# 12.1.1文件系统概述

### 文件类型

在程序运行时，程序本身和数据一般都存在内存中，当程序运行结束后，存放在内存中的数据被释放。

如果需要长期保存程序运行所需的原始数据，或程序运行产生的结果，就必须以文件形式存储到外部存储介质上。

文件一般指存储在外部介质上具有名字（文件名）的一组相关数据集合。用文件可长期保存数据，并实现数据共享。

PHP是以UNIX的文件系统为模型的。因此在Windows系统中我们只能获得”file”、”dir”或者“unknown”三种文件类型。而在UNIX系统中，我们可以获得block、char、dir、fifo、file、link和unknown七种类型。

可以使用函数filetype（）获取文件的具体类型。

语法：string filetype ( string filename )

### UNIX系统中7种文件类型说明

|  |  |
| --- | --- |
| **文件类型** | **描述** |
| **block** | **块设备文件，如某个磁盘分区，软驱，光驱CD-ROM等** |
| **char** | **字符设备是指在I/O传输过程中以字符为单位进行传输的设备，如键盘、打印机等** |
| **dir** | **目录类型，目录也是文件的一种** |
| **fifo** | **命名管道，常用于将信息从一个进程传递到另一个进程** |
| **file** | **普通文件类型，如文本文件或可执行文件等。** |
| **link** | **符号链接，是指向文件指针的指针。类似Windows中的快捷方式** |
| **unknown** | **未知类型。** |

is\_dir( ) -- 判断给定文件名是否是一个目录

语法结构：bool is\_dir（名称）

返回类型：文件名存在并且是一个目录则返回 true，否则返回 false。

is\_executable( ) -- 判断给定文件名是否可执行

语法结构：bool is\_executable（名称）

返回类型：如果文件存在且可执行则返回 true ，否则返回 false 。

is\_file( ) -- 判断给定文件名是否为一个正常的文件

语法结构：bool is\_file(名称)

返回类型：如果文件存在且为正常的文件则返回 true 。

is\_link( ) -- 判断给定文件名是否为一个符号连接

语法结构：bool is\_link(名称)

返回类型：如果文件存在并且是一个符号连接则返回 true。

is\_readable( ) -- 判断给定文件名是否可读

语法结构：bool is\_readable（文件名称）

返回类型：如果文件存在并且可读则返回 true 。

is\_writable( ) -- 判断给定的文件名是否可写

语法结构：bool is\_writable(文件名称)

返回类型：如果文件存在并且可写则返回 true 。

# 12.1.2 文件的属性

|  |  |  |  |
| --- | --- | --- | --- |
| **函数名** | **作用** | **参数** | **返回值** |
| **file\_exists( )** | **检查文件或目录是否存在** | **文件名** | **存在：true，不存在：false** |
| **filesize()** | **取得文件大小** | **文件名** | **返回大小字节数，出错：false** |
| **is\_readable()** | **判断文件是否可读** | **文件名** | **文件可读返回true** |
| **is\_writable()** | **判断文件是否可写** | **文件名** | **文件可写返回true** |
| **is\_executable()** | **判断文件是否可执行** | **文件名** | **文件可执行返回true** |
| **filectime()** | **获取文件的创建时间** | **文件名** | **返回UNIX时间戳格式** |
| **filemtime()** | **获取文件的修改时间** | **文件名** | **返回UNIX时间戳格式** |
| **fileatime()** | **获取文件的访问时间** | **文件名** | **返回UNIX时间戳格式** |
| **stat()** | **获取文件大部分属性** | **文件名** | **返回给定属性信息的数组** |

# 12.2.1 解析目录路径

### 1.目录的基本操作

**解析目录路径**

**遍历目录**

**统计目录大小**

**建立与删除目录**

**复制目录**

### 课程内容

解析目录路径

相对和绝对路径（.和..）

路径分隔符号“/”常量

**DIRECTORY\_SEPARATOR PATH\_SEPARATOR PHP\_EOL**

不同的根路径

路径解析函数

### 2.1 解析目录路径

使用PHP脚本可以方便对目录进行操作，如创建目录、遍历目录、复值目录与删除目录等操作。

常用的文件目录路径格式：

—$unixPath="/var/www/html/index.php";

//在UNIX系统中的绝对路径，必须使用"/"分隔

—$winPath="C:\\Appserv\\www\\index.php";

//在Windows系统的绝对路径，默认使用"\"分隔

—$winPath2="C:/Appserv/www/index.php";

//在Windows系统中也可使用“/”分隔。

注意使用绝对路径与相对路径。

### PHP文件路径相关函数

basename -- 返回路径中的文件名部分

—语法：string basename ( string path [, string suffix] )

—给出一个包含有指向一个文件的全路径的字符串，本函数返回基本的文件名。如果文件名 是以 suffix 结束的，那这一部分也会被去掉。

dirname -- 返回路径中的目录部分

—语法：string dirname ( string path )

—给出一个包含有指向一个文件的全路径的字符串，本函数返回去掉文件名后的目录名。

**例子**

<?php

$path = "/home/httpd/html/index.php";

$file = basename($path);// $file值："index.php"

$file = basename($path, ".php "); // $file值："index "

$file = dirname($path); // $file值："/home/httpd/html"

pathinfo -- 返回文件路径的信息

语法：array pathinfo ( string path [, int options] )

pathinfo() 返回一个联合数组包含有 path 的信息。包括以下的数组单元：dirname，basename 和 extension。

**例子**

<?php

$path\_parts = pathinfo("/www/htdocs/index.html");

echo $path\_parts["dirname"]."\n"; // /www/htdocs

echo $path\_parts["basename"]."\n"; // index.html

echo $path\_parts["extension"]."\n"; // html

?>

realpath -- 返回规范化的绝对路径名

语法：string realpath ( string path )

realpath() 扩展所有的符号连接并且处理输入的 path 中的 '/./', '/../' 以及多余的 '/' 并返回规范化后的绝对路径名。返回的路径中没有符号连接，'/./' 或 '/../' 成分。

# 视频上的note

<?php

/\*

\* C:\Appserv\www\aa.php \ windows下面的路径

\*

\* /usr/local/apache/ / linux下面的路径

\*

\*

\* 1. 注意： "C:/Appserv/www", 所有的程序中， 不管是什么操作系统， 全部都使用 "/"代表路径分隔符号 (PHP程序中， Apache配置文件中， PHP的配置文件中， 只要有目录的情况， 全部使用 /)

\*

\*

\* 2. windows ; Linux : 多个目录之间的分隔符号

\*

\* C:\Apaserv\www; C:\window\aaa

\* /usr/local:/etc/aaa

\*

\*

\* 3. . ..

\*

\* aaa/bbb/ccc/demo.php

\* ./aaa/bbb/ccc/demo.php

\*

\* ../aaa/bbb

\* ../../../aaa/bbb

\*

\* c:/aaa/bbb

\* /user/hello/

\*

\* http://www.lampbrother.net/aaa/bbb/php.php

\*

\* 4. 不同的根路径

\* 一个是文档根目录(在浏览中使用)， 一个是操作系统的根(PHP操作的)

1)php程序找根是从服务器系统找根

2)如果是前台来找根(前台访问阿帕奇)阿帕奇会返回www目录里面的根

比如：一个网页的图片

\*

\*

\*/

echo basename("http://www.lampbrother.net/aaa/bbb/demo.php")."<br>";

echo basename("/aaa/bbb/demo.php")."<br>";

echo basename("../../aaa/bbb/demo.php", ".php")."<br>";

echo dirname("../../aaa/bbb/ccc.php");

print\_r(pathinfo("/aaa/bbb/demo.php"));

# 12.2.2遍历目录

opendir -- 打开目录句柄

语法：resource opendir ( string path [, resource context] )

打开一个目录句柄，可用于之后的 closedir()，readdir() 和 rewinddir() 调用中。

readdir -- 从目录句柄中读取条目

语法：string readdir ( resource dir\_handle )

返回当前目录指针位置的文件名，没有返回false，并将指针向下移动一位。文件名以 在文件系统中的排序返回。

closedir -- 关闭目录句柄

语法：void closedir ( resource dir\_handle )

关闭由 dir\_handle 指定的目录流。流必须之前被 opendir() 所打开。

rewinddir -- 倒回目录句柄

语法：void rewinddir ( resource dir\_handle )

将 dir\_handle 指定的目录流重置到目录的开头。

### 例子：

<?php //遍历指定目录实例

$num=0; //用来统计子目录和文件的个数

$dirname='phpMyAdmin'; //定义一个目录名

$dir\_handle=opendir($dirname); //用opendir打开目录

//将遍历的目录和文件名使用表格格式输出

while($file=readdir($dir\_handle)) { //使用readdir循环读取目录里的内容

$dirFile=$dirname."/".$file; //追加目录名

echo "文件名: ".$file." | "; //显示文件名

echo "大小: ".filesize($dirFile)." | "; //显示文件大小

echo "类型: ".filetype($dirFile)." | "; //显示文件类型

echo "创建时间: ".date("Y/n/t",filectime($dirFile))."<br/>";

num++;

}

closedir($dir\_handle); //关闭文件操作句柄

echo ‘在<b>’.$dirname.‘</b>目录下共有文件<b>'.$num.'</b>个';

# 12.2.3统计目录中个数和大小

**disk\_free\_space -- 返回目录中的可用空间**

语法：float disk\_free\_space ( string directory )

给出一个包含有一个目录的字符串，本函数将根据相应的文件系统或磁盘分区返回可用的字节数。

**disk\_total\_space -- 返回一个目录的磁盘总大小**

语法：float disk\_total\_space ( string directory )

给出一个包含有一个目录的字符串，本函数将根据相应的文件系统或磁盘分区返回所有的字节数。

**例子：**

<?php

//定义一个统计指定目录大小的函数

function dirSize($dir){

$size=0;

//1. 判断参数是否有效

if(!file\_exists($dir) || !is\_dir($dir)){

die(“无效的参数！");

}

//2. 打开目录

$dd = opendir($dir);

//3. 跳过连个特殊目录："."和".."

readdir($dd);readdir($dd);

//4. 开始遍历目录中文件

while($f = readdir($dd)){

//加上目录名

$file = rtrim($dir,"/")."/".$f;

//判断是否是文件

if(is\_file($file)){

$size+=filesize($file);

}

//判断是否是目录

if(is\_dir($file)){

//采用递归调用，统计子目录大小

$size+=dirSize($file);

}

}

//5. 关闭目录

closedir($dd);

//6. 返回结果

return $size;

}

?>

# 12.2.4建立与删除目录

mkdir -- 新建目录

语法：bool mkdir (string pathname [,int mode])

尝试新建一个由 pathname 指定的目录。

rmdir -- 删除目录

语法：bool rmdir ( string dirname )

尝试删除 dirname 所指定的目录。 该目录必须是空的，而且要有相应的权限。如果成功则返回 TRUE，失败则返回 FALSE。

unlink -- 删除文件

语法：bool unlink ( string filename )

删除 filename。和 Unix C 的 unlink() 函数相似。如果成功则返回 TRUE，失败则返回 FALSE。

**例子**

<?php

//自定义删除目录函数

function delDir($dir){

//1. 判断是否是有效文件或目录

if(!file\_exists($dir)){

die("文件或目录{$dir}不存在！");

}

//2. 若是文件就直接删除

if(is\_file($dir)){

unlink($dir);return true;

}

//3. 打开目录

$dd = opendir($dir);

//4. 跳过两个特殊目录:"." ".."

readdir($dd);readdir($dd);

//5. 遍历目录，并删除里面的文件

while($f = readdir($dd)){

//加上全路径

$file = rtrim($dir,"/")."/".$f;

//判断是否是目录还是文件

if(is\_dir($file)){

//采用递归调用删除子目录

delDir($file);

}else{

//删除文件

unlink($file);

}

}

//6. 关闭目录

closedir($dd);

//7. 删除目录

rmdir($dir);

return true;

}

?>

# 12.2.5复制与移动目录

copy -- 拷贝文件

语法：bool copy ( string source, string dest )

将文件从 source 拷贝到 dest。如果成功则返回 TRUE，失败则返回 FALSE。

PHP中没有提供复制与移动目录的相关函数。如需要，只要自定义函数了。

如何定义一个目录复制函数呢？

# 12.3.1文件的一些基本操作函数

copy -- 拷贝文件

语法：bool copy ( string source, string dest )

将文件从 source 拷贝到 dest。如果成功则返回 TRUE，失败则返回 FALSE。

unlink -- 删除文件

语法：bool unlink ( string filename )

删除 filename。和 Unix C 的 unlink() 函数相似。如果成功则返回 TRUE，失败则返回 FALSE。

ftruncate -- 将文件截断到给定的长度

语法：bool ftruncate ( resource handle, int size )

接受文件指针 handle 作为参数，并将文件大小截取为 size。如果成功则返回 TRUE，失败则返回 FALSE。

rename -- 重命名一个文件或目录

语法：bool rename ( string oldname, string newname [, resource context] )

尝试把 oldname 重命名为 newname。 如果成功则返回 TRUE，失败则返回 FALSE。

### 文件内容的基本操作

File\_get\_contents();

file\_get\_contents — 将整个文件读入一个字符串

File\_put\_contents();file\_put\_contents — 将一个字符串写入文件

Readfile();

File()file — 把整个文件读入一个数组中

本地/远程

### 视频note

<?php

//创建一个空文件

//touch("./feng.txt");

//复制文件

//copy("feng.txt", "meize.txt");

//移动或重新命名一个文件

//rename("meize.txt", "meizi.txt");

//删除一个文件

//unlink("meizi.txt");

// $fp = fopen("feng.txt", "w");

// ftruncate($fp, 100);

//对文件内容的操作

// file\_get\_contents("");

//file\_put\_contents("meizi.txt", "妹子今年18！");

//file\_put\_contents("meizi.txt", "小妹子今年19！");

//echo file\_get\_contents("meizi.txt");

//file\_put\_contents("baidu.txt", file\_get\_contents("http://www.baidu.com"));

//readfile("http://www.baidu.com");

$arr = file("meizi.txt");

echo count($arr);

echo '<br>';

echo $arr[20];

# 12.3.2文件的打开与关闭

fopen -- 打开文件或者 URL

语法：resource fopen ( string filename, string mode [, bool use\_include\_path [, resource zcontext]] )

filename参数需要提供要被打开文件的URL。这个URL可以是脚本所在的服务器中的绝对路径，也可以是相对路径，还可以是网络资源用的文件。

mode 参数指定了所要求到该流的访问类型,(强烈建议附加b模式)。

如果也需要在 include\_path中搜寻文件的话，可以将可选的第三个参数 use\_include\_path 设为 '1' 或 TRUE。

如果打开失败，本函数返回 FALSE。

fclose -- 关闭一个已打开的文件指针

语法：bool fclose ( resource $handle )

将 handle 指向的文件关闭。 成功时返回 TRUE， 或者在失败时返回 FALSE.

### 例子

<?php

//使用绝对路径打开file.txt文件，选择只读模式，并返回资源$handle

$handle = fopen("/home/rasmus/file.txt", "r");

//访问文档根目录下的文件，也以只读模式打开

$handle = fopen(“{$\_SERVER['DOCUMENT\_ROOT']}/data/info.txt", "r");

//在 Windows 平台上，转义文件路径中的每个反斜线，或者用斜线，

以二进制和只写模式组合

$handle = fopen("c:\\data\\file.gif", "wb");

//使用相对路径打开file.txt文件，选择只读模式，并返回资源$handle

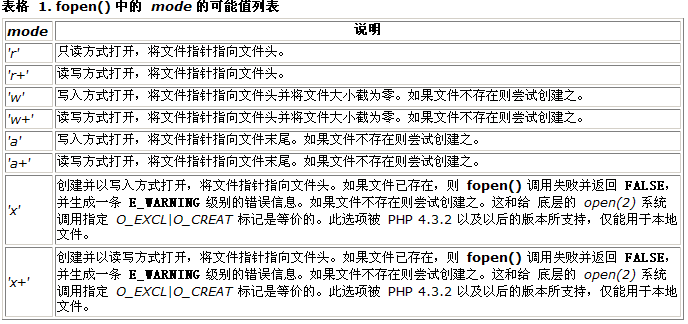
$handle = fopen("../data/info.txt", "r");

//打开远程文件， 使用HTTP协议只能以只读的模式打开

$handle = fopen("http://www.example.com/", "r");

//使用FTP协议打开远程文件，如果FTP服务器可写，则可以以写的模式打开

$handle = fopen("ftp://user:password@example.com/somefile.txt", "w");



# 12.3.3文件的内容操作

### 写入文件

fwrite -- 写入文件（可安全用于二进制文件）

语法：int fwrite ( resource handle, string string [, int length] )

fwrite() 把 string 的内容写入 文件指针 handle 处。 如果指定了 length，当写入了 length 个字节或者写完了 string 以后，写入就会停止，视乎先碰到哪种情况。

返回写入的字符数，出现错误时则返回 FALSE 。

### 读取文件内容

|  |  |
| --- | --- |
| **函数名** | **作用** |
| **fread( )** | **读取文件（可安全用于二进制文件）** |
| **file\_get\_contents()** | **将文件读入字符串** |
| **fgets()** | **从打开的文件中读取一行** |
| **fgetc()** | **从打开的文件中读取一个字符** |
| **file()** | **把文件读入一个数组中（无需使用fopen打开）** |
| **readfile()** | **读取一个文件，并输出到输出缓冲（无需使用fopen打开）** |

### 移动文件指针

ftell -- 返回文件指针读/写的位置

语法：int ftell ( resource handle )

返回由 handle 指定的文件指针的位置，也就是文件流中的偏移量。 如果出错，返回 FALSE。文件指针必须是有效的，且必须指向一个通过 fopen()或 popen()成功打开的文件。

fseek -- 在文件指针中定位

语法：int fseek ( resource handle,int offset [,int whence] )

在与 handle 关联的文件中设定文件指针位置。新位置，从文件头开始以字节数度量，是以 whence 指定的位置加上 offset。whence 的值定义为：

SEEK\_SET - 设定位置等于 offset 字节。

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。）

如果 没有指定 whence，默认为 SEEK\_SET。

成功则返回 0；否则返回 -1。注意移动到 EOF 之后的位置不算错误

rewind -- 倒回文件指针的位置

语法：bool rewind ( resource handle )

将 handle 的文件位置指针设为文件流的开头。 如果成功则返回 TRUE，失败则返回 FALSE。 文件指针必须合法，并且指向由 fopen() 成功打开的文件。

### 例子

<?php

//以只读模式打开文件

$fp = fopen('data.txt', 'r') or die("文件打开失败");

echo ftell($fp)."<br>"; //输出刚打开文件指针默认位置，指针在文件开头位置为0

echo fread($fp, 10)."<br>";//读取文件中的前10个字符输出，指针位置发生了变化

echo ftell($fp)."<br>"; //读取文件前10个字符后，指针移动位置在第10个字节处

fseek($fp, 100, SEEK\_CUR); //将文件指针的位置，由当前位置向后移动100个字节数

echo ftell($fp)."<br>"; //文件位置在110个字节处

echo fread($fp, 10)."<br>"; //读取110到120字节数位置的字串，读取后指针位置为120

fseek($fp, -10, SEEK\_END); //又将指针移动到倒数10个字节位置处

echo fread($fp, 10)."<br>"; //输出文件中最后10个字符

rewind($fp); //又移动文件指针到文件的开头

echo ftell($fp)."<br>"; //指针在文件的开头位置，输出0

fclose($fp); //关闭文件资源

# 12.3.4简单留言板

<?php

header("Content-Type:text/html;charset=utf8");

$filename = "message.txt";

//如果用户提交了， 就写入文件， 按一定格式写入

if(isset($\_POST['dosubmit'])) {

//字段的分隔使用||, 行的分隔使用[n]

$mess = "{$\_POST['username']}||".time()."||{$\_POST['title']}||{$\_POST['content']}[n]";

writemessage($filename, $mess);

}

if(file\_exists($filename)) {

readmessage($filename);

}

function writemessage($filename, $mess) {

$fp = fopen($filename, "a");

fwrite($fp, $mess);

fclose($fp);

}

function readmessage($filename) {

$mess = file\_get\_contents($filename);

$mess = rtrim($mess, "[n]");

$arrmess = explode("[n]", $mess);

foreach($arrmess as $m) {

list($username, $dt ,$title, $content) = explode("||", $m);

echo "<b>{$username}</b>, ".date("Y-m-d H:i").": <i>{$title}</i>, <u>{$content}</u><br><hr><br>";

}

}

?>

<form action="message.php" method="post">

用户: <input type="text" name="username" value="" /><br>

标题：<input type="text" name="title" value="" /><br>

内容：<textarea name="content" cols="40" rows="4"></textarea><br>

<input type="submit" name="dosubmit" value="留言" /><br>

</form>

# 12.3.5文件的锁定机制

flock -- 轻便的咨询文件锁定

语法： bool flock ( int handle, int operation [, int &wouldblock] )

PHP 支持以咨询方式（也就是说所有访问程序必须使用同一方式锁定, 否则它不会工作）锁定全部文件的一种轻便方法。

handle 必须是一个已经打开的文件指针。

operation 可以是以下值之一：

要取得共享锁定（读取程序），将 operation 设为 LOCK\_SH。

要取得独占锁定（写入程序），将 operation 设为 LOCK\_EX。

要释放锁定（无论共享或独占），将 operation 设为 LOCK\_UN。

如果你不希望 flock() 在锁定时堵塞，则给 operation 加上 LOCK\_NB。

如果成功则返回 TRUE，失败则返回 FALSE。

# 12.4.1文件上传的设置

### 客户端上传设置

在B/S程序中文件上传已经成为一个常用功能。其目的是客户可以通过浏览器(Browser)将文件上传到服务器（Server）上的指定目录。

PHP中文件上传的基础知识：

客户端form表单

服务器端对上传文件的操作

### 客户端文件上传的form表单：

<html>

<head><title>文件上传</title></head>

<body>

<form action="upload.php" method="post"

enctype="multipart/form-data">

<input type="hidden" name="MAX\_FILE\_SIZE" value="1000000">

选择文件：<input type="file" name="myfile">

<input type="submit" value="上传文件">

</form>

</body>

</html>

### 注意几个特征属性:

POST方法:

表单最常用的功能,向目标页面传递变量,我们在上传文件的时候，会在表单中设置相应的属性，来完成文件的传递

enctype="multipart/form-data"

这样服务器就会知道，我们要传递一个文件,这样服务器可以知道上载的文件带有常规的表单信息。

MAX\_FILE\_SIZE

此字段必须在文件输入字段之前，控制最大的传递文件的大小(字节)

<input type="file" name="userfile">

设置浏览器文件输入浏览按钮

### PHP配置文件中与文件上传有关的选项

|  |  |  |
| --- | --- | --- |
| **指令名** | **默认值** | **功能描述** |
| **file\_uploads** | **ON** | **是否开启文件上传** |
| **upload\_max\_filesize** | **2M** | **限制PHP处理上传文件大小的最大值，此值必须小于post\_max\_size** |
| **post\_max\_size** | **8M** | **限制通过POST方法可以接受信息的最大值，也就是整个POST请求的提交值。此值必须大于upload\_max\_filesize** |
| **upload\_tmp\_dir** | **NULL** | **上传文件存放的临时路径，可以是绝对路径。默认NULL则使用系统的临时目录。** |

### $\_FILES多维数组

超级全局数组$\_FILES

1、$\_FILES["myfile"]["name"]中的值是:

客户端文件系统的文件的名称

2、$\_FILES["myfile"]["type"]中的值是:

客户端传递的文件的类型

3、$\_FILES["myfile"]["size"]中的值是:

文件的字节的大小

4、$\_FILES["myfile"]["tmp\_name"]中的值是：

文件被上传后在服务器存储的临时全路径

5、$\_FILES["myfile"]["error"]中的值是:

文件上传的错误代码－php 4.2以后增加的功能

# 12.4.2文件上传的处理

### 在服务器端通过PHP处理上传

上传文件的接收和处理是通过PHP脚本来处理的，具体需要通过以下三个方面信息：

设置PHP配置文件中的指令:用于精细地调节PHP的文件上传功能。

$\_FILES多维数组：用于存储各种与上传文件有关的信息，其他数据还是使用$\_POST获取。

PHP的文件上传处理函数：用于上传文件的后续处理。

### 存储在$\_FILES["myfile"]["error"]中的值

伴随文件上传时产生的错误信息代码是在PHP4.2.0版本中引入的，具体如下：

值为0：表示没有发生任何错误。

值为1：表示上传文件的大小超出了约定值。文件大小的最大值是在PHP配置文件中指定的，该指令是：upload\_max\_filesize。

值为2：表示上传文件大小超出了HTML表单隐藏域属性的MAX＿FILE＿SIZE元素所指定的最大值。

值为3：表示文件只被部分上传。

值为4：表示没有上传任何文件。

值为6：表示找不到临时文件夹。PHP 4.3.10 和 PHP 5.0.3 引进。

值为7：表示文件写入失败。PHP 5.1.0 引进。

### 错误值对应的常量

UPLOAD\_ERR\_OK ： 对应值0

UPLOAD\_ERR\_INI\_SIZE ： 对应值1

UPLOAD\_ERR\_FORM\_SIZE ： 对应值2

UPLOAD\_ERR\_PARTIAL ： 对应值3

UPLOAD\_ERR\_NO\_FILE ： 对应值4

UPLOAD\_ERR\_NO\_TMP\_DIR ： 对应值6

UPLOAD\_ERR\_CANT\_WRITE ： 对应值7

### 常见数据格式（MIME）

|  |  |
| --- | --- |
| 文件类型 | MIME类型 |
| **图片文件** | **image/gif，image/jpg，image/jpeg，image/png，image/x-png** |
| **纯文本和HTML** | **text/txt，text/plain，text/html** |
| **二进制文件** | **application/octet-stream** |
| **音频格式** | **audio/basic** |
| **视频格式** | **video/mpeg** |

# 12.4.3处理多个文件上传

当需要上传多个文件的情况,有两种实现的解决方法：

使用不同的表单元素

<input type="file" name="file\_a">

<input type="file" name="file\_b">

<input type="file" name="file\_b">

使用数组格式的表单元素

<input type="file" name="file[]">

<input type="file" name="file[]">

<input type="file" name="file[]">

# 12.4.4文件上传类

# 12.4.5文件下载

<?php

//文件下载练习

$filename="./upload/aa.png";

$basename=pathinfo($filename);

header("Content-Type: image/png"); //指定下载文件类型的

header("Content-Disposition:attachment;filename=".

$basename["basename"]);

//指定下载文件的描述信息

header("Content-Length:".filesize($filename)); //指定文件大小的

readfile($filename);//将内容输出，以便下载。

?>