1. JSP
2. 概述

Java Server Pages，Java服务器端页面。其实就是一个特殊的页面，既可以定义html标签，又可以定义java代码。非常方便，用于简化书写。

1. 原理

浏览器请求.jsp文件，服务器解析请求消息，寻找是否有.jsp文件

如果找到了.jsp文件，则将.jsp文件转换为.java文件。

编译.java文件，生成.calss字节码文件

由字节码文件提供访问，其实该字节码文件本质上是一个Servlet。

1. JSP的脚本
2. <% 代码 % >：定义的Java代码，在service方法中。service方法中可以些什么，则该代码就可以写什么。
3. <%! 代码 %>：定义的java代码，在jsp转换后的java类的成员位置，即成员变量。
4. <%= 代码 %>：定义的java代码，会输出到页面上。即输出语句，该语句定义在service方法中。
5. JSP的内置对象
6. 概述

在.jsp页面中不需要获取和创建的对象，可以直接使用。如request, response, out…

1. out对象

字符输出流对象，可以将数据输出到页面上，和response.getWriter()类似。

response.getWriter()和out.write()的区别

在Tomcat服务器给客户端作出响应之前，会先在response缓冲区中查找数据，再到out缓冲区中查找数据。一般都用out，而不用response.getWriter()。

1. pageContext

真实类型为PageContext，当前页面共享数据，还可以获取其他的内置对象

1. request

真实类型为HttpServletRequest，一次请求访问的多个资源共享

1. session

真实类型为HttpSession，一次会话的多个请求间共享数据

1. application

真实类型为ServletContext，在所有的用户间共享数据。

1. response

真实类型为HttpServletResponse，是响应对象

1. page

真实类型为Object，当前页面(Servlet)的对象。

1. config

真是类型为ServletConfig，用于配置Servlet的配置对象。

1. exception

真实类型为Throwable，是异常对象。

1. JSP的指令
2. 作用

JSP指令用于配置JSP页面，导入资源文件。

1. 格式

<%@ 指令名称 属性名1=属性值1 属性名2=属性值2 %>

多个属性名和属性值间用空格隔开。

1. 指令名称分类
2. page

用于配置JSP页面。

属性名：

contentType:

设置响应体的mime类型以及字符集

设置当前JSP页面的编码

import:

导包

errorPage:

当前页面发生异常后，会自动跳转到指定的错误页面

isErrosPage:

标识当前页面是否是错误页面。若是，则可以使用内置对象Exception

1. include

导入页面的资源文件

如：

<%@ include file=”a.jsp” %>

1. taglib

导入资源

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

prefix:自定义的前缀名

1. JSP的注释
2. html注释

JSP页面中可以使用html的注释，但是该注释只能注释html代码片段

<!-- -->

1. JSP注释

JSP特有的注释，可以注释JSP页面中的所有内容

<%--- --%>

1. Session
2. 概述

服务器端的会话技术，在一次会话中的多次请求间共享数据，将数据保存在服务器端的对象中。

1. 使用
2. 获取HttpSession对象

HttpSession session = request.getSession();

1. 使用HttpSession对象

获取参数：

Object getAttribute(String name)

设置参数：

void setAttribute(String name, Object vlaue)

删除参数：

void removeAttribute(String name)

1. Session原理

Sesssion是依赖于Cookie的，Session是存储与服务器中的，那么当浏览器第一次请求时，服务器中没有Session，则会在服务器中创建一个新的Session对象，该对象有响应的id。

服务器通过response响应浏览器的请求时，会设置一个set-cookie响应头，存储的值为Session对象的id（JSESSIONID）。

浏览器获取到该响应头后，就可以将该Cookie存储在浏览器中，下一次请求时将该JSESSIONID键值对设置在cookie请求头中发送给服务器。

服务器收到该请求后，通过cookie查找对应JSESSIONID的Session对象。这就保证了在一次会话中多次请求的Session对象是相同的。

1. 注意事项
2. 客户端关闭，服务器不关闭，两次获取的Session是否为同一个？

否。如果想要相同的Session，则再创建一个Cookie，键为JSESSION，设置最大存活时间，让Cookie持久化保存。

1. 客户端不关闭，服务器关闭，两次获取的Session是同一个吗？

否。这样容易导致数据的丢失。要使两次获取的Session是同一个，就要做到Session的钝化与活化：

Session的钝化：

在服务器正常关闭之前，将Session对象序列化到硬盘上

Session的活化：

在服务器启动后，将Session文件转化为内存中的Session对象。

1. Session的销毁
2. 服务器关闭，则Session销毁
3. Session对象调用 invalidate()
4. Session默认失效时间为30分钟

可以在web.xml中修改：

<session-config>

<session-timeout>30</session-timeout>

</session-config>

1. Session的特点
2. Session用于存储一次会话的多次请求的数据，存在于服务器端
3. Session可以存储任意类型，任意大小的数据，不同于Cookie的限制。
4. Session数据安全，Cookie相对不太安全
5. MVC
6. 概述

M：Model，模型

V：View，视图

C: Controller，控制器

1. 原理

控制器通过调用模型，进行业务操作，模型将数据返回给控制器，控制器再将数据交给视图进行展示。

Java Web中使用Servlet充当控制器，使用JavaBean作为模型，使用JSP作为视图。

1. MVC的分工
2. Mode

模型使用JavaBean，完成具体的业务操作，如：查询数据库，封装对象

1. View

视图使用JSP来展示数据。

1. Controller

控制器使用Servlet来获取用户的输入，调用模型，接受模型的返回数据，将数据交给视图进行展示。

1. MVC的优劣
2. 优点

耦合性低，方便维护，利于分工合作，且代码的重用性高

1. 缺点

项目架构变得更加复杂。

1. EL表达式
2. 概述

Expression Language，表达式语言。用于替换和简化JSP页面中的Java代码的编写。

1. EL语法

${表达式}

1. 注意事项

JSP默认支持EL表达式，若想要忽略EL表达式，则设置JSP中的page指令，isELIgnored=”true”，就可以忽略所有的EL表达式了。

若想要忽略单个EL表达式，则在EL表达式前加个\即可：

\${表达式}

1. EL的使用
2. 运算

算数运算符，比较运算符，逻辑运算符，空运算符。

其他运算符用法都大同小异，重点介绍空运算符：empty

${empty list}：判断字符串，集合，数组对象是否为null并且长度是否为0

${not empty list}：判断字符串，集合，数组对象是否不为null，并且长度>0

1. 获取值

（1）获取方式

EL表达式只能从域对象中获取值。

（2）获取语法

${域名称.键名}：从指定域中获取指定键的值。

${键名}：依次从最小的域中查找是否有该键对应的值，直到找到为止。

${域名称.键名.属性名}：获取对象中的属性值，本质上调用对象的getter方法

${域名称.键名[索引]}：获取List集合中的值

${域名称.键名.key名称}：获取Map集合中的值

${域名称.键名[“key名称”]}：获取Map集合中的值

域名称：

pageScope (pageContext)

requestScope (request)

sessionScope (session)

applicationScope (application)

（3）举例

在request域中存储了name=hh，要想通过EL表达式获取该键值，则通过以下方式获取：

${requestScope.name}

1. JSTL
2. 概述

JavaServer Pages Tag Library，JSP标准标签库，是由Apache组织提供的开源的jsp标签

1. 作用

JSTL用于简化和替换jsp页面上的java代码

1. 使用步骤
2. 导入JSTL相关的jar包
3. 引入标签库：taglib指令： <%@ taglib %>
4. 使用标签
5. 常用的JSTL标签
6. if

相当于Java代码中的if语句。

标签中的属性：

test:接受boolean表达式，如果表达式为true，则显示if标签体内容；否则，不显示标签体内容。一般情况下，test属性值会结合EL表达式一起使用，在test属性中设置boolean表达式：

<c:if test=”${number % 2 == 0} ”>

a

</c:if>

1. choose

相当于Java代码中的switch语句。

使用choose标签声明，相当于switch声明

使用when标签作判断，相当于case

使用otherwise标签作其他情况的声明，相当于default

1. foreach

相当于Java代码中的for语句。

1. 完成重复的操作

设置一些属性：

begin:开始值

end:结束值

var:临时变量

step:步长

varStatus:循环状态对象

index:容器中元素的索引，从0开始

count:循环次数，从1开始

如：

<c:forEach begin=”1” end=”10” var=”i” step=”2” varStatus=”s”>

${i} ${s.index} ${s.count}

</c:forEach>

1. 遍历容器

属性：

items:容器对象

var:容器中元素的临时变量

varStatus:循环状态对象

index:容器中元素的索引，从0开始

count:循环次数，从1开始

如：

<c:forEach items=”${list}” var=”str” varStatus=”s”>

${s.index} ${s.count}

</c:forEach>

1. 三层架构
2. 概述
3. 表示层

用户通过界面上的组件和服务器进行交互

1. 业务逻辑层

该层用于处理业务逻辑

1. 数据访问层

该层用于操作数据存储文件

1. 原理
2. 浏览器发出请求后，表示层接受了用户的参数，表示层中的控制器（Servlet）调用了业务逻辑层进行处理。
3. 业务逻辑层组合了数据访问层中的简单方法，形成了复杂业务逻辑操作。
4. 数据访问层对数据库进行操作，返回数据给业务逻辑层。
5. 数据访问层将数据封装，返回给业务逻辑层。
6. 业务逻辑层将数据再传递给表示层的控制器(Servlet)，控制器将数据发送给视图（JSP），视图再将数据发送给浏览器进行展示。
7. 三层架构的作用
8. 表示层

表示层接受用户的参数，封装数据，控制器（Servlet）调用业务逻辑层完成处理，之后接受到返回的数据后转发视图（JSP）页面完成显示。

1. 业务逻辑层（service层）

组合数据访问层中的简单数据库操作方法，形成复杂的业务逻辑操作。

1. 数据访问层（dao层）

定义了对数据库的最基本的CRUD操作。

1. 三层标准包名
2. 表示层

cn.xx.项目名.web

1. 业务逻辑层

cn.xx.项目名.service

1. 数据访问层

cn.xx.项目名.dao

1. 信息列表展示实例
2. 需求分析

该实例是对用户信息的增删改查操作。

1. 设计
2. 技术选型

Servlet + JSP + MySQL + JDBCTemplate + Duird + BeanUtils + Tomcat

1. 数据库设计

create database info;

use info;

create table user(

id int primary key auto\_increment,

name varchar(20) not null,

gender varchar(5),

age int,

address varchar(32),

qq varchar(20),

email varchar(50)

);

1. 开发
2. 环境搭建

创建数据库环境

创建项目，导入需要的jar包

1. 查询所有用户信息实现流程
2. 浏览器发出请求到表示层的控制器（UserListServlet）中，该控制器调用service层的findAll()方法，返回List<User>集合。
3. service层中设置有UserService接口和UserServiceImpl实现类，在接口中有一个findAll()方法，实现类的findAll()方法调用了dao层的findAll()方法。
4. dao层设置有UserDao接口和UserDaoImpl实现类。接口中有findAll()方法，实现类中的findAll()方法用JDBC对数据库进行操作。
5. 数据返回到表示层中，数据的格式为List<User>，将list集合存入request域中。
6. list.jsp页面通过JSTL +EL技术用foreach标签遍历list集合生成表格table。
7. 转发list.jsp页面给浏览器进行展示。
8. 项目结构
9. src目录下存放所有的Java代码，包括web层，service层，dao层，工具类utils包，对象domain包，以及druid的配置文件druid.properties
10. web目录下存放所有的前端页面，所有的.html, .js, .jsp文件。以及WEB-INF目录，存放项目的依赖Jar包