我们在测试时，可能会遇到以下几种测试分组的场景：

一个测试类当中有多个测试方法，只想执行其中的几个相关的测试方法。

当几个相关的方法依赖相同的 setUp、tearDown操作。

当某个方法依赖几个相关的方法时，如几个相关的方法执行通过后，才执行该方法。

我们可以通过设置 **测试方法分组**的方式来解决上述问题。

分组测试是TestNG中的一个新的创新功能，它在JUnit框架中是不存在的。 在文中，我们将演示如何在TestNG中进行分组测试。

**场景一：一个测试类当中有多个测试方法，只想执行其中的几个相关的测试方法。**

代码示例如下，我们将TestDemo测试类的四个方法，通过@Test(groups = "") 方式，分成了两个组，分别为apiTest、databaseTest。

package framework.parse;

import org.testng.annotations.AfterGroups;

import org.testng.annotations.BeforeGroups;

import org.testng.annotations.Test;

public class TestDemo {

@Test(groups = "apiTest")

public void runApi1() {

System.out.println("runApi1()");

}

@Test(groups = "apiTest")

public void runApi2() {

System.out.println("runApi2()1");

}

@Test(groups = "databaseTest")

public void testOracle() {

System.out.println("testOracle()");

}

@Test(groups = "databaseTest")

public void testMySQL() {

System.out.println("testMySQL");

}

}

通过testng.xml文件进行运行配置管理，如下，我们仅运行framework.parse.TestDemo测试类中的databaseTest分组。

<?xml version="1.0" encoding="UTF-8"?><suite name="TestSuite">

<test name="testDemo">

<groups>

<run>

<include name="databaseTest" />

</run>

</groups>

<classes>

<class name="framework.parse.TestDemo" />

</classes>

</test>

</suite>

运行结果如下：

Run testMySql

Run testOracle

===============================================

TestSuite

Total tests run: 2, Passes: 2, Failures: 0, Skips: 0

===============================================

**场景二：当几个相关的方法依赖对应的 setUp、tearDown操作。**

我们可以通过@BeforeGroups、@AfterGroups方法实现用例组的setUp、tearDown操作，代码示例如下：

package framework.parse;

import org.testng.annotations.AfterGroups;

import org.testng.annotations.BeforeGroups;

import org.testng.annotations.Test;

public class TestDemo {

@BeforeGroups("databaseTest")

public void setUpDB() {

System.out.println("Run init database");

}

@AfterGroups("databaseTest")

public void tearDownDB() {

System.out.println("Run clean database");

}

@Test(groups = "databaseTest")

public void testOracle() {

System.out.println("Run testOracle");

}

@Test(groups = "databaseTest")

public void testMySql() {

System.out.println("Run testMySql");

}

}

运行结果如下，在执行databaseTest分组中的用例之前，执行了setUpDB 方法，在databaseTest分组分组中的用例执行完成后，执行了tearDownDB方法：

Run init database

Run testMySql

Run testOracle

Run clean database

===============================================

Default Suite

Total tests run: 2, Failures: 0, Skips: 0

===============================================

**场景三：当某个方法依赖几个相关的方法时**

我们可以通过 @Test(dependsOnGroups = { "" }) 方法实现优先执行依赖测试组，当依赖测试组执行通过后，则执行被依赖方法，否则跳过被依赖方法。代码示例如下：

import org.testng.annotations.AfterGroups;

import org.testng.annotations.BeforeGroups;

import org.testng.annotations.Test;

public class TestDemo {

@Test(groups = "apiTest")

public void runApi1() {

System.out.println("Run runApi1");

}

@Test(groups = "apiTest")

public void runApi2() {

System.out.println("Run runApi2");

}

@Test(groups = "databaseTest")

public void testOracle() {

System.out.println("Run testOracle");

}

@Test(groups = "databaseTest")

public void testMySql() {

System.out.println("Run testMySql");

}

@Test(dependsOnGroups = { "databaseTest", "apiTest" })

public void runFinal() {

System.out.println("runFinal");

}

}

上述代码示例中，runFinal 方法依赖databaseTest组、apiTest组方法，需要在这两个组的方法执行通过后，才可以执行runFinal方法，我们通过 @Test(dependsOnGroups = { "databaseTest", "apiTest" }) 注解的方式来实现，运行代码，执行结果如下：

Run runApi1

Run runApi2

Run testMySql

Run testOracle

runFinal

===============================================

Default Suite

Total tests run: 5, Failures: 0, Skips: 0

===============================================

import org.testng.Assert;

import org.testng.annotations.AfterGroups;

import org.testng.annotations.BeforeGroups;

import org.testng.annotations.Test;

public class TestDemo {

@Test(groups = "apiTest")

public void runApi1() {

System.out.println("Run runApi1");

// 运行失败

Assert.assertEquals(1,2);

}

@Test(groups = "apiTest")

public void runApi2() {

System.out.println("Run runApi2");

}

@Test(groups = "databaseTest")

public void testOracle() {

System.out.println("Run testOracle");

}

@Test(groups = "databaseTest")

public void testMySql() {

System.out.println("Run testMySql");

}

@Test(dependsOnGroups = { "databaseTest", "apiTest" })

public void runFinal() {

System.out.println("runFinal");

}

}

如上，示例代码中apiTest分组中的runApi1测试用例将断言失败，此时运行测试用例，执行结果如下，我们会看到runFinal方法被没有被执行，而是跳过。

Run runApi1

java.lang.AssertionError: expected [2] but found [1]

Expected :2

Actual :1

at org.testng.Assert.fail(Assert.java:94)

at org.testng.RemoteTestNGStarter.main(RemoteTestNGStarter.java:123)

Run runApi2

Run testMySql

Run testOracle

Test ignored.

===============================================

Default Suite

Total tests run: 5, Failures: 1, Skips: 1

===============================================