链滴

Java Web 之 Cookie 和 Session 详解

作者：Wuhon

原文链接：https://ld246.com/article/1605187771570

来源网站：链滴

Java Web之Cookie和Session详解

我们知道http协议是无状态的，也就是说就算客户端是第二次访问服务器，服务器还是把此次访当做个新的访问进行处理，因为服务端并不知道客户端之前是否访问过。而cookie和session则就是为了补这一缺陷出现的一种机制。

Cookie

Cookie为服务器给客户端的数据, 可以让服务器清楚的知道该客户端是否访问过自己

常用方法

```
Cookie cookie = new Cookie("name", value);//创建一个名为name, 值为value的cookie
cookie.setMaxAge(1*60*60*24);//cookie存在在本地的有效时长（单位为秒） 默认为-1 表示页面闭cookie就失效
cookie.setDomain("");//设置在某个域名下生效
cookie.setPath("/login.jsp");//设置访问该域名下某个路径时生效
cookie.setMaxAge(0);//cookie中的account
response.addCookie(cookie);//添加到response, 相当于服务器创建一个cookie并将之给客户端

 Cookie[] cookies=request.getCookies();//获取cookies
 cookie.getName();//cookie的name
 cookie.getValue();//cookie的value
```

- 利用cookie实现一个记住网站登陆时间的小demo

```
protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        req.setCharacterEncoding("utf-8");
        resp.setCharacterEncoding("utf-8");
        resp.setContentType("text/html");
        PrintWriter out = resp.getWriter();
        Cookie[] cookies = req.getCookies();
        if(cookies != null){
            // 如果存在
            out.write("你上一次访问的时间是：");
            for (int i=0;i<cookies.length;i++) {
                if(cookies[i].getName().equals("lastLoginTime")){//获取cookie的名字
                    long lastTime = Long.parseLong(cookies[i].getValue());  // 把字符串解析为长整型字符串转为时间戳
                    Date date = new Date(lastTime);
                    out.write(date.toLocaleString());
                }
            }
        }else {
            out.write("这是你第一次访问本网站！");
        }
        // 服务器给客户端响应一个cookie
        Cookie cookie = new Cookie("lastLoginTime", System.currentTimeMillis()+"");
        resp.addCookie(cookie);
    }
```

```
    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletEx
eption, IOException {
        doGet(req, resp);
    }
```

Session

- 服务器会给每个用户(浏览器)创建一个session对象
- 一个session独占一个浏览器, 只要浏览器没关, session就会一直存在
- 通常用来保存用户信息, 既可以保存数据, 也可以保存一个对象
-

session的用法

```
resp.setContentType("text/html");
        resp.setCharacterEncoding("utf-8");
        req.setCharacterEncoding("utf-8");
        // 得到session
        HttpSession session = req.getSession();
        // 给session中存东西
        session.setAttribute("name","wuhobin");  //可以存数据
        session.setAttribute("person",new Person("wuhobin","123",20));  // 也可以存一个对象
        String id = session.getId();
        // 判断session是不是新创建的
        if(session.isNew()){
            resp.getWriter().write("session创建成功，id：  "+id+","+session.getAttribute("name"));
            resp.getWriter().write(session.getAttribute("person").toString());
        }else {
            resp.getWriter().write("session已经存在，id：  "+id+","+session.getAttribute("name"));
            resp.getWriter().write(session.getAttribute("person").toString());
        }
```

session注销

```
protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletExcep
ion, IOException {
        HttpSession session = req.getSession();
        session.removeAttribute("name");
        session.removeAttribute("person");
        // 手动注销session
        session.invalidate();
    }

    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletEx
eption, IOException {
        doGet(req, resp);
    }
```

通过web.xml来注销

```
<!--  设置session默认的消失时间  -->
   <session-config>
<!--   15分钟后session自动消失     -->
       <session-timeout>15</session-timeout>
   </session-config>
```