# [PHP基本语法](https://www.cnblogs.com/staven/p/5142509.html)

一、PHP入门

1、指令分隔符“分号”

        语义分为两种：一种是在程序中使用结构定义语句，例如流程控制、函数定义、类的定义等，用来定义程序结构使用的语句。在结构语句后面不能使用分号作为结束；

        另一种是在程序中使用功能执行语句，例如变量的声明、内容的输出、函数的调用等，是用来在程序中执行某些特定功能的语句，这种语句也可以成为指令，php需要在每个指令后用分号结束。

2、程序注释

        php注释的三种形式：

以“/\*”和“\*/”闭合多行注释符，

“//”和"#"开始的单行注释符

3、在程序中使用空白的处理

下列情况应该总是使用两个空行：

一个源文件的两个代码片段之间。

两个类的声明之间。

下列情况应该总是使用一个空行：

两个函数声明之间。

函数内的局部变量和函数的第一条语句之间。

块注释或单行注释之前。

一个函数内的两个逻辑代码段之间，用以提高可读性。

空格的应用规则：

一般用于关键字和括号之间，函数名与左括号间不应使用空格。

一般函数的参数列表中的逗号后面添加空格。

数学算式的操作数与运算符之间应该添加空格(二进制元算与医院元算除外)。

for语句中表达式应该用逗号分开，后面添加空格。

二、变量

        第一次赋值，就创建了变量。必须使用"$"后跟变量名表示。再次给该变量赋值会覆盖之前的变量。

        php变量声明以后有一定的范围，变量的范围即它定义的上下文背景(也就是它生效范围)。大部分的php变量如果不是在函数里面声明的，只有在声明处到文件结束的一个单独范围内使用。这个单独的范围跨度不仅是<?php  标记开始处到?>结束标记处使用，可以在一个页面的所有开启的PHP模式下使用，也包含了include和require引入的文件。如果使用COOKIE或SESSION还可以在多个页面中应用。

        在变量的使用范围周期内，我们可以借助unset()函数释放指定的变量，使用isset()函数检测变量是否设置和使用empty()函数检测一个变脸是否为空。

        empty()与isset()的区别：

            如果empty()函数的参数是非空或非零的值，则empty()返回false。""、0、“0”、null、false、array()、var $var、以及没有任何属性的对象都将被认为是空的。如果参数为空，则返回true。

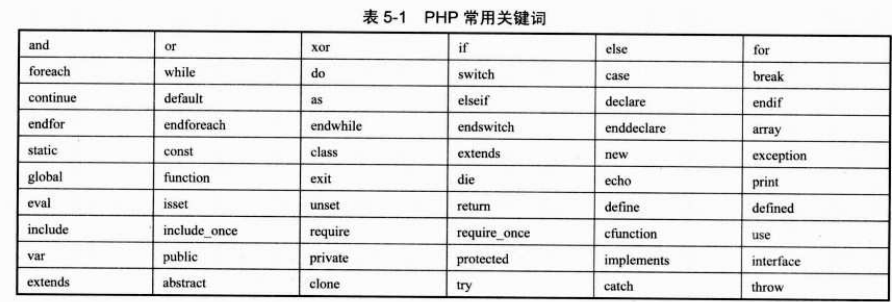
            如果函数isset()参数存在，则返回true。若使用isset()检测一个被设置成null的变量或使用unset()释放了一个变量，都将返回false。

            注意：字节NULL("\0")并不等同于null常数。

            建议：使用!empty($var)判断变量存在且不能为空。

1、变量的命名

        变量名区分大小写。但内置结构和关键字以及用户自定义的类名和函数名都是不区分大小写的。

[](http://images2015.cnblogs.com/blog/712855/201601/712855-20160119161020468-1536530589.png)

        驼峰命名法。如：$aaaBBBccc。

2、可变变量

        可变变量可使用一个普通变量的值作为这个可变变量的变量名。

3、变量的引用赋值

        变量总是传值赋值。也就是说，当将一个表达式的值赋予一个变量时，整个原始表达式的值被赋值到目标变量。改变其中一个变量的值，将不会影响到另一个变量。

        引用赋值，将“&”加到变量前。将变量的引用方式复制给新的变量，指向共同的引用，改变任何一个变量的值，都会影响另一个变量。只有有名字的变量才可以引用赋值。不会将引用赋值的两个变量在内存上同体，只是把各自的值相关联起来，基于这一点，unset()不会导致另一个变量消失。

4、变量的类型

4种标量类型：boolean(布尔型)、integer(整型)、float(浮点型)、string(字符串)

2种复合类型：array(数组)、object(对象)

2种特殊类型：resource(资源)、NULL

如果想看某个表达式的值和类型，可以使用函数var\_dump()。

布尔类型

以下值被认为是false

1）布尔值false。

2）整型值0(零)为假，-1和其他非零值(不论正负)一样，被认为是true。

3）浮点型值0.0（零）。

4）空白字符串和字符串“0”

5）没有成员变量的数组。

6）特殊类型NULL

整型

        有最大范围2 147 483 647 。PHP不支持无符号整数，如果给定的数超出最大范围，将会被解释为float。

浮点型

  永远不要比较两个浮点数是否相等。

  如果需要更高的精度，应该使用任意精度数学函数或gmp()函数。

字符串

单引号字符串：

        如果要在单引号中引用单引号，需要使用反斜杠(\)转义。

        如果在单引号之前或字符串结尾需要出现一个反斜线，需要用两个反斜线表示。

        其中出现的变量不会被变量的值替代。即php不会解析单引号中的变量，而是将变量名原样输出。定义简单字符串时，单引号字符串效率更高。

双引号字符串：其中的变量会被解析。

[](http://images2015.cnblogs.com/blog/712855/201601/712855-20160119161021218-422891999.png)

定界符字符串：

        使用定界符<<<开始，之后提供一个标识符，然后是包含的字符串，最后是同样顶格书写的标识符。

        定界符文本处理不能初始化类成员以外，表现的就和双引号字符串一样，只是没有双引号。这意味着在定界符文本中不需要转义引号，不过仍然可以用以上列出来的在双引号中可以使用的转义符号。而且定界符中的变量也会被解析，但当在定界符文本中表达复杂变量时和字符串一样同样也需要注意。所以能够很容易地使用定界符定义较长的字符串，通常用于从文本或者数据库中打断地输出文档。

数组(Array)

使用print\_r()函数查看数组中的全部内容。

对象(Object)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | <?php      class Person{          var $name;          function say(){              echo "Doing foo";          }      }      $p=new Person();      $p->name="staven";      $p->say();  ?> |

->：表示对象属性、方法的引用

资源类型(Resource)

NULL类型

        特殊的NULL值表示一个变量没有值，NULL类型唯一可能的值就是NULL。NULL不表示空格，也不表示零，也不是空字符串，而是表示一个变量的值为空。NULL不区分大小写，在下列情况下一个变量被认为是NULL。

a）将变量直接赋值为NULL。

b）声明的变量尚未被赋值。

c）被unset()函数销毁的变量。

伪类型

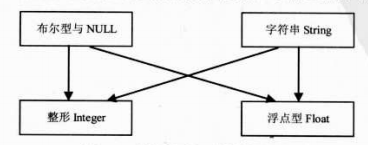
mixed：说明一个参数可以接受多种不同的(但并不必须是所有的)类型。例如getype()可以接受所有的php类型，str\_replace()可以接受字符串和数组。

number：说明一个参数可以是integer或者float。

callback：有些诸如call\_user\_function()或usort()的函数接受用户自定义的函数作为一个参数。callback函数不仅可以是一个简单的函数，他还可以是一个对象的方法，包括静态类的方法。一个php函数用函数名字符串来传递。可以传递任何内置的或者用户自定义的函数，除了array()，echo()，empty()，eval()，exit()，isset()，list()，print()和unset()。

数据类型之间相互转换

        自动转换通常发生在不同数据类型的变量混合运算时，若参加元算量的类型不同，则显转换成同一类型，然后在进行运算。通常只有四种标量类型(integer、float、string、boolean)才使用自动类型转换。这并没有改变这些运算书本身的类型，改变的近视这些运算数如何被求值。自动类型转换虽然石油系统自动完成的，但在混合运算时，自动转换要遵循转换按数据长度增加的方向进行，以保证精度不降低。



    有布尔型值参与运算时，true将转化为整型1，flase将转化为整型0后再参与运算。

    有NULL值参与运算时，NULL值转化为整型0在进行运算。

    有integer型和float型参与运算时，先把integer型变量转换成float类型后在进行运算。

    有字符串和数字型(integer、float)数据参与运算时，字符串先转换为数字，在参与运算。转化后的数字是从字符串开始的数值字符串，如果在字符串开始的数值型字符串不带小数点则转换为integer类型数字。如果带有小数点则转换为float类型数字。

强制类型转换

    再要转换的变量之前加上用括号括起来的目标类型，也可以使用具体的转换函数，即intval()、floatval()、strval()或者用setType()函数转换类型。

    intval()、floatval()、strval()强制类型转换没有改变变量本身的类型，而是通过转换将得到的新类型的数据赋值给新的变量，原变量的类型和值不变。如果需要改变自身的类型与值，可以使用setType()函数来设置变量的类型。

变量类型的测试函数

is\_bool()：判断是否是布尔型。

is\_int()、is\_integer()和is\_long()：判断是否是整型。

is\_float()、is\_double()和is\_real()：判断是否是浮点数。

is\_string()：判断是否是字符串。

is\_array()：判断是否是数组。

is\_object()：判断是否是对象。

is\_resource()：判断是否是资源类型。

is\_null()：判断是否为空。

is\_scalar()：判断是否是标量，也就是一个整数、浮点数、布尔型或字符串。

is\_numberic()：判断是否是任何类型的数字或数字字符串。

is\_callable()：判断是否是有效的函数名。

三、常量

常量只能包含四种标量数据。

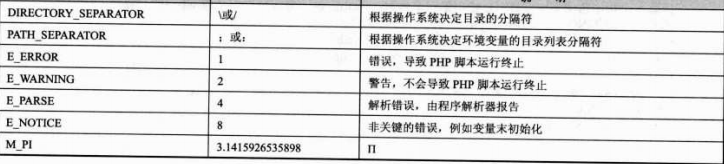
常量的定义

define(string name，mixed value[,bool case\_insensitive]);

第一个参数为常量名，第二个参数为常量的值或表达式，第三个参数是可选的，如果把case\_insensitive设为true，则常数将会定义成不区分大小写。如果只想检测是否定义了某常量，用defined()函数。

预定义常量

[](http://images2015.cnblogs.com/blog/712855/201601/712855-20160119161022140-196576463.png)

[](http://images2015.cnblogs.com/blog/712855/201601/712855-20160119161022640-1435213938.png)

循环语句

break

        在循环语句中中断循环语句，也就是结束循环语句的执行。

        break语句可以将深埋在嵌套循环中的语句退出指定层数或直接退出最外层，break是接受一个可选的数字参数来决定跳出几重语句。break可以跳出几重循环语句或是几重switch语句。

continue

        continue 语句只能使用在循环语句内部，功能是跳过该次循环，继续执行下一次循环结构。在while和do-while语句中continue语句跳转到循环条件处开始继续执行，对于for循环随后的动作是变量更新。.

        接受一个可选数字参数来决定条出多重语句。

        在循环中遇到continue语句后，就不会执行该循环中位于是continue后的任何语句。

        continue语句用于结束循环，继续下一次循环。

exit

        当前脚本只要执行到exit语句，而不管它在那个结构中都会直接退出当前脚本。exit()是一个函数，die(）函数就是exit()函数的别名，可以带有一个参数输出一条信息，并退出当前脚本。

函数

数组排序

PHP - 数组排序函数

sort() - 对数组进行升序排列

rsort() - 对数组进行降序排列

asort() - 根据关联数组的值，对数组进行升序排列

ksort() - 根据关联数组的键，对数组进行升序排列

arsort() - 根据关联数组的值，对数组进行降序排列

krsort() - 根据关联数组的键，对数组进行降序排列

超级全局变量

PHP中预定义了几个超级全局变量（superglobals） ，这意味着它们在一个脚本的全部作用域中都可用。

$GLOBALS

$\_SERVER

$\_REQUEST

$\_POST

$\_GET

$\_COOKIE

$\_FILES

$\_ENV

$\_SESSION

$GLOBAL 是一个包含了全部变量的全局组合数组。变量的名字就是数组的键。

$\_SERVER 是一个包含了诸如头信息(header)、路径(path)、以及脚本位置(script locations)等等信息的数组。这个数组中的项目由 Web 服务器创建。不能保证每个服务器都提供全部项目；

$\_REQUEST 用于收集HTML表单提交的数据。

$\_POST 被广泛应用于收集表单数据，在HTML form标签的指定该属性："method="post"

$\_GET 同样被广泛应用于收集表单数据，在HTML form标签的指定该属性："method="get"。

$\_GET 也可以收集URL中发送的数据。

魔术变量

有八个魔术常量它们的值随着它们在代码中的位置改变而改变。

\_LINE\_\_  
文件中的当前行号。

FILE  
文件的完整路径和文件名。如果用在被包含文件中，则返回被包含的文件名。

DIR  
文件所在的目录。如果用在被包括文件中，则返回被包括的文件所在的目录。

FUNCTION  
函数名称（PHP 4.3.0 新加）。自 PHP 5 起本常量返回该函数被定义时的名字（区分大小写）。在 PHP 4 中该值总是小写字母的。

CLASS 类的名称（PHP 4.3.0 新加）。自 PHP 5 起本常量返回该类被定义时的名字（区分大小写）。

在 PHP 4 中该值总是小写字母的。类名包括其被声明的作用区域（例如 Foo\Bar）。注意自 PHP 5.4 起 CLASS 对 trait 也起作用。当用在 trait 方法中时，CLASS 是调用 trait 方法的类的名字。

TRAIT Trait 的名字（PHP 5.4.0 新加）。自 PHP 5.4.0 起，PHP 实现了代码复用的一个方法，称为 traits。

Trait 名包括其被声明的作用区域（例如 Foo\Bar）。

从基类继承的成员被插入的 SayWorld Trait 中的 MyHelloWorld 方法所覆盖。其行为 MyHelloWorld 类中定义的方法一致。优先顺序是当前类中的方法会覆盖 trait 方法，而 trait 方法又覆盖了基类中的方法。

METHOD 类的方法名（PHP 5.0.0 新加）。返回该方法被定义时的名字（区分大小写）

NAMESPACE 当前命名空间的名称（区分大小写）。此常量是在编译时定义的（PHP 5.3.0 新增）。

命名空间

PHP 命名空间(namespace)是在PHP 5.3中加入的，如果你学过C#和Java，那命名空间就不算什么新事物。 不过在PHP当中还是有着相当重要的意义。

PHP 命名空间可以解决以下两类问题：

用户代码与PHP内部的类/函数/常量之间的名字冲突。

创建别名（或简短）的名称，提高源代码的可读性。

默认情况下，所有常量、类和函数名都放在全局空间下，就和PHP支持命名空间之前一样。  
如果一个文件中包含命名空间，它必须在其它所有代码之前声明命名空间

在声明命名空间之前唯一合法的代码是用于定义源文件编码方式的 declare 语句。所有非 PHP 代码包括空白符都不能出现在命名空间的声明之前。

命名空间使用  
PHP 命名空间中的类名可以通过三种方式引用：

非限定名称，或不包含前缀的类名称，例如 $a=new foo(); 或 foo::staticmethod();。如果当前命名空间是 currentnamespace，foo 将被解析为 currentnamespace\foo。如果使用 foo 的代码是全局的，不包含在任何命名空间中的代码，则 foo 会被解析为foo。 警告：如果命名空间中的函数或常量未定义，则该非限定的函数名称或常量名称会被解析为全局函数名称或常量名称。~~无命名空间前缀的引用，默认在当前命名空间查找，然后到全局命名空间找

限定名称,或包含前缀的名称，例如 $a = new subnamespace\foo(); 或 subnamespace\foo::staticmethod();。如果当前的命名空间是 currentnamespace，则 foo 会被解析为 currentnamespace\subnamespace\foo。如果使用 foo 的代码是全局的，不包含在任何命名空间中的代码，foo 会被解析为subnamespace\foo。 ~~带相对命名空间前缀的(类似相对路径)引用，自动在开头添加调用代码所处的命名空间

完全限定名称，或包含了全局前缀操作符的名称，例如， $a = new \currentnamespace\foo(); 或 \currentnamespace\foo::staticmethod();。在这种情况下，foo 总是被解析为代码中的文字名(literal name)currentnamespace\foo。~~带绝对命名空间前缀的引用(类似绝对路径) 引用路径已明确 不用查找

命名空间和动态语言特征

PHP 命名空间的实现受到其语言自身的动态特征的影响。因此，如果要将下面的代码转换到命名空间中，动态访问元素。

namespace关键字和\_\_NAMESPACE\_\_常量

使用\_\_NAMESPACE\_\_动态创建名称

关键字 namespace 可用来显式访问当前命名空间或子命名空间中的元素。它等价于类中的 self 操作符。

在PHP中，别名是通过操作符 use 来实现的. 下面是一个使用所有可能的三种导入方式的例子：

使用use操作符导入/使用别名

一行中包含多个use语句

导入和动态名称

导入和完全限定名称

使用命名空间：后备全局函数/常量

在一个命名空间中，当 PHP 遇到一个非限定的类、函数或常量名称时，它使用不同的优先策略来解析该名称。

类名称总是解析到当前命名空间中的名称(找不到就报错咯)。因此在访问系统内部或不包含在命名空间中的类名称时，必须使用完全限定名称

函数和常量来说，如果当前命名空间中不存在该函数或常量，PHP 会退而使用全局空间中的函数或常量。

全局空间  
如果没有定义任何命名空间，所有的类与函数的定义都是在全局空间，与 PHP 引入命名空间概念前一样。在名称前加上前缀 \ 表示该名称是全局空间中的名称，即使该名称位于其它的命名空间中时也是如此。PHP文件处理

打开文件

$file = fopen($fname, mode) //文件的名称, 打开模式

r 只读。在文件的开头开始。

r+ 读/写。在文件的开头开始。

w 只写。打开并清空文件的内容；如果文件不存在，则创建新文件。

w+ 读/写。打开并清空文件的内容；如果文件不存在，则创建新文件。

a 追加。打开并向文件末尾进行写操作，如果文件不存在，则创建新文件。

a+ 读/追加。通过向文件末尾写内容，来保持文件内容。

x 只写。创建新文件。如果文件已存在，则返回 FALSE 和一个错误。

x+ 读/写。创建新文件。如果文件已存在，则返回 FALSE 和一个错误。

注释：如果 fopen() 函数无法打开指定文件，则返回 0 (false)。

Filesystem 函数

basename() 返回路径中的文件名部分。

chgrp() 改变文件组。

chmod() 改变文件模式。

chown() 改变文件所有者。

clearstatcache() 清除文件状态缓存。

copy() 复制文件。

delete() 参见 unlink() 或 unset()

dirname() 返回路径中的目录名称部分。

disk\_free\_space() 返回目录的可用空间。

disk\_total\_space() 返回一个目录的磁盘总容量。

diskfreespace() disk\_free\_space() 的别名。

fclose() 关闭打开的文件。

feof() 测试文件指针是否到了文件末尾。

fflush() 向打开的文件刷新缓冲输出。

fgetc() 从打开的文件中返回字符。

fgetcsv() 从打开的文件中解析一行，校验 CSV 字段。

fgets() 从打开的文件中返回一行。

fgetss() 从打开的文件中返回一行，并过滤掉 HTML 和 PHP 标签。

file() 把文件读入一个数组中。

file\_exists() 检查文件或目录是否存在。

file\_get\_contents() 把文件读入字符串。~~获取文件内容

file\_put\_contents() 把字符串写入文件。~~写入文件内容

fileatime() 返回文件的上次访问时间。

filectime() 返回文件的上次修改时间。

filegroup() 返回文件的组 ID。

fileinode() 返回文件的 inode 编号。

filemtime() 返回文件内容的上次修改时间。

fileowner() 返回文件的用户 ID （所有者）。

fileperms() 返回文件的权限。

filesize() 返回文件大小。

filetype() 返回文件类型。

flock() 锁定或释放文件。

fnmatch() 根据指定的模式来匹配文件名或字符串。

fopen() 打开一个文件或 URL。

fpassthru() 从打开的文件中读数据，直到文件末尾（EOF），并向输出缓冲写结果。

fputcsv() 把行格式化为 CSV 并写入一个打开的文件中。

fputs() fwrite() 的别名。

fread() 读取打开的文件。

fscanf() 根据指定的格式对输入进行解析。

fseek() 在打开的文件中定位。

fstat() 返回关于一个打开的文件的信息。

ftell() 返回在打开文件中的当前位置。

ftruncate() 把打开文件截断到指定的长度。

fwrite() 写入打开的文件。

glob() 返回一个包含匹配指定模式的文件名/目录的数组。

is\_dir() 判断文件是否是一个目录。

is\_executable() 判断文件是否可执行。

is\_file() 判断文件是否是常规的文件。

is\_link() 判断文件是否是连接。

is\_readable() 判断文件是否可读。

is\_uploaded\_file() 判断文件是否是通过 HTTP POST 上传的。

is\_writable() 判断文件是否可写。

is\_writeable() is\_writable() 的别名。

lchgrp() 改变符号连接的组所有权。

lchown() 改变符号连接的用户所有权。

link() 创建一个硬连接。

linkinfo() 返回有关一个硬连接的信息。

lstat() 返回关于文件或符号连接的信息。

mkdir() 创建目录。

move\_uploaded\_file() 把上传的文件移动到新位置。

parse\_ini\_file() 解析一个配置文件。

parse\_ini\_string() 解析一个配置字符串。

pathinfo() 返回关于文件路径的信息。

pclose() 关闭由 popen() 打开的进程。

popen() 打开一个进程。

readfile() 读取一个文件，并写入到输出缓冲。

readlink() 返回符号连接的目标。

realpath() 返回绝对路径名。

realpath\_cache\_get() 返回高速缓存条目。

realpath\_cache\_size() 返回高速缓存大小。

rename() 重命名文件或目录。

rewind() 倒回文件指针的位置。

rmdir() 删除空的目录。

set\_file\_buffer() 设置已打开文件的缓冲大小。

stat() 返回关于文件的信息。

symlink() 创建符号连接。

tempnam() 创建唯一的临时文件。

tmpfile() 创建唯一的临时文件。

touch() 设置文件的访问和修改时间。

umask() 改变文件的文件权限。

unlink() 删除文件。

PHP JSON

json\_encode 对变量进行 JSON 编码

json\_decode 对 JSON 格式的字符串进行解码，转换为 PHP 变量

json\_last\_error 返回最后发生的错误

string json\_encode ( $value [, $options = 0 ] )  
参数

value: 要编码的值。该函数只对 UTF-8 编码的数据有效。  
options:由以下常量组成的二进制掩码：JSON\_HEX\_QUOT, JSON\_HEX\_TAG, JSON\_HEX\_AMP, JSON\_HEX\_APOS, JSON\_NUMERIC\_CHECK,JSON\_PRETTY\_PRINT, JSON\_UNESCAPED\_SLASHES, JSON\_FORCE\_OBJECT

mixed json\_decode ($json [,$assoc = false [, $depth = 512 [, $options = 0 ]]])  
参数

json\_string: 待解码的 JSON 字符串，必须是 UTF-8 编码数据

assoc: 当该参数为 TRUE 时，将返回数组，FALSE 时返回对象。

depth: 整数类型的参数，它指定递归深度

options: 二进制掩码，目前只支持 JSON\_BIGINT\_AS\_STRING 。