**COOKIE**

### 用途

1、网站所有的页面上都能访问到指定的值，比如：在demo2.php页面中访问demo2.php中$a的值。

2、将信息保存到客户端，比如：记住密码

### 语法

通过setcookie将cookie的值写到响应头中。

第一个参数：cookie名

第二个参数：cookie值

第三个参数：cookie的过期时间

第四个参数：有效目录

第五个参数：有效子域名

setcookie('name','tom'); //将name的值放到响应头中

cookie只能存放字符串。

### 临时性cookie

临时性cookie可以实现在不同的页面访问相同的值。关闭浏览器后cookie的值就消失。

setcookie('name','tom'); //将name的值放到响应头中

### 永久性cookie

### 通过设置setcookie的第三个参数，这个参数是一个时间戳。

将cookie通过文件的形式存储在客户端



不同的浏览器记录的cookie不能相互访问。

### 获取cookie的值

### $\_COOKIE[‘name’];

### 手动清空cookie文件

### cookie支持下标

cookie在浏览器上是没有数组的概念，只是浏览器请求到服务器，PHP将同名的值转换成数组。

setcookie('name[0]','qzk');

setcookie('name[1]','qzk1');

### 有效目录

默认cookie只能在当前目录及其后代目录中生效。

解决：setcookie(‘name’ , ’tt’ , 0, ’/’ )

### cookie的缺点

1. 因为cookie保存在浏览器上，所以安全性低
2. 可控性比较差，只能存放字符串，不能存放数组和对象
3. 因为数据存放在头信息中，所以增加请求数据的负载。
4. 大多数的浏览器对cookie有4K的限制。
5. Cookie可以被浏览器阻止

## session

1. session是基于cookie的技术
2. 默认情况下，会话不会自动启动，必须通过session\_start()开启会话。
3. 自动加载会话。在php的配置文件中找到session.auto\_start，将值设为1。并重启服务器。

Session.auto\_start = 1;

1. session\_id()：获取会话编号
2. Session\_name()：用户获取会话的名称
3. Sessoin可以保存除了资源以外的任意类型的数据
4. 会话不允许重复开启

### session的执行过程

1. 使用session\_start()函数开启会话，PHP从session仓库中加载已经存储的会话
2. 当session第一次启动时，一个唯一的标志id存储于cookie中。
3. 执行PHP脚本时候，将会话的值存放到当前的session文件中，获取从session文件中取值。
4. 当PHP脚本执行结束的时，没有被销毁的session值自动保存到服务器的session文件中。下次访问的时候可以再次访问。

### 关于session的配置

1. session.save\_path： session文件保存的路径
2. Cookie中保存的会话编号整站有效 session.cookid\_path = / 整站有效
3. 会话以文件方式存储 session.save\_handler = files
4. Session在服务器存储的时间（session的生命周期）session.gc\_maxlifetime = 1440 => 24m

### session\_destroy() 销毁会话

## session和cookie的区别

### 相同点

1. 都是会话技术，都有生命周期
2. 都是无状态性（当服务器执行完毕返回给客户端，服务器不再记录客户端的状态）

### 不同点

1. cookie客户端技术，session是服务器端技术
2. cookie保存的数据类型只有字符串，session可以保存除了资源以外的数据类型。
3. cookie的大小有4K限制，session没有
4. cookie可以被禁用，session不能被禁用

## 禁用cookie对session的影响

1. 如果禁用了cookie,基于cookie的所有技术默认情况下都无法使用
2. 可以通过设置改变session.use.only\_cookies = 1表示仅仅依赖cookie