第10章 文件系统

+ 文件是用来存储数据的方式之一，数据还可以存储在数据库中。但是相对数据库存储来说，文件在使用上更加方便和直接。PHP对文件系有很好的支持，提供了非常多的文件系统操作的函数。PHP还能非常好地支持文件上传功能。本章我们计算来学习这些知识。

10.1 文件处理

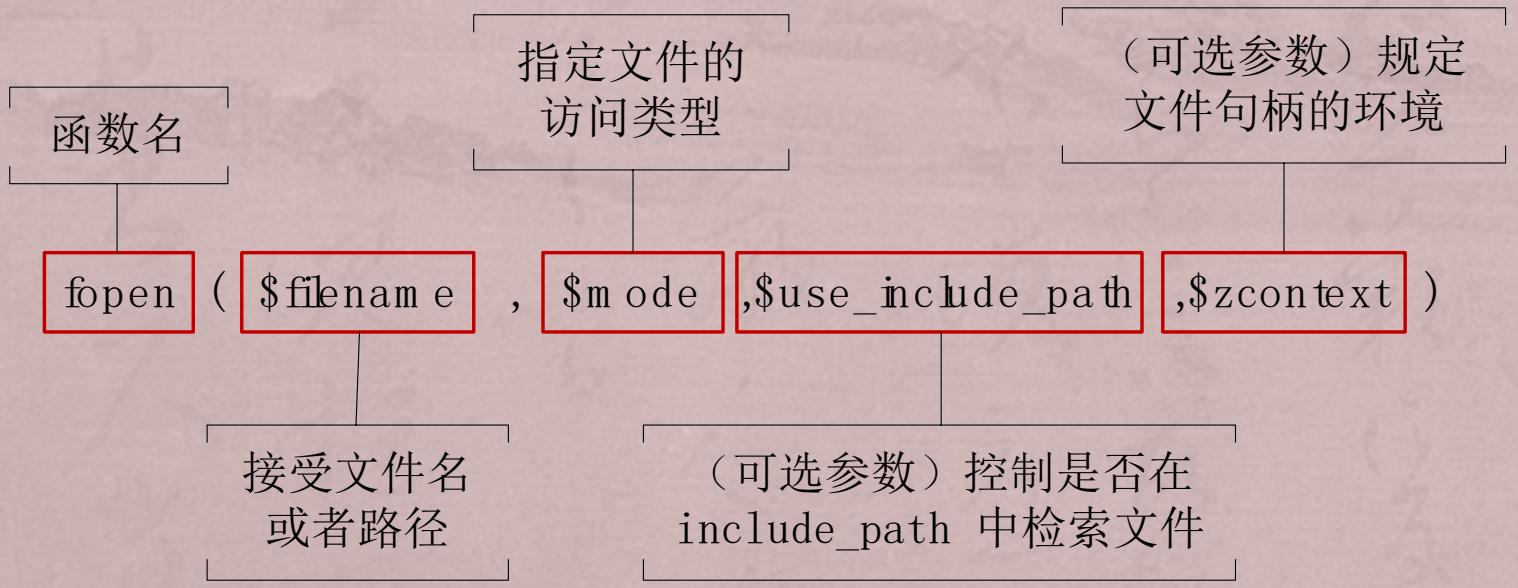
+ PHP提供了很多文件处理函数，我们在程序中通常就是调用这些函数来操作文件，下面我们就来系统地学习它们。

10.1.1 打开和关闭文件

- 在操作文件之前，我们首先要打开文件才可以，这是进行数据操作的第一步。而在操作完成后，又需要将打开的文件关闭以释放资源。

1.打开文件

- PHP中使用fopen()来打开文件或者URL，如果打开失败则返回FALSE。它的语法如图所示。
- 在fopen()的语法中$mode可选的参数如表所示。



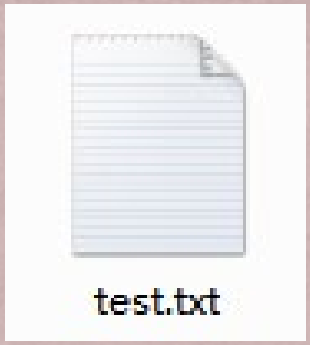
1.打开文件

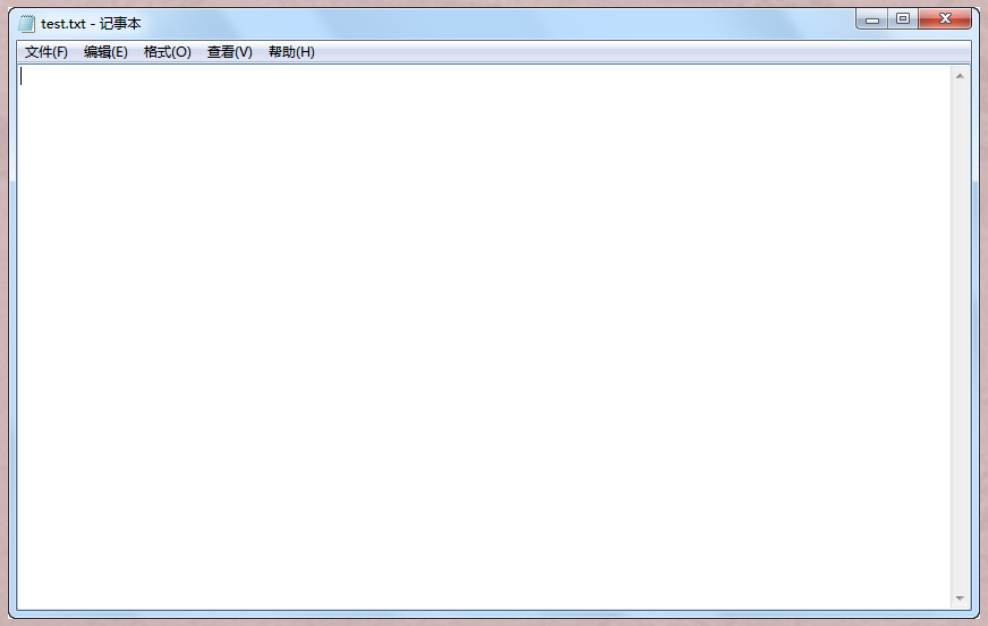
+ 在fopen()的语法中$mode可选的参数如表所示。

"r"	只读方式打开，将文件指针指向文件头
"r+"	读写方式打开，将文件指针指向文件头
"w"	写入方式打开，将文件指针指向文件头并将文件大小截为零。如果文件不存在则尝试创建
"w+"	读写方式打开，将文件指针指向文件头并将文件大小截为零。如果文件不存在则尝试创建
"a"	写入方式打开，将文件指针指向文件末尾。如果文件不存在则尝试创建
"a+"	读写方式打开，将文件指针指向文件末尾。如果文件不存在则尝试创建
"x"	创建并以写入方式打开，将文件指针指向文件头。如果文件已存在，则fopen()调用失败并返回FALSE，并生成一条E_WARNING级别的错误信息。如果文件不存在则尝试创建
"x+"	创建并以读写方式打开，将文件指针指向文件头。如果文件已存在，则fopen()调用失败并返回FALSE，并生成一条E_WARNING级别的错误信息。如果文件不存在则尝试创建

1.打开文件

- 下面我们就使用fopen()来打开一个文件。为了避免对其他文件造成破坏，我们在D盘根目录（D:\）建立一个名为“test.txt”的空文件，如图所示。
- (1)演示使用fopen()打开存在的文件“test.txt”和不存在的文件“test1.txt”并判断是否打开成功。
- 在示例的展示之前，我们需要在服务器主目录（默认XAMPP安装盘的\xampp\htdocs目录，这里为D:\xampp\htdocs）里创建一个文件“testfile.txt”文件，如图10.2所示。
- (2)演示使用相对路径访问与PHP源文件同目录下的“testfile.txt”文件。



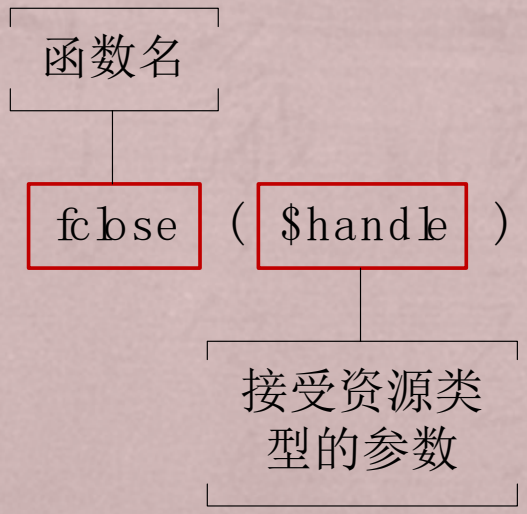


1.打开文件

- 我们再来使用相对路径访问一个jpg类型的文件。我们先在D盘根目录下建立folder文件夹，在folder文件夹下建立image文件夹，在image文件夹下放入一个jpg或者其他格式的图片文件（这里为image.jpg）。创建完成后它的路径和文件名，如图所示。
- (1)演示使用相对路径访问image.jpg文件。
- 在操作文件的程序中，如果一开始打开文件就出现错误，那么我们一般会希望程序不再向下执行。在前面我们学习过die()，我们可以使用它来完成这个操作。
- (2)使用die()控制程序执行。

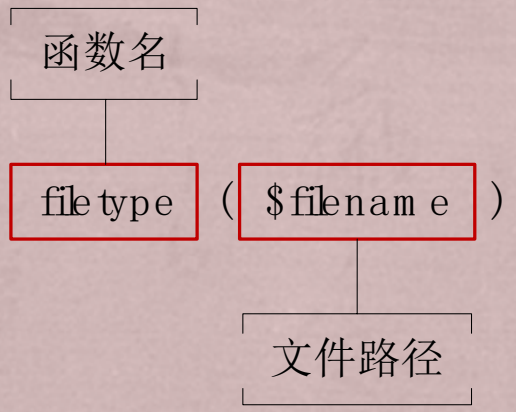
2.关闭文件

- 在PHP中，我们使用fclose()来关闭一个打开的文件，它接受一个文件资源类型的参数。成功关闭后会返回TRUE，失败则返回FALSE，语法如图所示。
- (1)演示使用fclose()关闭打开后的文件。



10.1.2 文件类型

- 在前面的小节中，我们学习了打开和关闭文件。示例中我们打开过“txt”类型的文件，也打开过“jpg”类型的文件。在PHP中也提供了判断文件类型的函数filetype()，它可以判断出传入的参数是什么类型，它的语法如图所示。



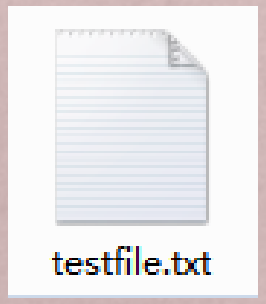
10.1.2 文件类型

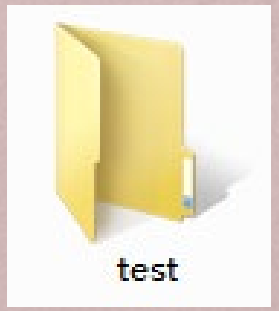
- filetype()会返回一个表示文件类型的字符串，如果出错则会返回FALSE。返回的字符串及其描述如表所示。
- 由于PHP是以UNIX文件系统为模型的，因此在Windows系统中我们只能获得“file”、“dir”和“unknown”三种文件类型。

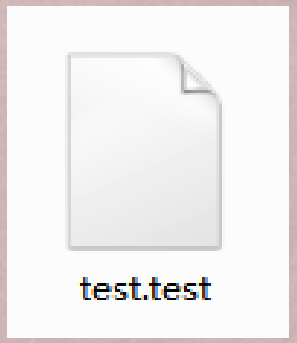
返回值	描述
char	字符串设备，指在I/O传输过程中以字符为单位传输的设备，如键盘，打印机等
block	块设备文件，如某个磁盘分区
dir	目录类型
fifo	命名管道，常用于把信息从一个进程传递到另一个进程
file	普通文件类型
link	符号链接，指向文件指针的指针，类似Windows中的快捷方式
unknown	未知文件类型

10.1.2 文件类型

+ 在下面的示例中我们在“D:\xampp\htdocs”目录下新建两个文件和一个文件夹以供我们测试函数。这些文件如图所示。
+ (1)演示使用filetype()获取图中文件类型。







10.1.3 文件属性

- 在进行编程的时候，需要用到一些文件的属性，如文件大小、问价修改时间、文件的访问权限等信息。PHP为我们提供了表所示的函数来取得这些信息。
- 表中的函数均接受一个文件名或者路径，因此我们不再详讲语法。在示例演示之前我们先在服务器根目录放入一个可执行文件。

函数名	描述	返回值
file_exists()	判断文件是否存在	存在返回TRUE，不存在返回FALSE
filesize()	取得文件大小	返回文件大小的字节数，出错返回FALSE
filectime()	取得文件创建时间	返回UNIX时间戳
filemtime()	取得文件修改时间	返回UNIX时间戳
fileatime()	取得文件访问时间	返回UNIX时间戳
is_readable()	判断文件是否可读	文件存在且可读返回TRUE
is_writable()	判断文件是否可写	文件存在且可写返回TRUE
is_executable()	判断文件是否可执行	文件存在且可执行返回TRUE

10.1.3 文件属性

+ (1)演示使用文件属性处理函数取得文件vim.exe的相关属性。

10.1.3 文件属性

- 除了使用这些函数分别获得文的各种属性之外，我们还可以使用stat()来获取文件的大部分属性。stat()接受一个文件名或者文件路径的参数。该函数在文件存在的情况下返回一个数组，文件错误的情况下输出FALSE。返回的数组信息如表所示。
- (1)演示使用stat()取得文件的属性并输出返回数组的详细信息。

数字下标	关联键名	说明
0	dev	设备名
1	ino	号码
2	mode	保护模式
3	nlink	被连接数目
4	uid	所有者的用户id
5	gid	所有者的组id
6	rdev	设备类型，windows下会返回-1
7	size	文件大小的字节数
8	atime	上次访问时间（Unix 时间戳）
9	mtime	上次修改时间（Unix 时间戳）
10	ctime	上次改变时间（Unix 时间戳）
11	blksize	文件系统I/O的块大小
12	blocks	所占据块的数目

10.1.4 读写文件

+ 前面我们以及学习了打开和关闭文件以及获取文件的各种属性。但是并没有实质性地操作过文件。本节开始，我们就来学习文件的读写操作。文件被打开后我们就可以读取或者修改其内容了。

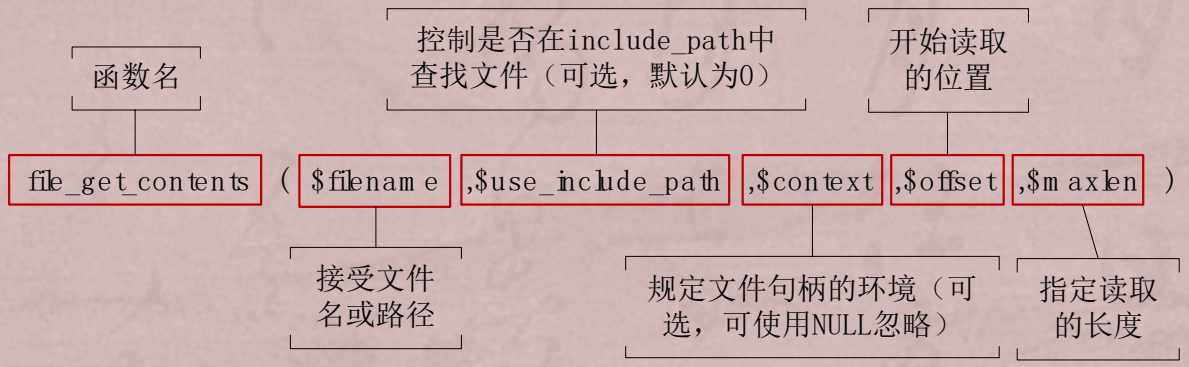
1.从文件中读取数据

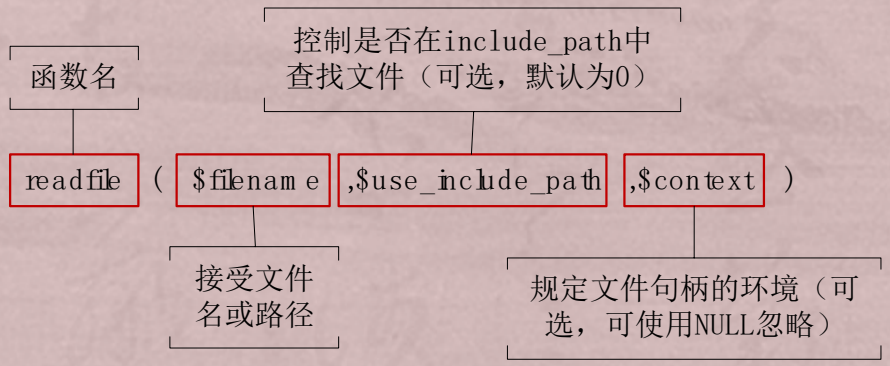
+ PHP提供了很多种读取文件中数组的函数。它们不仅可以读取整个文件的数据，还可以读取一个字符、一行字符串以及读取任意长度的字符。这些常用的函数如表所示。

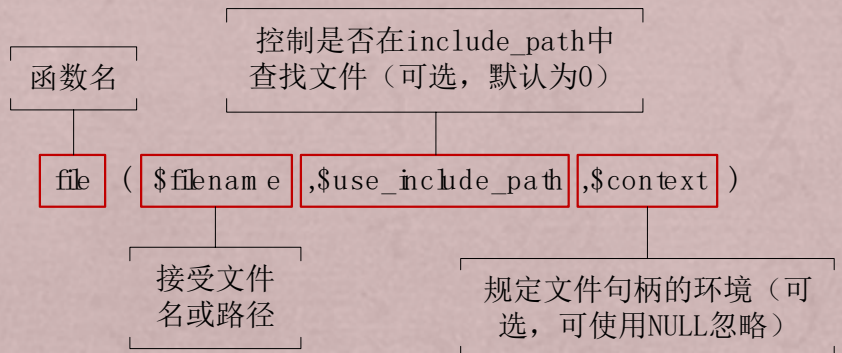
函数名	描述
readfile()	读入一个文件并写入到输出缓冲，出错则返回FALSE
file()	将整个文件读入一个数组中，出错则返回FALSE
file_get_contents()	将整个文件读入一个字符串，出错则返回FALSE
fgets()	从文件指针中读取一行，出错则返回FALSE
fgetss()	从文件指针中读取一行并过滤掉HTML和PHP标记，出错则返回FALSE
fgetc()	从文件指针中读取一个字符，出错则返回FALSE
fread()	从文件读取指定字节数的数据，出错则返回FALSE

1.从文件中读取数据

+ 表中函数的语法如图所示。

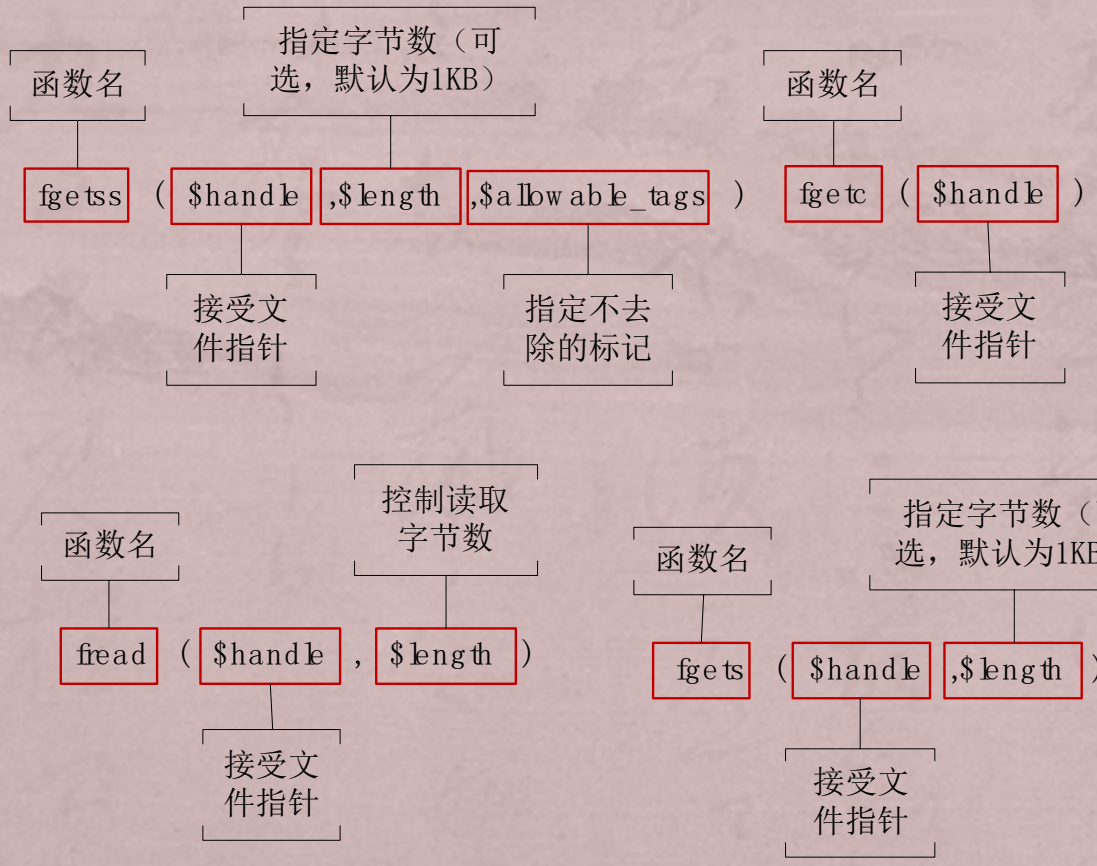






1.从文件中读取数据

+ 表中函数的语法如图所示。

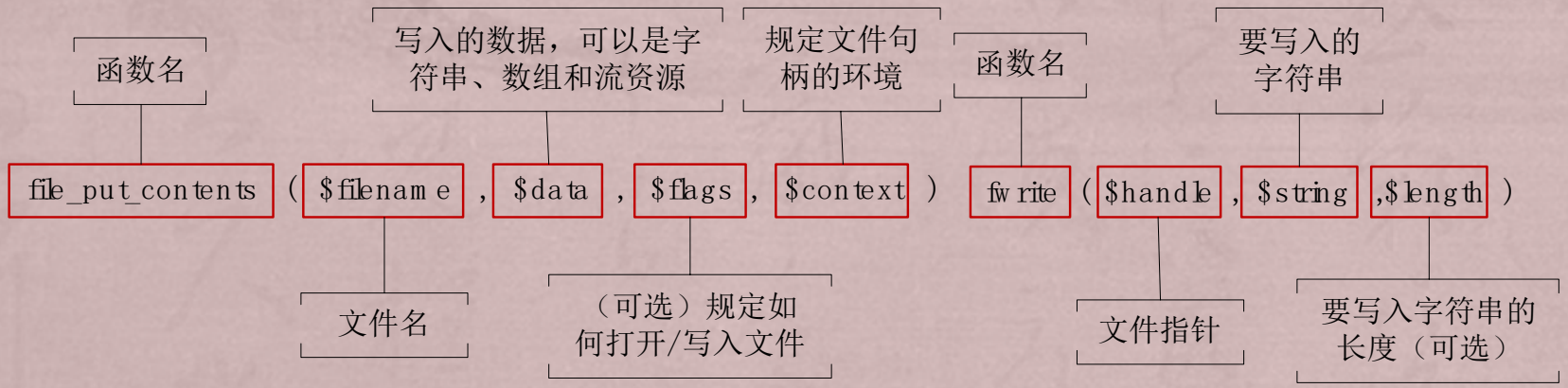


1.从文件中读取数据

- 我们在进行这些函数的使用之前，要先在服务器根目录建立一个文件（这里以在testfile.txt为例），并且在文件中写入一些数据。
- (1)演示使用readfile()读取文件testfile.txt文件中的数据并输出。
- (2)演示使用file()和file_get_contents()读取文件testfile.txt文件中的数据并输出。
- (3)演示使用fgets()读取文件testfile.txt文件中的数据并输出。
- (4)演示使用fgets()返回指定长度的数据并输出。
- 和fgets()功能和使用类似的还有fgetss()。它和fgets()的最主要区别就是它会过滤掉任何HTML和PHP标签。在演示示例之前我们需要先在服务器更目录建立一个有HTML和PHP代码的文件（这里以testhtml.php为例）。
- (5)使用fgetss()读取文件的数据并过滤HTML和PHP标签。
- (6)演示在fgetss()函数中指定不过滤的标签后输出文件数据。
- (7)演示使用fgetc()获取文件中的字符。
- (8)演示使用fread()读取指定长度的字符。

2.向文件中写入数据

- 我们使用fwrite()和file_put_contents()向文件中写入数据。fwrite()也可以被称为fputs()，它们的用法与作用是相同的。我们来看它们的语法，如图所示。
- fwrite()和file_put_contents()在写入成功后都会返回写入的字节数，失败则返回FALSE。



2.向文件中写入数据

- (1)演示使用fwrite()给文件testfile.txt中写入数据。
- (2)演示为不存在的文件写入数据程序会报错。
- 需要注意的是，在示例中我们使用了错误控制运算符“@”隐藏了提示信息，使程序仅通过条件来判断。如过使用不同的模式（这里为“r+”），例如“a”或者“w+”则会在文件不存在的时候创建文件，因此数据会写入成功。
- (3)演示使用“w+”模式打开不存在的文件并写入数据。
- 我们要明白，在不同的模式下写入数据会有不同的结果，特别是在会覆盖内容的模式要慎用，以免造成不必要的损失。file_put_contents()与fwrite()最大的不同是前者可以使用数组（不可为多维数组）作为写入的数据。
- (4)演示使用file_put_contents()向文件写入数据。
- (5)演示为file_put_contents()的$flags指定参数与不指定参数产生的结果。

10.1.5 操作文件

- 在PHP中除了可以对文件内容进行读写操作，对文件本身也同样可以进行操作。例如复制文件、重命名文件以及删除文件等操作。表列出了常用的文件操作函数。

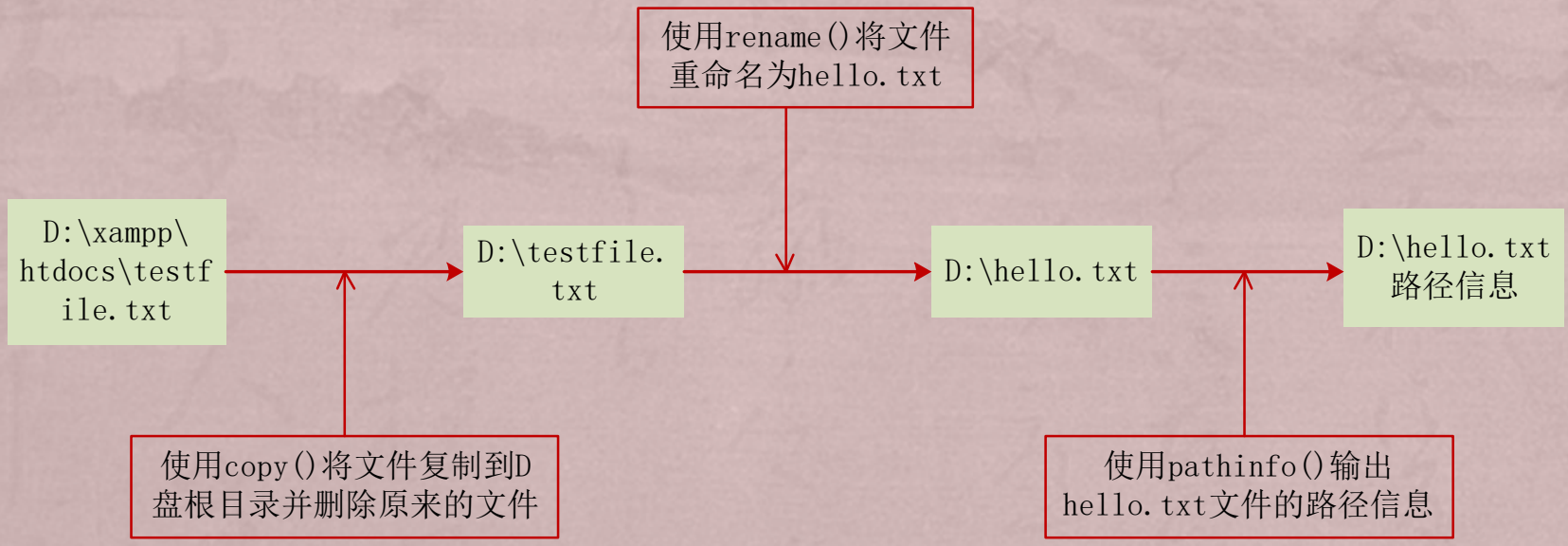
函数名	描述
copy()	将文件复制到指定路径,成功返回TRUE，失败返回FALSE
rename()	重命名一个文件
unlink()	删除一个文件
pathinfo()	返回文件的路径信息
realpath()	返回文件的绝对路径

10.1.5 操作文件

+ 如图所示为表中函数的语法。

10.1.5 操作文件

+ 下面我们以一个综合流程操作来完成学习这些函数。流程要求如图所示。
+ (1)实现图中的操作流程。



10.2 目录处理

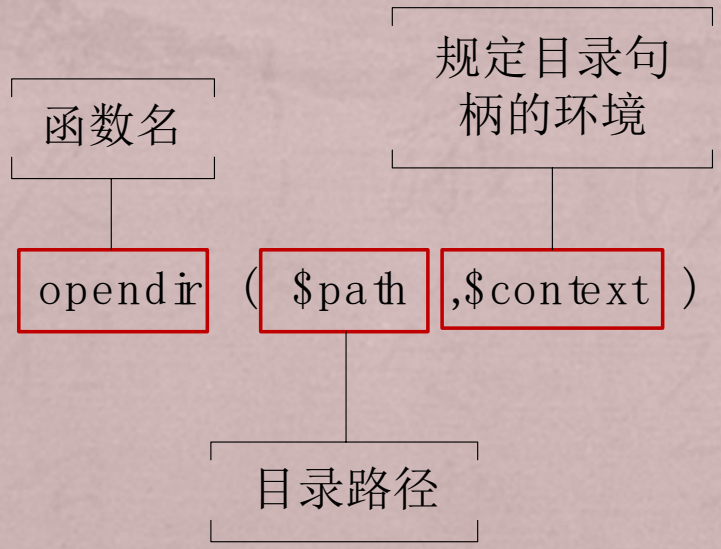
+ 目录是一种特殊的文件类型，通过对目录的操作，我们可以浏览其中的文件。也可以对其中的文件进行各类操作。

10.2.1 打开和关闭目录

- 对目录的操作同对普通文件的操作类似，在浏览之前要先打开目录，浏览完毕后同样需要关闭目录。

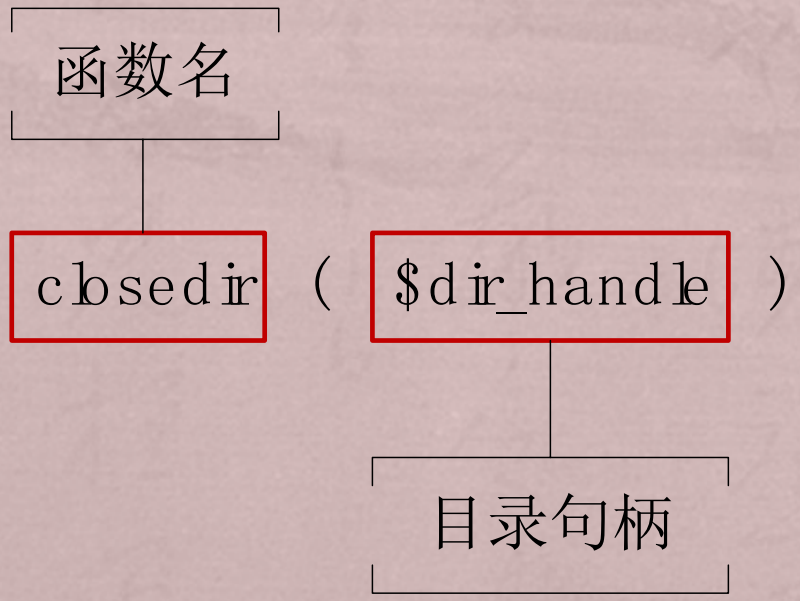
1.打开目录

- 在PHP中我们使用opendir()来打开一个目录，它的语法如图所示。
- opendir()执行成功后会返回一个资源类型的目录句柄，执行失败则返回FALSE。



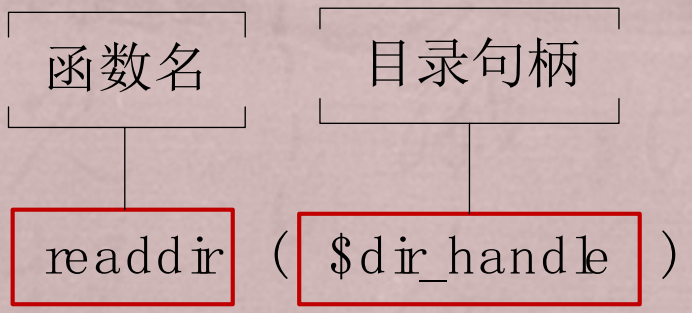
2.关闭目录

+ 在PHP中我们使用closedir()来关闭一个打开的目录，它返回一个空值，语法如图所示。
+ (1)演示打开和关闭一个目录。



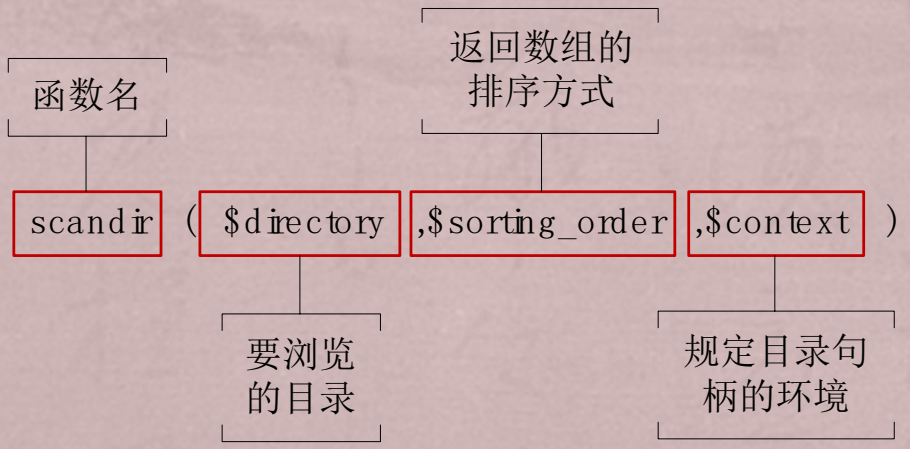
10.2.2 浏览目录

- 正确打开目录后我们就可以浏览目录内容了。在PHP中我们使用readdir()浏览目录内容，语法如图所示。
- readdir()在成功后返回目录中下一个文件的文件名，失败则返回FALSE。



10.2.2 浏览目录

- (1)演示使用readdir()循环输出目录文件名。
- 除了可以使用readdir()之外，还可以使用scandir()来浏览一个目录。scandir()与readdir()的不同之处在于不需要我们显式地打开和关闭目录，只需将目录作为scandir()的参数即可，它的语法如图所示。
- scandir()执行成功则返回包含浏览目录中的文件和目录的数组，执行失败则返回FALSE。



10.2.2 浏览目录

+ (1)演示使用scandir()浏览目录中的文件和目录。

10.2.3 操作目录

+ 目录是一种特殊的文件，因此文件的操作对目录操作同样有效。但操作目录还有一些专门的函数，如表所示。

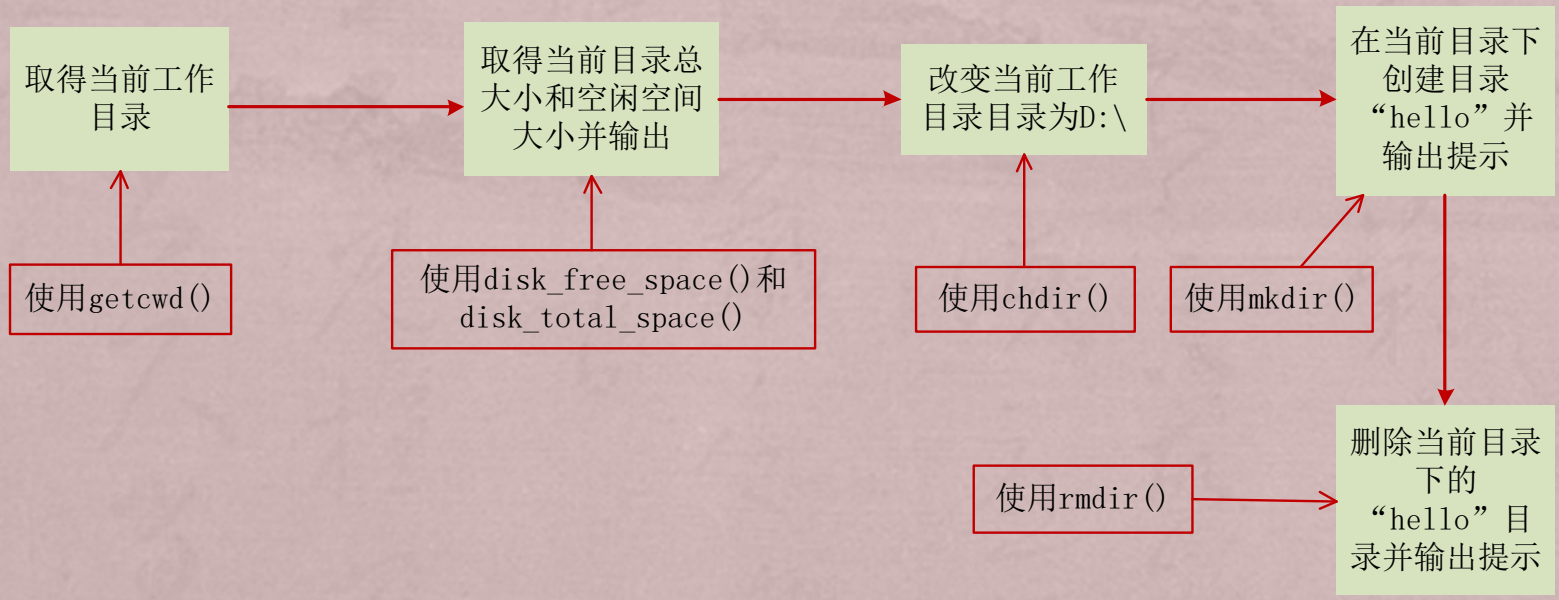
函数名	描述
mkdir()	新建一个目录
rmdir()	删除指定的目录，要求目录必须为空目录
getcwd()	取得当前工作目录
chdir()	改变目录
disk_free_space()	返回目录的可用空间
disk_total_space()	返回目录的总空间大小
rewinddir()	将目录句柄复位

10.2.3 操作目录

+ 表中函数的语法如图所示。

10.2.3 操作目录

- 我们同样以通过一个综合的示例来学习这些函数，我们规定要实现的操作流程如图所示。
- (1)实现图中所示的操作流程。
- 由于在程序中创建目录后接着就将目录删除了，因此在电脑硬盘里不会看到新创建的文件夹，读者可以自行将删除的语句注释后即可看到新建的文件夹。



10.3 文件处理的高级应用

+ 在PHP中，除了可以对文件进行基本的读写操作以外，还可以对文件指针进行查找、定位以及对正在读取的文件进行锁定等操作。下面我们就来学习这些文件处理的高级应用。

10.3.1 访问远程文件

- PHP支持URL格式的文件调用。实现这个功能只需要在php.ini中将如图所示的选项设置为ON后重启服务器即可。
- 当然由于我们使用的是集成环境，这个选项是默认开启的。当然读者最好去配置文件中确认一次。

```
894    allow_url_fopen = On
```

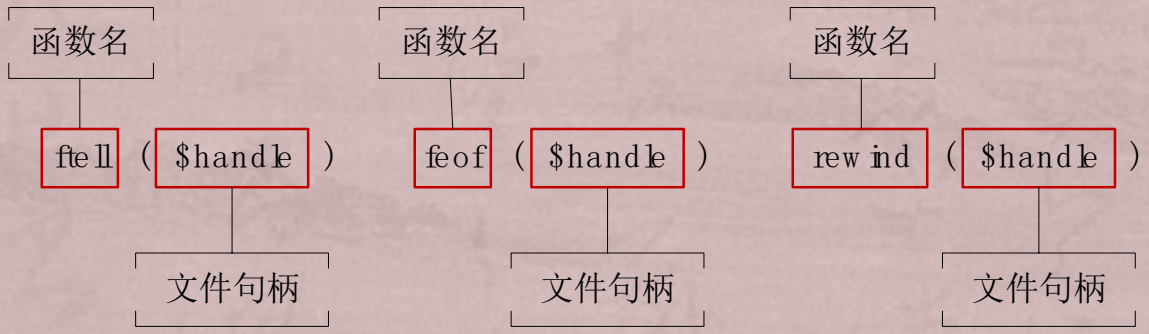
10.3.2 文件指针

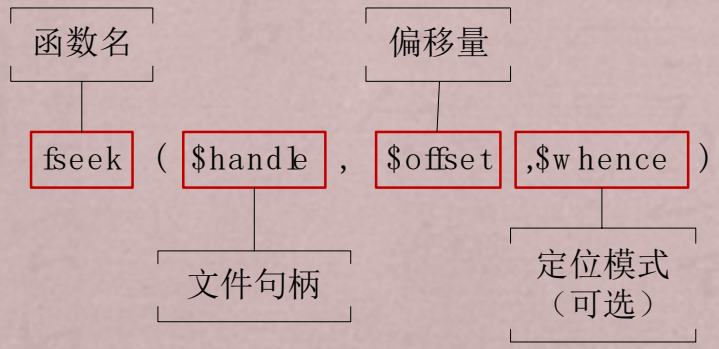
+ PHP可以实现文件指针的定位和查询，从而实现所需信息的快速查询，常用的文件指针相关的函数如表所示。

函数名	描述
ftell()	返回当前指针的位置
feof()	判断文件指针是否在文件结尾
fseek()	定位文件指针
rewind()	复位文件指针

10.3.2 文件指针

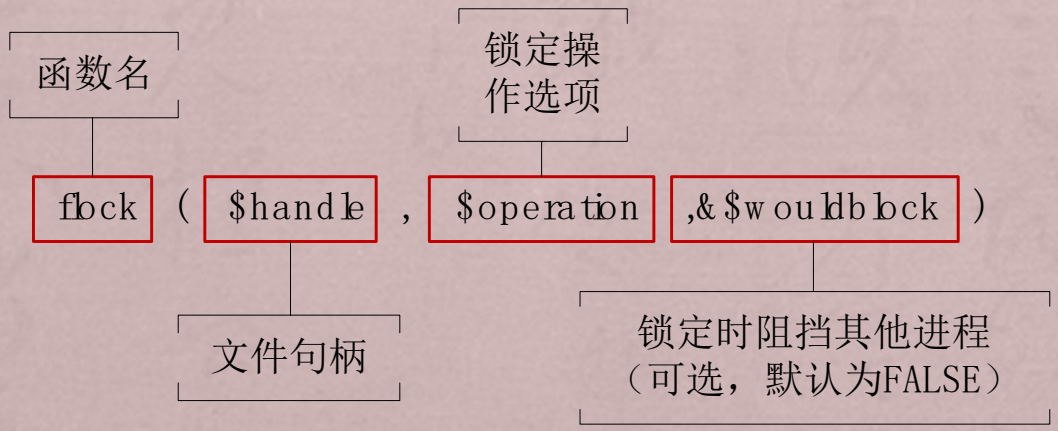
- 表中函数的语法如图所示。
- 在图中的函数，fseek()的$whence可以有以下选项：
- SEEK_SET：设定位置等于$offset字节，此选项为默认值。
- SEEK_CUR：设定位置为当前位置加上$offset。
- SEEK_END：设定位置为文件尾加上$offset。（要移动到文件尾之前的位置，需要给$offset传递一个负值。）
- 在开始学习这些函数之前，我们要先在磁盘中新建一个文件（这里以D:\xampp\htdocs\php.txt为例）。
- (1)演示文件指针操作函数的用法以及作用。





10.3.3 文件锁定

- 在向一个文件写入内容的同时，如过其他用户也修改这个文件，就有可能造成写入数据出错而发生信息的丢失。为了防止这种情况的发生，PHP提供了flock()来锁定文件，以避免其他用户同时修改，它的语法如图所示。
- flock()会在执行成功后返回TRUE，失败则返回FALSE。语法中的$operation()可以有如下选项：
- 要取得共享锁定（读取的程序），将$operation设为LOCK_SH。
- 要取得独占锁定（写入的程序），将$operation设为LOCK_EX。
- 要释放锁定（无论共享或独占），将$operation设为LOCK_UN。
- 如果不希望flock()在锁定时堵塞，将$operation设为LOCK_NB。
- (1)演示使用flock()锁定文件。



10.4 文件上传

+ 文件上传也是PHP文件系统的重要功能之一，在本小节的学习中，会涉及到非常一小部分的HTML知识。当然读者有HTML基础最好，没有也完全不要害怕，重点的知识在PHP而不是HTML。因此读者完全不用担心。

10.4.1 配置PHP.INI文件

- 在上传文件之前首先要配置php.ini中的如下选项，如图所示。
- 由于我们使用的是集成环境，这些选项都是已经配置好的，当然读者再确认一次最好。这些选项的含义如下：
- memory_limit：PHP中给一个指令分配的内存空间，以MB为单位。
- max_execution_time：PHP中执行一条指令可以使用的最长时间。
- file_uploads：是否支持文件上传。
- upload_tmp_dir：上传文件的临时目录。
- upload_max_filesize：允许上传文件的最大值，以MB为单位。
- 在配置好这些选项后就为文件上传做好了基础准备。

```
460  memory_limit = 128M
442  max_execution_time = 30
877  file_uploads = On
878
879  ; Temporary directory for HTTP u
880  ; specified).
881  ; http://php.net/upload-tmp-dir
882  upload_tmp_dir = "D:\xampp\tmp"
883
884  ; Maximum allowed size for uploa
885  ; http://php.net/upload-max-file
886  upload_max_filesize = 128M
```

10.4.2 认识预定义变量$_FILES

- $_FILES变量储存着上传文件的相关信息，对于上传文件是很重要的。它是一个二维数组，它的元素名及含义如表所示。
- (1)演示$_FILES结合HTML表单取得上传文件信息。

数组元素	保存的信息
$_FILES[filename][name]	保存上传文件的文件名
$_FILES[filename][size]	保存上传文件的大小
$_FILES[filename][tmp_name]	保存上传文件的临时名称
$_FILES[filename][type]	保存上传文件的类型
$_FILES[filename][error]	保存上传文件结果的代号，0则表示成功

10.4.3 单文件上传

- 通过上一小节中的代码我们以及可以取得上传文件的信息了，那么通过这些信息再联合move_uploaded_file()函数就可以实现文件上传。move_uploaded_file()的语法如图所示。
- 这里需要注意的是，$filename接受的是文件上传时候的临时名称而不是文件原名称，它可以通过$_FILES得到。这里我们以将名字为的文件上传到D盘根目录（D:\）为例。
- (1)演示将“image.jpg”文件上传到D盘根目录。

10.4.4 多文件上传

- 前面的小节中我们成功上传了一个文件，在有些时候是有一次上传多个文件的需求。例如上传相片到博客就需要多个文件同时上传。实现多文件上传也非常简单，
- (1)演示将多个文件上传到D:\image文件夹下。
- 从提交上传后的结果我们可以看出文件均上传成功了，在D:\image路径下也有了成功上传的文件。
- (2)演示提交多个上传文件后$_FILES的数组信息。

10.5 小结

+ 本章我们先通过学习简单的文件操作，然后又学习的目录的基本操作，再到后来的文件处理高级应用部分以及文件上传。循序渐进地学习了PHP的文件系统的操作。这个在实际应用中是应用非常广泛的。因为PHP就是偏向于处理文件和数据的，因此希望读者能牢固掌握本章的知识。