yoursunlight-qq-com

岁月如歌

keep calm | smile | focus | prospect

2010-06-24 通过网页

>>更多闲聊

浏览: 2297 次

性别:

来自: 成都

我现在离线

详细资料 留言簿

搜索本博客

搜索

最近访客 >>更多访客

hzhello guxue365

lgsoftwind b_l_east

博客分类

- 全部博客 (36)
- Spring 3 (1)
- ajax (0)
- jquery (10)
- ibatis 3 (0)
- oracle (1)
- mysql (0)
- DB (1)
- 报表 (1)
- js (1)

SQL的级联更新与级联删除 | Spring标签<form:checkbox>用法

2010-05-31

Spring 3.0 OXM - Spring 3.0 框架新特性

文章分类:Java编程

什么是OXM?

OXM是Object-to-XML-Mapping的缩写，它是一个O/M-mapper，将java对象映射成XML数据，或者将XML数据映射成java对象。它类似 XML-Marshalling 或者是 XML-Serialization，并不是什么新技术。目前Spring framework 3.0引入了该特性。

现在让我们看看Spring的承诺:

- 易于配置
- 一致的接口
- 一致的异常层次结构

听起来很不错，因为没有喜欢杂乱的接口，繁琐的配置，现在就让我们来检验一下Spring OXM的设计思想，本文着重讲述OXM特性的使用方法。

OXM in Spring 3.0

我们以一个Maven2实例工程来进行说明。首先，创建一个pom.xml文件:

Xml代码

```xml
<project>
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.unitedcoders.examples</groupId>
    <artifactId>SpringOxmExample</artifactId>
    <packaging>jar</packaging>
    <version>1.0.0-SNAPSHOT</version>
    <name>SpringOxmExample</name>
    <url>http://united-coders.com</url>
    <build>
        <finalName>SpringOxmExample</finalName>
        <plugins>
            <!-- Compiler Plugin to compile on Java 1.6 -->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <configuration>
                    <compilerVersion>1.6</compilerVersion>
                    <fork>true</fork>
                    <source>1.6</source>
                    <target>1.6</target>
                </configuration>
            </plugin>
        </plugins>
    </build>

    <!-- Properties -->
    <properties>
        <spring-version>3.0.0.RELEASE</spring-version>
    </properties>

    <!-- Dependencies -->
    <dependencies>
        <!-- Spring framework -->
        <dependency>
            <groupId>org.springframework</groupId>
```

- 插件 (1)
- 工具 (2)
- linux (5)
- java (6)

其他分类

- 我的收藏 (37)
- 我的论坛主题贴 (0)
- 我的所有论坛贴 (0)
- 我的精华良好贴 (0)

最近加入圈子

存档

- 2010-07 (4)
- 2010-06 (24)
- 2010-05 (8)
- 更多存档...

最新评论

- 详解 Spring 3.0 基于 Ann ...

  不知道3.0对泛型的扫描注入支持完善了没有,现在用2.5有点抓狂了..

  -- by tidelgl

评论排行榜

- 详解 Spring 3.0 基于 Annotation 的依赖注 ...
- 通过request.getContextPath获取绝对路径
- 对REST中无状态(stateless)的理解
- REST是什么[精品]
- JavaScript定义函数的方法

RSS Add to Google 订阅到 鲜果

[什么是RSS?]


```
36.             <artifactId>spring-core</artifactId>
37.             <version>${spring-version}</version>
38.         </dependency>
39.         <dependency>
40.             <groupId>org.springframework</groupId>
41.             <artifactId>spring-beans</artifactId>
42.             <version>${spring-version}</version>
43.         </dependency>
44.         <dependency>
45.             <groupId>org.springframework</groupId>
46.             <artifactId>spring-oxm</artifactId>
47.             <version>${spring-version}</version>
48.         </dependency>
49.
50.         <!-- Castor Xml -->
51.         <dependency>
52.             <groupId>org.codehaus.castor</groupId>
53.             <artifactId>castor</artifactId>
54.             <version>1.2</version>
55.         </dependency>
56.         <dependency>
57.             <groupId>xerces</groupId>
58.             <artifactId>xercesImpl</artifactId>
59.             <version>2.9.1</version>
60.         </dependency>
61.
62.         <!-- Logging -->
63.         <dependency>
64.             <groupId>commons-logging</groupId>
65.             <artifactId>commons-logging</artifactId>
66.             <version>1.1.1</version>
67.         </dependency>
68.
69.         <!-- Test dependencies -->
70.         <dependency>
71.             <groupId>junit</groupId>
72.             <artifactId>junit</artifactId>
73.             <version>4.4</version>
74.             <scope>test</scope>
75.         </dependency>
76.     </dependencies>
77. </project>
```

该实例中我用到了三个spring 模块,spring-core,spring-beans以及spring-oxm。另外，我使用Castor工程和Apache Xerces作为XML-Marshalling-dependencies.

接着，创建一个简单的javabean，将被序列换成XML字符串：

Java代码

```
1.  package com.unitedcoders.examples.spring.oxm.beans;
2.
3.  public class Person {
4.
5.      private String firstname;
6.
7.      private String lastname;
8.
9.      private boolean developer;
10.
11.     public String getFirstname() {
12.         return firstname;
13.     }
14.
15.     public void setFirstname(String firstname) {
16.         this.firstname = firstname;
```

```
    }

    public String getLastname() {
        return lastname;
    }

    public void setLastname(String lastname) {
        this.lastname = lastname;
    }

    public boolean isDeveloper() {
        return developer;
    }

    public void setDeveloper(boolean developer) {
        this.developer = developer;
    }

}
```

然后，为O/X-Mapper创建一个接口：

Java代码

```
package com.unitedcoders.examples.spring.oxm.mapper;

import java.io.IOException;

public interface OxMapper {

    /**
     * Serializes assigned Object into a file with the assigned name.
     *
     * @param object
     *             - Object that should be serialized
     * @param filename
     *             - name of the XML-file
     * @throws IOException
     */
    public abstract void writeObjectToXml(Object object, String filename) throws IOException;

    /**
     * Deserializes an object from the assigned file.
     *
     * @param filename
     *             - name of the file that should be deserialized
     * @return deserialized object
     * @throws IOException
     */
    public abstract Object readObjectFromXml(String filename) throws IOException;

}
```

我定义了两个方法，一个将对象转化成xml文件，一个将xml文件转换成对象。以下是该接口的实现类：

Java代码

```
package com.unitedcoders.examples.spring.oxm.mapper;

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

import javax.xml.transform.stream.StreamResult;
import javax.xml.transform.stream.StreamSource;

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

```java
import org.springframework.oxm.Marshaller;
import org.springframework.oxm.Unmarshaller;
import org.springframework.oxm.XmlMappingException;

public class OxMapperImpl implements OxMapper {

    private static final Log LOG = LogFactory.getLog(OxMapperImpl.class);

    private Marshaller marshaller;

    private Unmarshaller unmarshaller;

    public void writeObjectToXml(Object object, String filename) throws IOException {
        FileOutputStream fos = null;
        try {
            fos = new FileOutputStream(filename);
            marshaller.marshal(object, new StreamResult(fos));
        } catch (XmlMappingException e) {
            LOG.error("Xml-Serialization failed due to an XmlMappingException.", e);
        } catch (IOException e) {
            LOG.error("Xml-Serialization failed due to an IOException.", e);
        } finally {
            if (fos != null) {
                fos.close();
            }
        }
    }

    public Object readObjectFromXml(String filename) throws IOException {
        FileInputStream fis = null;
        try {
            fis = new FileInputStream(filename);
            return unmarshaller.unmarshal(new StreamSource(fis));
        } catch (IOException e) {
            LOG.error("Xml-Deserialization failed due to an IOException.", e);
        } finally {
            if (fis != null) {
                fis.close();
            }
        }
        return null;
    }

    public Marshaller getMarshaller() {
        return marshaller;
    }

    public void setMarshaller(Marshaller marshaller) {
        this.marshaller = marshaller;
    }

    public Unmarshaller getUnmarshaller() {
        return unmarshaller;
    }

    public void setUnmarshaller(Unmarshaller unmarshaller) {
        this.unmarshaller = unmarshaller;
    }

}
```

待续。。。

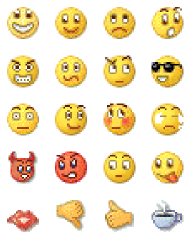
SQL的级联更新与级联删除 | Spring标签<form:checkbox>用法

01:11 | 浏览 (131) | 评论 (0) | 分类: Spring 3 | 相关推荐

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