# Chapter4

GET请求是幂等的：

请求多次不会改变什么：

与doGet()方法是有区别的

POST请求不是幂等的

如果同时存在get和post请求：

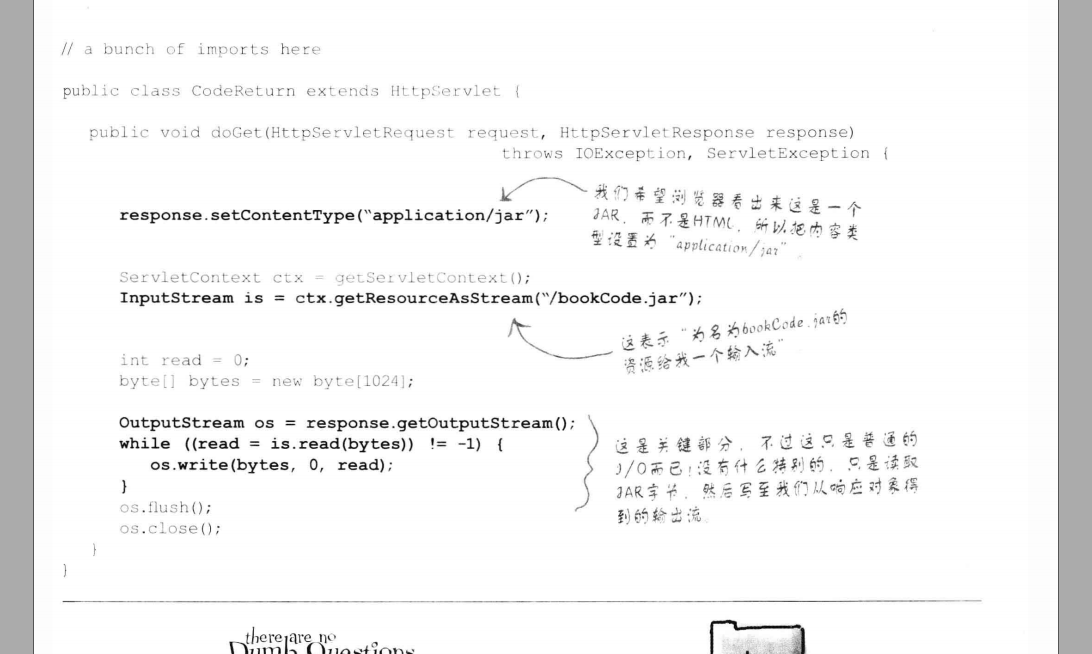
那么逻辑放在doGet()中调用doPost()

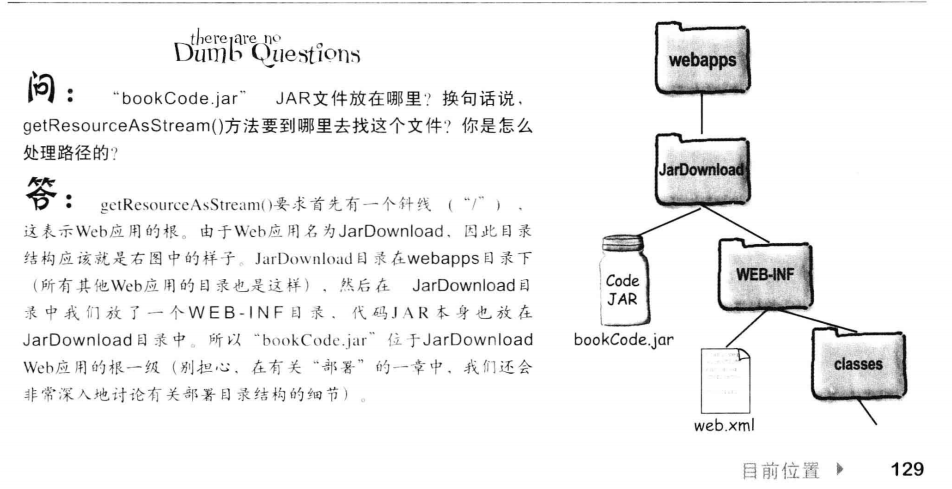
ServletConfig与ServletContext：

前者每个servlet都有一个初始化的

后者是整个Web应用的可以叫作AppContext

文件下载：

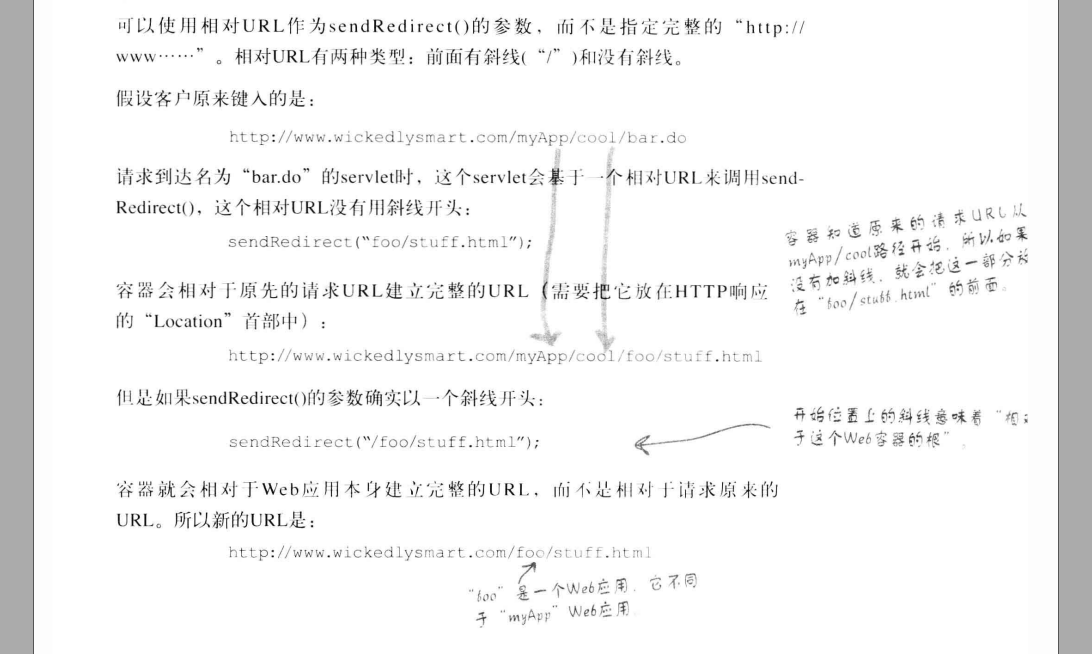




重定向与转发：

req.getRequestDispatcher("").forward(request, response);

resp.sendRedirect("");



问题：

既然是单例的Servlet，那么实例化多个会有影响吗：

可以多线程同时访问，但是同时只能实例化一个Servlet

可不可以先进行某些服务，再转发呢：

当然可以

# Chapter5

<context-param>

<param-name></param-name>

<param-value></param-value>

</context-param>

<servlet>

<servlet-name></servlet-name>

<servlet-class></servlet-class>

<init-param>

<param-name></param-name>

<param-value></param-value>

</init-param>

</servlet>

监听器listener：

实现ServletContextListener，

<listener>

<listener-class>

</listener-class>

</listener>





ServletContextAttributeListener：

知道Web应用上下文中是否增加，删除或替换了一个属性

HttpSessionListener：

知道有多少个用户，也就是跟踪活动的会话

ServletRequestListener：

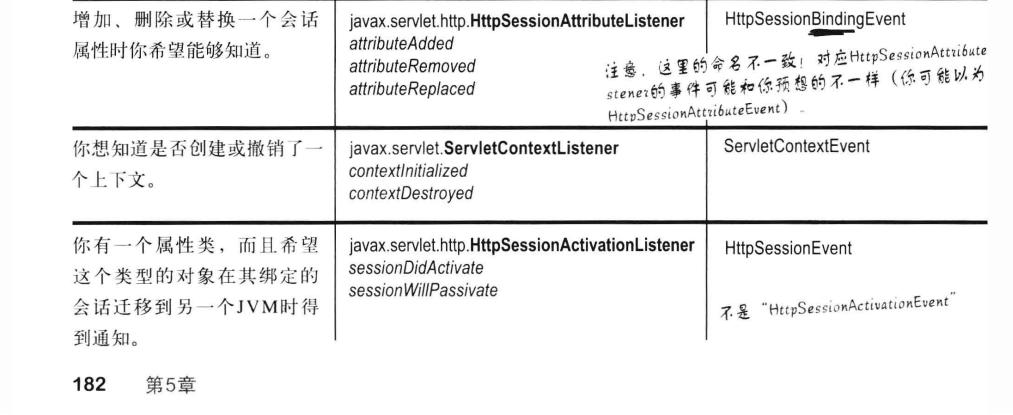
每次请求到来时，便于记录日志

ServletRequestAttributeListener：

增加，删除或替换一个请求属性是希望你能够知道

HttpSessionBingdingListener：

你有一个属性类，而且你希望这个类型的对象绑定到一个会话或从会话删除中得到通知





如果已经提交响应，就不能再进行转发了

# Chapter6



为了解决HTTPSession需要记住哪一个客户端，就需要向客户端输出一个特定的id

而这个id就可以通过cookie解决

禁用了cookie，无法通过正常方法获得id，都没有set-cookie响应首部



要想完成URL重写就必须动态的生成url，JSP中<c:URL>

设置session会话过期时间DD：

<session-config>

<session-timeout></session-timeout>

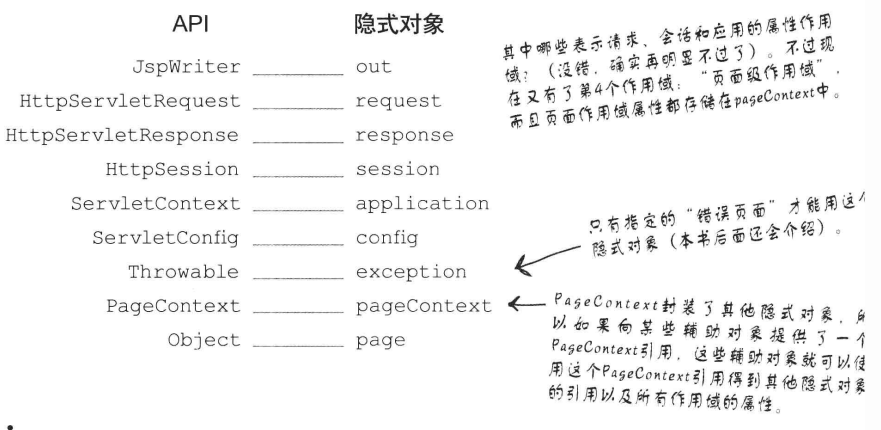
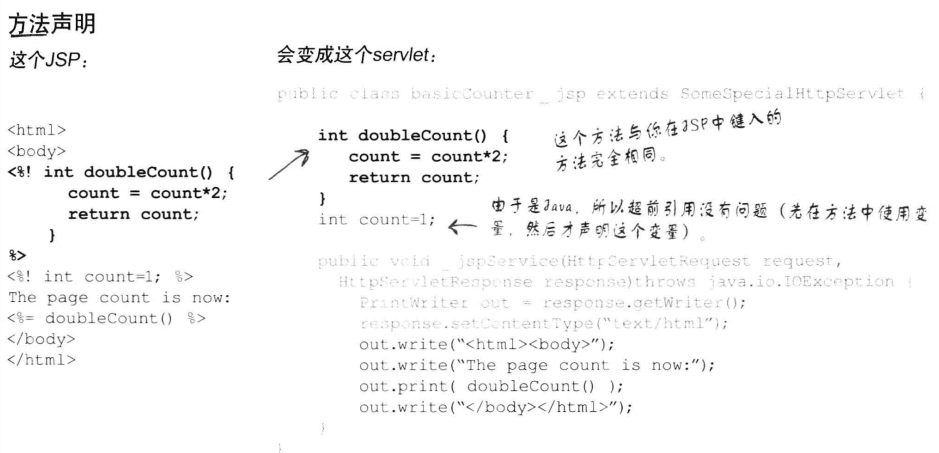
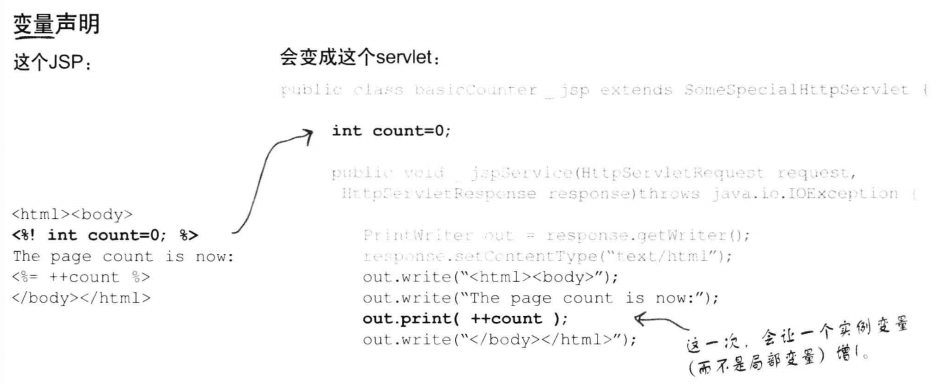
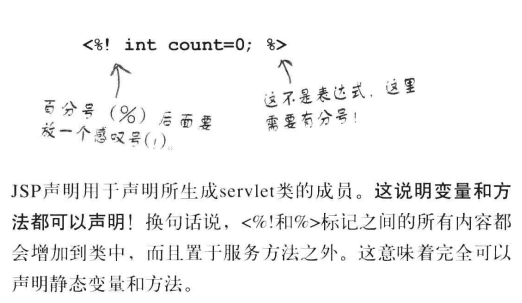
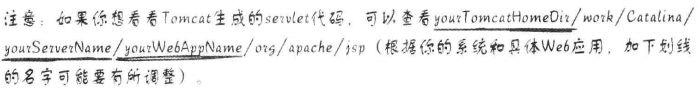
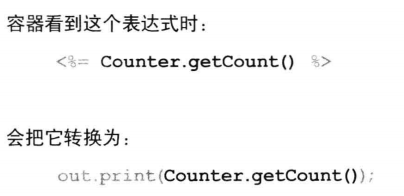
</session-config>

也可以在代码中调用HttpSession的方法



# Chapter7

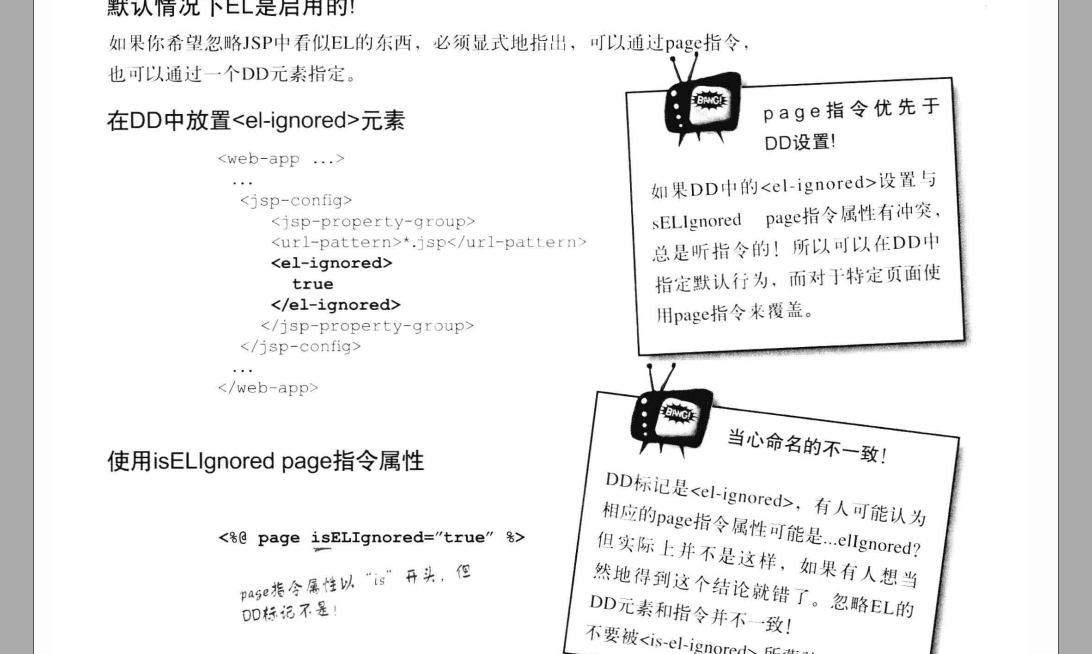
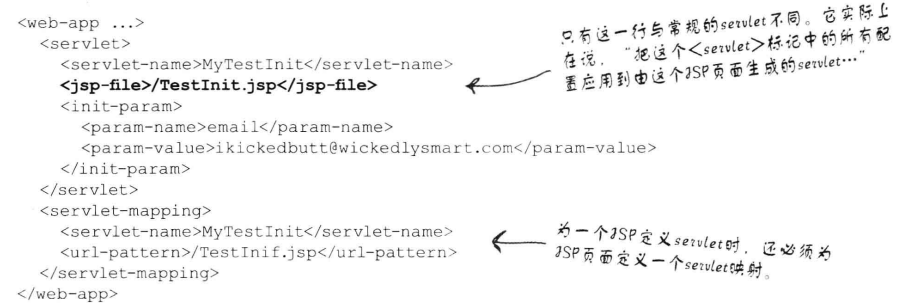
<%@ page import=*"java.io.\*，java.util"* %>



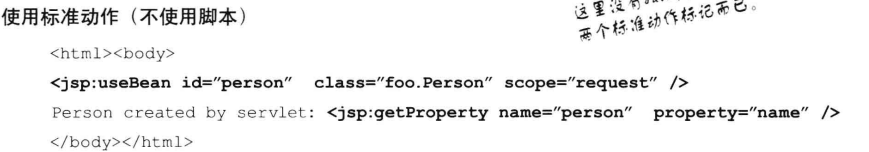
一般来说第一层请求Jsp都会比较慢：

后面会使用已经编译好的class文件

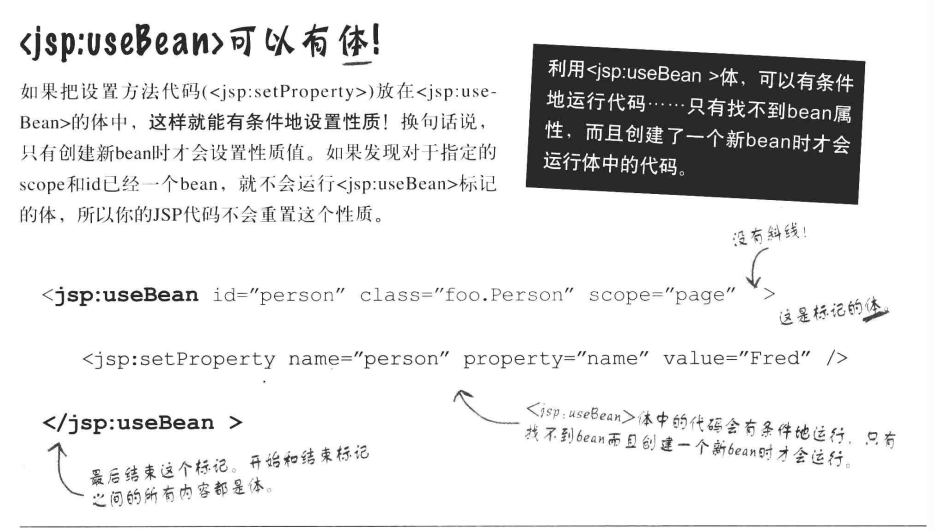
Jsp初始化的过程会先调用init，再启动一个新的线程来调用service方法

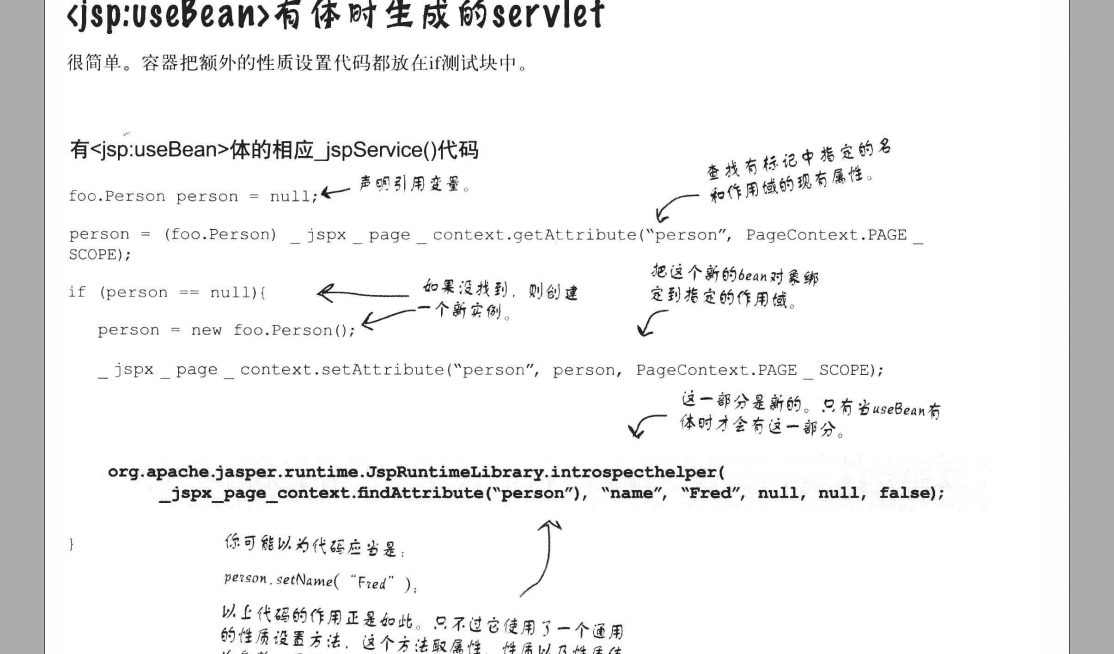


# Chapter8



如果不存在person这个对象就会创建一个



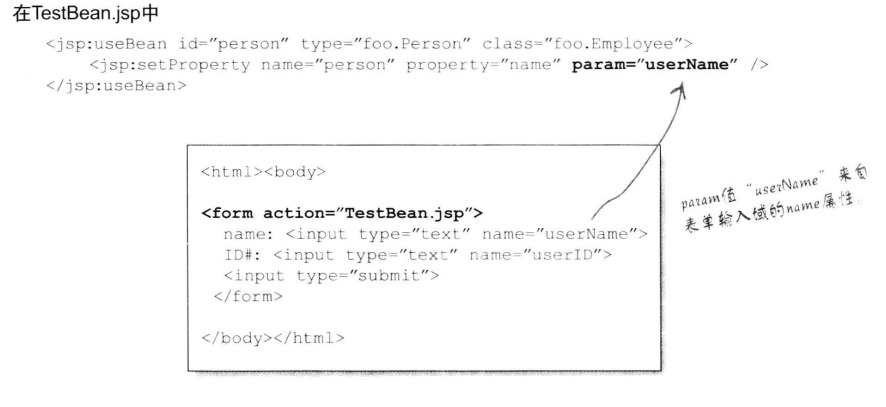
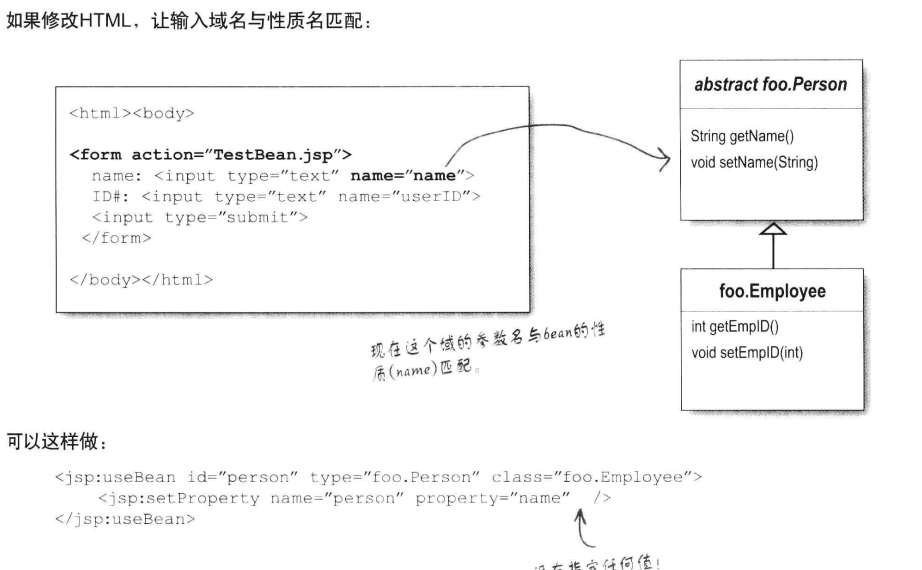


实现多态



设置参数Bean：

不用再写，request.getAttribute()了

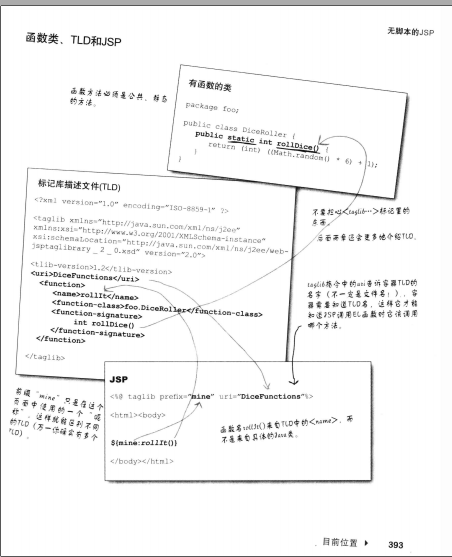
 

普通的.在EL中只能访问Bean对象的属性：

使用[]能够访问一个数组，list，Map

${musicList[0]} ${music[“1”]} 数组和List中的String索引会强制转为int

EL可以嵌套使用



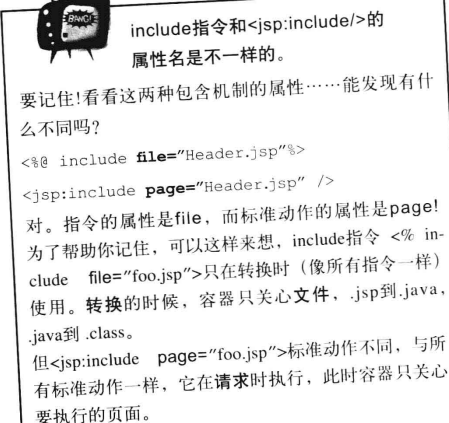
静态包含  <%@include file=“url”%>:

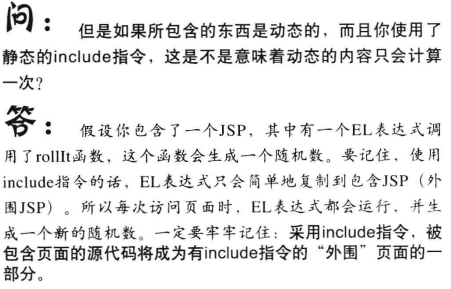
动态包含<jsp:include page="/header.jsp"></jsp:include>

**静态包含是先将页面代码包含进来，再运行。**

**因为静态包含是先将页面代码包含进来，再运行，所以页面之间相同标签属性配置不同的话会有冲突。至少两个jsp页面的<%@page contentType=“text/html；charset=utf-8”%>应该保持一致。同理，包含的jsp页面与被包含的jsp页面最好没有重复的html标签。否则会发生覆盖现象。**

**三个页面都能通过request接收到test.jsp页面传来的值，说明这三个或页面是共享一个request的。**





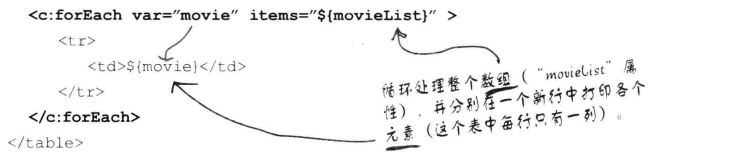
不要刷新之后再转发，不容会出现异常

<jsp:param/>:

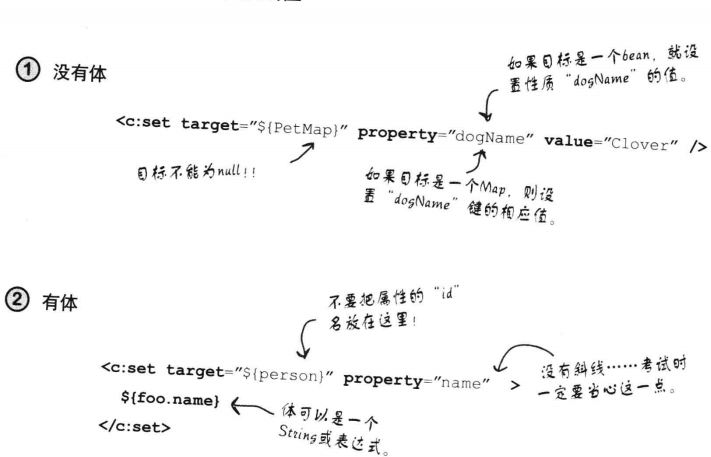
传递参数的标签

# Chapter9





If标签没有else



Target必须为一个对象，不能为bean或者map的属性名，String，id

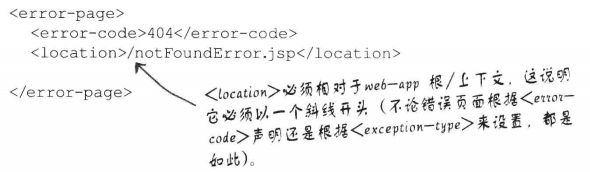
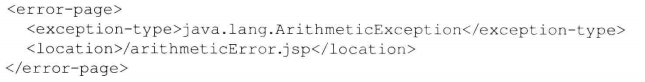
Set不能同时包含target和var：

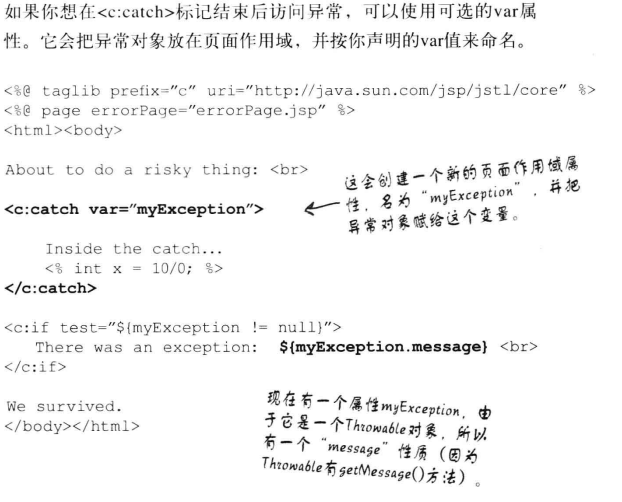
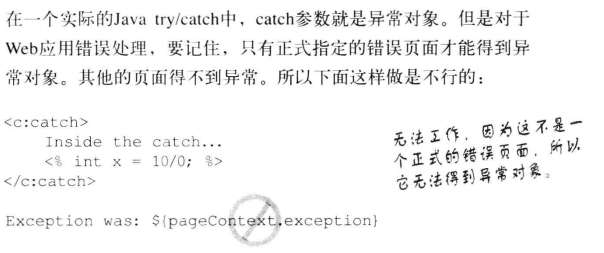
当var为null时，会删掉这个属性

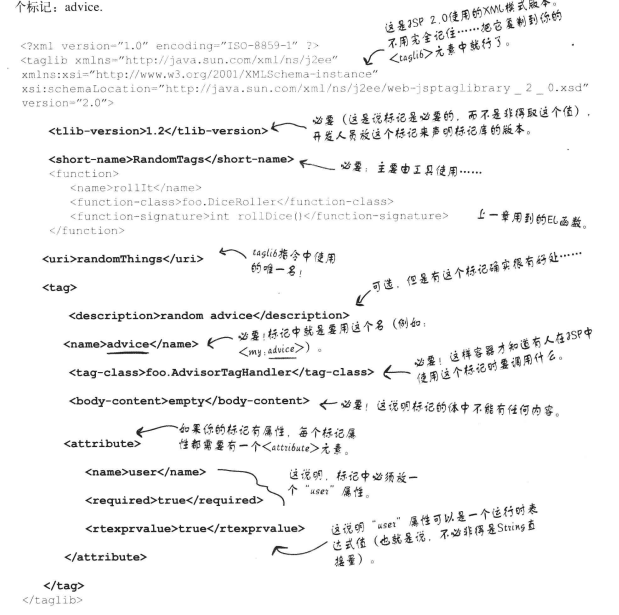
当target为null，会报错

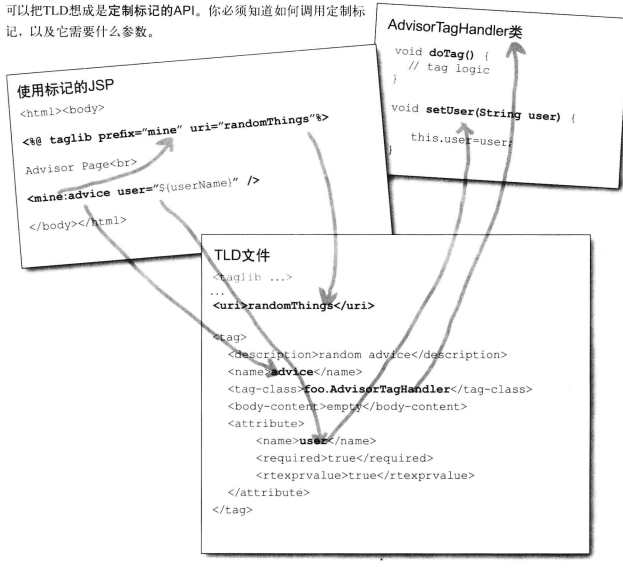
<c:url>可以完成url的重写，在jsp实现

<%@ page isErrorPage=”true” errorPage=”err.jsp” %>









# Chapter10

关于tag的，没有仔细看，自定义处理器

# Chapter11

无

# Chapter12

安全

# Chapter13

过滤器