《尚学堂》 马士兵 Struts2 Notes

No.1 已经做得笔记，待整理

No.2 Struts2的运行机制（本质：把请求和结果分开）

如果笔者想弄明白struts2的拦截器到底有什么用，必须先从struts2的访问流程来说起。

流程大致如下：

1、用户从浏览器页面发出请求

2、请求被struts2核心控制器（Filter）拦截

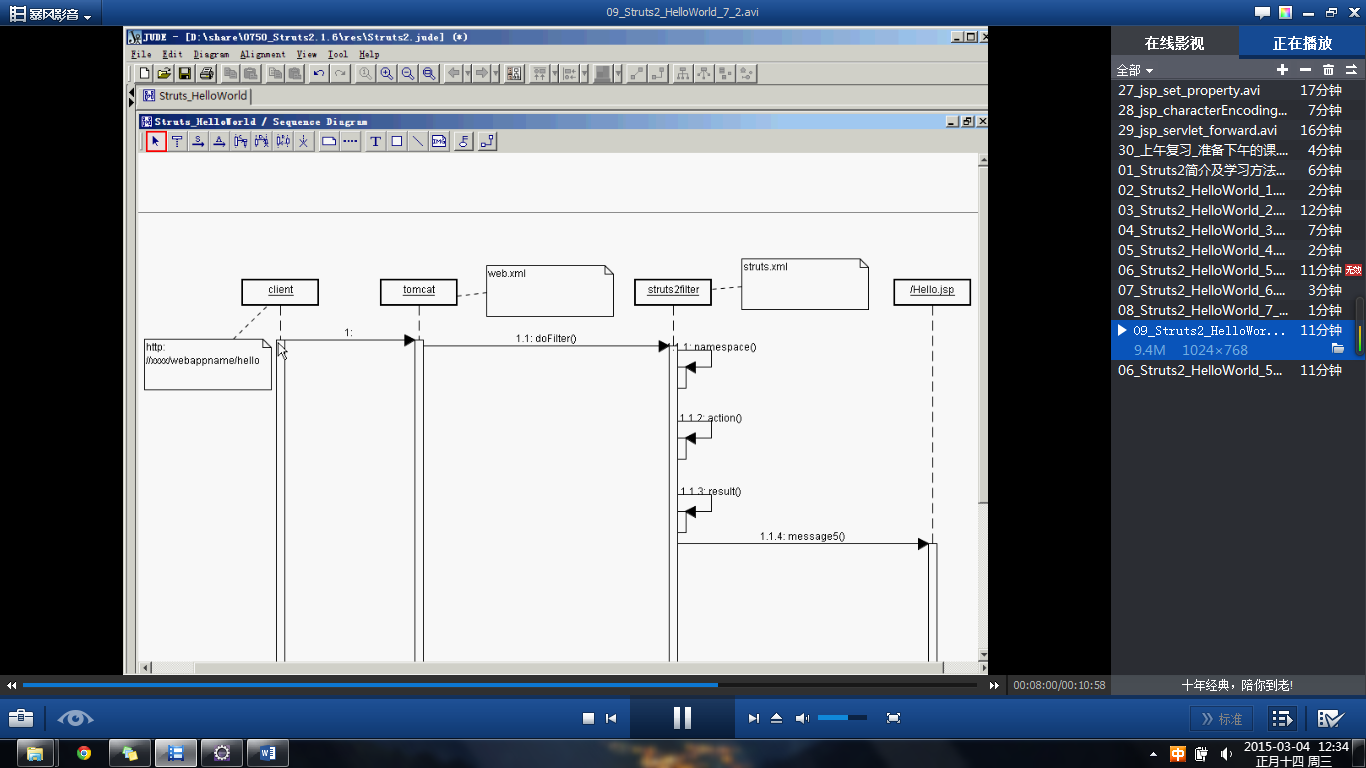
3、请求被验证拦截器、国际化拦截器拦截等等（拦截器的作用）

4、请求被用户Action的代理Action拦截，主要是用来获取前台穿过来的参数

5、访问用户编写的逻辑部分完成任务工作

请求的大致过程就是这样。

struts2的拦截器的作用就是做验证、国际化、和获取参数等等作用。



message5:forward。最后还有一步没有画出，是return(视图)回client

具体描述：

No.3 Namespace

1. Namespace控制着action的访问路径
2. Namespace默认值为空值，此时可以接受所有路径的action
3. Namespace可以为其他的值，如：/、/xxx、/xxx/yyy，此时对应的action访问路径就是他们后面加上/index.action,.action也可以省略

No.4 Struts2中间的package（此时往往package和namespace相对应）是用来区分action（因为不同包里面的action可能重名）重名的情况，作用和Java中package打包一样

No.5 Struts2中间的的Action

1. Struts2中的Action会默认执行一个public String execute()方法，根据其返回的字符串到struts.xml中找到要跳转的页面
2. Action有三种写法
   1. Public class action类名称 {}
   2. Pubic class action类的名称 implements Action {}
   3. Public class action类的名称 extends ActionSupport{} ,标准的是使用此种方法，因为ActionSupport中封装了action需要的类
3. Action并不是必须要有execute()方法
   1. 可以在struts2中的action标签中直接指定要调用的方法
   2. （建议使用这种方法）也可以在URL中动态的指定要调用的方法(DMI：动态方法调用)<http://localhost:8080/Struts2_0300_Action/user!addUser>
4. Action的匹配（含确定信息和\*的时候）问题
   1. 先匹配最精确的那个一个
   2. 当都有\*的时候，按照（文本）先后顺序
5. 提交给action数据的处理
   1. 利用request.getParameter()进行获取
   2. 利用bean进行获取，此时action会自动把提交给自己的参数封装在bean里面，此种方式强烈推荐使用

No.6 Struts2中的路径（namespace）问题

1. struts2中的路径是根据action的路径，而不是jsp路径来确定的，所以尽量不要使用相对路径，虽然可以使用redirect方法解决，但此类方法不建议使用，最简单的方法是统一使用绝对路径：在jsp中使用request.getContextRoot()来拿到webapp的路径加在要跳转的路径之前或者直接把得到的webapp路径赋给jsp中的base标签

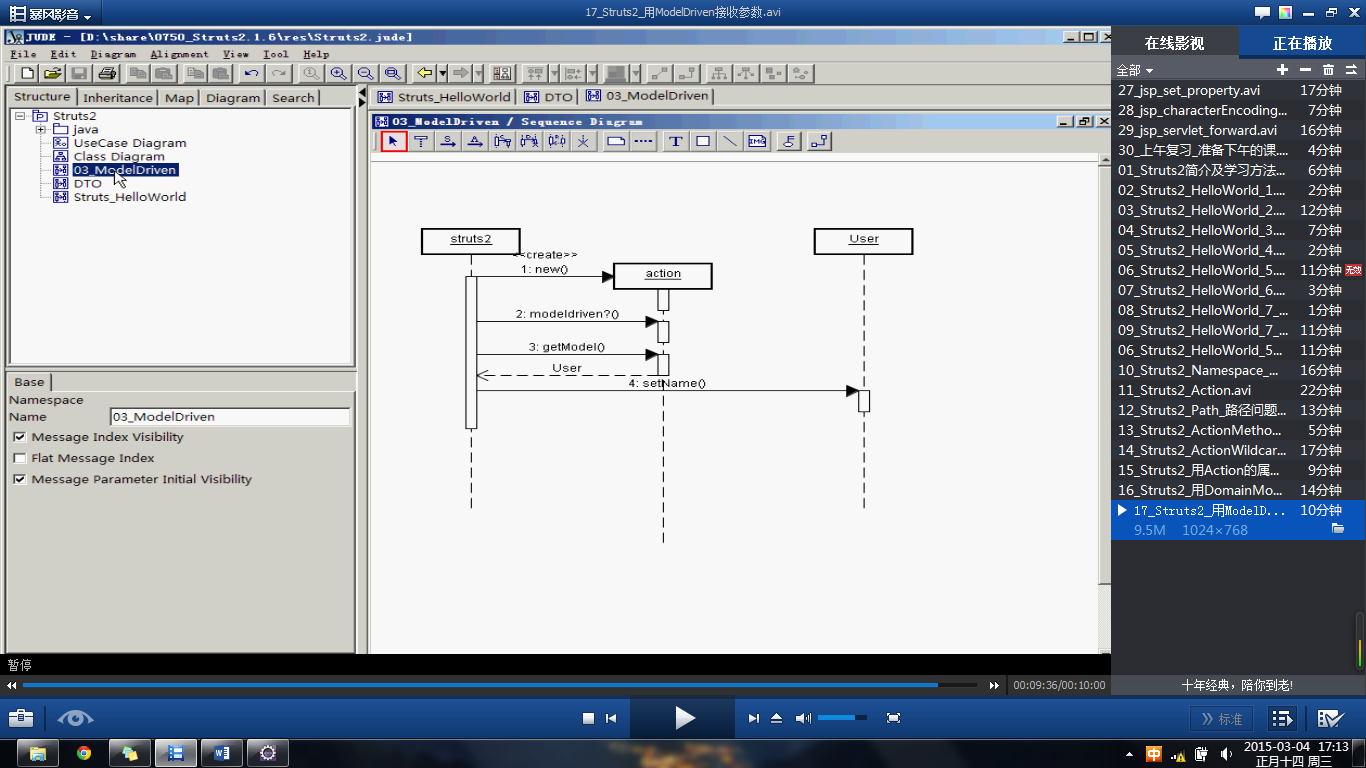
No.7 Struts2中的Model

1. Struts2中的Model是用来存放业务逻辑的，就是数据库里面的属性值，当然也可以存放参数
2. 当需要的属性多余Model的基本属性的时候，可以使用DTO（Data Transform Object）进行参数的过滤，也就是说，新建一个特殊的”Model”用于存放所有的数据，过滤好了之后再存到Model中

No.8 Struts2接受参数

1. 使用action的属性接受参数
2. 使用DomainModel接受参数（就是Model）常使用这一种
3. 使用ModelDriven接受参数此时action要继承ModelDriven<E>并且要有，此种方法使用较少

E getModel() 方法，此时该方法会把参数传递到E中



No.9 Struts2中常见的constant设置

1. <constant name="struts.devMode" value="true" />设置开发者模式，这样当代码改动的时候就不需要重启Tomcat服务器了
2. <constant name="struts.i18n.encodiing" value="utf-8" /> 设置参数传递过程中的编码格式（有可能是Struts2的一个bug）
3. 其他constant可以在struts2-core--.jar下面的default.properties中找到

No.10 数据校验

1. action端：this.addFieldError(“name”,”name is error!”)变量的名称、变量的值
2. view（返回到的页面）使用struts标签：<s:fielderror fieldname=”name” theme=”simple” />

<s:property value=”errors.name[0]”/>

就行了

另外一个实用的标签<s:debug></s:debug>会输出一些编码经常使用的参数

No.11 Context就是指当前的环境（上下文）

No.12 Struts2中web元素的访问（前两种Struts2自动把Map的数据的值自动映射到web元素的值，后两种直接取出web元素）

1. 方法一（Struts2访问Web元素的第一种方法：利用ActionContext.getContext.getXXX()方法获得Web元素的引用）：
   1. Action端：

request = ActionContext.getContext().get("request");

session = ActionContext.getContext().getSession();

application = ActionContext.getContext().getApplication();

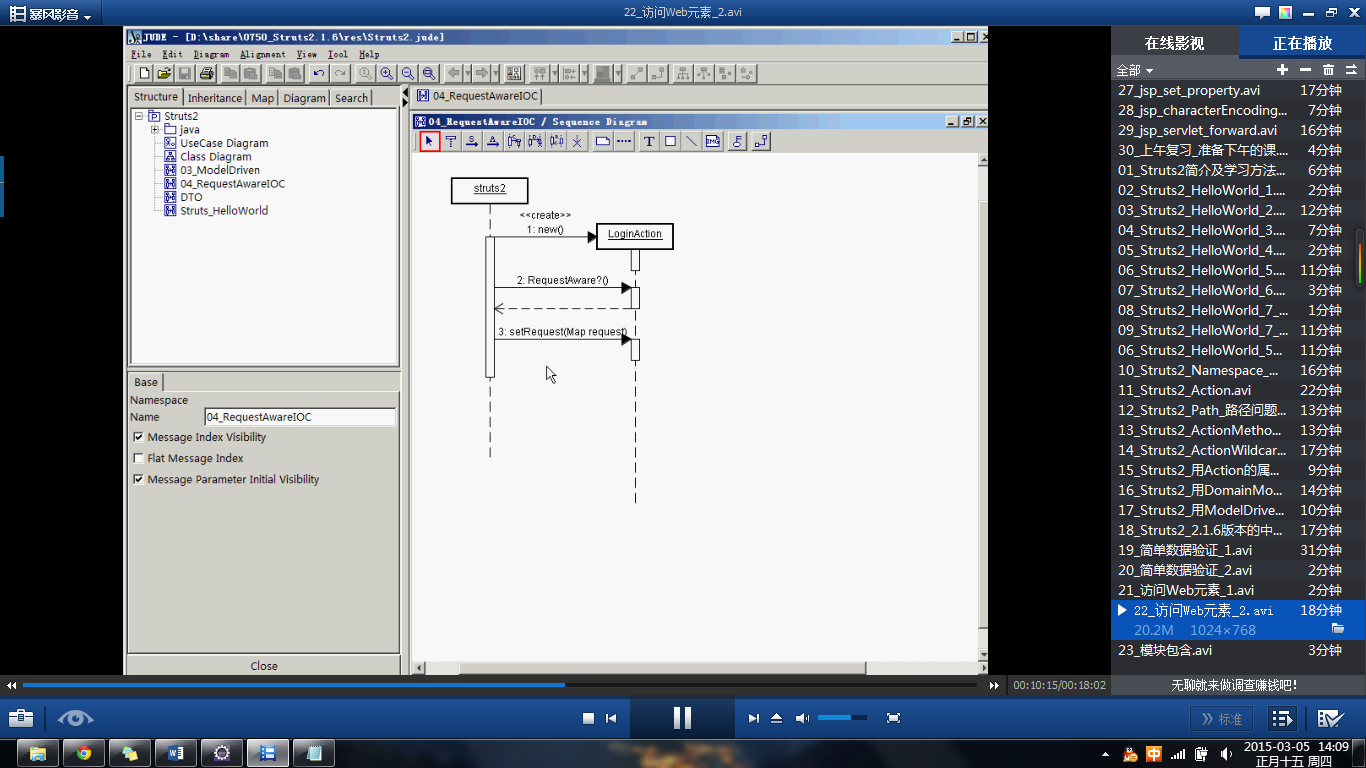
* 1. 返回的页面端（两种取参数的方法）：

<s:property value="#request.username" /> | <%=request.getAttribute("username") %><br>

<s:property value="#session.password" /> | <%=session.getAttribute("password") %><br>

<s:property value="#application.username"/> | <%=application.getAttribute("username") %><br>

1. 方法二（**最常用的，建议使用**）（实现RequestAware、SessionAware和ApplicationAware接口，然后补全他们的setXXX方法）：
   1. 此时用到了一种思想：依赖注入（DI:Dependency injection）或者是控制反转（IoC:Inverse of Control）。依赖注入，就是说要依赖接口进行数据的注入；控制反转，就是说本来是自己主动去取数据的值，现在自己不去取了而是等待“别人”给，控制反转了过来。



图解方法二

* 1. public void setSession(Map<String, Object> session) {

this.session = session;

}

…….

…….

1. 方法三:直接取出Web元素，取出基本的HttpServletRequest对象的引用，然后依次取出HttpSession和ServletContext对象的引用

private HttpServletRequest request;

private HttpSession session;

private ServletContext application;

request = ServletActionContext.getRequest();

session = request.getSession();

application = session.getServletContext();

request.setAttribute("username", username);

session.setAttribute("username", username);

application.setAttribute("username", username);

1. 方法四: 采用DI或者IOC，实现ServletRequestAware接口设置HttpServletRequest对象的request

public void setServletRequest(HttpServletRequest request) {

this.request = request;

this.session = request.getSession();

this.application = session.getServletContext();

}

No.13 Java再加载类的时候，类中第一个执行的是构造方法，然后才是其他默认执行的方法，当不知道其他默认执行方法的执行顺序的时候可以通过System.out.println()打印字符串来了解默认方法的执行顺序。

No.14 Struts2中配置默认action，此时会以action的name是index的action为默认action，也就是说如果此时访问的action不存在就会自动跳转到默认action

<default-action-ref name=”index” />

No.15 Action总结

1. 实现一个Action的最常见的方式：从ActionSupport继承
2. DMI动态方法调用
3. 通配符配置\*\*{1}{2}…
4. 接收参数的方法（一般使用属性或者DomainModel来接收）
5. 简单参数验证addFieldError
   1. 一般不适用Struts2.0的标签
6. 访问Web元素
   1. Map类型
      1. IoC/DI
      2. 依赖Struts2
   2. 原始类型
      1. IoC/DI
      2. 以来Struts2
7. 包含文件的配置
8. 默认Action的处理

No.16 Struts2Action中的result的类型(要清楚跳转路径)

基本语法：<result type=”dispatcher”>跳转的结果</result>

1. **Dispatcher（默认type，采用的是forward的形式传递参数，不能跳转到Action）**
2. **Redirect（是一种新的请求，地址栏里面是页面地址，而不是Action地址也不能跳转到Action）**
3. Chain(跳转到action)
   1. 返回的action前面不要加/,
4. redirectAction（客户端跳转到）
5. freemaker
6. httpheader(传送头信息)
7. stream（下载有关）
8. velocity（和freemaker相似）
9. xslt（和XML修饰有关）
10. plaintext（显示页面源码）
11. tiles（把页面分成块，每一块都可以动态指定）

No.17 Struts2标签<package>中的extends属性是用来代表该package继承另外的package，和java类的继承一样的原理

No.18 Struts标签中的<global-result>是用来设置全局返回结果的

<global-result>

<result-name=”mainpage”>

/main.jsp

</result>

</global-result>

No.19 Struts2的动态结果集

1. 在Struts2中使用${变量}进行取action的值，从而动态的配置结果集

<package name=”user” namespace=”” extends=””>

<action name=”” class=””>

<result>${r}</result>//r是action的属性，其值是要跳转页面路径

</package>

No.20 带参数的结果集(一个值栈对应着一个forward，要理解forward的原理：直接与服务器交互，redirect，返回到页面之后再与服务器交流)

1. <s:property value=”a”>,此种方式是要来取值栈中的变量a的值
2. <s:property value=”#parameter.t”>,此种方式是用来取actioncontext中的变量的值

No.21 结果集总结

1. 常用的四种类型
   1. Dispather（默认）
   2. Redirect
   3. Chain
   4. redirectAction
2. 全局结果集
   1. Global-results | extends
3. 动态结果（了解）
   1. 在action中保存一个属性，存储具体的结果location
4. 产地参数
   1. 客户端跳转才需要传递
   2. S{}表达式（不是EL）

No.22 平时要读：struts-tags

No.23 Struts2中的OGNL（调用action的属性值）(学的不是很深，反正就是从action中取各种类型的数据，需要的时候就查就可以了)

1. Object-Graph Navigation Language的缩写，它是一种功能强大的表达式语言，通过它简单一致的表达式语法，可以存取对象的任意属性，调用对象的方法，遍历整个对象的结构图，实现字段类型转化等功能。它使用相同的表达式去存取对象的属性。
2. 基本语法和注意点
   1. User.xxx只有传的时候才会构造，想初始化domain model，可以自己new，也可以通过传参数值自动构建，但是此时必须要有空的构造方法
   2. 访问值栈（值栈是和action一一对应的，访问那个action就是那个值栈）中的变量，可以直接使用<s:property value=”值栈中变量的名称
   3. [”>](mailto:\”>访问值栈中的静态方法<s:property value=\”@com.lcb.struts2.ognl.S@s())
   4. [访问值栈中的静态方法<s:property value=”@com.lcb.struts2.ognl.S@s()](mailto:\”>访问值栈中的静态方法<s:property value=\”@com.lcb.struts2.ognl.S@s())”>,不过在struts2.1及以后要能够直接访问还需要在struts.xml中加入<constant name=”struts.ognl.allowStaticMethodAccess” value=”true”>

No.24 Struts2的标签知道就行了，需要的时候再查就可以了，当然有时间的时候可以再看一下