|  |
| --- |
| 第五阶段课程内容:   1. javascript 2. jquery 3. bootstrap 更多是自学 4. servlet 5. jsp 6. 第三方组件/上传组件,图形插件(柱状图，饼图,折线图等 js echar),连接池 |
| Servlet:   1. 什么是servlet? Servlet什么作用?   Servlet（Server Applet）是Java Servlet的简称，称为小服务程序或服务连接器，用Java编写的服务器端程序，主要功能在于交互式地浏览和修改数据，生成动态Web内容。(java服务编写的服务器程序，生成动态网页的内容)  2)servlet是一个web 应用程序开发的接口(规范化—JAVAEE企业级规范),servlet是一JAVAEE一种规范。 JDBC也是一种规范  <1>web应用程序----基于bs模式的应用程序.  B/S模式 : 基于客户端(browser浏览器) 服务器Server,  优点:数据共享，不需要开发客户端(也就是说不需要安装客户端软件)  C/S模式: C client客户端(需要开发一个客户端) S Server QQ软件  //客户端维护成本，开发成本，   1. tomcat服务器容器(软件)   tomcat如何解决数据传输的问题?/封装(不需要程序员自己去解析发送过来的数据，以及响应返回浏览器)  DB数据库  request  Tomcat服务器  servlet  浏览器C(客户端)  servlet  response  思考问题?  网络编程?  数据传输问题?怎么传?  协议:计算机通讯  双方的一种约定.  Tcp/ip协议 编程  B 北京  3.jpg 10KB  A 上海  3.jpg 10KB  Tomcat软件目录结构:   1. 下载 jdk1.7, tomcat7.x 版本要对应   目录结构  /bin：存放windows或Linux平台上启动和关闭Tomcat的脚本文件  测试tomcat 启动成功:  <http://localhost:8088/> 出一只猫的界面.  有可能会出现闪退？  端口:地址: 65335个端口  一个软件都对应有一个端口号:  /conf：存放Tomcat服务器的各种全局配置文件，其中最重要的是server.xml和web.xml  /doc：存放Tomcat文档  /server：包含三个子目录：classes、lib和webapps  /server/lib：存放Tomcat服务器所需的各种JAR文件  /server/webapps：存放Tomcat自带的两个WEB应用admin应用和 manager应用  /common/lib：存放Tomcat服务器以及所有web应用都可以访问的jar文件  /shared/lib：存放所有web应用都可以访问的jar文件（但是不能被Tomcat服务器访问）  /logs：存放Tomcat执行时的日志文件  /src：存放Tomcat的源代码  /webapps：Tomcat的主要Web发布目录，默认情况下把Web应用文件放于此目录  //部署项目  /work：存放JSP编译后产生的class文件  Eclipse集成tomcat : 1)真正的tomcat使用位置?  F:\iotek0206\web\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\webapps  Servlet本质就是一个类/接口 (多线程) ==  一定要运行在tomcat容器之上的.=🡺Http协议本质是对tcp/ip协议的扩展.  http协议：html/xml  tcp/ip协议:只传字节流  servlet的作用就是用于接收客户端的请求信息，及响应客户端的信息, 担当：生成动态的网页.   1. 创建servlet的方式有二种: 2. 继承Servlet接口   只能由service方法进行服务处理;   1. HttpServlet类   doGet/doPost方式进行服务处理   1. servlet的生命周期: (servlet本质是一个类. )   <1>什么时候加载,创建 ：第一次请求时，实例化servlet对象,驻留在内存里.  <2>初始化 init() 方法 只执行一次  <3>服务处理 doGet --不停反复地执行的，处理请求与响应。  <4>销毁 destroy 服务器tomcat停止  <load-on-startup>1</load-on-startup> 启动加载时，实例化servlet, 数值越小，先加载实例化.  Life.do?name=请求  Web.xml  Tomcat  响应  doGet/doPost  DB  经过处理  Servlet对象  Servlet对象  IE |
| Servlet API(标准接口):   1. HttpServletRequest对象 简称: request对象 ：封装请求信息.   常用方法:  System.out.println(request.getServerName()); //localhost  System.out.println(request.getServerPort()); //8080  System.out.println(request.getScheme()); //http  System.out.println(request.getServletContext()); //上下文对象  System.out.println(request.getServletPath()); //servlet路径 /request.do  System.out.println(request.getContextPath()); //上下文路径/根路径;  /\*  System.out.println(request.getLocalAddr()); //客户端  System.out.println(request.getLocalName()); //客户端;    System.out.println(request.getRemoteAddr()); //客户端的IP地址  \*/  /\*  System.out.println(request.getMethod()); //获取请求方式    System.out.println(request.getQueryString());//获取请求参数列表信息  \*/    //解析请求头信息;  /\*  Enumeration<String> enums=request.getHeaderNames(); //获取请求头信息的键名 比如 host  while(enums.hasMoreElements()){  String name= enums.nextElement();  System.out.println("name="+name);    String value= request.getHeader(name);  System.out.println("value="+value);  }  \*/  Jdbc+servlet+html完成注册功能:   1. 导入jar包 ---- web-inf/lib ：工作目录tomcat下 2. 编写html页面 3. 编写servlet 4. Servlet调用service/--dao 5. servletConfig对象:获取servlet的初始化参数信息的   tomcat  ServletConfig对象  servletA  ServletConfig对象sss  servletB |
| 1. ServletContext对象 ---简称:application对象: 获取上下文环境对象：(理解为：在整个服务器运行间期都存在的，全局共享对象)   Tomcat 服务器  ServletContext对象  servletC  servletB  ServletA |
| 4)HttpSession对象: 简称为session对象: 保存客户端与服务器端的会话的信息。  Tomcat Session保存服务器  Session对象  Session对象  Session对象  浏览器  浏览器 北京  浏览器 上海  1)session对象的创建 与生命周期:  HttpSession session=request.getSession(); //实例化session;    session的生命周期: 默认配置文件30分钟  session生命周期计算怎么算?  客户端最后一次访问服务器开始计算?  web.xml ctrl+f session  <session-config>  <session-timeout>30</session-timeout>  </session-config>  2.硬编码：  session.setMaxInactiveInterval(10); //session生命周期10秒  注意:session对象是保存在服务器端，客户端关闭，session不没有销毁的，真的销毁的时间是session的设置生命周期时间范围之外.  乱码处理的问题:  :总：所有的编码要统一  没有设置输出类型，默认text/plain //原始字符串输出   1. 输出乱码   解决方案:  response.setContentType("text/html;charset=utf-8");   1. 请求的乱码request获取客户端参数的乱码   二种情况:  <1>post请求产生乱码的情况?  解决方案:  request.setCharacterEncoding("utf-8");  <2>get请求产生乱码情况 iso8859-1 解决方案1:  String userName=request.getParameter("username");  String un=new String(userName.getBytes("iso8859-1"),"utf-8");  解决方案2:  Server.xml配置 URIEncoding=utf=8  <Connector connectionTimeout="20000" port="8080" protocol="HTTP/1.1" redirectPort="8443" URIEncoding="utf-8"/>  注意不是通用的，只有tomcat服务器才有这个解决方案，其它服务器没有这样子的配置。 Weblogic服务器不支持  Cookie对象：保存客户端(浏览器)  特点: 不占用服务器资源(保存在本地浏览器)  Cookie可以被清除掉,浏览器禁用cookie  不安全.  Cookie的使用.  浏览商品历史记录  七天免登录(保存本地)  购物车使用cookie hadoop(非关系数据库)  Cookie对象 cookie  Response.addCookie(cookie)  浏览器 cookie  Cookie操作存取  Cookie 实现七天免登录功能:   1. 选中免登录复选框时，用户名与密码存在cookie中. 2. 下次登录就不需取数据库的值，取cookie的值与数据库中的数据进行比较。   服务器跳转/客户端跳转区别?  1）服务器跳转---内部跳转(转发)  1)url地址不会发生改变  2)可以共享request对象的信息  request.getRequestDispatcher("bb.do").forward(request, response);  3)不能跨域跳转  2)客户端跳转(重定向) //二次请向  1)url地址会发生改变  2)不能共享request对象的信息  3)支持跨域跳转  Servlet 路径问题  404表示路径问题，  500 表示服务器程序内部错误  JSP JAVA SERVER PAGE  Jsp本质就是一个servlet  Servlet 工作原理  Class文件  编译器转换成 .java文件  Jsp文件  public final class test\_jsp extends org.apache.jasper.runtime.HttpJspBase🡪HttpServlet  servlet 的优点:业务判断，流程控制..  JSP的作用：解决servlet 页面数据显示的问题。  Jsp页面的构成:   1. jsp相关指令 2. jsp相关的标签 3. html/css/javascript 4. 嵌入java 代码块   Jsp的脚本有三种方式:  1)<%! %> 声明部分  2)<% %> java块  3)<%=%> 输出表达式  Jsp三种指令:  指令:是发给JSP编译器的信息，告诉jsp编译器如何处理jsp页面，它不直接生成输出.   1. page指令 2. include指令 3. taglib指令   语法: <%@page {属性=值.......}%>  Page指令:  contentType=”text/html;charset=utf-8” 表示设置响应类型  language=”java”  pageEncoding="UTF-8" 表示页面编码设置  import="java.util.\*"  session="true" 默认是true session对象，  errorPage="error.jsp" 表示错误的页面的URL  isErrorPage="true" ,开启exception对象的.  表示可以处理错误的信息  buffer="16kb" 设置输出缓存区的大小.  isELIgnored="true" 设置true 不是 el表达式  include指令:表示插入在其它文件中所包含的文本.  静态包含: 先包含内容进来，再一起编译生成.java..  Include标签:  动态包含: 先分开编译，会产生二个.java文件,二个.class文件  org.apache.jasper.runtime.JspRuntimeLibrary.include(request, response, "includeD.jsp", out, false);  forward标签: 转发（服务端跳转—内部跳转）  JSP 注释:  <!-- 🡪  <%-- --%>  javaBean组件:  组件: 可重用的执行高级任务的程序部件. Api的集合  JSP 组件: JavaBean  EJB 组件  可视化组件 JButtion 按钮  非可视化组件 javaBean   1. 什么是JavaBean   遵循一定的设计原则的任何JAVA类都可以是Javabean组件. Javabean的本质就是一个类.   1. 可序列化 2. 无参数的构造方法 3. 类的私有属性 private 4. 通过公有的方法获取和设置属性 getter/setter 方法   JSP 的动作标签  Jsp的九大内置对象（隐式对象）  Request  Response  Session  Application  Config  Exception  pageContext 表示当前页面有效。工具类：获取其它八大内置对象;  page  out |
| EL表达式:Expression Language  基于jstl1.0的一种为方便获取数据的语言.  获取隐式对象数据;  格式: ${EL表达式语言}   1. El表达式基本类型:   布尔类型  数值  字符串  Null值   1. 算术运算符 + - \* / 或div 取余mod % 2. 关系运算符 == eq   != ne  < lt  >Gt  <= le  >= ge  4)逻辑运算符 && || ! and or not  5)empty运算符 可以用来判断值是NULL，空字符串  6) EL的隐式对象  pageScope pageContext对象  requestScope request  sessionScope sessioni  applicationScope application   1. 与请求有关的隐式对象   Param paramValues  Header headerValues   1. initParam 获取初始化参数的信息   cookie 获取cookie的值.   1. pageContext—页面上下文稿对象   <H1>获取头信息</H1>  <%=request.getHeader("Host") %>  ${header['host'] }    <%=request.getContextPath() %>  ${pageContext.request.contextPath }  ${pageContext.request.method }  ${pageContext.session.id }    <h1>获取上下文环境参数</h1>  <%=application.getInitParameter("cs") %>  ${initParam.cs}    <h1>获取cookie</h1>  <%  Cookie c=new Cookie("username","liayin");  c.setPath(request.getContextPath());  c.setMaxAge(60\*60);  response.addCookie(c);    %>  <%  Cookie[] cookies=request.getCookies();  for(Cookie ck:cookies){  out.println(ck.getName()+" "+ck.getValue());  }  %>  ${cookie.username.name }/${cookie.username.value }  JSTL标签库: JSP Standard Tag Library jsp标准标签库:  为什么要使用jstl？  JSTL标签的使用格式与HTML标签类似.  不必做重复工作  兼容性好  供应商可能提供更优化的实现  可移植性好  核心标签:  导入jar包  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>   1. <c:out> 输出 2. <c:set> 设置 3. <c:if> <c:choose> <c:when> <c:otherwise> 4. <c:forEach>   Servlet中的值转到页面，然后页面通过$()表达式取值  JSP+servlet+db  Mvc设计模式: Model View Controller 1982年  最桌面程序中:  Model 模型 (业务组件)---业务程序核心，比如数据库记录列表/javabean/ejb  VIEW 视图 (用户界面---页面) ---显示数据(jsp)  Controller 控制器(servlet) /输入输出 :将M和V的实现代码进行分离, M改变，V应该同步更新.  DBUtils工具类 :封装jdbc操作的工具类  一个servlet处理多个请求?  方法有二种?  Bootstrap框架编写页面  Bootstrap=🡺  下次课讲 el /jstl标签/ 上传 案例 周未安排 bootstrap/项目需求.  下次课讲 上传/dbutils工具类/  下次课 /过滤器/监听器/json/ajax  //jquery |
| 服务器资源: (servlet,jsp,css,images图片,html)  过滤器Filter接口:是处理客户端与服务器资源文件之间的一道过滤网,在访问资源文件之前，通过一系列的过虑器对请求进行修改，判断，把不符合规则的请求在途中拦截，或修改，也可以对响应进行过滤，拦截.  first.do(request)  Filter过滤器(执行过滤==放行 )  执行过滤器之前  目标资源first.do  (servlet,jsp,html,css,image图片)  浏览器  过滤器之后  过滤器的作用：过滤(拦截)一组请求。  <filter-mapping>  <filter-name>first</filter-name>  <url-pattern>\*.jsp</url-pattern>  </filter-mapping>  <filter-mapping>  <filter-name>first</filter-name>  <url-pattern>/user/\*</url-pattern>  </filter-mapping>  拦截资源的配置  多个过滤器：顺序问题? <Filter-mapping>配置顺序有关.  监听器:listener   1. 监听对象划分有三种监听器:   ServletContextListener 上下文监听器 application  ServletContextAttributeListener 监听application属性操作  SessionListener session  SessionAttributeListener  RequestListener request  RequesttAttributeListener   |  |  |  | | --- | --- | --- | | ip | 次数 |  | | 127.0.0.1 | 3 |  | | 192.168.1.22 | 4 |  | | 192.168.1.23 | 5 |  | | 192.168.1.24 | 1 |  |   下次课: ajax, json,jquery, |