Web应用开发
之
会话管理

赵小敏
浙江工业大学计算机科学与技术学院

会话管理

- 4.1会话管理
- 4.2使用会话实现购物车
- 4.3 Cookie及其应用
- 4.4 URL重写与隐藏表单域

4.3 Cookie及其应用

- 4.3.1 Cookie API
- 4.3.2 向客户端发送Cookie
- 4.3.3 从客户端读取Cookie
- 4.3.4 Cookie的安全问题
- 4.3.5 用Cookie实现自动登录

Cookie

- Cookie是客户访问Web服务器时，服务器在客户硬盘上存放的信息，好像是服务器送给客户的“点心”。
- Cookie实际上是一小段文本信息，客户以后访问同一个Web服务器时浏览器会把它们原样发送给服务器。
- 通过让服务器读取它原先保存到客户端的信息，网站能够为浏览者提供一系列的方便，例如，在线交易过程中标识用户身份、安全要求不高的场合避免客户登录时重复输入用户名和密码等等。

4.3.1 Cookie API

- 对Cookie的管理需要使用javax.servlet.http.Cookie类，构造方法如下：

  public Cookie(String name, String value)

- 参数name为Cookie名，value为Cookie的值，它们都是字符串。
- Cookie类的常用方法如下：

  public String getName()：返回Cookie名称，名称一旦创建不能改变。

  public String getValue()：返回Cookie的值。

  public void setValue(String newValue)：在Cookie创建后为它指定一个新值。

4.3.1 Cookie API

- public void setMaxAge(int expiry)：设置Cookie在浏览器中的最长存活时间，单位为秒。
- public int getMaxAge()：返回Cookie在浏览器上的最大存活时间。
- public void setDomain(String pattern)：设置该Cookie所在的域。
- public String getDomain()：返回为该Cookie设置的域名。

Cookie的管理包括两个方面：将Cookie对象发送到客户端和从客户端读取Cookie。

4.3.2 向客户端发送Cookie

- 要把Cookie发送到客户端，Servlet先要使用Cookie类的构造方法创建一个Cookie对象，通过setXxx()方法设置各种属性，通过响应对象的addCookie(cookie)方法把Cookie加入响应头。具体步骤如下：

1）创建Cookie对象

```
Cookie userCookie = new Cookie("username", "hacker");
```

4.3.2　向客户端发送Cookie

2）如果希望浏览器将Cookie对象存储到磁盘上，需要使用Cookie类的setMaxAge()方法设置Cookie的最大存活时间。

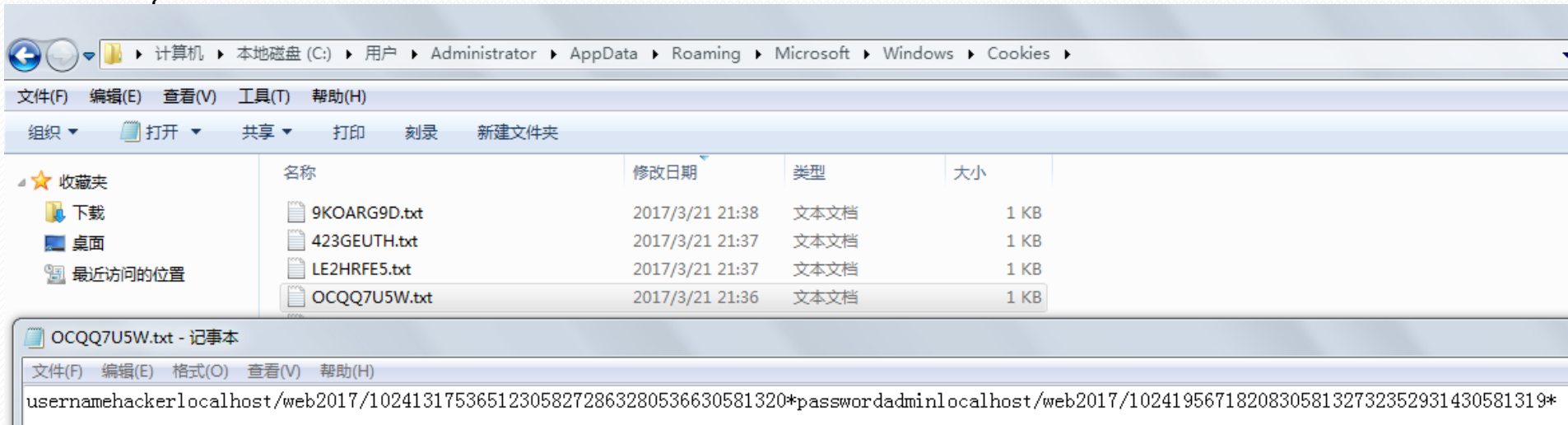
```
userCookie.setMaxAge(60*60*24*7);
```

3）向客户发送Cookie对象

```
response.addCookie(userCookie);
```

向客户端发送Cookie： SendCookieServlet.java

```
public class SendCookieServlet extends HttpServlet{
  public void doGet(HttpServletRequest request,
        HttpServletResponse response)   throws IOException,ServletException{
            Cookie userCookie = new Cookie("username", "hacker");
            userCookie.setMaxAge(60*60*24*7);
            response.addCookie(userCookie);
            response.setContentType("text/html;charset=UTF-8");
            PrintWriter out = response.getWriter();
            out.println("<html><title>发送Cookie</title>");
            out.println("<body><h3>已向浏览器发送一个Cookie。</h3></body>");
            out.println("</html>");
    }
```



4.3.3 从客户端读取Cookie

- 要从客户端读入Cookie，Servlet应该调用请求对象的getCookies()，该方法返回一个Cookie对象的数组。

- 循环访问该数组的各个元素寻找指定名字的Cookie，然后对该Cookie调用getValue()取得与指定名字关联的值。

4.3.3 从客户端读取Cookie

1）调用请求对象的getCookies方法

- Cookie[] cookies=request.getCookies();

2）对Cookie数组循环

```
for(int i = 0;i<cookies.length;i++){
        Cookie cookie = cookies[i];
        if(cookie.getName().equals(cookieName))
            cookieValue = cookie.getValue();
    }
```

- 通过循环访问它的每个元素，然后调用每个Cookie的getName()，直到找到一个与希望的名称相同的对象为止。
- 找到所需要的Cookie对象后，调用它的getValue()，并根据得到的值做进一步处理。

4.3.3 从客户端读取Cookie

程序：ReadCookieServlet.java

```
public class ReadCookieServlet extends HttpServlet{
    public void doGet(HttpServletRequest request,HttpServletResponse response)
                throws IOException,ServletException{
        String cookieName = "username";
        String cookieValue = null;
        Cookie[] cookies = request.getCookies();
        if (cookies!=null){
        for(int i = 0;i<cookies.length;i++){
            Cookie cookie = cookies[i];
            if(cookie.getName().equals(cookieName))
                cookieValue = cookie.getValue();
            }
         }
        response.setContentType("text/html;charset=utf-8");
        PrintWriter out=response.getWriter();
        out.println("<html><title>读取Cookie</title>");
        out.println("<body><h3>从浏览器读回一个Cookie</h3>");
        out.println("Cookie名:"+cookieName+"<br>");
        out.println("Cookie值:"+cookieValue+"<br>");
        out.println("</body></html>");
    }
}
```

4.3.4 Cookie的安全问题

- Cookie是服务器向客户机上写的数据，因此有些用户认为Cookie会带来安全问题，认为Cookie会带来病毒。事实上，Cookie并不会造成安全威胁，Cookie永远不会以任何方式执行。

① 浏览器一般只允许存放300个Cookie，每个站点的Cookie最多存放20个，每个Cookie的大小限制为4 KB，因此Cookie不会占据硬盘多大空间。

② 为了保证安全，许多浏览器还提供了设置是否使用Cookie的功能。

4.3.4 Cookie的安全问题

- 在IE浏览器中打开 “工具”菜单中的“Internet选项”对话框，在“隐私”选项卡中可以设置浏览器是否接受Cookie
- 在该对话框中可以通过一个滑块设置浏览器接收Cookie的级别。其中有6个级别
- 注意，即使客户将Cookie设置为“阻止所有Cookie”，浏览器仍然自动支持会话级的Cookie。

4.3.5 实例：用Cookie实现自动登录

- 许多网站都提供用户自动登录功能，即用户第一次登录网站，服务器将用户名和密码以Cookie的形式发送到客户端。
- 当客户之后再次访问该网站时，浏览器自动将Cookie文件中的用户名和密码随请求一起发送到服务器，服务器从Cookie中取出用户名和密码并且通过验证，这样客户不必再次输入用户名和密码登录网站，这称为自动登录。

4.3.5　实例：用Cookie实现自动登录

- 登陆页面 login1.jsp
- 登陆处理CheckUserServlet.java
- 登陆成功页面：welcome.jsp

4.4.1 URL重写

- 如果浏览器不支持Cookie或用户阻止了所有Cookie，可以把会话ID附加在HTML页面中所有的URL上，这些页面作为响应发送给客户。
- 这样，当用户单击URL时，会话ID被自动作为请求行的一部分而不是作为头行发送回服务器。这种方法称为URL重写（URL rewriting）。

4.4.1 URL重写

- 考虑下面的由名为HomeServlet的Servlet（没有进行URL重写）返回的HTML页面代码:

```
<html><body>
  点击链接查询:<br>
  <a href="/chapter04/ReportServlet">查询销售报表 </a><br>
  <a href="/chapter04/AccountServlet">查询账户信息 </a><br>
</body></html>
```

4.4.1 URL重写

```
<html><body>
 点击链接查询:<br>
  <a href=
"/chapter04/ReportServlet;jsessionid=C084B32241B2F8F060230440C0158114">
查询销售报表</a><br>
<a href=
"/chapter04/AccountServlet;jsessionid=C084B32241B2F8F060230440C0158114">
查询账户信息</a><br>
</body></html>
```

4.4.1 URL重写

- HttpServletResponse接口提供了两个方法。
  - String encodeURL(String url)
  - String encodeRedirectURL(String url)
- 程序: HomeServlet.java

4.4.2 隐藏表单域

- 在HTML页面中，可以使用下面代码实现隐藏的表单域:

- <input **type="hidden"** name="userName" value="hacker">

- 当表单提交时，浏览器将指定的名称和值包含在GET或POST的数据中。
- 这个隐藏域可以存储有关会话的信息。但它的缺点是：仅当每个页面都是由表单提交而动态生成时，才能使用这种方法。

小结：通过Cookie支持会话

- Cookie类的常用方法： getName、 getValue、setValue、setMaxAge和getMaxAge；
- 向客户端发送Cookie
  - 创建Cookie对象
    - `Cookie c = new Cookie("username", "hacker");`
  - 将Cookie放入到HTTP响应中
    - `response.addCookie(c)`
- 从客户端读取Cookie

```
Cookie[] cookies = request.getCookies();
 if (cookies!=null){
   for(int i = o;i<cookies.length;i++){
     Cookie cookie = cookies[i];
     if(cookie.getName().equals(cookieName))
       cookieValue = cookie.getValue();
   }
 }
```

小结：URL重写与隐藏表单域

- 重写正在重定向的URL
  - 调用encodeRedirectURL( )方法

```
response.sendRedirect(
    response.encodeRedirectURL(http://localhost/store/catalog)
```

- 在HTML页面中，可以使用下面代码实现隐藏的表单域：

```
<input type="hidden" name="session" value="a1234">
```

Thank You !