2-9管理员登录与视图分离

1. 课堂案例

课堂讲解

一、上节回顾

数据库的操作

添加

修改

查看

删除

二、学习目标

1、session与cookie

2、视图分离

2-1、include

2-2、require

3、初始化文件

一个项目或系统中，通过前后端分开设计师完成。如果把页面的html与php写在一起，那么就表示1个文件由2个人负责，若这2个人同时更新文件，则会造成混乱，为了方便管理，一般会把视图（前端html）分开放置，并由php加载。

三、教学过程描述

1、会话（session）

PHP session 变量用于存储关于用户会话（session）的信息，或者更改用户会话（session）的设置。Session 变量存储单一用户的信息，并且对于应用程序中的所有页面都是可用的。

PHP Session 变量

您在计算机上操作某个应用程序时，您打开它，做些更改，然后关闭它。这很像一次对话（Session）。计算机知道您是谁。它清楚您在何时打开和关闭应用程序。然而，在因特网上问题出现了：由于 HTTP 地址无法保持状态，Web 服务器并不知道您是谁以及您做了什么。

PHP session 解决了这个问题，它通过在服务器上存储用户信息以便随后使用（比如用户名称、购买商品等）。然而，会话信息是临时的，在用户离开网站后将被删除。如果您需要永久存储信息，可以把数据存储在数据库中。

Session 的工作机制是：为每个访客创建一个唯一的 id (UID)，并基于这个 UID 来存储变量。UID 存储在 cookie 中，或者通过 URL 进行传导。

1-1、session_start

在使用session前需要开启会话

注意：session_start() 函数必须位于 <html> 标签之前：

```php
<?php
session_start();
?>

<html>
<body>

</body>
</html>
```

1-2、储存session

存储和取回 session 变量的正确方法是使用 PHP $_SESSION 变量：

```php
<?php
session_start();
// 存储 session 数据
$_SESSION['views']='小夏';
?>

<html>
<head>
    <meta charset="utf-8">
    <title>PHP教程</title>
</head>
<body>

<?php
// 检索 session 数据
echo "浏览量：". $_SESSION['views'];
?>

</body>
</html>
```

1-3、销毁 Session

如果您希望删除某些 session 数据，可以使用 unset() 或 session_unset()或者session_destroy() 函数。

1、session_unset

释放当前在内存中已经创建的所有$_SESSION变量，但不删除session文件以及不释放对应的session id

```php
<?php
session_start();
session_unset();
```

2、unset

unset() 函数用于释放指定的 session 变量：

```php
<?php
session_start();
if(isset($_SESSION['views']))
{
    unset($_SESSION['views']);
}
?>
```

3、session_destroy

session_destroy()　销毁当前会话中的全部数据，　但是不会重置当前会话所关联的全局变量，也不会重置会话 cookie。如果需要再次使用会话变量，必须重新调用 session_start() 函数。

为了彻底销毁会话，比如在用户退出登录的时候，必须同时重置会话　ID。　如果是通过　cookie 方式传送会话 ID 的，那么同时也需要 调用 setcookie() 函数来 删除客户端的会话 cookie。

注释：session_destroy() 将重置 session，您将失去所有已存储的 session 数据。

2、cookie

Cookie 是什么？

cookie　常用于识别用户。cookie 是一种服务器留在用户计算机上的小文件。每当同一台计算机通过浏览器请求页面时，这台计算机将会发送 cookie。通过 PHP，您能够创建并取回 cookie 的值。

2-1、setcookie

setcookie() 函数用于设置 cookie。

注释：setcookie() 函数必须位于 <html> 标签之前。

语法：

setcookie(name,value,expire,path,domain,secure)。

name　必需。规定 cookie 的名称。

value　必需。规定 cookie 的值。

expire　可选。规定 cookie 的有效期。

path　可选。规定 cookie 的服务器路径。

domain 可选。规定 cookie 的域名。

secure　可选。规定是否通过安全的 HTTPS 连接来传输 cookie。

2-2、$_COOKIE

$_COOKIE 变量用于取回 cookie 的值。

2-3、删除cookie

当删除 cookie 时，您应当使过期日期变更为过去的时间点。

```
$value = "小夏带你们游玩cookie过时了=".(time()+60);
setcookie("TestCookie",$value);
// 设置 cookie 过期时间为过去 1 小时
setcookie("TestCookie", "", time()-3600);
```

3、加密函数

3-1、MD5

这是一种不可逆加密，执行如下的代码

```
$password = '123456';
echo md5($password);
```

得到结果是e10adc3949ba59abbe56e057f20f883e

3-2、Crypt

crypt() 返回一个基于标准 UNIX DES 算法或系统上其他可用的替代算法的散列字符串。

参数

str -- 待散列的字符串。

salt -- 可选的盐值字符串。如果没有提供，算法行为将由不同的算法实现决定，并可能导致不可预料的结束。

这是也一种不可逆加密，执行如下的代码

```
<?php
$password = '123456';
$salt = "test";// 只取前两个
echo crypt($password, $salt);
```

3-3、Sha1

参数

str -- 输入字符串。

raw_output -- 如果可选的 raw_output 参数被设置为 TRUE，那么 sha1 摘要将以 20 字符长度的原始格式返回，否则返回值是一个 40 字符长度的十六进制数字。

这是也一种不可逆加密，执行如下代码：

```
$password = '123456';
echo sha1($password);
```

得到的结果是7c4a8d09ca3762af61e59520943dc26494f8941b

3-4、Urlencode

这是一种可逆加密，urlencode方法用于加密，urldecode方法用于解密，执行如下代码：

```
$url = 'http://www.xxx.com/CraryPrimitiveMan/'
$encodeUrl = urlencode($url);
echo $encodeUrl . "\n";// 如果是在网页上展示的，就将\n修改为<br/>
echo urldecode($encodeUrl);
```

得到的结果如下

```
http%3A%2F%2Fwww.xxx.com%2FCraryPrimitiveMan%2F
http://www.xxx.com/CraryPrimitiveMan/
```

3-5、Base64信息编码加密

base64_encode加密base64_decode解密是可逆的

代码如下：

```
$name = 'CraryPrimitiveMan';
$encodeName = base64_encode($name);
echo $encodeName . "<br>";
echo base64_decode($encodeName);
```

结果：

```
Q3JhcnlQcmltaXRpdmVNYW4=
CraryPrimitiveMan
```

4、视图分离

4-1、include_once

语句在脚本执行期间包括并运行指定文件。此行为和 include() 语句类似，唯一区别是如果该文件中的代码已经被包括了，则不会再次包括。如同此语句名字暗示的那样，只会包括一次。

4-2、include

这个函数一般是放在流程控制的处理部分中。PHP 程序网页在读到 include 的文件时，才将它读进来。这种方式，可以把程序执行时的流程简单化。

4-3、require_one

require_once() 语句在脚本执行期间包括并运行指定文件。此行为和 require() 语句类似，唯一区别是如果该文件中的代码已经被包括了，则不会再次包括。

4-4、require

这个函数通常放在 PHP 程序的最前面，PHP 程序在执行前，就会先读入 require 所指定引入的文件，使它变成 PHP 程序网页的一部份。常用的函数，亦可以这个方法将它引入网页中。

四、小结

session保存在网站的服务器；cookie保存在客户端（用户浏览器）

session与cookie都有时间限制，但session的时间限制一般较短(默认20分钟)，且会话中断后结束；cookie时间较长，一般用于长时间记录，会话结束后可留待下次使用。

session安全性较高；cookie安全性差。

2. 课堂练习

1、分离上节课的PHP与mysql

2、编写会话和cookie

3. 课后练习

1、用session创建一个简单计数器。

2、设计用户登录表

3、创建用户登录

4. 资料扩展

1、加密函数

phpass

phpass是一个开源的类库，可以让我们方便的使用bcrpt加密算法

下载地址分别是：

CSDN：http://download.csdn.net/detail/xiao_bai8/9565233

官网：http://www.openwall.com/phpass/

```php
// 引入类文件
require 'PasswordHash.php';

// 初始化散列器为不可移植，安全性更好。false加密字符串是60位，true加密字符是34位
$hasher = new PasswordHash(8, true);

// 执行加密
$hashedPassword = $hasher->HashPassword('test123');

// 输出加密后的字符和对应的字符串长度
echo $hashedPassword;
echo '<br>';
echo strlen($hashedPassword).'<br>';

// 检查密码是否正确，第一个参数是密码的明文，第二个是加密后的字符串
$hasher->CheckPassword('test123', $hashedPassword);  // false

$hasher->CheckPassword('test1234', $hashedPassword);  // true
```

2、错误分析

2-1、错误类型

1、语法错误：Parse error，导致代码无法编译，1次只出1个，通常是因为缺少";","()"等导致

2、致命错误：Fatal error，导致代码无法往下执行，通常发生在操作系统级别上例如分配内存时或调用未定义函数发生的错误

3、警告：Warning，程序执行出现错误，与Fatal error的区别是程序依旧可以执行，通常被保留为环境错误，如无法执行脚本访问数据库

4、通知（潜在错误）：Notice，可能存在潜在的错误，例如使用了没有声明的变量或数组索引，程序可以继续运行

2-2、错误种类

E_ERROR：通常会显示出来，也会中断程序执行。

E_WARNING：通常都会显示出来，但不会中断程序的执行。

E_NOTICE：在脚本正常运行下发生的代码错误。

E_PARSE：语法解析错误。

E_CORE_ERROR：在PHP启动时发生的致命错误。

E_CORE_WARNING：报告在PHP启动时发生的非致命性错误。
E_COMPILE_ERROR：编译时发生的致命错误，指出脚本的错误。
E_USER_ERROR：用户产生的错误信息。

E_USER_WARNING：用户产生的警告信息。

E_USER_NOTICE：用户引发的注意消息。

E_STRICT：编码标准化警告，运行时发生的错误。

E_RECOVERABLE_ERROR：接近致命的运行时错误，若未被捕获则视同E_ERROR。

E_ALL：捕获所有的错误和警告。

2-3、屏蔽PHP错误

方法一：在有可能出错的函数前加@,然后or die("") 如：@mysql_connect(...) or die("Database Connect Error")

方法二：编辑php.ini，查找"display_errors ="，将“=”后面的值改为"off"

方法三：在php脚本前加error_reporting(0)，屏蔽所有错误提示。
其中，error_reporting 配置错误信息回报的等级。
error_reporting(错误提示级别)
错误提示常量
0：隐藏所有错误

E_ALL: 显示所有错误

E_ERROR: 显示致命错误

E_WARNING: 显示警告错误

E_NOTICE: 显示潜在错误

E_ERROR | E_WARNING : 显示致命错误和警告错误

E_ALL ^ E_NOTICE: 显示所有错误除了潜在错误

2-4、错误处理 函数

1、debug_backtrace — 产生一条回溯跟踪(backtrace)

2、debug_print_backtrace — 打印一条回溯。

3、error_clear_last — 清除最近一次错误

4、error_get_last — 获取最后发生的错误

5、error_log — 发送错误信息到某个地方

6、error_reporting — 设置应该报告何种 PHP 错误

7、restore_error_handler — 还原之前的错误处理函数

8、restore_exception_handler — 恢复之前定义过的异常处理函数。

9、set_error_handler — 设置用户自定义的错误处理函数

10、set_exception_handler — 设置用户自定义的异常处理函数

11、trigger_error — 产生一个用户级别的 error/warning/notice 信息

12、user_error — trigger_error 的别名