【济南中心】JavaEE就业班同步笔记第一阶段：JavaWeb之JSP&EL&...

小鲁哥哥 • 2017-2-12 15:16:04

【济南中心】JavaEE就业班同步笔记第一阶段：JavaWeb之核心技术--JSP&EL&JSTL

1 案例一：在JSP的页面中显示商品的信息.

1.1 需求:

数据库中存放了很多商品信息,现在将商品的信息全部显示到页面.

1.2 分析:

1.2.1 技术分析:

【JSP的概述】

什么是JSP：

* Java Server Pages（Java服务器端的页面）

为什么要学习JSP:

* SUN公司推出的Servlet自身有缺陷,没有办法与ASP,PHP进行竞争.推出了动态网页开发技术JSP.

使用JSP:

* JSP = HTML + Java代码 + JSP自身的东西.

执行JSP的过程:

* JSP翻译成Servlet,编译这个Servlet的类,生成class文件.得到执行.

【JSP的脚本】

<%! %>:翻译成Servlet中的成员内容. 定义变量，方法，类. – 存在线程安全问题，不建议.

义类,变量

【JSP的注释】-了解

HTML的注释 :<!-- 注释 --> ---jsp、翻译成servlet、html都存在（无法注释Java代码）

Java代码的注释 :// 单行注释 /*多行注释*/ /** 文档注释 */ ---jsp、class存在

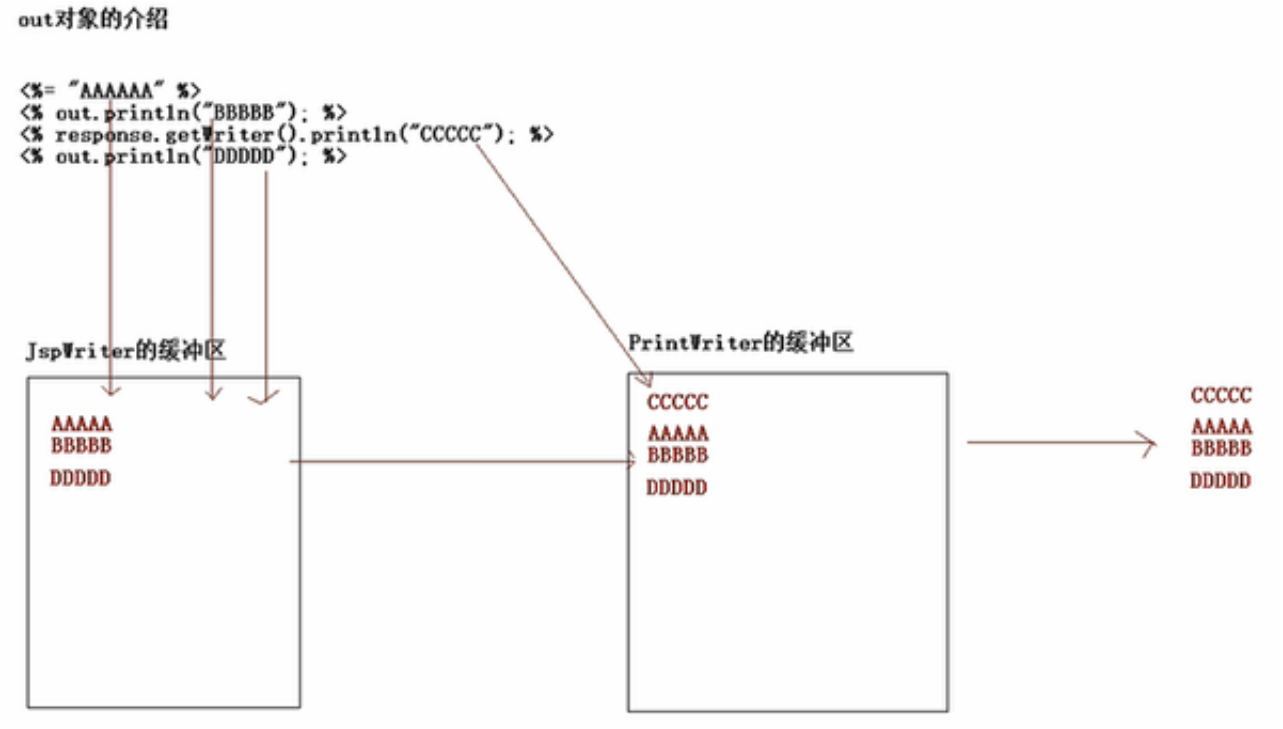
JSP的注释 :<%-- JSP的注释 --%> ---都不存在

【JSP的指令】
指令的语法:
<%@ 指令名称 属性名称=”属性值” 属性名称=”属性值” ...%>
JSP中有三个指令:page指令, include指令, taglib指令.
JSP中page指令:<%@ page %> -- 设置JSP的.
* language:JSP脚本中使用的语言.现在只能写java.
* contentType :设置浏览器打开这个JSP的时候采用的默认的字符集的编码.
* pageEncoding:设置文件保存到本地硬盘,以及生成Servlet后,Servlet保存到硬盘上的编码.
* import :在JSP中引入类对象、jar包.但是import可以出现多次、其他属性只能出现一次.
[mw_shl_code=html,true]<%@page import="java.util.ArrayList"%>
<%@page import="java.util.List"%>[/mw_shl_code]
* extends :设置JSP翻译成Servlet后继承的类,默认值:org.apache.jasper.runtime.HttpJspBase
,这个值要想修改,这个类必须是HttpServlet的子类
* autoFlush :设置JSP的缓存自动刷出.true:自动刷出.
* buffer:设置JSP的缓冲区的大小,默认8kb.
* session:设置在JSP中是否可以直接使用session对象.默认值是true.
* isELIgnored:设置在JSP中是否忽略EL表达式.默认值是false不忽略.
* errorPage:设置错误友好页面的提示.
* isErrorPage:通过这个设置显示JSP的错误信息.
* 开发中一般会设置全局的错误友好页面:
* 在web.xml中设置:
[mw_shl_code=xml,true]<error-page>
<error-code>404</error-code>
<location>/404.jsp</location>
</error-page>
<error-page>
<error-code>500</error-code>
<location>/500.jsp</location>
</error-page>[/mw_shl_code]
JSP中的include指令:静态包含指示JSP包含其他的页面（公共部分）.
[mw_shl_code=html,true]<%@ include file="logo.jsp" %>
<%@ include file="menu.jsp" %>

<h1>BODY部分</h1>
<%@ include file="footer.jsp" %>[/mw_shl_code]
JSP中的taglib指令:指示JSP引入标签库.
<%@ taglib uri="标签的URI的路径" prefix="标签的别名" %>
【JSP的内置对象(*****)】
JSP的内置对象:在JSP中可以直接使用的对象.
JSP中有9大内置对象:

内置对象	真实对象	方法
request	HttpServletRequest	getParameter(),setAttribute(String name,Object value)
response	HttpServletResponse	setHeader(String name,String value);getOutputStream();getWriter()
session	HttpSession	setAttribute();getAttribute()
application	ServletContext	setAttribute();getAttribute()
page	Object	toString();wait()
pageContext	pageContext	setAttribute();getAttribute()
config	ServletConfig	getServletName();getServletContext()
out	JspWriter	write(),print()
exception	Throwable	getMessage(),getCause()

page内置对象：真实对象是Object,就是JSP翻译成Servlet后的类的引用.
out内置对象：out和response.getWriter是不是同一个对象?区别是什么?
* 不是out真实对象JspWriter ,response获得Writer是PrintWriter.



pageContext内置对象　　:

* 获得其他的8个内置对象　:编写通用性代码或者框架的时候.

* 向JSP的四个域中存取数据　:

　JSP的四个域范围:

域	有效范围	内置对象	实际对象
PageScope	当前页面中有效	pageContext	PageContext
RequestScope	一次请求范围	equest	HttpServletRequest
SessionScope	一次会话范围(多次请求)	session	HttpSession
ApplicationScope	应用范围	application	ServletContext

pageContext.findAttribute(“name”):先从小范围到大范围

【JSP的动作标签】列出6个.：简化代码编写的一些标签

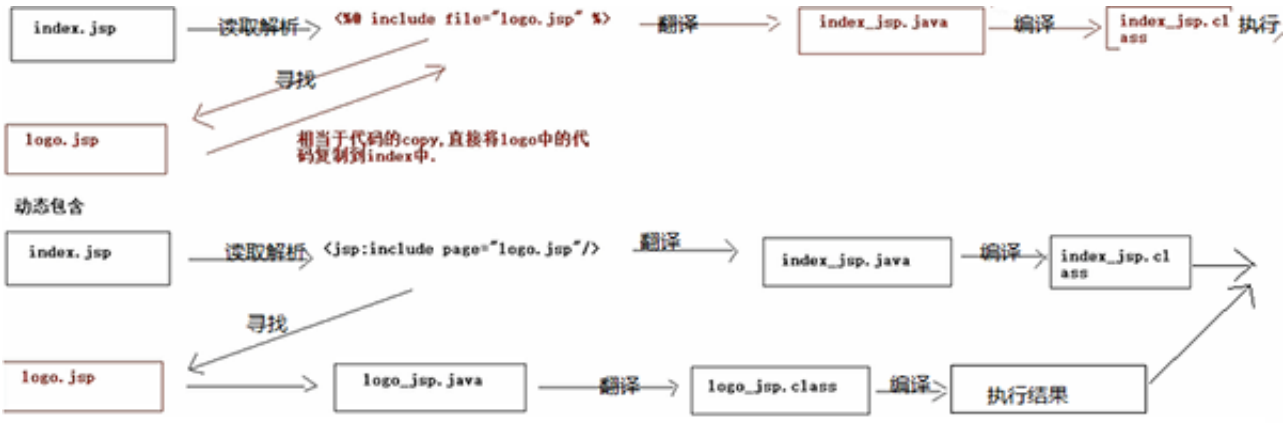
标签的作用:简化代码.

<jsp:forward />　　:用于页面的转发.

* <jsp:forward page="/demo1-jsp/demo3-object/demo3.jsp"></jsp:forward>

<jsp:include />　　:用于页面的包含.（动态包含）动态包含包含的是结果

*****静态包含和动态包含的区别?(<%@ include%>和<jsp:include>)



<jsp:param /> :用于带有路径的标签下,传递参数.

<jsp:useBean /> :用于在JSP中使用JavaBean.

<jsp:setProperty /> :用于在JSP中向JavaBean设置属性的.

<jsp:getProperty /> :用于在JSP中获得JavaBean的属性.

1.2.2 EL表达式:

【EL的概述】

什么是EL:

* EL是为了使jsp写起来更加简单，它提供了在jsp中简化表达式的方法。

为什么学习EL:

* 简化JSP的代码,而且减少<%%>

使用EL表达式:

* 语法:${ EL表达式 }

EL的功能:

* 获取数据:(JSP的四个域)

* 执行运算:

* 操作WEB开发的常用的对象:

* 调用Java中方法:--很少用.

【EL获取数据】

[mw_shl_code=html,true]<h3>存取是普通的单值数据</h3>

```
<%
    //pageContext.setAttribute("name", "pValue");
    //request.setAttribute("name", "rValue");
    //session.setAttribute("name", "sValue");
```

```
    application.setAttribute("name", "aValue");
%>
<%=pageContext.getAttribute("name") %> <!-- 如果没找到 返回null -->
<%=request.getAttribute("name") %>
<%=session.getAttribute("name") %>
<%=application.getAttribute("name") %>
<hr/>
${ pageScope.name } <!-- 返回的是"" -->
${ requestScope.name }
${ sessionScope.name }
${ applicationScope.name }
<hr/>
${ name } <!-- 类似findAttribute("name") 先从page域中查找,没找到去request域中查询,没有找到去session域中找，没有找到就去application域中找 -->
<h3>获取数组的数据</h3>
<%
    String[] arrs = {"李旭华","李冠希","杨凤","杨如花"};
    pageContext.setAttribute("arrs", arrs);
%>
${ arrs[0] }
${ arrs[1] }
${ arrs[2] }
${ arrs[3] }
<h3>获取List集合的数据</h3>
<%
    List<String> list = new ArrayList<String>();
    list.add("李芙蓉");
    list.add("杨芙蓉");
    list.add("王凤");
    pageContext.setAttribute("list", list);
%>
${ list[0] }
```

```
${ list[1] }
${ list[2] }
<h3>获取Map集合的数据</h3>
<%
    Map<String,String> map = new HashMap<String,String>();
    map.put("aaa","李旭华");
    map.put("bbb","杨久君");
    map.put("ccc","李芮");
    map.put("ddd","李凤");
    pageContext.setAttribute("map", map);
%>
${ map.aaa }
${ map.bbb }
${ map.ccc }
${ map.ddd }
<h3>获取对象的数据</h3>
<%
    User user = new User(1,"aaa","123");
    pageContext.setAttribute("user", user);
%>
${ user.id }
${ user.username }
${ user.password }
<h3>获取对象的集合的数据</h3>
<%
    User user1 = new User(1,"aaa","123");
    User user2 = new User(2,"bbb","123");
    User user3 = new User(3,"ccc","123");

    List<User> userList = new ArrayList<User>();
    userList.add(user1);
    userList.add(user2);
```

```
    userList.add(user3);
    pageContext.setAttribute("userList", userList);
%>
${ userList[0].id } - ${ userList[0].username } - ${ userList[0].password }<br/>
${ userList[1].id } - ${ userList[1].username } - ${ userList[1].password }<br/>
${ userList[2].id } - ${ userList[2].username } - ${ userList[2].password }<br/>[/mw_shl_code]
***** .和[]的区别.
    * []用于有下标的数据(数组,list集合) .用于有属性的数据(map,对象)
    * 如果属性名中包含有特殊的字符.必须使用[]
【EL执行运算】
[mw_shl_code=html,true]<h1>EL的功能二:执行运算</h1>
<h3>EL执行算数运算</h3>
<%
    pageContext.setAttribute("n1", "10");
    pageContext.setAttribute("n2", "20");
    pageContext.setAttribute("n3", "30");
    pageContext.setAttribute("n4", "40");
%>
${ n1 + n2 + n3 }
<h3>EL执行逻辑运算</h3>
${ n1 < n2 } - ${ n1 lt n2 } <!-- less than --><br/>
${ n1 > n2 } - ${ n1 gt n2 } <!-- great than --><br/>
${ n1 <= n2 } - ${ n1 le n2 } <!-- less equal --><br/>
${ n1 >= n2 } - ${ n1 ge n2 } <!-- great equal --><br/>
${ n1 == n2 } - ${ n1 eq n2 } <!-- equal --><br/>
<h3>EL执行关系运算</h3>
${ n1<n2 && n3 < n4 } - ${ n1<n2 and n3 < n4 }<br/>
${ n1<n2 || n3 < n4 } - ${ n1<n2 or n3 < n4 }<br/>
${ !(n1 < n2) } - ${ not(n1<n2) }
<h3>EL执行三元运算</h3>
${ n1 < n2 ? "正确":"错误" }
<h3>empty运算</h3>
```

```
${ user == null } - ${ empty user }
${ user != null } - ${ not empty user }
【EL操作WEB开发的常用对象11个】
<h1>EL功能三:操作WEB开发常用的对象</h1>
<!--
    pageScope,requestScope,sessionScope,applicationScope - 获取JSP中域中的数据
    param,paramValues     - 接收参数.
    header,headerValues - 获取请求头信息
    initParam           - 获取全局初始化参数
    cookie              - WEB开发中cookie
    pageContext           - WEB开发中的pageContext.
-->
<h3>接收请求的参数</h3>
<%= request.getParameter("id") %>
<%= request.getParameter("name") %>
<%= Arrays.toString(request.getParameterValues("hobby")) %>
<hr/>
${ param.id }
${ param.name }
${ paramValues.hobby[0] }
${ paramValues.hobby[1] }
<h3>获取请求头</h3>
<%= request.getHeader("User-Agent") %>
<hr/>
${ header["User-Agent"] }
<h3>获取全局初始化参数</h3>
${ initParam.username }
<h3>获取Cookie中的值</h3>
${ cookie.history.value }
<h3>获取PageContext中的对象</h3>
IP地址：${ pageContext.request.remoteAddr }
工程路径:${ pageContext.request.contextPath }[/mw_shl_code]
```

1.2.3 JSTL

【JSTL的概述】

什么是JSTL:

Jstl即jsp标准标签库。是一个开源的jsp标签库,

为什么学习JSTL:

* JSTL和EL结合 替换页面中<%%>

JSTL版本:

* JSTL1.0 :不支持EL表达式.
* JSTL1.1 和 1.2 :支持EL表达式.

JSTL的标签库:包含了五类标签.

* core(核心标签),fmt(国际化标签),xml(XML标签),sql(SQL标签),fn(JSTL提供EL函数库)

使用JSTL:

* 引入JSTL的相关的jar包.
* 在页面中引入标签库.<%@ taglib uri="" prefix=""%>

【JSTL的核心标签的用法】

* if
* forEach

【JSTL的提供EL的函数库】

[mw_shl_code=html,true]<h1>JSTL提供的EL的函数库</h1>
${ fn:contains("Hello World","Hello") }
${ fn:length("HelloWorld") }
${ fn:toLowerCase("ABCDE") }
<c:forEach var="i" items='${ fn:split("a-b-c-d","-") }'>
  ${ i }
</c:forEach>[/mw_shl_code]

1.3 代码实现:

1.3.1 创建数据库:

[mw_shl_code=sql,true]CREATE TABLE `product` (
 `pid` varchar(32) NOT NULL,
 `pname` varchar(50) DEFAULT NULL,
 `market_price` double DEFAULT NULL,
 `shop_price` double DEFAULT NULL,

```
  `pimage` varchar(200) DEFAULT NULL,
  `pdate` date DEFAULT NULL,
  `is_hot` int(11) DEFAULT NULL,
  `pdesc` varchar(255) DEFAULT NULL,
  `pflag` int(11) DEFAULT NULL,
  `cid` varchar(32) DEFAULT NULL,
  PRIMARY KEY (`pid`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;[/mw_shl_code]
```

1.3.2 页面显示:

```
[mw_shl_code=html,true]<c:forEach var="p" items="${list }">
  <tr>
    <td>${ p.pid }</td>
    <td>${ p.pname }</td>
    <td>${ p.shop_price }</td>
    <td>
      <c:if test="${ p.is_hot == 1 }">
        是
      </c:if>
      <c:if test="${ p.is_hot != 1 }">
        否
      </c:if>
    </td>
    <td>${ p.pdesc }</td>
  </tr>
  </c:forEach>[/mw_shl_code]
```

回帖

5条回帖

zmanx

楼主有没有关于API的知识总结啊,有试题也可以啊

沙发 • 2017-2-12 22:00:12

回帖

爱吃橘子的小泽

顶顶顶,厉害 牛牛牛

藤椅 • 2017-2-12 22:04:10

回帖

小虎同学

厉害，66666

板凳 • 2017-2-13 00:06:09

回帖

来自宇宙超级黑马专属苹果客户端

yibuhuiba

谢谢分享

报纸 • 2017-2-18 19:11:51

回帖

爱如少年、

ganxiefenxiang

地板 • 2018-3-19 15:45:33

回帖