod will also execute for every Test Case
3. We can set priority of Test cases by **adding (priority =1, 2 and** so on)
4. If priority not given test cases will be executed **alphabetic order**.
5. We can also provide Groups to the test cases which are to be executed as a similar category **(Groups = “Google”)**
6. We can also set dependency of Test case on another test case by giving **“DependsOnMethod= TestCaseName**”
7. If DependsOnMethod= TestCaseName” get failed or skipped then dependent method will **get skipped automatically.**
8. For validation of Pass – Fail we uses **Assert. AssertEquals**
9. For running same Test case multiple time, we can use **InvocationTestCount**
10. If out code is going infinite loop, we can add **invocationTimeout =2**.
11. For avoiding Exceptions, we can **write ExpectedExceptions = ThreadTimeoutExpections.class**
12. To run all the Test cases in Batch we define all the classnames in the **TestNG.xml** File
13. We can check Report**. Test-output Folder index.html**
14. **Parameterization** using @Parameters

Parameterization

1. **While Paramtertization run test cases using testNG.xml**

**@Test  
@Parameters({ "url","email", "password"})**  
public void loginSiteTest(String url , String email, String password){  
driver = new ChromeDriver();  
driver.get(**url**);  
driver.findElement(By.xpath("//input[@id='eMailAddress']")).sendKeys(**email**);  
driver.findElement(By.xpath("//input[@id='userPassword']")).sendKeys(**password**);  
driver.findElement(By.xpath("//button[@type='submit']")).submit();

1. **In TestNG.xml First define paramters ,then className**

<test name="ParameterTest">  
<parameter name="url" value="https://www.qtpselenium.com/login"/>  
<parameter name="email" value="Vrushisa36.13@gmail.com"/>  
<parameter name="password" value="Test123"/>  
  
<classes>  
<class name = "testNGBasics.ParameterTest"/>

1. **Accessing data using annotation @DataProvider**

Step 1 : @Test( dataProvider = getTestData)

Public void LoginTest( String usename, String Password)

{

Driver.findElement(By.Id (“”)).sendKeys(username);

Driver.findElement(By.Id (“”)).sendKeys(Password);

}

Step 2: @DataProvider

Public Object [][] getTestData() {

Object[][] LoginData = new Object [2][2];

Object[0][0] = [Vrushali.haldankar@gmai.com](mailto:Vrushali.haldankar@gmai.com);

Object[0][1] = “Test.123”;

Object[1][0] = [Vrusli.haldankar@gmai.com](mailto:Vrusli.haldankar@gmai.com);

Object[1][1] = [“abcd”](mailto:Vrushali.haldankar@gmai.com);

return LoginData;

For handling failed Scenario re-run failed test cases

1. I had used TestNG Interfaces **IretryAnalyzer** and **IannotationTransformer**.
2. Created on class which implements interface **IretryAnalyzer** which has **retry**() method by default it returns false
3. Declared two variable **retryCount** and **MaxRetryCount**
4. retryCount =0 , MaxtryCount = 3 , if test case fails , it will attempt to run 3 time , 4th time also fail then it will be remark fail .
5. for every @Test annotation: we provided (retryanalyzer = Interface className.class)

But if there are 100 Test case, writing every Test case (retryanalyzer will not be a good practice.

1. So, we will use Interface **IannotationTransformer**. Which will be having transform method having 4 parameters (annotation, testclass, constructor, testmethod)

Will override

annotation. setRetryAnalyzer (MyRetry.class);

1. will declare this Interface implemented class listener in **testNG.xml**

<listeners>  
 <listener class-name="testNGBasics.MyTransformer"/>  
</listeners>

<test name="TestNG Concept">  
 <classes>  
 <class name="testNGBasics.TestRetry"/>  
 </classes>  
  
 </test>

HardAssert and SoftAssert

1. if there is a test case, application is failing at important test step is which must not mean to proceed to further step, like login, opening browser, then we shall user hard assert, so further test step execution will be stopped
2. if there is test case, we are checking 2-3 things in a same test case, it is failing for one of the steps, then also tester want to continue further testing, then we use soft Assertion, we create object of SoftAssert class, and refers assertion method along with instance variable.
3. Also it is mandatory to write sft.assertAll(); so we get failed result for that test case , if we don’t write we will get pass result, even test case is failed.
4. SoftAssert sft = new SoftAssert();

Sft.assertEquals(true, false);

Sft.assertAll();

Taking Screenshot for failed Test Cases

1. We wrote test cases test\_case\_001()
2. Defined on method failed method() which save screenshot in specific folder and will pass test case method as parameter

Public void **failed**(String TestCaseName)

File scrFile = ((TakeScreenshot)driver).getScreenShotAs(OutputType.FILE)

Try{“

FileHandler.Copy(srcFile, “folderPath”+TestCaseName+”.jpeg”)

}

Catch(Exception e)

{

e.printStackTrace();

}

1. Created one Class customListner which will Implement ITestListner.
2. Override all it’s method
3. onTestFailure method body , we will declare failed method
4. failed(result.getMethod().getMethodName());
5. Also in the Test defined class add annotation @Listeners(customListner.Class)