‹‹ 上一主题 | 下一主题 ››

回复 发贴

急救！~！~关于jmock-2.3.0 打印

kkooo

中级站友

个人空间 发短消息

加为好友 当前离线

发表于 2007-10-7 20:17 只看该作者 小 中 大 1#

急救！~！~关于**jmock-2.3.0**

各位大虾：今天小弟初试jmock-2.3.0，就遇到棘手的问题（忒郁闷了！）。请各位大虾救救，下面是原代码

```
import org.jmock.*;
import javax.servlet.http.*;

public class MockRequestTest extends MockObjectTestCase{

    public void testMockRequest1(){

  // 构造一个 Mock 对象
Mock mock = new Mock(HttpServletRequest.class);

// 设置要执行的操作，以下设置表示要调用一次 HttpServletRequest 对象的

//getParameter 方法，传递的参数是 name, 期望的返回是 kongxx

  mock.expects(once()).method(getParameter).with(eq(name)).will(returnValue(kongxx));
// 根据 Mock 对象获取一个 HttpServletRequest 对象
```

```
HttpServletRequest request = (HttpServletRequest)mock.proxy();

// 断言调用结果
    assertEquals(kongxx ,request.getParameter(name));

}
```

为什么我构造一个 Mock 对象却没有构造成功。

它提示我先要新建一个Mock类 。我很郁闷！，不会真是要新建一个吗？

（我用的工具是eclipse3.2 /junit3.8/jmock-2.3.0-jars）

搜索更多相关主题的帖子: 急救

TOP

kkooo

中级站友

个人空间　发短消息

加为好友　当前离线

发表于 2007-10-8 14:30 只看该作者　　小 中 大 2#

回复 **1#** 的帖子

我们就简单测试一些 HttpServletRequest 的 getParameter方法来简单看一些 EasyMock 与 JMock 在Mock 上的区别：

我们先看看在JMock 中的一个简单的 TestCase：

```
package cn.allanchanly.test.jmock;

import javax.servlet.http.HttpServletRequest;

import org.apache.log4j.Logger;
import org.jmock.Expectations;
import org.jmock.Mockery;
import org.jmock.integration.junit3.MockObjectTestCase;

public class HttpServletMockTest extends MockObjectTestCase {
    private static Logger log = Logger.getLogger(cn.allanchanly.test.jmock.HttpServletMockTest.class);

    private Mockery m = new Mockery();
    private HttpServletRequest test;
```

```
    protected void setUp()throws Exception{
        test = m.mock(HttpServletRequest.class);
    }

    protected void tearDown()throws Exception{

    }

    public void testServlet(){
        m.checking(new Expectations(){{
            one(test).getParameter("userName");will(returnValue("allanchanly"));
            allowing(test).getParameter("password");will(returnValue("allanchanly-password"));
        }
        });

        log.debug(test.getParameter("userName"));
        log.debug(test.getParameter("password"));

        m.assertIsSatisfied();

//        log.debug(test.getParameter("userNamexxx"));
    }
}
```

然后我们再看看利用EasyMock 怎样做，最后我们会总结一下：

```
package cn.allanchanly.test.easymock;

import javax.servlet.http.HttpServletRequest;

import org.apache.log4j.Logger;

import junit.framework.TestCase;
```

```
import static org.easymock.EasyMock.*;

public class TestRequestByEasyMock extends TestCase {
    private static Logger log = Logger.getLogger(cn.allanchanly.test.easymock.TestRequestByEasyMock.class);

    private HttpServletRequest mockRequest;

    protected void setUp() throws Exception {
        super.setUp();
        log.debug("we set up to test HttpServletRequest by EasyMock!");
    }

    protected void tearDown() throws Exception {
        super.tearDown();
    }

    public void testGetparameter(){
        mockRequest = createMock(HttpServletRequest.class);
        expect(mockRequest.getParameter("userName")).andReturn("allanchanly");
        expectLastCall().times(2);
        replay(mockRequest);
        log.debug(mockRequest.getParameter("userName"));
        assertTrue(mockRequest.getParameter("userName").equals("allanchanly"));
        verify(mockRequest);
    }
}
```

我们将关键代码写在这里:

在JMock中我们这样写:

```
private Mockery m = new Mockery();
```

```
private HttpServletRequest test;

test = m.mock(HttpServletRequest.class);

m.checking(new Expectations(){{
        one(test).getParameter("userName");will(returnValue("allanchanly"));
        allowing(test).getParameter("password");will(returnValue("allanchanly-password"));
    }

log.debug(test.getParameter("password"));

m.assertIsSatisfied();
```

而在EasyMock中我们这样写：

```
private HttpServletRequest mockRequest;

mockRequest = createMock(HttpServletRequest.class);

expect(mockRequest.getParameter("userName")).andReturn("allanchanly");
expectLastCall().times(2);
replay(mockRequest);
log.debug(mockRequest.getParameter("userName"));
assertTrue(mockRequest.getParameter("userName").equals("allanchanly"));
verify(mockRequest);
```

简单的总结一下：

在JMock中我们首相构建一个Mockery 对象，然后去mock 一个class（待测试或者需要的特定类，这个类一般需要特定的容器才能提供，我们在这里Mock模拟一下）

然后我们利用Mockery.checking()给其中传递一个Expectations，利用匿名类组织我们的Expectation，其中什么one(),allowing(),will(),returnValue()阿都是常用来的模拟行为的。

然后我们就直接测试或者利用我们构建的行为了，其实我认为一般还是利用，比如你要测试自己的Servlet，需要一

个Request，没有人在哪儿把一个Request 测着玩的。

而在EasyMock中其实差不多，代码稍有不同而已，由于采用了静态引入（import static ）所以EasyMock中的方法我们可以直接使用， 一般过程还是先 createMock 一个class，然后开始定制我们的expect，其实就是“录制”一系列我们需要的行为，比如调用哪个方法，返回什么值等，或者哪个方法调用几次等，然后replay(Mock对象)，重新放映我们的行为，测试方法，最后别忘了还有个verify(mock对象)来检验一下，

TOP

‹‹ 上一主题 | 下一主题 ››

回复 发贴

版块跳转 ...