### 常量：

值不能修改

创建常量：

define(string 常量名，value 常量值，boolean 常量名是否大小写敏感)；第三个参数可以省略，默认值为FASLE，对大小写敏感

TRUE –大小写不敏感

FALSE—大小写敏感

如：定义一个名字为car，值为BMW，大小写敏感的常量define(“car”,”BMW”,false);

常量是全局的，在脚本的任意位置都可以访问。

魔术常量（值随着在代码中的位置改变而改变），不区分大小写：

1.\_LINE\_

文件中当前的行号

2.\_FILE\_

文件的完整路径和文件名（绝对路径）

**echo** "当前文件路径 ： ".\_\_FILE\_\_."<br>";

输出结果：D:\AppServ\www\php\_workspace\myproj\constant\define\_constant.php

3.\_DIR\_

当前文件所在的目录

4.\_\_FUNCTION\_\_

返回当前函数的被定义时的名称（区分大小写）

5.\_\_CLASS\_\_

返回类被定义时的名字（区分大小写）

**class** MyClass {

**function** \_print(){

**echo** "当前类名为 ： ".\_\_CLASS\_\_."<br>" ;

}

}

$myClass = **new** MyClass() ;

$myClass ->\_print() ;

6.\_TRAIT\_

7.\_\_METHOD\_

返回当前方法被定义时的名称

8.\_\_NAMESPACE\_\_

返回当前命名空间的名称

### 变量：

1.变量以$符号开始，后面跟着变量的名称

2.变量名必须以字母或者下划线字符开始

3.变量名只能包含字母数字字符以及下划线

4.变量名不能包含空格

5.变量名区分大小写

全局变量：在所有函数外部定义的变量

PHP中所有的全局变量都被保存在一个名为$GLOBAL[index]的数组中，index保存变量的名称，变量的名字就是数组的键。该数组可以在函数内部访问，也可以直接用来更新全局变量

局部变量：在函数内部声明，仅能在函数内部访问

变量的关键字

global: 函数内访问全局变量

在函数内部调用函数外定义的全局变量，需要在函数中的变量前加上global关键字

$x=5;

$y=10;

function myTest() {

global $x,$y;

$y=$x+$y;

}

超全局变量

1.$GLOBALS

PHP 在名为 $GLOBALS[index] 的数组中存储了所有全局变量。变量的名字就是数组的键。

<?php

$x = 75;

$y = 25;

function addition() {

$GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];

}

addition();

echo $z;

?>

2.$\_SERVER -- 保存关于报头、路径和脚本位置的信息

<?php

echo $\_SERVER['PHP\_SELF'];

echo $\_SERVER['SERVER\_NAME'];

echo $\_SERVER['HTTP\_HOST'];

echo $\_SERVER['HTTP\_REFERER'];

echo $\_SERVER['HTTP\_USER\_AGENT'];

echo $\_SERVER['SCRIPT\_NAME'];

?>

3.$\_REQUEST

 用于收集 HTML 表单提交的数据

<html>

<body>

<form method="post" action="<?php echo $\_SERVER['PHP\_SELF'];?>">

Name: <input type="text" name="fname">

<input type="submit">

</form>

<?php

$name = $\_REQUEST['fname'];

echo $name;

?>

</body>

</html>

4. $\_POST

用于收集提交 method="post" 的 HTML 表单后的表单数据。$\_POST 也常用于传递变量

<html>

<body>

<form method="post" action="<?php echo $\_SERVER['PHP\_SELF'];?>">

Name: <input type="text" name="fname">

<input type="submit">

</form>

<?php

$name = $\_POST['fname'];

echo $name;

?>

</body>

</html>

5.$\_GET

$\_GET 也可用于收集提交 HTML 表单 (method="get") 之后的表单数据

## 文件操作

1.readFile（文件名） 函数

2.fopen(文件名，文件打开模式)函数

<?php

$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");

echo fread($myfile,filesize("webdictionary.txt"));

fclose($myfile);

?>

文件打开模式：

|  |  |
| --- | --- |
| **模式** | **描述** |
| r | 打开文件为只读。文件指针在文件的开头开始。 |
| w | 打开文件为只写。删除文件的内容或创建一个新的文件，如果它不存在。文件指针在文件的开头开始。 |
| a | 打开文件为只写。文件中的现有数据会被保留。文件指针在文件结尾开始。创建新的文件，如果文件不存在。 |
| x | 创建新文件为只写。返回 FALSE 和错误，如果文件已存在。 |
| r+ | 打开文件为读/写、文件指针在文件开头开始。 |
| w+ | 打开文件为读/写。删除文件内容或创建新文件，如果它不存在。文件指针在文件开头开始。 |
| a+ | 打开文件为读/写。文件中已有的数据会被保留。文件指针在文件结尾开始。创建新文件，如果它不存在。 |
| x+ | 创建新文件为读/写。返回 FALSE 和错误，如果文件已存在。 |

3.读取打开的文件 fread 函数

fread(文件名，待读取的最大字节数);

filesize(文件名) ： 获取文件的大小

把 "webdictionary.txt" 文件读至结尾：

fread($myfile,filesize("webdictionary.txt"));

4.关闭已经打开的文件

fclose(需要关闭的文件的名称);

5.打开文件后通过fgets函数获取文件中的首行数据

调用 fgets() 函数之后，文件指针会移动到下一行

<?php

$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");

echo fgets($myfile);

fclose($myfile);

?>

6.feof(文件名) ： 判断文件读取是否结束 file – end – of - file

<?php

$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");

// 输出单行直到 end-of-file

while(!feof($myfile)) {

echo fgets($myfile) . "<br>";

}

fclose($myfile);

?>

7.fgetc(文件名) ： 读取单字符

<?php

$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");

// 输出单字符直到 end-of-file

while(!feof($myfile)) {

echo fgetc($myfile);

}

fclose($myfile);

8.创建新的文件

fopen函数即可打开文件，也可以创建一个新文件

$myfile = fopen("testfile.txt", "w")

或

$myfile = fopen("testfile.txt", "a")

9.向文件中写入数据

先要打开文件

fwrite(文件名，字符串)

关闭文件

如果文件中已经包含了一些数据，执行完fwrite函数后，文件中已有的数据会被擦除，并最终写入了新的数据

10.追加数据到文件中

fputs 函数：

$myfile = fopen("text", "a");

fputs($myfile, "continue to write file !");

fclose($myfile) ;

## Cookie

cookie 常用于识别用户。cookie 是服务器留在用户计算机中的小文件。每当相同的计算机通过浏览器请求页面时，它同时会发送 cookie。

1.创建cookie

setcookie(name, value, expire, path, domain);

在发送 cookie 时，cookie 的值会自动进行 URL 编码，在取回时进行自动解码（为防止 URL 编码，请使用 setrawcookie() 取而代之）。

2.创建不编码的cookie

setrawcookie("school","JGSU",time()+3600) ;

3.取回cookie

PHP 的 $\_COOKIE 变量用于取回 cookie 的值

<?php

// Print a cookie

echo $\_COOKIE["user"];

//打印输出所有的cookie

print\_r($\_COOKIE);

?>

4.通过isset()函数判断是否已经创建了某个cookie

isset($\_COOKIE["user"])

5.删除cookie

当删除 cookie 时，应当使过期日期变更为过去的时间点。

<?php

// set the expiration date to one hour ago

setcookie("user", "", time()-3600);

?>

## SESSION

1.开启SESSION

把用户信息存储到 PHP session 中之前，首先必须启动会话。

session\_start() 函数必须位于 <html> 标签之前：

<?php session\_start(); ?>

<html>

<body>

</body>

</html>

2.存储SESSION变量

存储和取回 session 变量的正确方法是使用 PHP $\_SESSION 变量：

<?php

session\_start();

// store session data

$\_SESSION['views']=1;

?>

<html>

<body>

<?php

//retrieve session data

echo "Pageviews=". $\_SESSION['views'];

?>

</body>

</html>

3. 简单的 page-view 计数器

<?php

session\_start();

if(isset($\_SESSION['views']))

$\_SESSION['views']=$\_SESSION['views']+1;

else

$\_SESSION['views']=1;

echo "Views=". $\_SESSION['views'];

?>

4. 删除某些 session 数据

unset() 函数用于释放指定的 session 变量

<?php

unset($\_SESSION['views']);

?>

5.  session\_destroy() 函数彻底终结 session：

<?php

session\_destroy();

?>

## 类

PHP中定义类：

**class** Student {

**var** $name ; //定义成员变量

**var** $gender ;

**var** $contacts ;

//定义成员方法

**function** say(){

**echo** "My Name is : ".**$this**->name."<br>My gender is : "

.**$this**->gender."My contact is : ".**$this**->contacts."<br>";

}}

### 类的构造方法：

如果不显示声明，和Java一样，存在一个不带参数且实现为空的构造方法

PHP 5.0以前命令规则：

function 类名([参数列表]){

//构造方法实现

}

PHP5.0及其以后命名规则（对之前的类名命名规则兼容）

两个下划线+construct关键字

\_\_construct([参数列表]){

// 构造方法实现

}

### 类的析造方法（PHP5.0之后添加的）：