**自动登陆**

---登录时记住密码：  
FrontendController控制器为前台基础控制器

class LoginController extends FrontendController

{

/\* 会员登录验证 \*/

public function check()

{

// ...

//登录成功后

$this->remember(); //两周内记住账号

// ...

}

/\* 两周内记住账号 \*/

private function remember()

{

$remember\_me = I('remember\_me', null, 'int');

if (!$remember\_me) {

return false;

}

$mobile = I('post.mobile', '', '/^1[34578]\d{9}$/');

$pwd = I('post.pwd', '');

if (empty($mobile) || empty($pwd)) {

return false;

}

$pwd = md5($pwd);

$account = $mobile . '|' . $pwd;

$account = encrypt($account, '', 1209600); //两周时间=60X60X24X14秒 加密存储

cookie('\_RM', $account, array('expire' => 1209600, 'httponly'=>1)); //写入客户端缓存

}

//...

}

---自动登录的代码：

class FrontendController extends Controller

{

/\* 两周内自动登录 \*/

public function \_initialize()

{

if (!IS\_LOGIN && $account = cookie('\_RM')) {

$account = decrypt($account); //解密

list($mobile, $pwd) = explode('|', $account); //解析出用户名和密码

$user = D('User');

$id = $user->getUserField(array('mobile' => $mobile, 'pwd' => $pwd), 'id'); //判断是否存在这个用户

if (!is\_null($id)) {

$user->autoLogin($id); //传入用户ID，实现自动登录

} else {

cookie('\_RM', null); //记录密码的缓存名称

}

}

}

//....

}

---其他：  
Home模块的配置

/\* SESSION 和 COOKIE 配置 \*/

'SESSION\_PREFIX' => 'home', //session前缀

'COOKIE\_PREFIX' => 'home\_', // Cookie前缀 避免冲突

/\* 数据缓存 \*/

'DATA\_CACHE\_PREFIX' => 'vc\_',

Admin模块的配置：

/\* SESSION 和 COOKIE 前缀 \*/

'SESSION\_PREFIX' => 'admin',

'COOKIE\_PREFIX' => 'admin\_',

/\* 数据缓存 \*/

'DATA\_CACHE\_PREFIX' => 'vc\_',

用户登录时，session的处理：

/\* 写入session 并数字签名 \*/

$user\_data = array(

'id' => $userid,

'mobile' => $mobile,

'last\_time' => $last\_time,

);

session('user', $user\_data);

退出时的操作：

class UserController extends FrontendController {

//前台

public function logout()

{

session('user', null);

cookie('\_RM', null); //清空 两周内自动登录 缓存

$this->redirect('/');

}

//...

}

**记住用户登录状态**

登录后在用户表随机生成一个token存入cookie

然后用token去取用户信息,这样可以防止同时两个地方登录帐

如果不需要防止同时登录就另外建张表与用户关联

$token = md5(md5($user\_name.md5(date('Y/m').$password.$salt).$user\_id);

if($\_COOKIE["user\_token"]==$token){

//保持登录

}

**[应用Session变量控制用户登录时间](https://www.cnblogs.com/haohaoyuan/p/7797820.html)**

1、创建login\_ok.php页面，在页面中通过查询数据库来判断用户名密码是否正确，如果正确，则为Session变量赋值

<?php

error\_reporting(0);

session\_start(); //开启session

header("content-type:text/html;charset=utf-8");//设置编码格式

include 'conn.php'; //包含数据库连接文件

$name=$\_POST['user'];

$pwd=$\_POST['pwd'];

$sql=mysql\_query("select \*from information where name='".$name."' and passworld='".$pwd."'");//执行sql语句判断数据库内是否含有

if(mysql\_num\_rows($sql)>0){

$\_SESSION['name']=$name;

$\_SESSION['time']=time();

echo "<script> alert('登录成功！'); location='index.php'</script>"; //提示登录成功

}

else{

echo "<script> alert('用户名密码错误！'); location='login.php'</script>"; //提示用户名密码错误

}

2、创建index.php页面，在页面中进行判断，如果Session变量有值且登录时间没有超过10分钟显示欢迎回来，如果十分钟内未操作，则显示登录超时，请重新登录

<?php

error\_reporting(0);

session\_start();

if($\_SESSION['time']==""){

echo "<script> alert('您无权限查看，请登录！'); location='login.php'</script>"; //不允许直接访问

}

elseif((time()-$\_SESSION['time'])<60){ //如果登录时间没有超过1分钟

$\_SESSION['time']=time(); //将当前的时间戳赋值给Session变量

**[通过session\_id限制仅一个用户登录](https://www.cnblogs.com/haohaoyuan/p/7797941.html)**

我们实现该功能只需要在普通用户验证的基础上，加一层session\_id的验证就可以了。我们在用户登录的时候，将此时的session\_id写入数据表，session\_id在本次session有效期是不会变化的。如果在别的地方，登录了该账户，则新的session\_id值会写入数据表，则在验证用户信息的时候，当前用户的session\_id和数据表里面的session\_id不一致。

| **id** | **name** | **password** | **session\_id** |
| --- | --- | --- | --- |
| 1 | root | 63a9f0ea7bb98050796b649e85481845 | 3olcdjkj5jjaq2u9t30mbuna96 |

核心代码：

indexController.class.php

<?php

class indexController extends Controller{

function \_\_construct(){

parent::\_\_construct();

}

// 显示用户信息页面

function index(){

$loginController = new loginController();

$loginController->isLogin();

$this->smarty->assign("name",$\_SESSION['name']);

$this->smarty->assign("id",$\_SESSION['id']);

$this->smarty->display("info.html");

}

}

loginController.class.php

<?php

class loginController extends Controller{

private $loginModel;

function \_\_construct()

{

parent::\_\_construct();

$this->loginModel = new loginModel();

}

//@return bool\* 判断当前登录用户是否合法

function isLogin(){

$res = $this->loginModel->getUserInfoById($\_SESSION['id']);

if(empty($res)){

echo "未登录";exit;

}else{

$sql\_token = md5($res['name'].$res['password']);

if($sql\_token != $\_SESSION['token']){

echo "用户验证失败";exit;

}else{

if(session\_id()!=$res['session\_id']){

echo "该账户已在别处登录";exit;

}else{

return true;

}

}

}

}

// 执行登录操作

function action(){

$name = $\_POST['name'];

$password = md5($\_POST['password']);

$session\_id = session\_id();

$res = $this->loginModel->loginCheck($name,$password,$session\_id);

if($res){

$\_SESSION['id'] = $res['id'];

$\_SESSION['name'] = $name;

$\_SESSION['token'] = md5($name.$password);

header("Location:http://login.com/index.php/index/index");

}else{

header("Location:http://login.com/index.php/login/form");

}

}

}

loginModel.class.php

<?php

class loginModel extends Model{

function getUserInfoById($id){

$sparam = array('id','name','password','session\_id');

$wparam = array("id[=]"=>$id);

$data = $this->select("admin",$sparam,$wparam);

if(empty($data)){

return false;

}else{

return $data[0];

}

}

function loginCheck($name,$password,$session\_id){

$sparam = array('id','name','password',"session\_id");

$wparam = array("name[=]"=>$name);

$data = $this->select("admin",$sparam,$wparam);

if(empty($data)){

return false;

}else{

$info = $data[0];

if($info['password']!=$password){

return false; //验证失败

}else{

//验证成功，更新session

$newdata = ["session\_id"=>$session\_id];

$wparam = ["id[=]"=>$info['id']];

$this->update("admin",$newdata,$wparam);

return $info;

}

}

}

}