**PHP 常用函数**

## 字符串函数(important)

|  |
| --- |
| addcslashes // 像 C 语言一样使用反斜线转义字符串中的字符 addslashes // 使用反斜线引用字符串 ★bin2hex // 将二进制数据转换成十六进制表示 chop // rtrim() 的别名函数 ★★chr // 返回相对应于ascii码值所指定的单个字符。 chunk\_split // 将字符串分割成小块 convert\_cyr\_string // 将字符由一种 Cyrillic 字符转换成另一种 convert\_uudecode // Decode a uuencoded string convert\_uuencode // Uuencode a string count\_chars // 返回字符串所用字符的信息 crc32 // 计算一个字符串的 crc32 多项式 crypt // 将字符串用 DES 编码加密 ★★★★★echo // 输出一个或者是多个字符串 ★★★★★explode // 使用一个字符串分割另一个字符串 fprintf // 格式化一个字符串并输出 get\_html\_translation\_table // Returns the translation table used by htmlspecialchars() and htmlentities() hebrev // Convert logical Hebrew text to visual text hebrevc // Convert logical Hebrew text to visual text with newline conversion ★★★html\_entity\_decode // 将所有的 HTML 实体转换成对应的字符 ★★★htmlentities // 将所有的字符转换成 HTML 实体 ★★★htmlspecialchars\_decode // 将特定的 HTML 实体转换为相应的字符 ★★★htmlspecialchars // 将特定的字符转换成 HTML 实体 ★★★★★implode // 将一个数组的所有元素连接成一个字符串 join // implode()函数的别名 levenshtein // Calculate Levenshtein distance between two strings localeconv // Get numeric formatting information ★★★★★ltrim // 去除一个字符串左边的空白(或者是其他的字符) md5\_file // 将一个给定的文件用 MD5 编码 ★★★★★md5 // 将一个给定的字符串用 MD5 编码 metaphone // Calculate the metaphone key of a string money\_format // 将一个数字按照货币的格式进行格式化 nl\_langinfo // Query language and locale information nl2br // 在字符串中插入 HTML 换行标签。 number\_format // Format a number with grouped thousands ★★ord // 返回一个字符的 ASCII 值 parse\_str // 解析字符串变量 ★★★★print // 输出一个字符串 ★★★★printf // 格式化并输出该字符串 quoted\_printable\_decode // Convert a quoted-printable string to an 8 bit string quotemeta // Quote meta characters ★★★★★rtrim // 去除一个字符串右边的空白(或者是其他的字符) setlocale // Set locale information sha1\_file // 将一个文件用 SHA1 进行编码(加密) sha1 // 将一个字符串用 SHA1 进行编码(加密) similar\_text // 计算两个字符串之间的相似性 soundex // 计算一个字符串的 soundex key sprintf // 返回一个格式化后的字符串 sscanf // Parses input from a string according to a format str\_ireplace // Case-insensitive version of str\_replace(). str\_pad // Pad a string to a certain length with another string ★str\_repeat // 重复字符串 str\_replace // 替换指定的字符串 str\_rot13 // Perform the rot13 transform on a string str\_shuffle // Randomly shuffles a string ★★★★★str\_split // 将一个字符串转换成一个数组 ★★★str\_word\_count // 返回一个字符串中单词的个数 strcasecmp // Binary safe case-insensitive string comparison strchr // strstr() 的别名 strcmp // 字符串比较(二进制字符串比较) strcoll // Locale based string comparison strcspn // Find length of initial segment not matching mask ★★★★★strip\_tags // 将一个字符串中的 HTML 和 PHP 标记去除 stripcslashes // 去除使用像 C 语言方式添加的反斜杠。 ★★★★★stripos // Find position of first occurrence of a case-insensitive string 在字符串中不区分大小写的查找字符(或者字符串)第一次出现的位置 stripslashes // 去除使用函数 addslashes()添加的反斜杠。 stristr // Case-insensitive strstr() ★★★★★strlen // 取得字符串的长度 strnatcasecmp // Case insensitive string comparisons using a "natural order" algorithm strnatcmp // String comparisons using a "natural order" algorithm ★★★★strncasecmp // 区分大小写比较两个字符串。 strncmp // Binary safe string comparison of the first n characters strpbrk // Search a string for any of a set of characters strpos // 查找一个字符在一个字符串中第一次出现的位置。 ★★★★★strrchr // 返回从某字符最后一次出现的位置开始一直到字符串结束的子串。 ★★★strrev // 将字符串颠倒输出 ★★★★★strripos // 返回某字符在字符串中最后一次出现的位置(不区分大小写)。 ★★★★★strrpos // 返回某字符在字符串中最后一次出现的位置(区分大小写)。 strspn // Find length of initial segment matching mask ★★★★★strstr // 返回从某字符第一次出现的位置开始一直到字符串结束的子串(包含该字符)。 strtok // Tokenize string ★★★strtolower // 将字符串所有字符转换为小写 ★★★strtoupper // 将字符串所有字符转换为大写 strtr // Translate certain characters substr\_compare // Binary safe optionally case insensitive comparison of 2 strings from an offset, up to length characters ★★★substr\_count // 计算子串出现的次数(区分大小写) substr\_replace // 字串替换 ★★★★★substr // 返回指定位置的字串 ★★★★★trim // 去掉字符串左右两边的空白(或者其他字符) ★★★ucfirst // 使一个字符串的第一个字符大写 ★★★ucwords // 将一个字符串的每个单词的第一个字母大写 vfprintf // Write a formatted string to a stream vprintf // 输出一个格式化后的字符串 vsprintf // 返回一个格式化后的字符串 ★★★wordwrap // 使字符串在指定位置换行  ★★chr // 返回相对应于ascii码值所指定 |

## 数组函数(important)

|  |
| --- |
| array\_change\_key\_case // 返回字符串键名全为小写或大写的数组 array\_chunk // 将一个数组分割成多个 array\_combine // 创建一个数组，用一个数组的值作为其键名，另一个数组的值作为其值 array\_count\_values // 统计数组中所有的值出现的次数 array\_diff\_assoc // 带索引检查计算数组的差集 array\_diff\_key // 使用键名比较计算数组的差集 array\_diff\_uassoc // 用用户提供的回调函数做索引检查来计算数组的差集 array\_diff\_ukey // 用回调函数对键名比较计算数组的差集 array\_diff // 计算数组的差集 array\_fill // 用给定的值填充数组 array\_filter // 用回调函数过滤数组中的单元 array\_flip // 交换数组中的键和值 array\_intersect\_assoc // 带索引检查计算数组的交集 array\_intersect\_key // 使用键名比较计算数组的交集 array\_intersect\_uassoc // 带索引检查计算数组的交集，用回调函数比较索引 array\_intersect\_ukey // 用回调函数比较键名来计算数组的交集 array\_intersect // 计算数组的交集 array\_key\_exists // 检查给定的键名或索引是否存在于数组中 array\_keys // 返回数组中所有的键名 array\_map // 将回调函数作用到给定数组的单元上 array\_merge\_recursive // 递归地合并一个或多个数组 ★★★★★array\_merge // 合并一个或多个数组 array\_multisort // 对多个数组或多维数组进行排序 array\_pad // 用值将数组填补到指定长度 ★★★★★array\_pop // 将数组最后一个单元弹出（出栈） array\_product // 计算数组中所有值的乘积 ★★★★★array\_push // 将一个或多个单元压入数组的末尾（入栈） ★★★array\_rand // 从数组中随机取出一个或多个单元 array\_reduce // 用回调函数迭代地将数组简化为单一的值 ★★★array\_reverse // 返回一个单元顺序相反的数组 ★★★★★array\_search // 在数组中搜索给定的值，如果成功则返回相应的键名 array\_shift // 将数组开头的单元移出数组 array\_slice // 从数组中取出一段 array\_splice // 把数组中的一部分去掉并用其它值取代 ★★★array\_sum // 计算数组中所有值的和 array\_udiff\_assoc // 带索引检查计算数组的差集，用回调函数比较数据 array\_udiff\_uassoc // 带索引检查计算数组的差集，用回调函数比较数据和索引 array\_udiff // 用回调函数比较数据来计算数组的差集 array\_uintersect\_assoc // 带索引检查计算数组的交集，用回调函数比较数据 array\_uintersect\_uassoc // 带索引检查计算数组的交集，用回调函数比较数据和索引 array\_uintersect // 计算数组的交集，用回调函数比较数据 ★★★★★array\_unique // 移除数组中重复的值 array\_unshift // 在数组开头插入一个或多个单元 array\_values // 返回数组中所有的值 array\_walk\_recursive // 对数组中的每个成员递归地应用用户函数 array\_walk // 对数组中的每个成员应用用户函数 ★★★★★array // 新建一个数组 ★★★★★★★★★★arsort // 对数组进行逆向排序并保持索引关系 ★★★★★★★★★★asort // 对数组进行排序并保持索引关系 compact // 建立一个数组，包括变量名和它们的值 count // 计算数组中的单元数目或对象中的属性个数 ★★★★current // 返回数组中的当前单元 ★★★★★each // 返回数组中当前的键／值对并将数组指针向前移动一步 ★★★★end // 将数组的内部指针指向最后一个单元 extract // 从数组中将变量导入到当前的符号表 ★★★★★★★★★★in\_array // 检查数组中是否存在某个值 ★★★★★key // 从关联数组中取得键名 ★★★★★★★★★★krsort // 对数组按照键名逆向排序 ★★★★★★★★★★ksort // 对数组按照键名排序 ★★★★list // 把数组中的值赋给一些变量 natcasesort // 用“自然排序”算法对数组进行不区分大小写字母的排序 natsort // 用“自然排序”算法对数组排序 ★★★★next // 将数组中的内部指针向前移动一位 pos // current() 的别名 ★★★★prev // 将数组的内部指针倒回一位 range // 建立一个包含指定范围单元的数组 ★★★★★reset // 将数组的内部指针指向第一个单元 ★★★★★rsort // 对数组逆向排序 shuffle // 将数组打乱 ★★★★★★★★★★sizeof // count() 的别名 ★★★★★sort // 对数组排序 uasort // 使用用户自定义的比较函数对数组中的值进行排序并保持索引关联 uksort // 使用用户自定义的比较函数对数组中的键名进行排序 usort // 使用用户自定义的比较函数对数组中的值进行排序 |

## 日历函数

|  |
| --- |
| cal\_days\_in\_month // Return the number of days in a month for a given year and calendar cal\_from\_jd // Converts from Julian Day Count to a supported calendar cal\_info // Returns information about a particular calendar cal\_to\_jd // Converts from a supported calendar to Julian Day Count easter\_date // Get Unix timestamp for midnight on Easter of a given year easter\_days // Get number of days after March 21 on which Easter falls for a given year FrenchToJD // Converts a date from the French Republican Calendar to a Julian Day Count GregorianToJD // Converts a Gregorian date to Julian Day Count JDDayOfWeek // Returns the day of the week JDMonthName // Returns a month name JDToFrench // Converts a Julian Day Count to the French Republican Calendar JDToGregorian // Converts Julian Day Count to Gregorian date jdtojewish // Converts a Julian day count to a Jewish calendar date JDToJulian // Converts a Julian Day Count to a Julian Calendar Date jdtounix // Convert Julian Day to Unix timestamp JewishToJD // Converts a date in the Jewish Calendar to Julian Day Count JulianToJD // Converts a Julian Calendar date to Julian Day Count unixtojd // Convert Unix timestamp to Julian Day |

## 时间函数(important)

|  |
| --- |
| checkdate // 验证一个格里高里日期 ★★★★★date\_default\_timezone\_get // 取得一个脚本中所有日期时间函数所使用的默认时区 ★★★★★date\_default\_timezone\_set // 设定用于一个脚本中所有日期时间函数的默认时区 date\_sunrise // 返回给定的日期与地点的日出时间 date\_sunset // 返回给定的日期与地点的日落时间 ★★★★★date // 格式化一个本地时间／日期 getdate // 取得日期／时间信息 gettimeofday // 取得当前时间 gmdate // 格式化一个 GMT/UTC 日期／时间 gmmktime // 取得 GMT 日期的 UNIX 时间戳 gmstrftime // 根据区域设置格式化 GMT/UTC 时间／日期 idate // 将本地时间日期格式化为整数 localtime // 取得本地时间 microtime // 返回当前 Unix 时间戳和微秒数 mktime // 取得一个日期的 Unix 时间戳 strftime // 根据区域设置格式化本地时间／日期 strptime // 解析由 strftime() 生成的日期／时间 strtotime // 将任何英文文本的日期时间描述解析为 Unix 时间戳 ★★★★★time // 返回当前的 Unix 时间戳 |

## PHP检测变量函数(important)

|  |
| --- |
| debug\_zval\_dump // Dumps a string representation of an internal zend value to output doubleval // floatval() 的别名 ★★★★★empty // 检查一个变量是否为空 ★★★★★floatval // 获取变量的浮点值 get\_defined\_vars // 返回由所有已定义变量所组成的数组 get\_resource\_type // 返回资源（resource）类型 ★★★★★gettype // 获取变量的类型 import\_request\_variables // 将 GET／POST／Cookie 变量导入到全局作用域中 ★★★★★intval // 获取变量的整数值 is\_array // 检测变量是否是数组 is\_bool // 检测变量是否是布尔型 is\_callable // 检测参数是否为合法的可调用结构 is\_double // is\_float() 的别名 is\_float // 检测变量是否是浮点型 is\_int // 检测变量是否是整数 is\_integer // is\_int() 的别名 is\_long // is\_int() 的别名 is\_null // 检测变量是否为 NULL is\_numeric // 检测变量是否为数字或数字字符串 is\_object // 检测变量是否是一个对象 is\_real // is\_float() 的别名 is\_resource // 检测变量是否为资源类型 is\_scalar // 检测变量是否是一个标量 is\_string // 检测变量是否是字符串 ★★★★★isset // 检测变量是否设置 ★★★★★print\_r // 打印关于变量或者数组的易于理解的信息。 serialize // 产生一个可存储的值的表示 ★★★★★settype // 设置变量的类型 ★★★★★strval // 获取变量的字符串值 unserialize // 从已存储的表示中创建 PHP 的值 ★★★★★unset // 释放给定的变量 ★★★★★var\_dump // 打印变量的相关信息 var\_export // 输出或返回一个变量的字符串表示 |

## FILE文件处理函数 (important)

|  |
| --- |
| ★★★★★basename // 返回路径中的文件名部分 chgrp // 改变文件所属的组 ★★★★★chmod // 改变文件模式 chown // 改变文件的所有者 clearstatcache // 清除文件状态缓存 ★★★★★copy // 拷贝文件 delete // 参见 unlink() 或 unset() ★★★★★dirname // 返回路径中的目录部分 ★★★★disk\_free\_space // 返回目录中的可用空间 ★★★★disk\_total\_space // 返回一个目录的磁盘总大小 diskfreespace // disk\_free\_space()的别名 ★★★★★fclose // 关闭一个已打开的文件指针 ★★★★★★★★★★feof // 测试文件指针是否到了文件结束的位置 fflush // 将缓冲内容输出到文件 ★★★★★fgetc // 从文件指针中读取字符 fgetcsv // 从文件指针中读入一行并解析 CSV 字段 ★★★★★★★★★★fgets // 从文件指针中读取一行 ★★★★★fgetss // 从文件指针中读取一行并过滤掉 HTML 标记 ★★★★★file\_exists // 检查文件或目录是否存在 ★★★★★★★★★★file\_get\_contents // 将整个文件读入一个字符串 file\_put\_contents // 将一个字符串写入文件 ★★★★★file // 把整个文件读入一个数组中 fileatime // 取得文件的上次访问时间 filectime // 取得文件的 inode 修改时间 filegroup // 取得文件的组 fileinode // 取得文件的 inode filemtime // 取得文件修改时间 fileowner // 取得文件的所有者 fileperms // 取得文件的权限 ★★★★★filesize // 取得文件大小 ★★★★★filetype // 取得文件类型 flock // 轻便的咨询文件锁定 fnmatch // 用模式匹配文件名 ★★★★★★★★★★fopen // 打开文件或者 URL fpassthru // 输出文件指针处的所有剩余数据 fputcsv // 将行格式化为 CSV 并写入文件指针 fputs // fwrite()的别名 ★★★★★fread // 读取文件（可安全用于二进制文件） fscanf // 从文件中格式化输入 fseek // 在文件指针中定位 fstat // 通过已打开的文件指针取得文件信息 ftell // 返回文件指针读/写的位置 ftruncate // 将文件截断到给定的长度 fwrite // 写入文件（可安全用于二进制文件） glob // 寻找与模式匹配的文件路径 ★★★★★★★★★★is\_dir // 判断给定文件名是否是一个目录 is\_executable // 判断给定文件名是否可执行 ★★★★★★★★★★is\_file // 判断给定文件名是否为一个正常的文件 is\_link // 判断给定文件名是否为一个符号连接 ★★★★★is\_readable // 判断给定文件名是否可读 is\_uploaded\_file // 判断文件是否是通过 HTTP POST 上传的 ★★★★★is\_writable // 判断给定的文件名是否可写 is\_writeable // is\_writable()的别名 link // 建立一个硬连接 linkinfo // 获取一个连接的信息 lstat // 给出一个文件或符号连接的信息 ★★★★★★★★★★mkdir // 新建目录 ★★★★★★★★★★move\_uploaded\_file // 将上传的文件移动到新位置 parse\_ini\_file // 解析一个配置文件 ★★★★★pathinfo // 返回文件路径的信息 ★★★★★pclose // 关闭进程文件指针 popen // 打开进程文件指针 readfile // 输出一个文件 readlink // 返回符号连接指向的目标 realpath // 返回规范化的绝对路径名 ★★★★★★★★★★rename // 重命名一个文件或目录 ★★★★★★★★★★rewind // 倒回文件指针的位置 ★★★★★★★★★★rmdir // 删除目录 set\_file\_buffer // stream\_set\_write\_buffer()的别名 stat // 给出文件的信息 symlink // 建立符号连接 tempnam // 建立一个具有唯一文件名的文件 ★★★★★tmpfile // 建立一个临时文件 touch // 设定文件的访问和修改时间 umask // 改变当前的 umask ★★★★★★★★★★unlink // 删除文件 |

## 目录函数(important)

|  |
| --- |
| ★★★★★mkdir// 新建目录★★★★★is\_dir// 判断给定文件名是否是一个目录  ★★★★★dirname// 返回路径中的目录部分  ★★★★★chdir // 改变目录  ★★★★★rmdir // 删除空目录 chroot // 改变根目录 dir // directory 类 ★★★★★closedir // 关闭目录句柄 getcwd // 取得当前工作目录 ★★★★★opendir // 打开目录句柄 ★★★★★readdir // 从目录句柄中读取条目 ★★★★★rewinddir // 倒回目录句柄 ★★★★★scandir // 列出指定路径中的文件和目录  // 综合应用实例 1：删除目录下的所有文件 function DeleteFile($path) {  $dh = opendir($path);  while (false !== ($filename = readdir($dh))) {  if ($filename == ‘.’ || $filename == ‘..’) {  continue;  }  if (is\_dir($path . $filename)) {  DeleteFile($path . $filename .”\\”);  } else if (is\_file($path . $filename)) {  unlink($path . $filename);  }  }  closedir($dh);  rmdir($path); }  // 综合应用实例 2：删除指定目录下的所有文件以及所有文件夹以及本身\*\*\*强烈推荐这种方法！ function rmdir\_tree($dirname) //定义一个函数 rmdir\_tree，要删除的文件名为$dirname {  $handle = opendir($dirname);//打开名为$dirname 的文件夹  while ($file = readdir($handle)) //跑回圈读取打开的文件夹  {  if (($file == ".") || ($file == "..")) continue;//继续跑的条件，如果不满足就不跑了  $fullname = $dirname . "/" . $file;//完整的文件名（包括路径）  if (filetype($fullname) == "dir")  rmdir\_tree($fullname);//调用刚刚定义的函数，实现递归  else  unlink($fullname);//删除文件  }  closedir($handle);//关闭打开了的文件夹  rmdir($dirname);//删除自身（文件夹） }  // 综合应用实例 3：//计算目录的大小 function directory\_size($directory) {  $directorySize = 0;  if ($dh = @opendir($directory)) {  while (($filename = readdir($dh))) {  if ($filename != "." && $filename != "..") {  if (is\_file($directory . "/" . $filename)) {  $directorySize += filesize($directory . "/" . $filename);  }  if (is\_dir($directory . "/" . $filename)) {  $directorySize += directory\_size($directory . "/" . $filename);  }  }  }  }  @closedir($dh);  return $directorySize; } $directory = "./"; $totalSize = round((directory\_size($directory) / 1024), 2); echo "Directory $directory: " . $totalSize . "kb.";  // 综合应用实例 4： function batch\_rename\_file($base\_dir, $postfix, $new\_file\_name, $num) {  $handle = opendir($base\_dir);  $i = $num;  $new\_name = $new\_file\_name . $i . $postfix;  while ($filelist = readdir($handle)) {  if ($filelist != '.' && $$filelist != '..') {  rename($base\_dir . $filelist, $base\_dir . $new\_name);  $i++;  $new\_name = $new\_file\_name . $i . $postfix;  }  }  closedir($handle); }  batch\_rename\_file("spring/", ".jpg", "tangjiali", 0); |

## 正则表达式函数(important)

|  |
| --- |
| ★★★★★ereg // 正则表达式匹配 ★★★★★eregi // 不区分大小写的正则表达式匹配 ★★★★★ereg\_replace // 替换正则表达式 ★★★★★eregi\_replace // 不区分大小写替换正则表达式 ★★★★★split // 用正则表达式将字符串分割到数组中 ★★★★★spliti // 用正则表达式不区分大小写将字符串分割到数组中 sql\_regcase // 产生用于不区分大小的匹配的正则表达式 |

## SESSION函数(important)

|  |
| --- |
| session\_decode // Session 资料解码。 ★★★★★session\_destroy // 结束 session。 session\_encode // Session 资料编码。 session\_id // 存取目前 session 代号。 ★★★★★session\_is\_registered // 检查变量是否注册。 session\_module\_name // 存取目前 session 模块。 session\_name // 存取目前 session 名称。 ★★★★★session\_register // 注册新的变量。 session\_save\_path // 存取目前 session 路径。 ★★★★★session\_start // 初始 session。 ★★★★★session\_unregister // 删除已注册变量。 |

## MYSQL数据库函数(important)

|  |
| --- |
| mysql\_affected\_rows // 取得前一次 MySQL 操作所影响的记录行数 mysql\_change\_user // 改变活动连接中登录的用户 mysql\_client\_encoding // 返回字符集的名称 ★★★★★mysql\_close // 关闭 MySQL 连接 mysql\_connect // 打开一个到 MySQL 服务器的连接 ★★★★★mysql\_create\_db // 新建一个 MySQL 数据库 mysql\_data\_seek // 移动内部结果的指针 mysql\_db\_name // 取得结果数据 mysql\_db\_query // 发送一条 MySQL 查询 ★★★★★mysql\_drop\_db // 丢弃（删除）一个 MySQL 数据库 mysql\_errno // 返回上一个 MySQL 操作中的错误信息的数字编码 ★★★mysql\_error // 返回上一个 MySQL 操作产生的文本错误信息 mysql\_escape\_string // 转义一个字符串用于 mysql\_query ★★★★★mysql\_fetch\_array // 从结果集中取得一行作为关联数组，或数字数组，或二者兼有 mysql\_fetch\_assoc // 从结果集中取得一行作为关联数组 ★★★★★mysql\_fetch\_field // 从结果集中取得列信息并作为对象返回 mysql\_fetch\_lengths // 取得结果集中每个输出的长度 ★★★★★mysql\_fetch\_object // 从结果集中取得一行作为对象 mysql\_fetch\_row // 从结果集中取得一行作为枚举数组 mysql\_field\_flags // 从结果中取得和指定字段关联的标志 ★★★mysql\_field\_len // 返回指定字段的长度 ★★★mysql\_field\_name // 取得结果中指定字段的字段名 mysql\_field\_seek // 将结果集中的指针设定为制定的字段偏移量 ★★★mysql\_field\_table // 取得指定字段所在的表名 ★★★mysql\_field\_type // 取得结果集中指定字段的类型 ★★★★★mysql\_free\_result // 释放结果内存 mysql\_get\_client\_info // 取得 MySQL 客户端信息 mysql\_get\_host\_info // 取得 MySQL 主机信息 mysql\_get\_proto\_info // 取得 MySQL 协议信息 mysql\_get\_server\_info // 取得 MySQL 服务器信息 mysql\_info // 取得最近一条查询的信息 ★★★★★mysql\_insert\_id // 取得上一步 INSERT 操作产生的 ID ★★★★★mysql\_list\_dbs // 列出 MySQL 服务器中所有的数据库 ★★★mysql\_list\_fields // 列出 MySQL 结果中的字段 mysql\_list\_processes // 列出 MySQL 进程 ★★★★★mysql\_list\_tables // 列出 MySQL 数据库中的表 ★★★mysql\_num\_fields // 取得结果集中字段的数目 ★★★★★mysql\_num\_rows // 取得结果集中行的数目 mysql\_pconnect // 打开一个到 MySQL 服务器的持久连接 mysql\_ping // Ping 一个服务器连接，如果没有连接则重新连接 ★★★★★mysql\_query // 发送一条 MySQL 查询 mysql\_real\_escape\_string // 转义 SQL 语句中使用的字符串中的特殊字符，并考虑到连接的当前字符集 mysql\_result // 取得结果数据 ★★★★★mysql\_select\_db // 选择 MySQL 数据库 mysql\_stat // 取得当前系统状态 mysql\_tablename // 取得表名 mysql\_thread\_id // 返回当前线程的 ID mysql\_unbuffered\_query // 向 MySQL 发送一条 SQL 查询，并不获取和缓存结果的行 |

## PostgreSQL 数据库函数

|  |
| --- |
| pg\_affected\_rows // 返回受影响的记录数目 pg\_cancel\_query // 取消异步查询 pg\_client\_encoding // 取得客户端编码方式 pg\_close // 关闭一个 PostgreSQL 连接 pg\_connect // 打开一个 PostgreSQL 连接 pg\_connection\_busy // 获知连接是否为忙 pg\_connection\_reset // 重置连接（再次连接） pg\_connection\_status // 获得连接状态 pg\_convert // 将关联的数组值转换为适合 SQL 语句的格式。 pg\_copy\_from // 根据数组将记录插入表中 pg\_copy\_to // 将一个表拷贝到数组中 pg\_dbname // 获得数据库名 pg\_delete // 删除记录 pg\_end\_copy // 与 PostgreSQL 后端同步 pg\_escape\_bytea // 转义 bytea 类型的二进制数据 pg\_escape\_string // 转义 text/char 类型的字符串 pg\_execute // Sends a request to execute a prepared statement with given parameters, and waits for the result. pg\_fetch\_all\_columns // Fetches all rows in a particular result column as an array pg\_fetch\_all // 从结果中提取所有行作为一个数组 pg\_fetch\_array // 提取一行作为数组 pg\_fetch\_assoc // 提取一行作为关联数组 pg\_fetch\_object // 提取一行作为对象 pg\_fetch\_result // 从结果资源中返回值 pg\_fetch\_row // 提取一行作为枚举数组 pg\_field\_is\_null // 测试字段是否为 NULL pg\_field\_name // 返回字段的名字 pg\_field\_num // 返回字段的编号 pg\_field\_prtlen // 返回打印出来的长度 pg\_field\_size // 返回指定字段占用内部存储空间的大小 pg\_field\_type\_oid // Returns the type ID (OID) for the corresponding field number pg\_field\_type // 返回相应字段的类型名称 pg\_free\_result // 释放查询结果占用的内存 pg\_get\_notify // Ping 数据库连接 pg\_get\_pid // Ping 数据库连接 pg\_get\_result // 取得异步查询结果 pg\_host // 返回和某连接关联的主机名 pg\_insert // 将数组插入到表中 pg\_last\_error // 得到某连接的最后一条错误信息 pg\_last\_notice // 返回 PostgreSQL 服务器最新一条公告信息 pg\_last\_oid // 返回上一个对象的 oid pg\_lo\_close // 关闭一个大型对象 pg\_lo\_create // 新建一个大型对象 pg\_lo\_export // 将大型对象导出到文件 pg\_lo\_import // 将文件导入为大型对象 pg\_lo\_open // 打开一个大型对象 pg\_lo\_read\_all // 读入整个大型对象并直接发送给浏览器 pg\_lo\_read // 从大型对象中读入数据 pg\_lo\_seek // 移动大型对象中的指针 pg\_lo\_tell // 返回大型对象的当前指针位置 pg\_lo\_unlink // 删除一个大型对象 pg\_lo\_write // 向大型对象写入数据 pg\_meta\_data // 获得表的元数据 pg\_num\_fields // 返回字段的数目 pg\_num\_rows // 返回行的数目 pg\_options // 获得和连接有关的选项 pg\_parameter\_status // Looks up a current parameter setting of the server. pg\_pconnect // 打开一个持久的 PostgreSQL 连接 pg\_ping // Ping 数据库连接 pg\_port // 返回该连接的端口号 pg\_prepare // Submits a request to create a prepared statement with the given parameters, and waits for completion. pg\_put\_line // 向 PostgreSQL 后端发送以 NULL 结尾的字符串 pg\_query\_params // Submits a command to the server and waits for the result, with the ability to pass parameters separately from the SQL command text. pg\_query // 执行查询 pg\_result\_error\_field // Returns an individual field of an error report. pg\_result\_error // 获得查询结果的错误信息 pg\_result\_seek // 在结果资源中设定内部行偏移量 pg\_result\_status // 获得查询结果的状态 pg\_select // 选择记录 pg\_send\_execute // Sends a request to execute a prepared statement with given parameters, without waiting for the result(s). pg\_send\_prepare // Sends a request to create a prepared statement with the given parameters, without waiting for completion. pg\_send\_query\_params // Submits a command and separate parameters to the server without waiting for the result(s). pg\_send\_query // 发送异步查询 pg\_set\_client\_encoding // 设定客户端编码 pg\_set\_error\_verbosity // Determines the verbosity of messages returned by pg\_last\_error() and pg\_result\_error(). pg\_trace // 启动一个 PostgreSQL 连接的追踪功能 pg\_transaction\_status // Returns the current in-transaction status of the server. pg\_tty // 返回该连接的 tty 号 pg\_unescape\_bytea // 取消 bytea 类型中的字符串转义 pg\_untrace // 关闭 PostgreSQL 连接的追踪功能 pg\_update // 更新表 pg\_version // Returns an array with client, protocol and server version (when available) |

## Oracle 数据库函数

|  |
| --- |
| OCI-Collection->append // 向 collection 增加单元 OCI-Collection->assign // 从现有的另一个 collection 向 collection赋值 OCI-Collection->assignElem // 给 collection 中的单元赋值 OCI-Collection->free // 释放关联于 collection 的对象的资源 OCI-Collection->getElem // 返回单元的值 OCI-Collection->max // 返回 collection 中单元的最大数目 OCI-Collection->size // 返回 collection 中的单元数目 OCI-Collection->trim // 从 collection 尾端开始删除单元 OCI-Lob->append // Appends data from the large object to another large object OCI-Lob->close // 关闭 LOB 描述符 OCI-Lob->eof // Tests for end-of-file on a large object's descriptor OCI-Lob->erase // Erases a specified portion of the internal LOB data OCI-Lob->export // 将 LOB 的内容导出到文件中 OCI-Lob->flush // Flushes/writes buffer of the LOB to the server OCI-Lob->free // 释放与 LOB 描述符所关联的资源 OCI-Lob->getBuffering // Returns current state of buffering for the large object OCI-Lob->import // 将数据从文件导入 LOB OCI-Lob->load // 返回大对象的内容 OCI-Lob->read // Reads part of the large object OCI-Lob->rewind // Moves the internal pointer to the beginning of the large object OCI-Lob->save // 将数据保存到大对象中 OCI-Lob->seek // Sets the internal pointer of the large object OCI-Lob->setBuffering // Changes current state of buffering for the large object OCI-Lob->size // Returns size of large object OCI-Lob->tell // Returns current position of internal pointer of large object OCI-Lob->truncate // Truncates large object OCI-Lob->write // Writes data to the large object OCI-Lob->writeTemporary // 写入一个临时的大对象 oci\_bind\_by\_name // 绑定一个 PHP 变量到一个 Oracle 位置标志符 oci\_cancel // 取消从游标读取数据 oci\_close // 关闭 Oracle 连接 oci\_commit // 提交未执行的事务处理 oci\_connect // 建立一个到 Oracle 服务器的连接 oci\_define\_by\_name // 在 SELECT 中使用 PHP 变量作为定义的步骤 oci\_error // 返回上一个错误 oci\_execute // 执行一条语句 oci\_fetch\_all // 获取结果数据的所有行到一个数组 oci\_fetch\_array // Returns the next row from the result data as an associative or numeric array, or both oci\_fetch\_assoc // Returns the next row from the result data as an associative array oci\_fetch\_object // Returns the next row from the result data as an object oci\_fetch\_row // Returns the next row from the result data as a numeric array oci\_fetch // Fetches the next row into result-buffer oci\_field\_is\_null // 检查字段是否为 NULL oci\_field\_name // 返回字段名 oci\_field\_precision // 返回字段精度 oci\_field\_scale // 返回字段范围 oci\_field\_size // 返回字段大小 oci\_field\_type\_raw // 返回字段的原始 Oracle 数据类型 oci\_field\_type // 返回字段的数据类型 oci\_free\_statement // 释放关联于语句或游标的所有资源 oci\_internal\_debug // 打开或关闭内部调试输出 oci\_lob\_copy // Copies large object oci\_lob\_is\_equal // Compares two LOB/FILE locators for equality oci\_new\_collection // 分配新的 collection 对象 oci\_new\_connect // 建定一个到 Oracle 服务器的新连接 oci\_new\_cursor // 分配并返回一个新的游标（语句句柄） oci\_new\_descriptor // 初始化一个新的空 LOB 或 FILE 描述符 oci\_num\_fields // 返回结果列的数目 oci\_num\_rows // 返回语句执行后受影响的行数 oci\_parse // 配置 Oracle 语句预备执行 oci\_password\_change // 修改 Oracle 用户的密码 oci\_pconnect // 使用一个持久连接连到 Oracle 数据库 oci\_result // 返回所取得行中字段的值 oci\_rollback // 回滚未提交的事务 oci\_server\_version // 返回服务器版本信息 oci\_set\_prefetch // 设置预提取行数 oci\_statement\_type // 返回 OCI 语句的类型 ocibindbyname // oci\_bind\_by\_name() 的别名 ocicancel // oci\_cancel() 的别名 ocicloselob // OCI-Lob->close 的别名 ocicollappend // OCI-Collection->append 的别名 ocicollassign // OCI-Collection->assign 的别名 ocicollassignelem // OCI-Collection->assignElem 的别名 ocicollgetelem // OCI-Collection->getElem 的别名 ocicollmax // OCI-Collection->max 的别名 ocicollsize // OCI-Collection->size 的别名 ocicolltrim // OCI-Collection->trim 的别名 ocicolumnisnull // oci\_field\_is\_null() 的别名 ocicolumnname // oci\_field\_name() 的别名 ocicolumnprecision // oci\_field\_precision() 的别名 ocicolumnscale // oci\_field\_scale() 的别名 ocicolumnsize // oci\_field\_size() 的别名 ocicolumntype // oci\_field\_type() 的别名 ocicolumntyperaw // oci\_field\_type\_raw() 的别名 ocicommit // oci\_commit() 的别名 ocidefinebyname // oci\_define\_by\_name() 的别名 ocierror // oci\_error() 的别名 ociexecute // oci\_execute() 的别名 ocifetch // oci\_fetch() 的别名 ocifetchinto // 获取下一行到一个数组 ocifetchistatement // oci\_fetch\_all() 的别名 ocifreecollection // OCI-Collection->free 的别名 ocifreecursor // oci\_free\_statement() 的别名 ocifreedesc // OCI-Lob->free 的别名 ocifreestatement // oci\_free\_statement() 的别名 ociinternaldebug // oci\_internal\_debug() 的别名 ociloadlob // OCI-Lob->load 的别名 ocilogoff // oci\_close() 的别名 ocilogon // oci\_connect() 的别名 ocinewcollection // oci\_new\_collection() 的别名 ocinewcursor // oci\_new\_cursor() 的别名 ocinewscriptor // oci\_new\_descriptor() 的别名 ocinlogon // oci\_new\_connect() 的别名 ocinumcols // oci\_num\_fields() 的别名 ociparse // oci\_parse() 的别名 ociplogon // oci\_pconnect() 的别名 ociresult // oci\_result() 的别名 ocirollback // oci\_rollback() 别名 ocirowcount // oci\_num\_rows() 的别名 ocisavelob // OCI-Lob->save 的别名 ocisavelobfile // OCI-Lob->import 的别名 ociserverversion // oci\_server\_version() 的别名 ocisetprefetch // oci\_set\_prefetch() 的别名 ocistatementtype // oci\_statement\_type() 的别名 ociwritelobtofile // OCI-Lob->export 的别名 ociwritetemporarylob // OCI-Lob->writeTemporary 的别名 |

## MATH 函数

|  |
| --- |
| abs // 绝对值 acos // 反余弦 acosh // 反双曲余弦 asin // 反正弦 asinh // 反双曲正弦 atan2 // 两个参数的反正切 atan // 反正切 atanh // 反双曲正切 base\_convert // 在任意进制之间转换数字 ★★★★bindec // 二进制转换为十进制 ★★★★★ceil // 进一法取整 cos // 余弦 cosh // 双曲余弦 ★★★★decbin // 十进制转换为二进制 ★★★★dechex // 十进制转换为十六进制 ★★★★decoct // 十进制转换为八进制 ★★★★★deg2rad // 将角度转换为弧度 exp // 计算 e（自然对数的底）的指数 expm1 // 返回 exp(number) - 1，甚至当 number 的值接近零也能计算出准确结果 ★★★★★floor // 舍去法取整 fmod // 返回除法的浮点数余数 getrandmax // 显示随机数最大的可能值 ★★★★hexdec // 十六进制转换为十进制 hypot // 计算一直角三角形的斜边长度 is\_finite // 判断是否为有限值 is\_infinite // 判断是否为无限值 is\_nan // 判断是否为合法数值 lcg\_value // 组合线性同余发生器 log10 // 以 10 为底的对数 log1p // 返回 log(1 + number)，甚至当 number 的值接近零也能计算出准确结果 log // 自然对数 ★★★★★max // 找出最大值 ★★★★★min // 找出最小值 mt\_getrandmax // 显示随机数的最大可能值 mt\_rand // 生成更好的随机数 mt\_srand // 播下一个更好的随机数发生器种子 ★★★★octdec // 八进制转换为十进制 pi // 得到圆周率值 pow // 指数表达式 rad2deg // 将弧度数转换为相应的角度数 ★★★★★rand // 产生一个随机整数 round // 对浮点数进行四舍五入 sin // 正弦 sinh // 双曲正弦 sqrt // 平方根 srand // 播下随机数发生器种子 tan // 正切 tanh // 双曲正切 |

## BC math 高精度数学函数

|  |
| --- |
| bcadd // Add two arbitrary precision numbers bccomp // Compare two arbitrary precision numbers bcdiv // Divide two arbitrary precision numbers bcmod // Get modulus of an arbitrary precision number bcmul // Multiply two arbitrary precision number bcpow // Raise an arbitrary precision number to another bcpowmod // Raise an arbitrary precision number to another, reduced by a specified modulus bcscale // Set default scale parameter for all bc math functions bcsqrt // Get the square root of an arbitrary precision number bcsub // Subtract one arbitrary precision number from another |

## 类和对象函数

|  |
| --- |
| call\_user\_method\_array // 调用一个用户方法，同时传递参数数组［已停用］ call\_user\_method // 调用特定对象的用户方法［已停用］ ★★★★★class\_exists // 检查类是否已定义 get\_class\_methods // 返回由类的方法名组成的数组 get\_class\_vars // 返回由类的默认属性组成的数组 get\_class // 返回对象的类名 get\_declared\_classes // 返回由已定义类的名字所组成的数组 get\_declared\_interfaces // Returns an array of all declared interfaces get\_object\_vars // 返回由对象属性组成的关联数组 get\_parent\_class // 返回对象或类的父类名 interface\_exists // Checks if the interface has been defined is\_a // 如果对象属于该类或该类是此对象的父类则返回 TRUE is\_subclass\_of // 如果此对象是该类的子类，则返回 TRUE ★★★★★method\_exists // 检查类的方法是否存在 property\_exists // Checks if the object or class has a property |

## Perl兼容正则表达式函数

|  |
| --- |
| // 模式修正符,解说正则表达式模式中使用的修正符 // 模式语法, 解说 Perl 兼容正则表达式的语法 preg\_grep // 返回与模式匹配的数组单元 preg\_match\_all // 进行全局正则表达式匹配 preg\_match // 进行正则表达式匹配 preg\_quote // 转义正则表达式字符 preg\_replace\_callback // 用回调函数执行正则表达式的搜索和替换 preg\_replace // 执行正则表达式的搜索和替换 preg\_split // 用正则表达式分割字符串 |

## URL函数

|  |
| --- |
| ★★★★★base64\_decode // 对使用 MIME base64 编码的数据进行解码 ★★★★★base64\_encode // 使用 MIME base64 对数据进行编码 get\_headers // Fetches all the headers sent by the server in response to a HTTP request get\_meta\_tags // 从一个文件中提取所有的 meta 标签 content 属性，返回一个数组 http\_build\_query // 生成 url-encoded 之后的请求字符串 ★★★★★parse\_url // 解析 URL，返回其组成部分 rawurldecode // 对已编码的 URL 字符串进行解码 rawurlencode // 按照 RFC 1738 对 URL 进行编码 ★★★★★urldecode // 解码已编码的 URL 字符串 ★★★★★urlencode // 编码 URL 字符串 |

## FTP函数

|  |
| --- |
| ftp\_alloc // Allocates space for a file to be uploaded ftp\_cdup // 切换到当前目录的父目录 ftp\_chdir // 在 FTP 服务器上切换当前目录 ftp\_chmod // Set permissions on a file via FTP ftp\_close // 关闭一个活动的 FTP 连接 ftp\_connect // 建立一个新的 FTP 连接 ftp\_delete // 删除 FTP 服务器上的一个文件 ftp\_exec // 请求运行一条 FTP 命令 ftp\_fget // 从 FTP 服务器上下载一个文件并保存到本地一个已经打开的文件中 ftp\_fput // 上传一个已经打开的文件到 FTP 服务器 ftp\_get\_option // 返回当前 FTP 连接的各种不同的选项设置 ftp\_get // 从 FTP 服务器上下载一个文件 ftp\_login // 登录 FTP 服务器 ftp\_mdtm // 返回指定文件的最后修改时间 ftp\_mkdir // 建立新目录 ftp\_nb\_continue // 连续获取／发送文件（non-blocking） ftp\_nb\_fget // Retrieves a file from the FTP server and writes it to an open file (non-blocking) ftp\_nb\_fput // Stores a file from an open file to the FTP server (non-blocking) ftp\_nb\_get // 从 FTP 服务器上获取文件并写入本地文件（non-blocking） ftp\_nb\_put // 存储一个文件至 FTP 服务器（non-blocking） ftp\_nlist // 返回给定目录的文件列表 ftp\_pasv // 返回当前 FTP 被动模式是否打开 ftp\_put // 上传文件到 FTP 服务器 ftp\_pwd // 返回当前目录名 ftp\_quit // ftp\_close() 的别名 ftp\_raw // Sends an arbitrary command to an FTP server ftp\_rawlist // 返回指定目录下文件的详细列表 ftp\_rename // 更改 FTP 服务器上指定的文件名 ftp\_rmdir // 删除一个目录 ftp\_set\_option // 设置各种 FTP 运行时选项 ftp\_site // 向服务器发送 SITE 命令 ftp\_size // 返回指定文件的大小 ftp\_ssl\_connect // Opens an Secure SSL-FTP connection ftp\_systype // 返回远程 FTP 服务器的系统类型 |

## HTTP 函数

|  |
| --- |
| ★★★★★header // 发送一个原始 HTTP 标头 headers\_list // 返回已发送（或准备发送）的响应标头的列表 headers\_sent // 检查标头是否已被发送以及在哪里被发送 ★★★★★setcookie // 发送一个 cookie 信息 setrawcookie // 发送一个其值未经 urlencode 编码的 cookie |

## IMAGE函数

|  |
| --- |
| ★★★★★gd\_info // 取得当前安装的 GD 库的信息 ★★★★★getimagesize // 取得图像大小 ★★★★★image\_type\_to\_extension // 取得图像类型的文件后缀 image\_type\_to\_mime\_type // 取得 getimagesize，exif\_read\_data， exif\_thumbnail，exif\_imagetype 所返回的图像类型的 MIME 类型 ★★★★★image2wbmp // 以 WBMP 格式将图像输出到浏览器或文件 imagealphablending // 设定图像的混色模式 imageantialias // 是否使用抗锯齿（antialias）功能 ★★★★imagearc // 画椭圆弧 ★★★★imagechar // 水平地画一个字符 ★★★★imagecharup // 垂直地画一个字符 imagecolorallocate // 为一幅图像分配颜色 imagecolorallocatealpha // 为一幅图像分配颜色 + alpha imagecolorat // 取得某像素的颜色索引值 imagecolorclosest // 取得与指定的颜色最接近的颜色的索引值 imagecolorclosestalpha // 取得与指定的颜色加透明度最接近的颜色 imagecolorclosesthwb // 取得与给定颜色最接近的色度的黑白色的索引 imagecolordeallocate // 取消图像颜色的分配 imagecolorexact // 取得指定颜色的索引值 imagecolorexactalpha // 取得指定的颜色加透明度的索引值 imagecolormatch // 使一个图像中调色板版本的颜色与真彩色版本更能匹配 imagecolorresolve // 取得指定颜色的索引值或有可能得到的最接近的替代值 imagecolorresolvealpha // 取得指定颜色 + alpha 的索引值或有可能得到的最接近的替代值 imagecolorset // 给指定调色板索引设定颜色 imagecolorsforindex // 取得某索引的颜色 imagecolorstotal // 取得一幅图像的调色板中颜色的数目 imagecolortransparent // 将某个颜色定义为透明色 imageconvolution // 用系数 div 和 offset 申请一个 3x3 的卷积矩阵 imagecopy // 拷贝图像的一部分 imagecopymerge // 拷贝并合并图像的一部分 imagecopymergegray // 用灰度拷贝并合并图像的一部分 imagecopyresampled // 重采样拷贝部分图像并调整大小 imagecopyresized // 拷贝部分图像并调整大小 imagecreate // 新建一个基于调色板的图像 imagecreatefromgd2 // 从 GD2 文件或 URL 新建一图像 imagecreatefromgd2part // 从给定的 GD2 文件或 URL 中的部分新建一图像 imagecreatefromgd // 从 GD 文件或 URL 新建一图像 imagecreatefromgif // 从 GIF 文件或 URL 新建一图像 imagecreatefromjpeg // 从 JPEG 文件或 URL 新建一图像 imagecreatefrompng // 从 PNG 文件或 URL 新建一图像 imagecreatefromstring // 从字符串中的图像流新建一图像 imagecreatefromwbmp // 从 WBMP 文件或 URL 新建一图像 imagecreatefromxbm // 从 XBM 文件或 URL 新建一图像 imagecreatefromxpm // 从 XPM 文件或 URL 新建一图像 imagecreatetruecolor // 新建一个真彩色图像 ★★★★imagedashedline // 画一虚线 ★★★★imagedestroy // 销毁一图像 ★★★★imageellipse // 画一个椭圆 ★★★★imagefill // 区域填充 ★★★★imagefilledarc // 画一椭圆弧且填充 ★★★★imagefilledellipse // 画一椭圆并填充 ★★★★imagefilledpolygon // 画一多边形并填充 ★★★★imagefilledrectangle // 画一矩形并填充 imagefilltoborder // 区域填充到指定颜色的边界为止 imagefilter // 对图像使用过滤器 ★★★★imagefontheight // 取得字体高度 ★★★★imagefontwidth // 取得字体宽度 imageftbbox // 给出一个使用 FreeType 2 字体的文本框 ★★★★imagefttext // 使用 FreeType 2 字体将文本写入图像 imagegammacorrect // 对 GD 图像应用 gamma 修正 ★★★★★imagegd2 // 将 GD2 图像输出到浏览器或文件 ★★★★★imagegd // 将 GD 图像输出到浏览器或文件 imagegif // 以 GIF 格式将图像输出到浏览器或文件 imageinterlace // 激活或禁止隔行扫描 imageistruecolor // 检查图像是否为真彩色图像 ★★★★★imagejpeg // 以 JPEG 格式将图像输出到浏览器或文件 imagelayereffect // 设定 alpha 混色标志以使用绑定的 libgd 分层效果 ★★★★imageline // 画一条线段 ★★★★imageloadfont // 载入一新字体 imagepalettecopy // 将调色板从一幅图像拷贝到另一幅 ★★★★★imagepng // 以 PNG 格式将图像输出到浏览器或文件 ★★★★imagepolygon // 画一个多边形 imagepsbbox // 给出一个使用 PostScript Type1 字体的文本方框 imagepscopyfont // 拷贝一个已加载的字体以备更改 imagepsencodefont // 改变字体中的字符编码矢量 imagepsextendfont // 扩充或精简字体 imagepsfreefont // 释放一个 PostScript Type 1 字体所占用的内存 imagepsloadfont // 从文件中加载一个 PostScript Type 1 字体 imagepsslantfont // 倾斜某字体 imagepstext // 用 PostScript Type1 字体把文本字符串画在图像上 ★★★★imagerectangle // 画一个矩形 imagerotate // 用给定角度旋转图像 imagesavealpha // 设置标记以在保存 PNG 图像时保存完整的 alpha 通道信息（与单一透明色相反） imagesetbrush // 设定画线用的画笔图像 imagesetpixel // 画一个单一像素 ★★★★imagesetstyle // 设定画线的风格 imagesetthickness // 设定画线的宽度 imagesettile // 设定用于填充的贴图 ★★★★imagestring // 水平地画一行字符串 ★★★★imagestringup // 垂直地画一行字符串 ★★★★★imagesx // 取得图像宽度 ★★★★★imagesy // 取得图像高度 imagetruecolortopalette // 将真彩色图像转换为调色板图像 imagettfbbox // 取得使用 TrueType 字体的文本的范围 ★★★★imagettftext // 用 TrueType 字体向图像写入文本 imagetypes // 返回当前 PHP 版本所支持的图像类型 ★★★★★imagewbmp // 以 WBMP 格式将图像输出到浏览器或文件 imagexbm // 将 XBM 图像输出到浏览器或文件 iptcembed // 将二进制 IPTC 数据嵌入到一幅 JPEG 图像中 iptcparse // 将二进制 IPTC http://www.iptc.org/ 块解析为单个标记 ★★★★jpeg2wbmp // 将 JPEG 图像文件转换为 WBMP 图像文件 ★★★★png2wbmp // 将 PNG 图像文件转换为 WBMP 图像文件 |

## ZIP 函数

|  |
| --- |
| zip\_close // Close a ZIP file archive zip\_entry\_close // Close a directory entry zip\_entry\_compressedsize // Retrieve the compressed size of a directory entry zip\_entry\_compressionmethod // Retrieve the compression method of a directory entry zip\_entry\_filesize // Retrieve the actual file size of a directory entry zip\_entry\_name // Retrieve the name of a directory entry zip\_entry\_open // Open a directory entry for reading zip\_entry\_read // Read from an open directory entry zip\_open // Open a ZIP file archive zip\_read // Read next entry in a ZIP file archive |

## Bzip2压缩函数

|  |
| --- |
| bzclose // Close a bzip2 file bzcompress // Compress a string into bzip2 encoded data bzdecompress // Decompresses bzip2 encoded data bzerrno // Returns a bzip2 error number bzerror // Returns the bzip2 error number and error string in an array bzerrstr // Returns a bzip2 error string bzflush // Force a write of all buffered data bzopen // Opens a bzip2 compressed file bzread // Binary safe bzip2 file read bzwrite // Binary safe bzip2 file write |

## Gzip函数

|  |
| --- |
| gzfile //Read entire gz-file into an array gzclose // Close an open gz-file pointer gzcompress // Compress a string gzdeflate // Deflate a string gzencode // Create a gzip compressed string gzeof // Test for end-of-file on a gz-file pointer gzgetc // Get character from gz-file pointer gzgets // Get line from file pointer gzgetss // Get line from gz-file pointer and strip HTML tags gzinflate // Inflate a deflated string gzopen // Open gz-file gzpassthru // Output all remaining data on a gz-file pointer gzputs // 别名 gzwrite() gzwrite() gzwrite() gzwrite() gzread // Binary-safe gz-file read gzrewind // Rewind the position of a gz-file pointer gzseek // Seek on a gz-file pointer gztell // Tell gz-file pointer read/write position gzuncompress // Uncompress a compressed string gzwrite // Binary-safe gz-file write |

## DOM函数

|  |
| --- |
| DOMAttr->\_\_construct() // Creates a new DOMAttr object DOMAttr->isId() // Checks if attribute is a defined ID DOMCharacterData->appendData() // Append the string to the end of the character data of the node DOMCharacterData->deleteData() // Remove a range of characters from the node DOMCharacterData->insertData() // Insert a string at the specified 16-bit unit offset DOMCharacterData->replaceData() // Replace a substring within the DOMCharacterData node DOMCharacterData->substringData() // Extracts a range of data from the node DOMComment->\_\_construct() // Creates a new DOMComment object DOMDocument->\_\_construct() // Creates a new DOMDocument object DOMDocument->createAttribute() // Create new attribute DOMDocument->createAttributeNS() // Create new attribute node with an associated namespace DOMDocument->createCDATASection() // Create new cdata node DOMDocument->createComment() // Create new comment node DOMDocument->createDocumentFragment() // Create new document fragment DOMDocument->createElement() // Create new element node DOMDocument->createElementNS() // Create new element node with an associated namespace DOMDocument->createEntityReference() // Create new entity reference node DOMDocument->createProcessingInstruction() // Creates new PI node DOMDocument->createTextNode() // Create new text node DOMDocument->getElementById() // Searches for an element with a certain id DOMDocument->getElementsByTagName() // Searches for all elements with given tag name DOMDocument->getElementsByTagNameNS() // Searches for all elements with given tag name in specified namespace DOMDocument->importNode() // Import node into current document DOMDocument->load() // Load XML from a file DOMDocument->loadHTML() // Load HTML from a string DOMDocument->loadHTMLFile() // Load HTML from a file DOMDocument->loadXML() // Load XML from a string DOMDocument->normalize() // Normalizes the document DOMDocument->relaxNGValidate() // Performs relaxNG validation on the document DOMDocument->relaxNGValidateSource() // Performs relaxNG validation on the document DOMDocument->save() // Dumps the internal XML tree back into a file DOMDocument->saveHTML() // Dumps the internal document into a string using HTML formatting DOMDocument->saveHTMLFile() // Dumps the internal document into a file using HTML formatting DOMDocument->saveXML() // Dumps the internal XML tree back into a string DOMDocument->schemaValidate() // Validates a document based on a schema DOMDocument->schemaValidateSource() // Validates a document based on a schema DOMDocument->validate() // Validates the document based on its DTD DOMDocument->xinclude() // Substitutes XIncludes in a DOMDocument Object DOMElement->\_\_construct() // Creates a new DOMElement object DOMElement->getAttribute() // Returns value of attribute DOMElement->getAttributeNode() // Returns attribute node DOMElement->getAttributeNodeNS() // Returns attribute node DOMElement->getAttributeNS() // Returns value of attribute DOMElement->getElementsByTagName() // Gets elements by tagname DOMElement->getElementsByTagNameNS() // Get elements by namespaceURI and localName DOMElement->hasAttribute() // Checks to see if attribute exists DOMElement->hasAttributeNS() // Checks to see if attribute exists DOMElement->removeAttribute() // Removes attribute DOMElement->removeAttributeNode() // Removes attribute DOMElement->removeAttributeNS() // Removes attribute DOMElement->setAttribute() // Adds new attribute DOMElement->setAttributeNode() // Adds new attribute node to element DOMElement->setAttributeNodeNS() // Adds new attribute node to element DOMElement->setAttributeNS() // Adds new attribute DOMAttr->\_\_construct() // Creates a new DOMEntityReference object DOMImplementation->\_\_construct() // Creates a new DOMImplementation object DOMImplementation->createDocument() // Creates a DOMDocument object of the specified type with its document element DOMImplementation->createDocumentType() // Creates an empty DOMDocumentType object DOMImplementation->hasFeature() // Test if the DOM implementation implements a specific feature DOMNamedNodeMap->getNamedItem() // Retrieves a node specified by name DOMNamedNodeMap->getNamedItemNS() // Retrieves a node specified by local name and namespace URI DOMNamedNodeMap->item() // Retrieves a node specified by index DOMNode->appendChild() // Adds new child at the end of the children DOMNode->cloneNode() // Clones a node DOMNode->hasAttributes() // Checks if node has attributes DOMNode->hasChildNodes() // Checks if node has children DOMNode->insertBefore() // Adds a new child before a reference node DOMNode->isSameNode() // Indicates if two nodes are the same node DOMNode->isSupported() // Checks if feature is supported for specified version DOMNode->lookupNamespaceURI() // Gets the namespace URI of the node based on the prefix DOMNode->lookupPrefix() // Gets the namespace prefix of the node based on the namespace URI DOMNode->normalize() // Normalizes the node DOMNode->removeChild() // Removes child from list of children DOMNode->replaceChild() // Replaces a child DOMNodelist->item() // Retrieves a node specified by index DOMProcessingInstruction->\_\_construct() // Creates a new DOMProcessingInstruction object DOMText->\_\_construct() // Creates a new DOMText object DOMText->isWhitespaceInElementContent() // Indicates whether this text node contains whitespace DOMText->splitText() // Breaks this node into two nodes at the specified offset DOMXPath->\_\_construct() // Creates a new DOMXPath object DOMXPath->evaluate() // Evaluates the given XPath expression and returns a typed result if possible. DOMXPath->query() // Evaluates the given XPath expression DOMXPath->registerNamespace() // Registers the namespace with the DOMXPath object dom\_import\_simplexml // Gets a DOMElement object from a SimpleXMLElement object |

## win32API函数

|  |
| --- |
| w32api\_deftype // 为 w32api\_functions 函数定一个类型 w32api\_init\_dtype // 创建了一个数据类型的实例，并且将函数传入的值填入其中 w32api\_invoke\_function // 带有一个参数的执行一个函数，参数传递在函数名的后面 w32api\_register\_function // 从函数库中使用 PHP 注册一个函数function\_name w32api\_set\_call\_method // 设置调用的方法 |

## COM和.NET函数

|  |
| --- |
| COM // COM 类 DOTNET // DOTNET class VARIANT // VARIANT 类 com\_addref // 增加组件引用计数。[被废弃] com\_create\_guid // Generate a globally unique identifier (GUID) com\_event\_sink // Connect events from a COM object to a PHP object com\_get\_active\_object // Returns a handle to an already running instance of a COM object com\_get // 获取 COM 组件的属性值 [被废弃] com\_invoke // 调用 COM 组件的方法。 com\_isenum // 获取一个 IEnumVariant com\_load\_typelib // 装载一个 Typelib com\_load // 创建新的 COM 组件的引用 com\_message\_pump // Process COM messages, sleeping for up to timeoutms milliseconds com\_print\_typeinfo // Print out a PHP class definition for a dispatchable interface com\_propget // com\_get() 的别名 com\_propput // com\_set() 的别名 com\_propset // com\_set() 的别名 com\_release // 减少组件引用计数。[被废弃] com\_set // 给 COM 组件的属性赋值 variant\_abs // Returns the absolute value of a variant variant\_add // "Adds" two variant values together and returns the result variant\_and // performs a bitwise AND operation between two variants and returns the result variant\_cast // Convert a variant into a new variant object of another type variant\_cat // concatenates two variant values together and returns the result variant\_cmp // Compares two variants variant\_date\_from\_timestamp // Returns a variant date representation of a unix timestamp variant\_date\_to\_timestamp // Converts a variant date/time value to unix timestamp variant\_div // Returns the result from dividing two variants variant\_eqv // Performs a bitwise equivalence on two variants variant\_fix // Returns the integer portion ? of a variant variant\_get\_type // Returns the type of a variant object variant\_idiv // Converts variants to integers and then returns the result from dividing them variant\_imp // Performs a bitwise implication on two variants variant\_int // Returns the integer portion of a variant variant\_mod // Divides two variants and returns only the remainder variant\_mul // multiplies the values of the two variants and returns the result variant\_neg // Performs logical negation on a variant variant\_not // Performs bitwise not negation on a variant variant\_or // Performs a logical disjunction on two variants variant\_pow // Returns the result of performing the power function with two variants variant\_round // Rounds a variant to the specified number of decimal places variant\_set\_type // Convert a variant into another type "in-place" variant\_set // Assigns a new value for a variant object variant\_sub // subtracts the value of the right variant from the left variant value and returns the result variant\_xor // Performs a logical exclusion on two variants dotnet\_load // 加载一个 DOTNET 模块 |

## Apache 特有函数

|  |
| --- |
| apache\_child\_terminate // 在本次请求结束后终止 apache 进程 apache\_get\_modules // Get a list of loaded Apache modules apache\_get\_version // Fetch Apache version apache\_getenv // Get an Apache subprocess\_env variable apache\_lookup\_uri // 对指定的 URI 执行部分请求并返回所有有关信息 apache\_note // 取得或设置 apache 请求记录 apache\_request\_headers // Fetch all HTTP request headers apache\_reset\_timeout // Reset the Apache write timer apache\_response\_headers // Fetch all HTTP response headers apache\_setenv // Set an Apache subprocess\_env variable ascii2ebcdic // Translate string from ASCII to EBCDIC ebcdic2ascii // Translate string from EBCDIC to ASCII getallheaders // Fetch all HTTP request headers virtual // Perform an Apache sub-request |

## PDF 操作函数

|  |
| --- |
| PDF\_add\_annotation // 加入注释。 PDF\_add\_outline // 目前页面加入书签。 PDF\_arc // 绘弧。 PDF\_begin\_page // 启始 PDF 文件页面。 PDF\_circle // 绘圆。 PDF\_clip // 组合所有向量。 PDF\_close // 关闭 PDF 文件。 PDF\_closepath // 形成封闭的向量形状。 PDF\_closepath\_fill\_stroke // 形成封闭的向量形状沿向量绘线并填满。 PDF\_closepath\_stroke // 形成封闭的向量形状并沿向量绘线。 PDF\_close\_image // 关闭图文件。 PDF\_continue\_text // 输出文字。 PDF\_curveto // 绘贝氏曲线。 PDF\_endpath // 关闭目前向量。 PDF\_end\_page // 关闭 PDF 文件页面。 PDF\_execute\_image // 放置 PDF 文件中图片到指定位置。 [Page] PDF\_fill // 填满目前的向量。 PDF\_fill\_stroke // 填满目前的向量并沿向量绘线。 PDF\_get\_info // 返回文件信息。 PDF\_lineto // 绘直线。 PDF\_moveto // 配置处理的坐标点。 PDF\_open // 建立新的 PDF 文件。 PDF\_open\_gif // 打开 GIF 图文件。 PDF\_open\_jpeg // 打开 JPEG 图文件。 PDF\_open\_memory\_image // 打开内存图文件。 PDF\_place\_image // 放置图片到 PDF 文件指定位置。 PDF\_put\_image // 放置图片到 PDF 文件。 PDF\_rect // 绘长方形。 PDF\_restore // 还原环境变量。 PDF\_rotate // 旋转类。 PDF\_save // 储存环境变量。 PDF\_scale // 缩放类。 PDF\_setdash // 配置虚线样式。 PDF\_setflat // 配置平滑值。 PDF\_setgray // 指定绘图的颜色为灰阶并填入。 PDF\_setgray\_fill // 指定填入的颜色为灰阶。 PDF\_setgray\_stroke // 指定绘图的颜色为灰阶。 PDF\_setlinecap // 配置 linecap 参数。 PDF\_setlinejoin // 配置连接参数。 PDF\_setlinewidth // 配置线宽。 PDF\_setmiterlimit // 配置斜边界限。 PDF\_setrgbcolor // 指定绘图的颜色为彩色并填入。 PDF\_setrgbcolor\_fill // 指定填入的颜色为彩色。 PDF\_setrgbcolor\_stroke // 指定绘图的颜色为彩色。 PDF\_set\_char\_spacing // 配置字符间距。 PDF\_set\_duration // 配置二页的切换时间。 PDF\_set\_font // 配置使用的字型及大小。 PDF\_set\_horiz\_scaling // 配置文字水平间距。 PDF\_set\_info\_author // 配置文件作者。 PDF\_set\_info\_creator // 配置建文件者字符串。 PDF\_set\_info\_keywords // 配置文件的关键字。 PDF\_set\_info\_subject // 配置文件主题。 PDF\_set\_info\_title // 配置文件标题。 PDF\_set\_leading // 配置行距。 PDF\_set\_text\_matrix // 配置文字矩阵。 PDF\_set\_text\_pos // 配置文字位置。 PDF\_set\_text\_rendering // 配置文字表现方式。 PDF\_set\_text\_rise // 配置文字高度。 PDF\_set\_transition // 配置页的转换。 PDF\_set\_word\_spacing // 配置字间距。 PDF\_show // 输出字符串到 PDF 文件。 PDF\_show\_xy // 输出字符串到指定坐标。 PDF\_stringwidth // 计算字符串的宽度。 PDF\_stroke // 沿向量绘线。 PDF\_translate // 移动原点。 FDF\_close // 关闭 FDF 文件。 FDF\_create // 建立新的 FDF 文件。 FDF\_get\_file // 取得 /F 键的值。 FDF\_get\_status // 取得 /STATUS 键的值。 [Page] FDF\_get\_value // 取得字段的值。 FDF\_next\_field\_name // 下一字段的名称。 FDF\_open // 打开 FDF 文件。 FDF\_save // 将 FDF 文件存文件。 FDF\_set\_ap // 配置显示字段。 FDF\_set\_file // 配置 /F 键。 FDF\_set\_status // 配置 /STATUS 键。 FDF\_set\_value // 配置字段的值。 |

## XML语法解析函数

|  |
| --- |
| xml\_error\_string // 取得 XML 错误字符串。 xml\_get\_current\_byte\_index // 取得目前解析为第几个位组。 xml\_get\_current\_column\_number // 获知目前解析的第几字段。 xml\_get\_current\_line\_number // 取得目前解析的行号。 xml\_get\_error\_code // 取得 XML 错误码。 xml\_parse // 解析 XML 文件。 xml\_parser\_create // 初始 XML 解析器。 xml\_parser\_free // 释放解析占用的内存。 xml\_parser\_get\_option // 取得解析使用的选项。 xml\_parser\_set\_option // 配置解析使用的选项。 xml\_set\_character\_data\_handler // 建立字符资料标头。 xml\_set\_default\_handler // 建立默认标头。 xml\_set\_element\_handler // 配置元素的标头。 xml\_set\_external\_entity\_ref\_handler // 配置外部实体参引的标头。 xml\_set\_notation\_decl\_handler // 配置记法宣告的标头。 xml\_set\_object // 使 XML 解析器用类。 xml\_set\_processing\_instruction\_handler // 建立处理指令标头。 [Page] xml\_set\_unparsed\_entity\_decl\_handler // 配置未解析实体宣告的标头。 |

## Simple XML函数

|  |
| --- |
| SimpleXMLElement->asXML // Return a well-formed XML string based on SimpleXML element SimpleXMLElement->attributes // Identifies an element's attributes SimpleXMLElement->children // Finds children of given node SimpleXMLElement->xpath // Runs Xpath query on XML data simplexml\_import\_dom // Get a SimpleXMLElement object from a DOM node. simplexml\_load\_file // Interprets an XML file into an object simplexml\_load\_string // Interprets a string of XML into an object |

## ODBC 函数

|  |
| --- |
| //使用 odbc 函数可以连接到任何类型的数据库。 //需要配置数据库的 odbc 数据源(需要对应的数据库的 odbc 驱动) odbc\_autocommit // 开关自动改动功能。 odbc\_binmode // 配置二进位资料处理方式。 odbc\_close // 关闭 ODBC 链接。 odbc\_close\_all // 关闭所有 ODBC 链接。 odbc\_commit // 改动 ODBC 数据库。 odbc\_connect // 链接至 ODBC 数据库。 odbc\_cursor // 取得游标名。 odbc\_exec // 执行 SQL 指令。 == odbc\_do // 执行 SQL 指令。 odbc\_execute // 执行预置 SQL 指令。 odbc\_fetch\_into // 取得返回的指定列。 [Page] odbc\_fetch\_row // 取得返回一行。 odbc\_field\_len // 取得字段资料长度。 odbc\_field\_name // 取得字段名称。 odbc\_field\_type // 取得字段资料类型。 odbc\_free\_result // 释出返回资料的内存。 odbc\_longreadlen // 配置返回栏的最大值。 odbc\_num\_fields // 取得字段的数目。 odbc\_num\_rows // 取得返回行的数目。 odbc\_pconnect // 长期链接至 ODBC 数据库。 odbc\_prepare // 预置 SQL 指令。 odbc\_result // 取得返回资料。 odbc\_result\_all // 返回 HTML 表格资料。 odbc\_rollback // 撤消当前交易。 odbc\_setoption // 调整 ODBC 配置。 |

## 其他有用的函数 (important)

|  |
| --- |
| ★★★★★Chop // 去除连续空白。 ★★★★★connection\_timeout // 若超过 PHP 程序执行时间则返回 true。 ★★★★★define // 定义一个常量 ★★★★★dl // 载入 PHP 扩充模块 ★★★★★die // 输出信息并中断 PHP 程序。 ★★★★★exec // 执行外部程序。 ★★★★★exit // 结束 PHP 程序。 ★★★★★function\_exists // 检查函数是否已定义。 ★★★★★getenv // 取得系统的环境变量 ★★★★★gethostbyaddr // 根据 IP 地址返回机器名称。 ★★★★★gethostbyname // 根据域名返回 IP 网址。 ★★★★★mail // 寄出电子邮件。 ★★★★★set\_time\_limit // 设置该页最久执行时间。 ★★★★★sleep(int seconds) // 暂停执行(以秒为单位)。 ★★★★★ require\_once() require\_once() require\_once() require\_once() // 在脚本执行期间包含并运行指定文件。 ★★★★★require() require() require() require() // 语句包含并运行指定文件。 ★★★★★include() include() include() include() // 语句包含并运行指定文件。 ★★★★★include\_once() include\_once() include\_once() include\_once() // 语句在脚本执行期间包含并运行指定文件。 |