Struts2学习笔记

目录

[Struts2学习笔记 1](#_Toc482234058)

[第一章 2](#_Toc482234059)

[第二章 5](#_Toc482234060)

[1. 结果页面配置 5](#_Toc482234061)

[2. 在action获取表达提交数据 7](#_Toc482234062)

[3. Struts2提供获取表单数据方式 8](#_Toc482234063)

[4. Struts2获取数据封装到集合中 12](#_Toc482234064)

[第三章 15](#_Toc482234065)

[1. Ognl概述 15](#_Toc482234066)

[2.什么是值栈 15](#_Toc482234067)

[3. 如何获取值栈对象 16](#_Toc482234068)

[4. 值栈内部结构 17](#_Toc482234069)

[S : debug标签 18](#_Toc482234070)

[5. 向值栈放数据 18](#_Toc482234071)

[向值栈存放对象 19](#_Toc482234072)

[向值栈放list集合 20](#_Toc482234073)

[6. 从值栈获取数据 21](#_Toc482234074)

[从值栈获取字符串 21](#_Toc482234075)

[从值栈获取对象 22](#_Toc482234076)

[从值栈中获取list集合 23](#_Toc482234077)

[7. 其他方法从值栈获取数据（了解即可） 23](#_Toc482234078)

[使用set方法向值栈存放数据，获取数据 23](#_Toc482234079)

[使用push方法向值栈存放数据，获取数据 23](#_Toc482234080)

[8. EL表达式为什么能获取数据？ 23](#_Toc482234081)

[9. OGNL的#、%使用 24](#_Toc482234082)

[#的使用 24](#_Toc482234083)

[%的使用 25](#_Toc482234084)

[第四章 25](#_Toc482234085)

[1. 拦截器概述 25](#_Toc482234086)

[1) Struts2拦截器概述 25](#_Toc482234087)

[2. 拦截器底层原理 26](#_Toc482234088)

[3. 过滤器和拦截器区别 28](#_Toc482234089)

[4. 自定义登录拦截器 28](#_Toc482234090)

[5. Struts2标签库（会用） 31](#_Toc482234091)

# 第一章

truts.xml核心配置文件的解析

位置：src/struts.xml

**package标签内的标签**

文件内有三个标签：package，action，result

**package标签**

1. 类似于代码包，区别不同的action，要配置action，必须首先写package标签里面必须先写个package
2. package标签里的属性
   * name
     + name属性值与其功能本身没有关系的。
     + 在一个配置文件中可以写多个package标签，name属性不能相同
   * extends
     + extends = "struts-defoult"
     + 属性值固定的，
     + 写了这个属性之后
   * namespace
     + namespace属性值和action标签里面的name属性值构成访问路径

**Action标签**

1. Action标签配置action访问路径

**result标签**

1. 根据action的方法返回值，配置到不同的路径里面去
2. result标签属性
   * name
     + 和方法返回值一样
   * type属性
     + 配置如何到路径中（转发或者重定向）
     + type属性默认值，做转发操作
     + 转发特点：只进行一次。地址栏内容不变，页面内容改变

**struts2常量值配置**

1. 常用的方式
   * 在struts.xml中进行配置
   * <constant name="struts.i18n.encoding" value="UTF-8" />
2. 常用的常量
   * struts.i18n.encoding = UTF-8

* 标签Action

1. Action标签配置action访问路径
2. Action标签熟悉
   1. Name属性

- Name属性值和action标签里面name属性值构成访问路径

- Package标签里可以写多个action标签，但是action中的name属性值不可相同

* 1. Class属性

Action全路径

* 1. Method属性

比如在action里面默认是execute方法，但是在action里面写有其他方法，

让aciton里面多个不同方法执行

* 标签Result

1. 根据action的方法返回值，配置到不同的路径中去（action、页面）
2. result属性
   1. name属性

和方法返回值一样

* 1. type属性

配置如何到路径中去（转发或者重定向）

Tyoe属性的默认值是：转发

-转发：浏览器地址栏的地址不会改变

-重定向：浏览器地址会发生改变

* 标签Constan （常量）

1. 表单提交属性到action里面，在aciton可以获取表单提交数据。
2. 表单提交数据有中文，有乱码问题解决：
   1. Post提交直接设置编码
   2. Get提交做编码转换
3. 如果在action获取表单通过post方式提交中文，中文乱码问题解决了，不需自己去处理问题

* 分模块开发

1. 单独写配置文件，把配置文件引入核心配置文件中
2. 在新建的xml文件中用include标签引入struts.xml文件（主文件包含副文件）

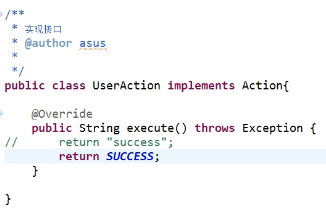


-----配置文件介绍结束

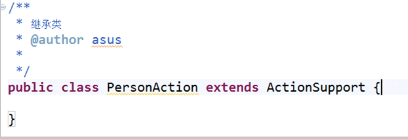
* Action编写方式
  + Action编写3种方式
    1. 创建普通类，不继承任何类，不实现任何接口



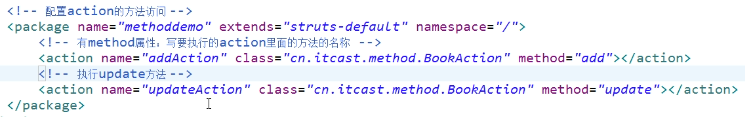
* + 1. 创建类，实现接口Action



* + 1. 创建类，继承actionSupport（其本身已经实现action接口）



* 访问action方法（重点0
  + 有3种方法实现
    1. 使用action标签的method属性，在这个属性里面写入执行的action方法



缺点：做法没问题，写法很麻烦。在方法比较少的情况下个可以使用，多的时候不方便

* + 1. 使用通配符方式实现（重点）
       1. 在Action标签里面name属性值里面写“\*”符号
          - \* 理解，表示匹配任意内容

比如访问hello, \*可以匹配到

比如访问 add方法 \*可以匹配到

Method的值要取\*的值，用 { 1 } , （ 1代表name中第几个 \* ）





* + 1. 动态实现



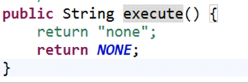
Struts配置是一般的配置方法



* + 错误提示
    1. 如果没有action返回值与配置文件中的result不一致，出现404错误、



* + 1. 在action里面的方法返回值，如果有返回值的时候返回值的类型必须是String
    2. Action里面的方法可以返回没有返回值，没有返回值的时候在resulte里面不需要配置返回值
       1. 把方法写出void
       2. 让返回值，返回一个”NONE” (要继承action类)

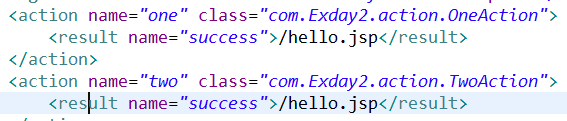


# 第二章

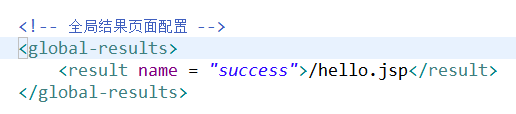
## 结果页面配置

* 1. 全局结果页面
  2. 局部结果页面
  3. Result标签type属性

1. Result标签配置action方法得返回值到不同得路径里面
2. 创建两个action，执行默认得方法execute方法，让两个action的方法都返回successs，
   1. 如果多个action，方法里面返回值相同的，到页面也相同的，这个时候可以使用全局结果页面配置

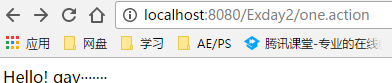


* 1. 全局配置这样

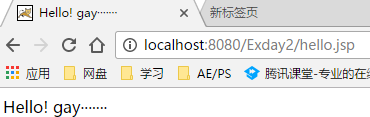
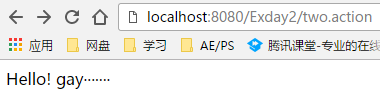


当配置右单个action 与全局配置，最终以局部为准！

1. Result标签里面除了name属性之外，还有一个属性type
   1. Type属性：如何到路径里面（转发还是重定向）
2. Type属性值
   1. 默认值，做转发操作 值“dispatcher” （一次请求）



* 1. 做重定向操作，值是“redirect” （两次请求）



* 1. 上面的dispatcher、redirect，这两个值一般针对到页面中配置，配置到其他action

-chain： 转发到action，一般不用，缓存问题

-redirectAction：重定向到action



## 在action获取表达提交数据

1. 使用actioncontent类获取
2. 使用serveractioncontext类获取
3. 使用接口注入方式获取

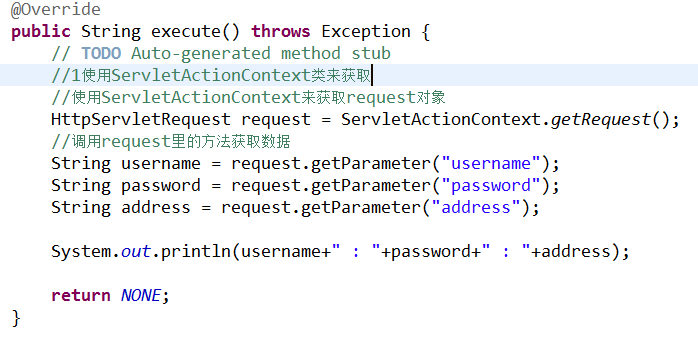
* 之前web阶段，提交表单到servlet里面，在servlet里面使用request对象里面的方法来获取：getParameter , getParameterMap
* 现在提交表单到action，诞生action没有request对象，不能直接使用request对象
* Action获取表单数据只要有三种方式
  + 使用ActionContext类



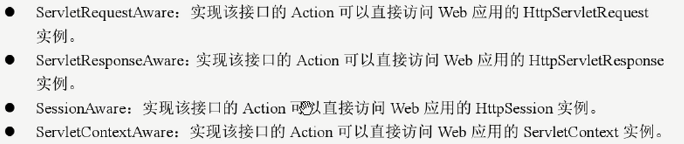
1. 因为方法不是静态方法，需要创建ActionContext类对象
2. 这个对象不是new出来的



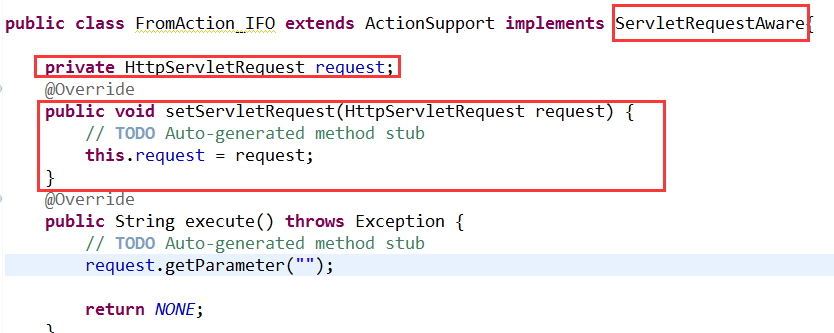
1. 具体步骤
   1. 创建表单，提交表单action里面
   2. 在action里面获取ActionContext数据
   * 使用servleActionContext



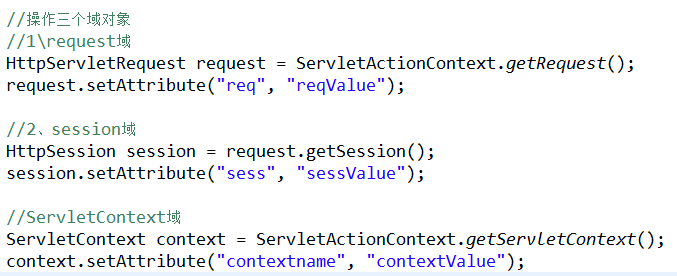
* + 使用接口注入方式



1. 让action实现接口，为了得到request对象



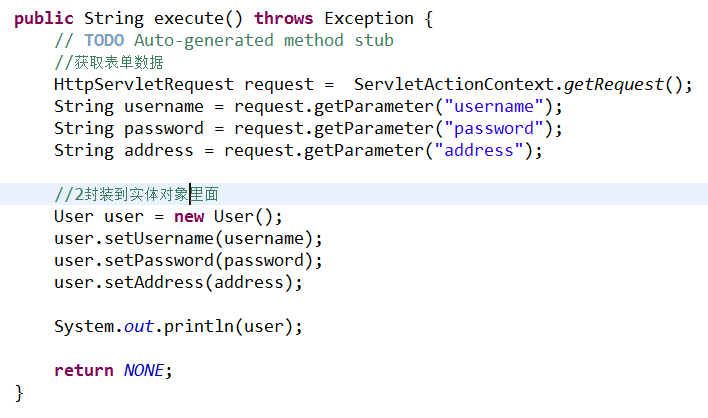
* + 在action操作域对象
    1. Request、seesion、servletContext
    2. 使用servletContext类操作



## Struts2提供获取表单数据方式

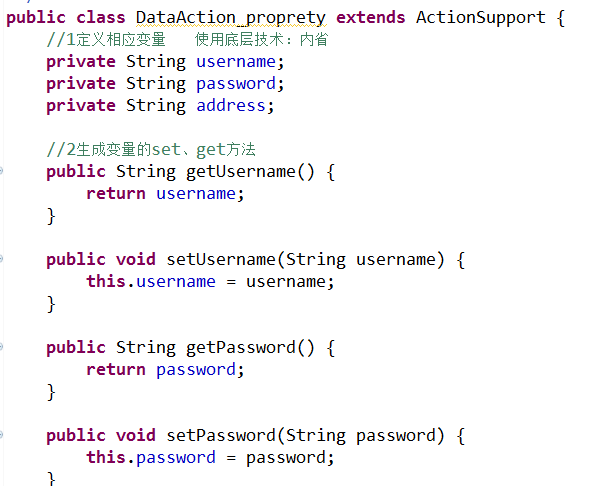
1. 属性封装
2. 模型驱动封装

* Struts2获取表单数据的方式
  1. 使用最原始的方式获取表单封装到实体对象中
     1. 需要创建一个实体类User
     2. 在action中这样写



* 1. 属性封装、

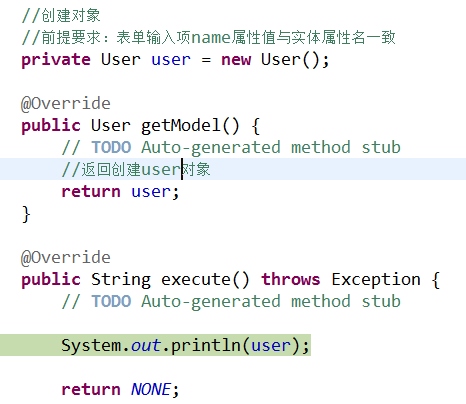
1. 直接把表单提交封装到action的属性里面
2. 实现步骤
   1. 在action的成员变量位置定义与表单中属性值相同的变量
   2. 生产变量set方法（建议get方法页写出来）



1. 使用属性封装获取表单数据到属性里面，不能把数据直接封装到实体类对象里面
   1. 模型驱动封装（重点）
      1. 使用模型驱动方式，可以直接将表单数据可以直接封装到对象中去
      2. 实现步骤
         1. Action实现接口 ModeIDriven



* + - 1. 实现接口里面的方法GetModel方法

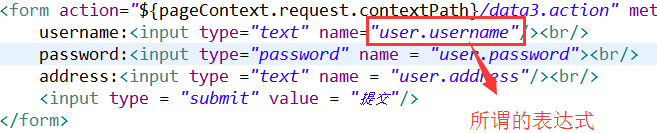


* + - 1. 在action里面创建实体类对象
    1. 使用模型驱动和属性封装注意问题：
       1. 在一个action中，获取表单数据可以是属性封装，也可以使用模型驱动封装。但是不能同时使用两个封装方式同时获取同一个数据。

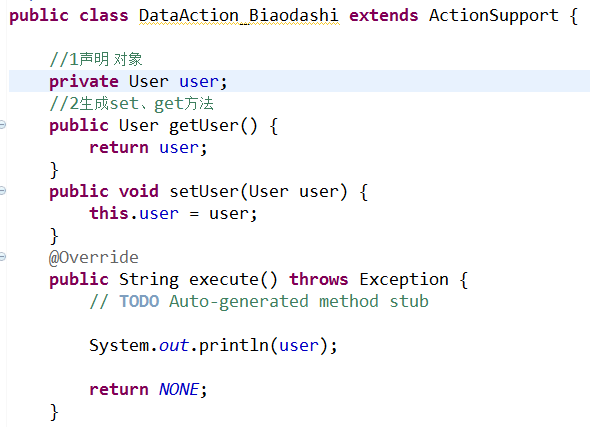
如果同时使用：默认使用模型驱动封装

* 1. 表达式封装（会用）
     1. 实现过程
        1. 使用表达式封装也可以把表单数据封装到实体类对象里面
        2. 在action里面声明实体类
        3. 生成实体类变量的set、get方法
        4. 在表单输入项name属性里面写表达式

表单中的代码：



Action的代码：



表达式封装也就是属性封装所以归为属性封装，但特性有必要独立出来

* 1. 比较表达式封装和模型驱动封装
     1. 使用表达式封装和模型驱动封装都可以把数据封装到实体类对象里面
     2. 不同点
        1. 使用模型驱动封装把数据封装到一个实体类对象里面。

在一个action里面不能使用模型把数据封装到不同的实体类对象里面

* + - 1. 使用表达式封装可以吧数据封装到不同的对象中去

## Struts2获取数据封装到集合中

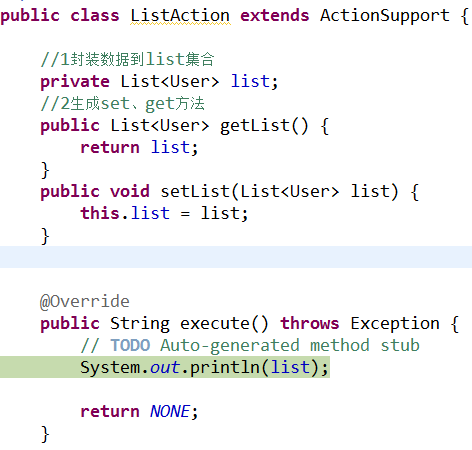
1. 封装到list集合
2. 封装到map集合

* 封装数据到list集合
  1. 在action里面声明List
  2. 生成list变量的set、get方法
  3. 在表单中输入类似表达式的的写法

表单代码：



Action代码：



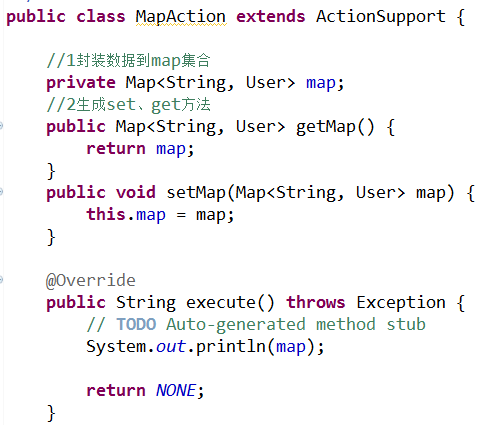
* 封装对象到Map集合

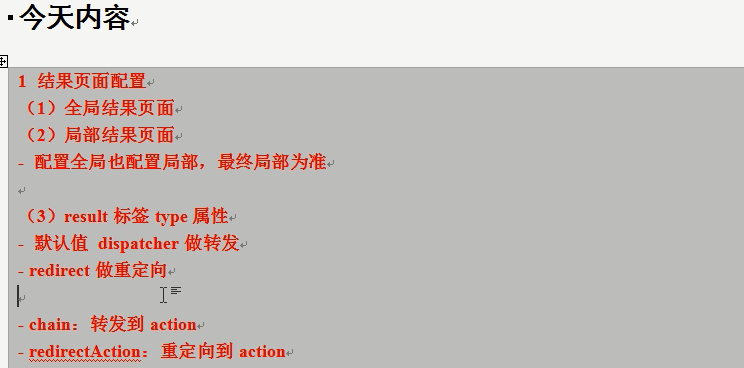
1. 生成Map集合
2. 生成get和set方法
3. 在表单输入项中写入表达式

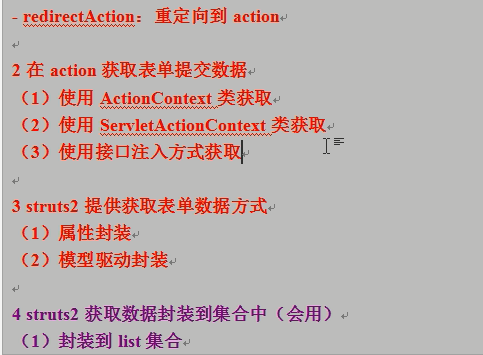
表单代码：



Action代码：

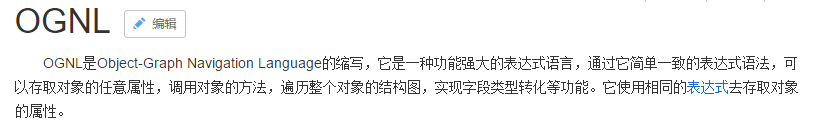


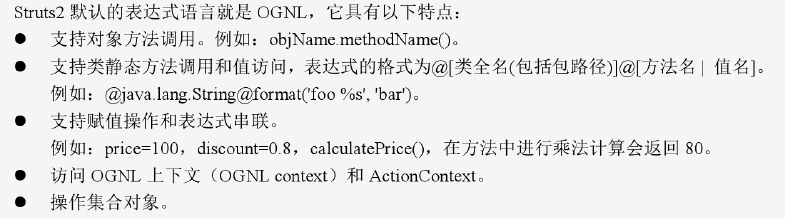




第三章

## Ognl概述





* 之前web学的El表达式在jsp中获取域对象
* Ognl是一种表达式，这个表达更加强大
  1. 在struts2里面操作值栈数据
  2. 一般把ognl在struts2操作：和struts2标签一起使用值栈
* OGNL不是strus2的一部分，是单独项目，只是经常与struts2一起使用
  1. 使用ognl时候要导入jar包，struts已经有了

#### Ognl入门案例

* 使用ognl + struts2标签实现计算字符串长度



* 1. 在java代码中，调用字符串.lenth();
* 使用Struts2标签
  1. 使用jstl时候，导入jar之外，在jsp页面引入标签库
  2. 使用Ognl，同样需要导入jar，和在jsp页面进入标签库



## 2.什么是值栈

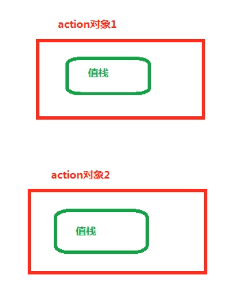
* 在web阶段，在servlet里面进行操作，把数据放到域对象里面，在页面中使用ognl表达式获取，域对象一定范围，存值和取值。
* 在struts2里面本身提供一种存储机制，类似于域对象，是值栈，可以存值和取值
  + 在action里面把数据放到值栈，在页面中获取到值栈数据

##### servlet和action区别

* Servlet：默认在第一次访问时候创建，创建一次
* Action：访问时候创建，每次访问Action时候，都会创建action对象，创建多次

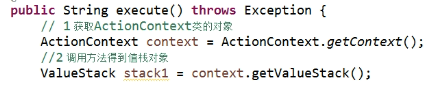
##### 值栈存储位置

* 每次访问action时候，都会创建action对象，
* 在每个action对象里面都会有一个值栈对象

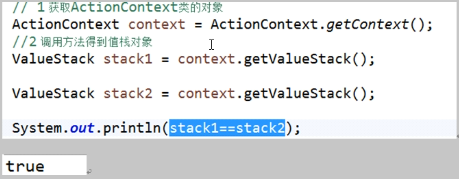


## 3. 如何获取值栈对象

* 获取值栈对象有多种方式
  + 使用一个类ActionContext类里面的方法得到值栈对象

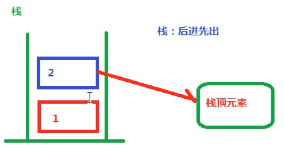


* + 每个action对象中只有一个值栈对象

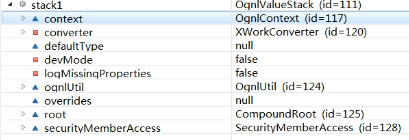


## 4. 值栈内部结构

* 栈的特点



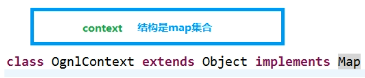
* 值栈分为两部分



1. 第一部分ROOT,本身是list集合
   1. 一般是root数据



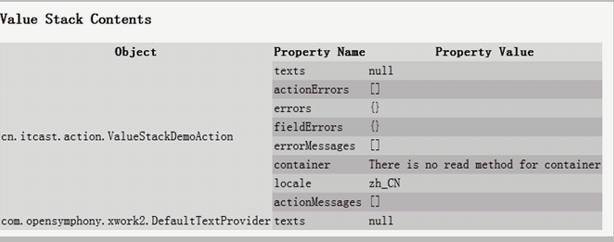
1. 第二部分context，结构map集合



Map部分：



### S : debug标签

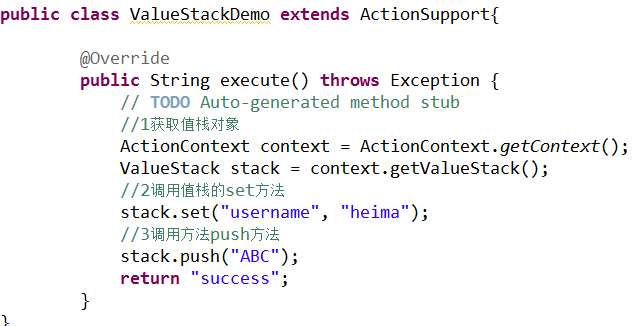


-Action对象里面有值栈对象

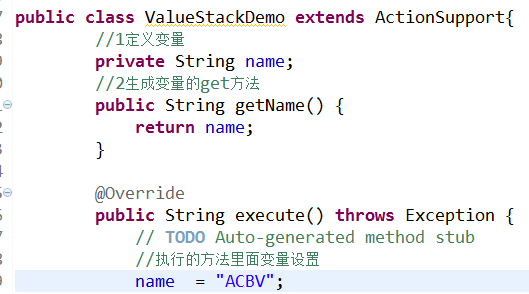
值栈对象里面有action的引用

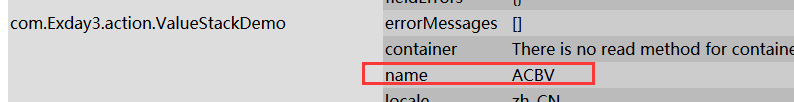
## 5. 向值栈放数据

* 向值栈放数据多种方式
  + 获取值栈对象，调用值栈对象的set方法
  + 获取值栈对象，调用值栈对象里面的push方法
  + 在action定义变量，生成变量get方法（常用）
* 调用值栈对象的set方法、push方法



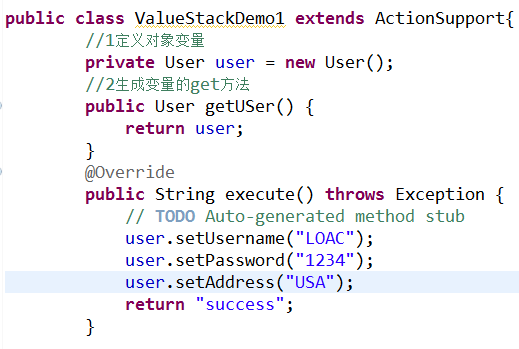
* + 在action定义变量，生成变量get方法（常用）

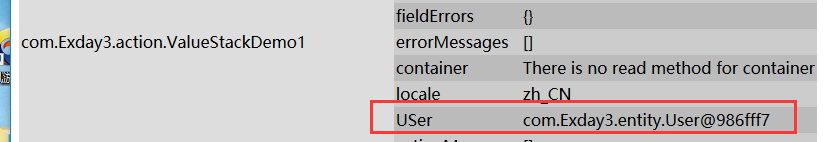




### 向值栈存放对象

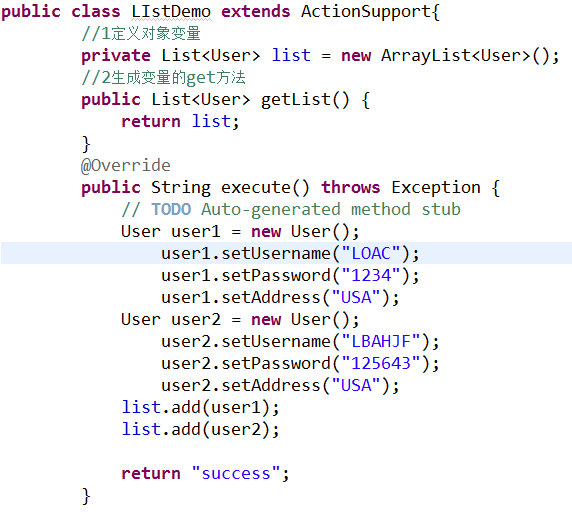
* 实现步骤
  + 定义一个对象变量
  + 生成变量的get方法
  + 在具体执行的方法里面想对象中设置值

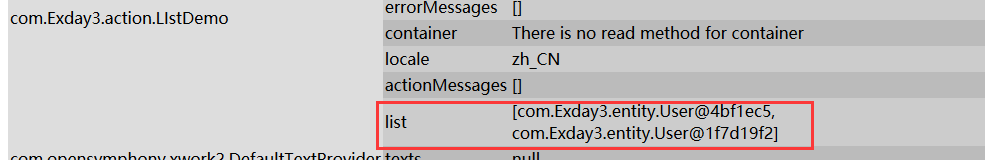




### 向值栈放list集合

* 实现步骤
  + 第一步定义list集合
  + 生成变量的get方法
  + 在执行方法里面向list集合设置值

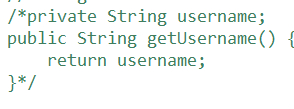


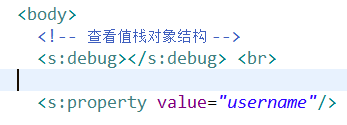


## 6. 从值栈获取数据

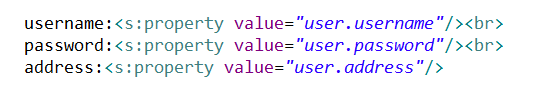
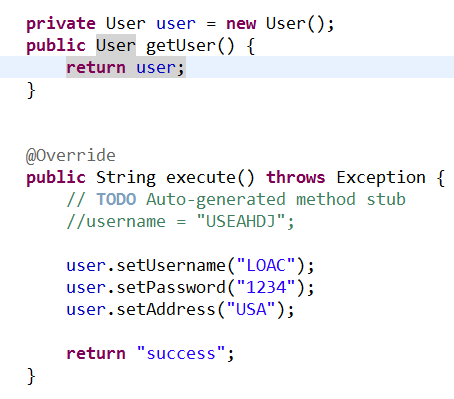
### 从值栈获取字符串

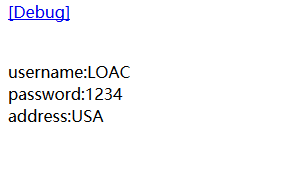
* 使用struts2的标签和Ognl表达式获取值栈对象数据
  + <S : property value = “ognl表达式”>





### 从值栈获取对象





### 从值栈中获取list集合

## 7. 其他方法从值栈获取数据（了解即可）

### 使用set方法向值栈存放数据，获取数据



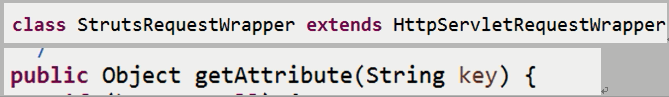
### 使用push方法向值栈存放数据，获取数据



* 使用push方法设置值，没有名字，只有设置的值
* 向值栈存放数据，向一个数组中存放数据，数组名: top
* 

## 8. EL表达式为什么能获取数据？

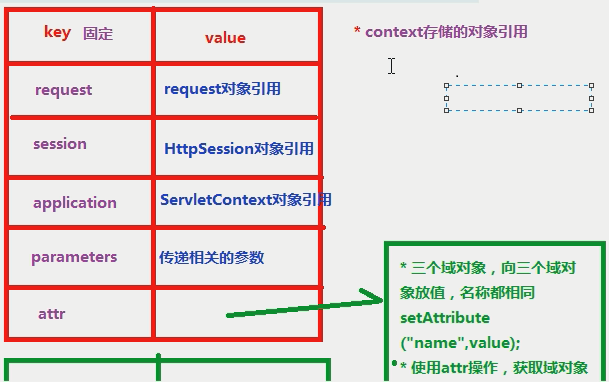
* EL表达式获取域对象
* 向域对象里面放值使用setAttribute方法
* 底层增强request对象里面的方法getAttribute方法
  + 首先从request域获取值，如果能获取到则直接返回
  + 如果从request域获取不到，到值栈中把值获取出来，把放到域对象中
* 查看源代码



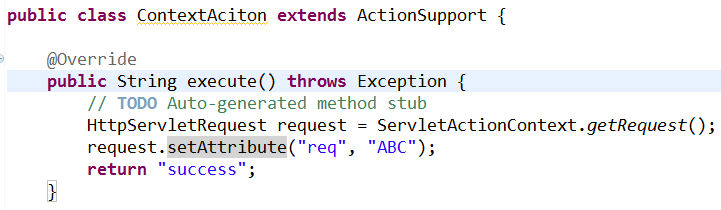
## 9. OGNL的#、%使用

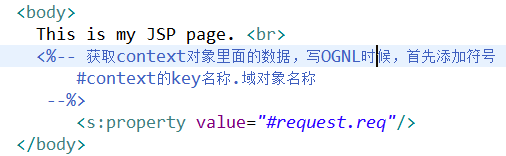
### #的使用

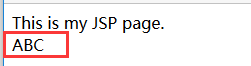
* 使用#获取context里面的数据



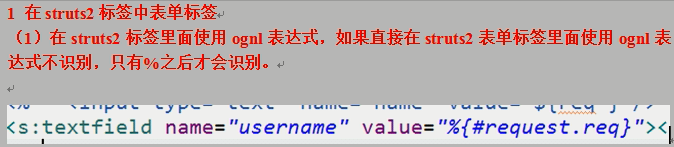
* 操作演示
  + 向request存放值
  + 在页面中使用ognl表达式获取







### %的使用

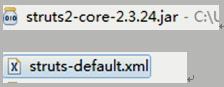


# 第四章

## 拦截器概述

### Struts2拦截器概述

* Struts2是框架，封装了很多的功能，struts2里面封装的很多功能都是封装在拦截器里面
* Struts2里面封装了很多的功能，有很多拦截器，不是每次这些拦截器都在执行，每次执行默认拦截器
* Struts2里面默认拦截器的位置





* 拦截器什么时候执行？

在action对象创建之后和 action方法之前执行

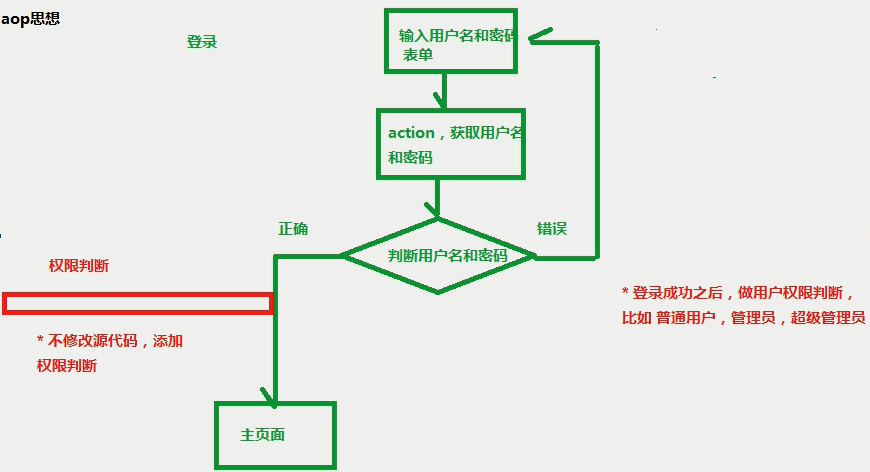
## 拦截器底层原理

* 拦截器底层使用两个原理
  + AOP思想
    - 文字描述

Aop是面向切面（方面）编程、有一个基本功能，想扩展功能，不通过修改源代码方式扩展功能

* + - 图文分析

不改源代码加功能

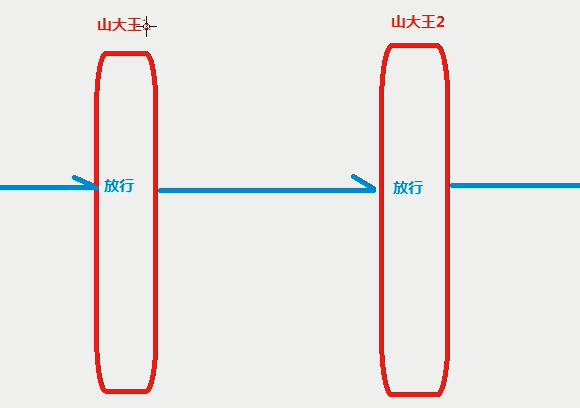


* + 责任链模式
    - Java中有很多设计模式，责任链模式是其中一种
    - 责任链与过滤链很相似

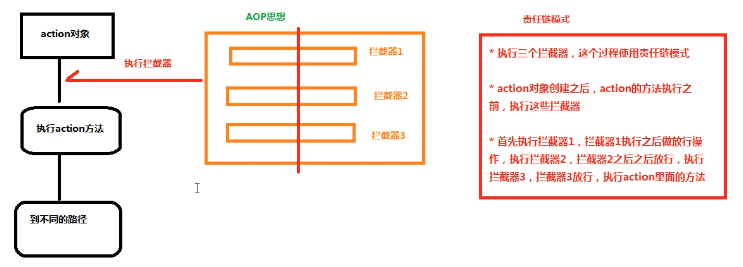
要执行多个操作：添加、修改、删除三个操作。

首先执行添加操作，添加操作执行之后，做雷士放行操作，执行修改操作，修改操作执行后，一样。

过滤链：一个请求有多个过滤器进行过滤，每个过滤只有放行才能到下一个操作



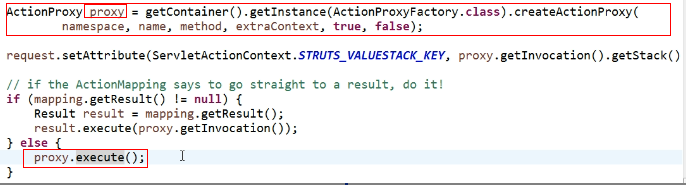
* AOP思想和责任链模式如何应用到拦截器里面
  + 文字描述
    - 拦截器在action执行之前，和action的方法执行之后，执行默认拦截器执行过程中用到AOP思想，在action里面没有直接调用拦截器方法，使用aop思想配置文件方式调用拦截器
    - 在执行拦截器的时候，执行很多的拦截器，这个过程是用责任链模式
    - 假如执行三个拦截器，执行拦截器1，执行拦截器1之后放行操作，之后2,3拦截一个样子。得知拦截去是先执行。执行完之后才去执行action方法
  + 图示



* 查看代码
  + 1、执行action



* + 2、创建action对象，使用动态代理方式创建



代理对象不是对象，但是具有对象的所有功能！

* + 放行，执行很多拦截器



## 过滤器和拦截器区别

* 过滤器和拦截区别
  + 过滤器
    - 定义：过滤器理论上可以任意内容，比如html、jsp、sevlet、action也可以也可以过滤
    - 在服务器启动时创建
  + 拦截器
    - 定义：拦截器只可以拦截action
* Sevlet和action区别
  + Servlet
    - 定义：servlet默认第一次访问时候创建，创建一次，单市实例对象
    - Action每次访问都创建一个实例对象。

## 自定义登录拦截器

* 在struts2里面有很多拦截器，这些拦截器是struts2封装的功能，但是在实际开发中struts2里面的拦截器中可以没有要使用的功能，，这个时候需要自己写拦截器实现功能。
* 拦截器结构
  + 查看源代码（modelDriven模型驱动为例）
    - 继承类：



在接口里面有三个方法

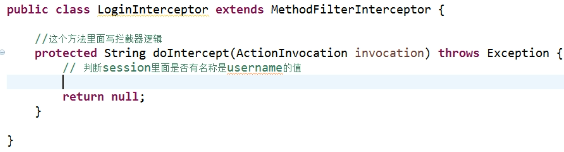


* 在开发中建议使用另一种方法

- 1、继承MethodFilterIntercept类实现让action里面某个方法不 进行拦截

- 2、截器和action调用拦截器方法，而通过配置文件方式让建立关系

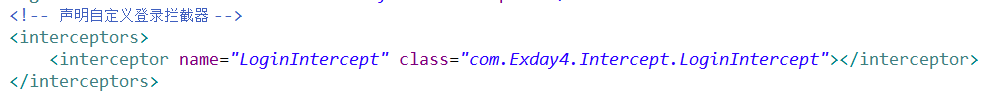
* 自定义登录拦截完器
  + 需求：在项目中，有很多的action的超链接，实现只有登录状态，才可以点击action的才能链接实现功能，如果不是登录状态，点击action 超链接返回登录页面
  + 登录的状态：使用session域对象实现
    1. 登录成功之后，就把数据放到session里面
    2. 判断session是否有值，可以知道是否是登录状态
  + 实现登录的基本功能
    1. 查询数据库判断用户名和密码
  + 添加登录拦截器功能
    1. 判断是否登录：是判断session里面的是否有登录数据
    2. 拦截器实现过程
       1. 创建类，继承MethodFilterIntercept类



* + - 1. 重写MethodFilterIntercept类的方法，写拦截器逻辑



* + - 1. 配置action和拦截器关系（注册拦截器）
         1. 在要拦截的action标签所在的package声明拦截器

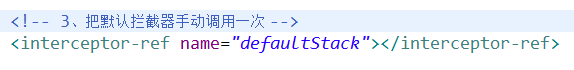


* + - * 1. 在具体的action标签里面使用声明拦截器

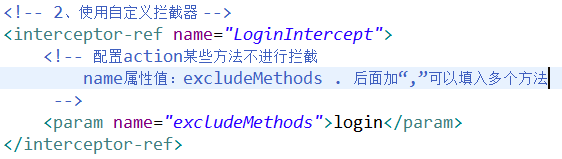


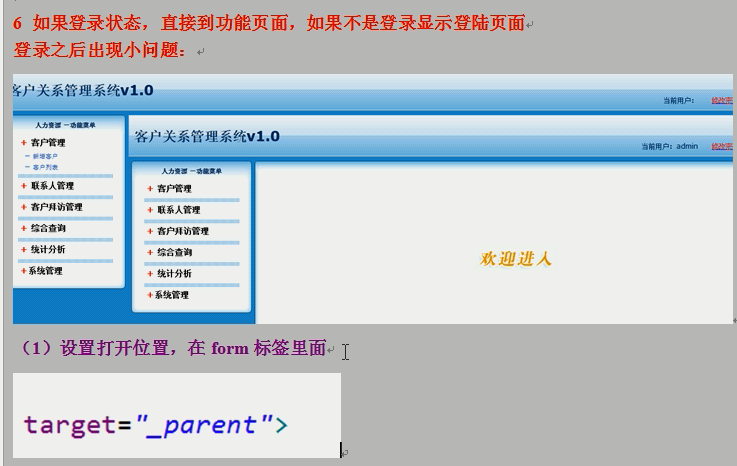
* + - * 1. Struts2里面执行很多的默认拦截器，但是如果在action里面配置自定义拦截器，则问题：默认拦截器不会执行。

解决方案：1、把默认拦截器手动调用一次！

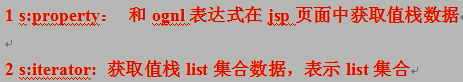


* + 1. 配置拦截器，对action里的所有方法都进行拦截，
       1. 在action里面有login方法，这个方法需要拦截，如果拦截了，问题：永远登录不进去。
       2. 解决：让login方法不进行拦截
          1. 直接通过配置方式让action里面某些方法不进行拦截。

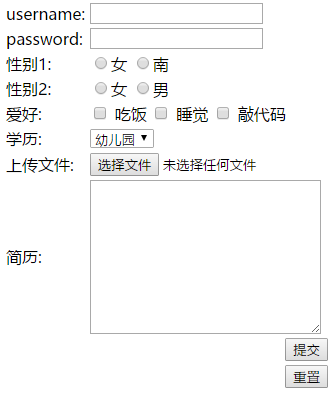




## Struts2标签库（会用）





* Struts2表单标签
  + Html中的标签
    1. Form：action、method、enctype
    2. 输入项input = 值
       1. Text普通文本输入项
       2. password密码输入项
       3. radio单选输入项
       4. checkbox复选框输入项
       5. file文件上传项
       6. hidden隐藏项
       7. button按钮
       8. submit提交项
       9. image图片提交
       10. reset重置按钮
       11. select下拉输入项
       12. textarea文本域
    3. 在struts2中大部分标签都有

