第 6 章 目录与文件

学习要点：

1.目录操作

2.磁盘、目录和文件计算

3.文件处理

主讲教师：李炎恢
合作网站：http://www.ibeifeng.com
讲师博客：http://hi.baidu.com/李炎恢

将相关的数据组织为文件和目录等实体，这一直是计算环境的核心概念。出于这个原因，程序员需要有一种方法来获得关于文件和目录的重要细节，如位置、大小、最后修改时间、最后访问时间和其他确定信息。

一. 目录操作

解析目录路径：basename()函数返回路径的文件名部分。

```
<?
    $path = 'C:\AppServ\www\Basic6\Demo1.php';
    echo 'path:'.basename($path);
?>
```

获取路径的目录：dirname()函数返回路径目录部分。

```
<?
    $path = 'C:\AppServ\www\Basic6\Demo1.php';
    echo 'path:'.dirname($path);
?>
```

关于路径的信息：pathinfo()函数创建一个关联数组，其中包括：目录名、基本名和扩展名。

```
<?
    $path = 'C:\AppServ\www\Basic6\Demo1.php';
    $array_path = pathinfo($path);
    print_r($array_path);
?>
```

确定绝对路径：realpath()函数将 path 中的所有符号链接和相对路径引用转换为相应的硬链接和绝对路径。

```
<?
    $path = '../index.php';
    echo realpath($path);
```

```
?>
```

二．磁盘、目录和文件计算

确定文件的大小：filesize()函数返回指定文件字节大小。

```
<?
    $file = 'C:\AppServ\www\Basic6\Demo1.php';
    echo round(filesize($file)/1024,2).'KB';
?>
```

计算磁盘的可用空间：disk_free_space()函数返回指定的目录所在磁盘分区的可用空间。

```
<?
    $drive = 'C:';
    echo round(disk_free_space($drive)/1024/1024,2).'MB';
?>
```

计算磁盘的总容量：disk_total_space()函数返回指定的目录所在磁盘分区的总容量。

```
<?
    $drive = 'C:';
    echo round(disk_total_space($drive)/1024/1024,2).'MB';
?>
```

确定文件的最后访问时间：fileatime()函数返回文件的最后访问时间，采用的 Unix 时间戳格式。

```
<?
    $file = 'C:\AppServ\www\Basic6\Demo1.php';
    echo date("Y-m-d,h:i:s",fileatime($file));
?>
```

确定文件的最后改变时间：filectime()函数返回文件的最后改变时间，采用 Unix 时间戳格式。

```
<?
    $file = 'C:\AppServ\www\Basic6\Demo1.php';
    echo date("Y-m-d,h:i:s",filectime($file));
?>
```

确定文件的最后修改时间：filemtime()函数返回文件的最后修改时间，采用 Unix 时间戳格式。

```
<?
    $file = 'C:\AppServ\www\Basic5\Demo1.php';
    echo date("Y-m-d,h:i:s",filemtime($file));
?>
```

三．文件处理

资源（resource）这个词常常与可以发起输入或输出流的实体联系起来。标准输入或输出、文件和网络套接字都是资源的例子。因此你会经常看到本节所介绍的很多函数都是在资源处理的上下文中讨论的，而不是文件处理，这本身是因为所有这些函数都能够与前面所述的资源结合使用。但是，由于这些函数与文件结合使用是应用中最常见的。

将数据写入一个文件，有 3 个步骤：

1. 打开这个文件。如果文件不存在，需要先创建它。
2. 将数据写入这个文件。
3. 关闭这个文件

同样，从一个文件中读出数据，也有 3 个步骤：

1. 打开这个文件。如果这个文件不能打开，就应该意识到这一点并且正确地退出。
2. 从文件中读出数据。
3. 关闭这个文件。

打开文件

要在 PHP 中打开一个文件，可以使用 fopen()函数。当打开一个文件的时候，还需要指定如何使用它。也就是，文件模式。

选择文件模式：当打开一个文件的时候，有 3 中选择：

1. 打开文件为了只读、只写或者读和写。

2. 如果要写一个文件，你可能希望覆盖所有已有的文件内容，或者仅仅将新数据追加到文件末尾。如果该文件已经存在，也可以终止程序的执行而不是覆盖该文件。

3. 如果希望在一个区分了二进制方式和纯文本方式的系统上写一个文件，还必须制定采用的方式。

函数 fopen()支持以上 3 中方式的组合。

$fp=fopen(文件路径 ， 文件模式 , [是否在 incloude_path 中搜索一个文件],[允许文件名以协议名称开始(如 http://)])

fopen()函数的文件模式总结

模式	模式名称	意义
r	只读	文件指针置于文件开头
r+	读写	文件指针置于文件开头
w	只写	在写入前，删除文件内容，将指针返回到文件开头。如果文件不存在，则尝试创建。
w+	读写	在读取或写入之前，删除文件内容，将指针返回到文件开头。如果文件不存在，则尝试创建。

a	只写	文件指针置于文件末尾。如果文件不存在，则尝试创建。此模式成为追加 （append）。
a+	读写	文件指针置于文件末尾。如果文件不存在，则尝试创建。此过程称为追加到文件。
b	二进制	二进制模式----用于与其他模式进行连接。如果文件系统能够区分二进制文件和文本文件，你可能会使用它。Windows 系统可以区分；而 UNIX 则不区分。推荐一直使用这个选项，以便获得最大程度的可移植性。二进制模式是默认的模式。
t	文本	用于与其他模式的结合。这个模式只是 Windows 系统下一个选项。它不是推荐选项，除非你曾经在代码中使用 b 选项。

如果 fopen()函数调用失败，函数将返回 false。可以用一种对于用户友好的方式来处理这个错误，可以通过抑制 PHP 的错误信息并且根据自己的方式给出错误信息。

```
@$fp=fopen("file.txt",'ab');     //ab 追加并且二进制方式
```

写文件

在 PHP 中写文件相对比较简单。可以使用 fwrite() 或者 fputs() 函数。我们可以使用如下方式调用 fwrite();

```
fwrite($fp, $outputstring, [int length]);   //第三个可选参数为最大字符数
```

这个函数告诉 PHP 将保存在$outputstring 中的字符串写入到$fp 指向的文件中。

fwrite()函数的一个新的替换函数是 file_put_contects().

可以通过 PHP 的内置 strlen()函数获得字符串的长度：

```
fwrite($fp, $outputstring, strlen($outputstring));
```

当然还有一种不需要资源句柄的写入方法：file_put_contents()。

关闭文件

当文件使用完毕后，应该将其关闭。应该调用 fclose()函数：

```
fclose($fp);
```

读出文件

```
$fp = fopen("file.txt","r");
```

fgetc()：读出一个字符，并将指针移到下一个字符。

fgets()：读出一行字符，可以指定一行显示的长度。

fgetss()：从文件指针中读取一行并过滤掉 HTML 标记。

fread()：读取定量的字符。

fpassthru()： 输出文件指针处的所有剩余数据。

file()：将整个文件读入数组中，以行分组。

readfile()：读入一个文件并写入到输出缓冲。

file_get_contents()：将整个文件读入一个字符串。

```
while (!feof($fp)) {
    echo fgetc($fp);
}
```

判断读完文件函数：feof()　，返回 true ,!feof() 返回 false;

查看文件是否存在：file_exists();

```
if (file_exists("file.txt")) {
    //..
}
else {
    //...
}
```

查看一个文件的大小：filesize();

删除一个文件：unlink();

在文件中定位：rewind()、fseek()和 ftell()；

rewind()函数可以将文件指针复位到文件的开始。

ftell()函数可以以字节为单位报告文件指针当前在文件中的位置。

fseek()函数可以将文件指针 fp 从 whence 位置移动 offset 字节。

文件锁定

假设两个客户试图同时订购同一件商品。那么他们同时打开了这个文件，同时的更新，就会出现少一个客户订购的数据。

为了避免这样的问题，可以使用文件锁定的方法。

flock()函数。

flock 的操作值

操作值	意义
LOCK_SH(以前为 1)	读写锁定。这意味着文件可以共享，其他人可以读该文件
LOCK_EX(以前为 2)	写操作锁定。这是互斥的。该文件不能被共享
LOCK_UN(以前为 3)	释放已有的锁定
LOCK_NB(以前为 4)	防止在请求加锁时发生阻塞

```
$fp = fopen("file.txt","ab");
flock($fp,LOCK_EX);     //锁定
fwrite($fp, $outsum);
flock($fp, LOCK_UN);     //释放
fclose($fp);
```

目录句柄操作

opendir()：打开路径指定的目录流。

closedir()：关闭目录流。

readdir()：返回目录中的各个元素。

```
$dir = opendir('C:\AppServ\www\Basic5');
while (!!$file = readdir($dir)) {
    echo $file.'<br />';
}
closedir($dir);
```

scandir()：将目录读入数组。

```
print_r(scandir('C:\AppServ\www\Basic5'));
```

rmdir()：删除指定的目录。

```
rmdir('C:\AppServ\www\Basic5\123');
```

rename()：重命名文件。

```
rename('Demo1.php','Demo01.php');
```

感谢收看本次教程！

本课程是由北风网(ibeifeng.com)

瓢城 Web 俱乐部(yc60.com)联合提供：

本次主讲老师：李炎恢

我的博客：hi.baidu.com/李炎恢/

我的邮件：yc60.com@gmail.com