


Struts2 拦截器概述

- 1、struts2 是框架，封装了很多的功能， Struts2 里面封装很多功能都是在拦截器里面
- 2、Struts2 里面封装了很多功能，有很多拦截器，不是每次这些拦截器都执行，每次执行默认拦截器
- 3、Struts2 里面默认拦截器位置

 struts-default.xml

```
1      <interceptor-stack name="defaultStack">
2          <interceptor-ref name="exception"/>
3          <interceptor-ref name="alias"/>
4          <interceptor-ref name="servletConfig"/>
5          <interceptor-ref name="i18n"/>
6          <interceptor-ref name="prepare"/>
7          <interceptor-ref name="chain"/>
8          <interceptor-ref name="scopedModelDriven"/>
9          <interceptor-ref name="modelDriven"/>
10         <interceptor-ref name="fileUpload"/>
11         <interceptor-ref name="checkbox"/>
12         <interceptor-ref name="datetime"/>
13         <interceptor-ref name="multiselect"/>
14         <interceptor-ref name="staticParams"/>
15         <interceptor-ref name="actionMappingParams"/>
16         <interceptor-ref name="params"/>
17         <interceptor-ref name="conversionError"/>
18         <interceptor-ref name="validation">
19             <param
name="excludeMethods">input,back,cancel,browse</param>
20         </interceptor-ref>
21         <interceptor-ref name="workflow">
22             <param
name="excludeMethods">input,back,cancel,browse</param>
23         </interceptor-ref>
24         <interceptor-ref name="debugging"/>
25         <interceptor-ref name="deprecation"/>
26     </interceptor-stack>
27
```

4、拦截器什么时候执行

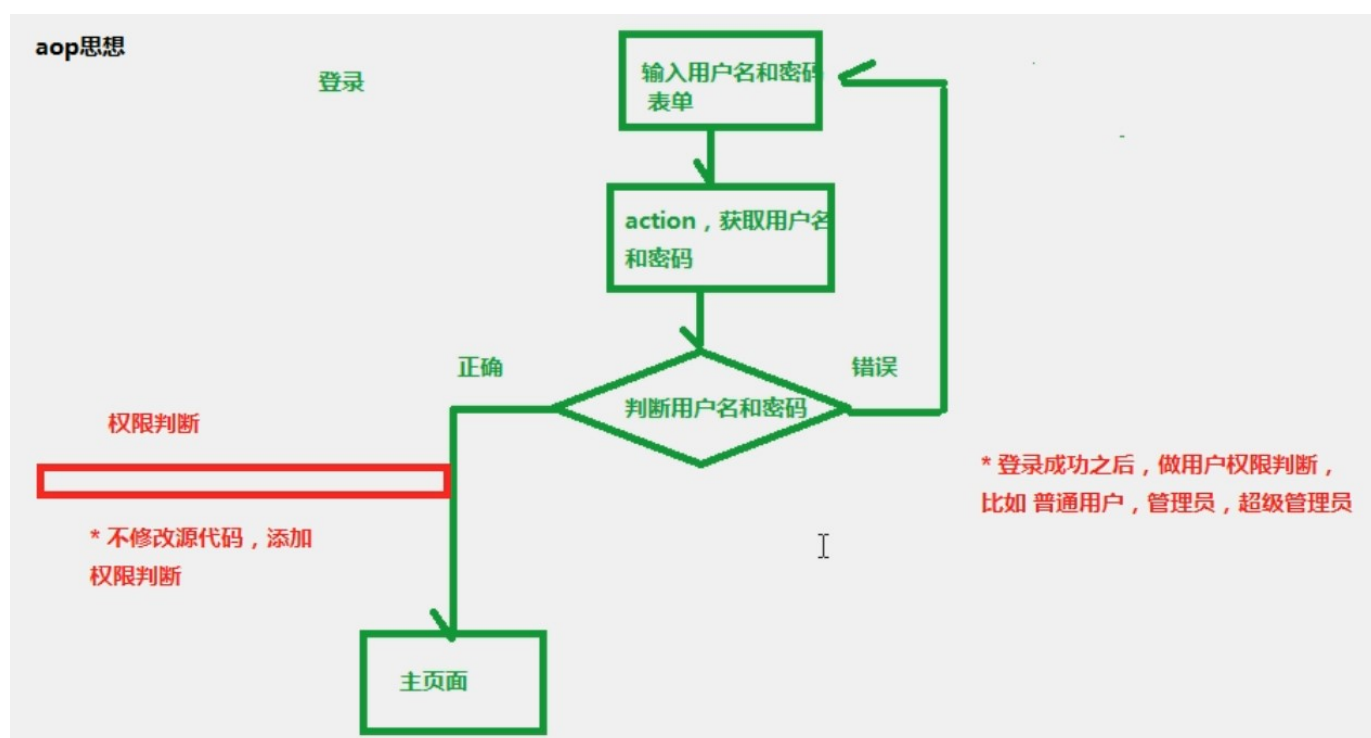
- 在 Action 对象创建之后，Action 的方法执行之前。

Struts2 拦截器底层

1、拦截器底层的两个原理

1)、aop 思想

Aop 面向切面 (方面) 编程,有基本功能, 拓展功能, 在不修改源代码的情况下增加功能



2)、责任链模式

- 1、在 [java](#) 中有很多的设计模式, 责任链模式是其中的一种
- 2、责任链模式和过滤链很相似的

责任链模式:

例如: 要执行多个操作, 有添加, 修改, 删除三个操作。

首先要执行添加操作, 添加操作执行之后, 做类似于放行操作, 执行修改操作, 修改操作执行之后类似于放行操作, 执行删除操作。

过滤链：一个请求可有多个过滤器进行过滤，每个过滤器只有做放行才能到下一个过滤器。

2、aop思想和责任链模式如何应用到拦截器里面

1)、文字描述

aop思想：

拦截器在Action 对象创建之后， Action 的方法执行之前执行

在Action 方法执行之前执行默认拦截器，执行过程使用 aop 思想，在 Action 没有直接调用拦截器的方法，使用配置文件方式进行操作。

责任链：

在执行拦截器的时候，执行很多的拦截器，这个过程使用 aop 思想

假如执行三个拦截器，执行拦截器1，执行拦截器2，执行拦截器1之后作放行操作，执行拦截器2，执行拦截器2之后做放行，执行拦截器3之后作放行，执行action

2)、画图描述



重要的概念

1、过滤器和拦截器区别

- 1)、过滤器：过滤器理论上可以过滤任意内容，比如 html，jsp，servlet，图片路径
- 2)、拦截器：拦截器只拦截 action

2、Servlet 和 Action 区别

- 1)、servlet 默认第一次访问时候创建，创建一次，单实例对象
- 2)、Action 每次访问时候创建，创建多次，多实例对象。

自定义拦截器

1、开发过程中，建议使用 MethodFilterInterceptor 的方式创建拦截器

- 写类，继承 MethodFilterIntercept 类是实现
- 让 Action 里面的某个方法不进行拦截

2、让拦截器和 Action 有关系

- 不是在Action 调用拦截器的方法，而是通过配置文件方式来建立关系

3、拦截配使用配置方法

第一步：编写前端页面这里

略

第二步：编写拦截器业务逻辑

- 类要继承 `MethodFilterInterceptor` 重写 `doIntercept` 方法
- `return invocation.invoke();` 表示放行

```
1 public class LoginInterceptor extends MethodFilterInterceptor {
2
3     @Override
4     protected String doIntercept(ActionInvocation invocation) throws Exception
5     {
6         Map<String, Object> session =
7         invocation.getInvocationContext().getSession();
8
9         User user = (User)session.get("user");
10        if( user!=null ) {
11            return invocation.invoke();
12        }
13        return Action.LOGIN;
14    }
```

第三步 配置 Action 和拦截器的关系（注册拦截器）

1)、在要拦截的 Action 标签所在的 package 里面声明拦截器

```
1      <interceptors>
2          <interceptor name="authlogin"
class="daiwei.learning.struts.interceptor.LoginInterceptor"></interceptor>
3      </interceptors>
```

2)、在具体的 Action 标签里面使用声明的拦截器（拦截器会对 Action 中的所有方法进行拦截）

```
1 <interceptor-ref name="authlogin"/>
```

3)、struts2 里面执行很多的默认拦截器的，但是如果在 Action 里面配置自定义拦截器

问题：默认拦截器不会执行了

解决：吧默认拦截器手动使用一次

```
1 <interceptor-ref name="defaultStack"/>
```

4、配置不拦截方法

```
1 <param name="excludeMethods">login</param>
```