Java web 项目中单元与集成测试

Walter on http://wind007.javaeye.com

单元测试的重要性早已深入人心,测试驱动开发随着这两年极限编程的推广也逐渐流行起来.在付出时间与精力写单元测试的同时,确实也收获了降低 regression bug,提高代码质量的好处

以几年前做的一个项目为例, 这个项目是以 Struts+Spring+iBatis 为框架扩展起来的一个 Java Web 项目,最初只是用 JUnit 对业务层的 Business Class 进行测试,后来逐步扩展到演示层和持久层.并结合了 CruiseControl 进行持续集成并生成测试覆盖率的报告,效果感觉不错.

1.工欲善其事,必先利其器

Java 项目的一大好处就是有许多方便快捷的框架和工具可供选择使用,就是要花点时间去寻找发现它们.在这里的单元测试过程中我们使用了如下的辅助工具

JUnit(http://www.junit.org/index.htm)

这个不用说了,Java 单元测试的开山之作

Ant(http://ant.apache.org/)

这个更加不用说了,这里主要用来编译和执行单元测试

StrutsTestCase(http://sourceforge.net/projects/strutstestcase/)

针对大量的 Struts Action 进行的单元测试的好工具

EasyMock(http://www.easymock.org/)

由于项目是分层分人开发的,Mock 工具的使用必不可少

JMeter(http://jakarta.apache.org/jmeter/)

主要用作 web 性能测试,并结合 badboy 进行脚本的录制

Spring(http://www.springframework.org/)

使用它的 BeanFactory 和相关的测试辅助类

Emma(http://emma.sourceforge.net/)

测试覆盖率工具,以生成测试覆盖率报告

CruiseControl(http://cruisecontrol.sourceforge.net/)

持续集成工具

2.单元测试在各个层次中的应用

首先建立一个与源代码目录结构相同的 Test 目录,以放置测试类
然后编辑 ant 的 build 文件,添加如下的 test 目标
执行全部测试类: ant test
执行单个测试类: ant test -Dtestcase=MenuActionUnitTest

```
<target name="test" depends="compile.test" description="Run JUnit Tests">
  <antcall target="runsql"/>
  <junit printsummary="on" fork="yes" haltonfailure="false" failureproperty="tests.failed"
showoutput="true">
      <classpath>
            <pathelement location="${build.dir}/classes"/>
          <pathelement location="${build.dir}/test"/>
          <path refid="master-classpath"/>
        </classpath>
          <formatter type="xml"/>
      <batchtest todir="${docs.dir}/results/junit" unless="testcase">
          <fileset dir="${build.dir}/test">
                <include name="**/*UnitTest.*" />
                <exclude name="**/*InteTest.*"/>
              </fileset>
        </batchtest>
       <batchtest todir="${docs.dir}/results/junit" if="testcase">
          <fileset dir="${build.dir}/test">
                <include name="**/*${testcase}*"/>
              <exclude name="**/*TestCase.class"/>
              </fileset>
        </batchtest>
  </junit>
```

```
    <junitreport todir="${docs.dir}/reports/junit">
        <fileset dir="${docs.dir}/results/junit">
                <include name="TEST-*.xml"/>
            </fileset>
            <report format="frames" todir="${docs.dir}/reports/junit/html"/>
        </junitreport>
        <fail message="JUnit tests failed. Please check reports." if="test.failed"/>
</target>
```

以比较简单的 Menu 模块为例

2.1.演示层

由于演示层使用了 struts 测试框架,故采用了 StrutsTestCase 工具,同时为了在其中插入模拟对象,在单元测试中对了 struts 的 RequestProcessor 进行了替换

2.1.1 编写 MockMenuAction,只是简单地从 MenuAction 继承,单元测试时只要测试 MockMenuAction 就好了

```
package com.company.web.struts.mock;

import com.company.web.struts.MenuAction;

/**
 * @author walter
 *
 */
public class MockMenuAction extends MenuAction {
    public MockMenuAction() {
        super();
            mockApp = new MockAppFacade();
        }

        private MockAppFacade mockApp;

  public MockAppFacade getApp() {
            return mockApp;
  }
}
```

2.1.2 在 MockAppFacade 中我们替换掉了实际的业务类

```
public class MockAppFacade implements AppFacade {
```

```
    public MockAppFacade() {
            super();
            //...
          mockMenuService       = new MockMenuService();
    }

        private MockUserService mockUserService;
        private MockMenuService mockMenuService;
        //...

    public MenuService getMenuService() {
          return mockMenuService;
    }

        //...
}
```

在 MockMenuService 类中,我们从预先写好的 CSV 文件中装载所需的模拟测试数据

```
package com.company.web.struts.mock;

//...
import com.company.business.MenuService;
import com.company.domain.Customer;
import com.company.domain.Module;
import com.company.test.TestData;

public class MockMenuService implements MenuService {

    public List<Customer> getCompanyByUser(long userID, long roleID) {
          return TestData.getCompanyByUser();
    }

        public List<Module> getMenuByRoleID(long roleID) {
            return TestData.getModuleList();
        }

        //...
}
```

这里使用了一个第三方的 csv 解析类 CsvReader

```
/**
```

```
 *
  */
package com.company.test;

import com.csvreader.CsvReader;
import com.company.domain.Module;
import com.company.domain.Site;
//...

/**
 * @author walter
 *
 */
public class TestData {

 static String path="./";
    static {
        if(!GWUtil.isFileExist(path+"test/data")){
              path="./projects/GW3_Page/";
        }
 }

     public static List<Module> getModuleList(){
       List<Module> moduleList=new ArrayList<Module>();

          try{
              CsvReader reader = new CsvReader(path+"test/data/menu.csv");

             reader.readHeaders();
             while (reader.readRecord())
             {
                   Module module=new Module();
                   module.setModuleID(Integer.parseInt(reader.get("moduleID")));
                    module.setUpModuleID(Integer.parseInt(reader.get("upModuleID")));
                    module.setModuleName(reader.get("moduleName"));
                     module.setModuleLink(reader.get("moduleLink"));
                     module.setModuleRight(Integer.parseInt(reader.get("moduleRight")));
                       module.setModuleStatus(Integer.parseInt(reader.get("moduleStatus")));
                    module.setModuleType(Integer.parseInt(reader.get("moduleType")));
                    moduleList.add(module);
               }
             reader.close();
          }catch(Exception e){
              e.printStackTrace();
```

```
        }
        return moduleList;
    }

    public static List<Customer> getCompanyByUser() {
        List<Customer> list=new ArrayList<Customer>();
        try{
            CsvReader reader = new CsvReader(path+"test/data/company.csv");

            reader.readHeaders();
            while (reader.readRecord())
            {
                String customerID = reader.get("customerID");
                String customerName = reader.get("customerName");

                Customer customer=new Customer();
                customer.setCustomerID(Long.parseLong(customerID));
                customer.setCustomerName(customerName);
                list.add(customer);
            }
            reader.close();
        }catch(Exception e){
            e.printStackTrace();
        }
        return list;
    }

    //...
}
```

所用到的 CSV 文件非常简单,可直接从 DB 中用 TOAD 导出或手工编写,避免了在测试类中编写冗长的代码来生成模拟数据 --test/data/menu.csv

```
moduleID,upModuleID,moduleName,moduleLink,moduleStatus,moduleRight,moduleType
0,0,Home,home.do,1,0,0
1,0,Network,null,1,0,0
...
```

2.1.3 针对 MockMenuAction 编写 MenuActionUnitTest 对其进行测试,其实也就是测试了 MenuAction,这个测试类是从 MockStrutsTestCase 中继承下来的

```
package com.company.web.struts;
import junit.framework.Test;
import junit.framework.TestCase;
import junit.framework.TestSuite;
```

```
import com.company.business.MeetingServiceTest;
import servletunit.struts.MockStrutsTestCase;
import com.company.web.struts.STCRequestProcessor;
//...
/**
 * The class <code>MenuActionUnitTest</code> contains tests for the class
 * {@link <code>MenuAction</code>}
 *
 * @pattern JUnit Test Case

 * @author walter.fan@gmail.com
 *
 * @version $Revision: 1.2 $
 */
public class MenuActionUnitTest extends MockStrutsTestCase {
 private static Log logger=LogFactory.getLog(MenuActionUnitTest.class);
     private HttpSession session;

    /**
     * Construct new test instance
    *
    * @param name the test name
      */
  public MenuActionUnitTest(String name) {
         super(name);
     }

     protected void setUp() throws Exception  {
        super.setUp();

          /* set session content */
        session = getSession();
       User user = new User();
       user.setUserID(1);
        user.setUserName("wbxsuper");
         user.setCustomerID(1);
        user.setRoleID(1);
        user.setRoleName("Super User");
       user.setActiveStatus(0);
           session.setAttribute(GWConstants.SESSION_USER_KEY, user);
        Privilege privilege = new Privilege();
       privilege.addPrivilege(0, 1); // Home module
      session.setAttribute(GWConstants.SESSION_PRIVILEGE_KEY, privilege);
```

```
        session.setAttribute(GWConstants.SESSION_COMPANYID_KEY, Long.parseLong("1"));
// WebEx Company

         /* set struts config file */
          setConfigFile("/WEB-INF/struts-config-unittest.xml,         /WEB-INF/conf/struts-config-
home.xml");
    }

     /**
      * Perform post-test clean up
      *
    * @throws Exception
       *
    * @see TestCase#tearDown()
   */
   protected void tearDown() throws Exception  {
          super.tearDown();

           /* clear session */
           Enumeration names = session.getAttributeNames();
       while (names.hasMoreElements()) {
            String name = (String) names.nextElement();
            session.removeAttribute(name);
           }
 }

     /**
      * Test MenuAction class doExecute method
      */
   public void testDoExecute() {
          STCRequestProcessor.addMockAction("com.company.web.struts.MenuAction",
"com.company.web.struts.mock.MockMenuAction");
        setRequestPathInfo("/menu");
         actionPerform();
           verifyForward(GWConstants.ACTIONMAPPING_SUCCESS);
        Map<String,                List<Customer>>                companyMap=(Map<String,
List<Customer>>)request.getAttribute(GWConstants.REQUEST_COMPMAP_KEY);
         List<Module>
menuList=(List<Module>)request.getAttribute(GWConstants.REQUEST_MENULIST_KEY);
        //logger.info("menuList="+menuList.size());
           //logger.info("companyMap="+companyMap.size());
           assertEquals(menuList.size(),28);
          assertEquals(companyMap.size(),27);
        assertNotNull(request.getAttribute(GWConstants.REQUEST_CUSTOMER_KEY));
```

```
        assertNotNull(request.getAttribute(GWConstants.REQUEST_MENU_USERNAME));
        assertNotNull(request.getAttribute(GWConstants.REQUEST_MENU_CURRENTDATE));

    }

    public static Test suite(){
        TestSuite suite=new TestSuite();
         suite.addTestSuite(MenuActionUnitTest.class);
          //suite.addTest(new MenuActionUnitTest("testDoExecute"));
        return suite;
    }

    public static void main(String[] args)
    {
        junit.textui.TestRunner.run(suite());
    }
}
```

注: 这里的 STCRequestProcessor 来自 IBM 的 Sunil Patil,扩展了 struts 的 RequestProcessor

```
package com.company.web.struts;

import java.io.IOException;
import java.util.HashMap;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.tiles.TilesRequestProcessor;
import org.apache.struts.util.RequestUtils;

/**
 * @author Sunil Patil of IBM Software Labs
 */
public class STCRequestProcessor extends TilesRequestProcessor {

  private static HashMap mockActionFormMap = new HashMap();
  private static HashMap mockActionMap = new HashMap();

    public static void addMockAction(String actionStr, String className) {
        mockActionMap.put(actionStr, className);
```

```
}

    public static void addMockActionForm(
        String actionFormStr,
         String className) {
      mockActionFormMap.put(actionFormStr, className);
}

    /**
    * We will insert Mock ActionForm for testing through this method
    */
protected ActionForm processActionForm(
        HttpServletRequest request,
     HttpServletResponse response,
        ActionMapping mapping) {


      // Create (if necessary a form bean to use
     String formBeanName = mapping.getName();
        String mockBeanClassName = (String) mockActionFormMap.get(formBeanName);
         if (mockBeanClassName == null)
               return super.processActionForm(request, response, mapping);

      ActionForm instance = null;
     try {
           Class formClass = Class.forName(mockBeanClassName );
            instance = (ActionForm) formClass.newInstance();
      } catch (ClassNotFoundException e) {
           e.printStackTrace();
      } catch (InstantiationException e) {
           e.printStackTrace();
      } catch (IllegalAccessException e) {
           e.printStackTrace();
      }

          instance.setServlet(servlet);
     if (instance == null) {
           return (null);
      }

          // Store the new instance in the appropriate scope
     if (log.isDebugEnabled()) {
               log.debug(
                   " Storing ActionForm bean instance in scope '"
```

```
                        + mapping.getScope()
                    + "' under attribute key '"
                     + mapping.getAttribute()
                    + "'");
        }
        if ("request".equals(mapping.getScope())) {
                request.setAttribute(mapping.getAttribute(), instance);
        } else {
               HttpSession session = request.getSession();
                session.setAttribute(mapping.getAttribute(), instance);
        }
        return (instance);

}

    /**
    * We will insert Mock Action Class through this method
 */
 protected Action processActionCreate(
         HttpServletRequest request,
        HttpServletResponse response,
           ActionMapping mapping)
            throws IOException  {
        String orignalClassName = mapping.getType();
           String mockClassName = (String)mockActionMap.get(orignalClassName);
        if( mockClassName == null )
                return super.processActionCreate(request,response,mapping);
           String className = mockClassName;
            if (log.isDebugEnabled()) {
                log.debug(" Looking for Action instance for class " + className);
           }

           Action instance = null;
         synchronized (actions) {

              // Return any existing Action instance of this class
                instance = (Action) actions.get(className);
                 if (instance != null) {
                   if (log.isTraceEnabled()) {
                            log.trace("    Returning existing Action instance");
                     }
                   return (instance);
               }
```

```
                // Create and return a new Action instance
            if (log.isTraceEnabled()) {
                    log.trace("    Creating new Action instance");
                }

                try {
                  instance = (Action) RequestUtils.applicationInstance(className);
                } catch (Exception e) {
                        log.error(
                          getInternal().getMessage("actionCreate", mapping.getPath()),
                          e);

                      response.sendError(
                          HttpServletResponse.SC_INTERNAL_SERVER_ERROR,
                          getInternal().getMessage(
                                  "actionCreate",
                                  mapping.getPath()));

                      return (null);
                }

                  instance.setServlet(this.servlet);
                actions.put(className, instance);
        }

            return (instance);
    }

}
```

--还需要在 struts 的配置文件中指明所用的 RequestRrocessor,如 struts-config-unittest.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 1.2//EN" "http://jakarta.apache.org/struts/dtds/struts-config_1_2.dtd">
<!--
            Created by Walter on 08/08/06 for GlobalWatch3.0
            plugin: validator, tile and Spring
    -->
<struts-config>
    <controller processorClass="com.company.web.struts.STCRequestProcessor"/>

      <message-resources parameter="com.company.web.ApplicationResources" null="false"/>
      <message-resources  key="alertRes"  parameter="com.company.web.ApplicationMessages" null="false"/>
```

```
<!-- ========================================== Plug Ins Configuration -->
    <plug-in className="org.apache.struts.tiles.TilesPlugin" >
        <!-- Path to XML definition file -->
        <set-property property="definitions-config" value="/WEB-INF/tiles-defs.xml" />
        <!-- Set Module-awareness to true -->
        <set-property property="moduleAware" value="true" />
    </plug-in>

<!-- ========================================== Validator plugin -->
    <plug-in className="org.apache.struts.validator.ValidatorPlugIn">
        <set-property    property="pathnames"    value="/WEB-INF/validator-rules.xml,/WEB-
INF/validation.xml"/>
    </plug-in>

<!-- ========================================== Spring plugin -->
  <plug-in className="org.springframework.web.struts.ContextLoaderPlugIn">
      <set-property property="contextConfigLocation"
          value="/WEB-INF/applicationContext.xml,/WEB-INF/dataAccessContext.xml"/>
   </plug-in>
</struts-config>
```

2.2 业务层

业务层的单元测试相比演示层,简单了很多,我们用 EasyMock 隔离模拟了对持久层的访问

```
package com.company.business;

import static org.easymock.EasyMock.createMock;
import static org.easymock.EasyMock.expect;
import static org.easymock.EasyMock.replay;

import java.util.ArrayList;
import java.util.List;

import junit.framework.Test;
import junit.framework.TestCase;
import junit.framework.TestSuite;

import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;

import com.csvreader.CsvReader;
import com.company.dao.MenuDao;
```

```
import com.company.domain.Module;
import com.company.test.TestData;

public class MenuServiceTest extends TestCase{

  private static Log logger=LogFactory.getLog(MenuServiceTest.class);
     private MenuServiceImpl menuService=new MenuServiceImpl();

  public MenuServiceTest(){
        super();
      }

       public MenuServiceTest(String s){
          super(s);
     }


     public void testGetMenuByRoleID()
  {
         List<Module> moduleListFromDao=TestData.getModuleListFromDao();
            List<Module> moduleList=TestData.getModuleList();

              long roleID=9;
           MenuDao menuDao=createMock(MenuDao.class);
           expect(menuDao.getMenuByRoleID(roleID)).andReturn(moduleListFromDao);
            menuService.setMenuDao(menuDao);
           replay(menuDao);

            List<Module> list=menuService.getMenuByRoleID(roleID);
           assertEquals(list,moduleList);
    }

       public static Test suite(){
          TestSuite suite=new TestSuite();
            suite.addTestSuite(MenuServiceTest.class);
            //suite.addTest(new MenuServiceInteTest("testGetMenuByRoleID"));
          return suite;
    }

     public static void main(String[] args) {
             TestCase test=new MenuServiceTest("testGetMenuByRoleID");
             junit.textui.TestRunner.run(test);
      }
}
```

2.3. 持久层

持久层的测试比较麻烦,因为依赖于数据库中的相关数据,我们编写了一个 SQL 文件,在运行持久层单元测试前运行一下,以装载所需的数据.具体实现是在 Ant 运行脚本 build.xml 中添加 runsql 目标

```
<target name="runsql" unless="testcase" >
        <sql driver="${jdbc.driverClassName}"
                        url="${jdbc.url}"
                        userid="${jdbc.username}"
                        password="${jdbc.password}"
                    src="test/testInitData.sql"
                    classpathref="master-classpath"/>
</target>
```

--testInitData.sql

```
--...
DELETE FROM gwuser WHERE username='userdaotest';
INSERT INTO gwuser(username,...)VALUES('userdaotest',...);
commit;
--...
```

实际的测试代码也很简单,TestCase 扩展自 Spring 的 AbstractTransactionalSpringContextTests

```
package com.company.dao;


import com.company.business.NoResultsFoundException;
import com.company.dao.MenuDao;
import org.springframework.test.*;
import com.company.util.GWUtil;
import java.util.*;

import com.company.domain.Customer;
import com.company.domain.Module;


public class MenuDaoTest extends AbstractTransactionalSpringContextTests {
    private MenuDao menuDao;

    public void setMeetingDao(MenuDao menuDao){
       this.menuDao = menuDao;
    }
```

```
    protected String[] getConfigLocations(){
        return new String[]{"test/applicationContext.xml"};
    }

    protected void onSetUpBeforeTransaction(){

}

    protected void onTearDownAfterTransaction(){

  }

    public void testSaveAccountActivity(){
      Calendar cale = Calendar.getInstance();
     cale.set(2006,9,10,10,10,10);
     int ret = this.menuDao.saveAccountActivity(100, 100, "test", cale.getTime(), "logout");
     assertEquals(1, ret);

    }

    public void testGetMenuByRoleID(){
      List<Module> list = this.menuDao.getMenuByRoleID(2);
        assertNotNull(list);
    }


  public void testGetCompanyByUser(){
    List<Customer> list = this.menuDao.getCompanyByUser(4, 9);
        assertNotNull(list);

        List list1 = null;
        try{
            list1 = this.menuDao.getCompanyByUser(99999999, 999999999);
    }catch(NoResultsFoundException e){

    }
    assertNull(list1);
  }

}
```

3. 单元测试与持续集成

3.1 应用 Emma 统计测试覆盖率

```
<taskdef resource="emma_ant.properties" classpathref="emma.lib" />

<target name="compile.emma" depends="compile.src">
<emma>
   <instr instrpath="${build.dir}/classes/com/webex/globalwatch/web/struts"
                  destdir="${build.dir}/emma/com/webex/globalwatch/web/struts"
             metadatafile="${build.dir}/emma/metadata.emma"
                  merge="true">
             <filter excludes="GenericBaseAction, *ValidationUtil, BaseActionForm" />
      </instr>
   <instr instrpath="${build.dir}/classes/com/webex/globalwatch/business"
                destdir="${build.dir}/emma/com/webex/globalwatch/business"
                  metadatafile="${build.dir}/emma/metadata.emma"
                  merge="true">
             <filter excludes="NoResultsFoundException, IntegrityViolationException" />
   </instr>
   <instr instrpath="${build.dir}/classes/com/webex/globalwatch/util"
                destdir="${build.dir}/emma/com/webex/globalwatch/util"
                  metadatafile="${build.dir}/emma/metadata.emma"
                  merge="true">
             <filter excludes="GWConstants, GWLDAP, GWTypeDefinition" />
  </instr>
   <instr instrpath="${build.dir}/classes/com/webex/globalwatch/dao"
                  destdir="${build.dir}/emma/com/webex/globalwatch/dao"
                   metadatafile="${build.dir}/emma/metadata.emma"
                  merge="true">
      </instr>
   <!--
      <instr instrpath="${build.dir}/classes/com/webex/globalwatch/domain"
            destdir="${build.dir}/emma/com/webex/globalwatch/domain"
            metadatafile="${build.dir}/emma/metadata.emma"
               merge="true">

           <filter includes="Privilege, DataSet" />
      </instr>
   -->
</emma>
</target>
<target name="test.emma" depends="compile.test, compile.emma">
<antcall target="runsql"/>
<junit printsummary="yes" haltonerror="no" haltonfailure="no" fork="yes">
   <classpath>
          <path refid="master-classpath"/>
           <path refid="emma.lib"/>
```

```
            <pathelement path="${build.dir}/test"/>
          <pathelement path="${build.dir}/emma"/>
          <pathelement path="${build.dir}/classes"/>
    </classpath>
    <jvmarg value="-Demma.coverage.out.file=${emma.output.file}"/>
    <jvmarg value="-Demma.coverage.out.merge=true"/>
       <formatter type="plain"/>
      <formatter type="xml"/>
  <batchtest fork="yes" todir="${docs.dir}/results/junit" unless="testcase">
          <fileset dir="${build.dir}/test">
                <include name="**/*Test.*"/>
                 <!--<include name="**/*UnitTest.*"/>

                 <exclude name="**/*InteTest.*"/>-->
          </fileset>
       </batchtest>
    <batchtest fork="yes" todir="${docs.dir}/results/junit" if="testcase">
          <fileset dir="${build.dir}/test">
                <include name="**/*${testcase}*"/>
          </fileset>
       </batchtest>
</junit>
<junitreport todir="${docs.dir}/reports/junit">
  <fileset dir="${docs.dir}/results/junit">
           <include name="TEST-*.xml"/>
       </fileset>
       <report format="frames" todir="${docs.dir}/reports/junit/html"/>
</junitreport>
<copy todir="${docs.dir}/reports/emma/dev" overwrite="true">
       <fileset file="${build.dir}/emma/metadata.emma" />
</copy>
<copy todir="${docs.dir}/reports/emma/dev" overwrite="true">
  <fileset file="${docs.dir}/results/emma/coverage.emma" />
</copy>
<emma>
<report sourcepath="${src.dir}" depth="method">
           <fileset dir="${docs.dir}/reports/emma/dev" >
                <include name="*.emma" />
           </fileset>
            <!-- <xml outfile="${coverage.dir}/coverage.xml" depth="package" />
  -->
    <html outfile="${emma.report.file}"
                depth="method"
                    columns="name,class,method,block,line" >
```

```
        </html>
    </report>
</emma>
</target>
```

3.2 应用 CruiseControl 进行持续集成

4 性能测试

1 利用 BadBoy 录制测试脚本
2 利用 Jmeter 进行性能压力测试

5 收获与教训

单元测试的加强大大提高了 Fix bug 的效率,减少了 regression bug,在重现和解决问题时节省了大量用于调试的时间,项目后期很少进行单步或断点调试,也不必劳动庞大的 JBoss,只要简单运行相关的 TestCase,加点断言或打一点 log 就搞定了

单元测试做得不够充分,相对于编码有点滞后,对于持久层测试采用的 testInitData.sql,应该有更好的办法,比如 DBUnit

6 参考文档

http://starrynight.blogdriver.com/starrynight/565540.html

http://www.ibm.com/developerworks/cn/java/j-stc/