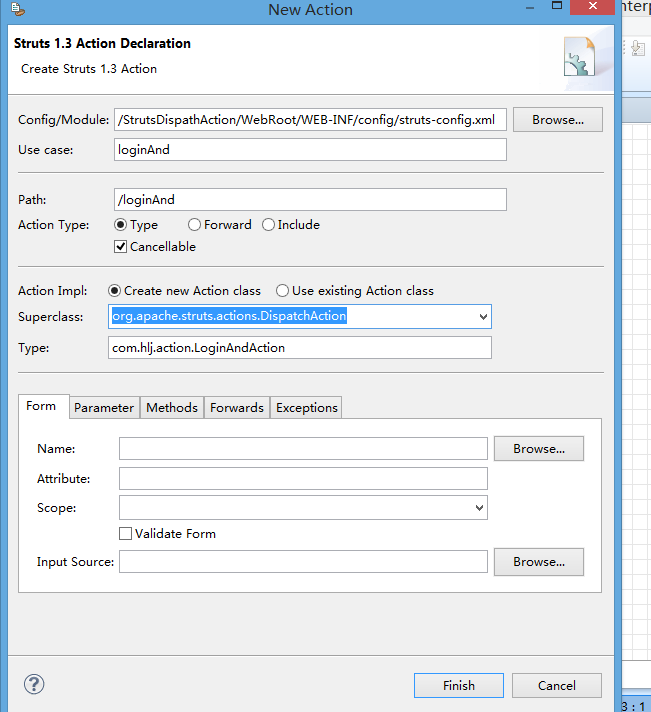


# 1.每个请求都需要一个action，这样比较臃肿，所以我们就出现了这个

# 1.1新建一个action，但是这个继承的是DiapathAction



## 1.2 dispathaction 是分派acion，是action的子类

## 1.1打开confi中，action添加参数，parameter

<!-- parameter用于将来区分，用户不同的请求 ，这个名字可以所以变化-->

<action

attribute=*"userForm"*

parameter=*"type"*

## 1.2使用dispathAction要自己重新命名函数，就是说不通的请求使用不同的方法，但是怎么实现，看1.3中的

**public** **class** LoginAndDisAction **extends** DispatchAction {

/\*使用分派dispathAction

\* 要自己重新命名函数\*/

**public** ActionForward *loginuser* ActionMapping mapping, ActionForm form,

HttpServletRequest request, HttpServletResponse response) {

UserForm userForm = (UserForm) form;// **TODO** Auto-generated method stub

**return** **null**;

}

//用于响应注销请求

**public** ActionForward userloginout(ActionMapping mapping, ActionForm form,

HttpServletRequest request, HttpServletResponse response) {

UserForm userForm = (UserForm) form;// **TODO** Auto-generated method stub

**return** **null**;

}

}

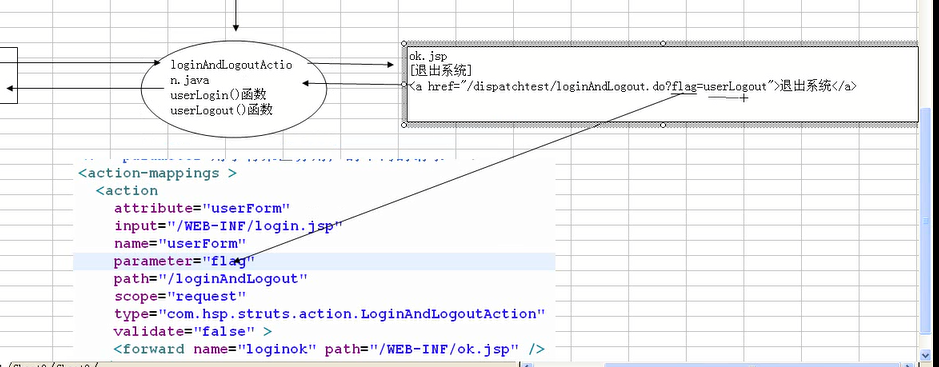
## 1.3 创建登录成功后的loginok.jsp,下面的type就是我们的parameter，然后userloginout是我们早dispatchAction中的一个函数方法，专门用来处理退出登录的

<body>

<h1>登陆成功</h1>

<a href=*"/StrutsDispathAction/loginAndDis.do/?type=userloginout"*></a>

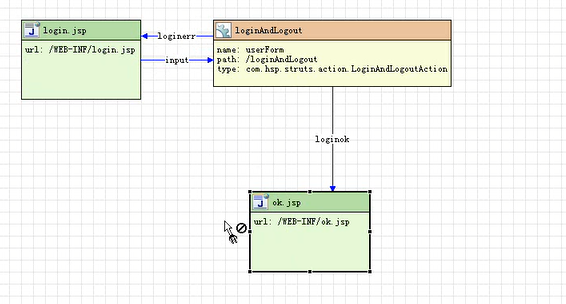
</body>



## 1.4对于登录

<form method=*"post"* action=*"/StrutsDispathAction/loginAndDis.do?type=loginuser"*>

### 1.1然后我们进行测试即可



**public** ActionForward loginuser(ActionMapping mapping, ActionForm form,

HttpServletRequest request, HttpServletResponse response) {

UserForm userForm = (UserForm) form;// **TODO** Auto-generated method stub

**if**(userForm.getPassword().equals("123")){

**return** mapping.findForward("loginok");

}

**return** **null**;

}

### 12用户注销

