冰云

冰云

浏览: 99069 次

性别:

我现在离线

详细资料 留言簿

敏捷需求分析 | Struts快速入门（五完）

2004-06-04

MockObjects的选择：EasyMock与JMock的比较

MockObjects的选择：**EasyMock**与**JMock**的比较

本文假设读者已经了解了MockObjects的使用目的和基本方式，不对MockTest之类的技术作过多解释。仅提醒一句：“不要测试你的MockObjects”。

本文作为一个评测结果的同时，也可以作为EasyMock和jMock的简短教程。他们本身都很易用，可惜带的示例过于复杂，都用了过多的模式。看过本文的例子，相信就可以从容的在项目中使用了。

Java中常用的MockObjects有EasyMock和jMock等。其中EasyMock开发较早，已经出了1.1版本，而jMock前几天才刚推出了1.0 final。作为刚成熟的小弟弟，jMock有什么竞争实力呢？

本比较针对于以下几个方面，代码请见附件。

1 是否能够对具体类进行模拟（当然，对接口模拟是基本功能）

2 是否能够对方法名，参数，返回值进行动态控制

3 基本代码行数

4 是否能够对具有构造参数的具体类模拟

现在比较开始了。首先制作若干测试文件，很简单。要模拟的有一个接口和一个具体类，叫做TheInterfaceToMock和TheClassToMock，另外，提供方法SampleReturn sampleMethod(Parameter p);以及同名无参数方法。

第一个测试是针对TheInterfaceToMock，提供ParameterImpl和SampleReturnImpl作为期待的参数和返回值。

jMock代码如下：

```
public class JMockUsage extends MockObjectTestCase {
  public void testReturnValueWithParemeter(){
    // 构造Mock控制器
    Mock m = new Mock(TheInterfaceToMock.class);
    // 这是要测试MockObject
    TheInterfaceToMock mock = (TheInterfaceToMock) m.proxy();
    // 期待的返回值
    SampleReturn sr = new SampleReturnImpl();
    // 期待的参数
    Parameter p = new ParameterImpl();

    // 控制器，期待一次，方法sampleMethod，参数等于p(equals)，将返回sr
    m.expects(once()).method("sampleMethod")
            .with(eq(p)).will(returnValue(sr));
```

- 默认类别 (4)

我的留言簿 >>更多留言

- 你好，我想问下，关于敏捷的培训你们近期还有嘛？ 能不能把课程相关信息告知我下， ...
  -- by louiscai

其他分类

- 我的收藏 (1)
- 我的论坛主题贴 (12)
- 我的所有论坛贴 (170)
- 我的精华良好贴 (6)

最近加入圈子

存档

- 2007-06 (2)
- 2007-04 (1)
- 2007-03 (1)
- 更多存档...

最新评论

- 敏捷团队建设
  看了这个 搞的我非常有冲动想跳槽-。-
  -- by java_mid4
- 敏捷团队建设
  高，这是我向往的团队。在广州从业开发两年了，跳了第两家公司都是以项目为主导的小公司 ...
  -- by chenhui
- 敏捷团队建设


```
        // 正式执行mockobject
        SampleReturn ret = mock.sampleMethod(new ParameterImpl());
        // 确定返回值是相同的
        assertSame(sr,ret);
    }

}
```

相同功能的easyMcok代码如下：

```
public class EasyMockUsage extends TestCase {
    public void testReturnValueWithParameter(){
        // 构造mock控制器
        MockControl control
                = MockControl.createControl(TheInterfaceToMock.class);
        // 这是要测试的MockObject
        TheInterfaceToMock mock
                = (TheInterfaceToMock) control.getMock();
        // 这是要返回的值
        SampleReturn sr = new SampleReturnImpl();
        // 这是要传入的参数
        Parameter p = new ParameterImpl();

        // 恢复到记录(record)状态
        control.reset();
        // 首先记录sampleMethod方法
        mock.sampleMethod(p);
        // 设定该方法的返回值
        control.setReturnValue(sr);
        // 切换状态为回复(reply)
        control.replay();
        // 正式执行mock object的方法，明显的，参数值是equals而不是same
        SampleReturn ret = mock.sampleMethod(new ParameterImpl());

        // 确定返回值是需要的值
        assertSame(sr,ret);
    }
```

从上面的代码可以看到，同样的功能，二者的行数相差3行。其主要原因，就是easyMcok的Mock机制是基于状态，首先是录制状态，记录下来待测的方法和参数，返回值等，然后切换为回复状态。而jMock没有切换这一步，直接将参数返回值用一句话写出来。确实是一句话：期待一次，方法sampleMethod，参数等于p(equals)，将返回sr。其中的一些辅助函数，例如returnValue,eq等等，位于父类MockTestCase。

结论：

廉价是没错了是不是好用和听话就难说了

-- by t0uch

- 敏捷团队建设

  有一个好的团队非常重要。

  -- by 狂放不羁

- 敏捷团队建设

  经济基础决定上层建筑，很多的优越条件都是建立在高利润率的基础之上，现在很多企业都挣 ...

  -- by w_y_g

评论排行榜

[什么是RSS?]

1 如果不能提供MockTestCase作为父类，请使用EasyMock
2 如果需要批量或动态生成测试，请使用更规则的jMock
3 如果喜欢看起来行数少一些，请用jMock
4 如果对状态切换看不顺眼，请用Mock

下面进行具体类测试，一个共同的点是，二者均使用了CGLIB作为增强器，因此效率差别几乎没有。将上面的测试稍稍修改，将TheInterfaceToMock改为TheClassToMock。发生了以下变化。

用jMock，需要将import替换为新的import，代码中其他部分完全不变！

```
原来
import org.jmock.Mock;
import org.jmock.MockObjectTestCase;
改为：
import org.jmock.cglib.Mock;
import org.jmock.cglib.MockObjectTestCase;
```

这是个相当体贴的设计，保证了接口的一致性。对于一套API来说，同样的类却有不同的使用方法是个噩梦。

用easyMock，需要新增加一个import。并且修改一些声明的地方。

```
原来
import org.easymock.MockControl;
增加
import org.easymock.classextension.MockClassControl;
```

```
// mock控制器
MockControl control = MockClassControl.createControl(TheClassToMock.class);
```

其他部分不需要变化。虽然这有些变化，但是变化带来了其他的好处，就是：能够支持带有构造参数的具体类，而jMock不支持。这对于大量使用了PicoContainer的代码来说不啻是一个福音。

结论：

5 如果需要构造参数，只能使用easyMock
6 如果喜欢用相同的API操作并且不在乎构造参数，请用jMock
7 如果愿意等待下一版本的jMock提供构造参数支持，请用jMock

参考比较表：

	EasyMock	jMock
通过接口模拟	是	是
控制方法有效次数	是	是
定制参数匹配	是	是
不需要状态转换	否	是
具体类模拟	是	是

具体类可有构造参数	是	否
接口统一	否	是
条件代码在一行中完成	否	是
支持其他参数规则，如not	否	是
自验证 verify()	是	是

综上，我选择了jMock。不过想想看，easyMock用3个类实现了大多数常用功能，很不简单啊。而jMock，如果能够提供对Constructor injection的支持就完美了。遗憾。不过从设计上看，jMock里的模式使用堪称典范，很好看哦。

本人对easyMock使用经验不多，如有谬误请指出。

下载地址：

http://www.jmock.org

http://www.easymock.org

比较代码：

http://icecloud.51.net/data/mockobjects.zip

需要：

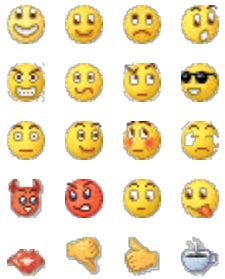
JUnit3.8.1，Cglib2，jMock1.0，EasyMock1.1

敏捷需求分析 | Struts快速入门（五完）

评论

发表评论

表情图标



B I U Quote Code List Img URL Flash Table

字体颜色: 标准 字体大小: 标准 对齐: 标准

提示：选择您需要装饰的文字, 按上列按钮即可添加上相应的标签

您还没有登录，请登录后发表评论(快捷键 Alt+S / Ctrl+Enter) 提交