

# jsp笔记

2019年4月1日 20:47

## jsp指令

page

### Jsp基础语法

#### Jsp指令

page指令语法：

```
<%@ page 属性1="属性值" 属性2="属性值1,属性值2"...  
属性n="属性值n"%>
```

属性	描述	默认值
language	指定JSP页面使用的脚本语言	java
import	通过该属性来引用脚本语言中使用到的类文件	无
contentType	用来指定JSP页面所采用的编码方式	text/html, ISO-8859-1

慕课网

## 注释 3种

- html注释 客户端可见
- jsp注释 客户端不可见
- jsp脚本注释 java注释 单行 多行

### Jsp页面元素

#### Jsp注释

在JSP页面的注释。

HTML的注释：

```
<!-- html注释--> //客户端可见
```

JSP的注释：

```
<%-- html注释--%> //客户端不可见
```

JSP脚本注释：

```
//单行注释  
/*多行注释
```

SHOT ON MI 9 SE  
AI TRIPLE CAMERA

慕课网

## jsp脚本



在jsp中加入java代码

<%java代码%>

**jsp的声明** 在jsp中定义变量或方法 用java脚本格式声明

<%! java声明%>

**jsp表达式**

<%=表达式%>

注意：表达式不以分号结束

**jsp生命周期**

两张ipad图片带插入

### ➤ Jsp内置对象

web容器创建的一组对象，不需要使用new关键字就可以使用的内置对象

如 out对象 out.print();

jsp9大内置对象

常用的有5个： out, request, response, session, application

不常用的4个： page,pageContext,exception,config

### 内置对象详细介绍

#### ❖ Web程序的请求响应模式

- 用户发送请求(request)
- 服务器给用户响应(response)
- **out对象**：缓冲区：内存的一块区域用来保存临数据。是JspWriter类的实例，向客户端输出内容常用的对象。常用方法如下：

1.void println()向客户端打印字符串 ----输出HTML标签

2.void clear()清除缓冲区的内容，如果在flush之后调用会抛出异常 ---不能在flush之后调用

3.void clearBuffer():清除缓冲区的内容，如果在flush之后调不用会抛出异常

4.void flush():将缓冲区内容输出到客户端

5.int getBufferSize() 返回缓冲区以字节数的大小，如不设缓冲区则为0

6.int getRemaining() 返回缓冲区还剩余多少可用

7.boolean isAutoFlush() 返回缓冲区满时，是自动清空还是抛出异常

8.void close() 关闭输出流

#### • get与post

<form name="regForm" action="动作" method="提交方式">

</form>

action表示该表单提交给哪个动作（可以是jsp文件）去处理，name 表示该表单所起的名



字。method主要有2种方式：get和post

表单的2种提交方式：

1.get:以明文的方式通过URL提交数据，数据在URL中可以看到。提交的数据最多不超过2kb.安全性较低但效率较post高。适合提交数据量不大，安全性不高的数据。比如：搜索、查询等功能。

2.post:将用户提交的信息封装在HTML HEADER内。适合提交数据量大，安全性高的用户信息。比如：注册、修改、上传等功能。

- **request对象**

客户端的请求信息被封装在request对象中，通过它才能了解到客户的需求，然后做出响应。它是HttpServletRequest类的实例。request对象具有请求域，即完成客户端的请求之前，该对象一直有效。常用方法如下：

- 1.String getParameter(String name)返回name指定参数的参数值--获得单个值
- 2.String[] getParameterValues(String name)返回包含参数name的所有值的数组 ---获得集合值
- 3.void setAttribute(String,Object) 存储此请求中的属性
- 4.object getAttribute(String name)返回指定属性的属性值 ---输入的是属性所对应的键，得到的是属性所对应的值
- 5.String getContentType()得到请求体的MIME类型
- 6.Sting getProtocol()返回请求用的协议类型及版本号
- 7.String getServerName()返回接受请求的服务器主机名
- 8.int getServerPort()返回服务器接受此请求所用的端口号
- 9.String getCharacterEncoding()返回字符编码方式
- 10.void setCharacterEncoding()设置请求的字符编码方式
- 11.int getContentLength()返回请求体的长度（以字节数）
- 12.String getRemoteAddr()返回发送此请求的客户端IP地址
- 13.String getRealPath(String path)返回一虚拟路径的真实路径
- 14.String request.getContextPath()返回上下文路径

处理中文乱码，最简单的方法是在接收的对象之前加上

```
<%  
    Request.setCharacterEncoding("utf-8"); //这里的utf-8必须和对象所在页面的编码一样，无法解  
    决url传参的时候出现的中文乱码问题，解决url传中文参需要配置tomcat的config.xml文件  
%>
```

- **response对象**

包含了响应客户请求的有关信息，但在JSP中很少直接用到它。它是HttpServletResponse类的实例。response对象具有页面作用域，即访问一个页面时，该页面内的response对象只能对这次访问有效，其它页面的response对象对当前页面无效。常用方法如下：

- 1.String getCharacterEncoding()返回响应应用的何种字符编码
- 2.void setContentType(String type)设置响应的MIME类型
- 3.PrintWriter getWriter()返回可以向客户端输出字符的一个对象（注意比较PrintWriter与内置对象out的区别）PrintWriter对象的输出流总是提前于out对象，解决方法，在out.println()之后使用out.flush()



4.sendRedirect(java.lang,String location)重新定向客户端的请求

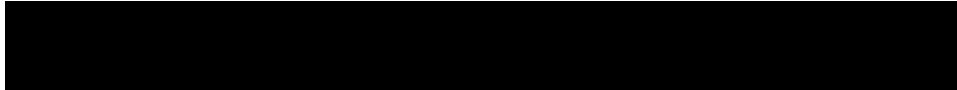
- **请求转发与请求重定向的区别**

请求重定向：客户端行为，response.sendRedirect()，从本质上讲等同于两次请求，前一次的请求对象不会保存，地址栏的URL地址会改变。

请求转发：服务器行为，request.getRequestDispatcher().forward(req,resp);是一次请求，转发后请求对象会保存，地址栏的URL地址不会改变。

- **Session对象：**

一次连接相当于一次session，同意服务器服务不同客户，每一个客户有一个session，直到用户关闭所有连接或者超时session才会断开连接。



## JSP内置对象

session对象

- session对象是一个JSP内置对象。
- session对象在第一个JSP页面被装载时自动创建，完成会话期管理。
- 从一个客户打开浏览器并连接到服务器开始，到客户关闭浏览器离开这个服务器结束，被称为一个会话。
- 当一个客户访问一个服务器时，可能会在服务器的几个页面之间切换，服务器应当通过某种办法知道这是一个客户，就需要session对象。
- session对象是HttpSession类的实例。

慕课网



## JSP内置对象

session对象常用方法如下：

- long getCreationTime()：返回SESSION创建时间
- public String getId()：返回SESSION创建时JSP引擎为它设的唯一ID号
- public Object setAttribute(String name, Object value)：使用指定名称将对象绑定到此会话
- public Object getAttribute(String name)：返回与此会话中的指定名称绑定在一起的对象，如果没有对象绑定在该名称下，则返回null
- String[] getValueNames()：返回一个包含此SESSION种所有可用属性的数组
- int getMaxInactiveInterval()：返回两次请求间隔多长时间此SESSION被取消（单位秒）

慕课网







## JSP内置对象

### Session的生命周期

#### 活动：

- 某会话当中通过超链接打开的新页面属于同一次会话。
- 只要当前会话页面没有全部关闭，重新打开新的浏览器窗口访问同一项目资源时属于同一次会话。
- 除非本次会话的所有页面都关闭后再重新访问某个Jsp或者Servlet将会创建新的会话。

注意事项：注意原有会话还存在，只是这个旧的SessionId仍然存在于服务端，只不过再也没有客户端会携带它然后交予服务端校验。



## JSP内置对象

### Session的生命周期

#### 销毁：

Session的销毁只有三种方式：

- 1、调用了session.invalidate()方法
- 2、Session过期（超时）
- 3、服务器重新启动





## JSP内置对象

### Session对象

- Tomcat默认session超时时间为30分钟。
- 设置session超时有两种方式：
  - 1.session.setMaxInactiveInterval(时间);//单位是秒
  - 2.在web.xml配置

```
<session-config>
<session-timeout>
    10
</session-timeout>
</session-config> //单位是分钟。
```

777777

慕课网

- **application对象**：--类似于Java中的static class
  - 1.application对象实现了用户数据的共享，可存放全局变量
  - 2.application对象开始于服务器的启动，终止于服务器的关闭
  - 3.在用户的前后连接或不同用户之间的连接中，可以对application对象的同一属性进行操作。
  - 4.在任何地方对application对象属性的操作，都将影响到其他用户对此的访问。
  - 5.服务器的启动和关闭决定了application对象的生命。
  - 6.是ServletContext类的实例

常用方法：

## JSP内置对象

### application对象:

常用方法如下：

- public void setAttribute(String name, Object value)使用指定名称将对象绑定到此会话。
- public Object getAttribute(String name)返回与此会话中的指定名称绑定在一起的对象，如果没有对象绑定在该名称下，则返回 null。
- Enumeration getAttributeNames() 返回所有可用属性名的枚举
- String getServerInfo() 返回JSP(SERVLET)引擎名及版本号

慕课网

- **page对象**:



指向当前JSP页面本身，有点像Java中类的this

## JSP内置对象

### page对象

page对象就是指向当前JSP页面本身，有点像类中的this指针，它是java.lang.Object类的实例。常用方法如下：

- class getClass() 返回此Object的类
- int hashCode() 返回此Object的hash码
- boolean equals(Object obj) 判断此Object是否与指定的Object对象相等
- void copy(Object obj) 把此Object拷贝到指定的Object对象中
- Object clone() 克隆此Object对象
- String toString() 把此Object对象转换成String类的对象
- void notify() 唤醒一个等待的线程
- void notifyAll() 唤醒所有等待的线程
- void wait(int timeout) 使一个线程处于等待直到timeout结束或被唤醒
- void wait() 使一个线程处于等待直到被唤醒

慕课网

### • pageContext对象

## JSP内置对象

### pageContext对象

- pageContext对象提供了对JSP页面内所有的对象及名字空间的访问
- pageContext对象可以访问到本页所在的session，也可以取本页面所在的application的某一属性值
- pageContext对象相当于页面中所有功能的集大成者
- pageContext对象的本类名也叫pageContext。

慕课网



## JSP内置对象

### pageContext对象

常用方法如下：

- JspWriter getOut() 返回当前客户端响应被使用的JspWriter流(out)
- HttpSession getSession() 返回当前页中的HttpSession对象(session)
- ■ Object getPage() 返回当前页的Object对象(page)
- ServletRequest getRequest() 返回当前页的ServletRequest对象(request)
- ServletResponse getResponse() 返回当前页的ServletResponse对象(response)
- void setAttribute(String name,Object attribute) 设置属性及属性值
- Object getAttribute(String name,int scope) 在指定范围内取属性的值
- int getAttributeScope(String name) 返回某属性的作用范围
- void forward(String relativeUrlPath) 使当前页面重定向到另一页面
- void include(String relativeUrlPath) 在当前位置包含另一文件

慕课网

### • Config对象

## JSP内置对象

### Config对象

config对象是在一个Servlet初始化时，JSP引擎向它传递信息用的，此信息包括Servlet初始化时所要用的参数（通过属性名和属性值构成）以及服务器的有关信息（通过传递一个ServletContext对象），常用方法如下：

- ServletContext getServletContext() 返回含有服务器相关信息的ServletContext对象
- String getInitParameter(String name) 返回初始化参数的值
- Enumeration getInitParameterNames() 返回Servlet初始化所需所有参数的枚举

慕课网

### • exception对象

在使用exception对象的 page 中要加入属性 isErrorPage='true'，表示该页面可使用exception对象在抛出异常的页面的中的page中加入属性 errorPage="exception.jsp" -----(这里面的exception.jsp即为自己写的处理异常的jsp页面)





## JSP内置对象

### Exception对象

exception对象是一个异常对象，当一个页面在运行过程中发生了异常，就产生这个对象。如果一个JSP页面要应用此对象，就必须把isErrorPage设为true，否则无法编译。他实际上是java.lang.Throwable的对象,常用方法如下：

- String getMessage() 返回描述异常的消息
- String toString() 返回关于异常的简短描述消息
- void printStackTrace() 显示异常及其栈轨迹
- Throwable FillInStackTrace() 重写异常的执行栈轨迹

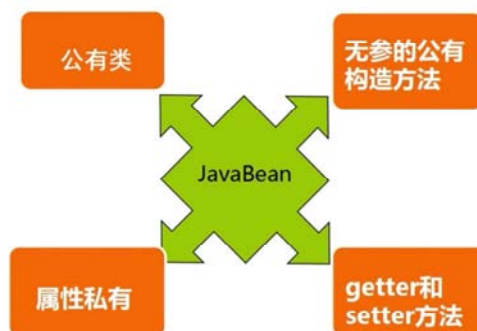
慕课网

### ➤ Javabean

- Javabean简介：Javabeans就是符合某种特定的规范的Java类。使用Javabeans的好处是解决代码重复编写，减少代码冗余，功能区分明确，提高了代码的维护性。
- 设计原则

## Javabean

### Javabean的设计原则



慕课网

- 什么是JSP动作元素

动作元素为请求处理阶段提供信息。动作元素遵循XML元素的语法，有一个包含元素名的开始标签，可以有属性、可选的内容、与开始标签匹配的结束标签。



# JavaBean

## 什么是Jsp动作

第一类是与存取JavaBean有关的，包括：

`<jsp:useBean>` `<jsp:setProperty>` `<jsp:getProperty>`

第二类是JSP1.2就开始有的基本元素，包括6个动作元素

`<jsp:include>` `<jsp:forward>` `<jsp:param>` `<jsp:plugin>` `<jsp:params>` `<jsp:fallback>`

第三类是JSP2.0新增的元素，主要与JSP Document有关，包括六个元素

`<jsp:root>` `<jsp:declaration>` `<jsp:scriptlet>` `<jsp:expression>` `<jsp:text>` `<jsp:output>`

第四类是JSP2.0新增的动作元素，主要是用来动态生成XML元素标签的值，包括3个动作

`<jsp:attribute>` `<jsp:body>` `<jsp:element>`

第五类是JSP2.0新增的动作元素，主要是用在Tag File中，有2个元素

`<jsp:invoke>` `<jsp:doBody>`



第一类：

如何在JSP页面中使用JavaBeans

- 1.像使用普通Java类一样，创建JavaBean实例。--需要在page中import相应的包和类
- 2.在JSP页面中使用jsp动作标签使用JavaBean（3种动作标签）：`useBean`动作、`setProperty`动作、`getProperty`动作

### `<jsp:userBeans>`

作用：在jsp页面中实例化或者在指定范围内使用javabeen:

`<jsp:useBean id="标识符" class="java类名" scope="作用范围">`

标识符：唯一的标识符，也就是 class中的类的实例名，用该实例可以调用class中的方法

Java类名：全名，包括包名和类名

scope可以写成 `scope= "page"`

### `<jsp:setProperty>`



## Javabean

### <jsp:setProperty>

作用：给已经实例化的Javabean对象的属性赋值,一共有四种形式。

<jsp:setProperty name = "JavaBean实例名" property = "\*" /> (跟表单关联)

<jsp:setProperty name = "JavaBean实例名" property = "JavaBean属性名" />  
(跟表单关联)

<jsp:setProperty name = "JavaBean实例名" property = "JavaBean属性名"  
value = "BeanValue" /> (手工设置)

<jsp:setProperty name = "JavaBean实例名" property = "propertyName" param  
= "request对象中的参数名"/> (跟request参数关联)

### <jsp:getProperty>

## Javabean

### <jsp:getProperty>

作用：获取指定Javabean对象的属性值。

<jsp:getProperty name="JavaBean实例名" property="属性名" />

根据表单里面的名字和实例的属性名进行匹配，自动匹配所有的属性

根据表单里面的名字和实例的属性名进行匹配，自动匹配部分的属性

跟表单没有关系，自己给属性设置值，value是自己认为设置的

## JavaBean的四个作用域范围

说明：使用useBeans的scope属性可以用来指定javabeen的作用范围。

- page //仅在当前页面有效
- request //可以通过HttpRequest.getAttribute()方法取得JavaBean对象。
- session //可以通过HttpSession.getAttribute()方法取得JavaBean对象。
- applicatoin //可以通过application.getAttribute()方法取得Javabeen对象

## ➤ Model 1简介

### Model1简介

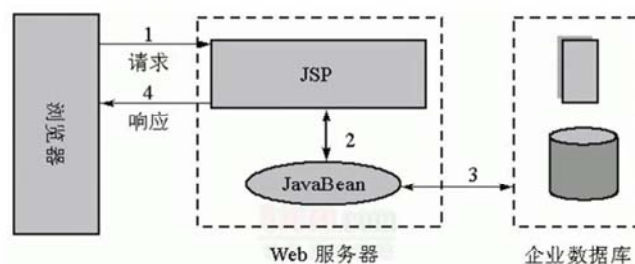
- Model 1模型出现前，整个Web应用的情况：几乎全部由JSP页面组成，JSP页面接收处理客户端请求，对请求处理后直接做出响应。

### Model1简介

- Javabeen的出现可以使jsp页面中使用Javabeen封装的数据或者调用Javabeen的业务逻辑代码,这样大大提升了程序的可维护性。

## JavaBean

Model1简介



## ➤ Http协议的无状态性

### Jsp状态管理

#### http协议的无状态性

无状态是指，当浏览器发送请求给服务器的时候，服务器响应客户端请求。

但是当同一个浏览器再次发送请求给服务器的时候，服务器并不知道它就是刚才那个浏览器。

简单地讲，就是服务器不会去记得你，所以就是无状态协议。





Jsp状态管理

什么是Cookie？

Cookie：中文名称为“小甜饼”，是Web服务器保存在客户端的一系列文本信息。

典型应用一：判定注册用户是否已经登录网站。

典型应用二：“购物车”的处理。

Jsp状态管理

Cookie的作用

- 对特定对象的追踪
- 保存用户网页浏览记录与习惯
- 简化登录

安全风险：容易泄露用户信息

Jsp状态管理

Jsp中创建与使用Cookie

创建Cookie对象

```
Cookie newCookie = new Cookie(String key,Object value);
```

写入Cookie对象

```
response.addCookie(newCookie);
```

读取Cookie对象

```
Cookie[] cookies = request.getCookies();
```

Jsp状态管理

Jsp中创建与使用Cookie

●常用方法

方法名称	说 明
void setMaxAge(int expiry)	设置cookie的有效期，以秒为单位
void setValue(String value)	在cookie创建后，对cookie进行赋值
String getName()	获取cookie的名称
String getValue()	获取cookie的值
int getMaxAge()	获取cookie的有效时间，以秒为单位

Session与Cookie

相同点：都是保存用户状态信息的一种机制；都会过期也就是都有一个生存周期

不同点：



## Jsp状态管理

### Session与Cookie对比



include指令--在一个JSP页面中显示另一个JSP页面，要显示的JSP页面写在URL处。

## 指令与动作

### include指令

语法：

```
<%@ include file="URL"%>
```

Include 动作

## 指令与动作

### include动作

语法：

```
<jsp:include page="URL" flush="true|false"/>
```

page

要包含的页面

flush

被包含的页面是否从缓冲区读取

## 指令与动作

### include指令与include动作比较

	include指令	jsp:include动作
语法格式	<%@ include file= ".." %>	<jsp:include page= ".." >
发生作用的时间	页面转换期间	请求期间
包含的内容	文件的实际内容	页面的输出
转换成的Servlet	主页面和包含页面转换为一个Servlet	主页面和包含页面转换为独立的Servlet
编译时间	较慢——资源必须被解析	较快
执行时间	稍快	较慢——每次资源必须被解析

forward动作：



## 指令与动作

### forward动作

语法：

```
<jsp:forward page="URL" />
```

等同于：

```
request.getRequestDispatcher("/url").forward(request,response);
```

param动作：

## 指令与动作

### param动作

语法：

```
<jsp:param name="参数名" value="参数值">
```

常常与 <jsp:forward >一起使用，作为其的子标签。

