链滴

PHP 语法基础与常用操作

作者：someone38063
原文链接：https://ld246.com/article/1560592263526
来源网站：链滴

PHP是一种运行在服务器端的脚本语言、可以嵌入到HTML中

PHP代码标记

在PHP历史发展中，可以使用多种标记来区分PHP脚本

1. ASP标记：　<% php代码%>
2. 短标记：　<?php代码?>
3. 脚本标记：　<script language="php">php代码</script>
4. 标准标记：　<?php php代码 ?>

(1、2基本弃用)

PHP注释

行注释：//comment、#comment

块注释：

```
/*
* @autor:
* @para:
* @version:
*/
```

@符号：阻止警告输出

PHP语句分隔符

在PHP中，代码都是以行为单位，系统需要判定行的结束，通常以";"结束

- **PHP中标记结束符?>有自带语句结束符的效果，最后一行PHP代码可以没有";"**
- **PHP很多代码的书写不是嵌入到HTML中，而是单独存在，通常书写习惯就不建议使用结束标记符?，PHP会认为从开始到结尾都是PHP代码，从而解析**

变量

删除变量：unset($var)

预定义变量

提前定义的变量，系统定义的变量，存储许多需要的数据(预定义变量都是数组)

$_GET:获取所有表单以GET方式提交的数据

$_POST:POST提交的数据都会保存在此

$_REQUEST:GET和POST提交的都会保存在此

$_GLOBALS:PHP所有全局变量

$_SERVER:服务器信息

$_SESSION:session会话数据

$_COOKIE:cookie会话数据

$_ENV:环境信息

$_FILES:用户上传的文件信息

可变变量

可变变量：如果一个变量保存的值刚好是另外一个变量的名字，那么可以通过直接访问一个变量得到外一个变量的值，在变量前面再加一个$符号

```
$a = 'b';

$b = 'string';

echo $$a;

//——>string
```

php变量命名规则：

1. 变量以$符号，其后是并变量
2. 变量名称必须以字母或下划线开头
3. 变量不能以数字开头
4. 变量名称只能包含数字、字母和下划线
5. 变量名称对大小写敏感

变量传值

把一个变量赋值给另外一个变量

1. 值传递：　　a = b将变量保存的值复制一份，将新的值给另外一个变量保存(两个变量没有关系)
2. 引用传递：　　a = &b将变量保存的值所在的内存地址，传递给另外一个变量；两个变量指向同一地址空间

- 在内存中，通常有以下几个分区

栈区：程序可以操作的内存部分(不存数据)，空间小但执行快

代码段：程序可以操作的内存部分(不执行)

数据段：存储普通数据(全局区和静态区)

堆区：存储复杂数据，(空间大效率低)

常量

常量一旦定义，通常数据不可改变

PHP5.3后有两种定义方式

1. 使用定义常量的函数：define('常量名','常量值');
2. 5.3之后：const 常量名=值;

命名规则：

1. 不需要使用'$'，一旦使用会认为是变量;
2. **通常大写**
3. 常量命名的规则比变量松散，可以使用一些特殊字符，该方式只能用define函数定义

define()和const定义的常量有**访问权限区别**

- 特殊符号的常量不能直接访问，需要用到访问常量的函数；比如不能

define('-_-','smile');

echo -_-;

应为 echo constant('-_-');

系统常量

系统帮助用户定义的常量，可以直接使用

- 常用的系统常量

PHP_VERSION：PHP版本

PHP_INT_SIZE：整型大小(字节数)

PHP_INT_MAX：整型最大值

- 特殊的常量，它们以双下划线开始+常量名+双下划线结尾，这种常量称之为系统魔术常量，它们值会跟着环境变化，用户无法改变

__DIR__:当前被执行的脚本所在电脑绝对路径

__FILE__:当前被执行的脚本所在电脑绝对路径(带文件名)

__LINE__:当前所属行数

__NAMESPACE__:当前所属命名空间

__CLASS__:当前所属类

__METHOD__:当前所属方法

数据类型

PHP是一种弱类型语言，变量本身没有数据类型

数据分为三大类八小类

1. 简单数据类型
2. 整型 4字节
3. 浮点型 8字节
4. 字符串型
5. 布尔类型 true false
6. 复合数据类型
7. 对象类型
8. 数组类型
9. 特殊数据类型
10. 资源类型：存放资源数据(PHP外部资源，如数据库、文件)
11. 空类型：值为NULL(不能运算)

类型转换

1. 自动转换：系统自动判定并转换
2. 强制转换：在变量前增加一个(),然后在里面写上对应类型

- 其他类型转数值的说明

1. 布尔true为1，false为0
2. 字符串转数值有自己的规则
3. 以字母为开头的字符串永远为0
4. 艺术字开头的字符串，取到碰到字符串为止(不会同时包含两个小数点)

函数

function functionName() {

```
code
}
```

- 函数名以字母或下划线开头
- 函数名对大小写不敏感

静态变量

```
function test() {
static $a=1;
$a++;
echo $a;
test();//1
test();//2
test();//3
}
```

1. 在函数执行完后，变量值仍然保存
2. 修饰属性或方法，可以通过类名访问，如果是修饰的是类的属性，保留值

内部变量与外部变量

- PHP中的全局变量global和$GLOBALS的区别

1. global

Global的作用是定义全局变量,但是这个全局变量不是应用于整个网站,而是应用于当前页面,包括include或require的所有文件。

但是在函数体内定义的global变量,函数体内可以使用,在函数体外定义的global变量不能在函数内使用,具体看下面示例。

2. $GLOBALS

在 $GLOBALS 数组中，每一个变量为一个元素，键名对应变量名，值对应变量的内容。$GLOBALS 所以在全局范围内存在，是因为 $GLOBALS 是一个超全局变量。注意GLOBALS 的写法，比如变量a1写法为$GLOBALS['a1']。

- 不能在用global声明变量的同时给变量赋值。

```
<?php
```

```
$name = "why";

function changeName(){

$name = "what";

}

changeName();

echo "my name is " . $name . "<br/>";

?>
```

代码执行结果：my name is why，因为函数体changeName内$name变量被缺省设置为局部变量,$name的作用域就是在changeName内

修改代码：

```
<?php

global $name;

$name = "why";

function changeName(){

$name = "what";

}

changeName();

echo "my name is " . $name . "<br/>";

?>
```

return

```
function demo() {

return 1;

$a = demo();

echo $a;//1

}
```

引用传参方式

function demo(&变量名){}

引用传递(pass-by-reference)过程中，被调函数的形式参数虽然也作为局部变量在堆栈中开辟了内存间，但是这时存放的是由主调函数放进来的实参变量的地址。被调函数对形参的任何操作都被处理成接寻址，即通过堆栈中存放的地址访问主调函数中的实参变量。正因为如此，被调函数对形参做的任操作都影响了主调函数中的。

同理 跟变量引用的原理是一样的

数组

数组就是在内存里有一块连续的内存空间（堆空间），这里面可以保存多个数据，数据没有数据类型限制。

- 基本语法

1. $arr = array(元素1，元素2，元素3...); //数据类型一定是数组
2. arr[] = 元素1; //定义一个arr变量，将元素1添加进来

arr[] = 元素2; //将元素2添加到arr变量中

访问数组

数组本质是变量，访问的时候就是直接使用变量，但是因为数组元素有多个，而系统没有办法直接区到底用户是访问的哪一个，所以需要用户在访问数组的时候，使用数组下标（键名）

语法：$数组变量[下标]

数值下标：$arr[数值]

字符串下标：$arr[‘字符串’]

```
$arr[]='a';

$arr[]='b';

$arr[]='c';

$arr[]='d';

var_dump($arr[1]);//string(1) "b"

echo $arr[2];//c
```

数组分类

根据数组的下标的不同进行分类

- 索引数组：当前数组的下标（键名）全是数字（整型）

定义方式：系统自动分配下标

定义方式：用户手动分配下标

系统分配所引方式

1. 系统是0开始分配下标，依次递增1
2. 用户可以手动分配下标（下标不能重复：重复的效果就是覆盖）
3. 如果用户在某一处指定了下标，那么后续元素自动分配的下标从前面的最大值（索引）开始

```
$arr1 = arry(0=>'a',1=>'b',2=>'c');
```

- 关联数组：当前数组的下标全是字符串（使用最多）

1. 下标必须使用引号包含
2. 下标不允许重复，重复会覆盖

```
$arr2['name'] = 'zhang';
$arr2['age'] = 'twenty';
$arr2['job'] = 'teacher';
```

```
$arr3 = array('name'=>'zhang','age'=>'twenty''job'=>'teacher');
```

- 混合数组：数组的下标既有数字又有字符串
- count() 函数返回数组中元素的数目。
- for循环遍历索引数组

```
$arr[]='a';
$arr[]='b';
$arr[]='c';
$arr[]='d';
for($i=0;$i<count($arr);$i++) {
echo $arr[$i];//abcd
```

```
}
```

关联或索引数组遍历

- 如果数组是一个关联数组，那么就完全不能使用for循环来进行遍历。PHP提供了一个foreach来实对数组元素的遍历
- 语法：

```
foreach(数组 as 键值对$key => $value){
        //使用$key当做当前元素的下标
        //使用$value当做当前元素的值
}
```

循环原理

```
$array = array('name'=>'zhang','age'=>'20',;height'=>'120');
```

1. 第一次进入foreach，系统会通过变量array找到其对应的内存地址
2. 找到当前数组的第一个元素，并把该元素取出来，元素包含两个部分：键和值(指针下移)
3. 将键赋值给key变量，将值赋给value变量
4. 执行循环体
5. 找到当前已经被获取元素的下一个元素，将其对应的两个部分赋值给key和value
6. 执行循环体

.....重复执行5和6，直到循环结束

each() 没有循环的指针

```
$arr = array('demo1'=> 'woaini','demo2'=> 2222,'demo3'=> 333,'demo4'=> 'vvvvv');

var_dump(each($arr)); //指针1
var_dump(each($arr)); //指针2
var_dump(each($arr));
var_dump(each($arr));
```

List的用法

```
list($first,$second,$third) = array('1','2','3');

echo($first); //1
```

获取时间戳

time() //获取当前时间戳

date(format,timestamp) //时间戳转换日期时间

参数	描述
format	必需。规定时间戳的格式。
timestamp	可选。规定时间戳。默认是当前时间日期。

时间戳转日期

Y： 年（四位数）大写

m : 月（两位数，首位不足补0） 小写

d： 日（两位数，首位不足补0） 小写

H：小时 带有首位零的 24 小时小时格式

h： 小时 带有首位零的 12 小时小时格式

i： 带有首位零的分钟

s： 带有首位零的秒（00 -59）

a：小写的午前和午后（am 或 pm）

```
var_dump(date('Y-m-d H:i:s', 1560416826));

var_dump(date('Y-m-d h:i:s', 1560416826));
```

打开或创建文件

● fopen() 的第一个参数包含被打开的文件名，第二个参数规定打开文件的模式。

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

参数	描述
r	打开文件为只读。文件指针在文件开头开始
w	打开文件为只写。删除文件的内容或创建一个的文件，如果文件不存在，文件指针在文件开头开始
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。如果文件不存在，创建新的文件

x 创建新文件为只写。如果文件已存在返回FALS
和错误

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
，如果文件不存在，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
，文件指针在文件结尾开始。如果不存在，创建新文件

x+ 创建新文件为读/写，如果文件已存在返回FLA
E和错误

读文件

Fgets($find) //读第一行文件，指针向后

Fread($find,获取字节) //指定获取内容

Filesize('./a.txt') // 获取文件字节

Fclose($find) //关闭资源

修改或添加内容

Fopen(./1.xx,“a+”);

Fwrite(‘文件’,’内容’);

Fclose(find);//关闭资源

- $ _GET变量：通过 URL 参数传递给当前脚本的变量的数组。

预定义的$ _GET变量用于收集来自method = “get”的表单中的值。

$_GET['数组键(索引)']

```
<?php
$a = $_GET['name'];
$file = fopen("./session.txt","w+");
fwrite($file,$a);
?>
```

这个脚本放在localhost/demo下，名为a.php

在浏览器中输入localhost/demo/a?name='String'，$_GET数组的值'String'会保存到当前目录下的s
ssoin.txt文件下

GET和POST接收

POST

index.html

```
<form action="a.php" method="post">
<input type="text" name="username" placeholder="请输入">
<p>
<input type="submit" value="提交">
</p>
</form>
```

a.php

```
<?php
header("Content-type: text/html; charset=utf-8");
include_once './index.html';
echo $_POST['username'];
?>
```

在浏览器中填入字段，会显示该字段

三元运算符

- empty(): 如果 **没有值**则为**真**，**有值**则为**假**
- isset(): 如果 **没有值**则为**假**，**有值**则为**真**

```
if(isset($_GET['name'])) {
echo $_GET['name'];
}
```

- 条件式?值1:值2

如果条件式为true，则为值1，否则为值2

```
$a = isset($_GET['name'])?echo $_GET['name']:
```

文件包含

```
<?php
include 'a.html';
?>
```

include_once 'a.html';会检查该文件(a.html)是否已经被包含，如果是则不再包含

PHP操作MYSQL数据库

php7以前的版本

- MySQL扩展的开启

1. 保证Apache读取了php.ini文件：PHPInDir php.ini文件路径
2. php.ini开启扩展extension：php_mysql.dll
3. 加载扩展路径：extension_Dir
4. 重启Apache

- PHP充当MySQL连接的客户端

1. 连接认证： new mysqli('host','user','passwd');
2. 主机地址：地址+端口：localhost:3306
3. 用户名
4. 用户密码

```
<?php
$link = new mysqli('localhost:3306','root','passwd');
var_dump($link);
?>
```

PHP7已经不能使用mysql_connect(),php7没有php_mysql这个扩展

SESSION和COOKIE

session默认不开启，可以通过php.ini查看session.auto_start = 0

1. 开启session机制： session_start()

2. 使用session数据：往$_SESSION数组添加元素
3. 读取session数据：　var_dump($_SESSION)
4. 销毁session数据：　session_destroy()

index.html

```
<form action="a.php" method="post">
<input type="text" name="username" placeholder="请输入">
<p>
<input type="text" name="password" placeholder="请输入">
</p>
<p>
<input type="submit" value="提交">
</p>
</form>
```

a.php

```
<?php
session_start();

$username = empty($_POST['username'])?'s':$_POST['username'];
$password = empty($_POST['password'])?'s':$_POST['password'];

if(isset($_SESSION['username'])) {
echo '1';
}
elseif($username == 'admin' and $password == '123') {
echo 'sign in successfully';
$_SESSION['username'] = '1';
```

```
}
else {
include_once './index.html';
}
?>
```

Cookie跨域

- 默认的，cookie只能对当前域名（完整域名：有效的二级域名）有效。
- cookie跨域：指的是允许cookie在不同的二级域名之间共享。（一级域名一致）

PHP设置COOKIE

Setcookie(“名字”,值,有效时间，‘有效路径’，“有效域");

Setcookie("PHPSESSID", session_id(),time()+100,'/',"xss.cn");

文件上传

- 要在服务器端开启文件上传功能

php.ini

```
file_uploads = On

; Temporary directory for HTTP uploaded files (will use system default if not
; specified).
; http://php.net/upload-tmp-dir
upload_tmp_dir ="D:/Wampserver/tmp"

; Maximum allowed size for uploaded files.
; http://php.net/upload-max-filesize
upload_max_filesize = 2M

; Maximum number of files that can be uploaded via a single request
max_file_uploads = 20
```

- 文件上传的临时目录

upload_tmp_dir

post默认提交字符流数据，不能提交二进制数据，如果需要提交二进制数据，需要给form表单额外加一个属性：enctype="multipart/form-data"

文件上传到服务器的时候，会先存放在临时目录里，PHP会用一组变量来保存临时文件，当脚本执行束后，PHP会释放全部所占内存，因此这部分文件会被操作系统回收，从而看不到临时文件，最终也有实现文件上传

- 文件上传原理

删除文件

unlink

输出信息

echo

print_r

var_dump