## 实验四 Servlet容器模型

一、实验目的

1. 掌握Web应用程序的DD文件web.xml的配置方法和作用；

2. 掌握如何通过ServletConfig获取Servlet的初始化参数；

3. 掌握ServletContext的概念，如何获得ServletContext对象；

4. 掌握如何通过RequestDispatcher转发请求。

二、实验原理

Web应用程序有严格定义的目录结构，它需要一个称为部署描述文件的文件web.xml来配置。对应用程序的每个Servlet，可以在web.xml中定义若干个初始化参数，然后通过ServletConfig对象获取这些参数。

ServletContext对象表示该应用程序的环境，通过该对象可以获得应用程序范围的初始化参数、容器版本信息、资源的URL、在其中存放共享数据以及登录日志等。

三、实验内容与步骤

（一）建立一个名为bookstore的Web应用程序

在Tomcat的webapps目录下，建立一个如下所示的目录结构：

|- bookstore

|- html (包含所有的HTML文件)

|- jsp (包含所有的JSP文件)

|- images (包含所有的GIF、JPEG和BMP图象文件)

|- index.html (默认的HTML文件)

|- WEB-INF

|-classes

├ lib

├ web.xml

图4.1 bookstore应用程序的目录结构

这样就建立了一个名为bookstore的Web应用程序，本实验后面的题目都在该应用程序下面完成。

（二）使用ServletConfig检索Servlet的初始化参数

【步骤1】编写ConfigDemoServlet程序，内容如下：

package com.demo;

import java.io.\*;

import java.sql.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class ConfigDemoServlet extends HttpServlet{

String servletName=null;

ServletConfig config=null;

String email=null;

String telephone=null;

public void init() {

config = **getServletConfig();**

servletName = config.getServletName();

email = **config.getInitParameter("email");**

telephone = **config.getInitParameter("telephone");**

}

public void service(HttpServletRequest request, HttpServletResponse response)

throws ServletException,IOException{

response.setContentType("text/html;charset=gb2312");

PrintWriter out = response.getWriter();

out.println("<html><body>");

out.println("Servlet名称："+servletName+"<br>");

out.println("Email地址："+email+"<br>");

out.println("电话："+telephone);

out.println("</body></html>");

}

}

【步骤2】修改web.xml文件，在其中添加下面Servlet定义和初始化参数定义

<servlet>

<servlet-name>configDemoServlet</servlet-name>

<servlet-class>com.demo.ConfigDemoServlet</servlet-class>

<init-param>

<param-name>**email**</param-name>

<param-value>**hacker@163.com**</param-value>

</init-param>

<init-param>

<param-name>**telephone**</param-name>

<param-value>**8899123**</param-value>

</init-param>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>configDemoServlet</servlet-name>

<url-pattern>/configDemo.do</url-pattern>

</servlet-mapping>

【步骤3】在浏览器的地址栏中输入下面URL访问该Servlet：

http://localhost:8080/bookstore/configDemo.do

程序运行结果如下图所示：



ConfigDemoServlet程序的运行结果

（三）ServletContext的使用

修改上面程序，将email地址和电话使用ServletContext初始化参数传递给ConfigDemoServlet。使用ServletContext对象将email地址和电话信息写入日志文件中，并查看日志文件。

（五）下面是一个Servlet，它可以获得有关服务器的信息。

package com.demo;

import java.io.\*;

import java.util.\*;

import javax.servlet.\*;

public class ServerInfo extends GenericServlet {

public void service(ServletRequest request, ServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html;charset=GB2312");

PrintWriter out = response.getWriter();

ServletContext sc = getServletContext();

out.println("<html>");

out.println("<body>");

out.println("<head>");

out.println("<title>获取服务器端信息</title>");

out.println("</head>");

out.println("<body>");

out.println("<h2>服务器端信息：</h2>");

out.println("<table width=\"500\" border=\"1\">");

out.println("<tr>");

out.println("<td width=\"175\">站点名</td>");

out.println("<td width=\"325\">" + request.getServerName() + "</td>");

out.println("</tr>");

out.println("<tr>");

out.println("<td>端口号</td>");

out.println("<td>" + request.getServerPort() + "</td>");

out.println("</tr>");

out.println("<tr>");

out.println("<td>服务器类型</td>");

out.println("<td>" + sc.getServerInfo() + "</td>");

out.println("</tr>");

out.println("<tr>");

out.println("<td>支持Servlet版本</td>");

out.println("<td>"+sc.getMajorVersion()+"."+sc.getMinorVersion()+"</td>");

out.println("</tr>");

out.println("<tr>");

out.println("<td>服务器属性</td>");

out.println("<td>");

// 获得服务器属性集合

Enumeration attributes = sc.getAttributeNames();

while (attributes.hasMoreElements()) {

String name = (String)attributes.nextElement();

out.println(name);

}

out.println("</td>");

out.println("</tr>");

out.println("</table>");

out.println("</body>");

out.println("</html>");

}

}

（六）简单的综合应用

【步骤1】创建一个名为input.html的HTML页面，其中包括一个表单，表单中包含两个文本域，分别供用户输入学号和姓名，该页面也包含提交和重置按钮。

【步骤2】定义一个名为com.demo.Student类，其中包括学号sno和姓名name两个private的成员变量，定义访问和修改sno和name的方法。

【步骤3】编写名为FirstServlet的Servlet，要求当用户在input.html中输入信息后点击“提交”按钮，请求FirstServlet对其处理。在FirstServlet中使用表单传递的参数（学号和姓名）创建一个Student对象并将其作为属性存储在ServletContext对象中，然后获得通过ServletContext的getRequestDispatcher()方法获得RequestDispatcher()对象，将请求转发到SecondServlet。

【步骤4】在SecondServlet中取出ServletContext上存储的Student对象，并显示输出该学生的学号和姓名。在SecondServlet的输出中应该包含一个超链接，点击该连接可以返回input.html页面。



input.html页面显示结果



SecondServlet显示结果

四、思考题

1. 如何通过ServletConfig获得Servlet参数？

2. 如何获得ServletContext对象，如何通过该对象获得上下文参数？

3. 如何使用ServletContext共享数据？