# 回顾

1.jsp内置对象

|  |  |  |
| --- | --- | --- |
| Jsp内置对象名(直接使用) | Servlet对象(原型) | 说明 |
| request | HttpServletRequest | request.setCharacterEncoding()  request.setAttribute(“”,””)  request.getAttribute(“”)  request.getParameter(“”)  request.getRequestDispatcher("").forward(request, response); |
| response | HttpServletResponse | reponse.sendRedirect();  reponse.setCharacterEncoding()  reponse.setContentType(“”) |
| session | HttpSession | setAttribute();getAttribute() |
| out | JspWriter/PrintWriter | out.print() |
| application | ServletContext | setAttribute();getAttribute() |
| page | Object | this 代指当前对象 |
| pageContext | PageContext | setAttribute();getAttribute() |
| exception | Throwable | 异常 |
| Config | ServletConfig | 和servlet或网站的配置信息,主要初始操作 |

1. 属性范围

pageContext:当前页面

request:转发的下一个url地址,如果一直转发,数据就一直都在

session:会话在,数据就一直都在,不管页面跳转是转发还是重定向

application:整个应用程序运行期一直都在,和浏览器是否关闭无关,和容器有关,容器停止,则对象中的数据消失

1. 转发和重定向:
2. 地址栏的变化:转发->原来的地址;重定向->新地址,地址和显示的内容页面是一致的
3. 携带数据:需要携带,最好是转发,因为request.setAttribute(“”,””),可以将数据传递到一个页面,可以传递对象,如果使用则不好传递
4. 位置:转发只能在服务器内部进行,必须是当前的项目;重定向可以使用任何的url,可以是网站内部的跳转也可以是外部的跳转
5. 原理:转发是服务器内部的行为,重定向每次都是一个从客户端到服务器的新的请求
6. 效率:转发效率相对高

# 1 EL表达式:

EL是Expression Language的简称，即表达式语言

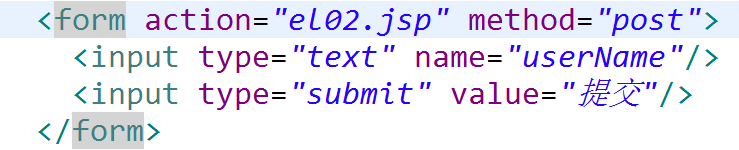
EL的主要作用:用来替代**jsp**中的表达式.<%=%>

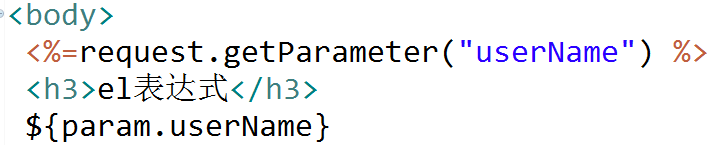
EL的基本语法: ${变量，表达式等}

<%=request.getParameter(“username”)%>

${param.username} param:是一个固定的写法 代表参数的意思

**必须是在jsp用的,不能在servlet**





## EL提供了11个内置对象：

* 2个内置对象为了方便输出请求参数：

param/paramValues；

* 4个内置对象为了方便输出各个范围的属性：

**pageScope/ requestScope /sessionScope /applicationScope**

**Scope->范围**

* 2个与请求头有关的内置对象：

header/headerValues

* 2个其他内置对象：

cookie/initParam

* 1个特殊的内置对象

**pageContext .> ${pageContext.request.contextPath}**

## 特殊对象说明pageContext:

内置对象pageContext：EL中的pageContext对象可以调用PageContext类中所有符合规范的getXxx方法，如PageContext类中有如下方法：

public abstract ServletRequest getRequest(),

${pageContext.request}

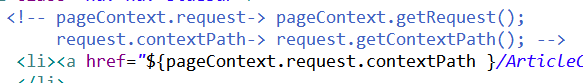
<%=request.getcontextPath() %>替换: ${pageContext.request.contextPath}



替换成::



其实都是方法的调用,封装了一下,变成属性的写法



## 四种属性范围:

* pageScope：页面范围，即pageContext范围
* requestScope：请求范围
* sessionScope：会话范围
* applicationScope：上下文范围

注意:检索顺序：当不指定范围时，例如，${user.username}，将自动从pageScope开始查找，直到applicationScope，如果没查到，则什么也不显示;(优先级)

${范围.key}

${requestScope.username} / ${username}

不要把Score丢了,, ${request.username}错误

## EL中的运算符:

* EL中的运算符包括：
* 算术运算符 (省略) :
* 比较运算符
* 逻辑运算符
* 其他运算符

### EL比较运算符：

* 有6种比较运算符，可以对值进行比较，返回值为true或fasle。
* = = 或eq表示等于(equals)
* != 或ne/neq表示不等于 (not equals)
* < 或lt表示小于

> 或gt表示大于

* <= 或le表示小于等于
* >= 或ge表示大于等于

### EL逻辑运算符

* &&或and表示交集，两个值都是true才返回true。
* ||或or表示并集，两个值只要有一个是true，即返回true。
* !或not表示非

### 集合运算符:

* 使用[]指定索引的方式可以获取数组或List中的元素；
* ${requestScope.goodsList[0]**.**goodsName}

### 其他运算符：

* 除了算术、比较、逻辑运算符外，还有三种其他运算符。
* empty运算符：判断值是否为null，如果是null，返回true，否则返回false。
* 关系运算符：${A?B:C}如果A为true，则执行B，如果A为false，则执行C。
* ()运算符：通过()可改变优先级
* ${empty uname}

## 小结:

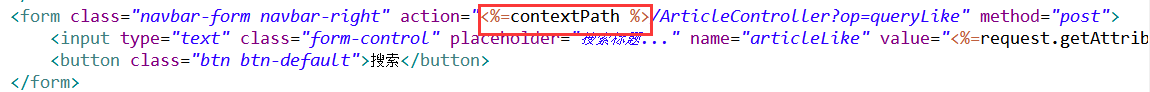
1. el表达式代替JSP的<%=%>

1. 语法:${}
2. ${el内置对象.key} 内置对象:param/pageScope/ requestScope /sessionScope /applicationScope
3. ${param.参数名} 相当于:<%=request.getParameter(“参数名”)%>:获取请求参数
4. ${requestScope.属性名} 相当于<%=request.getAttribute(“参数名”)%>:获取某个属性范围内的属性值

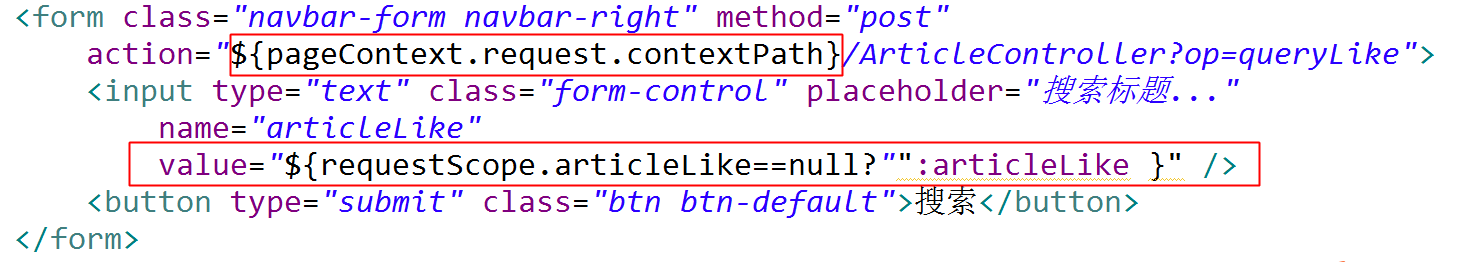
${username} 依次去属性范围内查找username的key所对应的值

6.上下文路径:${pageContext.request.contextPath} 相当于之前的<%=request.getContextPath()%>

## 任务:将之前的cms案例中的表达式全部改成用el表达式完成



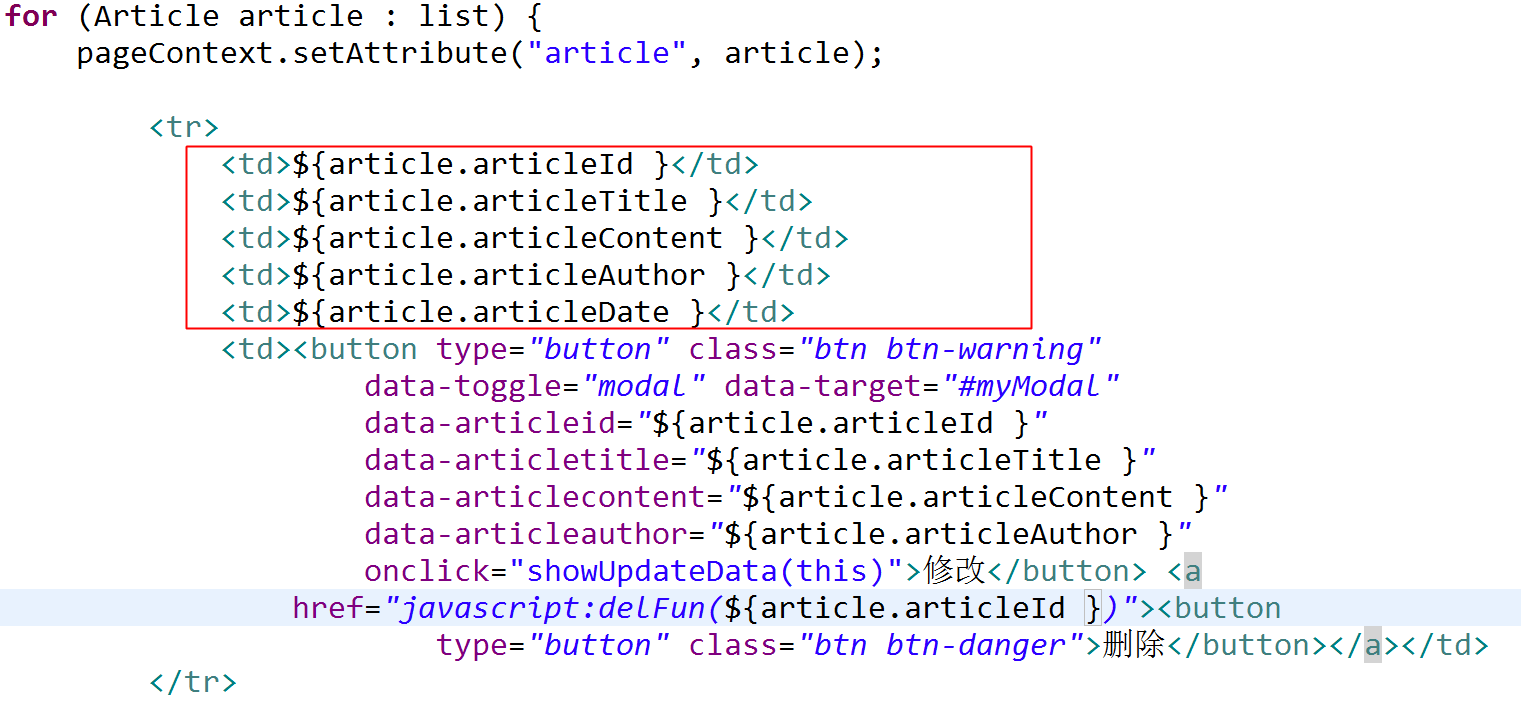
全部改成el表达式



还有这块:



或者:



回顾:

1. EL表达式(替换jsp的表达式<%=%>): ${}
2. ${内置对象.key}

${param.参数名}:<%=request.getParameter(“参数名”)%>:获取请求参数

el内置对象:pageScope/requestScope/sessionScope/applicationScope

${xxxScope.属性名} ${属性名}:获取某个属性范围内的属性值

<%=request.getAttribute(“属性名”)%>

1. 获取上下文路径:${pageContext.request.contextPath}
2. 运算符: ${1==1} ${sessionScope.uname==null} ${1 eq 1}

# 2.JSTL:jsp的标准标签库

分工:jsp只负责页面内容的显示，用户的交互操作;而现在的jsp代码中存在大量小脚本(其他jsp的内置对象):

我们希望这内容可以拿掉，用Jstl替代:

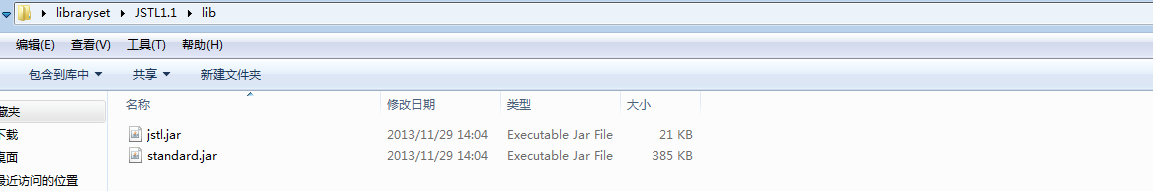
作用:分工明确，以前<% %>中的java代码都消失了，通过jstl来执行;

代码可读性更好；可维护性;

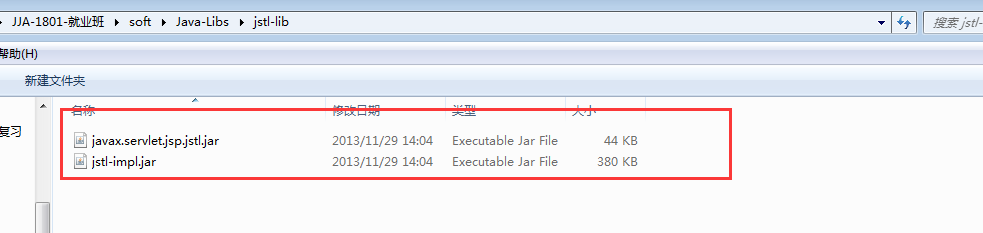
## 基本使用:

1. Eclipse中使用jstl.,需要导入外部的jar包;不同的jstl版本，jar包是不同：

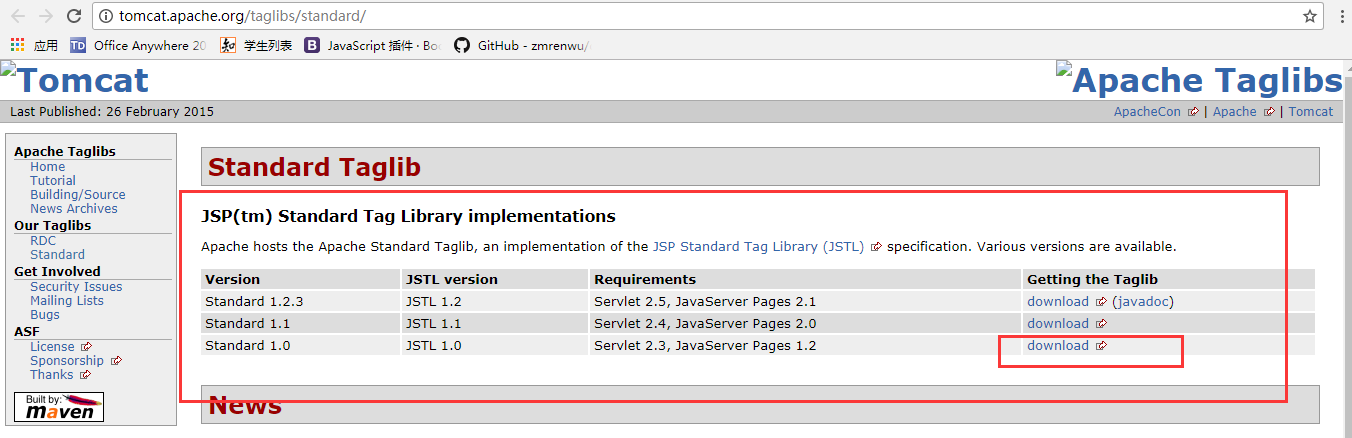
Jstl1.1： 是这个样子的



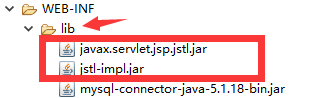
1.2.1 :是这个样子的



大家也可以去官方下载对应的jar包.



1. 添加到构建路径中来：

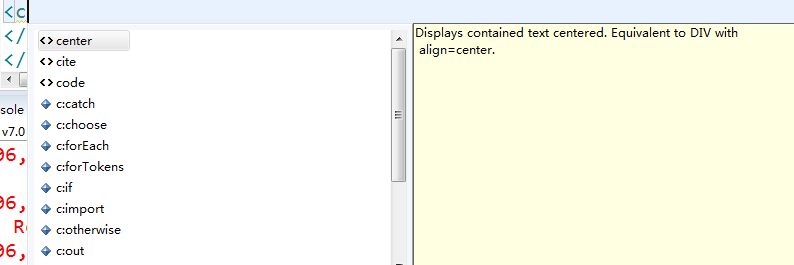


1. 在要使用jstl的jsp页面中导入标签库:使用指令

<%@ taglib prefix=”前缀**c**” uri=”http://...../jsp/**core**” %>

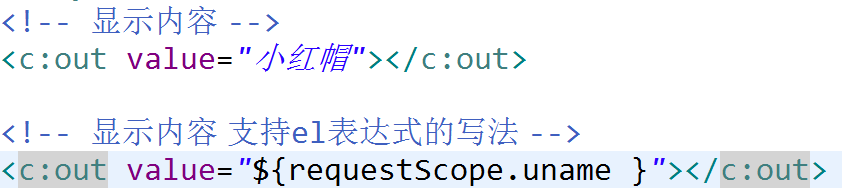
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>



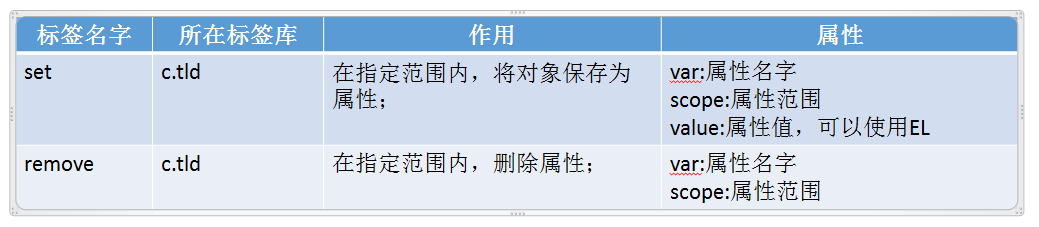


## Out:

显示使用 out也可以直接使用 ${….}

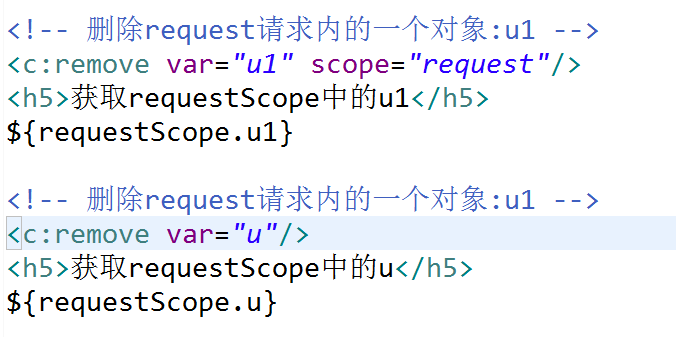


## 属性操作有关:



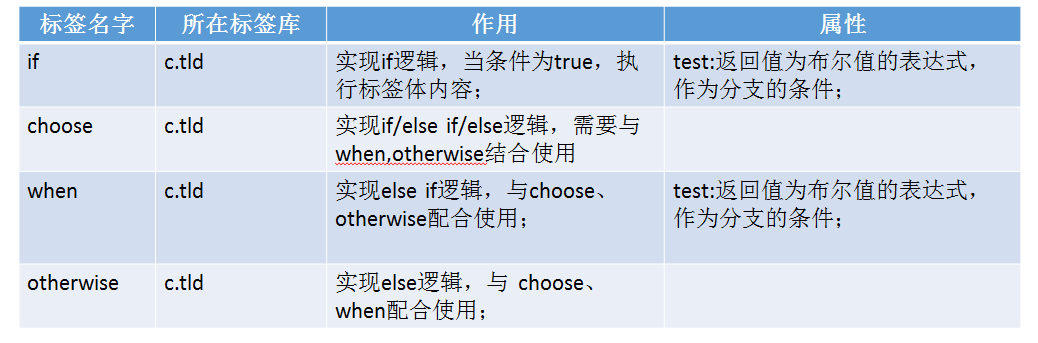


remove:

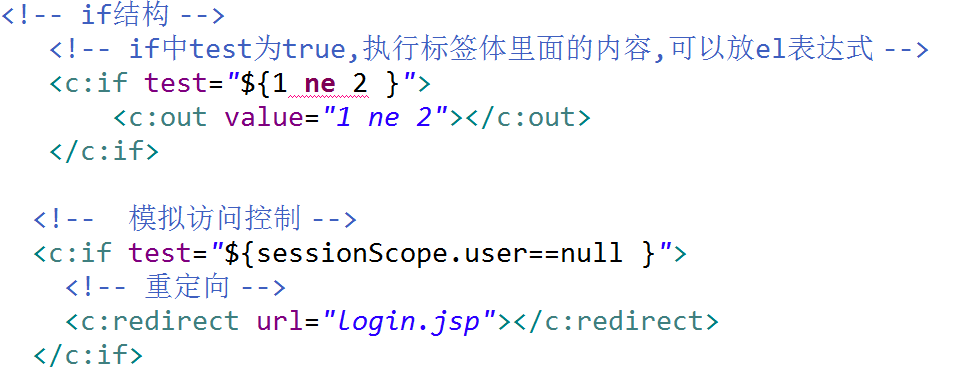


课堂练习：熟悉操作过程，以及c:set c:remove

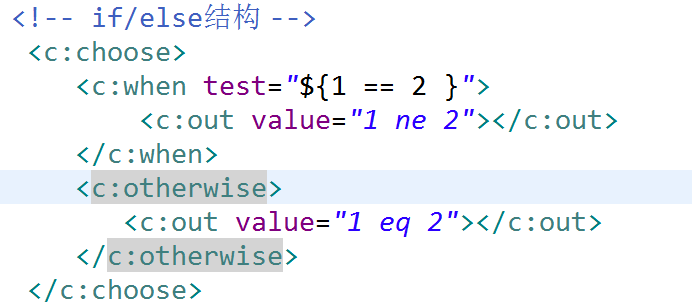
## 条件分支有关:



if结构:

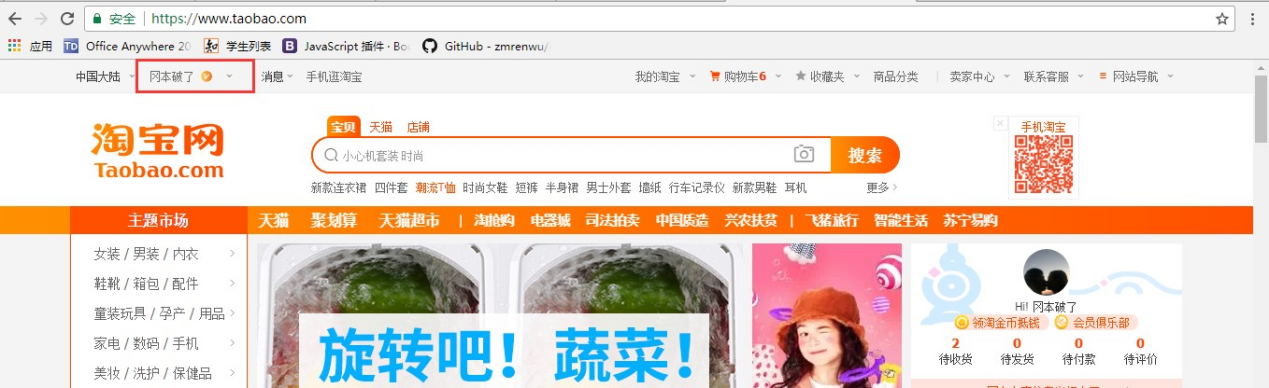


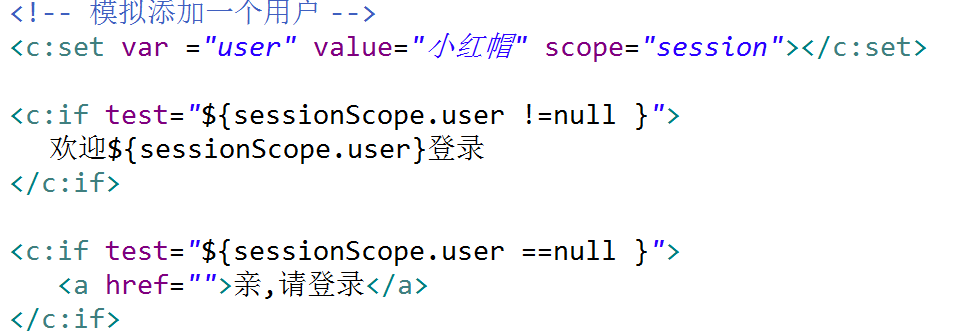
if/else结构:



## Jstl+el课堂练习:

判断session范围内是否存在某个用户,如果存在,将用户名显示出来,如果不存在,显示未登陆;

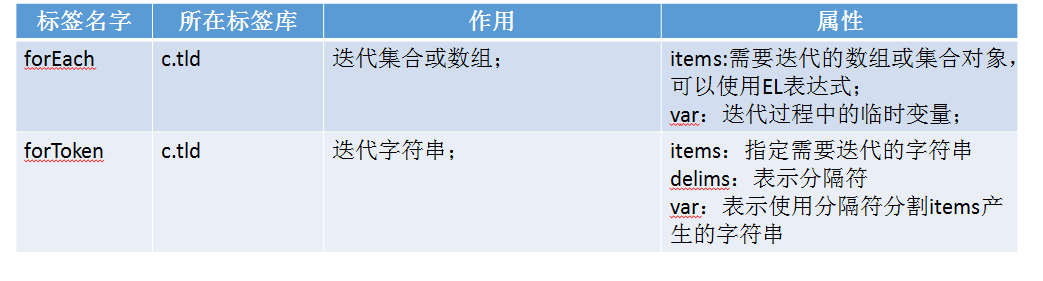




或者用c:choose:



## 迭代有关:



把showArticle.jsp改成jstl+el

第一步 先导库: <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>



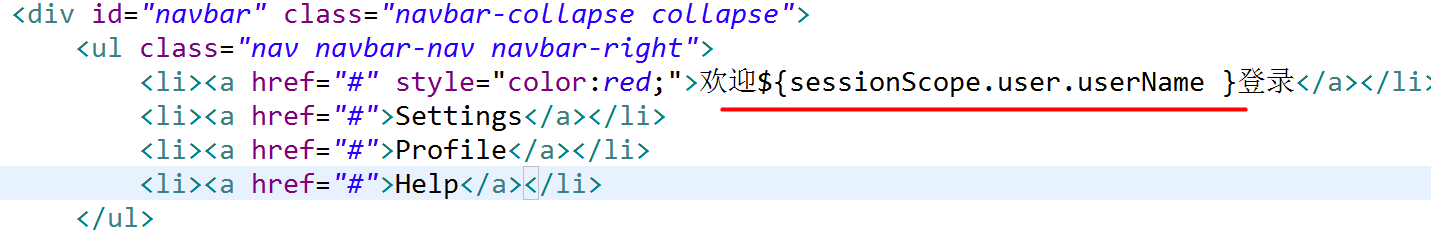
## 课堂练习:

1.将之前jsp页面中小脚本替换成jstl标签库,并且加入访问控制

有登录信息就在指定的位置直接显示用户名



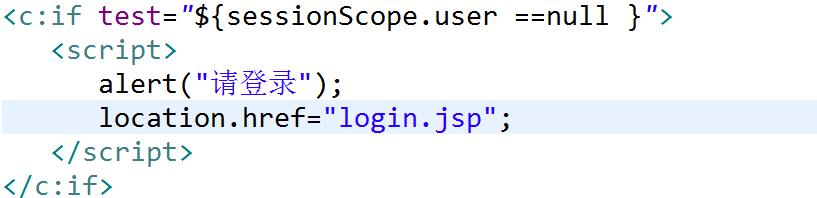
showUsers.jsp:



加入访问控制:

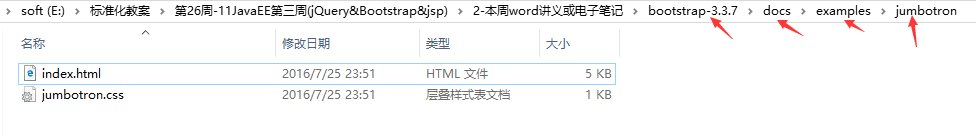


addUser.jsp:

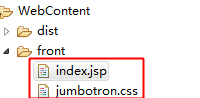


# 3.cms1.3加入前台首页

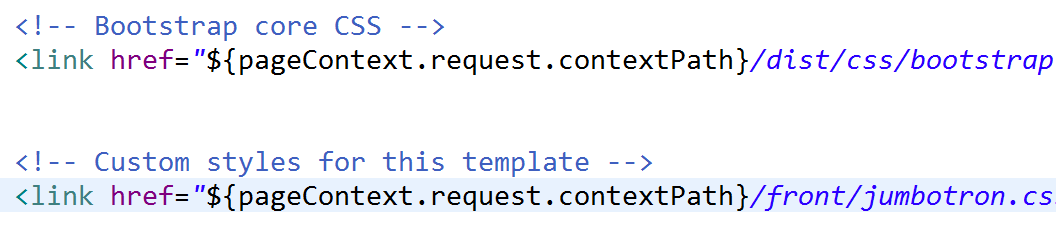
1.找模板:



1. 添加一个front目录,专门放前台的页面

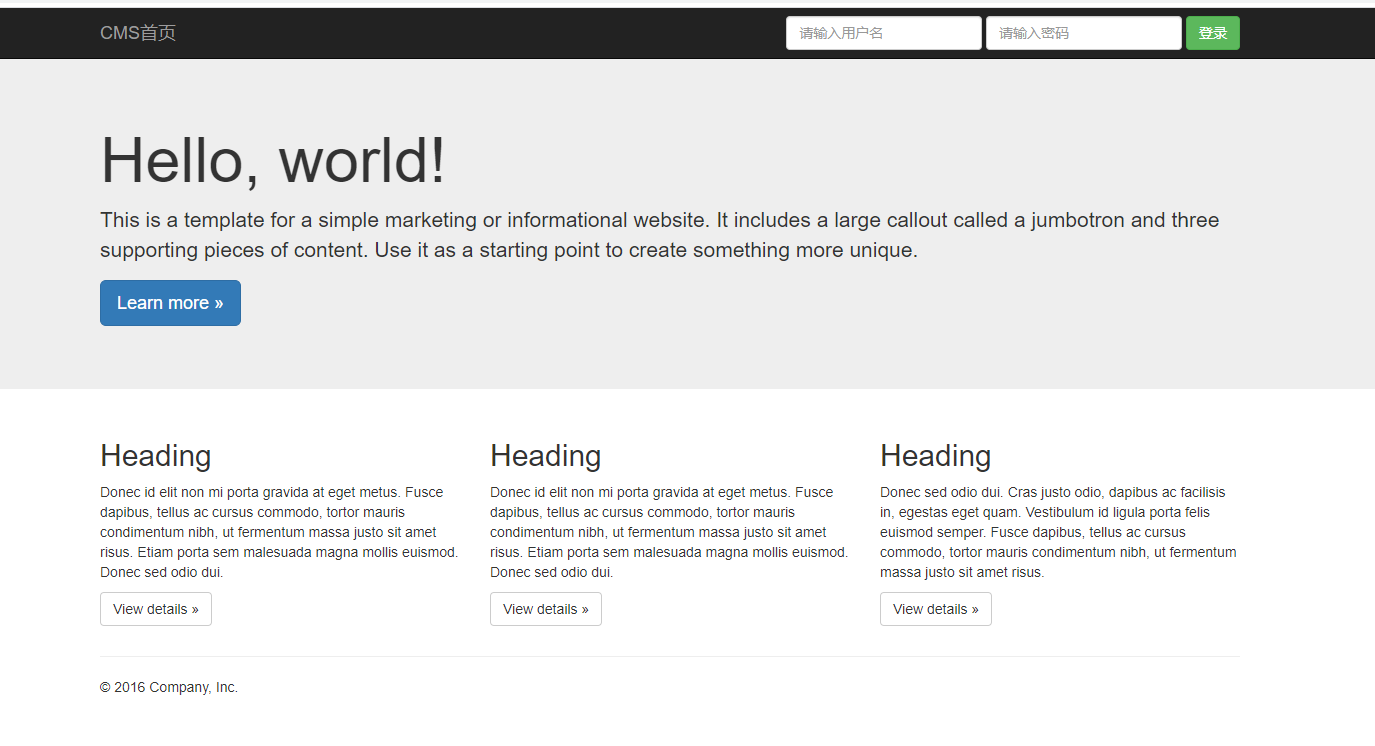


1. 样式:



4.把英文修改为中文,并且表单指定提交路径与控件name

修改完部分后:



1. 将文章的数据显示成数据库的数据,并不需要全部文章,只要前三(把最新文章显示出来)

怎么拿到最新的->根据发布时间进行排序,获取前3条数据

ArticleController:



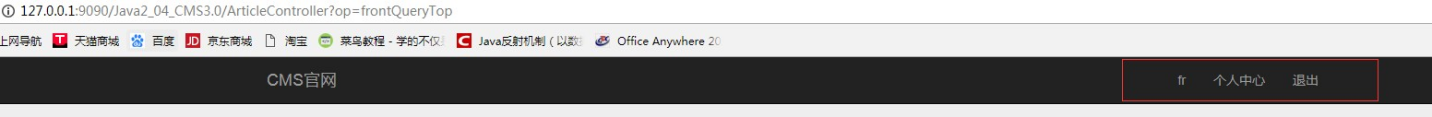
index.jsp: Heading的div只留一个即可



此时就可以直接访问index.jsp页面,而不用访问控制器

# 4.登录功能

登录成功之后

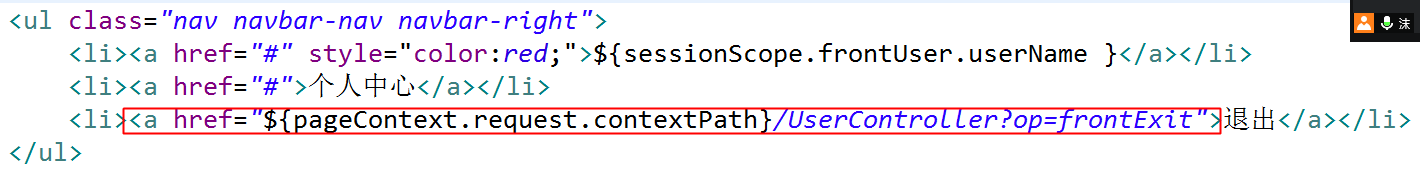




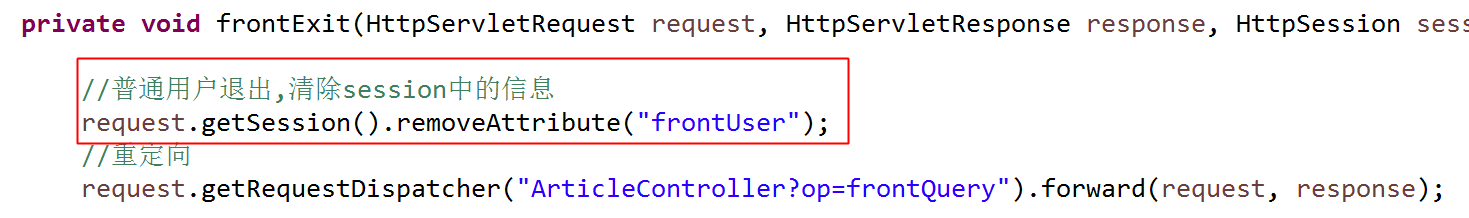


# 5.用户注销 注销用户后:

IMG_256

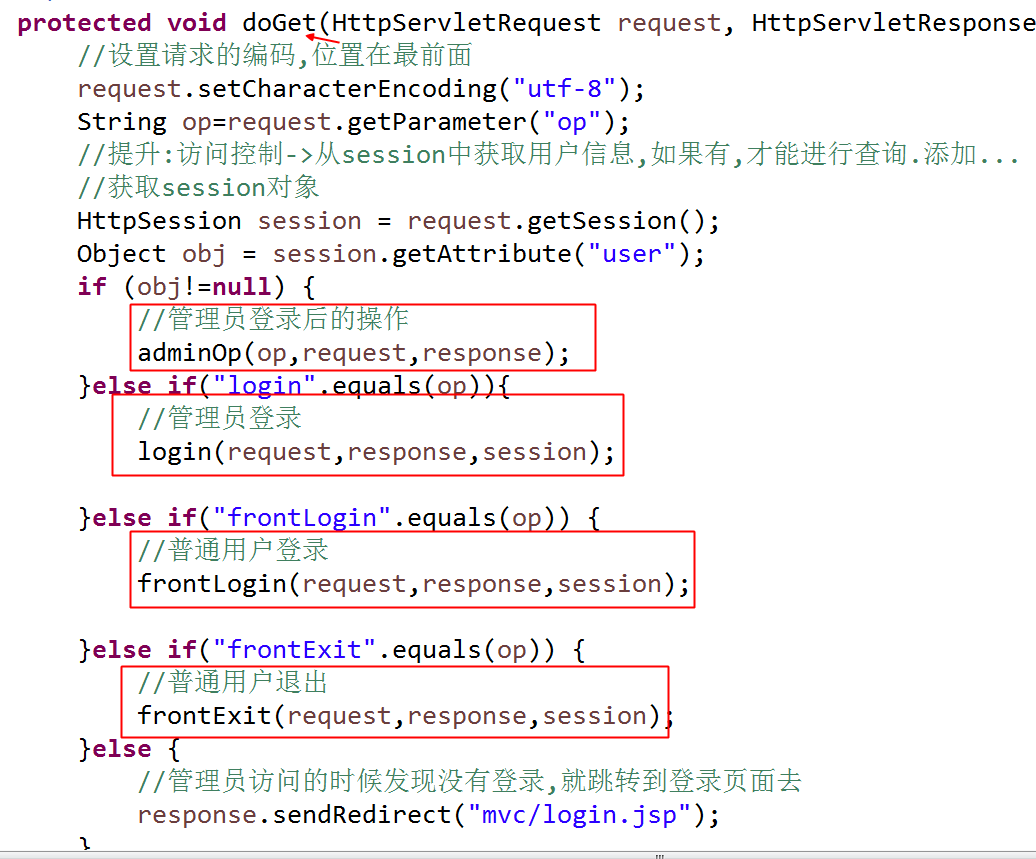


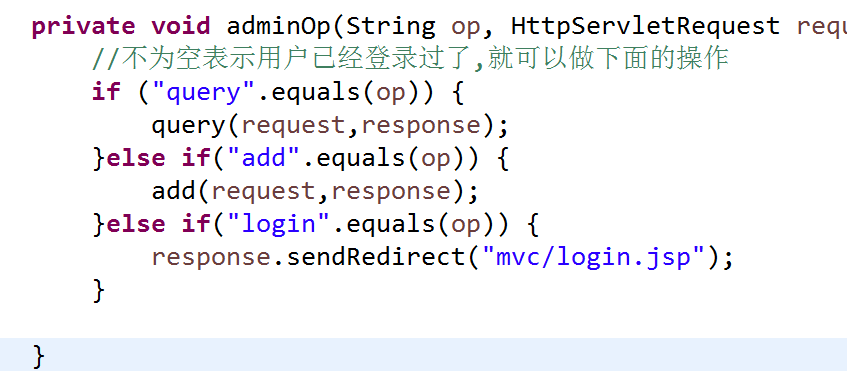
控制器:



# 6.优化代码:

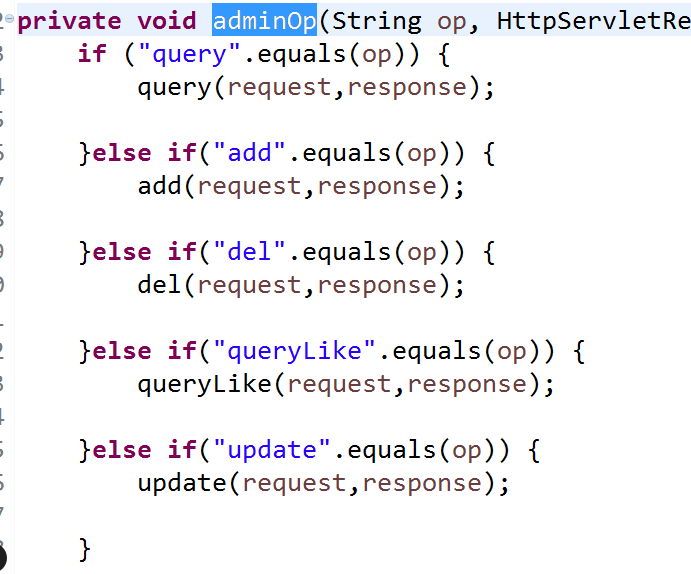
UserController:



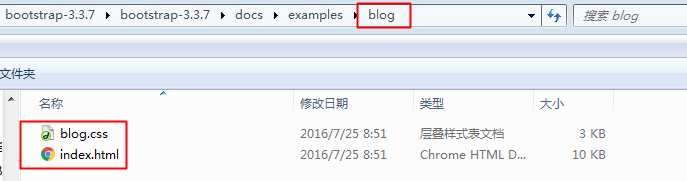


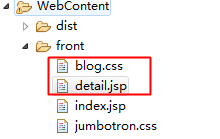
ArticleController:



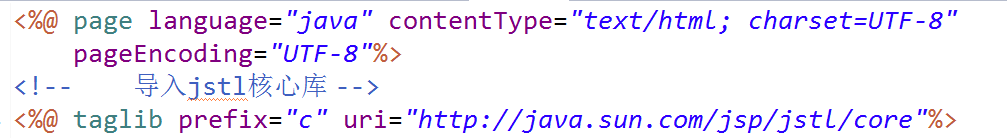


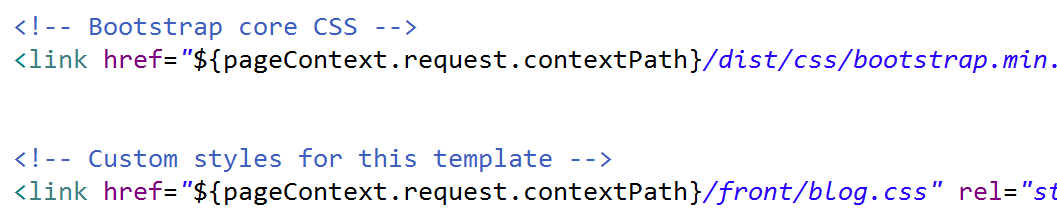
# 7.首页文章进入详情页





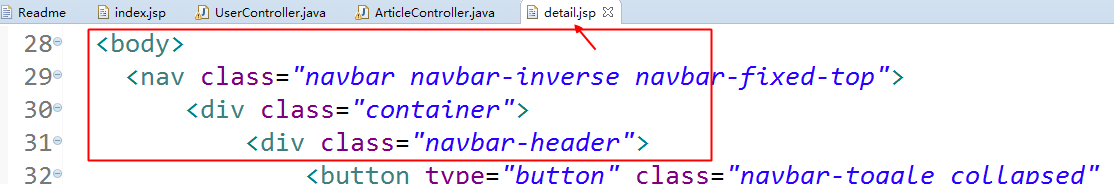
修改模板如下:





换头:

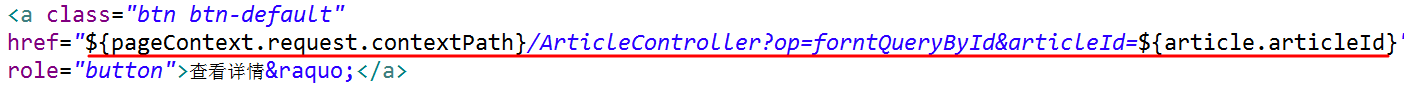
把首页index.jsp的整个nav标签复制到detail.jsp的最上面



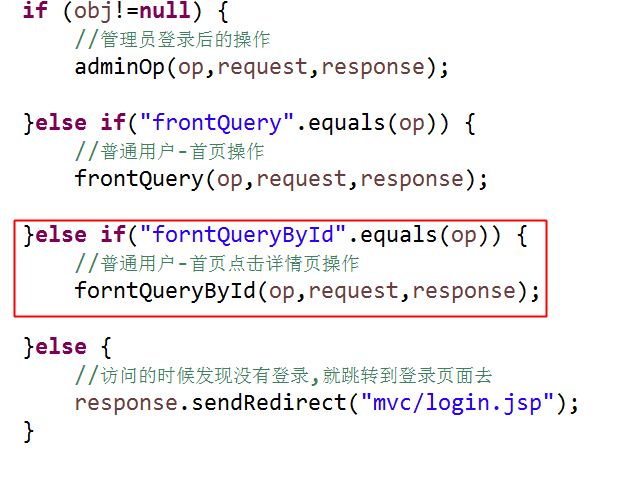


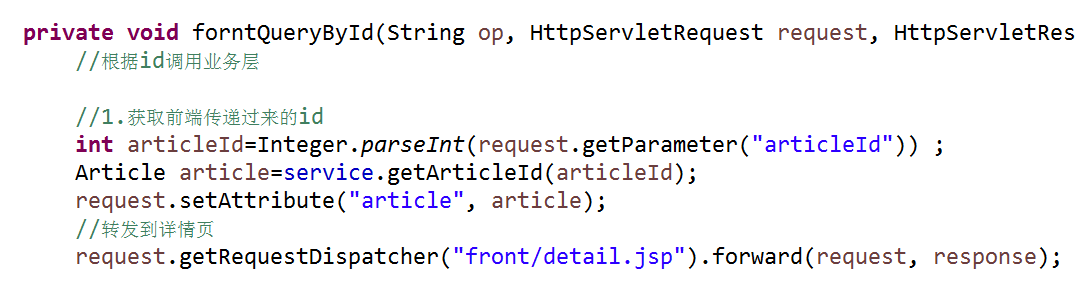
此时:首页点击查看详情进入详情页:

当点击首页的查看详情页按钮需要把这篇文章的id,传递到控制器

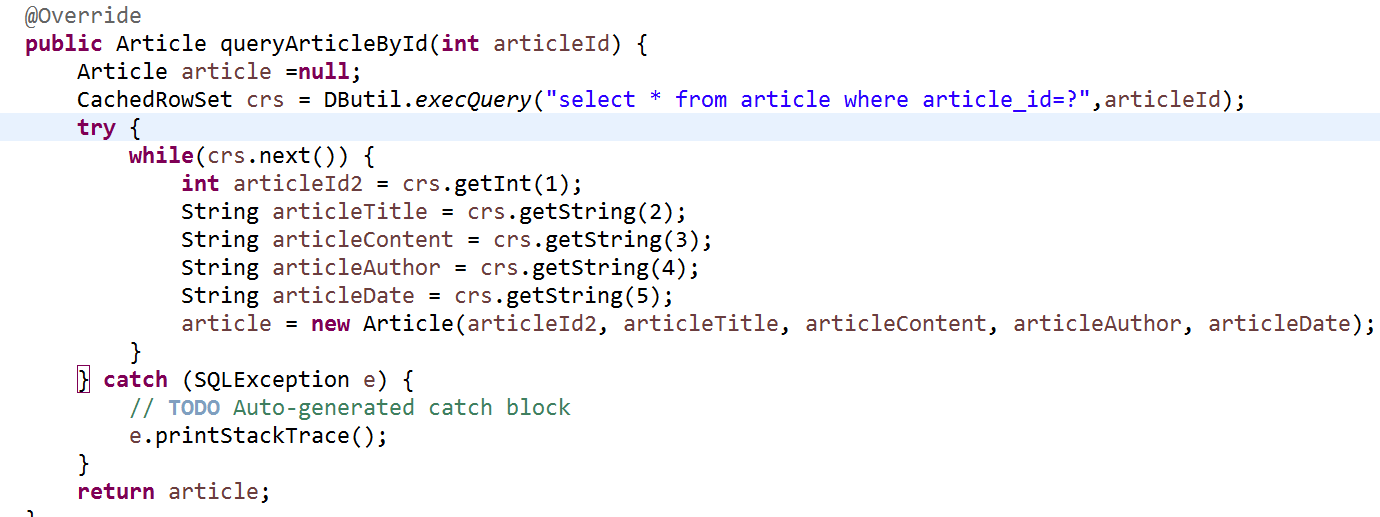


控制器:

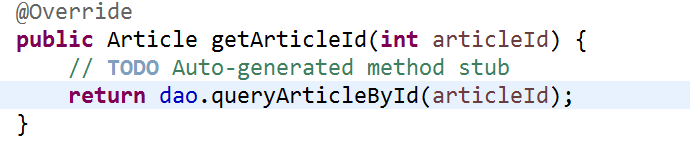




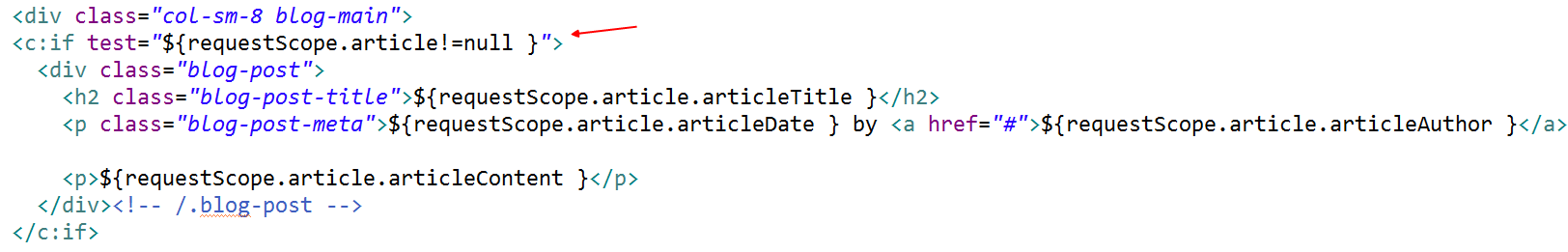
dao:



service:



detail.jsp显示数据:



加入访问控制:



单行文本溢出显示省略号..:在index.jsp中显示的内容加入样式:

<p style="width:*300px*;white-space: *nowrap*;text-overflow: *ellipsis*;overflow:*hidden*;">${article.articleContent }</p>



## 总结:

1. el表达式(代替jsp中的表达式<%= %>) ${} 属性范围:xxScope …

${ xxScope .username} :依据范围查找username的key所对应的的值

${username}:依次去属性范围内查找username的key所对应的的值

el运算符



1. jstl:标签库:

使用步骤:

1.导入2个库(jar包)

2.使用指令导入core的核心库(设置标签前缀c)

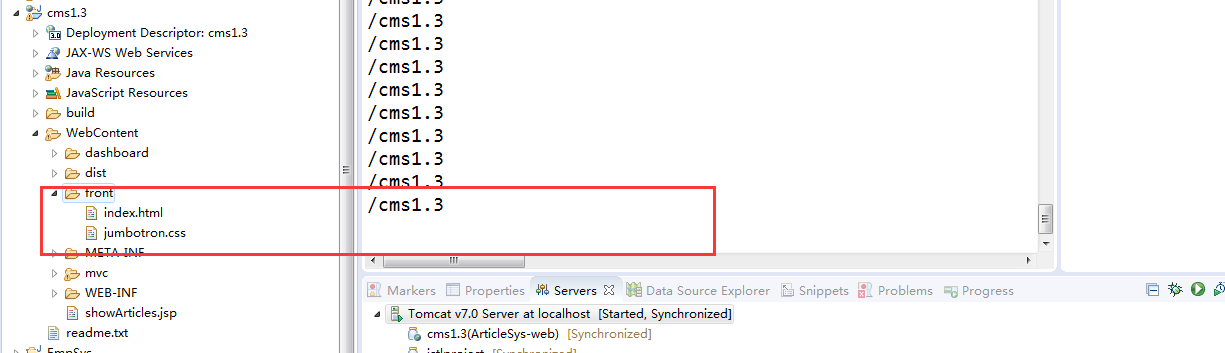
2.条件分支:访问控制

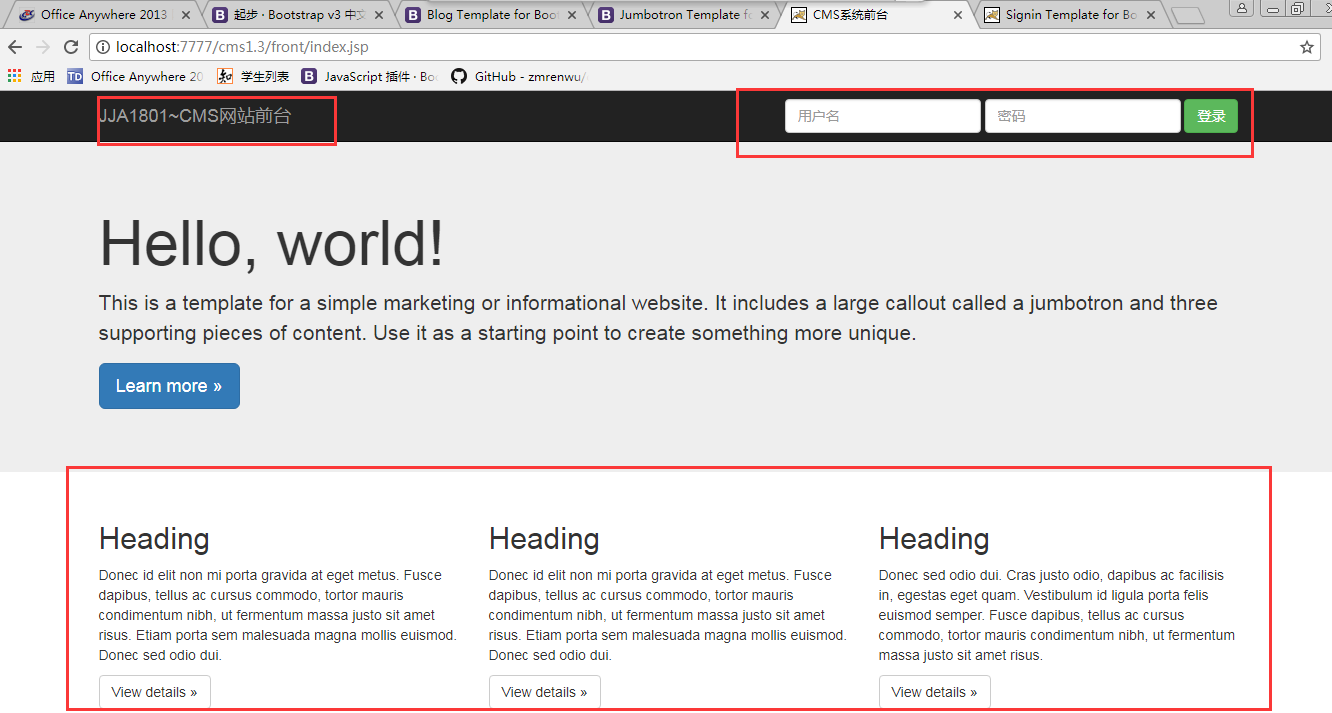
3.迭代相关:集合el表达式实现数据的显示

# 作业

完善cms1.3 ,jstl+el,实现网站的前台浏览;

而普通用户(游客)来到网站,可以浏览一些文章.网站的前台界面;





接下来将文章更换成数据库表中的记录;

