核心编程第5天

[学习目标 2](#_Toc521597853)

[COOKIE概述 2](#_Toc521597854)

[1、什么是COOKIE？ 2](#_Toc521597855)

[2、IE浏览器查看COOKIE数据 2](#_Toc521597856)

[3、使用COOKIE的好处 4](#_Toc521597857)

[4、COOKIE的工作原理 5](#_Toc521597858)

[COOKIE操作 6](#_Toc521597859)

[1、添加COOKIE数据 6](#_Toc521597860)

[2、读取COOKIE数据 6](#_Toc521597861)

[COOKIE设置 7](#_Toc521597862)

[1、COOKIE过期有效性设置 7](#_Toc521597863)

[（1）即时性COOKIE设置 7](#_Toc521597864)

[（2）有效性COOKIE设置 7](#_Toc521597865)

[2、COOKIE路径有效性 8](#_Toc521597866)

[3、COOKIE域名有效性 9](#_Toc521597867)

[4、是否仅https安全连接才能发送cookie呢？ 11](#_Toc521597868)

[5、是否只能通过http协议来使用COOKIE呢？ 13](#_Toc521597869)

[删除COOKIE数据 14](#_Toc521597870)

[1、设置COOKIE有效性为过去的某一个时间 14](#_Toc521597871)

[2、设置COOKIE的值为false或空字符串 14](#_Toc521597872)

[3、清理浏览器缓存 14](#_Toc521597873)

[COOKIE的总结 15](#_Toc521597874)

[SESSION概述 15](#_Toc521597875)

[1、什么是SESSION？ 15](#_Toc521597876)

[2、SESSION的工作原理 15](#_Toc521597877)

[3、开启SESSION会话功能 16](#_Toc521597878)

[SESSION操作 16](#_Toc521597879)

[1、添加SESSION数据 16](#_Toc521597880)

[2、读取SESSION数据 17](#_Toc521597881)

[3、删除SESSION数据 17](#_Toc521597882)

[4、销毁SESSION文件 17](#_Toc521597883)

[SESSION对应COOKIE的配置(php.ini) 18](#_Toc521597884)

[1、SESSION对应COOKIE过期时间设置 18](#_Toc521597885)

[2、SESSION对应COOKIE有效路径设置 18](#_Toc521597886)

[3、SESSION对应COOKIE域名有效性设置 18](#_Toc521597887)

[4、是否仅限https来发送SESSION对应的COOKIE数据 19](#_Toc521597888)

[5、是否仅限http来使用SESSION对应的COOKIE数据 19](#_Toc521597889)

[SESSION垃圾回收机制 19](#_Toc521597890)

[1、什么是SESSION的垃圾回收机制？ 19](#_Toc521597891)

[2、垃圾回收的周期：session.gc\_maxlifetime 19](#_Toc521597892)

[3、垃圾回收的概率：session.gc\_divisor 19](#_Toc521597893)

# 学习目标

* COOKIE概述、操作、删除、配置
* SESSION概述、操作、删除、配置
* SESSION的垃圾回收

# COOKIE概述

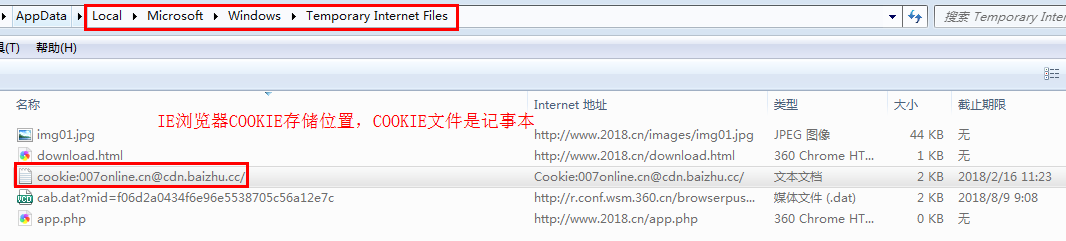
## 1、什么是COOKIE？

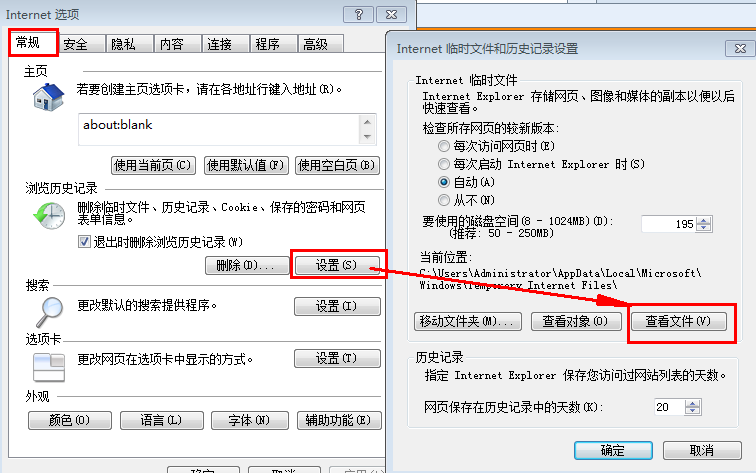
* 简单来说，Cookies就是服务器暂时存放在你的电脑里的资料（.txt格式的文本文件），好让服务器用来辨认你的计算机（通常经过加密）；
* Cookie是由服务器端生成，发送给User-Agent（一般是浏览器），浏览器会将Cookie的key/value保存到某个目录下的文本文件内，**下次请求同一网站时就发送该Cookie给服务器**。
* Cookie名称和值可以由服务器端开发者定义，这样服务器可以知道该用户是否是合法用户以及是否需要重新登录等，服务器可以设置或读取Cookies中包含信息，借此维护用户跟服务器会话中的状态。

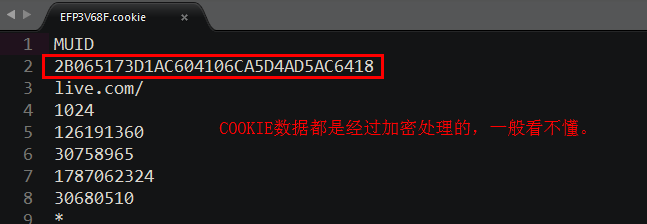
**注意：每种浏览器都可以存储COOKIE数据；各个浏览器之间的COOKIE数据不能共享。**

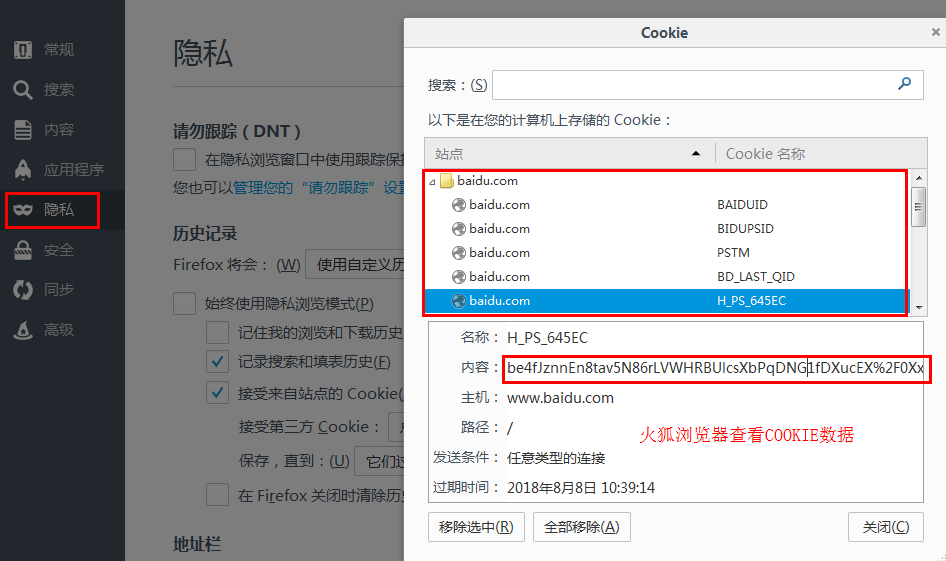
## 2、IE浏览器查看COOKIE数据

访问路径：工具栏菜单 >> internet选项 >> 常规选项卡 >> 浏览器历史记录 >> 设置 >> 查看文件





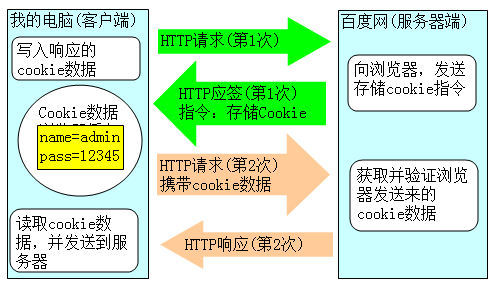




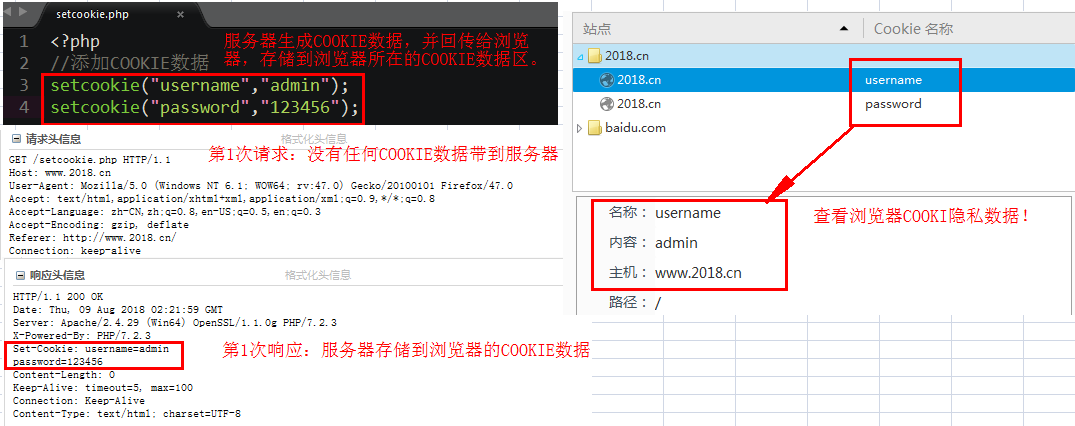
## 3、使用COOKIE的好处

* 用户登录时，记住登录账号和密码，为下一次登录时简化操作；
* 另一个重要应用场合是“购物车”之类处理。用户可能会在一段时间内在同一家网站的不同页面中选择不同的商品，这些信息都会写入Cookies，以便在最后付款时提取信息。
* 跟踪统计用户访问网站的习惯。什么时间访问，访问了哪些页面，页面停留时间等，为用户提供个性化服务，为网站营销提供参考。

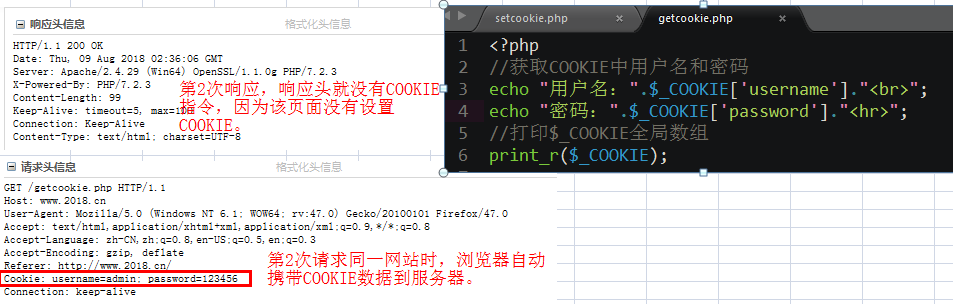
## 4、COOKIE的工作原理



**第1次请求和第1次响应**



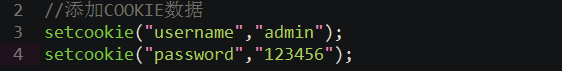
**第2次请求和第2次响应**



# COOKIE操作

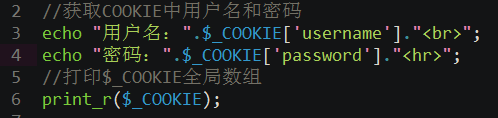
## 1、添加COOKIE数据

* 描述：向客户端发送一个 HTTP cookie。
* 语法：**bool setcookie ( string $name [, string $value [, int $expire = 0 [, string $path [, string $domain [, bool $secure = false [, bool $httponly = false ]]]]]] )**
* 参数：
  + $name，cookie的名称；
  + $value，cookie的值，会自动进行 URL 编码。接收时会进行 URL 解码。**COOKIE值的类型只能是标量数据类型，不可是数组、对象、资源**。
  + $expire，cookie的有效期；
  + $path，cookie的服务器路径；
  + $domain，cookie的域名；
  + $secure，规定是否通过安全的 HTTPS 连接来传输 cookie。
* 返回：设置成功返回TRUE，失败返回FALSE。



## 2、读取COOKIE数据

* 描述：获取cookie数据是通过超全局数组**$\_COOKIE**来实现的。
* 语法：**value = $\_COOKIE[key]**

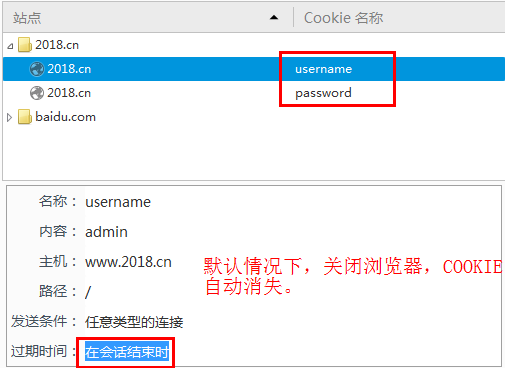


# COOKIE设置

## 1、COOKIE过期有效性设置

### （1）即时性COOKIE设置

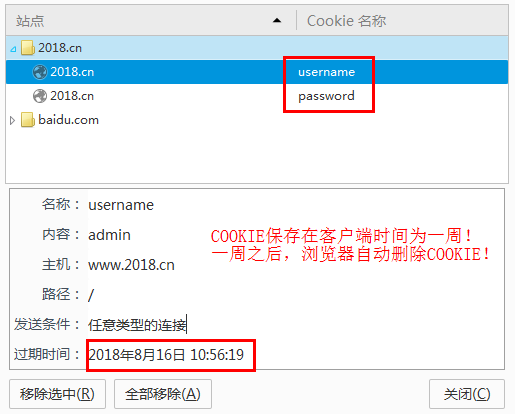
* 描述：默认cookie的有效期是关闭浏览器时，自动失效。该cookie称为临时cookie；
* 语法：**bool setcookie ( string $name , string $value [, int $expire = 0 ])**
* 参数：$expire可选，指定cookie保存的时间，默认为0，关闭浏览器失效；



### （2）有效性COOKIE设置

* 语法： **bool setcookie ( string $name , string $value [, int $expire = 0 ])**
* 参数：$expire可选，指定cookie保存的时间，默认为0，关闭浏览器失效；
* 说明：$expire为一个时间戳。一般用**time()+N**表示新的时间戳。在当前时间点，再加上N秒后，产生一个新的时间戳。
* 提示：一旦指定了有效的过期时间，cookie失效跟关闭浏览器无关(持久cookie)。



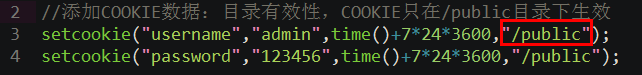


## 2、COOKIE路径有效性

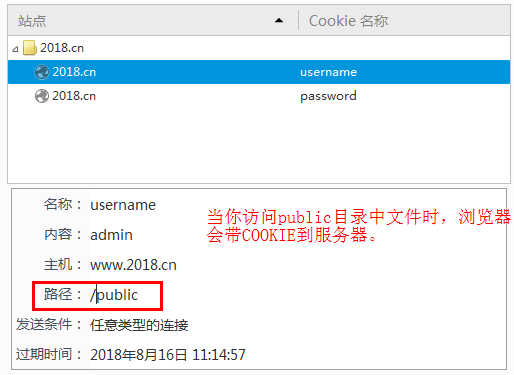
* 描述：设置COOKIE只能在指定的目录及其子目录下有效
* 语法：**bool setcookie ( string $name [, string $value [, int $expire = 0 [, string $path]]] )**
* 参数：**$path默认为"/"**，代表当前网站根目录，在当前目录及其所有子目录都可以使用。
* 说明：由于浏览器每次请求时，携带到服务器端很多的COOKIE数据，会导致请求数据变得庞大；将COOKIE划分到各个路径中，减少每个路径下COOKIE的数量。

**例如：京东保存在客户端的有1万个COOKIE数据，大小为100MB，每次请求都需要1个小时。这样一来，每次请求带1个万COOKIE，但是当前页码只使用2个COOKIE，造成带宽浪费。**

**解决方案是：请求页面需要什么COOKIE，就带什么COOKIE过去，有针对性，会节省宽带。**

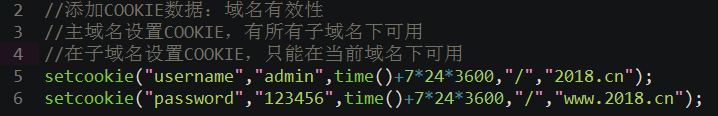
****



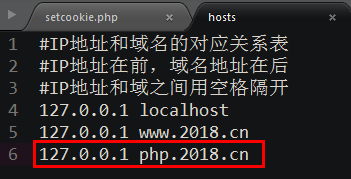


## 3、COOKIE域名有效性

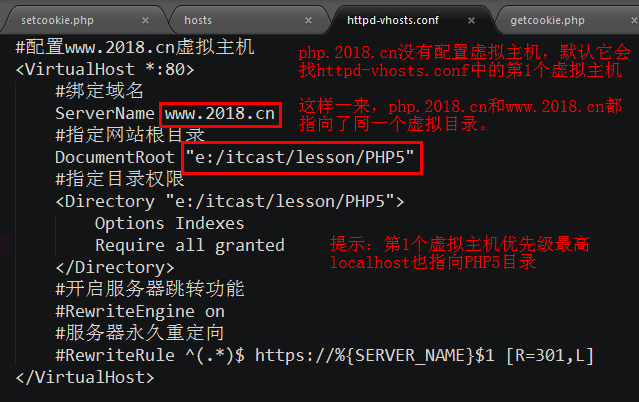
* 描述：给一个COOKIE指定访问的域名
* 语法：**bool setcookie ( string $name [, string $value [, int $expire = 0 [, string $path [, string $domain]]]] )**
* 提示：默认情况下，cookie只能在当前域名下有效。
* 域名划分：
  + 主域名：itcast.cn
  + 一级域名：www.itcast.cn
  + 二级域名：news.itcast.cn
  + 三级域名：php.news.itcast.cn
* 说明：如果COOKIE域名有效性，设置为主域名，则在所有域名下都可以使用。如果设置为子域名，则只能在子域名下使用该COOKIE。



**第一步：修改hosts文件添加二级域名**

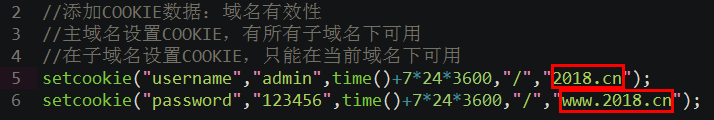


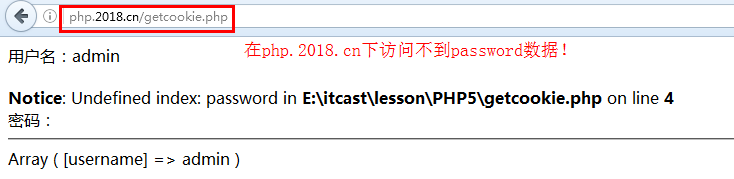
**第二步：将两个域名都指向同一个虚拟目录**

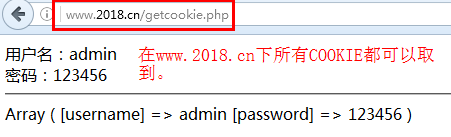


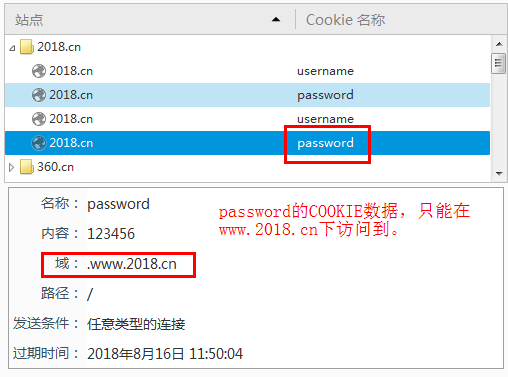


**第三步：测试结果**







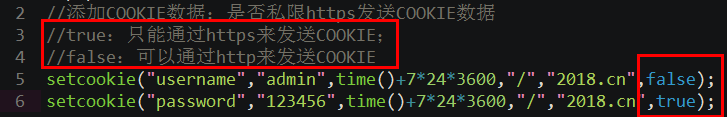


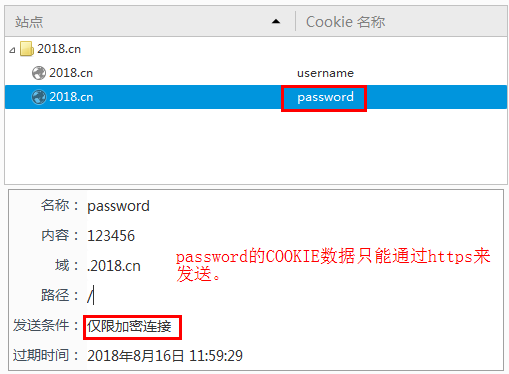
## 4、是否仅https安全连接才能发送cookie呢？

* 描述：是否仅限https来发送COOKIE数据，默认情况下，为http协议发送；
* 语法：**bool setcookie ( string $name [, string $value [, int $expire = 0 [, string $path [, string $domain [, bool $secure = false]]]]] )**
* https：是以安全为目标的HTTP通道，简单讲是HTTP的安全版。即HTTP下加入SSL层，HTTPS的安全基础是SSL，因此加密的详细内容就需要SSL 。









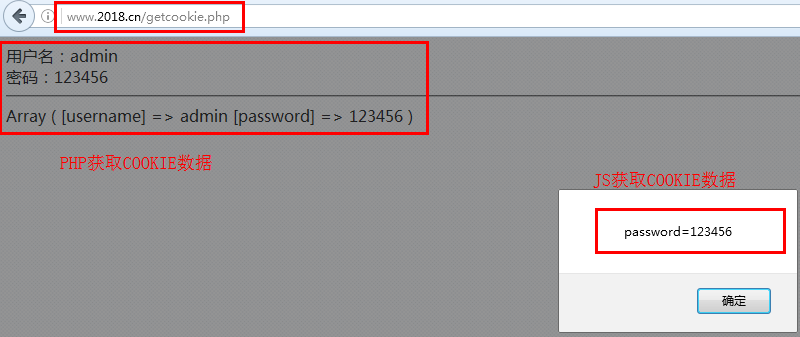
## 5、是否只能通过http协议来使用COOKIE呢？

* 描述：客户端的COOKIE除了可以通过HTTP协议来使用外，还可以使用JS来使用COOKIE。
* 语法：**bool setcookie ( string $name [, string $value [, int $expire = 0 [, string $path [, string $domain [, bool $secure = false [, bool $httponly = false ]]]]]] )**

**提示：除了COOKIE通过http发往服务器使用，JS也可以使用该COOKIE。**



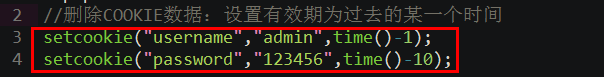




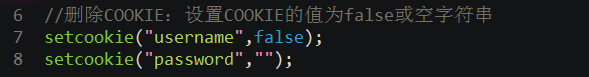
# 删除COOKIE数据

## 1、设置COOKIE有效性为过去的某一个时间

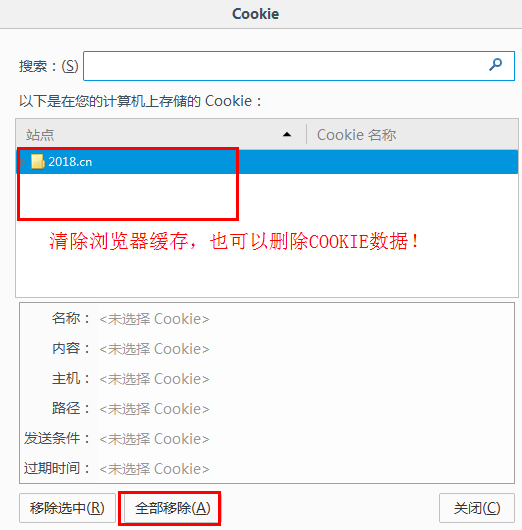
* 描述：设置有效期为过去的某一个时间
* 语法：**bool setcookie ( string $name [, string $value [, int $expire = 0 ]] )**



## 2、设置COOKIE的值为false或空字符串



## 3、清理浏览器缓存



# COOKIE的总结

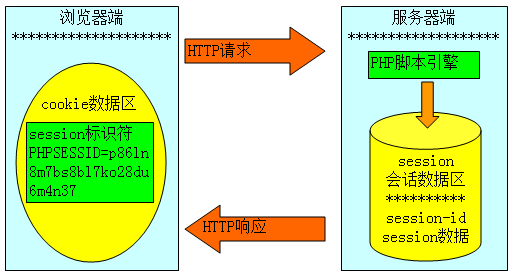
* COOKIE存储在客户端，相对来说不太安全。
* COOKIE存储的数据类型，只能是字符串。
* COOKIE存储的数据容量，大约只能存储4KB数据。
* 浏览器可以禁用缓存，也可以禁用COOKIE，则COOKIE功能就失效。

# SESSION概述

## 1、什么是SESSION？

* Session对象存储特定用户的会话数据；
* Session将会话数据存储在服务器端；
* Session是基于Cookie技术的，没有Cookie就没有Session；
* Session在整个用户会话中，一直存在下去；
* 一个用户会话时效：从用户登录开始，到用户登录结束；
* Session存储的数据量要比cookie大的多；
* Session存储的内容类型，不限于字符串；
* Session数据存储在服务器端，更安全，可靠。

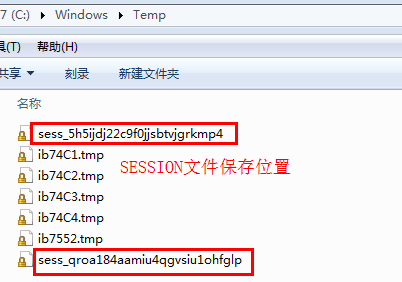
## 2、SESSION的工作原理



## 3、开启SESSION会话功能

* 描述：启动新会话或者重用现有会话 ；
* 语法：**bool session\_start (void )**
* 提示：$\_SESSION变量默认是不存在的，与$\_POST、$\_GET、$\_FILES不一样。因此，必须先开启SESSION，才能使用$\_SESSION变量；
* 提示：每个需要SESSION数据的页面，都要开启SESSION功能；
* 注意：SESSION功能，不能重复开启(同一个页面不能开2次)；
* 返回：成功开始会话返回 TRUE ，反之返回 FALSE；

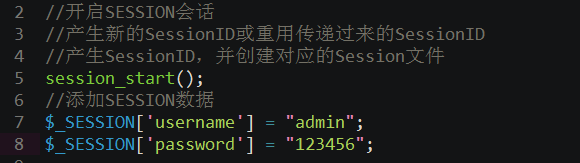
**提示：SESSION文件的保存位置 c:\windows\temp**

****

# SESSION操作

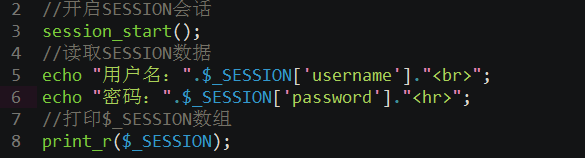
## 1、添加SESSION数据

* 描述：对Session的管理(增删改查)，都是通过超全局数组**$\_SESSION**进行的；
* 语法：**$\_SESSION[key] = value**
* 提示：一个网站不管添加多少个SESSION数据，最终在浏览器端只存储一个SESSION的ID值，该ID值是经过加密的，并且永不重复。Key只能是String类型的数据。
* SESSION文件在服务器端的存储位置：c:\windows\temp



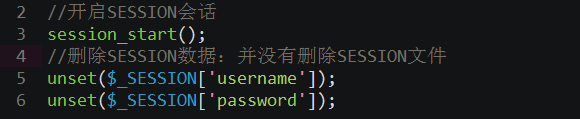
## 2、读取SESSION数据

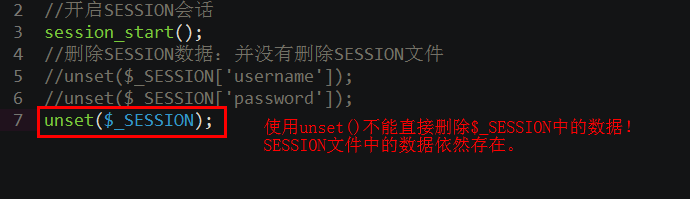
* 语法：**$value = $\_SESSION[key]**
* 提示：每次SESSION操作，都要先开SESSION功能；

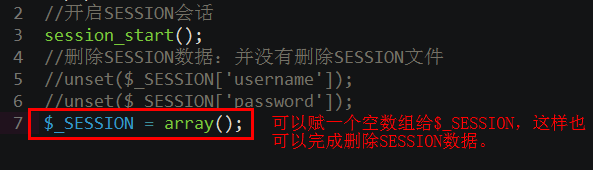


## 3、删除SESSION数据

* 使用 unset() 函数，删除一个SESSION数据；







## 4、销毁SESSION文件

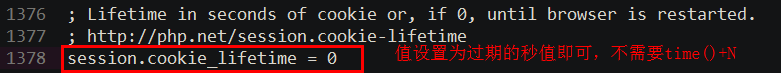
* 描述：删除当前的SESSION文件，不影响其它的SESSION文件。
* 语法：**bool session\_destroy ( void )**



# SESSION对应COOKIE的配置(php.ini)

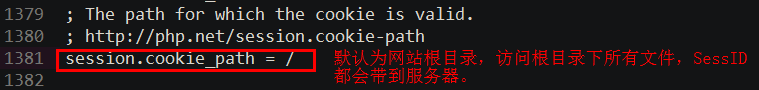
## 1、SESSION对应COOKIE过期时间设置

**修改php.ini配置项：session.cookie\_lifetime**



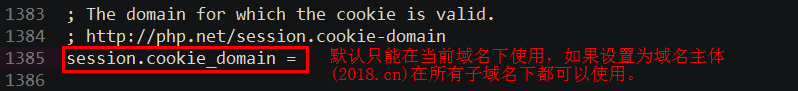
## 2、SESSION对应COOKIE有效路径设置

**修改php.ini配置项：session.cookie\_path**



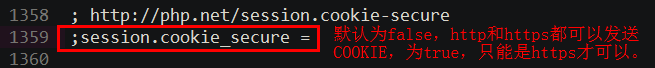
## 3、SESSION对应COOKIE域名有效性设置

**修改php.ini配置项：session.cookie\_domain**



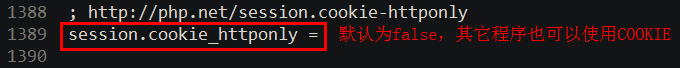
## 4、是否仅限https来发送SESSION对应的COOKIE数据

**修改php.ini配置项：session.cookie\_secure**



## 5、是否仅限http来使用SESSION对应的COOKIE数据

**修改php.ini配置项：session.cookie\_httponly**

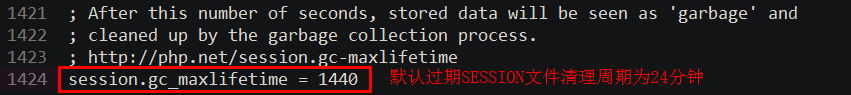


# SESSION垃圾回收机制

## 1、什么是SESSION的垃圾回收机制？

* SESSION垃圾回收， 就是将过期的SESSION服务器文件删除的机制；
* SESSION会自动删除那些过期的服务器端session数据区文件；
* **修改php的配置文件php.ini**。

## 2、垃圾回收的周期：session.gc\_maxlifetime



**提示：每24分钟清理一次。**

## 3、垃圾回收的概率：session.gc\_divisor

**回收周期：每24分钟，清理一次SESSION垃圾文件。**

**回收概率：1/1000，得到一个百分比，如果达到了100%就清理。每1000人访问，才会清理一次。**

首先判断时间是否达到24分钟，如果时间达到24分钟，再看访问的人数够不够1000。如果人数够1000，则进行清理，如果人数没有达到1000，则不清理，要等第2个24分钟。



