会话控制

一、为什么使用回话控制机制

HTTP 最刚开始的时候作用就是为了实现文档的共享。无状态特性完全是为了快。

“ HTTP协议 ”是无状态协议，不能区分用户是否是从同一个网站上来的，同一个用户请求不同的页面不能看做是同一个用户

我们在浏览网站，通过 URL 访问 Web 服务器请求页面都需要使用“HTTP”协议实现，但是 HTTP 协议又是一个无状态的协议，当一个用户使用一个客户端进行访问同一个网站（网站服务器）不同页面时识别是同一个用户。

在一个页面访问产生的变量跳到另一个页面都会消失这个事HTTP协议的特征

会话控制的思想就是允许服务器跟踪同一客户端做出的连续请求。

会话控制：会话控制的思想就是允许服务器跟踪同一个客户端做出的连续请求

二、从逻辑上面区分变量的级别：目前页面之间传递变量方式：get,post,ajax,url,cookie,session

服务器中用户的登陆信息都在脚本中用变量保存的，登陆时要和数据库中数据进行匹配

1. 变量只能在同一个页面中使用 page 级 ，脚本执行结束就会被释放了。
2. get 和 post 方法传递， 两个页面之间传递（不是跟踪用户的首选），以链接的方式传递到另一个页面中。不适合大量变量 实例：单击修改操作可以修改对应信息,也能维持两个页面的状态

```
demo.php
<?php
        $var="hello";
        <a href="test1.php?var=<?php echo $var ?>">test</a>

test1.php
<?php
            $var=$_GET["var"];
            echo $var;
```

补足：

数据与表单处理脚本之间用问号?作为分隔符，表单包含多个控件用&分隔。

3、将变量存在文件或是数据库中 （所有人共用变量， 也不跟踪用户的首选）保存文件的方式。

```
文件中：
dmeo.php
        $var="hello";
        file_put_contents("var.txt",$var);
test.php
        echo file_get_contents("var.txt");
```

任何一个用户都可以访问,读取文件中变量的值,所有的用户共用变量,无法区分用户,用谁登陆都可以从文件中读取文件的变量值,这样就不是跟踪用户。假如一个A在一个购物网站中添加商品在购物车中对应有变量进行存放变量值,这样B用户登陆时同样读取文件中系统变量值,这样就不可逻辑,比如说按用户进行创建文件,那要创建多个文件.

```
数据库：
admin                       user                    hello
        1. 2. 3                     1. 2. 3                 1. 2. 3
```

不同的用户访问同一个网站不同的页面,变量值存入文件或者数据库中,这样在业务逻辑中需要进行判断那些变量是那个用户访问生成的变量值;如果按照用户访问进行存入数据库,那么每个访问你网站的产生的变量都需要存入数据库,需要足够大的空间,一个脚本存在很多变量,存入这些变量没有价值。

会话跟踪的方式：
HTTP是无状态的协议，所以不能维护两个事物间的状态。但一个用户在请求一个页面以后再请求另一个页面时，需要让服务期知道这是一个用户。总共有3种数据传递方式。
》超链接或者header()函数等重定向方式
》使用Cookie将用户的信息状态，存放在客户端的计算机中。
》使用Session将用户的信息状态，存放在服务器之中。

4.会话控制：跟踪用户

让一个用户访问每个页面，服务器都知道是那个用户在访问

三、Cookie 技术 cookie和session技术能全站维持状态,传值性较强

$_GET[] 获得以GET方法提交的变量数组

$_POST[] 获得以POST方法提交的变量数组

$_COOKIE[] 获取和设置当前网站的Cookie标识

$_SESSION[] 取得当前用户访问的唯一标识，以数组形式体现，
如sessionid以及自定义session数据.

Cookie 是服务器给客户端的礼物，前提条件是客户端要用一个文件进行保存。用来记录用户名，状态，值。当再次访问同一个服务器中其他 PHP 脚本时，就会自动携带 Cookie 中的数据一起访问（通过头信息传回给服务器）。在同一个网站（服务器上）的每个脚本都可以接受 Cookie 中的数据，并重新对登陆者的身份进行验证，不需要重新登陆了。

bool **setcookie** (string *$name* [, string *$value* [, int *$expire* = *0* [, string *$path*

setcookie('名','值','保存的时间')

注意：失效时间：若不写默认关闭浏览器即消失。
如果不写保存的时间，拿就是关闭浏览器的时候就删除cookie

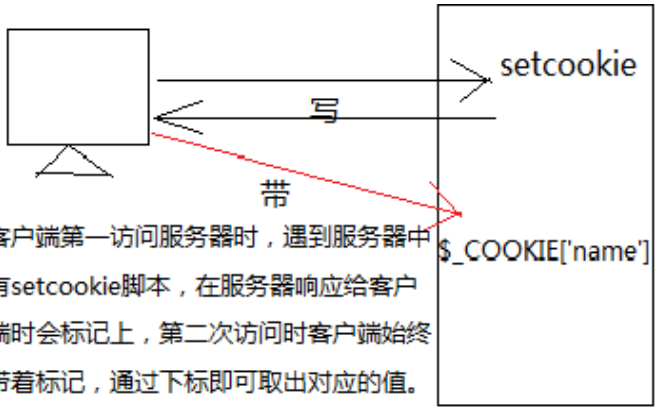
设置Cookie

```
setCookie('name','wowowowo',time()+60*60*24);
$_COOKIE['user']=11111;
var_dump($_COOKIE);
```

原理:

第一次请求服务器：

1. cookie是在http协议下，服务器或脚本可以维护客户端信息的一种方式。
2.Cookie是一种由服务器发送给客户端的片段信息，存储在客户端浏览器的内存或者硬盘上。常用于保存用户名，密码，个性化设置，个人偏好记录等。当用户访问服务器时，服务器可以设置和访问cookie的信息。
3.cookie保存在客户端，通常是IE或Firefox浏览器的cookie临时文件夹中，可以手动删除。注意：如果浏览器上cookie太多，超过了系统所允许范围，浏览器也会自动对它进行删除。

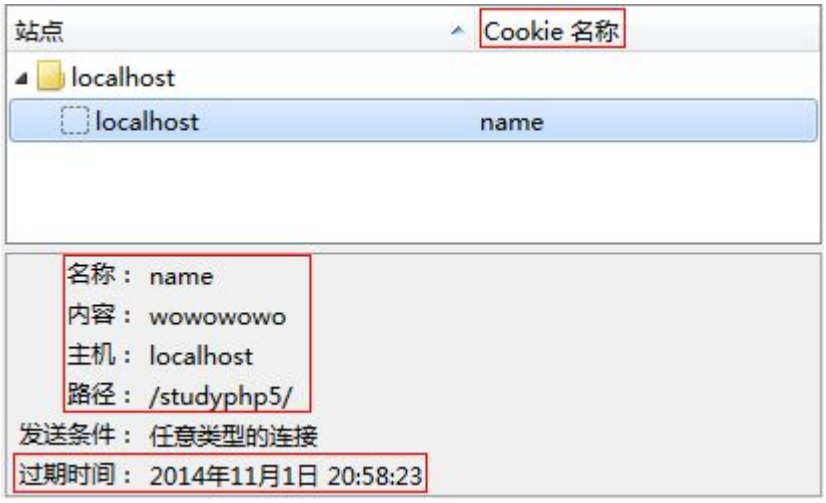


第二次访问服务器：

1、设置cookie的值必须使用setcookie设置不能用$_COOKIE['']这种方式设置
2、第一次设置完cookie之后在页面上面马上打印是打印不出来的，因为第一次只是通知客户端将这些值进行存储。

四、cookie 存放的位置

火狐浏览器：cookie 信息不是纯文本信息，他的信息是经过加密的，如果要看可以在浏览器中使用工具->选项->隐私->历史记录->使用自定义历史记录设置->显示 cookie



IE 浏览器：C:\Documents and Settings\用户名\Cookies

五、取得 setCookie 值

读取Cookie

```
setCookie('name','wowowowo',time()+60*60*24);
echo $_COOKIE['name'];
```

```
wowowowo
```

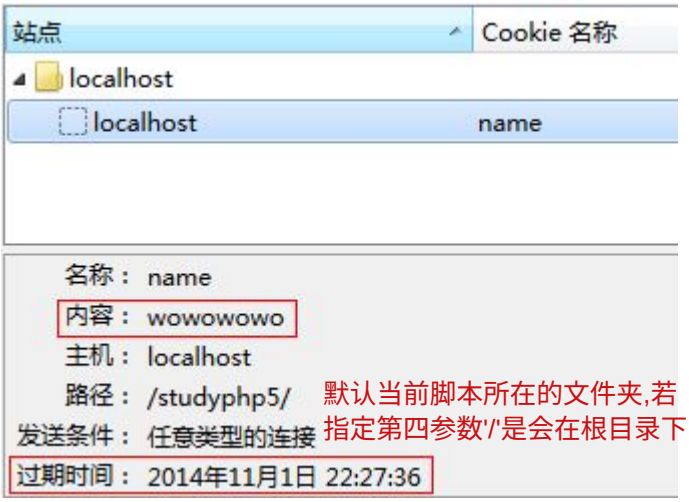
删除Cookie 六、干掉 setCookie 值

直接在指定文件夹下删除保存的 Cookie 文件即可

setcookie(名，值，当前时间-1)，告诉客户端这个值已经过期。

单设置此值时,服务器中只有指定路径下的网页或者程序可以存取Cookie
默认是从根目录开始设置的,一般不指定。

存在 Cookie 值



默认当前脚本所在的文件夹,若指定第四参数'/'是会在根目录下

```php
<?php
//使用cookie做一个计算器

//1. 尝试从cookie中获取
  $m = $_COOKIE['num'];

//2. 加加
  $m++;

//3. 放回到cookie中
  setCookie("num",$m,time()+3600,"/");

//4. 输出
  echo "<h3>访问次数：{$m}</h3>";
```

```
setCookie('name','wowowowo',time()+60*60*24);
setCookie('name','',time()-1);
```

只要指定让Cookie过期(当前时间过一秒也是过期)即可

七、保存 setCookie 值为数组（关联和索引）

3、cookie里面能存数组吗？能，能存关联数组吗？能 能存索引数组吗？ 能
4、cookie里面如果设置的是索引数组并且不指定索引的下标的话，那么下标永远是0不会自动增长。
关联数组 ： setcookie('lx[phone]','值',时间)
索引数组： setcookie('lx[]','值'，时间);

说明第 4 点：

```
setCookie('name[]','wowowowo1',time()+60*60*24);
setCookie('name[]','wowowowo2',time()+60*60*24);
setCookie('name[]','wowowowo3',time()+60*60*24);
setCookie('name[]','wowowowo4',time()+60*60*24);

print_r($_COOKIE);
```

Array ([name] => Array ([0] => wowowowo4))

下标一定要从 0 开始，确保还有值，其他有值有指定下标，则取最后一个值

```
setCookie('name[1]','wowowowo1',time()+60*60*24);
setCookie('name[2]','wowowowo2',time()+60*60*24);
setCookie('name[3]','wowowowo3',time()+60*60*24);
setCookie('name[4]','wowowowo4',time()+60*60*24);

print_r($_COOKIE);
```

Array ([name] => Array ([0] => wowowowo4 [1] => wowowowo1 [2] => wowowowo2 [3] => wowowowo3 [4] => wowowowo4))

直接指定下标从 0 开始

```
setCookie('name[0]','wowowowo1',time()+60*60*24);
setCookie('name[1]','wowowowo2',time()+60*60*24);
setCookie('name[2]','wowowowo3',time()+60*60*24);
setCookie('name[3]','wowowowo4',time()+60*60*24);

print_r($_COOKIE);
```

Array ([name] => Array ([0] => wowowowo1 [1] => wowowowo2 [2] => wowowowo3 [3] => wowowowo4))

关联数组，无需使用双引号，使用就当初字符了

```
setCookie('name[1]','wowowowo1',time()+60*60*24);
setCookie('name[a]','wowowowo2',time()+60*60*24);
setCookie('name["bb"]','wowowowo3',time()+60*60*24);
setCookie('name[3]','wowowowo4',time()+60*60*24);

print_r($_COOKIE);
```

```
Array ( [name] => Array ( [1] => wowowowo1 [a] => wowowowo2 ["bb"] => wowowowo3 [3] => wowowowo4 ) )
```

八、SESSION

1、开启session(只有开启session之后，才能告诉PHP调度服务器的session机制)

```
session_start()
```

session_start()的作用：

1、第一时会向客户端发送sessionid，并且创建session的文件

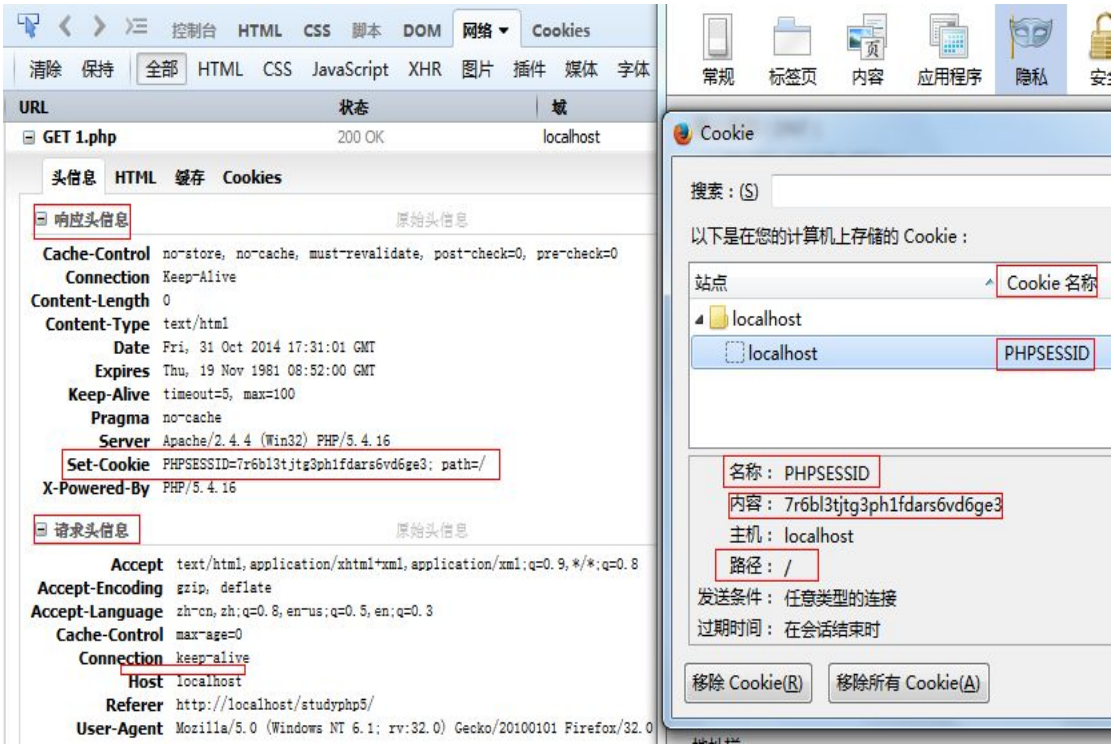
2、以后使用的时，根据http协议和cookie机制将传过来的session id的值。找服务器里面的session文件

在客户端存有 session id ,在服务器中保存着 session 文件保存着用户的信息，只要客户端带过来 session id 就能在服务器中找到对应的用户的信息。

设置session

```
session_start();
$_SESSION['username']="admin@123";
```

第一次访问服务响应了 PHPSESSION



cookie和session在php中的使用区别:

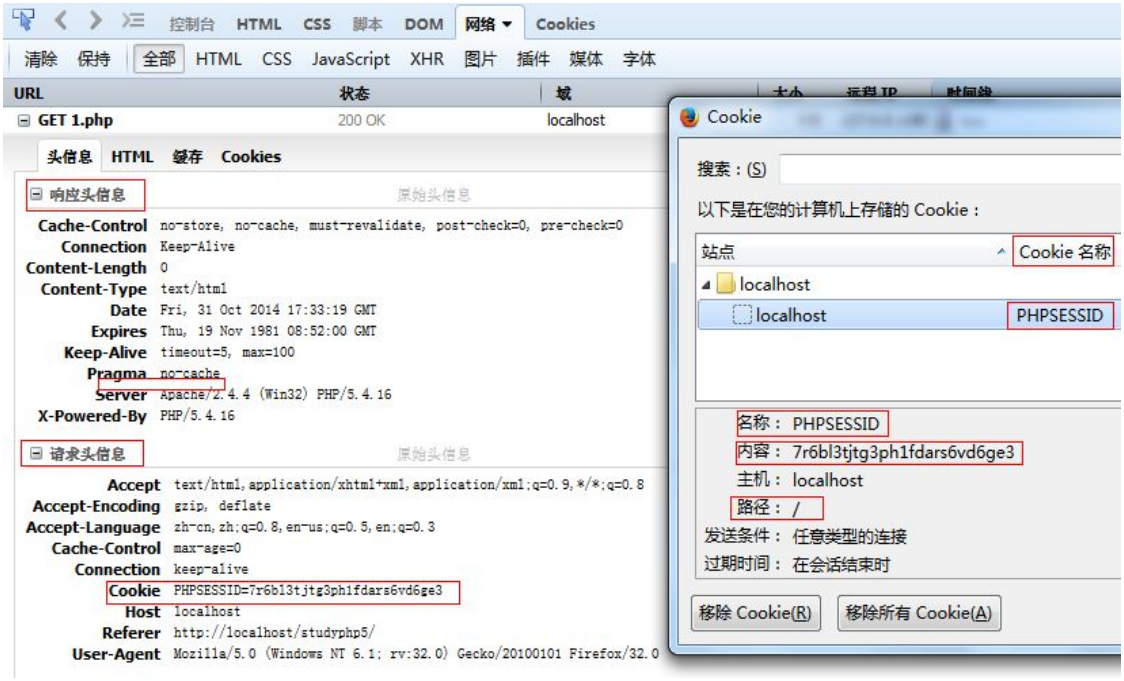
1.cookie和session都可以暂时保存在多个页面中使用的变量，但是它们有本质的差别。

cookie存放在客户端浏览器中，

session保存在服务器上。

2.它们之间的联系是session ID 一般保存在cookie中，或者放在URL上。

第二次访问服务器请求信息有 PHPSESSION



读取session

2、怎么存储值　只要开启session机制页面都可以通过下标取值，$_SESSION是特殊数组

```
$_SESSION['你要存的下标']=你要存的值
PHPSESSID=7r6bl3tjtg3ph1fdars6vd6ge3
```

关闭浏览器时session_id立即消失，但是session信息继续保留，直到session自动回收机制清空。

```
session.save_path = "路径" php.ini中的这个选项就是session文件存储的路径
```

删除session

3、删除session怎么办？

1、开启session_start();
2、清空session数组 unset();
3、使用session_destroy();来销毁session的文件
4、销毁客户端的session ID
setcookie（'下标','值',time()-1）　PHPSESSID

```
// 移除session信息
session_start(); //启用session会话技术

//删除session中的登陆信息
//$_SESSION["loginuser"]=null;
unset($_SESSION["loginuser"]);//ok

//session_destroy(); //销毁当前session（所有信息如购物车,登陆等）
```

```
php.ini中的session.name = PHPSESSID选项就是 所对应的cookie的下标
```

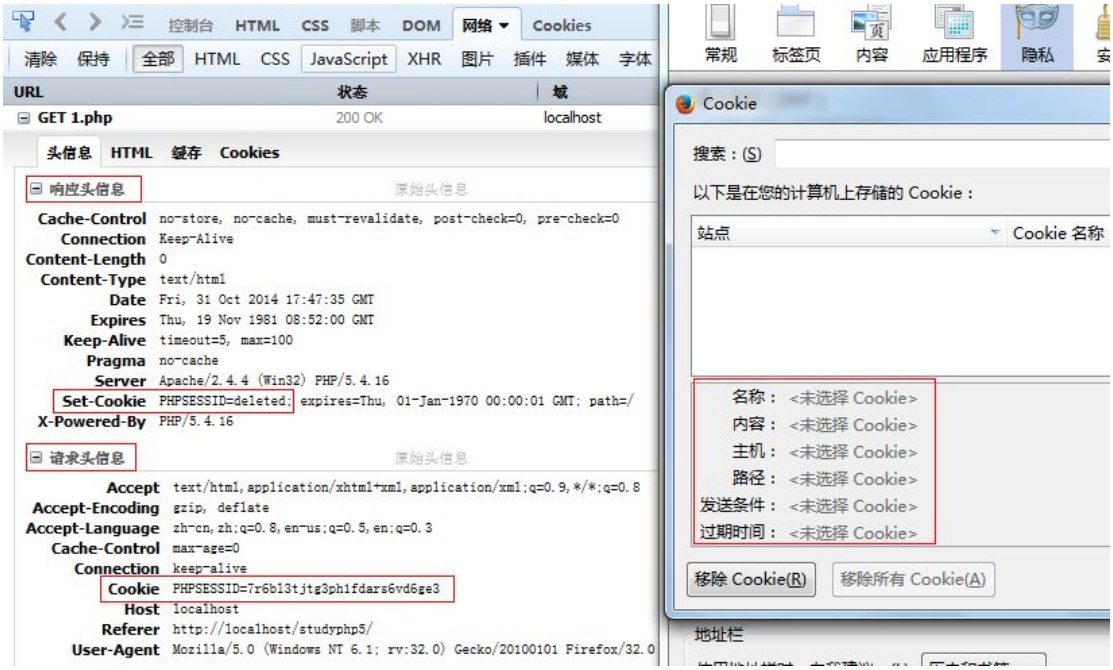
注意：

1、如果在文件夹里面删除PHPSESSION对应的这个cookie文件的时候一定要加上'/'根目录,因为设置session所对应的session id 这个cookie值的路径是根目录

```
<?php
//使用session做一个计算器
session_start(); //启用session会话

//1. 尝试从session中获取
  $m = $_SESSION['num'];

//2. 加加
  $m++;

//3. 放回到session中
   $_SESSION['num']=$m;

//4. 输出
  echo "<h3>访问次数：{$m}</h3>";
```

```
//开启SESSION
session_start();

//清空SESSION值
$_SESSION=array();

//删除客户端的在COOKIE中的Sessionid
if(isset($_COOKIE[session_name()])){
    setCookie(session_name(), '', time()-3600, '/');
}
//彻底销毁session
session_destroy();
```

Session的自动回收机制：
php.ini中相关的配置
session.cookie_lifetime=0; 关闭浏览器相应的cookie文件即被删除
session.gc_maxlifetime;设置过期session时间，默认1440秒（24分钟）
session.gc_probability/session.gc_divisor;启动垃圾回收机制的概率（建议值为1/1000～5000）



进一步理解：

```php
//开启SESSION
session_start();
//清空SESSION值
$_SESSION=array();
print_r($_SESSION).'<br>';
echo session_name().'<br>';
echo session_id().'<br>';
//删除客户端的在COOKIE中的Sessionid
if(isset($_COOKIE[session_name()])){
    setCookie(session_name(), '', time()-3600, '/');
}
echo session_name().'<br>';
echo session_id().'<br>';
//彻底销毁session
session_destroy();
echo session_name().'<br>';
echo session_id().'<br>';
```

```
Array ( ) PHPSESSID
8vbf7vu2bu9tktn9b5ro8n4d65
PHPSESSID
8vbf7vu2bu9tktn9b5ro8n4d65
PHPSESSID
```

登陆注册注意：

php.ini文件和Session有关的几个常用配置选项

- session.auto_start = 0 ; 在请求启动时初始化session
- session.cache_expire = 180 ; 设置缓存中的会话文档在 n 分钟后过时
- session.cookie_lifetime = 0 ; 设置按秒记的cookie的保存时间， 相当于设置Session的过期时间，为0时表示直到浏览器被重启
- session.auto_start=1，这样就无需每次使用session之前都要调用session_start()。但启用该选项也有一些限制，如果确实启用了 session.auto_start，则不能将对象放入会话中，因为类定义必须在启动会话之前加载以在会话中重建对象
- session.cookie_path = / ; cookie的有效路径
- session.cookie_domain = ; cookie的有效域
- session.name = PHPSESSID; 用在cookie里的session的名字
- session.save_handler = files ; 用于保存/取回数据的控制方式
- session.save_path = /tmp ; 在 save_handler 设为文件时传给控制器的参数， 这是数据文件将保存的路径.
- session.use_cookies = 1 ; 是否使用cookies

```php
$result = mysql_query($sql);

/*mysql_query() 对 SELECT语句返回一个资源标识符(资源永远为真)，如果查询执行
不正确(SQL语句出错)则返回 FALSE,用户名和密码出错一样返回资源(为真)。对于其它
类型的 SQL 语句，mysql_query() 在执行成功时返回 TRUE，出错时返回 FALSE*/

$userDetail = mysql_fetch_assoc($result);//在数据库中按照条件返回结果表示为真,没有对应的行返回FALSE

//根据结果集来判断是否登录成功，登陆成功我就保存用户信息变量保存到$_SESSION中以便带到其他页面中
if($userDetail){
    $_SESSION['id']= $userDetail['id'];//存储用户id
    $_SESSION['username'] = $userDetail['username'];//存储用户名
    //我这块多加一个标志位，如果设置了这个，说明用户登录成功了，如果没有那就说明压根没有登录
    $_SESSION['isLogin'] = true;
    header('location:./index.php');

}else{
    header('location:./login.html');
}
```

```php
<h3>浏览购物车</h3>
<table width="500" border="1">
    <tr>
        <th>id</th>
        <th>商品名称</th>
        <th>单价</th>
        <th>数量</th>
        <th>小计</th>
        <th>操作</th>
    </tr>
    <?php
        //判断购物车中是否有商品
        if($_SESSION['shoplist'] && count($_SESSION['shoplist'])>0){
            foreach($_SESSION['shoplist'] as $shop){
                echo "<tr>";
                echo "<td>{$shop['id']}</td>";
                echo "<td>{$shop['name']}</td>";
                echo "<td>{$shop['price']}</td>";
                echo "<td>{$shop['num']}</td>";
                echo "<td>".($shop['price']*$shop['num'])."</td>";
                echo "<td><a href='cartaction.php?a=del&id={$shop['id']}'>删除</a></td>";
                echo "</tr>";
            }
        }
    ?>
</table>
```

```php
<?php include("menu.php");//导入导航栏 ?>
<center>
<form action="cartaction.php?a=add" method="post">
    <input type="hidden" name="id" value="1001"/>
    <input type="hidden" name="name" value="数码照相机"/>
    <input type="hidden" name="price" value="1600.00"/>
    数码照相机: 1600.00元   
    <input type="submit" value=" 购 买 ">
</form><p />
```

```php
<?php
//购物车信息处理页
session_start();//启用session会话技术

//获取参数a的值，并判断做对应操作

switch($_GET['a']){
    case "add": //向购物车中添加商品信息
        //获取要放入购物的信息
        $id=$_POST['id'];
        $shop['id']=$_POST['id'];
        $shop['name']=$_POST['name'];
        $shop['price']=$_POST['price'];
        $shop['num']=1; //购买数量
        //将当前的商品信息放入到购物车中
        //选判断购物车中是否已存在此商品
        if(isset($_SESSION['shoplist'][$id])){
            //若存在则数量加1
            $_SESSION['shoplist'][$id]['num']+=1;
        }else{
            //若不存在，则作为新商品放进去
            $_SESSION['shoplist'][$id]=$shop;
        }
        break;

    case "del": //删除购物车中的某个商品
        unset($_SESSION['shoplist'][$_GET['id']]);
        break;

    case "clear": //清空购物车
        unset($_SESSION['shoplist']);
        break;
}

header("Location:show.php");
```

```php
<div align="center"><br>
  § <a href="shop.php">我要购物</a>
  § <a href="show.php">查看购物车</a>
  § <a href="cartaction.php?a=clear">清空购物车</a> §
</div>
```