# Structs2

## <include/>

在分模块开发的时候，有时候每一个人写的Structs.xml的名称是不一样的，所以需要主配置文件对该文件进行一个合并加载

<include file=”\*.\*.\*.xml”/>

## Action的写法

实现接口的方式：impelement action

提供了几个常量值：

Success：

Error：

Login：登陆出错，跳转到登陆页面

Input：表单校验出错

None：不跳转

实现接口2的方式：implement actionsupport

该接口提供了许多支持的功能

## Action的访问

1通过方法的方式访问：method指定，但是太繁琐

2通过通配符的方式访问：

例如：

<action name=”\*\_\*” class=”xxx.{1}” method=”{2}”/>

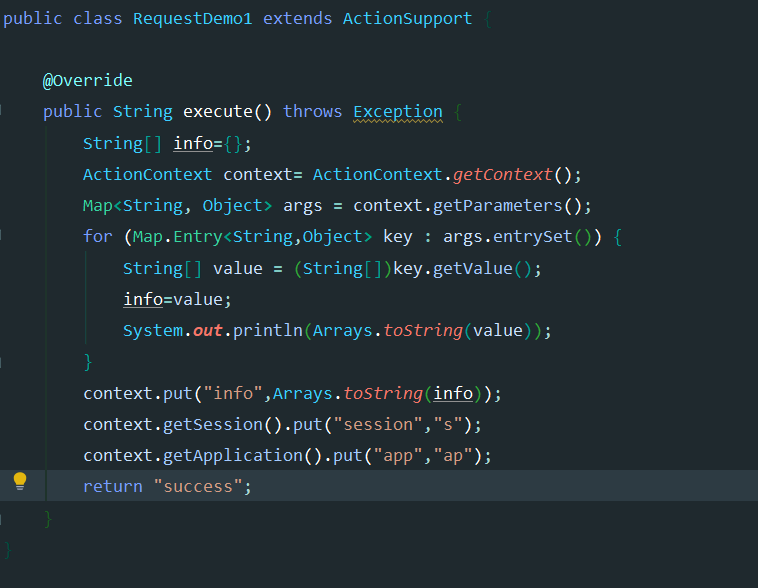
\*\_\* 分别代表着通配符 {1}，{2}

例如URL：/product\_find.action {1}=product {2}=find

# Struts和Servlet的api解耦合

方式一：

解耦合方式：我们看不到request对象，就可以进行传递。实现原理用的是actioncontext对象进行的整合

方式二：

用ServletActionContext对象进行实例出request的对象。

这样就能够把request的域对象继续获取到了

这样既能够获取数据，又能够操作方法

方式三：

接口注入的方式：实现 ServletRequestAware ，ServletContextAware，ServletResponseAware

# 关于模式：

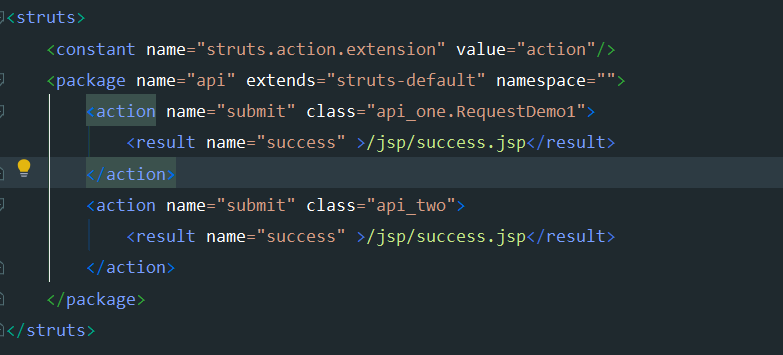
Servlet是单例的，多个程序访问同一个Servlet只会创建一个Servlet实例，会出现线程问题

Action是多例的，多个程序访问同一个action会创建多个Servlet实例，不会出现线程问题

所以，对于成员变量，问题action不用考虑安全问题

# 结果页面的配置：

默认采用的转发的形式，对页面进行一个转发操作；

全局结果页面：在包中的配置一次，其他的在这个包中所有的action只要返回了这个值，都可以跳转到这个页面。

逻辑视图“success”对应了实际视图 /jsp/success.jsp

# 数据封装

三种数据封装的方式：Struts是一个web层面的框架，它本身就提供了数据封装的功能

## 属性驱动：提供set，get方法

### 方式一

属性驱动提供setget方法：

属性类： 提供set，get方法

Action类：提供set，get方法

Jsp：form表单提交数据

这样action可以接受到form提交的数据了 ，并且，Struts已经为我们提供了数据类型转换了

### 方式二：页面提供表达式的方式

在页面中为数据封装提供表达式

Action类：

提供私有对象User：

提供User的set和get方法：

提供execute()方法：

Jsp：

表单name=”user.username”

## 模型驱动:采用模型驱动的方式（最常用）

我们如果学过spring，那么我们这里的模型就是mvc中的mode，可以自动封装实体数据

方式：

页面：普通方式

编写action：实现 ModelDriven<User>

Private User user=new User();

Public User getModel(){

Return user;

}

# Input逻辑视图的配置

默认栈下的拦截器都会依次执行

# OGNL和值栈

Ognl：对象图导航语言，是一种功能强大的开源表达式语言，使用这种表达式语言。可以存取java对象的任意属性