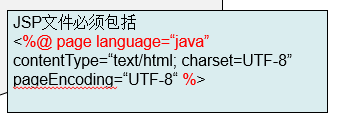
**Ch01 JSP简介**

1. jsp组成：HTML文件+Java片段+JSP标签
2. 
3. 编写、运行JSP项目：

打开Eclipse，新建Dynamic project“03-01”

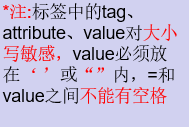
在WebContent目录下新建index.jsp文件(JSP文件与WEB-INF同级目录)

编写index.jsp文件

运行“03-01”项目

1. JSP页面元素：伪指令、注释、声明、动作、脚本、表达式
   1. 伪指令标签：<%@ %> 设置全局变量，声明类、要实现的方法及输出内容的类型
   2. 注释标签：JSP中显示<%-- --%>，JSP脚本元素// 、/\* \*/，JSP生成的HTML在客户端显示<!-- -->
   3. 声明标签：包括的是一段java代码<%! %>，<%! Int count=0;%>
   4. 动作标签：请求处理阶段向JSP容器发送指令；<jsp:include page=”header.html”/>

jsp:include，jsp:forward，jsp:useBean，jsp:getProperty，jsp:setProperty；



* 1. 脚本标签：镶嵌在JSP页面中的java代码，用于处理客户端的请求 <% %>
  2. 表达式标签：访问可用的java取值或其他表达式，给动作指令动态传递参数

表达式不能用分号结尾 <%= %>

<%= “hello world”%>，<% String name =“header.jsp”%>，

<jsp:include page=”<%=name%>”%>

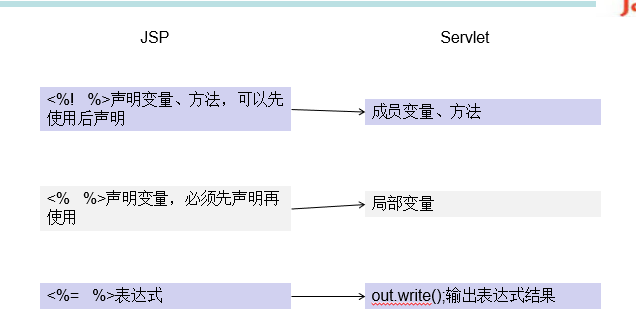
**Ch02JSP基础**

1. JSP的Servlet本质：JSP是嵌套java代码的HTML代码。JSP转换成对应的Servlet类运行。

Servlet是嵌套HTML代码的java代码。

1. JSP的工作原理：JSP引擎吧JSP文件转换成一个servlet，JSP引擎使用javac吧转换成的servlet源文件编译成相应的class文件
2. JSP生命周期：转译-编译-加载-实例化-初始化-服务-销毁
   * + - 1. \_jspInit()初始化
         2. \_jspService()服务
         3. \_jspDestory()销毁
3. JSP转换Servlet：声明标签-servlet类成员、脚本标签-\_jspService()中的代码

HTML代码、表达式标签-\_jspService()中的out.write()



1. Page伪指令属性：
   1. Import描述JSP页面中使用类的全包名，可选属性:<[%@page import=”java.io.\*,java.sql.\*”%](mailto:%25@page%20import=)>
   2. Session指定页面是否需要一个http会话:<%@page session=”true/false”%>
   3. errorPage和isErrorPage（jsp异常处理）JSP产生一个不能捕获的溢出时跳转到一个错误页面:<[%@page errorPage=”error.jsp”%](mailto:%25@page%20errorPage=)>

error.jsp中必须有<%@page isErrorPage=”true”%>才能进行错误处理

* 1. info指定jsp的描述信息:<%@page info=”information”%>
  2. language默认是java
  3. buffer和autoFlush，buffer指定jsp的输出缓存区最小值<%@page buffer=”32kb”%>

autoFlush满时是否将数据发送给客户端默认true，<%@page autoFlush=”false”%>

* 1. contentType和pageEncoding

**Ch03JSP进阶**

1. JSP内置对象：在\_jspService()中声明并初始化9个内置对象
   1. 内置对象out：Javax.servlet.jsp.JspWriter

向客户端输出数据，作用域：page，每个页面都有一个自己的out对象

方法：print()/println()/write()

* 1. 内置对象request：Javax.servlet.http.HttpServletRequest、

来自客户端的请求经过servlet处理后由request对象进行封装

作用域：request，此次请求结束后生命周期结束

方法：getParameter(key),getParameterValues(key),

getRequestDispatcher(“list.jsp”).forward(request,response),setAttribute(key,Object),getAttribute(key),setCharacterEncoding(“UTF-8”)

* 1. 内置对象response：Javax.servlet.http. HttpServletResponse

封装了jsp的响应，发送到客户端响应客户请求；作用域：page

方法：sendRedirect(“list.jsp”), setCharacterEncoding(“UTF-8”)

* 1. 内置对象session：Javax.servlet.http.HttpSession

一个会话，保存用户信息，跟踪每个用户状态，一段时间内客户端和服务器之间的一连串的相关的交互过程；作用域：session

方法：setMaxInactiveInterval(1000\*60)设置session失效时间，Invalidate()销毁当前会话，getId(),Isnew()

* 1. 内置对象application:javax.servlet.ServletContext

从servlet配置对象获得的servlet上下文；作用域：application

* 1. 内置对象pageContent：javax.servlet.jsp.PageContext

本jsp的页面上下文；作用域：page

* 1. 内置对象config：本jsp的ServletConfig，作用域：page
  2. 内置对象page：实现处理本页当前请求的类的实例，转换后的Servlet类本身
  3. 内置对象exception：本jsp页面的异常对象

1. JSP的作用域：Application>Session>Request>Page

**Ch04 web组件复用**

1. 静态包含：<%@include file=“地址”%>，一个file属性，相对路径
2. 动态包含：
   1. Include动作指令：<jsp:include page=“地址” flush=“true”>
   2. Forward动作指令：<jsp:forward page=“地址”/>
3. JavaBean：
   1. 属性最好定义为private，所有属性必须有setter和getter方法
   2. 必须有一个没有参数的公共构造方法
   3. <jsp:useBean>操作元素来加载JavaBean程序
      1. Scope作用域范围
      2. beanName创建实例类名

<jsp:setProperty>操作元素来设置JavaBean属性值

1. name JavaBean标识名称
2. property JavaBean属性名称

<jsp:getProperty>操作元素获取JavaBean的属性值

**Ch05表达式语言**

1. EL表达式语言：输出page、request、session、application范围的数据
2. 标志 ${}没有声明变量
3. 隐含对象：pageContent,

pageScope,requestScope,sessionScope,applicationScope,

param,paramValues,

header,headerValues,

cookie,

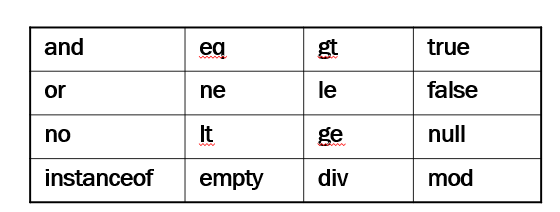
initParam

1. EL变量：存取变量数据${user}

自动搜索顺序page-request-session-application

指定范围：${pageScope.user}

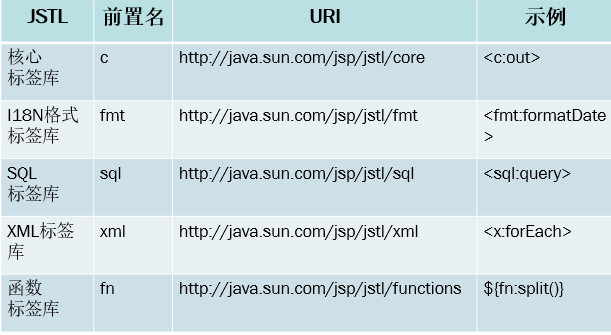
1. EL保留字：变量名不可以使用EL保留字



1. 关系与逻辑运算符：相等eq不相等ne，小于lt大于gt，小于等于le大于等于ge
2. 其他运算符：empty判断是否为null和空的
3. 自定义EL语言函数：
   1. 编写方法所在源文件 .java文件
   2. 编写标签库描述符 .tld文件
   3. 修改部署描述符 web.xml文件
   4. 编写JSP文件 .JSP文件

**Ch06标准标签库（JSTL）**

1. 安装JSTL：在JSP中使用taglib伪指令
2. 标签库集：



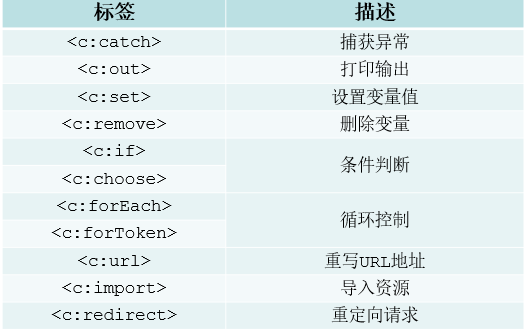
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>

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

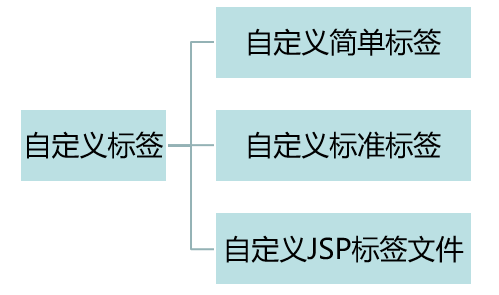
<%@ taglib prefix="x" uri="http://java.sun.com/jsp/jstl/xml" %>

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

JSTL核心标签库：



**Ch07自定义标签**



1 简单标签

1. 编写简单标签实现类
2. 编写TLD文件
3. 在web.xml中配置TLD文件
4. 在JSP页面中使用taglib
5. 标准标签
6. 编写标准标签实现类
   * 1. 实现标准接口
     2. 继承标准标签类
7. 编写TLD文件
8. 在web.xml中配置TLD文件
9. 在JSP页面中使用taglib
10. 标签文件：以.tag或.tags为后缀，JSP元素不能在标签文件中使用

<%@taglib prefix=“前缀” tagdir=“标签文件地址”%>来导入标签文件所在文件夹

**Ch08安全**

认证，授权，数据完整性，数据私密性，审核，恶意代码，网站攻击