# 尚马教育 JAVA WEB课程

# Jsp入门

文档编号：B13

创建日期： 2017-06-13

最后修改日期：2019-09-23

版 本 号：V2.0

电子版文件名：尚马教育-第二阶段-3.jsp入门.docx

**文档修改记录：**

|  |  |  |  |
| --- | --- | --- | --- |
| 更新日期 | 更新作者 | 更新说明 | 版本号 |
| 2018-08-01 | 王绍成 | 初始版本 | V1.0 |
| 2019-09-13 | 郭雪岩 | 内容更新 | V2.0 |

**主讲人：**

**郭雪岩**

目录

[尚马教育 JAVA WEB课程 1](#_Toc28937)

[Jsp入门 1](#_Toc30446)

[1. JSP介绍 3](#_Toc17679)

[1.1. 什么是JSP 3](#_Toc16815)

[1.2. Jsp本质 3](#_Toc17269)

[2. Jsp语法格式 4](#_Toc4385)

[3. Jsp九种内置对象及4种作用域 5](#_Toc16721)

[3.1. 什么是的内置对象 5](#_Toc28312)

[3.2. 九种内置对象 5](#_Toc20722)

[3.3. 四种作用域 6](#_Toc16450)

[4. Jsp与servlet配合使用 7](#_Toc28774)

[4.1. Jsp与servlet分工 7](#_Toc25135)

[4.2. 跳转方式 7](#_Toc20714)

[5. El表达式 8](#_Toc6242)

[5.1. El表达式作用 8](#_Toc10027)

[5.2. El表达式特点 8](#_Toc10925)

[5.3. El表达式操作 9](#_Toc12867)

[5.4. El表达式隐式对象 9](#_Toc16419)

[6. Jstl标签 11](#_Toc22475)

[6.1. Jstl标签作用 11](#_Toc32614)

[6.2. Jstl使用 11](#_Toc25998)

[6.3. Jstl中常用标签 11](#_Toc25117)

[6.4. Jstl其他标签库 12](#_Toc19854)

## JSP介绍

### 什么是JSP

* JSP（Java Server Page）是JavaEE规范中的Web组件，用来编写动态页面；
* JSP文件以.jsp为后缀，在Eclipse的工程目录中存在WebContent目录下，跟静态资源使用方式非常相似；
* JSP文件中的内容就是 HTML+Java代码，静态部分使用HTML和文本即可，动态部分使用Java代码；

### Jsp本质

* 结合由tomcat自动生成的jsp相关java文件，

理解Web容器处理JSP文件请求需要经过3个阶段

* 翻译阶段
* 编译阶段
* 执行阶段

1、翻译：Web服务器找到test.jsp，对其进行翻译，生成test\_jsp.java文件；查看路径： 工作空间 work\Catalina\localhost

2、编译：服务器将test\_jsp.java编译成类文件，翻译和编译的过程遵守Servlet规范，

3、实例化并提供服务：服务器实例化类，调用类中的\_jspService方法提供服务

* 总结：
* Servlet生成动态页面比较繁琐，使用JSP生成动态页面比较便捷，因为其中的静态内容可以使用HTML生成；
* JSP的执行过程是：翻译-编译-实例化-提供服务；
* JSP的本质就是Servlet，不过是服务器将JSP进行了翻译和编译；可以说，JSP也是一个Java类；

## Jsp语法格式

Jsp中嵌入java代码的格式

|  |
| --- |
| * Page指令   <%@ 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"%>   * Include指令   <%@ include file="left.jsp" %>   * 小脚本   <% %> 小脚本中可以直接声明变量   * 输出信息到页面   <%= %> 直接输出信息到页面，不能带分号   * 声明   <%! %> 声明中定义方法   * 注释   <%-- --%> 与<!-- -->区别 <%-- --%>不出现在页面中 |

## Jsp九种内置对象及4种作用域

### 什么是的内置对象

* 在jsp开发中会频繁使用到一些对象.如果每次我们在jsp页面中需要使用这些对象都要自己亲自动手创建就会特别的繁琐，SUN公司因此在设计jsp时,在jsp页面加载完毕之后自动帮开发者创建好了这些对象,开发者只需要使用相应的对象调用相应的方法即可这些系统创建好的对象就叫做内置对象.
* Jsp文件翻译成java文件时，有一些对象被自动写在了java文件中，这些被自动写进去的对象叫内置对象。

### 九种内置对象

|  |
| --- |
| 1、request对象  request 对象是 javax.servlet.httpServletRequest类型的对象。 该对象代表了客户端的请求信息，主要用于接受通过HTTP协议传送到服务器的数据。（包括头信息、系统信息、请求方式以及请求参数等）。request对象的作用域为一次请求。  2、response对象  response 代表的是对客户端的响应，主要是将JSP容器处理过的对象传回到客户端。response对象也具有作用域，它只在JSP页面内有效。  3、session对象  session 对象是由服务器自动创建的与用户请求相关的对象。服务器为每个用户都生成一个session对象，用于保存该用户的信息，跟踪用户的操作状态。session对象内部使用Map类来保存数据，因此保存数据的格式为 “Key/value”。 session对象的value可以使复杂的对象类型，而不仅仅局限于字符串类型。  4、application对象  application 对象可将信息保存在服务器中，直到服务器关闭，否则application对象中保存的信息会在整个应用中都有效。与session对象相比，application对象生命周期更长，类似于系统的“全局变量”。  5、out 对象  out 对象用于在Web浏览器内输出信息，并且管理应用服务器上的输出缓冲区。在使用 out 对象输出数据时，可以对数据缓冲区进行操作，及时清除缓冲区中的残余数据，为其他的输出让出缓冲空间。待数据输出完毕后，要及时关闭输出流。  6、pageContext 对象  pageContext 对象的作用是取得任何范围的参数，通过它可以获取 JSP页面的out、request、reponse、session、application 等对象。pageContext对象的创建和初始化都是由容器来完成的，在JSP页面中可以直接使用 pageContext对象。  7、config 对象  config 对象的主要作用是取得服务器的配置信息。通过 pageConext对象的 getServletConfig() 方法可以获取一个config对象。当一个Servlet 初始化时，容器把某些信息通过 config对象传递给这个 Servlet。 开发者可以在web.xml 文件中为应用程序环境中的Servlet程序和JSP页面提供初始化参数。  8、page 对象  page 对象代表JSP本身，只有在JSP页面内才是合法的。 page隐含对象本质上包含当前 Servlet接口引用的变量，类似于Java编程中的 this  9、exception 对象  exception 对象的作用是显示异常信息，只有在包含 isErrorPage="true" 的页面中才可以被使用，在一般的JSP页面中使用该对象将无法编译JSP文件。excepation对象和Java的所有对象一样，都具有系统提供的继承结构。exception 对象几乎定义了所有异常情况。在Java程序中，可以使用try/catch关键字来处理异常情况； 如果在JSP页面中出现没有捕获到的异常，就会生成 exception 对象，并把 exception 对象传送到在page指令中设定的错误页面中，然后在错误页面中处理相应的 exception 对象。 |

### 四种作用域

|  |
| --- |
| 使用的对象 作用域名  ServletContext application域  HttpServletRequet request域  HttpSession session域  PageContext page域 |

## Jsp与servlet配合使用

### Jsp与servlet分工

* Servlet与jsp本质上是一个东西，但是在写法上各有优势，servlet中写java代码方便，jsp编写页面显示内容方便，所以一般会由servlet来负责代码的调用，由jsp负责数据的显示，那么会需要将servlet中处理的数据，传递到jsp上。

### 跳转方式

* 响应重定向

|  |
| --- |
| 响应重定向通过response对象调用，让浏览器向重定向的地址重新发送一次请求  response.sendRedirect("error.jsp"); |

注意：是因为重定向会发送新的请求，会导致第一次请求的相关数据在第二次请求中读取不到，如果我们需要这些数据可以使用转发的跳转方式。

* 请求转发

|  |
| --- |
| 请求转发通过request对象调用，把请求对象和响应对象与要跳转的地址共享  request.getRequestDispatcher("showIP.jsp").forward(request, response); |

* 两者区别

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **转发（较常用 效率更高）** | **重定向** | | 浏览器URL不变 | 浏览器URL的地址栏改变 | | 服务器行为（服务器内部） | 客户端行为（告诉客户端重新请求） | | 浏览器只做了一次访问请求 | 浏览器做了至少两次的访问请求的 | | 2次跳转之间传输的信息不会丢失 | 2次跳转之间传输的信息会丢失  （request范围） | |

|  |
| --- |
|  |

## El表达式

### El表达式作用

* EL即Expression Language（表达式语言）
* 页面中嵌入java代码的写法容易造成结构混乱，不易阅读和调试，可以使用一些其他技术来提高可阅读性和对原始使用方式有一定优化，如el表达式和jstl标签。
* El表达式的主要作用为从域对象或参数中取值并输出到页面中。

### El表达式特点

* 自动转换类型

EL得到某个数据时可以自动转换类型

对于类型的限制更加宽松

* 使用简单

相比较在JSP中嵌入Java代码，EL应用更简单

* EL的语法

以"${"作为开始，以"}"作为结束

### El表达式操作

|  |
| --- |
| request域中  request.setAttribute(“testuser”,myuser);  使用el表达式取域中user对象的username属性,并输出到页面  ${testuser.username}  操作符“.”  获取对象的属性，例如：${news.title}  操作符“[]”  获取集合中的对象，例如：${newsList[0]} |

El表达式中允许使用运算符

|  |
| --- |
|  |

### El表达式隐式对象

EL表达式包含一些隐式对象

* 作用域访问对象 可用来指定从某个域中取值 默认从上到下取值

pageScope

requestScope

sessionScope

applicationScope

* 参数访问对象 可指定从request的参数部分取值

param

paramValues

* Jsp配置对象

pageContext

## Jstl标签

### Jstl标签作用

* El表达式主要是取值和输出功能，需要配合jstl标签来完成逻辑控制（条件判断、循环）
* JSTL(JavaServerPages Standard Tag Library) jsp基本标签库，主要用来替换页面中跟逻辑判断相关的java代码，经常和el表达式配合使用

### Jstl使用

Jstl需要jar包支持

<http://archive.apache.org/dist/jakarta/taglibs/standard/binaries/>

下载jstl.jar和standard.jar包

在使用jstl的页面中，需要加入taglib指令，导入核心标签库

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

### Jstl中常用标签

通用标签

set：设置指定范围内的变量值

out：计算表达式并将结果输出显示

remove：删除指定范围内的变量

|  |
| --- |
| <c:set var= "example" value="${100+1}" scope="session" />  <c:out value="${example}"/>  <c:remove var= "example" scope="session"/> |

条件标签

If：条件判断语句

|  |
| --- |
| <c:if test="${empty loginUser }" var="islogin" scope="session">  <!--test：判断条件 可以将结果保存在域中 var是key scope是域 -->  <c:redirect url="testEL.jsp"></c:redirect><!--配合跳转标签使用 -->  </c:if>  <c:if test="${not islogin}">  <!--对之前保存的判断结果取反 完成else结构 -->  已登录  </c:if> |

迭代标签

forEach：遍历集合中的元素时使用

|  |
| --- |
| <c:forEach items="${listEitity }" var="myentity">  <!--items是遍历的集合 var是每次遍历到的对象 每次会存在域中 --> <tr><td>${myentity.ename }</td><td>${myentity.eage }</td><td>${myentity.eaddr }</td></tr>  <!--遍历中取值直接使用遍历到并存在域中的key，通过el表达式取值 -->  </c:forEach> |

### Jstl其他标签库

|  |
| --- |
|  |