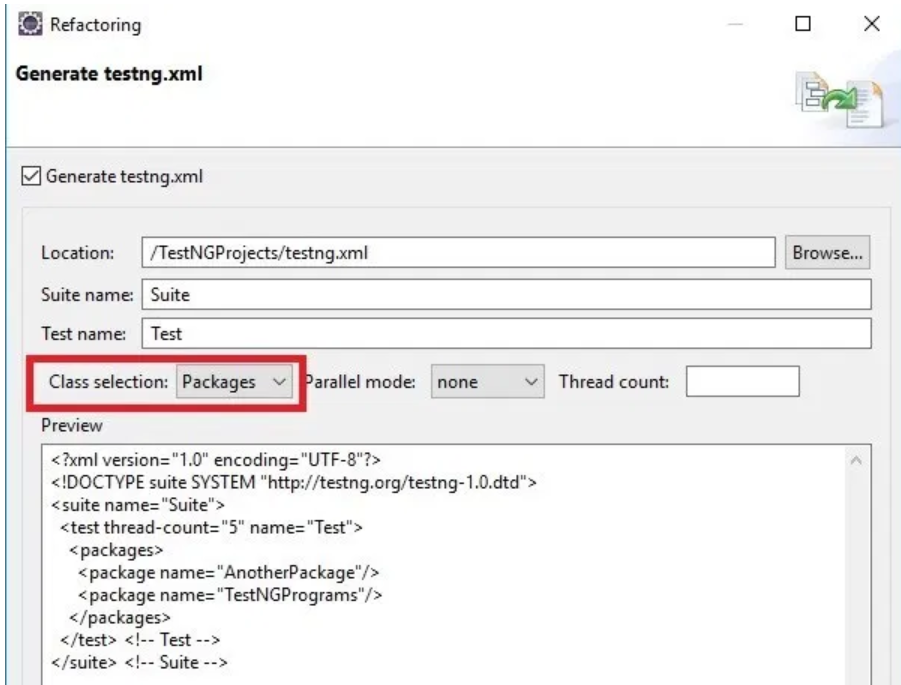
* “TestNG is a testing framework designed to simplify a broad range of testing needs, from unit testing (testing a class in isolation of the others) to integration testing (testing entire systems made of several classes, several packages and even several external frameworks, such as application servers).”.
* We can install TestNG in different ways given below:
* Download and add to build path.
* Using market place
* Using Install New Software option of Eclipse
* Using Maven’s pom.xml
* Using Gradle
* Writing a test is typically a three-step process:
* Write the business logic of your test and insert TestNG annotations in your code.
* Add the information about your test (e.g. the class name, the groups you wish to run, etc…) in a testng.xml file or in build.xml.
* Run TestNG.
* Since, TestNG is designed to automate control of testing activities, it is called a Testing Framework.
* testng xml is generated for all testng classes which are in scope based on user selection in project hierarchy. Remember you can not select multiple project to generate testng xml.
* Individual classes or packages can be selected and then testng.xml can be generated.Only the classes in scope(which are in selection) will be shown in the testng.xml

1. ****Method  : Test Cases****
2. ****Class       : Test Scenarios****
3. ****Classes   : Module Testing****
4. ****Test        : Integration Testing****
5. ****Suite      : System Testing****

* You can have only single <classes> tag in a <test> tag.
* testng.xml will include only those classes which has at least one @Test annotated method in it.

To generate a testng.xml using package tag, you just need to select “****Packages****” in ****Class selection**** drop-down in testng.xml generation window as shown below:



If you want to include all packages, TestNG allows you to use expressions in stead of multiple <package> tags as below:



* You can not use same concept in case of classes.It throws TestNG Exception.****org.testng.TestNGException: Cannot find class in classpath**** exception because it will search for a class with name “.\*”.
* if you want to run all classes of all packages in a Project, it is good to use <package> tags.

if we have multiple packages lets say 3 packages (A,B,C) so execution of those packages will happen from Top to bottom? means i want to execute all classes of A Package First, then all classes of B Package and at last for for C Package. will this ensure order of execution.

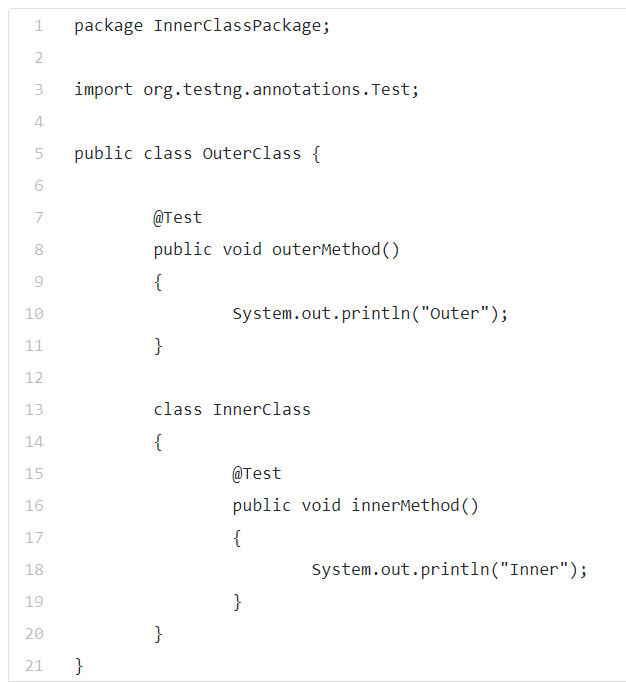
Preserver-order is an attribute to do so which you can set as true at test tag.Perserver-order preserves what you provide in testng xml.

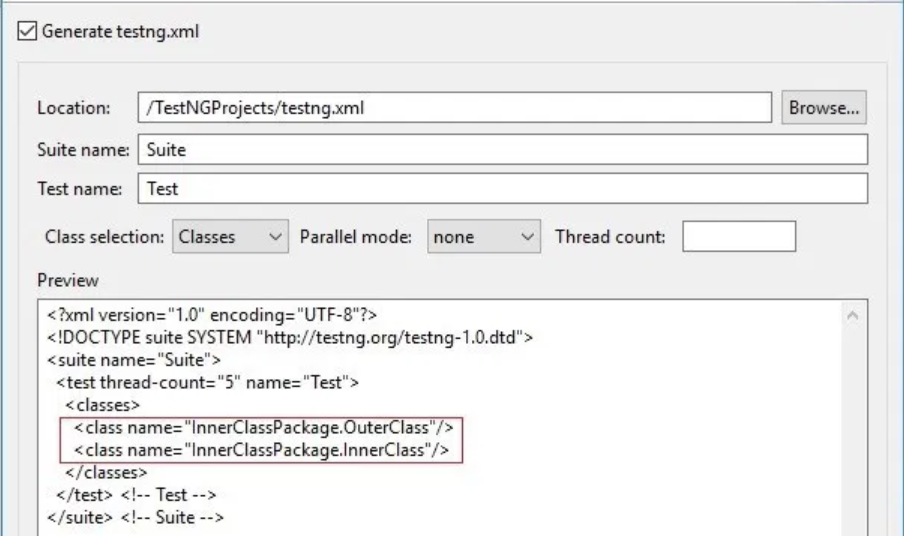
how can we make a test suite which will include all classes from package and sub packages?

****1.Select the Project and create testng.xml. It will include all class names from all packages and sub packages.**** Problem with this approach is that it will include all other packages also which you may not want to run.

1. You can use regular expressions with package name. You can take help of [<package>](http://makeseleniumeasy.com/2018/05/13/testng-tutorials-10-what-is-package-tag-and-how-to-use-in-testng-xml/) tag.
2. ****To include all sub packages of a package:  <package name=”MainPackage.\*”/>****

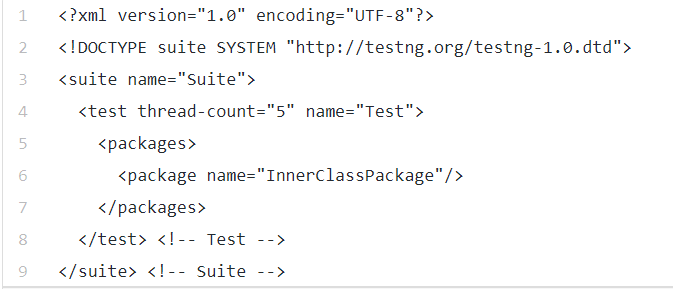
B. ****To include all classes and sub packages of a sub package: <package name=”MainPackage.SubPackage.\*”/>****





Though the inner class is included in the TestNG.xml, the inner class won’t run.throwing an exception stating can not find class in class path.

Solution: You need to take help of <package> tag here. When you use package name, it will include inner class also and run successfully.



* an interface can have TestNG annotated methods in it but you require an implemented class of interface to run it. But practically getting Null Pointer exception
* TestNG does not allow you to put any type of parameters to a TestNG annotated method. So we can overload methods directly in TestNG.It throws TestNG exception.
* To overload a method in testNG, we can use Data Provider/ @parameter annotation