**Struts2**

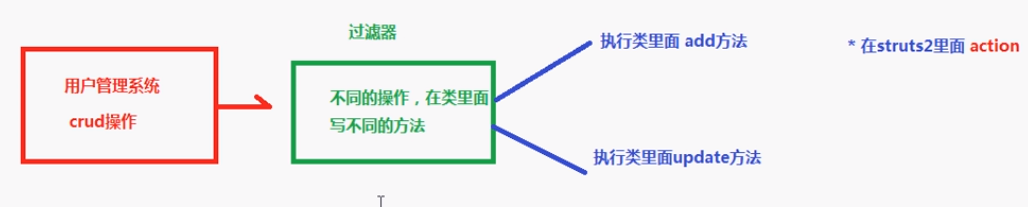
**一、Struts2入门**

**1.1 Struts2概述**

1、Struts2框架应用于javaee三层结构的web层框架

2、Struts2框架在Struts1和webwork基础之上发展全新的框架

3、Struts2解决的问题：



4、Struts2版本

5、web层常用框架：Struts2、SpringMVC

**1.2 Struts2入门案例**

1、创建步骤

第一步：导入Strut2 jar包

第二步：创建action类：访问action，每次访问时默认执行execute方法

第三步：配置action类访问路径

(1)创建struts2核心配置文件，其名称和位置固定，必须在src目录下，名称为struts.xml

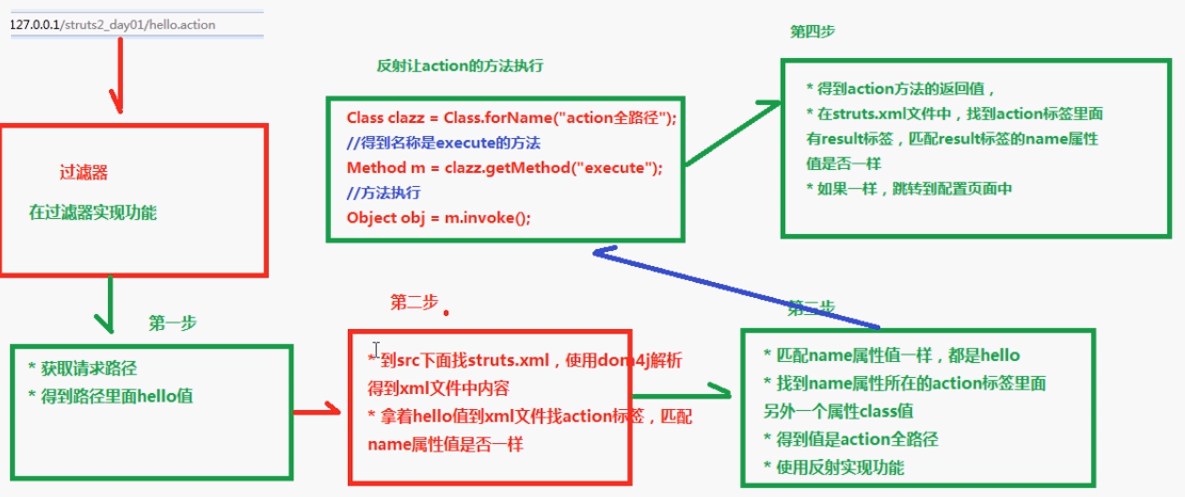
(2)引入dtd约束

(3)action配置

第四步：配置struts2过滤器

在web.xml配置

**1.3 Struts2基本执行过程**

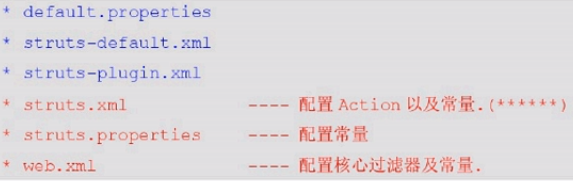


**1.4 Struts2源代码**



1、过滤器在服务器启动的时候创建，创建过滤器是执行init方法。

(1) init方法中主要进行配置文件加载,配置文件加载顺序：



**1.5 Struts2配置**

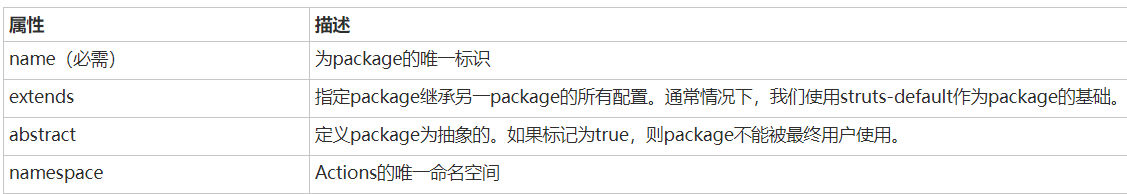
1、struts2核心配置文件，其名称和位置固定，必须在src目录下，名称为struts.xml。

2、配置文件中主要有三个标签package、action、result，标签里面有各自属性。

（1）package标签

标签声明不同的包。 这里的<package>标签允许配置的分离和模块化。

<package>标签具有以下属性：



（2）action标签

定义<action>标签对应于我们想要访问的每个URL，并且使用execute()方法定义一个访问相应的URL时将要访问的类。

包含属性: name属性、class属性、method属性

（3）result标签

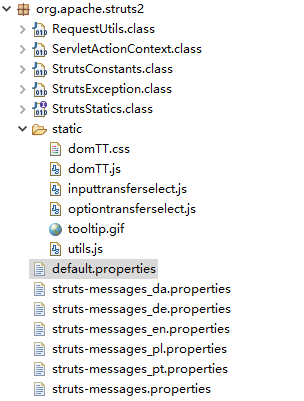
根据action的方法返回值，配置到不同的路径去。

主要属性：name（和方法返回值一直）和type（转发或者重定向，默认为转发操作）

1.6 Struts2常量配置

struts2默认常量位置

在struts2-core.jar中



1、在struts.xml中进行配置

<constant>标签以及name和value属性将用于覆盖default.properties中定义的任一属性

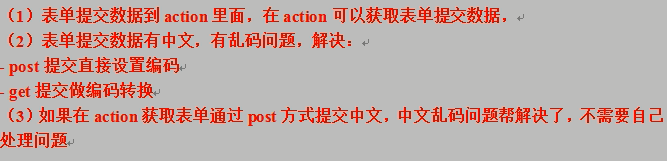
2、其他方式：

在src目录下创建struts.properties，进行修改。

在web.xml中配置

3、常用常量

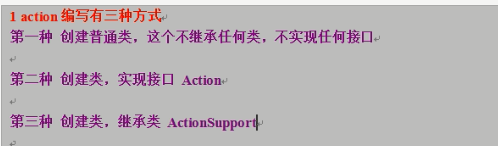
struts.i18n.encoding=UTF-8



1.7 Struts2分模块开发



1.8 Action三种编写方式



1.9 Action的方法访问介绍



action方法如果有返回值则必须为String类，如果没有返回值，则可以void修饰或者返回字符串”none”;

2.0 Action的方法访问（使用method属性）

2.3 Action的方法访问（使用通配符实现）

1、在action标签的name属性中写星号 \* ，星号表示匹配任意内容



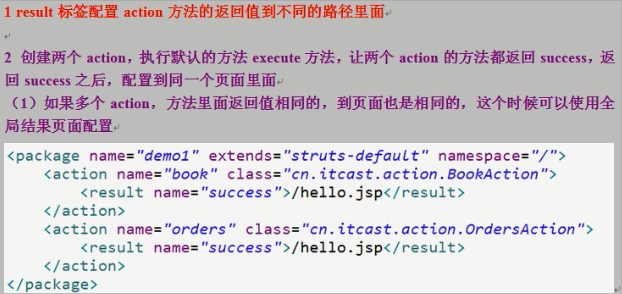
2.4 Action的方法访问（通配符分析）

2.5案例—客户列表

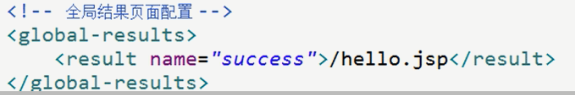
**二、Struts2数据操作**

2.1结果页面配置

（1）全局结果页面



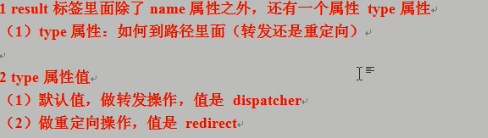
在package标签里配置

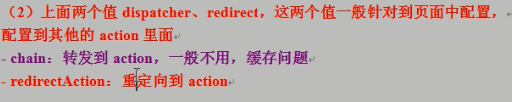


（2）局部结果页面



（3）result标签type属性

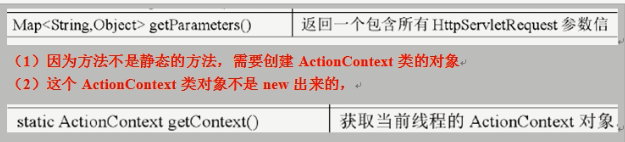


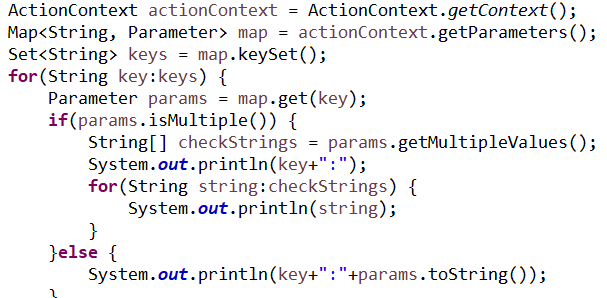


2.2在action获取表单提交数据（原始方式）

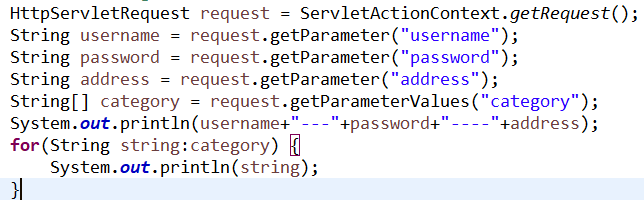


（1）使用ActionContext类获取





（2）使用ServletActionContext类获取

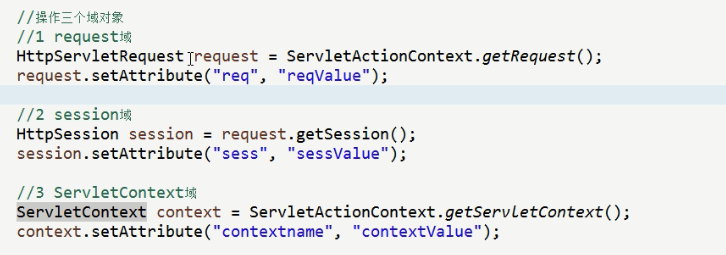


（3）使用接口注入方式获取



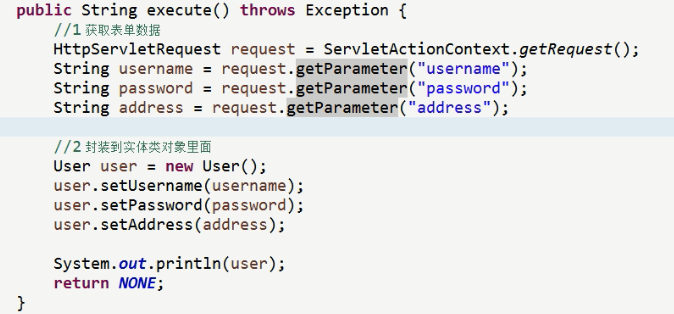
2.4在action操作域对象

1、request、session、servletContext域对象

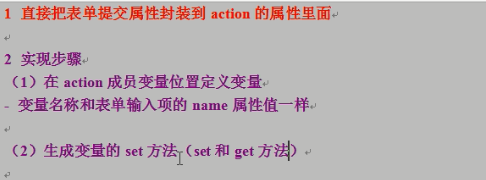


2.3 struts2提供获取表单数据方式

原始方式封装：



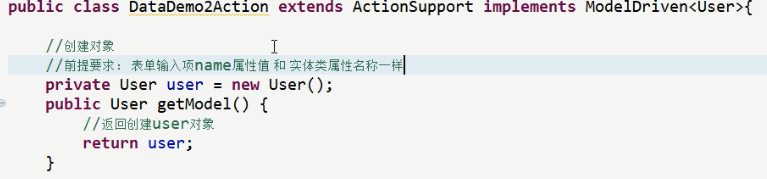
（1）属性封装



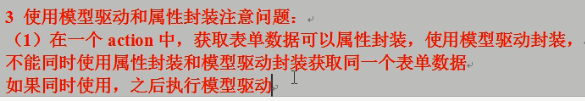


（2）模型驱动封装（重点）

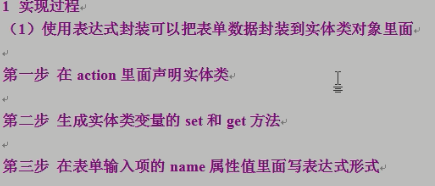


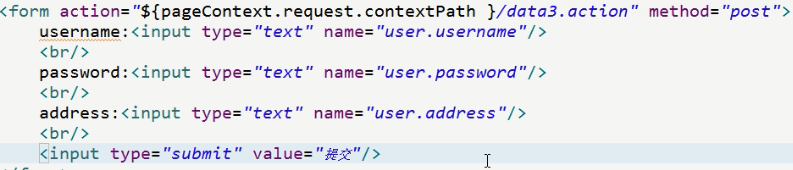


注意事项：

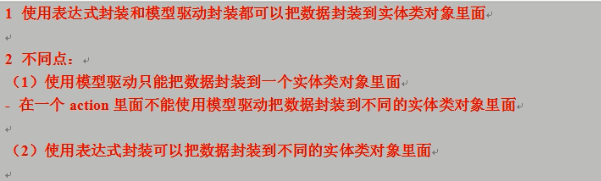


（3）表达式封装（也可归类到属性封装中）





（4）比较表达式封装和模型驱动封装



2.4 struts2获取表单数据封装到集合中

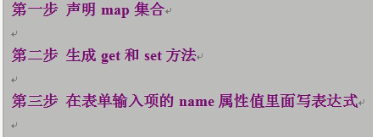
（1）封装到List集合

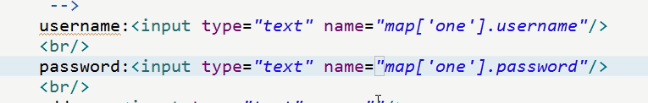






（2）封装到Map集合





2.5 案例

**三、Struts2值栈**

3.1、OGNL概述

OGNL（Object-Graph Navigation Language，对象图导航语言）是一种强大的表达式语言，用于引用和操作值栈上的数据，还可用于数据传输和类型转换。OGNL非常类似于JSP表达式语言。OGNL基于上下文中存有根对象或默认对象的理念，使用标记符号（即#号）来引用默认或根对象的属性。

如前面所述，OGNL是基于上下文的，而Struts构建了一个ActionContext映射以供OGNL使用。 ActionContext映射包含以下内容：

应用程序 - 应用程序作用域变量

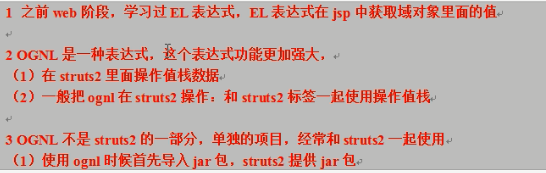
会话 - 会话作用域变量

根/值栈 - 所有的action变量都存储在这里

请求 - 请求作用域变量

参数 - 请求参数

属性 - 存储在页面，请求，会话和应用程序作用域中的属性



3.2、OGNL使用案例



使用struts2标签：



3.3、值栈

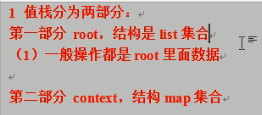
1、什么是值栈

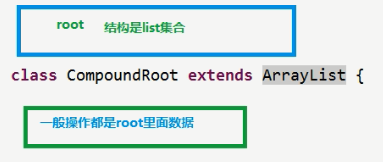


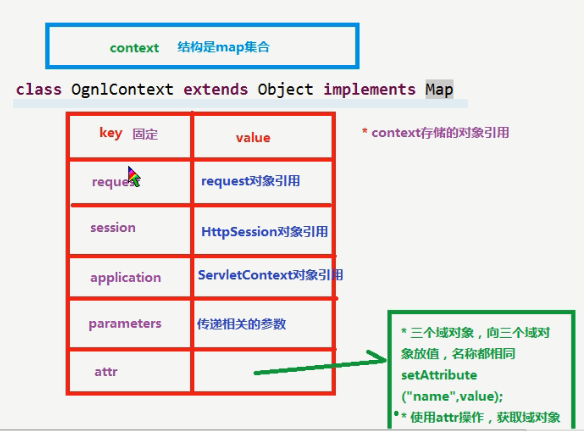
2、获取值栈对象



3、值栈的内部结构

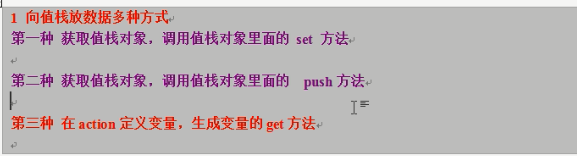




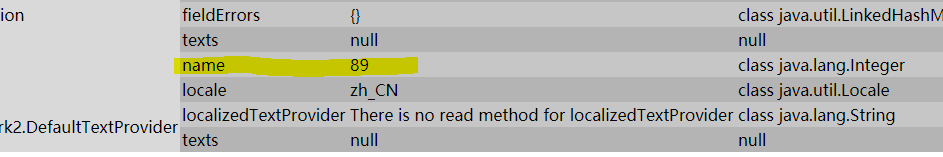


3.4 向值栈中存、取数据

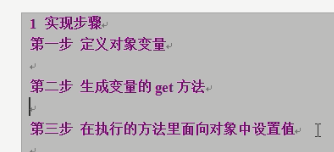
1、放数据



常用第三种（其直接将存入的数据放入action中，不会开辟新空间）



2、向值栈中放对象和list





3、获取数据（第三种方式存放的数据）



获取字符串



获取对象



获取list（三种方式）

a、



b、



c、

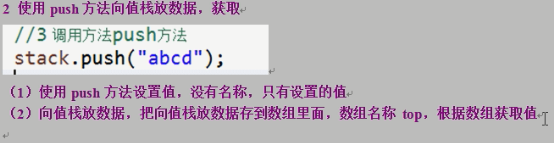


其他操作（使用set和push存放的数据）

a、使用set向值栈存放的数据，取得方式

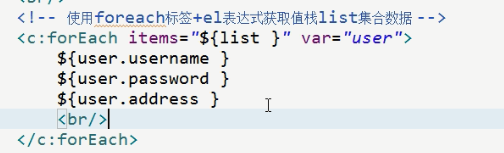


b、使用push方法向值栈存放的数据，取得方式

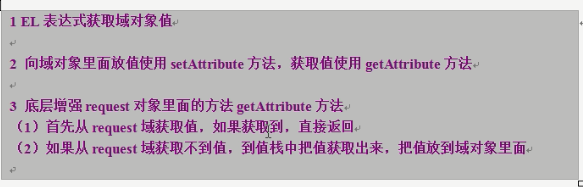




3.5 EL表达式为什么可以获取值栈的数据



为什么：

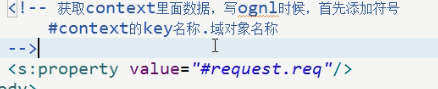


3.6 ognl表达式中：# 和 % 的使用

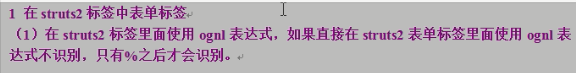
1、# 使用







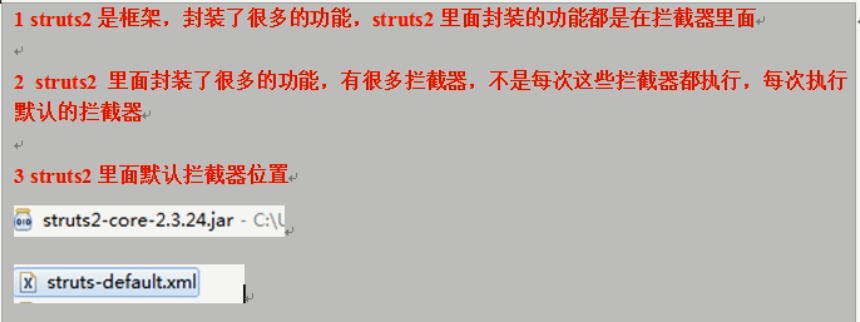
2、% 使用





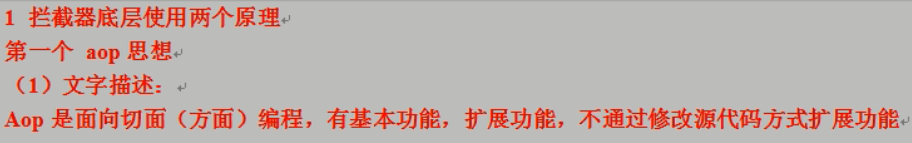
**四、Struts2拦截器**

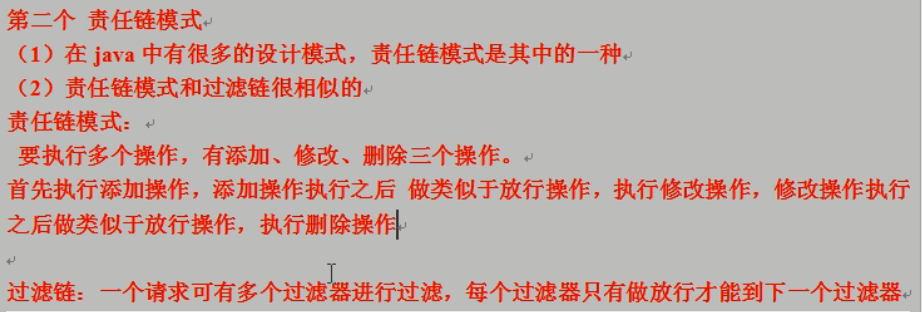
4.1 拦截器概述

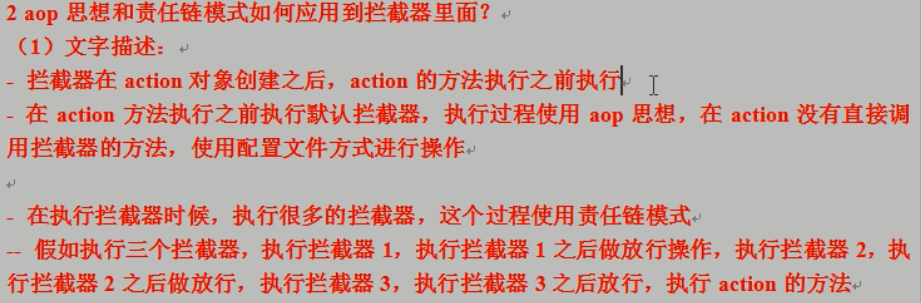




4.2 拦截器底层原理

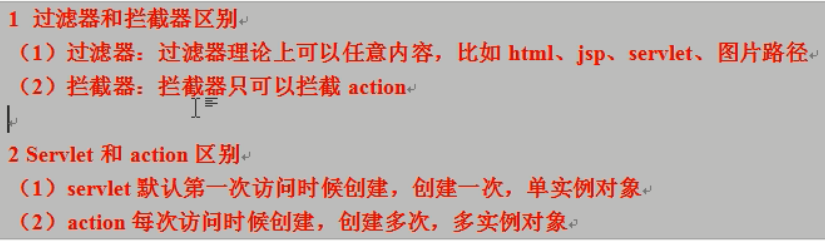






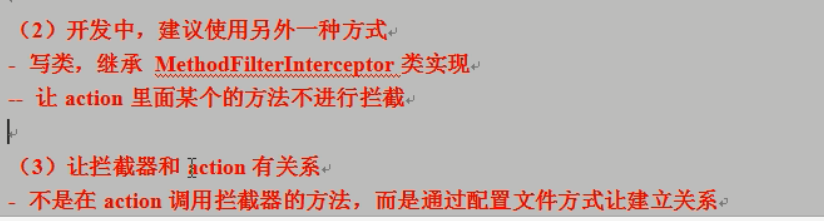


4.3 过滤器和拦截器的区别

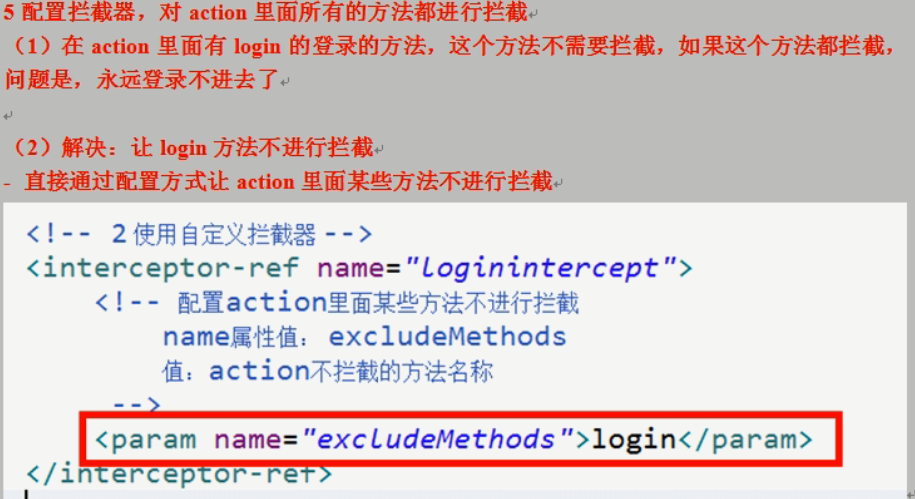


4.4 自定义拦截器





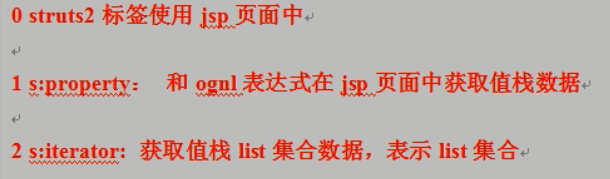




4.5 struts2标签库



常用标签



4.6 struts2表单标签



