实习二 **Servlet** 技术

# 实习要求

学习 Serlvet 的工作机制及编写方法学习 Filter 的工作机制及编写方法学习 Servlet 生命周期事件

# 实验目的

掌握 Servlet 的工作机制及编写方法掌握 Filterde 工作机制及编写方法了解 Servlet 生命周期事件

掌握数据库访问方法

掌握 Cookie 对象的使用

# 环境要求

服务器：Tomcat6.0 或更高、mysql5 或更高集成开发环境：MyEclipse 2014 或更高

# 实验内容

**1**、**Servlet**

(1) 第一个 Servlet

第一步：创建一个 Web 工程，名称为 servletDemo。

第二步：通过向导创建 Servlet，点击菜单“File”->“New”->“Servlet”打开向导如图 1。在“Name” 中输入类名“MyServlet”，并作选项如图 1；然后点击“Next”按钮显示第二步如图 2，选择复选

框“Generate/Map web.xml”，并点击“Finish”按钮完成创建。第三步：查看 Eclipse 生产的 servlet 配置。

打开“WebRoot”→“WEB-INF”→“web.xml”，有如下内容：

Servlet的注册

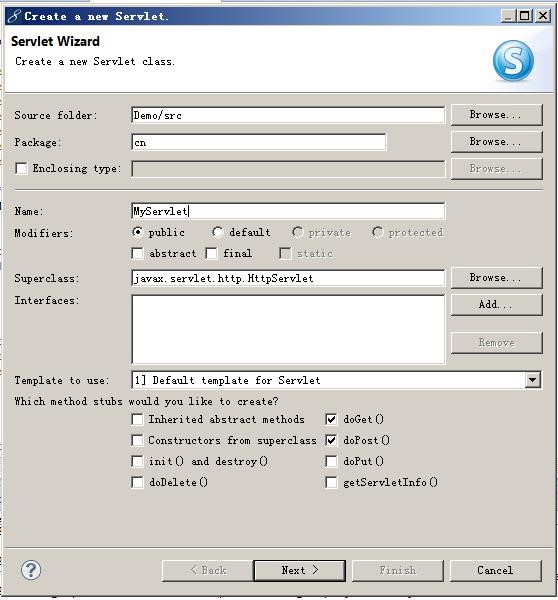


图 1 Servlet 向导 1

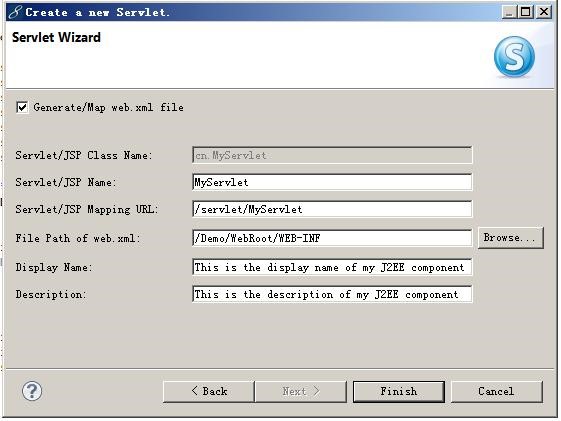


图 2 Servlet 向导 2

<servlet>

<description></description>

<display-name>MyServlet</display-name>

<servlet-name>MyServlet</servlet-name> servlet的引用名

<servlet-class>cn.MyServlet</servlet-class> 具体的类

</servlet>

Servlet的URL映射

<servlet-mapping> <servlet-name>MyServlet</servlet-name> 上面servlet的引用名

<url-pattern>/MyServlet</url-pattern> 访问servlet的url地址

</servlet-mapping>

第四步：编写代码

打开刚建好的 MyServlet 类，找到 **protected void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException

方法，在改变其中代码如下：

response.setCharacterEncoding("UTF-8"); response.getWriter().println("<h1>我的第一个Servlet！！</h1>");

并 在 **protected void** doPost(HttpServletRequest request,

HttpServletResponse response) **throws** ServletException, IOException

方法中其中代码如下：

doGet(request,response);

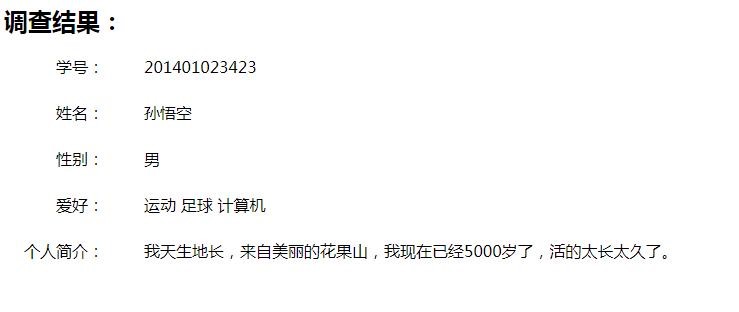
然后保存。第五步：部署并访问该工程部署到 Tomcat 中，并启动。然后浏览器的地址栏中输入：

http://localhost:8080/servletDemo/MyServlet

1. 通过 **Servlet** 进行表单处理创建表单页面 form.jsp，界面如图 3，点击“确定”按钮后将表单内容提交到 check（servlet）处理提交结果，然后将处理结果交由 result.jsp 页面显示如图 4。



图 3 表单 form.jsp 页面



图

4

结果显示页面

result.jsp

***form.jsp***

*<%@ page language="java" import="java.util.\*" pageEncoding="ISO-8859-1"%>  
<%  
String path = request.getContextPath();  
String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";  
%>  
  
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">  
<html>  
  <head>  
    <base href="<%=basePath%>">  
      
    <title>My JSP &apos;form.jsp&apos; starting page</title>  
      
<meta http-equiv="pragma" content="no-cache">  
<meta http-equiv="cache-control" content="no-cache">  
<meta http-equiv="expires" content="0">      
<meta http-equiv="keywords" content="keyword1,keyword2,keyword3">  
<meta http-equiv="description" content="This is my page">  
<!--  
<link rel="stylesheet" type="text/css" href="styles.css">  
-->  
  
  </head>  
    
  <body>  
    <form action="check" method="post">  
         <fieldset>  
            <legend align="left">个人爱好调查</legend><br>  
           &emsp;&emsp;学号：<input name="studentnu" type="text"/><br><br>  
           &emsp;&emsp;姓名：<input name="uname" type="text"/><br><br>  
           &emsp;&emsp;性别：<input name="sex" value="男" type="radio"/>男   
                 <input name="sex" value="女" type="radio"/>女<br><br>  
           &emsp;&emsp;爱好：<input type="checkbox" name="hobs" value="文学">文学  
                 <input type="checkbox" name="hobs" value="运动">运动  
                 <input type="checkbox" name="hobs" value="旅游">旅游  
                 <input type="checkbox" name="hobs" value="足球">足球  
                 <input type="checkbox" name="hobs" value="计算机">计算机  
                 <input type="checkbox" name="hobs" value="网游">网游<br><br>  
           &emsp;&emsp;个人简介：<br>  
           &emsp;&emsp;&emsp;&emsp;<textarea rows="4" cols="50" name="resume"></textarea><br>  
           &emsp;&emsp;&emsp;&emsp;&emsp;<input type="submit" value="提交" >  
         </fieldset>  
      </form>  
  </body>  
</html>*

***result.jsp***

*<%@ page language="java" import="java.util.\*" pageEncoding="ISO-8859-1"%>  
<%  
String path = request.getContextPath();  
String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";  
%>  
  
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">  
<html>  
  <head>  
    <base href="<%=basePath%>">  
      
    <title>My JSP &apos;result.jsp&apos; starting page</title>  
      
<meta http-equiv="pragma" content="no-cache">  
<meta http-equiv="cache-control" content="no-cache">  
<meta http-equiv="expires" content="0">      
<meta http-equiv="keywords" content="keyword1,keyword2,keyword3">  
<meta http-equiv="description" content="This is my page">  
<!--  
<link rel="stylesheet" type="text/css" href="styles.css">  
-->  
  
  </head>  
    
  <body>  
     <% request.setCharacterEncoding("UTF-8");   
    String[] hobs=request.getParameterValues("hobs");  
    String hobStr="";  
        for(String hob:hobs){  
        hobStr+=hob+" ";  
        }%>  
    <fieldset>  
    <h2>调查结果</h2>  
    &emsp;&emsp;学号：<%=request.getParameter("studentnu")%><br /><br>  
    &emsp;&emsp;姓名：<%=request.getParameter("uname") %><br/><br>  
    &emsp;&emsp;性别：<%=request.getParameter("sex")%><br/><br>  
    &emsp;&emsp;爱好：<%=hobStr %><br/><br>  
       个人简历：<%=request.getParameter("resume")%><br/><br>  
    </fieldset>  
  </body>  
</html>*

***check.java***

*package servletDemo;  
  
import java.io.IOException;  
import java.io.PrintWriter;  
  
import javax.servlet.ServletException;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
  
public class check extends HttpServlet {  
  
/\*\*  
 \* Constructor of the object.  
 \*/  
public check() {  
super();  
}  
  
/\*\*  
 \* Destruction of the servlet. <br>  
 \*/  
public void destroy() {  
super.destroy(); // Just puts "destroy" string in log  
// Put your code here  
}  
  
/\*\*  
 \* The doGet method of the servlet. <br>  
 \*  
 \* This method is called when a form has its tag value method equals to get.  
 \*   
 \* @param request the request send by the client to the server  
 \* @param response the response send by the server to the client  
 \* @throws ServletException if an error occurred  
 \* @throws IOException if an error occurred  
 \*/  
public void doGet(HttpServletRequest request, HttpServletResponse response)  
throws ServletException, IOException {  
response.setContentType("text/html;charset=UTF-8");  
        response.setCharacterEncoding("UTF-8");  
        String studentnu=request.getParameter("studentnu");  
        String name=request.getParameter("uname");  
        String sex=request.getParameter("sex");  
        String hobs=request.getParameter("hobs");  
        String resume=request.getParameter("resume");  
  
response.setContentType("text/html");  
PrintWriter out = response.getWriter();  
out.println("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\">");  
out.println("<HTML>");  
out.println("  <HEAD><TITLE>A Servlet</TITLE></HEAD>");  
out.println("  <BODY>");  
out.print("    This is ");  
out.print(this.getClass());  
out.println(", using the GET method");  
out.println("  </BODY>");  
out.println("</HTML>");  
out.flush();  
out.close();  
}  
  
/\*\*  
 \* The doPost method of the servlet. <br>  
 \*  
 \* This method is called when a form has its tag value method equals to post.  
 \*   
 \* @param request the request send by the client to the server  
 \* @param response the response send by the server to the client  
 \* @throws ServletException if an error occurred  
 \* @throws IOException if an error occurred  
 \*/  
public void doPost(HttpServletRequest request, HttpServletResponse response)  
throws ServletException, IOException {  
request.getRequestDispatcher( "result.jsp").forward(request, response);  
  
response.setContentType("text/html");  
PrintWriter out = response.getWriter();  
out.println("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\">");  
out.println("<HTML>");  
out.println("  <HEAD><TITLE>A Servlet</TITLE></HEAD>");  
out.println("  <BODY>");  
out.print("    This is ");  
out.print(this.getClass());  
out.println(", using the POST method");  
out.println("  </BODY>");  
out.println("</HTML>");  
out.flush();  
out.close();  
}  
  
/\*\*  
 \* Initialization of the servlet. <br>  
 \*  
 \* @throws ServletException if an error occurs  
 \*/  
public void init() throws ServletException {  
// Put your code here  
}  
  
}*

1. **Cookie** 的使用创建登录页面 login.jsp 界面如图 5，在执行 login.jsp 时，首先查看 Cookie 中是否保存有用户名和口令，若有则将保存信息填充到表单中的用户名和口令中，如没有则不填充。当点击“登录”按钮时，将表单提交给 check（Servlet）进行处理，若用户名和口令正确则将正确的用户名和口令保存到 Cookie 中以备下次登录时（login.jsp）自动填充表单。

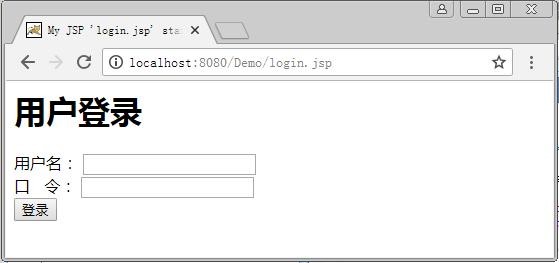


图 5 登录页面 login.jsp

示例代码：

login.jsp

<%@ page language=*"java"* import=*"javax.servlet.http.Cookie"* pageEncoding=*"UTF-8"*%>

<!DOCTYPE>

<html>

<head>

<title>Login</title>

</head>

<body>

<%

Cookie[] cks = request.getCookies();

String uname=""; String pwd=""; **for**(Cookie ck:cks){ **if**(ck.getName().equals("uname")){ uname = ck.getValue();

} **if**(ck.getName().equals("passwd")){ pwd = ck.getValue();

}

}

%>

<form action=*"check"* method=*"post"*>

<h1>用户登录</h1>

<label>用户名：

<input name=*"uname"* value=*"*<%=uname%>*"*/>

</label><br />

<label>口&nbsp;&nbsp; 令：

<input name=*"passwd"* type=*"password"* value=*"*<%=pwd%>*"*/>

</label><br />

<input type=*"submit"* value=*"*登录*"* />

</form>

</body>

</html>

## Check.java

@WebServlet("/check") **public class** Check **extends** HttpServlet { **public void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

Cookie ck = **new** Cookie("uname",request.getParameter("uname")); response.addCookie(ck); ck = **new** Cookie("passwd",request.getParameter("passwd")); response.addCookie(ck); response.sendRedirect("login.jsp");

}

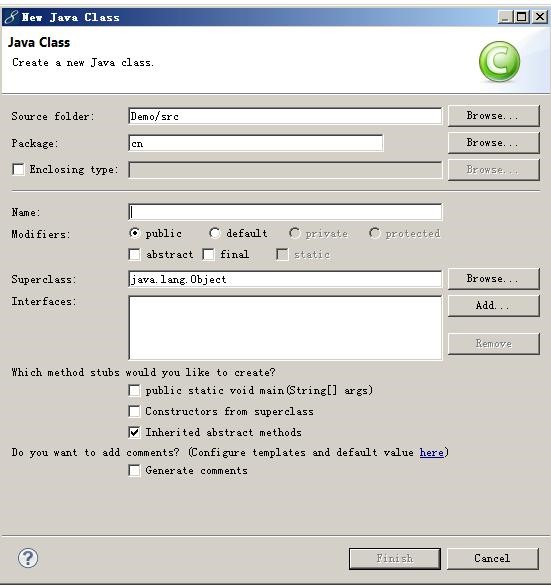
**public void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException { doGet(request, response);

}

}

如果用正常的web浏览，服务端关闭之后，并不能记录用户的信息，而通过cookie的使用，就能将客户端的信息记录下来，但是只能将名字记下来，对应的密码不能自动填充。（如何将密码填充到对应的账号？）

2、Filter——过滤器

（1）第一个过滤器创建一个过滤器，对所有的请求进行过滤，并在控制台上打印被

请求的 URI。创建过滤器

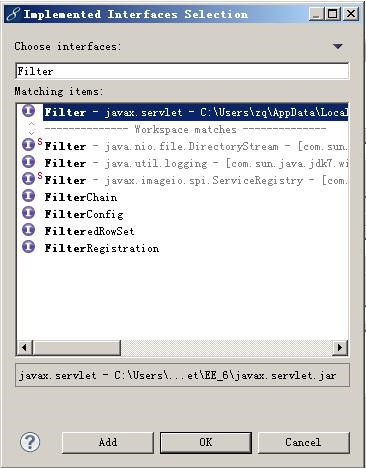
点击菜单“File”->“New”-> “Class”打开类创建向导对话框如图 6。点击对话框中的“Add”按钮添加要实现的接口，弹出如图 7 对

话框。在“Choose Interfaces”文本库中输入“Filter”进行筛选，在筛

选 结 果 列 表 中 选 择

“Filter--javax.servlet”，然后点击

“OK”按钮添加 Filter 接口道图 6 图 6 类创建向导

的“Interfaces”列表中，并在“Name” 中输入类名“FilterDemo”，最后点击“Finish”按钮创建 FilterDemo 类。配置过滤器

打开 web.xml，在标签</web-app>之前添加如下内容：

<!-- 过滤器注册 -->

<filter>

<filter-name>filterDemo</filter-name>

<filter-class>cn.FilterDemo</filter-class>

</filter>

<!-- 过滤器映射 -->

<filter-mapping>

<filter-name>filterDemo</filter-name>

<url-pattern>/\*</url-pattern>

图 7 添加实现接口对话框

</filter-mapping>

编写代码

**public class** FilterDemo **implements** Filter {

@Override **public void** destroy() {}

@Override **public void** doFilter(ServletRequest arg0, ServletResponse arg1,

FilterChain arg2) **throws** IOException, ServletException {

HttpServletRequest req = (HttpServletRequest)arg0;

String uri = req.getRequestURI();

System.*out*.println("URI===" + uri);//在控制台上输出 arg2.doFilter(arg0, arg1);

} @Override **public void** init(FilterConfig arg0) **throws** ServletException {}

}

测试重启服务器后，在浏览器中访问任何一个资源，然后在控制台上查看输出。



过滤器主要存在于服务端和客户端之间，截取客户端和请求 目标资源之间的请求和响应，并对请求/响应信息进行预处理或过滤，起到选择的作用，而本监听器则是捕捉了每次运行的URL

（**2**）使用过滤器保护指定资源在 WebRoot 中创建一个目录 protect 对中的内容进行访问保护，只有登录用户才能访问。若访问 protect 目录中的内容时用户没有登录则跳转到登录页面（login.jsp）。参考教材 P76 应用示例完成。

**Myservlet.java**

***package cn;***

***import java.io.IOException;***

***import javax.servlet.ServletException;***

***import javax.servlet.http.HttpServlet;***

***import javax.servlet.http.HttpServletRequest;***

***import javax.servlet.http.HttpServletResponse;***

***import javax.servlet.http.HttpSession;***

***public class MyServlet extends HttpServlet {***

***public void doGet(HttpServletRequest request, HttpServletResponse response)***

***throws ServletException, IOException {***

***String uname=request.getParameter("uname");***

***String upassword=request.getParameter("passwd");***

***HttpSession session=request.getSession(true);***

***boolean checkOK=false;***

***if(uname!=null&&upassword != null){***

***checkOK=true;***

***session.setAttribute("userinfo",uname);***

***}***

***if(checkOK){***

***response.sendRedirect("protect/MyHtml.html");***

***}else{***

***response.sendRedirect("login.jsp");***

***}***

***}***

***public void doPost(HttpServletRequest request, HttpServletResponse response)***

***throws ServletException, IOException {***

***doGet(request,response);***

***}***

***}***

**tempFilter.java**

***package cn;***

***import java.io.IOException;***

***import javax.servlet.Filter;***

***import javax.servlet.FilterChain;***

***import javax.servlet.FilterConfig;***

***import javax.servlet.ServletException;***

***import javax.servlet.ServletRequest;***

***import javax.servlet.ServletResponse;***

***import javax.servlet.http.HttpServletRequest;***

***import javax.servlet.http.HttpServletResponse;***

***import javax.servlet.http.HttpSession;***

***public class tempFilter implements Filter {***

***public void destroy() {}***

***public void doFilter(ServletRequest arg0, ServletResponse arg1, FilterChain chain)***

***throws IOException, ServletException {***

***HttpServletRequest request = (HttpServletRequest)arg0;***

***HttpServletResponse response = (HttpServletResponse)arg1;***

***HttpSession session=request.getSession(true);***

***Object tmp=session.getAttribute("userinfo");***

***if(tmp==null){***

***response.sendRedirect("../login.jsp");***

***}else{***

***chain.doFilter(request,response);***

***}***

***}***

***public void init(FilterConfig arg0) throws ServletException {}***

***}***

**login.jsp**

***<%@ page language="java" import="java.util.\*" pageEncoding="UTF-8"%>***

***<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">***

***<html>***

***<head>***

***<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">***

***<title>系统登录</title>***

***</head>***

***<body>***

***<h2 align="center">系统登录</h2>***

***<form method="post" action="MyServlet">***

***<p align="center">用户名：<input type="text" name="uname"></p>***

***<p align="center">口&nbsp;&nbsp;令：<input type="password" name="passwd"></p>***

***<p align="center"><input type="submit" name="submit" value="登录"></p>***

***</form>***

***</body>***

***</html>***

**Protect目录下对登录用户的检测**

***<!DOCTYPE html>***

***<html>***

***<head>***

***<title>MyHtml.html</title>***

***<meta http-equiv="content-type" content="text/html; charset=UTF-8">***

***</head>***

***<body>***

***只用登录用户才会看的到我. <br>***

***</body>***

***</html>***

**web.xml的配置 在标签</web-app>之前添加如下内容**

***<!-- 过滤器注册 -->***

***<filter>***

***<filter-name>tempFilter</filter-name>***

***<filter-class>cn.tempFilter</filter-class>***

***</filter>***

***<!-- 过滤器映射 -->***

***<filter-mapping>***

***<filter-name>tempFilter</filter-name>***

***<url-pattern>/Protect/\*</url-pattern>***

***<filter-mapping>***

***<servlet>***

***<servlet-name>MyServlet</servlet-name>***

***<servlet-class>cn.MyServlet</servlet-class>***

***</servlet>***

***<servlet-mapping>***

***<servlet-name>MyServlet</servlet-name>***

***<url-pattern>/MyServlet</url-pattern>***

***</servlet-mapping>***

******

**3**、**Servlet** 生命周期事件

使用生命周期事件统计当前应用的在线人数、运行时间、请求数。

创建生命周期事件监听器点击菜单“File”->“New”->“Class”打开类创建向导对话框如图 7。通过“Add”按钮添加

ServletContextListener 监听器、ServletRequestListener 监听器和 HttpSessionListener 监听器；然后点击“Finish”

按钮完成创建。

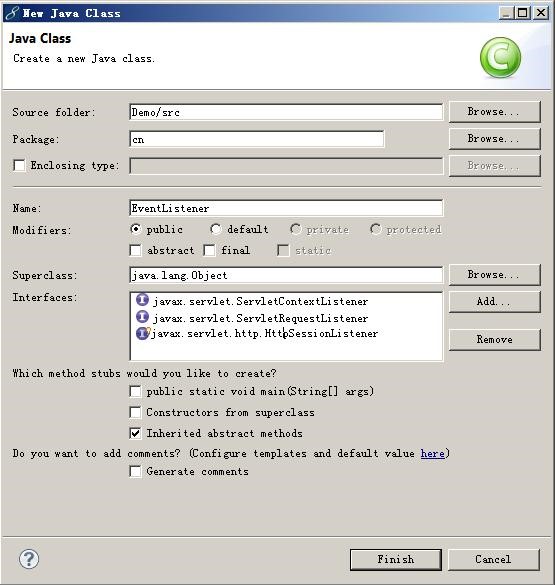


图 7 创建生命周期事件监听器类对话框

编写监听器类代码：

**public class** EventListener **implements** ServletContextListener,

ServletRequestListener, HttpSessionListener {

@Override **public void** sessionCreated(HttpSessionEvent arg0) {

// 当有一个会话创建，则给当前在线人数加1

**int** cnt = 0;

ServletContext context = arg0.getSession().getServletContext();

Object sessionCnt = context.getAttribute("sessionCnt");

**if**(sessionCnt != **null**){ cnt = (Integer)sessionCnt;

} cnt++; context.setAttribute("sessionCnt", cnt);

}

@Override **public void** sessionDestroyed(HttpSessionEvent arg0) {

// 当有一个会话关闭时，则给当前在线人数减1

**int** cnt = 0;

ServletContext context = arg0.getSession().getServletContext(); Object sessionCnt = context.getAttribute("sessionCnt"); **if**(sessionCnt != **null**){ cnt = (Integer)sessionCnt; cnt--; } context.setAttribute("sessionCnt", cnt);

} @Override **public void** requestDestroyed(ServletRequestEvent arg0) {}

@Override **public void** requestInitialized(ServletRequestEvent arg0) {

// 每有一个请求，则计数

**int** cnt = 0;

Object reqcnt = arg0.getServletContext().getAttribute("reqcnt"); **if**(reqcnt != **null**){ cnt = (Integer)reqcnt;

} cnt++; arg0.getServletContext().setAttribute("reqcnt", cnt); } @Override **public void** contextDestroyed(ServletContextEvent arg0) {}

@Override **public void** contextInitialized(ServletContextEvent arg0) {

// 保存应用启动时间

arg0.getServletContext().setAttribute("StartDate", **new** Date());

}

}

创建一个jsp页面显示统计结果:

<%@ page language=*"java"* import=*"java.util.\*"* pageEncoding=*"UTF-8"*%>

<!DOCTYPE>

<html>

<head>

<title>生命周期事件</title>

</head>

<body>

<p>当前应用启动时间：<%=application.getAttribute("StartDate") %></p>

<p>当前在线人数：<%=application.getAttribute("sessionCnt") %></p>

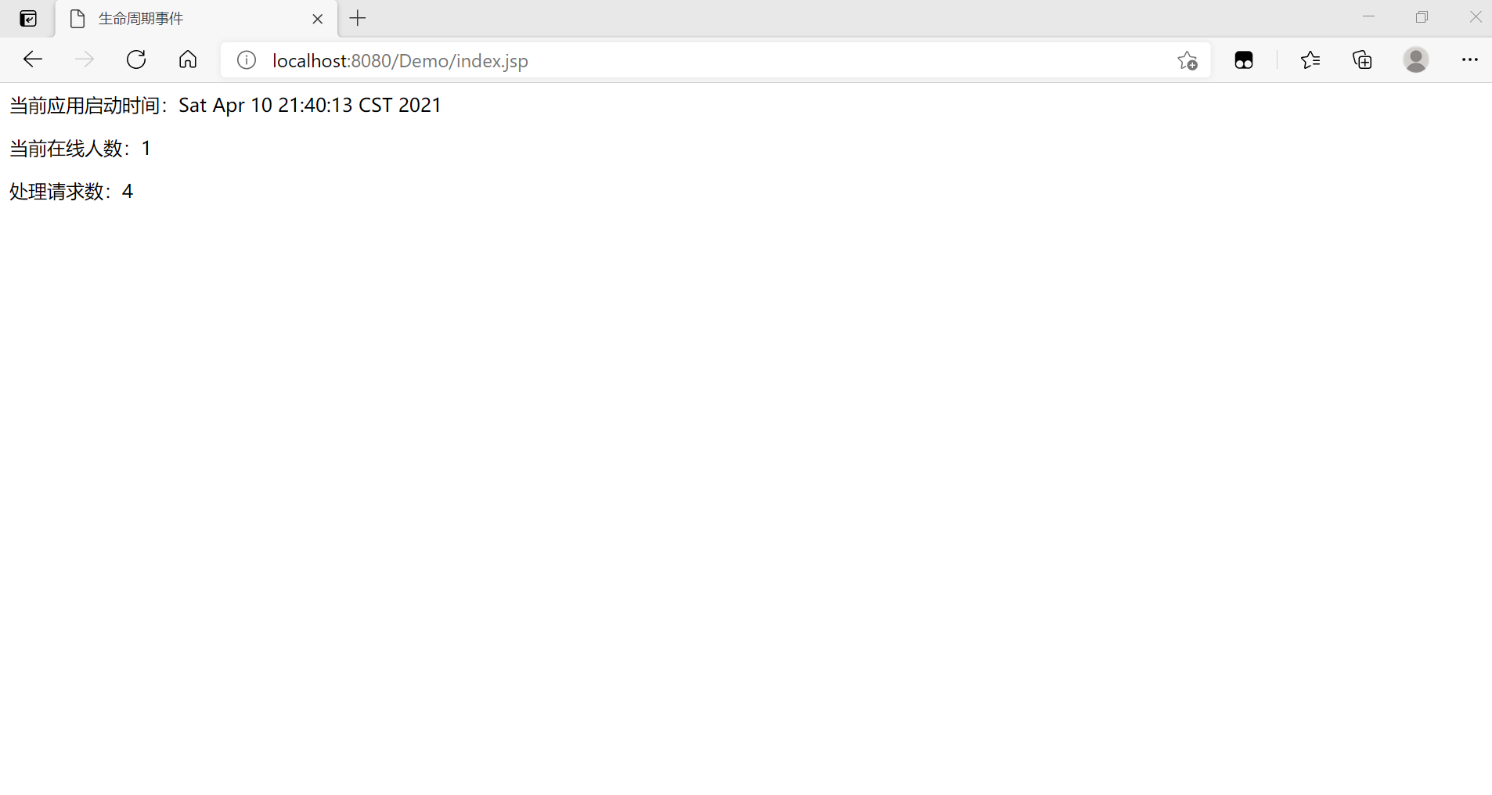
<p>处理请求数：<%=application.getAttribute("reqcnt") %></p>

</body>

</html>

**还需要在web.xml注册监听器类的配置代码**

**<listener><listener-class>cn.EventListener</listener-class> </listener>**

****

# 学习扩展

1. 编写一个 Filter 用于将所有请求的编码设置为指定编码，编码在 web.xml 中通过过滤器初始化参数提供。

用Filter过滤所有的请求，并将编码格式同一改为utf-8

EncodeFilter类：

***public class EncodeFilter implements Filter {***

***private FilterConfig config;***

***@Override***

***public void destroy() {***

***this.config = null;***

***}***

***@Override***

***public void doFilter(ServletRequest arg0, ServletResponse arg1,***

***FilterChain arg2) throws IOException, ServletException {***

***HttpServletRequest request = (HttpServletRequest) arg0;***

***request.setCharacterEncoding(config.getInitParameter("encoding"));***

***arg2.doFilter(arg0, arg1);***

***}***

***@Override***

***public void init(FilterConfig arg0) throws ServletException {***

***this.config = arg0;***

***}***

***}***

然后在web.xml中进行配置

*<filter>*

*<filter-name>EncodeFilter</filter-name>*

*<filter-class>com.phonecontact.filter.EncodeFilter</filter-class>*

*<init-param>*

*<param-name>encoding</param-name>*

*<param-value>utf-8</param-value>*

*</init-param>*

*</filter>*

*<filter-mapping>*

*<filter-name>EncodeFilter</filter-name>*

*<url-pattern>/\*</url-pattern>*

*</filter-mapping>*

<url-pattern> 为可过滤的url，这里的/\*表示可以过滤该项目所有的url。<init-param>、

<param-name> 、<param-value> 分别表示初始化参数，初始化参数名，初始化参数值。由此来改变所有请求的编码格式

1. 使用过滤器及 cookie 实现自动登录。

登陆成功就把登录信息存储到cookie中，将用户名存到session中，下一次进入界面只需要用一个Filter过滤器拦截获取cookie，如果有登录信息的cookie就取出来，进行自动登录处理，每一次访问也通过Filter过滤器进行拦截，查询session中的name属性值得容器中有没有值，如果没有重定向到登录界面，如果有则可进行下一步。