原 dbunit实践指南 收藏

- 什么是**dbunit**以及为什么要使用它?

dbunit是一个基于junit扩展的数据库测试框架。它提供了大量的类对与数据库相关的操作进行了抽象和封装，虽然在80%的情况，你只需使用它极少的api。它通过使用用户自定义的数据集以及相关操作使数据库处于一种可知的状态，从而使得测试自动化、可重复和相对独立。虽然不用dbunit也可以达到这种目的，但是我们必须为此付出代价（编写大量代码，测试及维护），既然有了这么优秀的开源框架，我们又何必再造轮子。

- **dbunit**的原理

dbunit的与单元测试相关的两个最重要的核心是org.dbunit.database.IDatabaseConnection和 org.dbunit.dataset.IDataSet ，前者是产品代码使用的数据库连接的一个简单的封装，后者是对单元测试人员自定义的数据集（通常以xml文件的形式存在，且xml文件的格式也有好几种）的封装。

还有一个很重要的咚咚就是org.dbunit.operation.DatabaseOperation，该类是一个抽象类代表了对数据库的操作，例如CUD以及其组合等， 它采用了退化的工厂模式，可直接通过它获取其具体的子类（代表具体的某种操作）如下：

DatabaseOperation.UPDATE
DatabaseOperation.DELETE
DatabaseOperation.DELETE_ALL

search engine
server
web
翻译

链接

http://script.aculo.us/

http://www.psytopic.com/

一个藏袍

存档

2009年12月(6)
2009年11月(1)
2009年10月(8)
2009年09月(3)
2009年06月(1)
2008年11月(1)
2008年10月(5)
2008年09月(1)
2008年06月(4)
2008年04月(3)
2008年03月(2)
2008年01月(3)
2007年12月(8)
2007年11月(5)
2007年10月(2)
2007年07月(3)
2007年06月(2)
2007年05月(4)
2007年04月(6)
2007年03月(8)
2007年02月(3)
2007年01月(1)

DatabaseOperation.TRUNCATE
DatabaseOperation.REFRESH
DatabaseOperation.CLEAN_INSERT
DatabaseOperation.NONE

工作流程如下：
1)testcase.setup--->testcase.getConnection-->getDataSet----->operation.execute(
通常DatabaseOperation.CLEAN_INSERT)
2)testcase.testSomeMethod---->dao.someMethod
3)testcase.teardown---->operation.execute(
通常DatabaseOperation.DELETE_ALL或者DatabaseOperation.NONE)

- 实战

以一个真实的系统（share@mofile)为例，

1. 建立一个测试数据库
2. 写自定义的数据

```
<?xml version='1.0' encoding='UTF-8'?>

<dataset>

        <FileType typeId='1' typeName='type1' />

        <FileType typeId='2' typeName='type2' />

        <FileType typeId='3' typeName='type3' />

        <UserInSharedSystem userId='1' loginName='2' userNo='a' />

        <DefaultFile fileId='1' fileName='a1' fileSize='10' fileStatus='1'
                pickupCode='4414402888619758' releaseDate='2006-1-31 16:17:18'
                storageFileHome='solar' storageFileId='1' userId='1' typeId='1'
                description='ss' expiredDate='2006-1-31 16:17:18'
                uploadDate='2006-1-31 16:17:18' extName='exe' totalPoints='0'
                currentMonthPoints='0' lastMonthPoints='0' />
```

```
        <DefaultFile fileId='2' fileName='a2' fileSize='20' fileStatus='1'
                pickupCode='4414402888619759' releaseDate='2006-1-31 16:17:18'
                storageFileHome='solar' storageFileId='2' userId='1' typeId='2'
                description='ss' expiredDate='2006-1-31 16:17:18'
                uploadDate='2006-1-31 16:17:18' extName='exe' totalPoints='0'
                currentMonthPoints='0' lastMonthPoints='0' />

        <DefaultFile fileId='3' fileName='a3' fileSize='30' fileStatus='1'
                pickupCode='4414402888619768' releaseDate='2006-1-31 16:17:18'
                storageFileHome='solar' storageFileId='3' userId='1' typeId='3'
                description='ss' expiredDate='2006-1-31 16:17:18'
                uploadDate='2006-1-31 16:17:18' extName='exe' totalPoints='0'
                currentMonthPoints='0' lastMonthPoints='0' />

        <FileTag tagId='1' tagName='t1' />

        <FileTag tagId='2' tagName='t2' />

        <FileTag tagId='3' tagName='t3' />


        <FileTagRel fileId='1' tagId='1' />

        <FileTagRel fileId='1' tagId='2' />

        <FileTagRel fileId='2' tagId='1' />

        <FileTagRel fileId='2' tagId='3' />

        <FileTagRel fileId='3' tagId='2' />

</dataset>
```

注意数据的顺序，否则会有约束违例，特别是外键约束

3. 写基本的测试类

```
package mofile.share.dao;
```

```
import javax.sql.DataSource;



import mofile.common.utils.SpringBeanProxy;


import org.dbunit.DatabaseTestCase;
import org.dbunit.database.DatabaseConnection;
import org.dbunit.database.IDatabaseConnection;
import org.dbunit.dataset.IDataSet;
import org.dbunit.operation.DatabaseOperation;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

/**
* 使用dbunit
*
* @author weip
* @time 2006-3-9 13:42:32
*
*/
public abstract class BaseDaoTest extends DatabaseTestCase {

	protected final static ApplicationContext ctx;

	protected IDatabaseConnection conn = null;

	protected IDataSet dataSet = null;

	static {
		// String pkg = ClassUtils.classPackageAsResourcePath(Constants.class);
		String[] paths = { "classpath:applicationContext-share-database.xml",
			"classpath:applicationContext-share-hibernate.xml",
			"applicationContext-share-property.xml"/*
				* ,
				* "classpath:applicationContext-central-database.xml"
```

```
                        */};
        ctx = new ClassPathXmlApplicationContext(paths);
        SpringBeanProxy.setApplicationContext(ctx);
}

public BaseDaoTest(String methodName) {
        super(methodName);
}

/**
*
* @author weip
* @time 2006-3-9 13:46:45 (non-Javadoc)
* @see org.dbunit.DatabaseTestCase#getConnection()
*
*/

protected IDatabaseConnection getConnection() throws Exception {
        DataSource ds = (DataSource) ctx.getBean("shareDataSource");

conn = new DatabaseConnection(ds.getConnection());

return conn;
}

/**
*
* @author weip
* @time 2006-3-9 13:52:27 (non-Javadoc)
* @see org.dbunit.DatabaseTestCase#getSetUpOperation()
*
*/

protected DatabaseOperation getSetUpOperation() throws Exception {
        return DatabaseOperation.CLEAN_INSERT;
}
```

```
        /**
        *
        * @author weip
        * @time 2006-3-9 13:52:31 (non-Javadoc)
        * @see org.dbunit.DatabaseTestCase#getTearDownOperation()
        *
        */

        protected DatabaseOperation getTearDownOperation() throws Exception {
                //this.closeConnection(conn);
                //conn=null;
                return DatabaseOperation.NONE;
        }


}
```

4. 写具体的测试用例

```
package mofile.share.dao;

import java.io.FileInputStream;
import java.util.List;

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

import mofile.share.domain.DefaultFile;
import mofile.share.domain.DefaultUser;
import mofile.share.domain.FileTag;
import mofile.share.domain.FileType;
import mofile.share.vo.FileForm;

import org.dbunit.dataset.IDataSet;
import org.dbunit.dataset.xml.FlatXmlDataSet;
import org.dbunit.operation.DatabaseOperation;

/**
```

```
*
* @author weip
* @time 2006-3-9 11:13:34
*
*/
public class FileDaoTest extends BaseDaoTest {
	private FileDao fileDao;

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

		/*try {
			DatabaseOperation.CLEAN_INSERT.execute(conn, dataSet);
		} finally {
			conn.close();
		}*/
		fileDao = (FileDao) ctx.getBean("fileDao");
	}

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

	public FileDaoTest(String methodName) {
		super(methodName);
	}

	public static Test suite() {
		TestSuite suite = new TestSuite();

	suite.addTest(new FileDaoTest("testQueryAllFiles"));
	suite.addTest(new FileDaoTest("testQueryFilesByFileType"));

	suite.addTest(new FileDaoTest("testAddFiles"));

	suite.addTest(new FileDaoTest("testRemoveObject"));
```

```
return suite;
}

/**
*
*  @author weip
*  @time 22:53:34 2006-3-16
*  @see org.dbunit.DatabaseTestCase#getDataSet()
*/
protected IDataSet getDataSet() throws Exception {
        dataSet = new FlatXmlDataSet(new FileInputStream("d:/Projects/Share/Mofile_share/htdocs/WEB-I
        NF/classes/mofile/share/dao/testFileDB.xml"));
        return dataSet;

}

/**
*
* @author weip
* @time 22:53:17 2006-3-16
* @throws Exception
*/
public void testAddFiles() throws Exception {
        FileType fileType = new FileType();
        fileType.setTypeId(new Integer(1));

FileTag fileTag = new FileTag();
fileTag.setTagName("dev3");

DefaultUser user = new DefaultUser();
// user.setLoginName("");
// user.setUserNo(new Long(1));
user.setUserId(new Long(1));

DefaultFile file = new DefaultFile();

file.setFileName("file001");
```

```
file.setFileSize(20);
file.setFileStatus(1);
file.setPickupCode("0001");
file.setStorageFileHome("xxx");
file.setStorageFileId(new Long(1));
file.setUser(user);
file.setFileType(fileType);
file.addTag(fileTag);

fileDao.saveObject(file);

DefaultFile fileafter=fileDao.getFileByStorageFileId("xxx",new Long(1));

assertNotNull(fileafter);

assertEquals("file001",fileafter.getFileName());


}

/**
*
* @author weip
* @time 2006-3-9 14:01:05
* @throws Exception
*/
public void testQueryAllFiles() throws Exception {
        FileForm frm = new FileForm();
        frm.setFileStatus(-1);
        frm.setFileType(-1);

List list = fileDao.queryFile(frm);

assertEquals(3, list.size());

}
```

```
/**
*
* @author weip
* @time 22:47:01 2006-3-16
* @throws Exception
*/
public void testQueryFilesByFileType() throws Exception {
        FileForm frm = new FileForm();
        frm.setFileStatus(-1);
        frm.setFileType(1);

List list = fileDao.queryFile(frm);

assertEquals(1, list.size());

}

/**
*
* @author weip
* @throws Exception
* @time 23:10:33 2006-3-16
*/
public void testRemoveObject() throws Exception {

fileDao.removeObject(DefaultFile.class, new Long(2));

FileForm frm = new FileForm();
frm.setFileStatus(-1);
frm.setFileType(-1);

List list = fileDao.queryFile(frm);

assertEquals(2, list.size());

}
```

```
}
```

对比以下以前写的测试用例

```
public class FileDaoImplTestCase extends BaseDaoTestCase {
	private FileDao fileDao;

	protected void setUp() throws Exception {
		super.setUp();
		fileDao = (FileDao) ctx.getBean("fileDao");
	}

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

	public FileDaoImplTestCase(String methodName) {
		super(methodName);
	}

	public static Test suite() {
		TestSuite suite = new TestSuite();
		// suite.addTest(new FileDaoImplTest("testRemoveObject"));

	// suite.addTest(new UserDaoImplTest("testSaveObject"));
	// suite.addTest(new FileDaoImplTestCase("testSaveObject"));
	// suite.addTest(new FileDaoImplTestCase("testQueryFileByTag"));
	//suite.addTest(new FileDaoImplTestCase("testQueryFile4Pagenation"));
	suite.addTest(new FileDaoImplTestCase("testUpdateLastMonthPoints"));

	return suite;
	}

	/*
	* Test method for
	* 'mofile.share.dao.impl.BaseHibernateDaoImpl.getObject(Class,
	* Serializable)'
```

```
*/
public void testGetObject() {

}

/**
* 插入一条文件记录 ，注意插入记录需检查是否存在重复的filetag
*
* @author weip
* @time 16:42:20 2006-1-15
* @throws Exception
*
* Hibernate: insert into DefaultFile (fileName, fileSize, fileStatus,
* pickupCode, releaseDate, storageFileHome, storageFileId, userId, typeId)
* values (?, ?, ?, ?, ?, ?, ?, ?, ?) Hibernate: insert into FileTag
* (tagName) values (?) Hibernate: insert into FileTagRel (fileId, tagId)
* values (?, ?)
*
*/
public void testSaveObject() throws Exception {
        FileType fileType = new FileType();
        fileType.setTypeId(new Integer(1));

FileTag fileTag = new FileTag();
fileTag.setTagName("dev3");

DefaultUser user = new DefaultUser();
// user.setLoginName("");
// user.setUserNo(new Long(1));
user.setUserId(new Long(1));

DefaultFile file = new DefaultFile();

file.setFileName("file001");
file.setFileSize(20);
file.setFileStatus(1);
file.setPickupCode("0001");
```

```
file.setStorageFileHome("xxx");
file.setStorageFileId(new Long(1));
file.setUser(user);
file.setFileType(fileType);
file.addTag(fileTag);

fileDao.saveObject(file);

}

/**
*
* 删除一条文件记录
*
* @author weip
* @time 16:47:29 2006-1-15
*
* Hibernate: select defaultfil0_.fileId as fileId2_, defaultfil0_.fileName
* as fileName2_, defaultfil0_.fileSize as fileSize2_,
* defaultfil0_.fileStatus as fileStatus2_, defaultfil0_.pickupCode as
* pickupCode2_, defaultfil0_.releaseDate as releaseD6_2_,
* defaultfil0_.storageFileHome as storageF7_2_, defaultfil0_.storageFileId
* as storageF8_2_, defaultfil0_.userId as userId2_, defaultfil0_.typeId as
* typeId2_, defaultuse1_.userId as userId0_, defaultuse1_.loginName as
* loginName0_, defaultuse1_.userNo as userNo0_, filetype2_.typeId as
* typeId1_, filetype2_.typeName as typeName1_ from DefaultFile defaultfil0_
* left outer join UseInSharedSystem defaultuse1_ on
* defaultfil0_.userId=defaultuse1_.userId left outer join FileType
* filetype2_ on defaultfil0_.typeId=filetype2_.typeId where
* defaultfil0_.fileId=? Hibernate: select tagset0_.fileId as fileId__,
* tagset0_.tagId as tagId__, filetag1_.tagId as tagId0_, filetag1_.tagName
* as tagName0_ from FileTagRel tagset0_ inner join FileTag filetag1_ on
* tagset0_.tagId=filetag1_.tagId where tagset0_.fileId=? Hibernate: delete
* from FileTagRel where fileId=? Hibernate: delete from DefaultFile where
* fileId=?
*/
public void testRemoveObject() {
```

```
fileDao.removeObject(DefaultFile.class, new Long(2));

}

/**
* 测试按标签获取文件
*
* @author weip
* @throws Exception
* @time 2006-1-24 20:12:07
*/
public void testQueryFileByTag() throws Exception {

List list = fileDao.queryFileByTag(54, 0, 10);

assertNotNull(list);

DefaultFile file = null;
if (list.size() > 0)
        file = (DefaultFile) list.get(0);
}

public void testQueryFile4Pagenation() throws Exception {
        FileForm frm = new FileForm();
        frm.setFileStatus(-1);
        frm.setFileType(-1);
        fileDao.queryFileSum(frm);
}

/**
* 测试 批量更新上月得分总数
*
* @author weip
* @throws Exception
* @time 2006-2-21 16:07:32
*/
```

```
        public void testUpdateLastMonthPoints() throws Exception {
                fileDao.updateLastMonthPoints();
                assertTrue(true);

        }

}
```

- 总结

通过与以前的例子比较发现：使用dbunit的testcase更自动化和可重复，以前写的testcase与数据库中的数据严重耦合，所以一般都不敢写断言，写了之后怕数据又发生变化，所以测试也是不可重复，并且也不是自动化，因为没有断言，你不得不测试完之后还得检查数据库。

当然dbunit也许并不是银弹，它在并发测试的时候得表现我没有实践过，也不敢妄下断言，而且是不是应该另外再建一个同样的数据专门测试dao还值得思考
我们在项目中为每个开发人员自建一个数据库解决并发问题,也许这个方案并非最佳,但实用

发表于 @ 2006年05月08日 15:37:00 | 评论(7) | 举报 | 收藏

旧一篇: org.apache.commons.logging的用法（下) | 新一篇:EasyMock实践指南

xiaodongx □□□Thursday, May 11, 2006 21:50:00 □□ □□

□□□□□□□□□□□□□□□□□□□□?

pwlazy EX □□□Saturday, May 13, 2006 05:23:00 □□ □□

□□□xiaodongx

□□

□□□□□□□ □□□Wednesday, December 17, 2008 10:41:10 □□ □□

□□□□□□□

□□□□□□□□□□ □□□Thursday, January 01, 2009 11:38:00 □□ □□

□□□□□□□□

sunshinme □□□Monday, September 14, 2009 17:51:14 □□ □□

□□ □□□□□□□ □□□□□□□

sunshinme □□□Monday, September 14, 2009 18:04:14 □□ □□

□□ public class DB{ private Connection conn□ public DB□Connection conn1□{ this.conn=conn1□} public UserInfo queryByID(int userID) { UserInfo user=null; PreparedStatement pstmt = null; ResultSet resultSet = null; try{ pstmt = conn.prepareStatement("select * from user_tb where id=?"); pstmt.setInt(1, userID); □□□□□□□□□□□□□□□□ }catch(SQLException e){} } } □□□□queryByID,□□□□□

pwlazy EX □□□Monday, September 14, 2009 19:32:34 □□ □□

1□ dbunit□□□□□□□□□□ ,□□ □□□□□□□□□□□□□dbunit,□□□□□□□□□□□□□□□□ 2□ □□□□□□□□□□□□ protected IDatabaseConnection getConnection() throws Exception { conn = new DatabaseConnection(ds.getConnection()); return conn; } □□□□□□□□□□□□□□□□□□□□ds.getConnection()

发表评论 “评论王争夺赛”第3期活动开始啦！

表 情:

评论内容:

用 户 名: 匿名用户 登录 注册

验 证 码:  重新获得验证码

发表评论