1.2相关知识点

1.2.1 Struts2 访问 Servlet 的 API

前面已经对Struts2的流程已经执行完成了，但是如果表单中有参数如何进行接收又或者我们需 要向页面保存一些数据，又要如何完成呢？我们可以通过学习Struts2访问Servlet的API来实现这 样的功能。

在Struts2中，Action并没有直接和Servlet API进行耦合，也就是说在Struts2的Action中不能 直接访问Servlet APL虽然Struts2中的Action访问Servlet API麻烦一些，但是这却是Struts2中Action 的重要改良之一，方便Action进行单元测试。

尽管Action和Servlet API解耦会带来很多好处，然而在Action中完全不访问Servlet API几乎 是不可能的，在实现业务逻辑时，经常要访问Servlet中的对象，如session> request和application等。 在Struts2中，访问Servlet API有3种方法，具体如下：

1.2.1.1通过 ActionContext 类访问

Struts2 框架提供了 ActionContext 类来访问 Servlet APL ActionContext 是 Action 执行的上下文对象，在ActionContext中保存了 Action执行所需要的所有对象，包括parameters, request, session, application等。下面列举ActionContext类访问Servlet API的几个常用方法，具体如表所示。

方法声明	功能描述
void put(String key, Object value)	将 key-value 键值对放入 ActionContext 中， 模拟 Servlet API 中的 HttpServletRequest 的 setAttnbute () 方法。
Object get(Strmg key)	通过参数key来查找当前ActionContext中的 值。
Map< Str mg, Obj ec t> getApplicationQ	返回一•个Application级的Map对象。
static ActionContext getContext()	获取当前线程的ActionContext对象。
Map< Str mg, Obj ec t> getParametersQ	返回一个包含所有HttpServletRequest参数信息的对象
Map<String,Object> getSession()	返回一个Map类型的HttpSession对象。
void setApplication(Map<String,Object〉 application)	设置 Application上下文。
void setSession(Map<Strmg,Object〉 session)	设置一个Map类型的Session值。

列举的是ActionContext类访问Servlet API的常用方法，要访问Servlet API,可以通过如下方式 进行，具体示例代码如下：

```
ActionContext context = ActionContext.getContext();
context.put("name", "admiral");
context. getApplication () . put ("name", "admiral");
context.getSession().put("name","admiral");
```

在上述示例代码中，通过ActionContext类中的方法调用，分别在request、application和session中放入了("name", "admiral")对。可以看到，通过ActionContext类可以非常简单地访问JSP内置对象的属性。

为了让大家更好地掌握如何通过ActionContext类来访问Servlet API,接下来通过一个具体的案 例来演示ActionContext的使用。

(1) 在Eclipse中创建一个名称为struts2_day02的Web项目，将Struts2所需的jar包复制到项 目的lib目录中，并发布到类路径下。在WebContent目录下编写一个简单的登录页面demo1.jsp,如 文件所示。

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <form action="${ pageContext.request.contextPath }/requestDemo1.action"
method="post">
        账号:<input type="text" name="username"><br/>
        密码:<input type="text" name="password"><br/>
        <input type="submit" value="提交">
    </form>
</body>
</html>
```

(2)在WEB-INF目录下创建一个名称为web.xml的文件，在该文件中配置Struts2的核心控制 器，如文件所示。

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" id="WebApp_ID" version="2.5">
  <display-name>Struts2_day02</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>

    <filter>
    <filter-name>struts</filter-name>
```

```
    <filter-
class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter</fil
ter-class>
  </filter>
  <filter-mapping>
    <filter-name>struts</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>

</web-app>
```

(3)在src目录下创建struts.xml文件，如下所示

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo1" extends="struts-default" namespace="/">
        <action name="requestDemo1" class="com.admiral.web.RequestDemo1">
            <result name="success">/demo1/demo2.jsp</result>
        </action>
    </package>

</struts>
```

(4)在 src 目录下创建一个 com.admiral.web 包，再在 com.admiral.web 包中创建 RequestActionDemo 1类，进行业务逻辑处理，如下所示

```
/**
* @Title: RequestDemo1.java
* @Package com.admiral.web
* @Description:
* @author 白世鑫
* @date 2020-9-27
* @version V1.0
*/
package com.admiral.web;

import java.util.Arrays;
import java.util.Map;

import com.opensymphony.xwork2.ActionContext;
import com.opensymphony.xwork2.ActionSupport;

public class RequestDemo1 extends ActionSupport{

    public String execute() {
        // 一. 接收参数
        // 利用 Struts2 中的 ActionContext 对象
```

```
        ActionContext actionContext = ActionContext.getContext();
        // 调用 ActionContext 中的方法
        Map<String, Object> map = actionContext.getParameters();

        for (String key : map.keySet()) {
            String[] values = (String[]) map.get(key);
            System.out.println(key + "  " + Arrays.toString(values));
        }

        // 二. 向域对象中存入数据
        actionContext.put("reqName","reqValue");//相当于调用 request.setAttribute;
        actionContext.getSession().put("sessName", "sessValue");// 相当于调用
session.setAttribute;
        actionContext.getApplication().put("appName", "appValue");


        return SUCCESS;
    }
}
```

(5)在WebContent目录下创建demo2.jsp,如下所示

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    ${reqName }
    ${sessName }
    ${appName }
</body>
</html>
```

1.2.1.2 使用Servlet的API的原生方式

编写JSP

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
```

```
    <h1>Servlet API 的访问</h1>
    <h3>完全解耦合的方式</h3>
    <form action="${ pageContext.request.contextPath }/requestDemo1.action"
method="post">
        账号:<input type="text" name="username"><br/>
        密码:<input type="text" name="password"><br/>
        <input type="submit" value="提交">
    </form><br/>
    <h3>使用原生Servlet API</h3>
    <form action="${ pageContext.request.contextPath }/requestDemo2.action"
method="post">
        账号:<input type="text" name="username"><br/>
        密码:<input type="text" name="password"><br/>
        <input type="submit" value="提交">
    </form><br/>
</body>
</html>
```

编写Action

```
/**
* @Title: RequestDemo2.java
* @Package com.admiral.web
* @Description:
* @author 白世鑫
* @date 2020-9-27
* @version V1.0
*/
package com.admiral.web;

import javax.servlet.http.HttpServletRequest;

import org.apache.struts2.ServletActionContext;

import com.opensymphony.xwork2.ActionSupport;

public class RequestDemo2 extends ActionSupport {

    @Override
    public String execute() throws Exception {
        //一. 接收参数
        //获取 request 通过 ServletActionContext
        HttpServletRequest request = ServletActionContext.getRequest();
        System.out.println(request.getParameter("username"));
        System.out.println(request.getParameter("password"));


        //向域对象中保存数据
        request.setAttribute("reqName", "reqValue2");
        request.getSession().setAttribute("sessName", "sessValue2");
        ServletActionContext.getServletContext().setAttribute("appName",
"appValue2");


        return SUCCESS;
```

```
    }
}
```

编写 配置文件

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo1" extends="struts-default" namespace="/">
        <action name="requestDemo1" class="com.admiral.web.RequestDemo1">
            <result name="success">/demo1/demo2.jsp</result>
        </action>
        <action name="requestDemo2" class="com.admiral.web.RequestDemo2">
            <result name="success">/demo1/demo2.jsp</result>
        </action>
    </package>

</struts>
```

1.2.1.3 接口注入的方式

编写jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h1>Servlet API 的访问</h1>
    <h3>完全解耦合的方式</h3>
    <form action="${ pageContext.request.contextPath }/requestDemo1.action"
method="post">
        账号:<input type="text" name="username"><br/>
        密码:<input type="text" name="password"><br/>
        <input type="submit" value="提交">
    </form><br/>
    <h3>使用原生Servlet API</h3>
    <form action="${ pageContext.request.contextPath }/requestDemo2.action"
method="post">
```

```
        账号:<input type="text" name="username"><br/>
        密码:<input type="text" name="password"><br/>
        <input type="submit" value="提交">
    </form><br/>
    <h3>接口注入的方式</h3>
    <form action="${ pageContext.request.contextPath }/requestDemo3.action"
method="post">
        账号:<input type="text" name="username"><br/>
        密码:<input type="text" name="password"><br/>
        <input type="submit" value="提交">
    </form><br/>
</body>
</html>
```

编写Action

```
/**
* @Title: RequestDemo3.java
* @Package com.admiral.web
* @Description:
* @author 白世鑫
* @date 2020-9-28
* @version V1.0
*/
package com.admiral.web;

import javax.servlet.ServletContext;
import javax.servlet.http.HttpServletRequest;

import org.apache.struts2.interceptor.ServletRequestAware;
import org.apache.struts2.util.ServletContextAware;

import com.opensymphony.xwork2.ActionSupport;

public class RequestDemo3 extends ActionSupport implements
ServletRequestAware,ServletContextAware{

    private HttpServletRequest request;
    private ServletContext context;

    @Override
    public String execute() throws Exception {
        // 获取参数
        System.out.println(request.getParameter("username"));
        System.out.println(request.getParameter("password"));

        // 保存数据
        request.setAttribute("reqName", "reqValue3");
        request.getSession().setAttribute("sessName", "sessValue");
        context.setAttribute("appName", "appValue3");
        return super.execute();
    }

    @Override
    public void setServletRequest(HttpServletRequest request) {
        this.request = request;
    }
```

```
    @Override
    public void setServletContext(ServletContext context) {
        this.context = context;
    }
}
```

编写配置文件

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo1" extends="struts-default" namespace="/">
        <action name="requestDemo1" class="com.admiral.web.RequestDemo1">
            <result name="success">/demo1/demo2.jsp</result>
        </action>
        <action name="requestDemo2" class="com.admiral.web.RequestDemo2">
            <result name="success">/demo1/demo2.jsp</result>
        </action>
        <action name="requestDemo3" class="com.admiral.web.RequestDemo3">
            <result name="success">/demo1/demo2.jsp</result>
        </action>
    </package>

</struts>
```

1.2.2结果页面的配置

1.2.2.1全局结果页面

- 全局结果页面：全局结果页面指的是，在包中配置一次，其他的在这个包中的所有的action只要返回了这个值，都可以跳转到这个页面。
    - 针对这个包下的所有的action的配置都有效。

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo1" extends="struts-default" namespace="/">

        <global-results>
            <result name="success">/demo1/demo2.jsp</result>
        </global-results>

        <action name="requestDemo1" class="com.admiral.web.RequestDemo1">
        </action>
        <action name="requestDemo2" class="com.admiral.web.RequestDemo2">
        </action>
        <action name="requestDemo3" class="com.admiral.web.RequestDemo3">
        </action>
    </package>

</struts>
```


```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo1" extends="struts-default" namespace="/">

        <global-results>
            <result name="success">/demo1/demo2.jsp</result>
        </global-results>

        <action name="requestDemo1" class="com.admiral.web.RequestDemo1">
        </action>
        <action name="requestDemo2" class="com.admiral.web.RequestDemo2">
        </action>
        <action name="requestDemo3" class="com.admiral.web.RequestDemo3">
        </action>
    </package>

</struts>
```


1.2.2.2局部结果页面

- 局部结果页面：局部结果页面指的是，只能在当前的action中的配置有效。
    - 针对当前的action有效。

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo1" extends="struts-default" namespace="/">

        <global-results>
            <result name="success">/demo1/demo2.jsp</result>
        </global-results>

        <action name="requestDemo1" class="com.admiral.web.RequestDemo1">
            <result name="success">/demo1/demo3.jsp</result>
        </action>
        <action name="requestDemo2" class="com.admiral.web.RequestDemo2">
        </action>
        <action name="requestDemo3" class="com.admiral.web.RequestDemo3">
        </action>
    </package>

</struts>
```


```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo1" extends="struts-default" namespace="/">

        <global-results>
            <result name="success">/demo1/demo2.jsp</result>
        </global-results>

        <action name="requestDemo1" class="com.admiral.web.RequestDemo1">
            <result name="success">/demo1/demo3.jsp</result>
        </action>
        <action name="requestDemo2" class="com.admiral.web.RequestDemo2">
        </action>
        <action name="requestDemo3" class="com.admiral.web.RequestDemo3">
        </action>
    </package>

</struts>
```


1.2.2.3 result标签的配置

- result标签用于配置页面的跳转。在result标签上有两个属性：
  - name属性 ：逻辑视图的名称。默认值：success
  - type属性 ：页面跳转的类型。
    - dispatcher ：默认值，请求转发。（Action转发JSP）
    - redirect ：重定向。（Action重定向JSP）
    - chain ：转发。（Action转发Action）
    - redirectAction ：重定向。（Action重定向Action）
    - stream ：Struts2中提供文件下载的功能。

1.2.3 Struts的数据封装

1.2.3.1属性驱动:提供属性set方法的方式（不常用）

准备JavaBean

```
/**
* @Title: User.java
* @Package com.admiral.web
* @Description:
* @author 白世鑫
* @date 2020-9-28
* @version V1.0
*/
package com.admiral.web2;

public class User {
    private String username;
    private String password;
    private Integer age;
    private Double salary;
    public User() {
        super();
        // TODO Auto-generated constructor stub
    }
    public User(String username, String password, Integer age, Double salary) {
        super();
        this.username = username;
        this.password = password;
        this.age = age;
        this.salary = salary;
    }
    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public Integer getAge() {
        return age;
    }
    public void setAge(Integer age) {
        this.age = age;
    }
    public Double getSalary() {
        return salary;
    }
    public void setSalary(Double salary) {
        this.salary = salary;
```

```
    }
}
```

编写JSP

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h1>Struts2 数据的封装</h1>
    <h3>属性驱动:提供 set 方法</h3>
    <form action="${pageContext.request.contextPath }/userAction1.action"
method="post">
        账号:<input type="text" name="username"><br>
        密码:<input type="text" name="password"><br>
        年龄:<input type="text" name="age"><br>
        工资:<input type="text" name="salary"><br>
        <input type="submit" value="提交">
    </form><br>
</body>
</html>
```

编写 Action

```
/**
* @Title: UserAction.java
* @Package com.admiral.web
* @Description:
* @author 白世鑫
* @date 2020-9-28
* @version V1.0
*/
package com.admiral.web2;

import com.opensymphony.xwork2.ActionSupport;

public class UserAction extends ActionSupport {

    private String username;
    private String password;
    private Integer age;
    private Double salary;

    public void setUsername(String username) {
        this.username = username;
    }

    public void setPassword(String password) {
        this.password = password;
```

```
    }

    public void setAge(Integer age) {
        this.age = age;
    }

    public void setSalary(Double salary) {
        this.salary = salary;
    }

    @Override
    public String execute() throws Exception {
        // 接收数据
        System.out.println(username);
        System.out.println(password);
        System.out.println(age);
        System.out.println(salary);
        // 保存数据
        return NONE;
    }
}
```

编写配置文件

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo2" extends="struts-default" namespace="/">
        <action name="userAction1" class="com.admiral.web2.UserAction">
        </action>
    </package>

</struts>
```

引入配置文件

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <include file="com/admiral/web/struts_demo1.xml"></include>
    <include file="com/admiral/web2/struts_demo2.xml"></include>

</struts>
```

1.2.3.2 属性驱动:页面中提供表达式方式

编写 JSP

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h1>Struts2 数据的封装</h1>
    <h3>属性驱动:提供 set 方法</h3>
    <form action="${pageContext.request.contextPath }/userAction1.action"
method="post">
        账号:<input type="text" name="username"><br>
        密码:<input type="text" name="password"><br>
        年龄:<input type="text" name="age"><br>
        工资:<input type="text" name="salary"><br>
        <input type="submit" value="提交">
    </form><br>
    <h3>属性驱动:页面提供表达式的方式</h3>
    <form action="${pageContext.request.contextPath }/userAction2.action"
method="post">
        账号:<input type="text" name="user.username"><br>
        密码:<input type="text" name="user.password"><br>
        年龄:<input type="text" name="user.age"><br>
        工资:<input type="text" name="user.salary"><br>
        <input type="submit" value="提交">
    </form><br>
</body>
</html>
```

编写 Action

```
/**
* @Title: UserAction2.java
* @Package com.admiral.web2
* @Description:
* @author 白世鑫
* @date 2020-9-28
* @version V1.0
*/
package com.admiral.web2;

import com.opensymphony.xwork2.ActionSupport;

public class UserAction2 extends ActionSupport {
```

```
    // 提供一个User对象
    private User user;
    //提供 getter 和 setter 方法,注意,一定要提供 get 方法.
    public User getUser() {
        return user;
    }

    public void setUser(User user) {
        this.user = user;
    }

    @Override
    public String execute() throws Exception {
        System.out.println(user);
        return NONE;
    }
}
```

编写 配置文件

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo2" extends="struts-default" namespace="/">
        <action name="userAction1" class="com.admiral.web2.UserAction">
        </action>
        <action name="userAction2" class="com.admiral.web2.UserAction2">
        </action>
    </package>

</struts>
```

1.2.3.3 模型驱动

编写 JSP

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
```

```
<title>Insert title here</title>
</head>
<body>
    <h1>Struts2 数据的封装</h1>
    <h3>属性驱动:提供 set 方法</h3>
    <form action="${pageContext.request.contextPath }/userAction1.action"
method="post">
        账号:<input type="text" name="username"><br>
        密码:<input type="text" name="password"><br>
        年龄:<input type="text" name="age"><br>
        工资:<input type="text" name="salary"><br>
        <input type="submit" value="提交">
    </form><br>
    <h3>属性驱动:页面提供表达式的方式</h3>
    <form action="${pageContext.request.contextPath }/userAction2.action"
method="post">
        账号:<input type="text" name="user.username"><br>
        密码:<input type="text" name="user.password"><br>
        年龄:<input type="text" name="user.age"><br>
        工资:<input type="text" name="user.salary"><br>
        <input type="submit" value="提交">
    </form><br>
    <h3>模型驱动:模型驱动的方式</h3>
    <form action="${pageContext.request.contextPath }/userAction3.action"
method="post">
        账号:<input type="text" name="username"><br>
        密码:<input type="text" name="password"><br>
        年龄:<input type="text" name="age"><br>
        工资:<input type="text" name="salary"><br>
        <input type="submit" value="提交">
    </form><br>
</body>
</html>
```

编写 Action

```
/**
* @Title: UserAction3.java
* @Package com.admiral.web2
* @Description:
* @author 白世鑫
* @date 2020-9-28
* @version V1.0
*/
package com.admiral.web2;

import com.opensymphony.xwork2.ActionSupport;
import com.opensymphony.xwork2.ModelDriven;

public class UserAction3 extends ActionSupport implements ModelDriven<User>{

    private User user = new User();

    @Override
    public User getModel() {
        return user;
    }
```

```
    @Override
    public String execute() throws Exception {
        System.out.println(user);
        return NONE;
    }



}
```

编写配置文件

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo2" extends="struts-default" namespace="/">
        <action name="userAction1" class="com.admiral.web2.UserAction">
        </action>
        <action name="userAction2" class="com.admiral.web2.UserAction2">
        </action>
        <action name="userAction3" class="com.admiral.web2.UserAction3">
        </action>
    </package>

</struts>
```

- 模型驱动方式最常用的方式：
  - 缺点：只能同时向一个对象中封装数据。
- 使用第二种可以向多个对象中同时封装数据：

1.2.3.4 INPUT逻辑视图的配置

- Action接口中提供了五个逻辑视图的名称：
  - SUCCESS
  - ERROR
  - LOGIN
  - **INPUT　　：input在某些拦截器中会使用。**
  - NONE

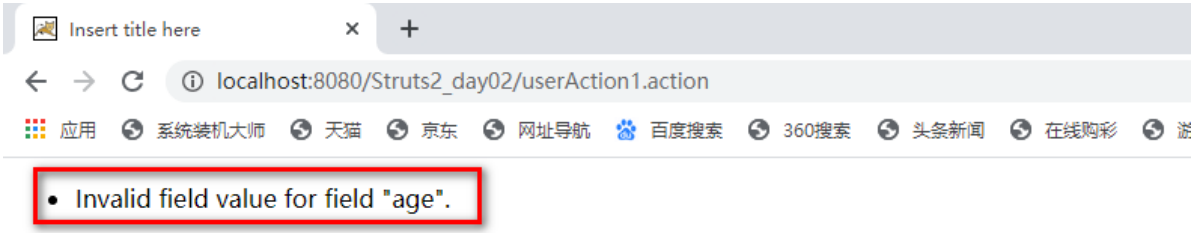
```xml
<struts>

    <package name="demo2" extends="struts-default" namespace="/">

        <global-results>
            <result name="input">/demo2/demo1.jsp</result>
        </global-results>

        <action name="userAction1" class="com.admiral.web2.UserAction">
        </action>
        <action name="userAction2" class="com.admiral.web2.UserAction2">
        </action>
        <action name="userAction3" class="com.admiral.web2.UserAction3">
        </action>
```

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="/struts-tags" prefix="s" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <s:fielderror></s:fielderror>
    <h1>Struts2 数据的封装</h1>
    <h3>属性驱动:提供 set 方法</h3>
    <form action="${pageContext.request.contextPath }/userAction1.action" method="post">
        账号:<input type="text" name="username"><br>
        密码:<input type="text" name="password"><br>
        年龄:<input type="text" name="age"><br>
        工资:<input type="text" name="salary"><br>
        <input type="submit" value="提交">
    </form><br>
    <h3>属性驱动:页面提供表达式的方式</h3>
    <form action="${pageContext.request.contextPath }/userAction2.action" method="post">
        账号:<input type="text" name="user.username"><br>
        密码:<input type="text" name="user.password"><br>
```



1.2.4 Struts2中封装集合类型的数据

1.2.4 Struts2中封装集合类型的数据

编写JavaBean

```
/**
* @Title: Product.java
* @Package com.admiral.web3
* @Description:
* @author 白世鑫
* @date 2020-9-28
* @version V1.0
*/
package com.admiral.web3;

public class Product {

    private String name;
    private Float price;

    public Product() {
        super();
    }

    public Product(String name, Float price) {
        super();
        this.name = name;
        this.price = price;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Float getPrice() {
        return price;
    }

    public void setPrice(Float price) {
        this.price = price;
    }

    @Override
    public String toString() {
        return "Product [name=" + name + ", price=" + price + "]";
    }

}
```

编写JSP

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h1>封装复杂的数据类型</h1>
    <h3>封装数据到 List 集合中</h3>
    <form action="${ pageContext.request.contextPath}/productAction1.action"
method="post">
        商品名称:<input type="text" name="products[0].name"><br/>
        商品价格:<input type="text" name="products[0].price"><br/>
        商品名称:<input type="text" name="products[1].name"><br/>
        商品价格:<input type="text" name="products[1].price"><br/>
        商品名称:<input type="text" name="products[2].name"><br/>
        商品价格:<input type="text" name="products[2].price"><br/>
        <input type="submit" value="提交">
    </form>
</body>
</html>
```

编写Action

```
/**
* @Title: ProductAction1.java
* @Package com.admiral.web3
* @Description:
* @author 白世鑫
* @date 2020-9-28
* @version V1.0
*/
package com.admiral.web3;

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

import com.opensymphony.xwork2.ActionSupport;

public class ProductAction1 extends ActionSupport {

    private List<Product> products = new ArrayList<Product>();

    public List<Product> getProducts() {
        return products;
    }

    @Override
    public String execute() throws Exception {
        for (Product product : products) {
            System.out.println(product);
        }
        return NONE;
```

```
    }
}
```

编写配置文件

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo3" extends="struts-default" namespace="/">
        <action name="productAction1" class="com.admiral.web3.ProductAction1">
        </action>
    </package>

</struts>
```

引入配置文件

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <include file="com/admiral/web/struts_demo1.xml"></include>
    <include file="com/admiral/web2/struts_demo2.xml"></include>
    <include file="com/admiral/web3/struts_demo3.xml"></include>

</struts>
```

1.2.4.2封装数据到Map集合

编写JSP页面

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
```

```
<body>
    <h1>封装复杂的数据类型</h1>
    <h3>封装数据到 List 集合中</h3>
    <form action="${ pageContext.request.contextPath}/productAction1.action"
method="post">
        商品名称:<input type="text" name="products[0].name"><br/>
        商品价格:<input type="text" name="products[0].price"><br/>
        商品名称:<input type="text" name="products[1].name"><br/>
        商品价格:<input type="text" name="products[1].price"><br/>
        商品名称:<input type="text" name="products[2].name"><br/>
        商品价格:<input type="text" name="products[2].price"><br/>
        <input type="submit" value="提交">
    </form>
    <h3>封装数据到 Map 集合中</h3>
    <form action="${ pageContext.request.contextPath}/productAction2.action"
method="post">
        商品名称:<input type="text" name="map['one'].name"><br/>
        商品价格:<input type="text" name="map['one'].price"><br/>
        商品名称:<input type="text" name="map['two'].name"><br/>
        商品价格:<input type="text" name="map['two'].price"><br/>
        商品名称:<input type="text" name="map['three'].name"><br/>
        商品价格:<input type="text" name="map['three'].price"><br/>
        <input type="submit" value="提交">
    </form>
</body>
</html>
```

编写Action

```
/**
* @Title: ProductAction2.java
* @Package com.admiral.web3
* @Description:
* @author 白世鑫
* @date 2020-9-28
* @version V1.0
*/
package com.admiral.web3;

import java.util.HashMap;
import java.util.Map;

import com.opensymphony.xwork2.ActionSupport;

public class ProductAction2 extends ActionSupport {

    private Map<String, Product> map;

    public Map<String, Product> getMap() {
        return map;
    }

    public void setMap(Map<String, Product> map) {
        this.map = map;
    }

    @Override
```

```
    public String execute() throws Exception {
        for (String key : map.keySet()) {
            Product product = map.get(key);
            System.out.println(key + "    " + product);
        }
        return NONE;
    }
}
```

编写配置文件

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

    <package name="demo3" extends="struts-default" namespace="/">
        <action name="productAction1" class="com.admiral.web3.ProductAction1">
        </action>
        <action name="productAction2" class="com.admiral.web3.ProductAction2">
        </action>
    </package>

</struts>
```

1.3案例代码

1.3.1 环境准备

1.3.2 代码实现