javaMVC模式是将servlet和jsp结合起来的技术。servlet适合数据处理，而jsp适合显示。这个模式充分发挥了每项技术的优点。  
一、MVC 需求  
如果需要大量的数据处理servlet很合适，但是servlet对html的处理很冗长难以修改，这就是使用jsp的原因。将视图从复杂的数据和逻辑处理中解放出来。  
但是当jsp显示内容需要根据不同的需求生成差异较大的页面的时候我们就需要将jsp显示和servlet的逻辑处理结合起来。通常使用的模式是MVC模式。可以  
使每项技术发挥它最大的功效。初始请求由servlet来处理，调用逻辑和数据处理代码，并穿件bean来表示响应的结果（即模型）。然后servlet确定由哪个jsp页面显示比较合适并将请求转发到jsp（即视图）。servlet就是控制器。  
  
使用MVC的目的就是将逻辑操作和数据操作和显示代码分离。在相对较复杂的项目中使用框架是更好的更方便的选择。（Apache  struts2）。但是相对于使用框架来说  
从头开始自己创建MVC的实现（使用RequestDispatcher请求转发）更直观更灵活。  
  
二、使用RequestDispatcher来手动创建MVC。  
1.定义bean来表示数据  
2.使用servlet来处理请求  
3.填写bean  
4.将bean存储到请求，会话或者servletcontext中  
5.将请求转发给jsp  
    这里要说的是当一个请求转发的时候是将HttpServletRequest对象转发到jsp页面，而重定向（sendRedirect）是重新创建一个请求。（这里要注意的是如果jsp页面的显示只有在servlet上下文中才有意义的话可以讲页面放在WEB-INF目录中，因为这个目录用户是不能直接访问的，但是服务器可以访问）  
    如果是将bean存储在会话或者servletcontext 中时有的情况下使用重定向要比请求转发更有效  
6.从bean中读取数据并显示  
读取的时候jsp页面是使用的jsp:useBean标签及子标签来显示的，但是需要注意的是在这不能重新创建bean使用的标签应该这样:  
<jsp:useBean ...   type="package.class" />来代替创建新bean 的<jsp:useBean ... class="package.class" />  
还有一点要注意的是jsp页面不能用来设置bean只能读取即<jsp:getProperty />标签。还要注意读取bean是基于那种共享（Request，session，application）  
  
  
三、目的页面中相对URL的解释  
    使用RequestDispatcher来转发请求时最终URL是最初的servlet请求维持最初的请求不变，使用sendRedirect重定向的时候是重新创建请求并且URL是最终显示页面的URL  
还有一个需要注意的是如果jsp页面使用图像和导入js或者css的时候，RequestDispatcher的相对路径应该是相对于初始servlet的相对路径，而sendRedirect 的文件相对路径是相对于最终请求的页面的相对路径  
  
四、三种数据共享方式的对比  
当数据在每个请求不同的时候使用请求共享，  
当数据在会话的时候不变的情况下使用会话共享，  
如果在整个程序运行的状态下数据都不需要重新定义的话使用应用程序共享。  
  
五、在jsp中使用转发  
    在jsp中使用转发的方式可以是<jsp:forward page="relative URL">和scriptlet中，但是使用xml标签显然要比封装到RequestDispatcher中要方便许多，还有一个原因是效率要高。jsp:forward标签可以使用jsp:param作为子标签，为转到的页面提供额外的参数。在servlet中的后台实现就是RequestDispatcher类的forward方法。  
  
六、包含页面  
    使用jsp转发是利用servlet产生数据用jsp输出servlet不产生输出。还有一种方法是使用页面包含，可以讲servlet自身的输出及一个或多个jsp页面的输出组合到一起。RequestDispatcher的include方法就是jsp标签<jsp:include>的后台实现。如：  
import javax.servlet.\*;  
  
//doSomething  
RequestDispatcher rd=request.getRequestDispatcher("jsp路径");  
    rd.include(当前servlet的Request对象，当前servlet的response对象);  
//doSomething  
  
这端代码就是jsp:include的实现方法只包含页面的输出内容而不是如<%@ include%>命令那样讲整个文件的内容包含进去。  
  
七、EL表达式  
    EL(表达式语言)是为了计算和输出存储在标准位置的java对象的值。另一个特性是用jsp语法(而非Java语法)定义定制标签的能力。  
优点：1.精确访问存储对象。输出用setAttribute存储的对象。  
2.bean属性的简略记法。使用bean(name).attribute(name)访问bean属性。  
3.对集合元素的简单访问。使用${variable[index or key]}  
4.请求参数、cookie和其他请求数据的简单访问。  
5.实现一些简单的运算。  
6.选择性输出。(${test?option1:option2})  
7.自动类型转换。  
8.空值取代错误消息。  
    停用EL表达式  
1.停用整个WEB应用中的表达式语言。  
        修改web.xml文件的兼容版本  
2.停用多个jsp页面中的表达式语言。  
        修改web.xml文件，在其中添加如下标签：  
            <jsp-property-group>  
                <url-pattern>jsp Path</url-pattern>  
                <el-ignored>true or false</el-ignored>  
            </jsp-property-group>  
        true为停用false为启用  
3.停用个别页面中的表达式语言。  
        在需要停用EL表达式的页面加入如下语句：  
        <%@ page ignored="false"%>  
        true为启用false为停用  
4.停用表达式语言的个别语句。  
        将$符号用$替换，这个表达式就失效了  
    需要注意的是在jsp2.0的jsp页面上如果想要输出${}这个符号需要使用转意字符\${}表示这个和将$用$是一样的效果