JMockit

C-2 创建: 张林伟, 最后修改: 张林伟 04-03 17:11

Maven配置

代码块 XML

```xml
<!--如果你是通过mvn test来运行你的测试程序 , 请确保JMockit的依赖定义出现在JUnit的依赖之前-->
<dependency>
    <groupId>org.jmockit</groupId>
    <artifactId>jmockit</artifactId>
    <version>1.36</version>
  </dependency>
```

JMockit测试程序结构

Record-Replay-Verification

1. Record: 即先录制某类/对象的某个方法调用，在当输入什么时，返回什么
2. Replay: 即重放测试逻辑
3. Verification: 重放后的验证。比如验证某个方法有没有被调用，调用多少次

@Mocked注解

可以修饰类和接口，生成一个mocked对象（**是针对其修饰类的所有实例**，即时再自己new一个对象，也是被mock的）

方法返回类型为short,int,float,double,long则返回0，方法返回类型为String则返回null，方法返回类型为其他对象则返回mocked对象

@Injectable注解

只是针对其修饰的实例，生成一个Mocked对象

@Tested注解

@Tested表示被测试对象。如果该对象没有赋值，JMockit会去实例化它，若@Tested的构造函数有参数，
则JMockit通过在测试属性&测试参数中查找@Injectable修饰的Mocked对象注入@Tested对象的构造函数来实例化，
不然，则用无参构造函数来实例化。除了构造函数的注入，JMockit还会通过属性查找的方式，把@Injectable对象注入到@Tested对象中。

@Capturing注解

只知道父类或接口时，但我们需要控制它所有子类的行为时，子类可能有多个实现（可能有人工写的，也可能是AOP代理自动生成时）。就用@Capturing。

Expectations使用

1.通过引用外部类的Mock对象(@Injectabe,@Mocked,@Capturing)来录制

2.通过构建函数注入类/对象来录制：可以达到只mock类/对象的部分行为的目的

MockUp & @Mock

可以直接mock指定方法，改写方法内部逻辑

Verifications使用

通常，在实际测试程序中，我们更倾向于通过JUnit/TestNG/SpringTest的Assert类对测试结果的验证， 对类的某个方法有没调用，调用多少次的测试场景并不是太多。因此在验证阶段，我们完全可以用JUnit/TestNG/SpringTest的Assert类取代new Verifications() {{}}验证代码块。

除非，你的测试程序关心类的某个方法有没有调用，调用多少次，你可以使用new Verifications() {{}}验证代码块。

如果你还关心方法的调用顺序，你可以使用new VerificationsInOrder() {{}}

参考资料：

http://jmockit.cn/index.htm