<%@ **page** language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  
<html>  
<body>  
<form action="sendMyName" method="post" >  
 <table>  
 <tr>  
 <td>用户名：</td>  
 <td><input type="text" name="user"/></td>  
 </tr>  
 <tr>  
 <td><input type="submit" value="提交"/></td>  
 </tr>  
 </table>  
</form>>  
</body>  
</html>



JSP表单页面代码分析与实验心得

核心内容分析

1.表单结构优化

使用 <table> 布局表单元素，提升视觉对齐（用户名标签与输入框水平排列）。

表单指定 action="sendMyName"，明确提交目标为名为 sendMyName 的 Servlet。

保留 method="post" 确保数据传输安全。

2.输入字段设计

用户名：文本输入框 name="user"，后端通过 request.getParameter("user") 获取值。

移除了冗余字段（对比前一版本），聚焦核心功能。

3.字符编码与标准

保持 contentType="text/html; charset=UTF-8" + pageEncoding="UTF-8"，支持中文输入。

4.提交机制

单字段提交简化用户操作，<input type="submit"> 触发 POST 请求。

实验心得：在 Java Servlet 中使用 Session

1. 技术实践

Session 创建与存储：

在 Servlet 中通过 request.getSession() 获取或创建 session，存储用户名：

java

HttpSession session = request.getSession();

session.setAttribute("username", request.getParameter("user"));

Session 生命周期管理：

默认有效期 30 分钟（可通过 web.xml 配置 session-timeout）。

主动销毁：session.invalidate()。

跨请求数据传递：

Session 存储的数据在同一用户会话中跨请求共享（如页面跳转后仍可读取）。

2. 优点

状态保持：解决 HTTP 无状态问题，实现用户登录态维护。

数据安全：Session ID 通过 Cookie 或 URL 重写传递，敏感数据存于服务端。

便捷存取：通过键值对 (setAttribute/getAttribute) 快速操作数据。

1. 改进

增加会话验证：

在 Servlet 中检查用户名的有效性（如非空、长度限制）：

java

String user = request.getParameter("user");

if (user == null || user.trim().isEmpty()) {

response.sendRedirect("error.jsp"); // 跳转错误页

return;

}

添加 Session 超时处理：

在 JSP 中检查 session 是否有效：

jsp

<% if (session.getAttribute("username") == null) { %>

<script>alert("会话已过期"); location.href="login.jsp";</script>

<% } %>

安全性增强：

退出时主动销毁 session：<a href="logout?action=invalidate">退出</a>。

防止 Session 固定攻击：session=request.getSession(true) 创建新 session。

4. 学习价值

理解会话机制：

掌握 Session 与 Cookie 的协作原理（如 Session ID 传递）。

用户状态管理：

为登录系统、购物车等场景提供技术基础。

安全实践：

学习会话劫持防护、敏感数据存储规范。