jsp使用详解

# jsp概述

jsp：sun公司制订的一种服务器端动态页面生成技术规范。因为直接使用servlet虽然也可以生成动态页面,但是过于繁琐(大量使用out.println语句)，维护也困难，所以，sun公司制订了jsp规范： jsp实际是一个以".jsp"为后缀的文件，该文件主要由html和少量的java代码组成，容器会将jsp文件先转换成一个对应的servlet然后再去执行。

JSP的运行机制

**翻译阶段**：当有一个对JSP页面的客户请求到来时，JSP容器检验JSP页面的语法是否正确，将JSP页面转换为Servlet(jsp-->servlet)

**编译阶段**：调用javac工具类编译将Servlet源文件生成字节码文件。

**执行阶段**：Servlet容器加载转换后的Servlet类，实例化一个对象处理客户端的请求，在请求处理完成后，响应对象被JSP容器接受，容器将HTML格式的响应信息发送到客户端。 调用\_jspService()

第一次加载JSP页面时，因为要将JSP文件转换为Servlet类，响应速度较慢，再次请求时，JSP容器就会直接执行第一次请求时产生的Servlet，不会重新转换JSP文件，在JSP执行期间，JSP容器会检查JSP文件，看是否有更新或修改，如果有，JSP容器会再次编译JSP或者Servlet，如果没有更新或者修改，直接执行前面产生的Servlet.

JSP页面转换后的Servlet类必须实现javax.servlet.jsp.JspPage接口，绝大多数情况下，JSP页面都使用HTTP协议，所以在JSP页面转换后的Servlet类必须实现javax.servlet.jsp.HttpJspPage接口，该接口继承自JspPage接口。

\_jspService(HttpServletRequest request, HttpServletResponse respons);

该方法用来处理业务逻辑。

# 备注

<%-- jsp备注不会发送到浏览器，也不能进行嵌套--%>

<!-- html备注，会发送到浏览器 -->

# 隐式对象

|  |
| --- |
| **#9种隐式对象**  page对象: 当前的jsp页面，指针this  pageContext对象: 当前的jsp页面的上下文  request对象： 封装请求的信息  response对象： 封装响应  session对象： 会话  application对象： 指web应用  out对象： 输出流  config对象： 配置信息  exception对象： 异常信息  **对象和范围**  **page范围**：具有page范围的对象被绑定到javax.servlet.jsp.PageContext对象中，在这个范围中的对象，只能在创建对象的页面中访问。page范围内的对象，在客户端每次请求JSP页面时创建，在页面向客户端发送回响应或请求被转发到其他资源后被删除。  **request范围：**具有request范围的对象被绑定到javax.servlet.http.HttpServlet Request对象中。在调用forward()方法转向的页面或调用include()方法包含的页面中，都可以访问这个范围内的对象。  **session范围**：具有session返回的对象被绑定到javax.servlet.http.HttpSession对象中，JSP容器为每次会话，创建一个HttpSession对象，在会话期间，可以访问session范围内的数据  **application范围**:具有application范围的对象被绑定到javax.servlet.Servlet Context中，在web应用程序运行期间，所有页面都可以访问这个范围内的对象 |

# 指令

## page指令

**语法**: <%@ page attributeName="attribuerValue"...%>

**作用**: 利用page指令中的属性可以和容器进行通信，这些属性的设置，对整个jsp都有影响。

**page指令中的属性**:

language="java"： 定义当前页面使用的编程语言，目前只能是java

contentType="text/html;charset=UTF-8"：定义response中的内容类型和编码格式和jsp页面的编码格式，翻译后变成response.setContentType("text/html;charset=UTF-8")。

pageEncoding="UTF-8"：定义当前jsp页面的编码格式

isErrorPage="true|false" 默认为false，表明当前的页面是否为其它页面的errorPage目标，如果设置为true，则可以使用exception对象。反之，如果设置为false，则不可以使用exception对象

errorPage="error.jsp"默认忽略,定义此页面出现异常时调用的页面

import="package"和一般的java import意义一样，用","来隔开

session="true|false" 默认为true，指定这个jsp中是否可以使用session

buffer="none|8kb|sizekb" 默认为8kb。指定到客户端输出流的缓冲模式，如果是none则不缓冲，如果指定数值，那么输出就用不小于这个值的缓冲区进行缓冲。

autoFlush="true|false" 默认为true，true 当缓冲区满时，到客户端的输出被刷新false 当缓冲区满时，出现运行异常，表示缓冲溢出

isThreadSafe="true|false" 默认为true，用来设置jsp文件是否能支持多线程的使用。如为true，那么一个jsp能够同时处理多个用户的请求，反之，一个jsp只能一次处理一个请求。

info="text"：关于jsp页面的信息，使用getServletInfo()获得

extends="package.class"指定jsp对应的servlet类继承某个父类

isELIgnored="true|false"，指定EL表达式语言是否被忽略，如为true则忽略，反之可以使用EL。

## include指令

**语法**: <%@ include file="url"%>，称为静态导入(静态包含)

**作用**: 在一个页面中导入另一个页面的内容(这些内容一般不会有变化，如公司的标题和版权等信息)。

**特点**: 在jsp翻译成servlet后就将被导入的页面内容嵌入到servlet中。导入时间发生在翻译阶段。被导入的资源:html、xml、jsp等

**优点**: 执行效率高(只翻译一次)

**缺点**: 当被导入页面的内容发生变化，那么jsp必须要重新被翻译。如果导入的多个页面，其中的一个页面的内容发生变化，就需要将所有的页面都翻译一次。

<% include file="copy.jsp'%>

## taglab指令

<%@ taglib uri="" prefix=""%> //导入jstl标签库

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

指定在jsp中要使用的标签库描述文件(\*.tld)的路径

# 脚本元素

## java代码块

语法：<% 代码 %>

## 表达式

**语法**: <%=expression%>

**作用**: 将expression输出到out(输出流)中，

expression可以是算术、逻辑、常量、 关系表达式、变量、有返回值的方法、jsp中的9种隐含对象。

## 声明

**语法**: <%!声明的内容%> //可以定义在jsp任意位置

**作用**: 在servlet中声明一些成员变量、成员方法和内部类

**特点**: 声明被翻译进servlet后变成了成员变量、成员方法和内部类

**注意**: 不能使用表达式和隐含对象

## 关闭脚本元素

jsp2建议在EL中访问服务器对象，而不是在jsp中编写代码。

<jsp-property-group>

<url-pattern> \*.jsp</url-pattern>

</jsp-property-group>

# 动作

语法: <jsp:tagName/>(访问某个java对象，或者调用某个方法)

## useBean

//获取对象

<jsp:useBean id="变量名" class="变量所在类">

## setProperty和getProperty

#设置或获取属性值

<jsp:setProperty name="employee" property="name" value="hello"/>

<jsp:getProperty name="employee" property="name"/>

## include

动态导入发生在执行阶段，也就是在运行serlvet的时候才动态生成被导入页面的内容，然后嵌入到调用页面，最后将两个页面的内容一起，返回给客户端

<jsp:include page="jspf/menu.jsp">

<jsp:param name="text" value="hello"/>

</jsp:include>

## forward

//转发

<jsp:forward page="jsp/login.jsp">

<jsp:param name="text" value="please login"/>

</jsp:forward>

## 错误处理

<%@page isErrowPage="true">定义一个错误页面。

在其他的jsp引入错误页面

<%@page errorPage="error.jsp"%>