

动态 ActionForm 在 Struts 框架中的应用研究

Application Research of DynaActionForm in Struts Framework

张文梅 廖福保 (广东农工商职业技术学院 广东 广州 510507)

摘要: Struts 是当前 Web 应用开发中最为流行的框架之一,在 Struts 框架中 ActionForm 对应于页面表单,随着表单增多,ActionForm 也会增多。由此结合具体开发实例讨论了动态 ActionForm 在 Struts 中的应用,不必创建一个具体的 ActionForm 类,而由 Struts 框架通过配置文件动态创建 ActionForm。

关键词: ActionForm 动态 ActionForm Struts 框架

1 引言

当前, J2EE 正逐渐成为 Web 应用软件的开发标准和规范,它提供了一种在分布式环境下基于组件的设计、开发、集成、部署的企业应用系统的方法。为了提高分布式应用系统中组件的重用能力和事务控制的灵活性,一般采用 MVC 模式来分离系统的用户界面、业务逻辑及业务控制, Struts 体系结构就是 MVC 设计模式的一种具体实现,它已成为 Web 应用开发中最为流行的框架之一。

Struts 为模型部分提供了 ActionForm 对象。ActionForm 对象可以通过定义属性来描述客户端表单数据,利用 ActionForm 对象和 Struts 提供的自定义标签库可以实现对客户端表单数据的良好封装和支持, ActionForm 对象实现了视图和模型之间的交互。程序开发人员要为每个表单都创建一个 ActionForm,以维护 Web 应用程序的会话状态。然而随着表单的增多, ActionForm 也要随着增多。

本文就此介绍如何利用动态 ActionForm (DynaActionForm) 来减少 ActionForm 的数目,这样就不必创建一个具体的 ActionForm 类,只需要在配置文件中配置出所需的虚拟 ActionForm,由 Struts 框架通过配置文件动态创建 ActionForm,从而能提高开发效率,简化编程。本文最后给出了一个利用动态 ActionForm 进行开发的实例。

2 Struts 的体系结构及工作流程

Struts 由一组相互协作的类(组件), Servlet 以及 JSP tag lib 组成,其中, Servlet 用于 HTTP 请求的分配及 JSP 的调用;标签库用于页面的动态生成;图 1 显示了 Struts 的基本构成及工作流程。

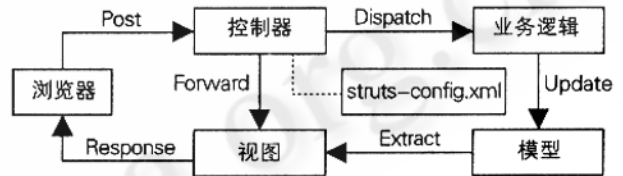


图 1 MVC 的 Struts 实现

(1) Client Browser(客户浏览器)。来自客户浏览器的 HTTP 请求。

(2) Controller(控制器)。控制器中的 ActionServlet 类是 Struts 框架中的核心组件,它在 MVC 模型中扮演中央控制器的角色,主要负责接收 HTTP 请求信息,根据配置文件 struts - config. xml 的配置信息,把请求转发给适当的 Action 对象。

(3) Business Logic(业务逻辑)。业务逻辑负责调用模型的方法,更新模型的状态,并帮助控制应用程序的流程。

(4) Model(模型)。模型表示应用程序的状态。

(5) View(视图)。视图就是一组 JSP 文件。在这些 JSP 文件中没有业务逻辑,也没有模型信息。只有

标签,这些标签可以是标准的 JSP 标签或客户化标签,如 Struts 标签库中的标签。

对于采用 Struts 框架的 Web 应用,其工作流程可描述为:在 Web 应用启动时就会加载并初始化 ActionServlet,ActionServlet 从 struts - config. xml 文件中读取配置信息,把它们存放在各种配置对象中。当 ActionServlet 接收到一个客户请求时,将执行如下流程:

(1) ActionServlet 根据 struts - config. xml 中的配置信息,检索与客户请求匹配的 ActionMapping 实例。

(2) 创建一个 ActionForm 对象,把客户提交的表单数据保存到 ActionForm 对象中,然后根据配置信息要求调用 ActionForm 的 validate() 方法来进行表单验证。

(3) ActionServlet 根据 ActionMapping 实例包含的信息映射到响应的 Action 类。然后调用 Action 类的 execute() 方法。

(4) Action 的 execute() 方法返回一个 ActionForward 对象。ActionServlet 再把客户请求转发给 ActionForward 对象指向的 JSP 页面。

3 ActionForm 类和动态 ActionForm 类

ActionForm 是 Struts 中数据传递的中介者,它既负责从页面中获取数据后提交给 Action 类处理,也负责将 Action 类的处理结果传递给页面。这时,Action 对象可以直接对它进行读写,而不再需要和 request、response 对象进行数据交互。

ActionForm 类是一个抽象类,必须为每个输入表单创建该类的子类,为表单中出现的每个字段定义一个属性。开发者可以从它派生子类对象,利用它和 Struts 提供的自定义标签库相结合,就可以实现对客户端表单数据的良好封装和支持。使用它的目的是为了存储用户在相关表单中输入的数据,在数据校验通过后,由控制器传送给相应的 Action 方法进行业务逻辑操作。同时将 Action 类的处理结果传递给页面。

由于 ActionForm 对应于页面表单,所以随着页面的增多,ActionForm 将会急聚增加。如果表单的字段发生变化,就必须修改并编译相关的 ActionForm 类。动态 ActionForm (DynaActionForm) 即为减少 ActionF-

orm 的数目而设计的,利用它就不必创建一个具体的 ActionForm 类,只需要在配置文件中配置出所需的虚拟 ActionForm,而由 Struts 框架通过配置文件动态创建这个 ActionForm。

DynaActionForm 类是 ActionForm 类的子类。配置动态 ActionForm 的方法是:在 struts - config. xml 文件中配置一个 < form - bean > 元素,将 type 属性设置为 DynaActionForm 或它的某个子类的全名,< form - bean > 的 < form - property > 子元素用来设置动态 ActionForm 的属性,< form - property > 元素的 name 属性指定属性名即表单字段,type 属性指定 ActionForm 的属性的类型,如果属性为 Java 基本类型,应该把属性设置为相应的 Java 包装类型。

Action 类和 JSP 都可以访问动态 ActionForm。在普通 ActionForm 中针对每个属性都提供了 get/set 方法,来读取或设置属性。而 DynaActionForm 把所有属性保存在一个 Map 对象中,并提供了下面访问所有属性的通用方法:

```
public Object get( String name )
public void set( String name, Object value )
```

其中,get(String name) 方法根据指定的属性名返回属性值;set(String name, Object value) 方法用于为给定的属性赋值。例如,若要访问 DynaActionForm 类中的 Address 属性,可以采用如下代码:

```
form. set( " Address" , " 广州" ) //set Address
String Address = ( String ) form. get ( " Ad-
dress" ) //get Address
```

4 动态 ActionForm 应用实例

本实例用来显示如何利用动态 ActionForm (DynaActionForm) 来减少 ActionForm 的数目,而不必创建一个具体的 ActionForm 类,却能负责从页面中获取数据后提交给 Action 类处理,也能负责将 Action 类的处理结果传递给页面。

该程序主要用到的文件有:名为 hello. jsp 和 helloWorld. jsp 的两个页面文件;helloAction. java 和 helloWorldAction. java 两个 Action 类;以及 struts - config. xml 配置文件。下面是相关文件的核心代码。

4.1 hello.jsp 文件的核心代码

```
<% @ page contentType = "text/html; charset =
GBK" % >
<% @ taglib uri = "/WEB-INF/struts-html.tld"
prefix = "html" % >
<% @ taglib uri = "/WEB-INF/struts-bean.tld"
prefix = "bean" % >
<% @ taglib uri = "/WEB-INF/struts-logic.tld"
prefix = "logic" % >
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" >
<html:html locale = "true" >
<BODY leftMargin = 0 topMargin = 0 MS_POSITIONING = "GridLayout" >
<html:form action = "/hello.do" >
<TABLE cellSpacing = 1 cellPadding = 1 width = "100%" align = center border = 0 >
<TBODY >
<TD align = right width = 65 > 你好!
</TD >
<TD > <html:text property = "userNameHello" size = "15" /> </TD >
<TD align = right width = 80 > 你现在的位置在 </TD >
<TD width = 160 > <html:text property = "addressHello" /> </TD >
</TBODY >
</TABLE >
<html:submit />
</html:form >
</BODY >
</html:html >
```

该文件中 `<html:text>` 标签是 `<html:form>` 的子标签,用于创建表单的文本框, `userNameHello`、`addressHello` 属性和 `ActionForm Bean` 中的属性相关联。

4.2 helloWorld.jsp 文件的核心代码

```
<% @ page contentType = "text/html; charset =
GBK" % >
<% @ taglib uri = "/WEB-INF/struts-html.tld"
```

```
prefix = "html" % >
<% @ taglib uri = "/WEB-INF/struts-bean.tld"
prefix = "bean" % >
<% @ taglib uri = "/WEB-INF/struts-logic.tld"
prefix = "logic" % >
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" >
<html:html locale = "true" >
<BODY leftMargin = 0 topMargin = 0 MS_POSITIONING = "GridLayout" >
<html:form action = "/helloWorld.do" >
<TABLE cellSpacing = 1 cellPadding = 1 width = "100%" align = center border = 0 >
<TBODY >
<TD align = right width = 65 > 大家好!
</TD >
<TD > <html:text property = "userNameHelloWorld" size = "15" /> </TD >
<TD align = right width = 80 > 大家去
</TD >
<TD width = 160 > <html:text property = "addressHelloWorld" /> </TD >
</TBODY >
</TABLE >
<html:submit />
</html:form >
</BODY >
</html:html >
```

该文件中的 `userNameHelloWorld`、`addressHelloWorld` 属性和 `ActionForm Bean` 的属性相关联。

4.3 helloAction.java 文件的核心代码(略)

当该类被调用时, `Struts` 框架会把已包含表单数据的动态 `ActionForm` 传给它。该文件中我们利用 `get` 方法得到 `hello.jsp` 中 `userNameHello`、`addressHello` 属性表单的数据。

4.4 helloWorldAction.java 文件的核心代码(略)

当该类被调用时, `Struts` 框架会把数据的传给动态 `ActionForm`,然后传给 `helloWorld.jsp` 中的 `userNameHelloWorld`、`addressHelloWorld` 属性显示出来。该文件

中我们利用 set 方法给 helloWorld.jsp 中的 userNameHelloWorld、addressHelloWorld 属性进行赋值。

4.5 struts-config.xml 配置文件

```
<? xml version = " 1. 0" encoding = " UTF - 8" ?
>
<! DOCTYPE struts - config PUBLIC " -//Apache
Software Foundation//DTD Struts Configuration 1. 2//
EN" " http://struts. apache. org/dtds/struts - config_1_
2. dtd" >
<struts - config >
  <data - sources / >
  <form - beans >
    <form - bean name = " helloForm" type = "
org. apache. struts. action. DynaActionForm" >
      <form - property name = " userNameHello"
type = " java. lang. String" / >
      <form - property name = " addressHello "
type = " java. lang. String" / >
    </form - bean >
    <form - bean name = " helloWorldForm"
type = " org. apache. struts. action. DynaActionForm" >
      <form - property name = " userNameHelloWorld"
type = " java. lang. String" / >
      <form - property name = " addressHelloWorld"
type = " java. lang. String" / >
    </form - bean >
  </form - beans >
  <global - exceptions / >
  <global - forwards / >
  <action - mappings >
    <action
      path = " /hello" name = " helloForm" scope
= " request" type = " helloAction " >
      <forward name = " sayHello" path = " /
hello. jsp" / >
```

```
</action >
<action
  path = " /helloWorld" name = " helloWorld-
Form" scope = " request"
  type = " helloWorldAction" >
  <forward name = " sayHelloWorld" path
= " /helloWorld. jsp" / >V </action >
</action - mappings >
</struts - config >
```

该配置文件中配置了两个 <form-bean> 元素来对应两个 hell.jsp 和 helloWorld.jsp 页面,并将 type 属性设置为 DynaActionForm。<form-property> 元素的 name 属性指定为与相应表单中的字段相对应的属性名字,type 设置为相应的 Java 包装类型。

5 结束语

Struts 提供了一个非常好的 MVC 框架,本文较详细介绍了动态 ActionForm 在 Struts 框架中的应用,并以一个具体的实例说明了如何利用动态 ActionForm (DynaActionForm) 来减少 ActionForm 的数目,而不必创建一个具体的 ActionForm 类,由 Struts 框架通过配置文件来动态创建 ActionForm。这样对于开发大型的 Struts 应用,利用动态 ActionForm 更能简化编程,易于程序的维护扩展。

参考文献

- 1 孙卫琴,精通 Struts:基于 MVC 的 Java Web 设计与开发[M],北京:电子工业出版社,2004.
- 2 张南华、陈阔中、卫志华,基于 Struts 框架构建网上办公系统[J],微型电脑应用,2003,19(9):27~29.
- 3 马黎明、熊前兴,基于 Struts 架构的 Web 应用系统开发研究[J],交通与计算机,2004,22(1):96~98.
- 4 郭梅、江红,Struts 在实现 MVC 架构中的应用[J],计算机与现代化,2004,(1):106~108.