Struts2 零配置

一、简介

Struts2 零配置的使用方式体现了“约定优于配置”的开发思想，通过遵循一定的约定规则或使用一些专门的注解，省略 XML 文档中的配置信息达到简化开发的目的。

二、HelloWorld

我们首先通过一个简单的例子感受一下 Struts2 零配置的工作方式。

1.要求：不使用 struts.xml 配置文件访问 Action 类，并转向到一个结果页面。

2.做法：

①加入 JAR 包：struts2-convention-plugin-2.3.15.3.jar

②在 index.jsp 页面中添加超链接

```
<a href="${pageContext.request.contextPath }/hello-world">HelloWorld</a>
```

③创建/WEB-INF/content/hello-world.jsp

```
<h1>HelloWorld</h1>
```

3.分析

以上操作相当于如下配置

```
<action name ="hello-world"
        class= "com.opensymphony.xwork2.ActionSupport"
        method= "execute">
     <result name= "success">/WEB-INF/content/hello-world.jsp </result >
</action >
```

在 Struts2 零配置工作模式下，结果资源默认保存在/WEB-INF/content 目录下，文件名和 URL 地址一致。

三、创建 Action 类

①包的要求

Convention 插件会在名为 struts、struts2、action 或 actions 的包下查找 Action 类。参照如下配置

```
<constant name="struts.convention.package.locators"
value= "action,actions,struts,struts2"/>
```

说明：以上配置可以在如下文件中找到

```
struts2-convention-plugin-2.3.15.3.jar/struts-plugin.xml
```

例如：

```
com.atguigu.struts
com.atguigu.user.struts2
com.atguigu.book.action.show
com.atguigu.order.actions
```

②Action 类的要求

实现 com.opensymphony.xwork2.Action 接口或类名以 Action 结尾。参照如下

配置

```
<constant name= "struts.convention.action.suffix"
value="Action"/>
```

例如：

```
com.atguigu.actions.MainAction
com.atguigu.actions.products.Display (implements com.opensymphony.xwork2.Action)
com.atguigu.struts.company.details.ShowCompanyDetailsAction
```

四、访问 Action 类

①名称空间：完整包名中关键包到类名之间的部分。上例中的 Action 类对应的名称空间是

```
com.atguigu.actions.MainAction -> /
com.atguigu.actions.products.Display -> /products
com.atguigu.struts.company.details.ShowCompanyDetailsAction -> /company/details
```

②URL 地址：类名去掉 Action 后缀，将剩下的部分按照驼峰式命名规则拆分成独立的单词，并全部转为小写，再以“-”连接。所以访问上例中的 Action 类的完整 URL 地址是

```
com.atguigu.actions.MainAction -> /main
com.atguigu.actions.products.Display -> /products/display
com.atguigu.struts.company.details.ShowCompanyDetailsAction ->
/company/details/show-company-details
```

参考以下配置

```
<constant name= "struts.convention.action.name.separator" value="-"/>
```

五、Result

①JSP 文件默认保存的位置/WEB-INF/content 目录，参考以下配置

```
<constant name= "struts.convention.result.path"
 value="/WEB-INF/content/"/>
```

②Action 方法返回的字符串是 success，则 JSP 文件的命名规则是：Action 简单类名去掉 Action 后缀，将剩下的部分按照驼峰式命名规则拆分成独立的单词，并全部转为小写，再以“-”连接。则上例中的 Action 类对应的结果 JSP 是

```
com.atguigu.actions.MainAction ->/WEB-INF/content/main.jsp
com.atguigu.actions.products.Display ->/WEB-INF/content/display.jsp
com.atguigu.struts.company.details.ShowCompanyDetailsAction->/WEB-INF/content
/show-company-details.jsp
```

③其实结果资源并非必须是 JSP，URL 地址和结果资源的对应关系参见下表

URL	Result	File that could match	Result Type

/hello	success	/WEB-INF/content/hello.jsp	Dispatcher
/hello	success	/WEB-INF/content/hello-success.htm	Dispatcher
/hello	success	/WEB-INF/content/hello.ftl	FreeMarker
/hello-world	input	/WEB-INF/content/hello-world-input.vm	Velocity
/test1/test2/hello	error	/WEB-INF/content/test/test2/hello-error.html	Dispatcher

④如果 Action 方法返回的不是 success，而是其他值，则目标资源的命名规则是：xxx-resultCode.扩展名，例如：

Action 类名	返回值	结果资源名
MainAction	save	main-save.jsp
ShowCompanyDetailsAction	zero	show-company-details-zero .jsp

六、常用注解

1.@Action 注解

①标记在 Action 方法上，指定访问当前 Action 方法的 URL 地址

```
package com.example.actions;

import com.opensymphony.xwork2.ActionSupport;
import org.apache.struts2.convention.annotation.Action;

public class HelloWorld extends ActionSupport {
  @Action("/different/url")
  public String execute() {
    return SUCCESS;
  }
}
```

②与@Actions 注解配合，为当前 Action 方法匹配多个 URL 地址

```
public class HelloWorld extends ActionSupport {
    @Actions({
        @Action("/different/url"),
        @Action("/another/url")
    })
    public String execute() {
        return SUCCESS;
    }
}
```

③标记在 Action 方法上，定义 execute()以外的 Action 方法

```
public class HelloWorld extends ActionSupport {
  @Action("/different/url")
  public String execute() {
    return SUCCESS;
  }

  @Action("url")
  public String doSomething() {
    return SUCCESS;
  }
}
```

④指定当前 Action 方法引用的拦截器或拦截器栈

```
@Action(interceptorRefs={@InterceptorRef("validation"),
@InterceptorRef("defaultStack")})
    public String execute() {
        return SUCCESS;
    }
@Action(interceptorRefs=@InterceptorRef(value="validation",params={"programma
tic", "false", "declarative", "true"}))
    public String execute() {
        return SUCCESS;
    }
```

2.Result 注解

①为 Action 类声明类级别的 result

```
@Results({
  @Result(name="failure", location="fail.jsp")
})
public class HelloWorld extends ActionSupport {
  @Action(value="/different/url",
    results={@Result(name="success",          location="http://struts.apache.org",
type="redirect")}
  )
  public String execute() {
    return SUCCESS;
  }

  @Action("/another/url")
```

```
  public String doSomething() {
    return SUCCESS;
  }
}
```

②指定结果类型并设置参数

```
    @Action(value="/different/url",
        results={@Result(name="success", type="httpheader", params={"status", "500",
"errorMessage", "Internal Error"})}
    )
    public String execute() {
        return SUCCESS;
    }
```