jsessionid的作用

在web应用的开发中我们会经常看到这样的url：http://www.xxx.com/xxx\_app;jsessionid=xxxxxxxxxx?a=x&b=x...。这跟一般的url基本一样，只有一个地方有区别，那就是“;jessionid=xxxxxxxx”。这个参数有时候有，有时候又没有，说它是参数可又跟一般传递的参数不同，它是紧跟在url后面用分号来分隔的，用一般的request.getParameter()方法还取不到。那这个参数到底是干嘛用的呢？要了解它还要先了解session的实现方式。

session的实现方式

做web开发的同学都知道，http是无状态的会话协议，也就是说无法保存用户的信息（主要是用户的Id）。那如果有一些信息需要在用户的浏览活动中一直保持，该怎么做呢？我们可以把这些信息在每次请求的时候作为参数传递给服务器，但这样做既麻烦又耗费资源，这时候就体现出了session的重要性。session是web开发中不可或缺的一个特性。它是对于一个特定的用户请求，在web服务器上保存的一个全局变量。有了它我们就可以把用户的一些信息保存在服务器上，而不用在服务器和客户端之间来回传递。知道了session的作用，那session是怎么实现的呢？服务器上为每个用户都保存了一个session，那当用户请求过来的时候是怎么知道某一个用户应该对应哪个session呢？这时jsessionid就派上用场了。每一个session都有一个id来作为标识，这个id会传到客户端，每次客户端请求都会把这个id传到服务器（浏览器默认实现），服务器根据id来匹配这次请求应该使用哪个session（通过Request.getSession()方法获取）。jsessionid就是客户端用来保存sessionid的变量，主要是针对j2ee实现的web容器，没有研究过其他语言是用什么变量来保存的。一般对于web应用来说，客户端变量都会保存在cookie中，jsessionid也不例外。不过与一般的cookie变量不同，jsessionid是保存在内存cookie中的，在一般的cookie文件中是看不到它的影子的。内存cookie在打开一个浏览器窗口的时候会创建，在关闭这个浏览器窗口的时候也同时销毁。这也就解释了为什么session变量不能跨窗口使用，要跨窗口使用就需要手动把jsessionid保存到cookie里面。

jsessionid的作用

在以上的文字中我们了解了session的实现原理，同时也知道了session跟jsessionid紧密不可分割的联系。只有通过jsessionid才能使session机制起作用，而jsessionid又是通过cookie来保存。看到这里，也许你会发现一个问题，如果用户禁用了cookie，那jsessionid不是就不能保存了吗？session不是不起作用了吗？我们真的对此束手无策了吗？当然不是。在用户禁用了cookie时候，我们可以通过url重写来实现jsessionid的传递。这就是我上面指出的那样的url:http://www.xxx.com/xxx\_app;jsessionid=xxxxxxxxxx?a=x&b=x..。jessionid通过这样的方式来从客户端传递到服务器端，从而来标识session。注意一点，jsessionid跟一般的url参数传递方式是不同的，不是作为参数跟在？后面，而是紧跟在url后面用；来分隔。这样在用户禁用cookie的时候我们也可以传递jsessionid来使用session了，只不过需要每次都把jseesionid作为参数跟在url后面传递。那这样岂不是很麻烦，每次请求一个url都要判断cookie是否可用，如果禁用了cookie，还要从url里解析出jsessionid，然后跟在处理完后转到的url后面，以保持jsessionid的传递。这些问题sun当然已经帮我们想到了，所以提供了2个方法来使事情变得简单：response.encodeURL()和response.encodeRedirectURL()。这2个方法会判断cookie是否可用，如果禁用了会解析出url中的jsessionid，并连接到指定的url后面，如果没有找到jessionid会自动帮我们生成一个。至于为什么要有2个方法？这2个方法有什么不同？google了一下，说是这2个方法在判断是否要包含jsessionid的逻辑上会稍有不同。在调用HttpServletResponse.sendRedirect前，应该先调用encodeRedirectURL()方法，否则可能会丢失Sesssion信息。这2个方法的使用方法如：response.sendRedirect(response.encodeURL("/myapp/input.jsp"));。如果cookie没有禁用，我们在浏览器地址栏中看到的地址是这样的：/myapp/input.jsp，如果禁用了cookie，我们会看到：/myapp/input.jsp;jsessionid=73E6B2470C91A433A6698C7681FD44F4。所以，我们在写web应用的时候，为了保险起见，应该在程序里的每一个跳转url上都使用这2个方法，来保证session的可用性。

说道这里，大家应该对jsessionid和session的关系，以及jsessionid的作用有个了一个大致的了解，具体应用还要自己在项目中具体情况具体对待。

 =============================================================================================

jsessionid=CA72488F94BC8A3E92FEEDA8CC736FDC       这个jsessionid是session的一个标识。       我在这里转贴jdbc老大的部分讲解       session机制是一种服务器端的机制，服务器使用一种类似于散列表的结构（也可能就是使用散列表）来保存信息。       当程序需要为某个客户端的请求创建一个session的时候，服务器首先检查这个客户端的请求里是否已包含了一个session标识 - 称为 session id，如果已包含一个session id则说明以前已经为此客户端创建过session，服务器就按照session id把这个 session检索出来使用（如果检索不到，可能会新建一个），如果客户端请求不包含session id，则为此客户端创建一个session并且生成一个与此session相关联的session id，session id的值应该是一个既不会重复，又不容易被找到规律以仿造的字符串，这个 session id将被在本次响应中返回给客户端保存。       保存这个session id的方式可以采用cookie，这样在交互过程中浏览器可以自动的按照规则把这个标识发挥给服务器。一般这个cookie的名字都是类似于SEEESIONID，而。比如weblogic对于web应用程序生成的cookie，JSESSIONID= ByOK3vjFD75aPnrF7C2HmdnV6QZcEbzWoWiBYEnLerjQ99zWpBng!-145788764，它的名字就是 JSESSIONID。       由于cookie可以被人为的禁止，必须有其他机制以便在cookie被禁止时仍然能够把session id传递回服务器。经常被使用的一种技术叫做URL重写，就是把session id直接附加在URL路径的后面，附加方式也有两种，一种是作为URL路径的附加信息，表现形式为http://...../xxx;jsessionid= ByOK3vjFD75aPnrF7C2HmdnV6QZcEbzWoWiBYEnLerjQ99zWpBng!-145788764另一种是作为查询字符串附加在URL后面，表现形式为http://...../xxx?jsessionid=ByOK3vjFD75aPnrF7C2HmdnV6QZcEbzWoWiBYEnLerjQ99zWpBng!-145788764这两种方式对于用户来说是没有区别的，只是服务器在解析的时候处理的方式不同，采用第一种方式也有利于把session id的信息和正常程序参数区分开来。为了在整个交互过程中始终保持状态，就必须在每个客户端可能请求的路径后面都包含这个session id。       另一种技术叫做表单隐藏字段。就是服务器会自动修改表单，添加一个隐藏字段，以便在表单提交时能够把session id传递回服务器。比如下面的表单<form name="testform" action="/xxx"><input type="text"></form>       在被传递给客户端之前将被改写成<form name="testform" action="/xxx"><input type="hidden" name="jsessionid" value="ByOK3vjFD75aPnrF7C2HmdnV6QZcEbzWoWiBYEnLerjQ99zWpBng!-145788764"><input type="text"></form>       这种技术现在已较少应用，笔者接触过的很古老的iPlanet6(SunONE应用服务器的前身)就使用了这种技术。实际上这种技术可以简单的用对action应用URL重写来代替。       在谈论session机制的时候，常常听到这样一种误解“只要关闭浏览器，session就消失了”。其实可以想象一下会员卡的例子，除非顾客主动对店家提出销卡，否则店家绝对不会轻易删除顾客的资料。对session来说也是一样的，除非程序通知服务器删除一个session，否则服务器会一直保留，程序一般都是在用户做log off的时候发个指令去删除session。然而浏览器从来不会主动在关闭之前通知服务器它将要关闭，因此服务器根本不会有机会知道浏览器已经关闭，之所以会有这种错觉，是大部分session机制都使用会话cookie来保存session id，而关闭浏览器后这个 session id就消失了，再次连接服务器时也就无法找到原来的session。

       如果服务器设置的cookie被保存到硬盘上，或者使用某种手段改写浏览器发出的HTTP请求头，把原来的session id发送给服务器，则再次打开浏览器仍然能够找到原来的session。       恰恰是由于关闭浏览器不会导致session被删除，迫使服务器为seesion设置了一个失效时间，当距离客户端上一次使用session的时间超过这个失效时间时，服务器就可以认为客户端已经停止了活动，才会把session删除以节省存储空间。

---------------------

作者：叶升升

来源：CSDN

原文：https://blog.csdn.net/ye\_sheng/article/details/48395755

版权声明：本文为博主原创文章，转载请附上博文链接！