TestNG Annotation Attributes

While writing the test cases in the TestNG, you need to mention the @Test annotation before the test method.

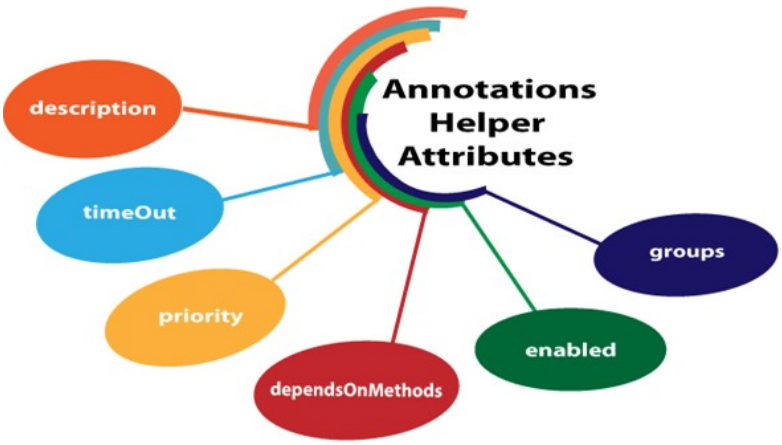
```
1. @Test
2. public void testcase1()
3. {
4.    System.out.println("This is testcase1");
5. }
```

In the above code, we have specified the @Test annotation before the test method, i.e., testcase1().

We can also explicitly specify the attributes in a **@Test** annotation. Test attributes are the test specific, and they are specified at the right next to the **@Test** annotation.

```
1. @Test(attribute="value")
2. public void testcase2()
3. {
4.  System.out.println("This is testcase2");
5. }
```

Some of the common attributes are described below:



- **description**
- **timeOut**
- **priority**
- **dependsOnMethods**

- **enabled**
- **groups**

description

It is a string which is attached to the @Test annotation that describes the information about the test.

Let's understand through an example.

```
1. package com.javatpoint;
2. import org.testng.annotations.Test;
3. public class Class1
4. {
5.
6. @Test(description="This is testcase1")
7. public void testcase1()
8. {
9. System.out.println("HR");
10. }
11. @Test(description="This is testcase2")
12. public void testcase2()
13. {
14. System.out.println("Software Developer");
15. }
16. @Test(description="This is testcase3")
17. public void testcase3()
18. {
19. System.out.println("QA Analyst");
20. }
21. }
```

In the above code, we have added the description attribute in every test. The "**description**" attribute provides information about the test.

dependsOnMethods

When the second test method wants to be dependent on the first test method, then this could be possible by the use of "**dependOnMethods**" attribute. If the first test method fails, then the dependent method on the first test method, i.e., the second test method will not run.

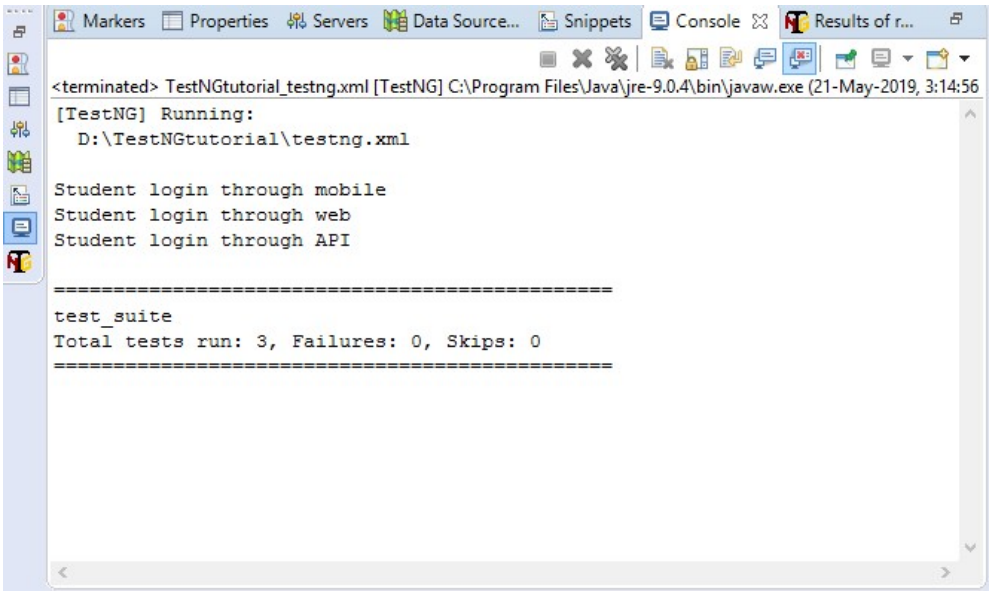
Let's understand through an example.

First case: When a single value is passed in a parameter.

```
1. package com.javatpoint;
2. import org.testng.annotations.Test;
3. public class Class1
4. {
5. @Test
6. public void WebStudentLogin()
7. {
8. System.out.println("Student login through web");
9. }
10. @Test
11. public void MobileStudentLogin()
12. {
13. System.out.println("Student login through mobile");
14. }
15. @Test(dependsOnMethods= {"WebStudentLogin"})
16. public void APIStudentLogin()
17. {
18. System.out.println("Student login through API");
19. }
20. }
```

We know that the TestNG executes the test methods in alphabetical order so, in the above program, APIStudentLogin() will execute first. However, we want WebStudentLogin() method to be executed before the execution of the APIStudentLogin() method, so this would only be possible through the "dependsOnMethods" attribute. In the above program, we have specified "dependsOnMethods" attribute in an APIStudentLogin() test method and its value is "WebStudentLogin" which means that WebStudentLogin() method will be executed before the APIStudentLogin() method.

Output



In the above output, MobileStudentLogin() runs before the WebStudentLogin() method as TestNG runs the test methods in an alphabetical order.

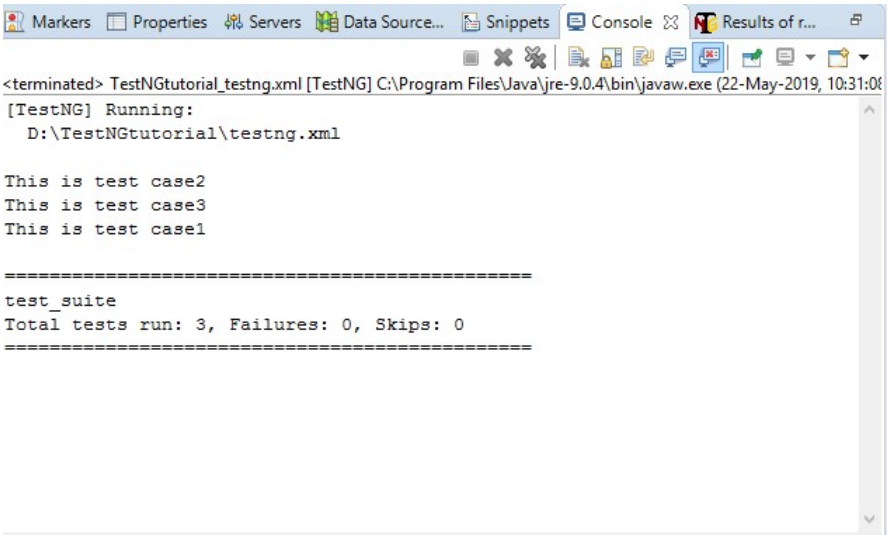
Second case: When multiple values are passed in a parameter.

```
1.  package com.javatpoint;
2.  import org.testng.annotations.Test;
3.  public class Depends_On_Groups
4.  {
5.   @Test(dependsOnMethods= {"testcase3","testcase2"})
6.   public void testcase1()
7.   {
8.       System.out.println("This is test case1");
9.   }
10.  @Test
11.  public void testcase2()
12.  {
13.      System.out.println("This is test case2");
14.  }
15.  @Test
16.  public void testcase3()
17.  {
18.      System.out.println("This is test case3");
19.  }
20.
```

```
21. }
```

In the above code, testcase1() is dependent on two methods, i.e., testcase2() and testcase3(), which means that these two methods will be executed before the testcase1().

Output



priority

When no 'priority' attribute is specified then the TestNG will run the test cases in alphabetical order. Priority determines the sequence of the execution of the test cases. The priority can hold the integer values between -5000 and 5000. When the priority is set, the lowest priority test case will run first and the highest priority test case will be executed last. Suppose we have three test cases and their priority values are -5000, 0, 15, then the order of the execution will be 0,15,5000. If priority is not specified, then the default priority will be 0.

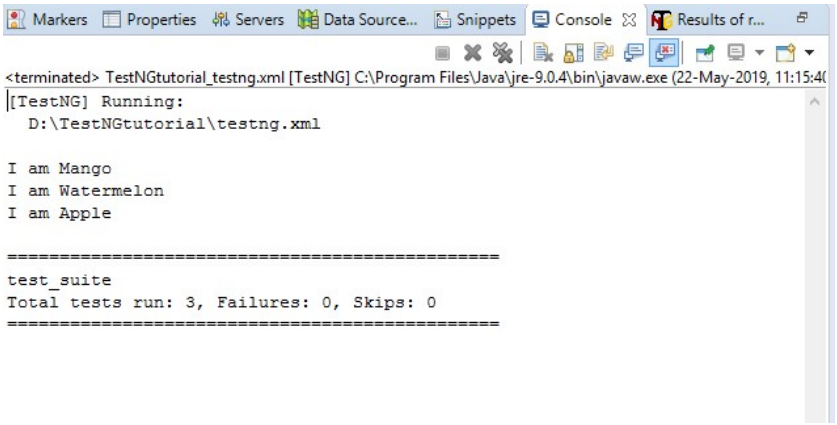
Let's understand through an example.

```
1. package com.javatpoint;
2. import org.testng.annotations.Test;
3. public class Fruits
4. {
5. @Test
6. public void mango()
7. {
8. System.out.println("I am Mango");
9. }
10. @Test(priority=2)
```

```
11. public void apple()
12. {
13. System.out.println("I am Apple");
14. }
15. @Test(priority=1)
16. public void watermelon()
17. {
18. System.out.println("I am Watermelon");
19. }
20. }
```

In the above code, the default priority of mango() test method is 0, so it will be executed first. The watermelon() test method will run after mango() method as the priority of watermelon() test method is 2. The apple() test method has the highest priority, so it will be executed last.

Output



enabled

The 'enabled' attribute contains the boolean value. By default, its value is true. If you want to skip some test method, then you need to explicitly specify '**false**' value.

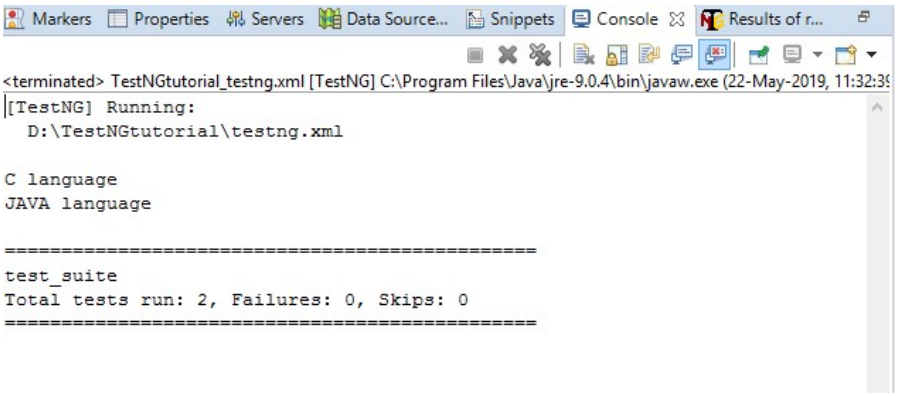
Let's understand through an example.

```
1. package com.javatpoint;
2. import org.testng.annotations.Test;
3. public class Programming_languages
4. {
5. @Test
6. public void c_language()
```

```
{
System.out.println("C language");
}
@Test(enabled=false)
public void jira()
{
System.out.println("JIRA is a testing tool");
}
@Test
public void java()
{
System.out.println("JAVA language");
}
}
```

In the above code, the value of the enabled attribute in jira() test method is false, so this method will not be invoked.

Output



groups

The 'groups' attribute is used to group the different test cases that belong to the same functionality.

Let's understand through an example.

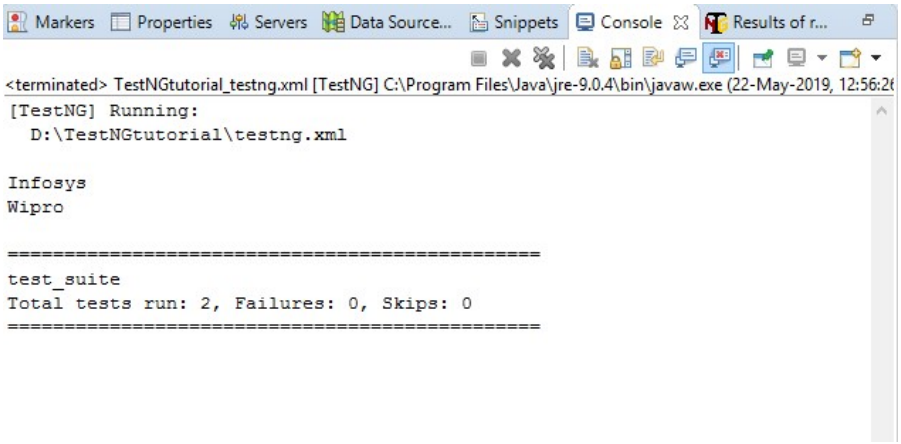
```
package com.javatpoint;
import org.testng.annotations.Test;
public class Software_Company
{
```

```
@Test(groups= {"software company"})
public void infosys()
{
System.out.println("Infosys");
}
@Test
public void technip()
{
System.out.println("Technip India Ltd");
}
@Test(groups= {"software company"})
public void wipro()
{
System.out.println("Wipro");
}
}
```

testng.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="test_suite">
<test name="Software Company">
<groups>
<run>
<include name="software company"/>
</run>
</groups>
<classes>
<class name="com.javatpoint.Software_Company"/>
</classes>
</test> <!-- Test -->
</suite> <!-- Suite -->
```

Output



timeOut

If one of the test cases is taking a long time due to which other test cases are failing. To overcome such situation, you need to mark the test case as fail to avoid the failure of other test cases. The timeOut is a time period provided to the test case to completely execute its test case.

Let's understand through an example.

```
1. package com.javatpoint;
2. import org.testng.annotations.Test;
3. public class Timeout_program
4. {
5. @Test(timeOut=200)
6. public void testcase1() throws InterruptedException
7. {
8. Thread.sleep(500);
9. System.out.println("This is testcase1");
10. }
11. @Test
12. public void testcaes2()
13. {
14. System.out.println("This is testcase2");
15.     }
16. @Test
17. public void testcase3()
18. {
19. System.out.println("This is testcase3");
```

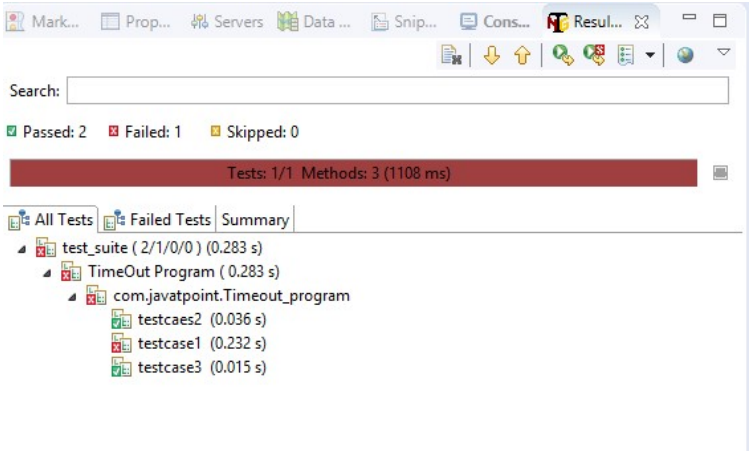
```
20. }
21. }
```

In the above code, inside the testcase1() method, we have Thread.sleep(500) which means that the testcase1() method will be executed after 500 milliseconds, but we have provided timeOUT attribute with the value 200 means that the testcase1() will be failed after 200 milliseconds.

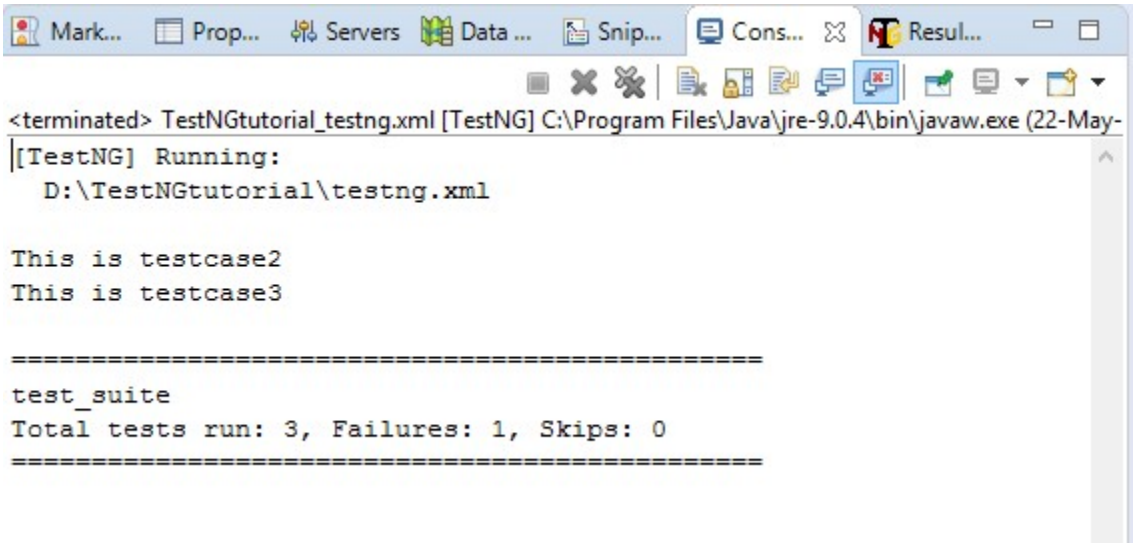
testng.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
3. <suite name="test_suite">
4. <test name="TimeOut Program">
5. <classes>
6. <class name="com.javatpoint.Timeout_program"/>
7. </classes>
8. </test> <!-- Test -->
9. </suite> <!-- Suite -->
```

Output



The above screen shows that one test case is failed and other test cases are passed.


Mark...
Prop...
Servers
Data ...
Snip...
Cons...
Resul...
<terminated> TestNGtutorial_testng.xml [TestNG] C:\Program Files\Java\jre-9.0.4\bin\javaw.exe (22-May-
[TestNG] Running:
D:\TestNGtutorial\testng.xml
This is testcase2
This is testcase3
test_suite
Total tests run: 3, Failures: 1, Skips: 0