**TestNG Framework in Selenium Part-2**  
  
**Introduction to TestNG Testing Framework**  
I) Overview  
  
II) Install TestNG and write First TestNG Test Case  
  
III) Create multiple Test Cases and execute  
---------------------------------------------  
**TestNG Framework in Selenium Part-2**  
  
IV) Executing multiple programs / classes using XML file  
  
V) Grouping Test Cases  
  
VI) Parallel Test Execution  
-------------------------------------------------  
**IV) Executing multiple programs / classes using XML file**  
  
**Tags in XML**  
  
<suite name = "Suite Name">  
  <test name ="Test Name">  
    <classes>  
      <class name = "package.Class1Name"/>  
      <class name = "package.Class2Name"/>      
</classes>  
</test>  
</suite>  
----------------------------------------------  
**Create XML file**  
  
Select Java project/Package > Right click > New > Other...  
> Enter TestNG and Select TestNG Class  
> Enter source and package names  
> Enter XML file Name  
----------------------------------  
**XML File**  
<suite name="Ecommerce">  
  <test name="SanityTests">  
    <classes>  
      <class name="abcd.NewTest1"/>  
      <class name="abcd.NewTest2"/>  
    </classes>  
  </test>   
</suite>   
---------------------------  
**Class 1**  
  
public class NewTest1 {  
@BeforeClass  
public void login(){  
System.out.println("Login Successful");  
}  
@AfterClass  
public void logout(){  
System.out.println("Logout Successful");  
}  
@Test (priority = 1)  
public void addVendor(){  
System.out.println("Add Vendor Successful");      
}  
@Test(priority = 2)  
public void addProduct(){  
System.out.println("Add Product Successful");      
}  
@Test(priority = 3)  
public void addCurrency(){  
System.out.println("Add Currency Successful");      
}  
}  
----------------------------------  
**Class 2**  
  
@BeforeClass  
public void login(){  
System.out.println("Login Successful");  
}  
@AfterClass  
public void logout(){  
System.out.println("Logout Successful");  
}  
@Test (priority = 1)  
public void deleteVendor(){  
System.out.println("Delete Vendor Successful");      
}  
@Test(priority = 2)  
public void deleteProduct(){  
System.out.println("Delete Product Successful");      
}  
@Test(priority = 3)  
public void deleteCurrency(){  
System.out.println("Delete Currency Successful");      
}  
}  
-----------------------------------  
TestNG Annotations  
  
@Test - The annotated method is a part of a Test Case  
  
@BeforeMethod - The annotated method will be run before each Test method  
@AfterMethod - The annotated method will be run after each Test Method  
  
@BeforeClass - The annotated method will be run before the first test method in the current class is   
  
invoked.  
@AfterClass - The annotated method will be run after all the Test methods in the current class have   
  
been run.  
  
@BeforeTest - The annotated method will be run before any Test method belonging to classes inside   
  
the tag is run  
@AfterTest - The annotated method will be run after all the Test methods belonging to the classes   
  
inside the tage have run.  
-----------------------------------------------------  
**V) Grouping Test Cases**  
  
**XML File**  
  
<suite name="Suite" >  
  <test name="Test">  
  <groups>  
  <run>  
  <include name = "regression"/>  
  </run>  
  </groups>  
    <classes>  
      <class name="abcd.NewTest3"/>  
      </classes>  
  </test> <!-- Test -->  
</suite> <!-- Suite -->  
---------------------------------------------  
**Class File**  
  
public class NewTest3 {  
@Test(groups = {"sanity", "regression"}, priority =1)  
public void login(){  
System.out.println("Login Successful");  
}  
@Test (groups = {"sanity"}, priority =3)  
public void fundTransfer(){  
System.out.println("Fund Transfer Successful");  
}  
@Test(groups = {"sanity"}, priority =2)  
public void search(){  
System.out.println("Search Successful");  
}  
@Test (groups = {"regression"}, priority =2)  
public void advancedSearch(){  
System.out.println("Advanced Search Successful");  
}  
@Test(groups = {"regression"}, priority =3)  
public void prePaidRecharge(){  
System.out.println("PrePaid Recharge Successful");  
}  
@Test(groups = {"regression"}, priority =4)  
public void billPayments(){  
System.out.println("Bill Payments Successful");  
}  
@Test(groups = {"sanity", "regression"}, priority =10)  
public void logout(){  
System.out.println("Logout Successful");  
}  
}  
-------------------------------------  
**VI) Parallel Test Execution**  
  
1) Parallel Test Execution (Methods)  
**XML File**  
  
<suite name="Suite" parallel="methods" thread-count ="3">  
  <test name="Test">  
    <classes>  
      <class name="abcd.NewTest5"/>  
    </classes>  
  </test> <!-- Test -->  
</suite> <!-- Suite -->  
  
**Class File**  
  
public class NewTest5 {  
@Test  
public void testCase1(){  
long id = Thread.currentThread().getId();  
System.out.println("Test Case 1 is Successful - Thread id is: "+ id);        
}  
@Test  
public void testCase2(){  
long id = Thread.currentThread().getId();  
System.out.println("Test Case 2 is Successful - Thread id is: "+ id);        
}  
@Test  
public void testCase3(){  
long id = Thread.currentThread().getId();  
System.out.println("Test Case 3 is Successful - Thread id is: "+ id);        
}  
}  
----------------------------------------------------   
Note: A Thread is a concurrent unit of execution.   
------------------------------------   
2) Parallel Test Execution (Classes)  
  
**XML File**  
  
<suite name="Suite" parallel="classes" thread-count ="2">  
  <test name="Test">  
    <classes>  
      <class name="abcd.NewTest5"/>  
      <class name="abcd.NewTest6"/>  
      </classes>  
  </test> <!-- Test -->  
</suite> <!-- Suite -->  
---------------------------------------------  
**Class 1 File**  
  
public class NewTest5 {  
@Test  
public void testCase1(){  
long id = Thread.currentThread().getId();  
System.out.println("Test Case 1 is Successful - Thread id is: "+ id);        
}  
@Test  
public void testCase2(){  
long id = Thread.currentThread().getId();  
System.out.println("Test Case 2 is Successful - Thread id is: "+ id);        
}  
@Test  
public void testCase3(){  
long id = Thread.currentThread().getId();  
System.out.println("Test Case 3 is Successful - Thread id is: "+ id);        
}  
}  
---------------------------------------  
**Class 2 File**  
  
public class NewTest6 {  
@Test  
public void testCase4(){  
long id = Thread.currentThread().getId();  
System.out.println("Test Case 4 is Successful - Thread id is: "+ id);        
}  
@Test  
public void testCase5(){  
long id = Thread.currentThread().getId();  
System.out.println("Test Case 5 is Successful - Thread id is: "+ id);        
}  
@Test  
public void testCase6(){  
long id = Thread.currentThread().getId();  
System.out.println("Test Case 6 is Successful - Thread id is: "+ id);        
}  
}