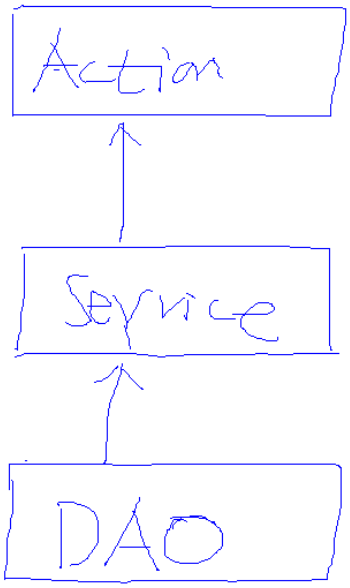
Struts2 Lesson 4

1. Struts2 应用的分层体系架构



2. Struts2 的模型驱动（Model Driven），之前所讲的称作属性驱动（Property Driven）

3. 属性驱动与模型驱动的比较

1） 属性驱动灵活，准确；模型驱动不灵活，因为很多时候，页面所提交过来的参数并不属于模型中的属性，也就是说页面所提交过来的参数与模型中的属性并不一致，这是很常见的情况。

2） 模型驱动更加符合面向对象的编程风格，使得我们获得的是对象而不是一个个离散的值。

小结：推荐使用属性驱动编写 Action。

4. 服务器端代码的单元测试有两种模式：

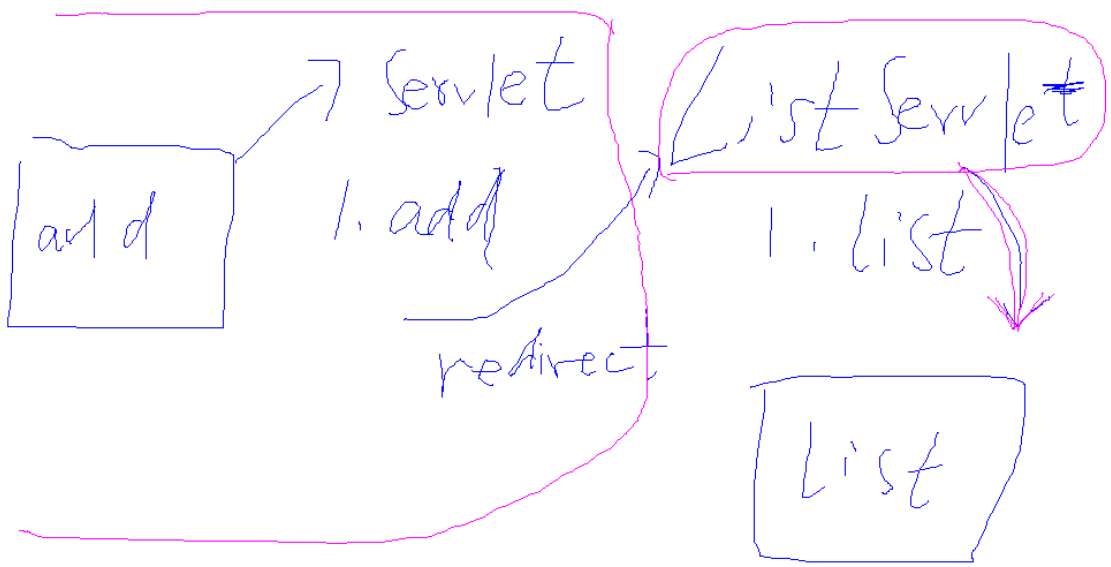
1） 容器内测试（Jetty）

2） Mock 测试（继承 HttpServletRequest、HttpSession、HttpServletResponse 等 Servlet API）。

5. Preparable 接口的作用是让 Action 完成一些初始化工作，这些初始化工作是放在 Preparable 接口的 prepare 方法中完成的，该方法会在 execute 方法执行之前得到调用。

6. 采取请求转发的方式完成表单内容的添加会造成内容的重复插入。

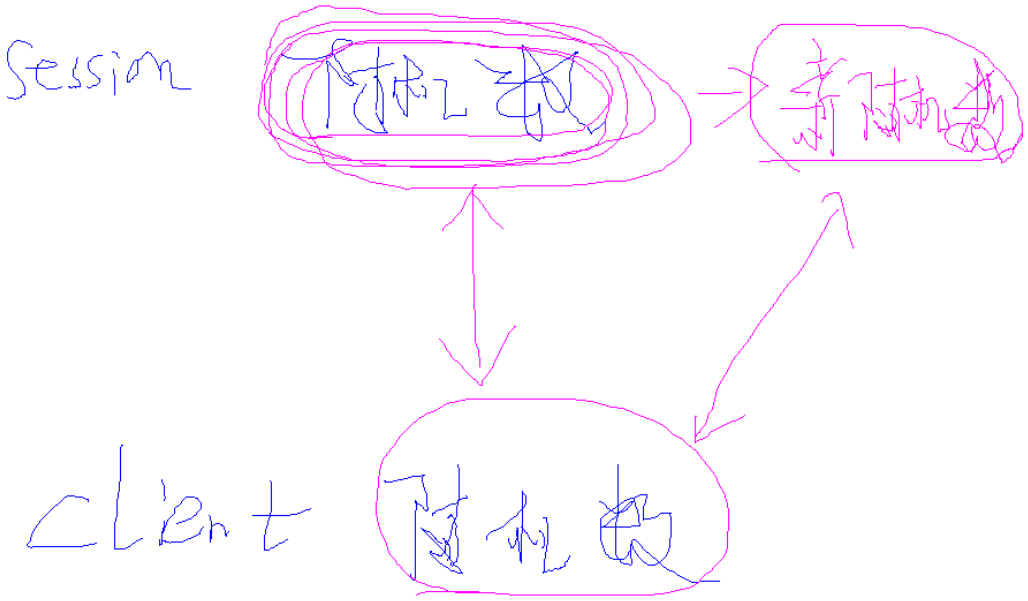
7. 采取重定向的方式实现数据的添加不会导致数据的重复插入。



8. 防止表单重复提交的两种方式

1） 通过重定向

2） 通过 Session Token（Session 令牌）：当客户端请求页面时，服务器会通过 token 标签生成一个随机数，并且将该随机数放置到 session 当中，然后将该随机数发向客户端；如果客户第一次提交，那么会将该随机数发往服务器端，服务器会接收到该随机数并且与 session 中所保存的随机数进行比较，这时两者的值是相同的，服务器认为是第一次提交，并且将更新服务器端的这个随机数值；如果此时再次重复提交，那么客户端发向服务器端的随机数还是之前的那个，而服务器端的随机数则已经发生了变化，两者不同，服务器就认为这是重复提交，进而转向 invalid.token 所指向的结果页面。



9. **拦截器（Interceptor）**：拦截器是 Struts2 的核心，Struts2 的众多功能都是通过拦截器来实现的。

10. 拦截器的配置

1）编写实现 Interceptor 接口的类。

2）在 struts.xml 文件中定义拦截器。

3）在 action 中使用拦截器

11. 一旦定义了自己的拦截器，将其配置到 action 上后，我们需要在 action 的最后加上默认的拦截器栈：defaultStack。