**实验三：SSH测试**

* [**单元测试Struts2Spring项目的Action和Service**](http://www.blogjava.net/qileilove/archive/2013/11/14/406328.html)

<http://www.blogjava.net/qileilove/archive/2013/11/14/406328.html>

**1.特别说明。**

　　http://blog.csdn.net/fansunion/article/details/12118043  单元测试Struts2的Action（包含源码），这篇文章主要讲述的是如何使用**JUnit**等单元测试框架测试 Struts2这一个框架的Action。

而本篇侧重Struts2和Spring这2个框架集成的情况。

**2.新建工程，加入相关jar包。**

　　Struts相关jar包

　　Spring相关jar包

　　JUnit, spring-test-3.2.3.RELEASE.jar,struts2-junit-plugin-2.2.3.1.jar等测试相关jar包

Tomcat的Servlet/JSP jar包

**3.新建Action。**

|  |
| --- |
| package action;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Controller;  /\*\*  \* 一个简单的Action。  \* @author FansUnion  \*  \*/  @Controller  public class UserAction {  @Autowired  private UserService userService;  public String getName() {  userService.getName("FansUnion");  return "success";  }  } |

**4.新建Service。**

|  |
| --- |
| package action;  import org.springframework.stereotype.Service;  @Service  public class UserService {  /\*\*  \* 简单的返回用户名  \*/  public String getName(String name) {  // 这个地方实际上应该调用Dao持久层的代码，在此简化了。  // 本文只是单元测试Struts2Spring整合中，Struts的Action和Spring注入的Service。  // Struts2Spring+Hibernate/Mybatis将在后续篇章中介绍。  return name;  }  } |

**5.单元测试Action。**

|  |
| --- |
| package unittest;  import org.apache.struts2.StrutsSpringTestCase;  import org.junit.Test;  import action.UserAction;  import com.opensymphony.xwork2.ActionProxy;  /\*\*  \* 测试StrutsSpring集成的时候，需要继承StrutsSpringTestCase这个类，  \* 而单独测试Struts的时候，继承StrutsTestCase。  \*  \* @author http://blog.csdn.net/fansunion/  \*  \*/  public class ActionUnitTest extends StrutsSpringTestCase {  // 重写父类方法,指定配置文件的名字  protected String[] getContextLocations() {  return new String[] { "struts.xml", "applicationContext-spring.xml" };  }  @Test  public void testExecute() throws Exception {  ActionProxy proxy = getActionProxy("/unitTest");  UserAction test = (UserAction) proxy.getAction();  assertNotNull(test);  String result = proxy.execute();  assertEquals("success", result);  }  } |

**6.单元测试Service。**

|  |
| --- |
| package unittest;  import org.junit.AfterClass;  import org.junit.BeforeClass;  import org.springframework.mock.web.MockServletContext;  import org.springframework.web.context.WebApplicationContext;  import org.springframework.web.context.support.XmlWebApplicationContext;  /\*\*  \* 测试Service的基类  \*  \* @author http://blog.csdn.net/fansunion/  \*  \*/  public class JUnitTestBase {  public static XmlWebApplicationContext context = null;  public static String[] CONFIG\_FILES = { "file:src/applicationContext-\*.xml" };  public JUnitTestBase() {  System.out.println("JUnitTestBase");  }  @BeforeClass  public static void setUp() {  System.out.println("Test start…");  context = new XmlWebApplicationContext();  context.setConfigLocations(CONFIG\_FILES);  MockServletContext msc = new MockServletContext();  context.setServletContext(msc);  context.refresh();  msc.setAttribute(  WebApplicationContext.ROOT\_WEB\_APPLICATION\_CONTEXT\_ATTRIBUTE,  context);  }  @AfterClass  public static void tearUp() {  System.out.println("Test end!");  }  }  package unittest;  import static org.junit.Assert.assertEquals;  import org.junit.Test;  import action.UserService;  public class UserServiceTest extends JUnitTestBase {  private UserService userService;  public UserServiceTest() {  userService = context.getBean(UserService.class);  }  @Test  public void test() {  String name = "http://FansUnion.cn";  String myName = userService.getName(name);  assertEquals(name, myName);  }  } |

**7.Struts2配置。**

|  |
| --- |
| <struts>  <!– Development Mode –>  <constant name="struts.devMode" value="true" />  <constant name="struts.enable.DynamicMethodInvocation" value="false" />  <constant name="struts.objectFactory" value="spring" />  <constant name="struts.objectFactory.spring.autoWire" value="name" />  <package name="manager" namespace="/" extends="struts-default">  <action name="unitTest" class="userAction" method="getName">  <result name="success">unitTest.jsp  </result>  </action>  </package>  </struts> |

**8.Spring配置。**

（注意：这里的配置项是复制过来的，很多配置根据实际需要可以删减）

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <beans xmlns="http://www.springframework.org/schema/beans"  xmlns:flex="http://www.springframework.org/schema/flex" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xmlns:context="http://www.springframework.org/schema/context"  xmlns:aop="http://www.springframework.org/schema/aop" xmlns:tx="http://www.springframework.org/schema/tx"  xsi:schemaLocation="  http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans-2.5.xsd  http://www.springframework.org/schema/tx  http://www.springframework.org/schema/tx/spring-tx-3.0.xsd  http://www.springframework.org/schema/aop  http://www.springframework.org/schema/aop/spring-aop-3.0.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context-3.0.xsd  http://www.springframework.org/schema/flex  classpath:/spring-flex-1.0.xsd  ">  <context:annotation-config />  <context:component-scan base-package="\*" />  <aop:aspectj-autoproxy />  </beans> |

**9.web.xml配置。**

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <web-app id="WebApp\_ID" version="2.4"  xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app\_2\_4.xsd">  <!– Spring Context –>  <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>classpath:/applicationContext\*.xml</param-value>  </context-param>  <!– Spring Context Listener –>  <listener>  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>  </listener>  <filter>  <filter-name>struts2</filter-name>  <filter-class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter</filter-class>  </filter>  <filter-mapping>  <filter-name>struts2</filter-name>  <url-pattern>\*.action</url-pattern>  </filter-mapping>  </web-app> |