会话技术 -- cookie

2019年5月21日 星期二 9:13

1. 会话是什么？

   为了完成某一个功能，浏览器和服务器建立连接之后，进行一次或多次请求和响应操作，这些请求和响应构成了浏览器和服务器之间的一次会话操作。

2. 会话的作用：

   会话主要是解决在浏览器请求中数据的共享问题而出现的技术。

   http是一个无状态协议。 --- 请求与请求之间是没有关联的。

   --- 利用会话技术可以解决请求之间没有关联的问题。

3. 会话技术--cookie

   a. Cookie的实现原理：

      在第一次请求过后，由服务器发出一个携带set-Cookie响应头的响应，到达浏览器之后，会将其中的数据在浏览端保存一份，以备下次发送请求时自动携带。在下一次发送请求时，请求中自动包含一个请求头cookie其中包含的值是set-cookie在浏览器端保存的值。

   b. 浏览器请求服务器，服务器响应的结果数据会保留在浏览器端一份，这个数据可以通过cookie技术来使用。所以cookie技术是一门浏览器端的技术。

4. 用户上次访问页面的时间

   在set-cookie中保存的时间是用户本次的访问时间，这个访问时间会在下一次请求中包含，并且通过请求头cookie获取，展示在浏览器端。

```
//在浏览器中回显用户上次访问页面的时间
public class CookieDemo1 extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=utf-8");
        //产生一个时间值
        Date date = new Date();
        String time = date.toLocaleString();
        //实现cookie。
        //1.设置setcookie响应头
        response.setHeader("set-cookie", time);
        //2.获取一个请求头cookie
        String cookie = request.getHeader("cookie");
        //如果用户是第一次访问当前页面，则在浏览器中没有cookie请求头，
        //所以无法获取到上次访问的时间
        if(cookie == null){
            response.getWriter().write("您是初次访问这个页面");

        }else{
            //如果不是初次访问，
            //则在cookie请求中会读取浏览器中保留的set-cookie响应头中上次访问时间
            response.getWriter().write("您上次访问本页面的时间是："+cookie);
```

```
        }
    }

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

    }

}
```

5. Cookie类型

Sun公司提供的Cookie类，便于开发者创建、使用cookie。

a. 创建cookie

```
Cookie cookie  = new Cookie(String name, String value);
```

b. 发送cookie到浏览器

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

c. 设置最大生命时长

```
cookie.setMaxAge();//单位为秒
```

如果不设置cookie 的最大生命时长，则cookie是一个会话级别的cookie，在浏览器关闭之后就会销毁，如果定义的了cookie 的生命时长，则cookie会以文件的形式保存在本地磁盘一份(浏览器的临时文件目录)，在到达执行生命时长之后cookie会自动失效，销毁.

d. 设置有效路径

```
cookie.setPath();
```

所谓有效路径就是cookie可以被访问的路径级别。

如果不指定cookie的有效路径级别，则只在当前资源的映射路径下有效。

如果指定cookie的有效路径，则在指定路径及其级别下都可以获取到当前cookie。

推荐在设置路径时加上一个"/"。如果web应用的虚拟路径为缺省路径，则此时将会以"/"作为有效路径，可以避免缺省路径带来的不必要的错误。

e. 获取cookie

```
Cookie[] cs = request.getCookies();
```

结果是一个cookie数据，将数组中的数据遍历使用。

f. 删除cookie

cookie中没有删除cookie的方法。

i. 创建一个名称要和删除的cookie名称相同的cookie，并且要设置这个cookie的path与要删除的相同，然后将新发送的cookie生命时长设置为0，这样就可以删除一个cookie。

ii. 原因：

浏览器在判断两个cookie是否相同时，判断三个内容：Name+Path+domain，默认情况下每一个cookie的domain都是相同的，所以只需要发送一个name和path与要删除的cookie相同的cookie到浏览器就可以完成删除操作了。

```
//创建一个name和path与要删除的完全一致的cookie，并且设置生命时长为0
    //创建cookie
```

```
Cookie cookie = new Cookie("time","");
//path一致
cookie.setPath(request.getContextPath()+"/");
//设置生命时长为0
cookie.setMaxAge(0);
//将cookie发送到浏览器
response.addCookie(cookie);
```

g. cookieAPI

cookie.getName()　----获取cookie 的名称

cookie. getValue()　　----- 获取cookie中包含的值

h. ~cookie小细节

一个cookie标识一种信息。

同一个cookie信息可以在多个浏览器中保存。

一个浏览器中可以保存不同网站的cookie。

6. Cookie类型使用cookie：

```
//cookie生命时长、有效路径设置
public class CookieDemo3 extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=utf-8");
        Date date = new Date();
        String time = date.toLocaleString();
        //创建cookie
        Cookie cookie = new Cookie("time",time);
        //修改cookie生命时长、有效路径
        //设置cookie生命时长（以秒值为单位）
        cookie.setMaxAge(60*60*24);
        //设置路径
        //修改为web应用的虚拟路径
        //如果是缺省web应用，则在request.getContextPath中获取到一个空的返回值。
        //这个值不能作为setPath的值，需要在其后加上一个"/"，来表明是当前web应用的路径。
        cookie.setPath(request.getContextPath()+"/");
        //发送cookie
        response.addCookie(cookie);
        Cookie timeC = null;
        //获取cookie
        Cookie[] cs = request.getCookies();
        if(cs != null){
            for(Cookie c:cs){
                if("time".equals(c.getName())){
                    timeC = c;
                }
            }
        }
        if(timeC != null){
            response.getWriter().write("您上次访问页面的时间为："+timeC.getValue());
        }else{
            response.getWriter().write("您是初次访问本页面");

        }
```

```
    }

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

    }

}
```

7. 修改EasyMall ： 添加登录功能
   a. 添加login.jsp页面
      i. 在课前资料中有login页面的全部资源，粘贴到webRoot目录下。
      ii. 修改login.html为login.jsp页面
      iii. 删除login.html
   b. 添加LoginServlet：

      代码如下：

```
package com.easymall.servlet;

import java.io.IOException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.servlet.ServletException;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.easymall.utils.JDBCUtils;
//登录功能
public class LoginServlet extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        //1.乱码处理
        request.setCharacterEncoding("utf-8");
        response.setContentType("text/html;charset=utf-8");
        //2.获取用户信息
        String username = request.getParameter("username");
        String password = request.getParameter("password");
        String remname = request.getParameter("remname");
        //如果remname值为true则要求记住用户名
        if("true".equals(remname)){
            //为true要求保存用户名
            Cookie cookie = new Cookie("remname",username);
            cookie.setMaxAge(60*60*24*30);
            cookie.setPath(request.getContextPath()+"/");
            response.addCookie(cookie);
        }else{
            //如果不为true则应该销毁cookie
```

```
                Cookie cookie = new Cookie("remname","");
                cookie.setMaxAge(0);
                cookie.setPath(request.getContextPath()+"/");
                response.addCookie(cookie);
            }
            //3.访问数据库，比对用户信息
            Connection conn = null;
            PreparedStatement ps = null;
            ResultSet rs = null;
            try {
                conn = JDBCUtils.getConnection();
                ps = conn.prepareStatement("select * from user where username =? and password
                = ?");
                ps.setString(1, username);
                ps.setString(2, password);
                rs = ps.executeQuery();
                if(rs.next()){
                    //登录操作
                    //如果一致则保存用户登录信息，
                    //TODO:session
                    //4.回到首页
                    response.sendRedirect("http://www.easymall.com");
                }else{
                    //如果不一致则在页面中提示用户名或密码不正确
                    request.setAttribute("msg","用户名或者密码不正确");
                    request.getRequestDispatcher("/login.jsp").forward(request, response);
                    return;
                }
            } catch (SQLException e) {
                e.printStackTrace();
            }finally{
                JDBCUtils.close(conn, ps, rs);
            }



    }

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

    }

}
```