# Ch01\_JSP简介

1. Java动态网页技术标准(Java Server Pages)
2. JSP组成：HTML文件+Java片断+JSP标签
3. JSP页面元素：
4. 伪指令标签 <%@ %> 设置全局变量，声明类、要实现的方法以及输出内容的类型等
5. 注释标签 <%--注释--%> 增强代码的可读性
6. 声明标签 <%! %> java代码 例：<%! int count =0;%>
7. 脚本标签 <% %> 镶嵌在JSP页面中的java代码，主要用于处理客户端的请求
8. 动作标签 <jsp:actionName/> 在请求处理阶段向JSP容器发送指令
9. 表达式标签 <%= %> JSP提供的一种简单方法用于访问可用的Java取值或其他表达式，还可用于给动作指令动态传递参数 例:<%=“hello world”%>

# Ch02\_JSP基础

1. JSP的Servlet本质：
2. JSP：JSP是嵌套Java代码的HTML代码，JSP转换成对应的Servlet类运行
3. Servlet：Servlet是嵌套HTML代码的Java代码
4. JSP页面生命周期：转译、编译、加载、实例化、初始化、服务、销毁
5. JSP与Servlet的内部转换：JSP生命周期第一阶段
6. page伪指令属性
7. import属性：相当于java中的import，例：<%@page import=“java.io.\*,java.sql.\*”%>
8. session属性：session属性指定页面是否需要一个http会话，例：<%@page session=“true/false”%>
9. errorPage属性:JSP异常处理,例：<%@page errorPage=“error.jsp”%>
10. isErrorPage属性：，在error.jsp页面中必须加入<%@page isErrorPage=“true”%>才能进行错误处理
11. info属性：用于指定JSP的描述信息，例：<%@page info=“information”%>
12. language属性：指定JSP页面中声明、脚本、和表达式中使用语言，默认是Java，
13. extends属性：extends属性用于指定JSP页面对应Servlet类的基类，极少使用，
    * 1. buffer属性：指定JSP的输出缓存区的最小值，例：<%@page buffer=“32kb” %>
14. autoFlush属性：指定当缓存区满时，是否自动将数据发送给客户端，默认为true
15. contentType属性：指定页面响应的MIME类型和字符编码，例：<%@page contentType =“text/html;charset=UTF-8” %>
16. pageEncoding属性：指定JSP页面字符编码格式，默认值为ISO-8859-1

# Ch03\_JSP进阶

1. JSP内置对象 重要方法见PPT
2. Out：

作用：主要用来向客户端输出数据

作用域：page，每个页面都有一个自己的out对象

1. Request：

作用：封装经过Servlet容器处理的来自客户端的请求

作用域：request，这次请求结束后，它的生命周期就结束了

1. Response：

作用：封装JSP 的响应，被发送到客户端以响应客户的请求。

作用域：page

1. Session

作用：表示一个会话，用来保存用户信息，以便跟踪每个用户的状态。

作用域：session

1. Application

描述：从servlet配置对象获得的servlet上下文

作用域：application

1. PageContext

描述：本JSP的页面上下文

作用域：page

1. Config

作用：本JSP的 ServletConfig

作用域：page

1. Page

作用：实现处理本页当前请求的类的实例，转换后的Servlet类本身

作用域：page

1. Exception

作用：本JSP页面的异常对象

作用域：page

1. JSP的作用域
2. Application：作用域范围最大

可以被不同页面、不同用户共享

可以通过application内置对象或ServletContext访问Application作用域对象

1. Session：仅次于Application作用域

可以被同一个客户端的请求访问

可以通过session内置对象访问Session作用域对象

1. Request：小于Session作用域

可以被forward()方法转向页面或者include()包含页面访问

可以通过request内置对象访问request作用域对象

1. Page：范围最小

只能被创建对象的页面访问

可以通过pageContext内置对象访问Page作用域对象

# Ch04\_Web组件复用

1. 静态包含：

JSP中的include伪指令实现静态包含，include标准语法：<%@include file=“地址”%>，file属性值不能向被包含的文件传递参数

1. 动态包含：

客户端请求时，将请求转发给当前JSP文件包含的其他对象

作用域： request、session、application

1. include动作指令

允许当前页面将客户端请求暂时转交给所包含的对象，一旦对象执行完毕，返回当前JSP页面

标准语法：<jsp:include page=“地址” flush=“true”>

1. forward动作指令

允许将客户请求转发到另一个资源文件，即每当遇到此动作指令就停止执行当前的JSP，转而执行被转发的指定资源。

标准语法：<jsp:forward page=“地址”/>

区别：forward动作指令在调用转发操作后不会继续处理指令后的任何代码

include动作指令在调用转发操作后，完成转发操作后将继续处理指令后的代码

动态指令传参：<jsp:param name=“” value=“”/>可以传递一个或多个参数给动态页面

1. 使用JavaBean组件

定义：通过封装属性和方法成为具有某种功能或者处理某个业务的对象，简称Bean

命名规范

满足条件：必须有一个没有参数的公共构造方法；所有的属性必须有setter和getter方法，属性本身最好定义为私有的，已实现JavaBean的封装性。

使用：动作指令见PPT

1. <jsp:useBean>操作元素来加载JavaBean程序
2. <jsp:setProperty>操作元素来设置JavaBean的属性值
3. <jsp:getProperty>操作元素获取JavaBean的属性值

# Ch05\_表达式语言（EL

1. 表达式语言简介
2. EL(Expression Language): 表达式语言
3. JSP表达式的代码：标志<% %>，有变量声明<%! %>

The outside temperature is<%= temp%> degrees

1. 表达式语言(EL)表达式的代码：标志${ }，没有变量声明

The outside temperature is ${temp} degrees

1. 11个隐含对象：

与范围有关的：applicationScope:sessionScope:requestScope:pageScope:

与输入有关的：param:paramValues:

其他隐含对象：cookie:header:headerValues:initParam:pageContext

1. 默认的范围查找顺序(自动搜索顺序)

Page🡪 Request🡪 Session🡪 Application

也可以指定范围：${pageScope.user}

1. 表达式语言运算符
2. 属性/集合访问运算符

属性访问运算符.

例如a.b代表a对象的名为b的属性：${header.host}

集合访问运算符[]

例如a[b]代表集合a的关键字为b所指向的元素：${header[“host”]}

区别：①当要存取的属性名称中包含一些特殊字符，象”.”或”-”等并非字母或数字的

符号，一定要用[]，如${user.My-Name}应写为${user[“My-Name”]}

②如想动态的取值时，也要使用[]，如${user[data]}，data相当于一个变量

1. 算数运算符
2. 逻辑运算符
3. 关系运算符
4. 表达式语言函数

# Ch06\_标准标签库（JSTL

JSP标准标签库(JSP Standard Tag Library)是一个实现 Web 应用程序中常见的通用功能的定制标记库集

功能包括迭代和条件判断、数据管理格式化、XML 操作以及数据库访问

1. JSTL标签库安装

在JSP中使用taglib伪指令导入JSTL标签库

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

导入JSTL核心标签库，通过<c:XXX>可以使用JSTL核心标签库标签

1. 一般用途的JSTL标签
2. <c:catch>捕获JSP页面抛出的异常

例：<c:catch var=“e”> action that might throw an exception</c:catch>

1. <c:out>打印输出内容到页面

类似JSP表达式

属性value包含特殊字符，使用转义字符

属性name变量没有初始化，使用default属性指定默认值

1. 属性用途的JSTL标签
2. <c:set>标签设置变量值或对象的属性值

<c:set var=“num” value=“${4\*4}” />

<c:set target=“user”property=“userName”>${“susan”}</c:set>

1. <c:remove>用于从指定作用域内删除变量

<c:remove var=“num” scope=“session”/>

1. 控制用途的JSTL标签
2. <c:if>与Java语言的if语句最大的不同是没有else语句对应标签

<c:if test=“${x==9}”>${x}</c:if>

1. <c:choose>类似Java的switch-case语句。

该标签只当做<c:when>和<c:otherwise>的父标签

1. <c:forEach>和<c:forTokens>类似于Java的for语句

<c:forEach>使用begin、end、step属性指定循环次数

对于集合和字符串使用items属性指定循环对象

<c:forEach>提供varStatus属性，用来存放现在指到的成员的信息。另外四个属性：

index：现在指到成员的索引

count：总共指到成员的总数

first：现在指到的成员是否为第一个成员

last：现在指到的成员是否为最后一个成员

1. 其他种类标签
2. 处理URL相关的标签

<c:url>用来产生一个URL，“/”表示上下文路径

<c:import>可以把其他静态或动态文件包含至本身JSP页面。与JSP动作<jsp:include>的不同在于<c:import>可以包含位于Web程序之外的资源

<c:redirect>用于把客户请求重定向到另一个资源

1. JSTL格式化标签：格式化并输出文本、日期、时间、数字。

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

1. JSTL SQL标签库提供了与关系型数据库进行交互的标签。

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

1. JSTL XML标签库提供了创建和操作XML文档的标签。

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

1. JSTL包含一系列标准函数，大部分是通用的字符串处理函数。

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

# Ch07\_自定义标签

1. 定制标签简介
2. 空标签体的定制标签

<prefix:tagName></prefix:tagName>&<mytag:required></mytag:required>

1. 带属性的定制标签

<前缀:标签名 属性1=“值1”属性2=“值2”/> &<test:function user="john" num="100"/>

1. 步骤 具体实现见PPT

编写标签实现类

编写TLD文件

在web.xml中配置TLD文件

在JSP页面中使用taglib

1. 简单标签简介及使用

简单标签提供了SimpleTag接口和SimpleTagSupport实现类

1. 标准标签简介及使用
2. JSP中使用标签文件
3. 标签文件是由JSP代码组成的，以.tag或.tags为后缀，标签文件可以包含表达式、JSP伪指令、标准标签和自定义标签
4. 在JSP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
5. 标签文件的伪指令

variable伪指令

声明JSP变量<%@variable name-given=“x”%>

tag伪指令

对整个文件进行设置，类似page指令

attribute伪指令

指定属性为静态属性

fragment属性为true允许静态属性中插入JSP代码

1. 标签文件的动作指令

标签文件包含了JSP中一系列的动作指令

jsp:invoke指令：使attribute伪指令中的fragment生效

jsp:doBody指令：用于处理标签内容体

# Ch08\_安全

1. 基本概念

认证、授权、数据完整性、数据私密性、审核、恶意代码、网站攻击

1. 认证机制

Servlet定义的4种认证机制：HTTP基本认证、HTTP摘要认证、HTTPS客户端认证、

HTTP表单认证

1. 安全声明
2. 安全编程

HttpServletRequest接口提供了3个方法用于识别用户和角色

1. String getRemoteUser()

如果用户通过认证，该方法返回用户的登录名称。否则返回null值

1. Principal getUserPrincipal()

该方法返回一个包含认证通过用户的java.security.Principal对象

1. Boolean isUserInRole(String rolename)

该方法用于判断用户是否是指定的角色

# Ch09\_Servlet程序结构与部署

1. JavaEE应用程序结构-
2. JavaEE应用程序的组成:Servlet、JSP、工具类、jar包、HTML页面
3. 部署描述符：XML格式的文档
4. 部署JavaEE应用

# Ch10\_部署描述符

1. 部署描述符文件的作用
2. 常用元素的声明与配置
3. Icon：包含small-icon和large-icon元素，为大型和小型GIF或JPEG图标图片指定文件名
4. display-name：定义应用的名称。
5. Description：对应用做出描述
6. context-param：应用范围内初始化参数
7. Filter：声明了Web应用程序中的过滤器
8. filter-mapping：容器使用filter-mapping决定哪个过滤器以什么样的顺序应用到请求
9. listener：指定事件监听程序
10. servlet：用于声明一个servlet
11. servlet-mapping：定义了servlet和URL模式之间的映射
12. session-config：控制会话超时
13. mime-mapping：关联文件与MIME类型
14. welcome-file-list：指定欢迎页
15. error-page：指定错误页面
16. jsp-config：提供Web应用程序中的JSP文件的全局配置信息

提供安全性：

1. 指定验证的方法：<login-config>
2. 限制对web资源的访问：<security-constraint>
3. 分配角色名：<security-role>
4. JavaEE元素
5. <distributable/>：支持集群的服务器可安全的在多个服务器上分布Web应用
6. <resource-env-ref>：声明一个与某个资源有关的管理对象
7. <resource-ref>：声明一个资源引用

# Ch11\_JavaMail

1. 电子邮件(Email)

邮件传输协议

1. SMTP：简单邮件传输协议，应用层协议
2. POP：邮局协议
3. JavaMail

主要组件：Session、Message、Address、Authenticator、Transport

# Ch12\_文件的上传和下载

1. 表单元素
2. <input>标签向服务器端提交数据，常用类型有9种类型

text：文本域 password：密码文本域

radio：单选按钮 checkbox：多选按钮

file：文件上传 hidden：隐藏域

reset：重置按钮 submit：提交按钮

button：按钮

1. <textarea>
2. <select>
3. 文件上传
4. 文件下载

暂不需要

# Ch13\_分页

1. JavaWeb程序中访问数据库

JDBC数据库连接技术

1. 加载数据库的驱动程序
2. 建立数据库连接
3. 执行数据库操作SQL
4. 关闭数据库连接
5. 对数据的分页技术的实现
6. 将数据一次性取出，缓存在内存中，当用户从页面发出查看特定页面的请求时，将内存中缓存的数据根据请求返回给客户

效率高，内存占用大

1. 当用户从页面发出查看特定页面的请求，会根据请求向数据库中取出一页的数据，每次请求都进行一次数据库检索的操作

效率低，节省内存

# Ch14\_jQuery

1. jQuery的简介
2. JavaScript 工具库，让 HTML 文档遍历和操作、事件处理、动画和 Ajax 操作更加简单。
3. jQuery作用

快速获取文档元素 提供漂亮的页面动态效果

创建AJAX无刷新网页 提供对JavaScript语言的增强

增强的事件处理 更改网页内容

1. jQuery的安装
2. 掌握jQuery基础
3. 语法：通过选取 HTML 元素，并对选取的元素执行某些操作。基础语法：

$(selector).action()，例：$("p").hide() - 隐藏所有 <p> 元素

1. 选择器：基于元素的 id、类、类型、属性、属性值等"查找"（或选择）HTML 元素。

元素选择器： $("p") Id选择器:$("#test")

class选择器：$(".test"） 其他选择器

1. 事件：事件处理程序指的是当 HTML 中发生某些事件时所调用的方法。

$("p").click();

$("p").click(function(){

// 动作触发后执行的代码!!

});

1. jQuery的技术应用 具体代码见PPT
2. jQuery效果：隐藏、显示、切换，滑动，淡入淡出，以及动画
3. jQuery HTML：jQuery 拥有可操作 HTML 元素和属性的强大方法
4. jQuery 遍历：用于根据其相对于其他元素的关系来"查找"（或选取）

HTML 元素

1. jQuery Ajax：AJAX 是与服务器交换数据的技术，它在不重载全部页面的情况下，实现了对部分网页的更新。
2. jQuery 插件
3. 整合其他框架

# Ch15\_ajax

1. AJAX的简介
2. Asynchronous JavaScript and XML（异步的 JavaScript和XML）
3. 一种创建交互式[网页](https://baike.baidu.com/item/%E7%BD%91%E9%A1%B5/99347)应用的网页开发技术
4. AJAX 是一种在无需重新加载整个网页的情况下，能够更新部分网页的技术。
5. AJAX适用场景

表单驱动的交互 文本输入场景（需要保证数据的唯一性）

深层次的树的导航 用户间的交流响应（最新的热点新闻，天气预报等）

类似投票等场景 过滤场景（按照时间和名称排序等等）

1. AJAX工作原理
2. 传统AJAX的使用
3. 创建 XMLHttpRequest 对象
4. 发送请求到服务器
5. 监听响应
6. 处理响应
7. jQuery AJAX

不刷新浏览器的情况下从服务器加载数据。

1. load() 方法从服务器加载数据，并把返回的数据放入被选元素中。

语法：$(selector).load(URL,data,callback);

1. jQuery get() 和 post() 方法用于通过 HTTP GET 或 POST 请求从服务器请求数据。

$.get(*URL*,*callback*);

$.post(*URL,data,callback*);

# Ch16\_日志

1. 日志的使用场景
2. 常用框架

Log4j1，Log4j2，Commons Logging，Slf4j，Logback，Jul( java.util.logging)等

1. logback实现记录日志 具体步骤见PPT
2. 日志组件

Loggers：日志的记录器，Logger 负责捕捉事件，把它关联到应用的对应的context

上后，主要用于存放日志对象，并将其发送给合适的 Appender。

Appenders ：也被称为 Handlers，负责将日志事件记录到目标位置，指定日志输

出的目的地

Layouts ：也被称为 Formatters，它负责对日志事件中的数据进行转换和格式化。

1. 日志级别：DEBUG、INFO、WARN 和 ERROR

# Ch17\_自定义注解

1. 常见的注解
2. @Override用于标识该方法继承自超类
3. @Deprecated 该类或者该方法已经不推荐使用，已经过期了，，会生成编译的警告。
4. @SuppressWarnings用于忽略的编译器警告信息
5. @WebServlet
6. Spring中大量的注解
7. Java验证的注解@NotNull @Email
8. 注解简介及作用
9. 简介：注解是一种元数据形式。

注解属于java的一种数据类型，类似类、接口、数组、枚举。

注解用来修饰，类、方法、变量、参数、包。

注解不会对所修饰的代码产生直接的影响。

1. 作用：生成文档。

跟踪代码依赖性，实现替代配置文件功能。

在编译时进行格式检查。如@override 放在方法前，如果你这个方

法并不是覆盖了超类方法，则编译时就能检查出。

类属性自动赋值

1. 元注解

@Documented – 注解是否将包含在JavaDoc中

@Retention – 什么时候使用该注解

@Target – 注解用于什么地方

@Inherited – 是否允许子类继承该注解

1. 自定义注解

自定义注解格式：public @interface 注解名{ 定义体}

注解内容的语法格式：数据类型 属性名() default 默认值

# Ch18\_加密

1. 加密的概念和分类
2. 常见的加密算法及使用场景 具体使用见PPT

Base64、MD5、DES、AES、PBE、RSA

# Ch19\_缓存

1. 缓存简介
2. 缓存的分类
3. 操作系统磁盘缓存->减少磁盘机械操作
4. 数据库缓存->减少文件系统I/O
5. 应用程序缓存->减少对数据库的查询
6. Web服务器缓存->减少应用服务器请求
7. 客户端浏览器缓存->减少对网站的访问
8. 本地缓存
9. 缓存的实体（值、创建时间、过期时间）
10. 管理缓存（添加、删除）（设置、获取）
11. 缓存测试（缓存中没有从数据库或其他数据源取数据，有的话从本地数据源取数据）
12. 远程缓存

缓存框架ehcache