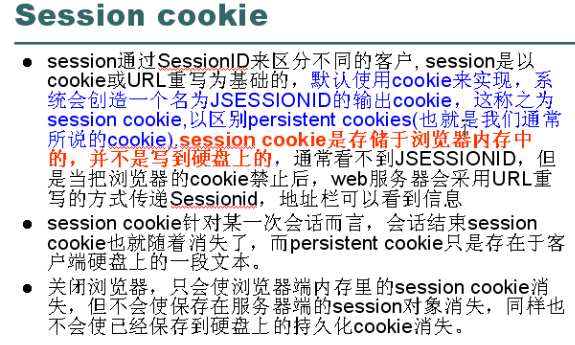
Session

# Session cookie

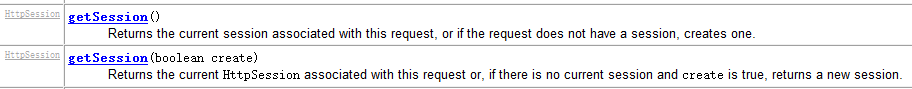


# Session的创建与删除

## Session的创建：

**只有在HttpServletRequest调用getSession（true）时，才会创建Session。**

**返回的是HttpSession对象。**



一般使用**getSession(true)**:

如果当前存在Session，则返回当前的HttpSession，否则，如果设置为true，则返回一个新的HttpSession。

**注意：**

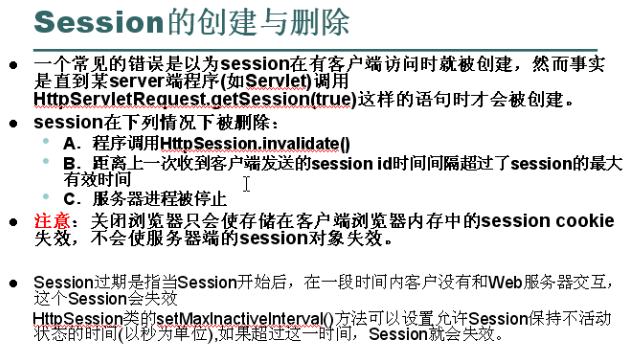
**获取HttpSession的方法只有是通过HttpServletRequest的getSession方法，并没有提供通过SessionID来获取HttpSession对象，具体的获取方法利用Servlet容器来实现。**

**调用getSession，则不会产生Session会话。**





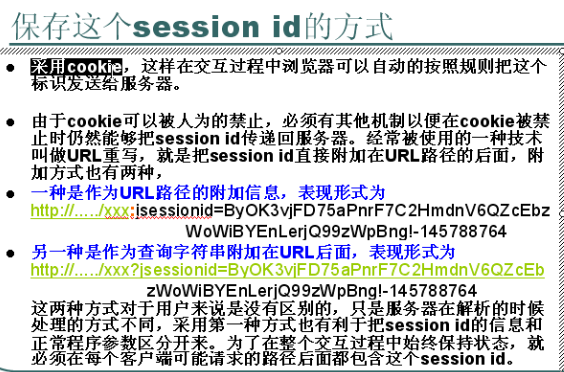


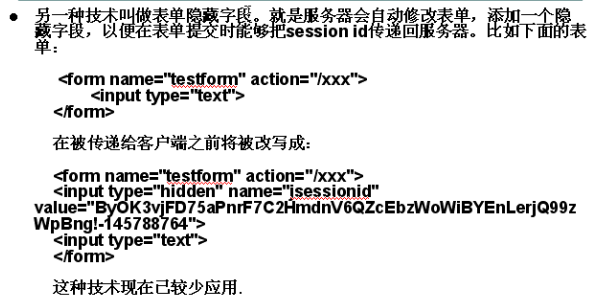


注意：



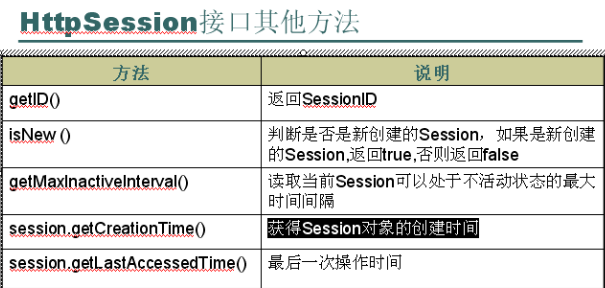
## SessionId的保存





该方法已经基本淘汰了，不再使用了。

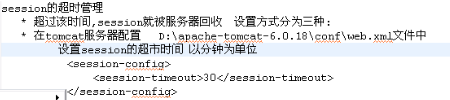
# HttpSession接口的API



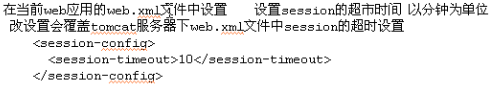
# HttpSession的超时管理

方法介绍：

## 设置方法1：修改Tomcat服务器配置:web.xml文件中修改。



## 设置方法2：

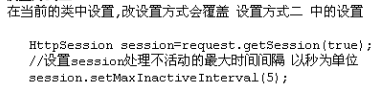




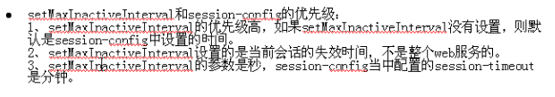
## 设置方式3：

**利用HttpSession的setMaxInactiveInterval方法**。

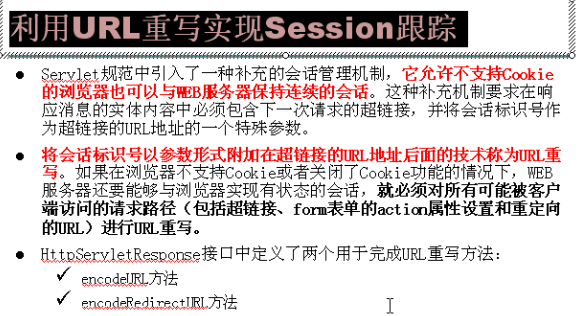




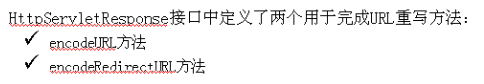
## 三种方法的优先级

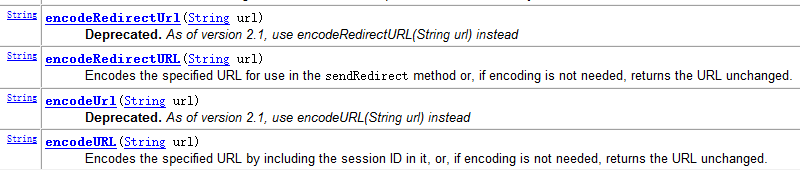


# 利用URL重写实现Session跟踪



# URL重写

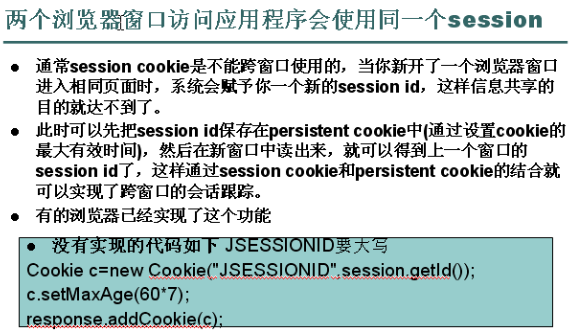




示例：



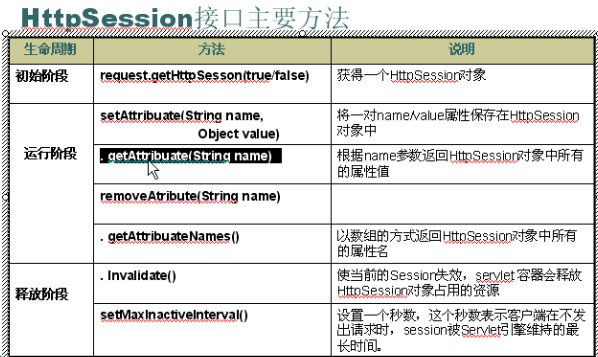
# 两个浏览器窗口访问应用程序会使用同一个session



项目中只需要写到**访问入口处**即可，后面的访问就可以共享会话了。



# HttpSession中的信息共享



HttpSession就是一个容器，可以存放数据，使得数据能够共享。Session中可以存放任何数据类型，即可以存放Object对象及其子类对象。

### 通过setAttribute方法即可存放。



### 通过getAttribute（String name） 方法获取，返回Object。





### removeAttribute（String name）：删除。



# 应用

可以通过Session，把用户名及密码存放到Session中，然后以后的访问都可以共享这些数据。



