← Servlet 异常处理 Servlet Session 跟踪 →

Servlet Cookie 处理

Cookie 是存储在客户端计算机上的文本文件，并保留了各种跟踪信息。Java Servlet 显然支持 HTTP Cookie。

识别返回用户包括三个步骤：

- 服务器脚本向浏览器发送一组 Cookie。例如：姓名、年龄或识别号码等。
- 浏览器将这些信息存储在本地计算机上，以备将来使用。
- 当下一次浏览器向 Web 服务器发送任何请求时，浏览器会把这些 Cookie 信息发送到服务器，服务器将使用这些信息来识别用户。

本章将向您讲解如何设置或重置 Cookie，如何访问它们，以及如何将它们删除。

Servlet Cookie 处理需要对中文进行编码与解码，方法如下：

```
String   str   =   java.net.URLEncoder.encode("中文", "UTF-8");           //编码
String   str   =   java.net.URLDecoder.decode("编码后的字符串","UTF-8");   // 解码
```

Cookie 剖析

Cookie 通常设置在 HTTP 头信息中（虽然 JavaScript 也可以直接在浏览器上设置一个 Cookie）。设置 Cookie 的 Servlet 会发送如下的头信息：

```
HTTP/1.1 200 OK
Date: Fri, 04 Feb 2000 21:03:38 GMT
Server: Apache/1.3.9 (UNIX) PHP/4.0b3
Set-Cookie: name=xyz; expires=Friday, 04-Feb-07 22:03:38 GMT;
                 path=/; domain=runoob.com
Connection: close
Content-Type: text/html
```

正如您所看到的，Set-Cookie 头包含了一个名称值对、一个 GMT 日期、一个路径和一个域。名称和值会被 URL 编码。expires 字段是一个指令，告诉浏览器在给定的时间和日期之后"忘记"该 Cookie。

如果浏览器被配置为存储 Cookie，它将会保留此信息直到到期日期。如果用户的浏览器指向任何匹配该 Cookie 的路径和域的页面，它会重新发送 Cookie 到服务器。浏览器的头信息可能如下所示：

```
GET / HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.6 (X11; I; Linux 2.2.6-15apmac ppc)
Host: zink.demon.co.uk:1126
```

```
Accept: image/gif, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
Cookie: name=xyz
```

Servlet 就能够通过请求方法 *request.getCookies()* 访问 Cookie，该方法将返回一个 *Cookie* 对象的数组。

Servlet Cookie 方法

以下是在 Servlet 中操作 Cookie 时可使用的有用的方法列表。

序号	方法 & 描述
1	**public void setDomain(String pattern)** 该方法设置 cookie 适用的域，例如 runoob.com。
2	**public String getDomain()** 该方法获取 cookie 适用的域，例如 runoob.com。
3	**public void setMaxAge(int expiry)** 该方法设置 cookie 过期的时间（以秒为单位）。如果不这样设置，cookie 只会在当前 session 会话中持续有效。
4	**public int getMaxAge()** 该方法返回 cookie 的最大生存周期（以秒为单位），默认情况下，-1 表示 cookie 将持续下去，直到浏览器关闭。
5	**public String getName()** 该方法返回 cookie 的名称。名称在创建后不能改变。
6	**public void setValue(String newValue)** 该方法设置与 cookie 关联的值。
7	**public String getValue()** 该方法获取与 cookie 关联的值。
8	**public void setPath(String uri)** 该方法设置 cookie 适用的路径。如果您不指定路径，与当前页面相同目录下的（包括子目录下的）所有 URL 都会返回 cookie。
9	**public String getPath()** 该方法获取 cookie 适用的路径。
10	**public void setSecure(boolean flag)** 该方法设置布尔值，表示 cookie 是否应该只在加密的（即 SSL）连接上发送。
11	**public void setComment(String purpose)**

	设置cookie的注释。该注释在浏览器向用户呈现 cookie 时非常有用。
12	**public String getComment()** 获取 cookie 的注释，如果 cookie 没有注释则返回 null。

通过 Servlet 设置 Cookie

通过 Servlet 设置 Cookie 包括三个步骤：

(1) 创建一个 Cookie 对象：您可以调用带有 cookie 名称和 cookie 值的 Cookie 构造函数，cookie 名称和 cookie 值都是字符串。

```
Cookie cookie = new Cookie("key","value");
```

请记住，无论是名字还是值，都不应该包含空格或以下任何字符：

```
[ ] ( ) = , " / ? @ : ;
```

(2) 设置最大生存周期：您可以使用 setMaxAge 方法来指定 cookie 能够保持有效的时间（以秒为单位）。下面将设置一个最长有效期为 24 小时的 cookie。

```
cookie.setMaxAge(60*60*24);
```

(3) 发送 Cookie 到 HTTP 响应头：您可以使用 **response.addCookie** 来添加 HTTP 响应头中的 Cookie，如下所示：

```
response.addCookie(cookie);
```

实例

让我们修改我们的 表单数据实例，为名字和姓氏设置 Cookie。

```
package com.runoob.test;

import java.io.IOException;
import java.io.PrintWriter;
import java.net.URLEncoder;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
```

```
/**
 * Servlet implementation class HelloServlet
 */
@WebServlet("/HelloForm")
public class HelloForm extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public HelloForm() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException,
 IOException
    {
        // 为名字和姓氏创建 Cookie
        Cookie name = new Cookie("name",
                URLEncoder.encode(request.getParameter("name"), "UTF-8")); // 中文转码
        Cookie url = new Cookie("url",
                      request.getParameter("url"));

        // 为两个 Cookie 设置过期日期为 24 小时后
        name.setMaxAge(60*60*24);
        url.setMaxAge(60*60*24);

        // 在响应头中添加两个 Cookie
        response.addCookie( name );
        response.addCookie( url );

        // 设置响应内容类型
        response.setContentType("text/html;charset=UTF-8");

        PrintWriter out = response.getWriter();
        String title = "设置 Cookie 实例";
        String docType = "<!DOCTYPE html>\n";
        out.println(docType +
                "<html>\n" +
                "<head><title>" + title + "</title></head>\n" +
                "<body bgcolor=\"#f0f0f0\">\n" +
                "<h1 align=\"center\">" + title + "</h1>\n" +
                "<ul>\n" +
                "  <li><b>站点名：</b>："
```

```
                + request.getParameter("name") + "\n</li>" +
                "  <li><b>站点 URL：</b>："
                + request.getParameter("url") + "\n</li>" +
                "</ul>\n" +
                "</body></html>");
        }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletExcept
ion, IOException {
        // TODO Auto-generated method stub
        doGet(request, response);
    }

}
```

编译上面的 Servlet **HelloForm**，并在 web.xml 文件中创建适当的条目:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app>
  <servlet>
    <!-- 类名 -->
    <servlet-name>HelloForm</servlet-name>
    <!-- 所在的包 -->
    <servlet-class>com.runoob.test.HelloForm</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>HelloForm</servlet-name>
    <!-- 访问的网址 -->
    <url-pattern>/TomcatTest/HelloForm</url-pattern>
  </servlet-mapping>
</web-app>
```

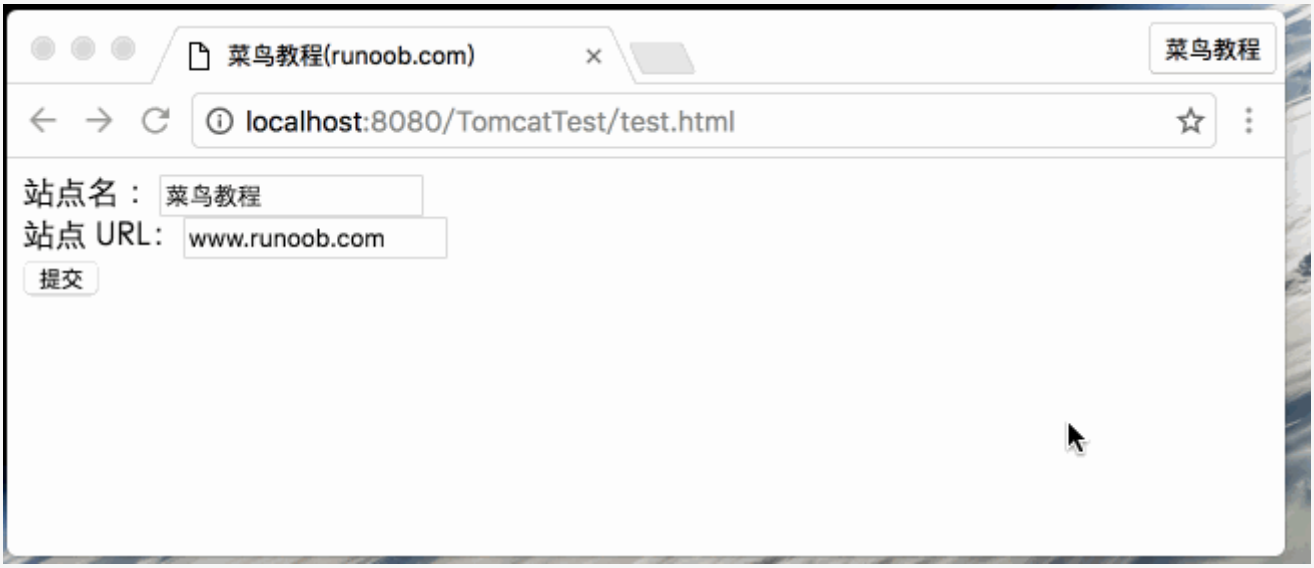
最后尝试下面的 HTML 页面来调用 Servlet。

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>
<form action="/TomcatTest/HelloForm" method="GET">
站点名 ： <input type="text" name="name">
<br />
```

```
站点 URL: <input type="text" name="url" /><br>
<input type="submit" value="提交" />
</form>
</body>
</html>
```

保存上面的 HTML 内容到文件 /TomcatTest/test.html 中。

接下来我们访问http://localhost:8080/TomcatTest/test.html，Gif 演示如下：



> ***注意：****以上的一些路径需要根据你项目实际路径修改。*

通过 Servlet 读取 Cookie

要读取 Cookie，您需要通过调用 *HttpServletRequest* 的 **getCookies( )** 方法创建一个 *javax.servlet.http.Cookie* 对象的数组。

然后循环遍历数组，并使用 getName() 和 getValue() 方法来访问每个 cookie 和关联的值。

实例

让我们读取上面的实例中设置的 Cookie

```
package com.runoob.test;

import java.io.IOException;
import java.io.PrintWriter;
import java.net.URLDecoder;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
```

```
/**
 * Servlet implementation class ReadCookies
 */
@WebServlet("/ReadCookies")
public class ReadCookies extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public ReadCookies() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException,
 IOException
    {
        Cookie cookie = null;
        Cookie[] cookies = null;
        // 获取与该域相关的 Cookie 的数组
        cookies = request.getCookies();

         // 设置响应内容类型
         response.setContentType("text/html;charset=UTF-8");

         PrintWriter out = response.getWriter();
         String title = "Delete Cookie Example";
         String docType = "<!DOCTYPE html>\n";
         out.println(docType +
                   "<html>\n" +
                   "<head><title>" + title + "</title></head>\n" +
                   "<body bgcolor=\"#f0f0f0\">\n" );
          if( cookies != null ){
            out.println("<h2>Cookie 名称和值</h2>");
            for (int i = 0; i < cookies.length; i++){
               cookie = cookies[i];
               if((cookie.getName( )).compareTo("name") == 0 ){
                    cookie.setMaxAge(0);
                    response.addCookie(cookie);
                    out.print("已删除的 cookie: " +
                                  cookie.getName( ) + "<br/>");
               }
               out.print("名称: " + cookie.getName( ) + ", ");
               out.print("值: " +  URLDecoder.decode(cookie.getValue(), "utf-8") +" <br/>");
```

```
            }
         }else{
             out.println(
               "<h2 class=\"tutheader\">No Cookie founds</h2>");
         }
         out.println("</body>");
         out.println("</html>");
        }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletExcept
ion, IOException {
        // TODO Auto-generated method stub
        doGet(request, response);
    }

}
```

编译上面的 Servlet **ReadCookies**，并在 web.xml 文件中创建适当的条目。尝试运行 *http://localhost:8080/TomcatTest/ReadCookies*，将显示如下结果：

```
Cookie 名称和值

名称：JSESSIONID，值：60A8DBEC591EC628B0C572AA5EBF5DD9
名称：name，值：菜鸟教程
名称：url，值：www.runoob.com
名称：CNZZDATA1254569789，值：1732093401-1466493487-|1468220728
名称：pgv_pvid，值：1015625625
```

通过 Servlet 删除 Cookie

删除 Cookie 是非常简单的。如果您想删除一个 cookie，那么您只需要按照以下三个步骤进行：

- 读取一个现有的 cookie，并把它存储在 Cookie 对象中。
- 使用 **setMaxAge()** 方法设置 cookie 的年龄为零，来删除现有的 cookie。
- 把这个 cookie 添加到响应头。

实例

下面的例子将删除现有的名为 "url" 的 cookie，当您下次运行 ReadCookies 的 Servlet 时，它会返回 url 为 null。

```
package com.runoob.test;

import java.io.IOException;
import java.io.PrintWriter;
```

```
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class DeleteCookies
 */
@WebServlet("/DeleteCookies")
public class DeleteCookies extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public DeleteCookies() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException,
 IOException
    {
        Cookie cookie = null;
        Cookie[] cookies = null;
        // 获取与该域相关的 Cookie 的数组
        cookies = request.getCookies();

            // 设置响应内容类型
        response.setContentType("text/html;charset=UTF-8");

        PrintWriter out = response.getWriter();
        String title = "删除 Cookie 实例";
        String docType = "<!DOCTYPE html>\n";
        out.println(docType +
                  "<html>\n" +
                  "<head><title>" + title + "</title></head>\n" +
                  "<body bgcolor=\"#f0f0f0\">\n" );
         if( cookies != null ){
           out.println("<h2>Cookie 名称和值</h2>");
           for (int i = 0; i < cookies.length; i++){
              cookie = cookies[i];
              if((cookie.getName( )).compareTo("url") == 0 ){
                   cookie.setMaxAge(0);
```

```
                    response.addCookie(cookie);
                    out.print("已删除的 cookie: " +
                                cookie.getName( ) + "<br/>");
                }
                out.print("名称: " + cookie.getName( ) + ", ");
                out.print("值: " + cookie.getValue( )+" <br/>");
             }
          }else{
              out.println(
                "<h2 class=\"tutheader\">No Cookie founds</h2>");
          }
          out.println("</body>");
          out.println("</html>");
          }

      /**
       * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
       */
      protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletExcept
  ion, IOException {
          // TODO Auto-generated method stub
          doGet(request, response);
      }

  }
```

编译上面的 Servlet **DeleteCookies**，并在 web.xml 文件中创建适当的条目。现在运行 *http://localhost:8080/TomcatTest/DeleteCookies*，将显示如下结果：

```
Cookie 名称和值

名称：JSESSIONID，值：60A8DBEC591EC628B0C572AA5EBF5DD9
已删除的 cookie：url
名称：url，值：www.runoob.com
名称：CNZZDATA1254569789，值：1732093401-1466493487-%7C1468220728
名称：pgv_pvid，值：1015625625
```

← Servlet 异常处理 Servlet Session 跟踪 →