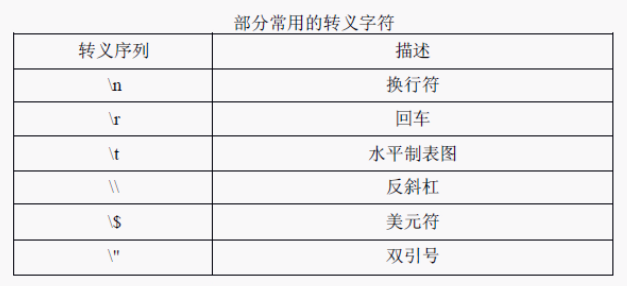
Php注意实现：

中文乱码问题：<**meta http-equiv="Content-Type"content="text/html;charset=utf-8"**>

php可以跟html嵌套使用，在使用php语言时，需要将代码外用<?php 内容?>括起来

布尔型真为1 假为空



常用转义字符：



常用类型标识符：

单双引号：双引号解析变量，单引号不解析变量；

Array打头的一般是创建新数组，建议前面进行赋值；

编写格式种类：

简短风格：

<? echo”<p>My php</p>”; ?>

Script风格：

<script language=”php”>

Echo “<p>My php</p>”;

<script>

ASP风格：

<% echo ” <p>My php</p>”; %>

常用代码实例

简单的提交数据：

Demo1.php:

<meta http-equiv=”Content-Type”content=”text/html;charset=GBK”>

<form method=”post” action=”Demo2.php”>

姓名：<input type=”text” name=”username” value=”123”/>

<input type=”submit” value=”提交”/>;

<form>

Demo2.php:

<?php

echo $\_POST[‘uesrname’];

?>

解析：method=”post/get”要与Demo1.php中的method一，样post:将与指定的action服务器地址建立联系，一旦建立，将数据发送给指定服务器。Type=”submit”提交按钮，将数据提交到Web服务器。$\_POST[‘username’] 将上一张表单的value值提取出来;

php数组map用法；

$SumArray=array(‘小红’=>16,‘张三’=>88)；

echo $SumArray[‘小红’]->16；

添加：$SumArray[‘李四’]=66;

Echo $SumArray[‘李四’]->66

Php map数组的遍历：

Foreach($SumArray as $key=>$value){//as $value

Echo $key.”=>”.$value.”<br/>”;

}

解析：遍历数组内容，参数一：数组，参数二：下标名字，参数三：下标名字对应内容。类似C迭代器，遍历全部输出后结束。

数组赋值：

List($var1, $var2, $var3)=$SumArray;=>将SumArray里的值赋值给list，list中取了前三个，并且赋值给指定变量；

Echo $var1;=>输出SumArray里的第一个值；

Tips：无法赋值map类型，只能赋值下标为0-n的数组

获取当前时间：

初始化时区：*date\_default\_timezone\_set*(**'Asia/Shanghai'**);

输出当前时间：**echo** *date*(**'Y-m-d H:i:s'**);//Y年 m月 d日 H时 i分 s秒 详情看php手册，搜date，可，加参数二：时间戳，如加入fileatime(‘路径/文件’)获取这个文件的最后访问时间；

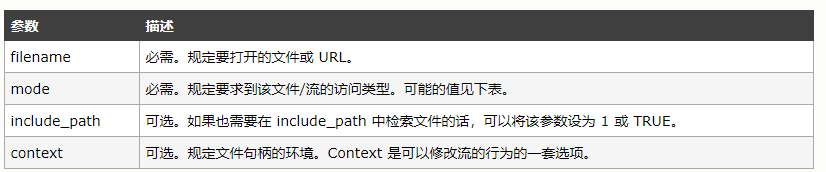
打开文件与修改:

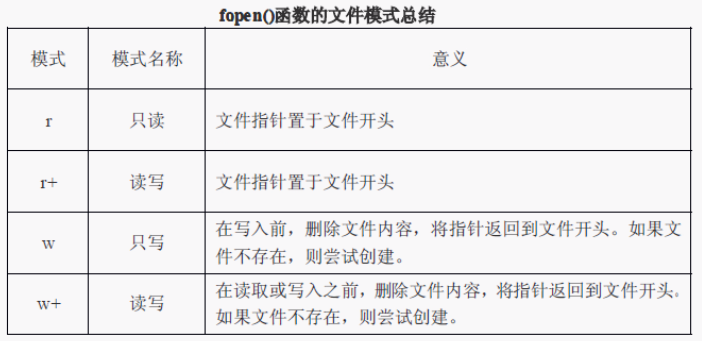
$fp=fopen(‘filename’,‘mode，看下表’,’include\_path’,’context’)；

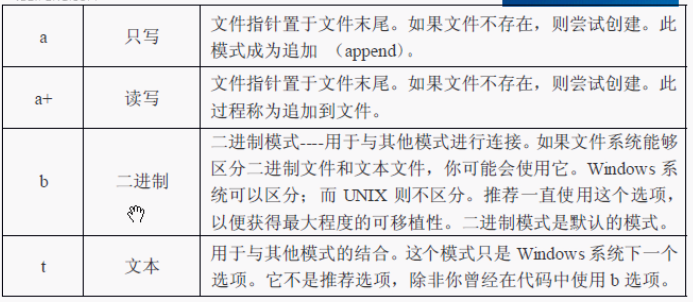
Fclose($fp)//清除fopen打开的缓存，在用完$fp时用这个；

Fopen返回值句柄，就是一个特定的指针指向这个文件；

例：fopen(‘filename’,’ab’)//只写+二进制；





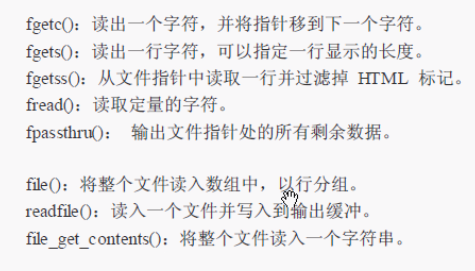


$fp=Fopen(‘filename’,’w’)//指定目录没有这个文件时，创建这个文件，如果又这个文件，清楚内容，光标回到开头。

Fwrite($fp,’字符串’，lenght)//写入数据,length为指定长度

File\_put\_contents(‘filename’,’字符串’)；//写入数据，php5新函数

读出/查看文件：



$fp=Fopen(‘filename’,’r’)；//打开文件只读，光标在开头

Fgetc($fp);//返回下一个字符，指针向下移动一位字符

Fgets($fp,lenght);//返回光标除向下length-1长度的字符

Fgetss($fp);//返回字符串中，取消html标记

Fread($fp,length(必选));//返回length个字符

Fpassthru($fp);//输出光标指针后所有数据，返回剩余字符串长度

Readfile(‘filename’);//输出全部字符串，返回字符串长度

File(‘filename’);//返回数组，按照文件中字符串每行来分组

File\_get\_contents($fp);//将数据读入缓冲区，通过echo打印，自带fclose();

Feof($fp);//判断光标是否到结尾，可配合while，返回值布尔型，光标到结尾返回true,否则false;

File\_exists(‘filename’);//判断目标文件是否存在，返回布尔型，存在返回true，否则false；

Unlink(‘filename’);//删除指定目标文件

Rewind(‘filename’);//光标指针回调到首位

Ftell($fp);//当前光标位置，默认首位为0

Fseek($fp,number);//将光标移动到number位置；

Flock($fp,参数二);//锁定释放文件，参数二：LOCK\_EX锁定，LOCK\_UN释放

目录查看与修改：

$dir=Opendir(‘绝对路径/相对路径’)；//设置句柄

Readdir($dir);//返回目录名字，指针指向下一个，类型字符串

Scandir($dir);//返回数组，将所有文件夹，文件放入数组,print\_r输出查看；

Rmdir(‘路径\目录’);//删除指定目录

Rename(‘目录/文件’,’字符串’)；//修改目录或者文件的名字

Closedir（$dir）;//关闭目录

自定义函数：

Function name(){ 自定义内容 }//自定义函数，可传参，跟C差不多;

Function name(&$a){ //函数传参变量前加 & 可以将变量从局部变成全局;

Global $b;//函数中定义变量前加global 将局部变成全局；global全球的意思

}

$GLOBALS[‘a’]=5;//定义超级全局变量，返回类型数组。

文件包含：

//定义一个文件function.php存放分装函数，这里简称函数库。

Include ‘函数库文件路径’//调用函数库,要使用函数库的函数,用这个调用；

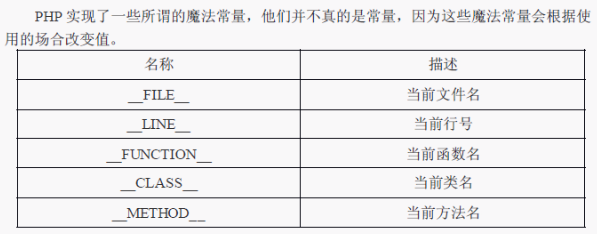
//定义一个Demo1.php内容echo 测试;

Include ‘Demo1.php’//可以将Demo1.php代码包含进来,此时就会输出->测试；

Include\_once ‘Demo1.php’//只会包含一次，include可重复包含；

Require/require\_once 跟include/include\_once差不多//require出错会直接报错停止执行，include出错会报错，还是会继续执行下面的代码；

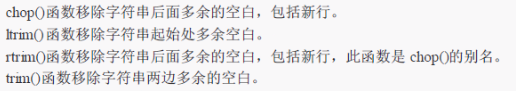
魔法常量：（滑稽



\_\_FILE\_\_//当前文档文件目录，返回字符串，常用方法，用dirname(\_\_FILE\_\_)获取当前文档的目录位置，后面.上文件名，这样获取文件完整目录，使用在require/include打开；

\_\_FUNCTION\_\_//返回包含它的函数的函数名；

字符串处理：



Tips：字符串中左右两边存在空格，在web中不显示，但在文本中显示；

Chop($str)//清理两边的空格以及换行，trim($str)不会；

Ltrim($str)//清理左边的空格，L->left的意思,R->Right的意思；

nl2br($str)//将\n换成<br/>,这样在web中就能显示换行;

$str=’<strong>啦啦啦</strong>’(下四个使用

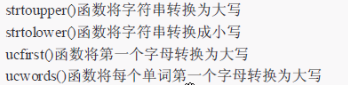
htmlentitles($str)//所有字符转换成html格式(会把中文也乱码了，屌用？

Htmlspecialchars($str)//转换所有特殊字符成html格式(比上面有点屌用！

Strip\_tags($str)//将html字符去除；

Addslashes($str)//为所有特殊字符加\，插入数据库使用(目前用处不大。。

Stripcslashes($str)// 为所有特殊字符前去除转义字符\，数据库取出使用(me too..

字符串转大写小写问题，简单看图= =

Str\_pad(‘字符串’,length,’字符串2’,参数4)//填充字符串，length定义填充长度，超过字符串长度部分默认用空格，否则用字符串2，参数4:STR\_PAD\_LEFT左边填充，..\_RIGHT右边填充，..\_BOTH两边填充；

Explode(‘#’，$str)//切割字符串，返回数组，根据#字符分割数组存放；

Implode(‘#’,$Array)//组合数组，返回字符串，用#字符链接；

Strtok($str，’,.#’)//字符串分割，称令牌,返回值分割第一次的字符串；

解析案例：

$str=**"i,will.#back"**;//定义字符  
$tok=*strtok*($str,**',.#'**);//将第一次的分割出的存放在tok里  
**while**($tok){//用while循环，每次tok会存放下一次分割出的字符  
 **echo** $tok.**'<br/>'**;//输出  
 $tok=*strtok*(**' ,.#'**);//这里不用使用字符，直接用分割符，然后自动向下一个。(不是很懂，套版用吧。  
}

//这个案例很实用，可以根据不同的字符来分割字符串。

Substr($str,start,lenght)//截取字符串，从start开始截取Length长度字符；

Str\_split($str，lenght)//字符串转数组，根据length长度存放数组，默认1；

Strrev($str)//逆置字符；

Strcmp($str1,$str2)//比较字符串，返回值int,对较ASCII,相等->0,小于->-1，大于->1;

Strcasecmp($str1,$str2)//跟上面一样，但是不区分大小写；

Strnatcmp($str1,$str2)//自然比较，类似数字字符串转数字用逻辑比较；

Strcspn($str1,$str2,start,lenght)//返回比较两个字符串相同部分的长度,可以指定开始位置与长度；(不常用

Strlen($str)//返回字符串长度；

Substr\_count($str,$str2)//返回寻找$str2字符串在$str字符串中出现的次数；

Strstr($str,$str2)//返回在$str中从$str2字符串开始输出之后的字符串;

Stristr($str,$str2)//不区分大小写，功能跟上面一样；

Strops($str,$str2)//查找字符串位置，返回在$str中$str2首次出现位置,0开始；

Strrpos($str,$str2)// 查找字符串位置,返回在$str中$str2最后出现位置,0开始；

Stripos($str,$str2)// 同上，不区分大小写；

Str\_replace($str,$str1,$str2)//替换字符串,参数一：要替换的字符串，参数二：替换成的字符串，参数三：原字符串,返回替换后字符串；

Substr\_replace($str,$str2,start,lenght)//选择位置替换字符串，将$str中，start开始length字符串替换成$str2,返回替换后的字符串;

处理中文字符：

Mb\_strlen($str,’utf-8’)//中文字符串长度，参数二是字符串编码；

Mb\_substr($str,start,length,’utf-8’)//都一样，加Mb可以加编码方式；

//大部分字符串处理加mb\_可以在最后添加字符编码方式，但有时候添加了还是乱码，没辙。。。；

常用代码：

echo =>echo “111”输出内容返回：void无返回值

print =>printf “111”输出内容返回值：整型 echo print “222”->2221

printf()和sprintf()是C语言模式=>printf(“我%d岁了”,1);

print\_r()=>打印全部内容

$temp=>创建变量，自动类型,类型转换(float)$temp

$temp=$double =>隐式转换，$temp的值与类型等于$double变量

$temp=\_POST[‘name’]=>将上一张表单中指定name的value指赋值给$temp；

正则表达式：

Tips：语法Perl风格，模式规则，在字符串两边加/->/php/；

Preg\_match(‘/模式/’,’字符串’)//执行正则表达式；

解析：如果函数模式与字符串比较后：成立返回1,否则0;

Preg\_match(‘/ph+p/’,’phhhp’) =>1,+号是前导字符至少一个；

Preg\_match(‘/ph\*p/’,’pp’) =>1,\*号是前导字符零个或者很多个；

Preg\_match(‘/ph?p/’,’pp’) =>1,?号是前导字符另个或者一个；

Preg\_match(‘/ph.p/’,’phxp’) =>1,.指任意字符.\*就是前导任意字符零个或者多个；

Preg\_match(‘/ph{3}p/’,’phhhp’) =>1,{3}表示前导字符要3个；

Preg\_match(‘/ph{2,5}p/’,’phhhhp’) =>1，{2,5}表示前导字符个数在[2,5]区间；

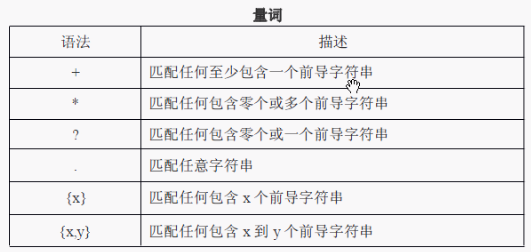
Preg\_match(‘/ph{3,}p/’,’phhhhhp’) =>1,{3,}表示前导字符至少3个；

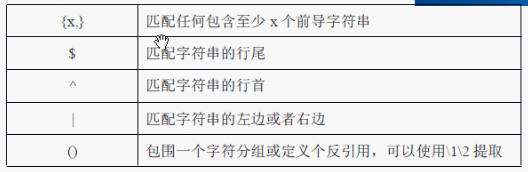
Preg\_match(‘/php$/’,’xxxxxxphp’) =>1,$表示前面的字符串从尾巴进行匹配；

Preg\_match(‘/^php/’,’phpxxxxx’) =>1,^表示后面字符串从开头开始匹配,^php$就跟==一样了；

Preg\_match(‘/php|asp/’,’asp’) =>1,|就是或者的意思，匹配两边其中一个就OK；

//如果要匹配这些量词，需要加上\转义；





元字符：

Preg\_match(‘/[abcd]/’,’asp’) =>1,[abcd]里面有一个字符匹配就OK,可以[a-z]这样匹配[a-z]范围内的;

Preg\_match(‘/[a-zA-z0-9]/’,’asp’) =>1,[ a-zA-z0-9]这样就全匹配了a-z A-z 0-9都可以；

Preg\_match(‘/[a-zA-z0-9][wer]/’,’asp’) =>0,这样前面匹配的后面要紧跟着[wer]范围也要匹配；

Preg\_match(‘/[^a-z]/’,’www’) =>1,[^a-z]表示出了a-z的字符串；

Preg\_match(‘/[/w]/’,’php’) =>1,[/w]=[a-zA-z0-9]php里为了方便；

Preg\_match(‘/[/W]/’,’php’) =>0,[/W]=[^a-zA-z0-9]除了这些；

Preg\_match(‘/[/d]/’,’php’) =>0,[/d]=[0-9]匹配数字；

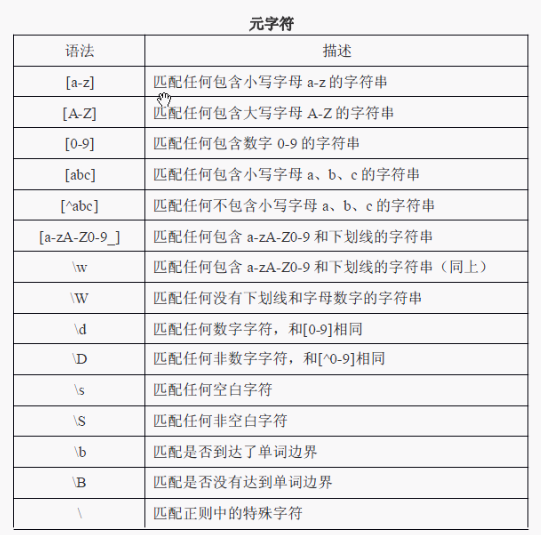
Preg\_match(‘/[/D]/’,’php’) =>1,[/D]=[^0-9]匹配非数字；

Preg\_match(‘/[/s]/’,’php’) =>0,[/s]匹配任何空白字符；

Preg\_match(‘/[/S]/’,’php’) =>1,[/S]匹配没有空白字符；

Preg\_match(‘/php\b/’,’php lla’) =>1,\b表示到了字符的边界，边界可以指空格；

Preg\_match(‘/php\B/’,’phppp’) =>1,\B表示没有到字符边界；



修饰符：

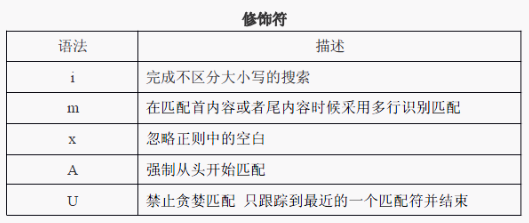
Tips：修饰符在/php/i放在/外面；

Preg\_match(‘/php/i’,’PHP’) =>1,不区分大小写；

Preg\_match(‘/php$/m’,”this is php\n,dont java”) =>1,多行匹配,如果遇到换行也应该承认是结尾；

Preg\_match(‘/p h p/x’,’PHP’) =>1，忽略规则模式中的空白；

Preg\_match(‘/php/U’,’PHP’) =>



Perl风格函数：

Preg\_match(‘/模式/’,’字符串’) =>匹配字符串；

Preg\_grep(‘/模式/’,$Array) =>对所有数组进行匹配，匹配返回进入数组，返回数组；

Preg\_match\_all(‘/模式/’,’字符串’,$Array) =>全局正则表达式匹配，将所有匹配存放到$Array数组里，返回二维数组；

Preg\_quote(‘字符串’) =>定界正则，对所有的特殊字符加\返回，返回字符串；

Preg\_replace(‘/模式/’,’替换的字符串’,‘原字符串’) =>搜索匹配的结果，然后替换掉,返回字符串；

Preg\_split(‘/模式/’,’字符串’) =>不区分大小写划分不用的字符串，返回数组；

日期与时间：

初始化时区：*date\_default\_timezone\_set*(**'Asia/Shanghai'**);

Checkdate(int month,int day,int year) =>验证日期，提供日期有效返回true,否则false,返回类型布尔值；

Date(‘y-m-d’) =>返回日期字符串，详情看手册；

Dete常用参数解析：

可以放两个参数---参数一：日期和时间的格式化，参数二：时间戳；

d-day日期，有前导数，两位数,单数默认前导0；

D-day日期，文本表示Mon-Sun;

Y-年期，四位数的年份；

y-年期，两位数的年份；

n-月期，没前导数；

m-月期,有前导数；

M-月期,英文月期缩写；

h-小时，十二小时制；

H-小时，二十四小时制；

i-分钟，有前导数；

z-今年年份中的第几天；

T-本机所在时区；

s-秒数，有前导数；

1. 小写上午或下午值,am/pm;

Gettimeofday() =>取得当前时间,返回数组，详情看手册，参数’sec’获取当前时间戳；

Getdate() =>取得时间戳，返回数组，详情看手册；

Time() =>获取当前时间戳；

Mktime(h,i,s,n,j,Y) =>获取特定时间戳，指定时间；

Strtotime() =>将任何英文文本描述转换成时间戳,例：2018-10-10 8：8：8，返回数值；

Getlastmod() =>获取当前文件的修改时间戳；

Putenv() =>配置系统环境变量，例：TZ=Asia/Shanghai 设置当前时区；

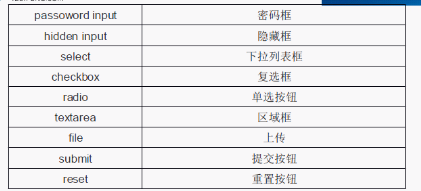
Date\_default\_timezone\_get() =>获取当前时区；

Date\_default\_timezone\_set() =>设置当前时区；

Localtime() =>取得本地时间，返回值数组，详情看手册；

Microtime() =>获取当前UNIX时间戳和微秒，两个数相加获得精准的秒数；

表单与验证:

常用表单代码；

Tips:在执行header()函数要注意，之前不能有任何浏览器输出,包括echo,html的任何输出,所以将header()函数放在第一行最保险,或者ob\_start()，开启缓冲（少用。

Header(“location:URL地址”) =>重新导向一个URL；

//例：内部跳转：header(“location:Demo2.php”),外部跳转：header(“location:http://www.baidu.com”);

header(“Content-Type:text.html;charset=utf-8”) =>设置页面编码；

提交数据：

代码模版：

<form method=”post/get” action=”Demo2.php”>

<input type=”text” name=”username” /><br/>

<input type=”submit” value=”提交”>

</form>

Tips:js中有个history.back()，回退一次，超级全局变量可以跨页面，cookie不占服务器资源；

$\_POST[“username”] =>超级全局变量，返回表单提交的信息，参数是input的name,获取指定name的value;

$\_GET[“username”] =>超级全局变量，返回表单提交的信息，参数是input的name,获取指定name的value;

//<a href=”Demo2.php?username=111”>提交</a> 默认为get传输，传输name=username,value=111;

Exit =>结束语句，下面所以代码不执行；

Setcookie($name,$str,time()当前时间+n秒过期时间) =>设置网页的cookie,详情看php手册，$str设置为空，为删除cookie，time()时间到了后 刷新页面isset[$\_COOKIE]返回false，否则返回true;

Isset($\_COOKIE[‘name’]) =>判断COOKIE是否存在，返回布尔值；

unset($\_SESSION[‘name’]) =>删除变量；

Tips:session是存放在服务器端的，一般存放1440秒，如果没有任何操作，或者关闭网页会自动销毁，及时性的。

Session\_start() =>开启session,输出$\_SESSION 需要开启这个;

$\_SESSION[‘name’]=’Lee’ =>创建session;

Session\_destroy() =>销毁所有$\_SESSION;

函数：

Gettype($temp)=>获得类型，返回$temp类型

Settype($temp，“类型”)=>类型转换，参数一变量，参数二转换类型

Insert($temp)=>判断变量是否存在,存在返回1，不存在返回空

Unsert($temp)=>销毁变量，将$temp变量移除

Empty($temp)=>检查是否非空，返回值布尔型

is\_int($temp)=>类型判断,判断是否为Int性，is\_double/string….

Intval($temp)=>临时转换,返回$temp的int型，$temp本身不变,intval()/floatval()/strval();

Define(“name”,value)=>定义name为value， 一旦被定义不能再次更改；

Is\_numeric()=>判断数字，返回值布尔型

Rand(min,max)=>产生随机数，范围为min-max，不写参数产生整数范围

Mt\_rand(min,max)=>同上面，速度是上面的四倍，更快。

Getrandmax()=>产生最大随机值

Mt\_getrandmax()=>产生比上面还大的最大随机值

Number\_format()=>格式化数据。参数一：数字变量，参数二：保留小数位，参数三：小数点符号，参数四：整数每三位隔开符号。

Array\_unique($Array)=>不重复数组，类似set，参数为数组变量，将值相同的去除

Array\_flip($Array)=>交换数组中的键和值，值与下标进行交换；

Count($Array)=>返回数组大小，返回数组内的元素个数。

Sort($Array)=>数组排序，无返回值，跟C类似。Rsort反向。Tips：没保留索引关系，改变了下标值，但内容还是在原来的位置，下标值指向的内容也会更改。

Assort($Array)=>数组排序，保持索引关系，更改下标，下标所对应的值会跟着下标更改。

Shuffle($Array)=>数组随机打乱，保留键，打乱值。

Array\_reverse($Array)=>反向排序，返回值数组；

Array\_unshift($Array,’str’)=>在开头插入数据，返回值布尔型，插入成功返回1，否则空

Array\_push($Array,’str’)=>在结尾插入数据，返回值布尔型，插入成功返回1，否则返回空

Array\_shift($Array)=>删除开头的元素。

Array\_rand($Array,number)=>获取数组键，参数一数组，参数二随机获取几个键,如果获取一个键，返回一个整数，多个键返回数组。

Current($Array)=>获取指针的当前元素，默认情况指针在第一条数据，不会将指针移到下一个；

Next($Array)=>指针向下一位；

Prev($Array)=>指针向上一位；

Reset($Array)=>指针重新指向第一个单元；

Sizeof($Array)=>返回数组大小；

Array\_count\_values($Array)=>统计数组中所以值出现的次数；

Extract($Array)=>标量函数，将数组里的键转换成变量。Echo $键 可以输出键指向的值；

Basename(‘路径’)=>返回文件的名称+扩展名；

Dirname(‘路径’)=>返回路径的目录部分，就是上级全部地址

Pathinfo(‘路径’)=>将目录每一级都存放在数组里，返回数组。可Print\_r查看；

Realpath(‘部分路径或文件名’)=>返回绝对路径,就是这个文件的地址；

Filesize(‘部分路径或文件名’)=>返回文件大小，返回字节；

Round(小数，number)=>对浮点型四舍五入，返回类型double，参数二保留位数；

Disk\_free\_space(‘路径’)=>获取可用空间，返回字节；

Disk\_total\_space(‘路径’)=>获取总空间,返回字节；

Fileatime(‘路径’)=>获取最后打开时间的时间戳；

Filectime(‘路径’)=>获取最后权限修改时间的时间戳；

Filemtime(‘路径’)=>获取最后内容修改时间的时间戳；

Md5(‘字符串’) =>加密函数，返回加密内容；