HttpSession常见问题

1、session在何时被创建

一个常见的误解是以为session在有客户端访问时就被创建，然而事实是直到某server端程序调用 HttpServletRequest.getSession(true)这样的语句时才被创建，注意如果JSP没有显式的使用 <%@page session="false"%> 关闭session，则JSP文件在编译成Servlet时将会自动加上这样一条语句HttpSession session = HttpServletRequest.getSession(true);这也是JSP中隐含的session对象的来历。

由于session会消耗内存资源，因此，如果不打算使用session，应该在所有的JSP中关闭它。

2、session何时被删除

综合前面的讨论，session在下列情况下被删除a.程序调用HttpSession.invalidate();或b.距离上一次收到客户端发送的session id时间间隔超过了session的超时设置;或c.服务器进程被停止（非持久session）

3、如何做到在浏览器关闭时删除session

严格的讲，做不到这一点。可以做一点努力的办法是在所有的客户端页面里使用javascript代码 window.oncolose来监视浏览器的关闭动作，然后向服务器发送一个请求来删除session。但是对于浏览器崩溃或者强行杀死进程这些非常规 手段仍然无能为力。

4、有个HttpSessionListener是怎么回事

你可以创建这样的listener去监控session的创建和销毁事件，使得在发生这样的事件时你可以做一 些相应的工作。注意是session的创建和销毁动作触发listener，而不是相反。类似的与HttpSession有关的listener还有 HttpSessionBindingListener，HttpSessionActivationListener和 HttpSessionAttributeListener。

5、存放在session中的对象必须是可序列化的吗

不是必需的。要求对象可序列化只是为了session能够在集群中被复制或者能够持久保存或者在必要时 server能够暂时把session交换出内存。在 Weblogic Server的session中放置一个不可序列化的对象在控制台上会收到一个警告。我所用过的某个iPlanet版本如果session中有不可序列化 的对象，在session销毁时会有一个Exception，很奇怪。

6、如何才能正确的应付客户端禁止cookie的可能性

对所有的URL使用URL重写，包括超链接，form的action，和重定向的URL，具体做法参见[6] http://e-docs.bea.com/wls/docs70/webapp/sessions.html#100770

7、开两个浏览器窗口访问应用程序会使用同一个session还是不同的session

参见第三小节对cookie的讨论，对session来说是只认id不认人，因此不同的浏览器，不同的窗口打开方式以及不同的cookie存储方式都会对这个问题的答案有影响。

8、如何防止用户打开两个浏览器窗口操作导致的session混乱

这个问题与防止表单多次提交是类似的，可以通过设置客户端的令牌来解决。就是在服务器每次生成一个不同的id 返回给客户端，同时保存在session里，客户端提交表单时必须把这个id也返回服务器，程序首先比较返回的id与保存在session里的值是否一 致，如果不一致则说明本次操作已经被提交过了。可以参看《J2EE核心模式》关于表示层模式的部分。需要注意的是对于使用javascript window.open打开的窗口，一般不设置这个id，或者使用单独的id，以防主窗口无法操作，建议不要再window.open打开的窗口里做修改 操作，这样就可以不用设置。

9、为什么在Weblogic Server中改变session的值后要重新调用一次 session.setValue

做这个动作主要是为了在集群环境中提示Weblogic Server session中的值发生了改变，需要向其他服务器进程复制新的session值。

10、为什么session不见了

排除session正常失效的因素之外，服务器本身的可能性应该是微乎其微的，虽然笔者在iPlanet6SP1加若干补丁的Solaris版本上倒也遇到过；浏览器插件的可能性次之，笔者也遇到过3721插件造成的问题；理论上防火墙或者代 理服务器在cookie处理上也有可能会出现问题。
出现这一问题的大部分原因都是程序的错误，最常见的就是在一个应用程序中去访问另外一个应用程序。我们在下一节讨论这个问题。

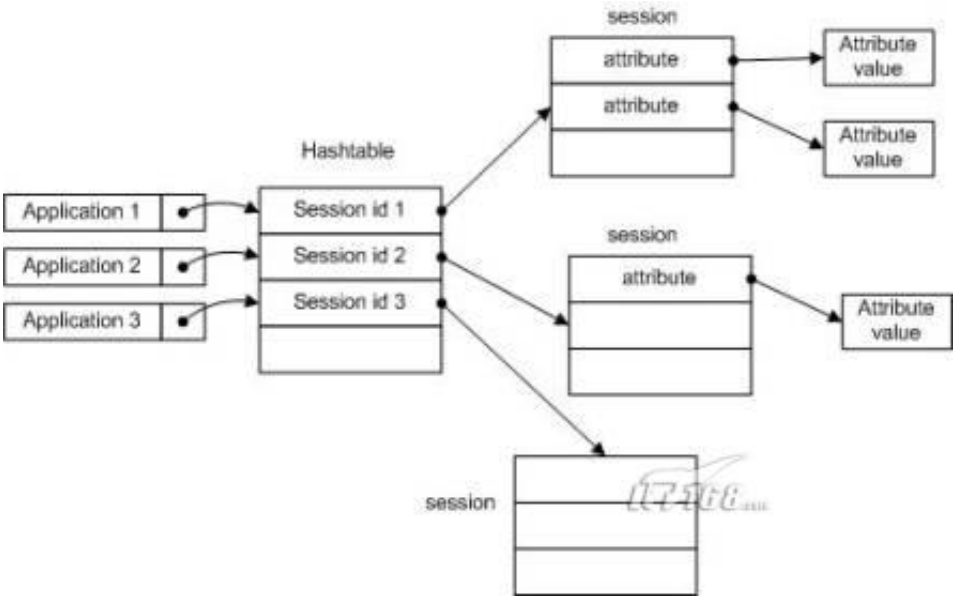
七、跨应用程序的session共享

常常有这样的情况，一个大项目被分割成若干小项目开发，为了能够互不干扰，要求每个小项目作为一个单独的 web应用程序开发，可是到了最后突然发现某几个小项目之间需要共享一些信息，或者想使用session来实现SSO(single sign on)，在session中保存login的用户信息，最自然的要求是应用程序间能够访问彼此的session。

然而按照Servlet规范，session的作用范围应该仅仅限于当前应用程序下，不同的应用程序之间是不 能够互相访问对方的session的。各个应用服务器从实际效果上都遵守了这一规范，但是实现的细节却可能各有不同，因此解决跨应用程序session共 享的方法也各不相同。

首先来看一下Tomcat是如何实现web应用程序之间session的隔离的，从Tomcat设置的cookie路径来看，它对不同的应用程序设置的 cookie路径是不同的，这样不同的应用程序所用的session id是不同的，因此即使在同一个浏览器窗口里访问不同的应用程序，发送给服务器的session id也可以是不同的。

根据这个特性，我们可以推测Tomcat中session的内存结构大致如下。



笔者以前用过的iPlanet也采用的是同样的方式，估计SunONE与iPlanet之间不会有太大的差别。对于这种方式的服务器，解决的思路很简单， 实际实行起来也不难。要么让所有的应用程序共享一个session id，要么让应用程序能够获得其他应用程序的session id。

iPlanet中有一种很简单的方法来实现共享一个session id，那就是把各个应用程序

的cookie路径都设为/（实际上应该是/NASApp，对于应用程序来讲它的作用相当于根）。

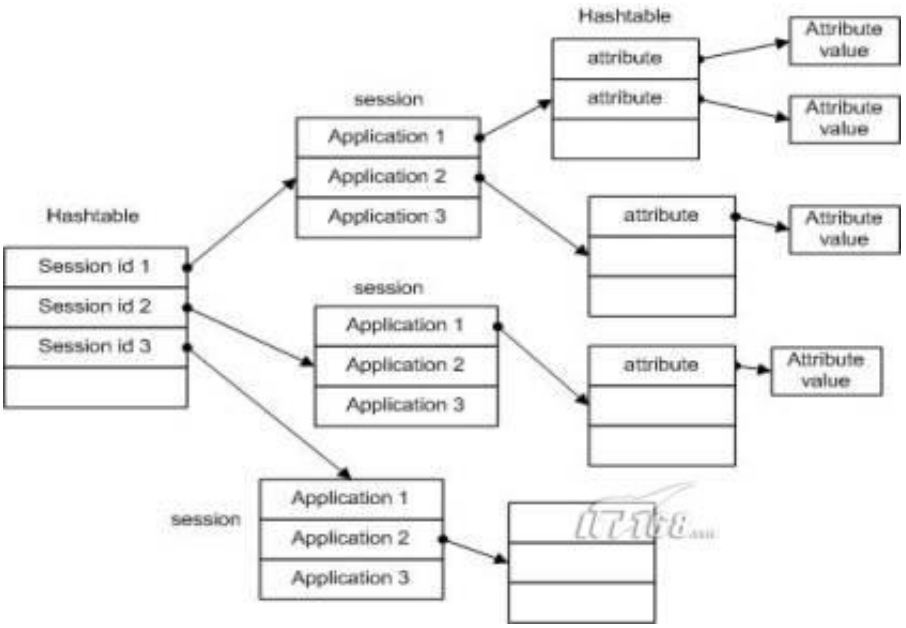
```
<session-info>
<path>/NASApp</path>
</session-info>
```

需要注意的是，操作共享的session应该遵循一些编程约定，比如在session attribute名字的前面加上应用程序的前缀，使得setAttribute("name", "neo")变成setAttribute("app1.name", "neo")，以防止命名空间冲突，导致互相覆盖。

在Tomcat中则没有这么方便的选择。在Tomcat版本3上，我们还可以有一些手段来共享session。对于版本4以上的Tomcat，目前笔者尚 未发现简单的办法。只能借助于第三方的力量，比如使用文件、数据库、JMS或者客户端cookie，URL参数或者隐藏字段等手段。

我们再看一下Weblogic Server是如何处理session的。

从截屏画面上可以看到Weblogic Server对所有的应用程序设置的cookie的路径都是/，这是不是意味着在Weblogic Server中默认的就可以共享session了呢？然而一个小实验即可证明即使不同的应用程序使用的是同一个session，各个应用程序仍然只能访问 自己所设置的那些属性。这说明Weblogic Server中的session的内存结构可能如下：



对于这样一种结构，在session机制本身上来解决session共享的问题应该是不可能的了。除了借助于第三方的力量，比如使用文件、数据库、JMS 或者客户端cookie，URL参数或者隐藏字段等手段，还有一种较为方便的做法，就是把一个应用程序的session放到ServletContext 中，这样另外一个应用程序就可以从ServletContext中取得前一个应用程序的引用。示例代码如下，

应用程序A ：

```
context.setAttribute("appA", session);
```

应用程序B ：

```
contextA = context.getContext("/appA");
HttpSession sessionA = (HttpSession)contextA.getAttribute("appA");
```

值得注意的是这种用法不可移植，因为根据ServletContext的JavaDoc，应用服务器可以处于安全的原因对于context.getContext("/appA");返回空值，以上做法在Weblogic Server 8.1中通过。

那么Weblogic Server为什么要把所有的应用程序的cookie路径都设为/呢？原来是为了SSO，凡是共享这个session的应用程序都可以共享认证的信息。一 个简单的实验就可以证明这一点，修改首先登录的那个应用程序的描述符weblogic.xml，把cookie路径修改为/appA访问另外一个应用程序 会重新要求登录，即使是反过来，先访问cookie路径为/的应用程序，再访问修改过路径的这个，虽然不再提示登录，但是登录的用户信息也会丢失。注意做 这个实验时认证方式应该使用FORM，因为浏览器和web服务器对basic认证方式有其他的处理方式，第二次请求的认证不是通过session来实现 的。具体请参看[7] secion 14.8 Authorization，你可以修改所附的示例程序来做这些试验。

八、总结

session机制本身并不复杂，然而其实现和配置上的灵活性却使得具体情况复杂多变。这也要求我们不能把仅仅某一次的经验或者某一个浏览器，服务器的经验当作普遍适用的经验，而是始终需要具体情况具体分析。

来源： 〈http://www.blogjava.net/hello-yun/archive/2011/04/09/347920.html〉

Method Summary	
java.lang.Object	getAttribute(java.lang.String name) Returns the object bound with the specified name in this session, or null if no object is bound under the name.
java.util.Enumeration	getAttributeNames() Returns an Enumeration of String objects containing the names of all the objects bound to this session.
long	getCreationTime() Returns the time when this session was created, measured in milliseconds since midnight January 1, 1970 GMT.
java.lang.String	getId() Returns a string containing the unique identifier assigned to this session.
long	getLastAccessedTime() Returns the last time the client sent a request associated with this session, as the number of milliseconds since midnight January 1, 1970 GMT, and marked by the time the container received the request.
int	getMaxInactiveInterval() Returns the maximum time interval, in seconds, that the servlet container will keep this session open between client accesses.
ServletContext	getServletContext() Returns the ServletContext to which this session belongs.

t	
HttpSessionContext	getSessionContext() Deprecated. As of Version 2.1, this method is deprecated and has no replacement. It will be removed in a future version of the Java Servlet API.
java.lang.Object	getValue(java.lang.String name) Deprecated. As of Version 2.2, this method is replaced by getAttribute(java.lang.String).
java.lang.String[]	getValueNames() Deprecated. As of Version 2.2, this method is replaced by getAttributeNames()
void	使无效 Invalidates this session then unbinds any objects bound to it.
布林值	isNew() Returns true if the client does not yet know about the session or if the client chooses not to join the session.
void	putValue(java.lang.String name, java.lang.Object value) Deprecated. As of Version 2.2, this method is replaced by setAttribute(java.lang.String, java.lang.Object)
void	removeAttribute(java.lang.String name) Removes the object bound with the specified name from this session.
void	removeValue(java.lang.String name) Deprecated. As of Version 2.2, this method is replaced by removeAttribute(java.lang.String)
void	setAttribute(java.lang.String name, java.lang.Object value) Binds an object to this session, using the name specified.
void	setMaxInactiveInterval(int interval) Specifies the time, in seconds, between client requests before the servlet container will invalidate this session.

来源： <http://tomcat.jaxwiki.org/servletapi/javax/servlet/http/HttpSession.html>