此文是根据php官方文档与php程序设计一书完成的。

## php简介

php(HyperText preprocessor,超文本预处理器)，是一种被广泛应用的开源代码的多用途脚本语言，可以嵌入到html中，尤其适合web开发。

与c写的脚本的区别—与用大量的命令来编写程序以输出html不同的是，php页面就是html.只不过在其中嵌入了一些代码来做一些事情

与客户端的javascript不同的是，php代码运行在服务端。

## php能做什么

### 服务端脚本

创建动态的web内容

### 命令行脚本

Php通过使用命令行脚本来执行系统管理任务，例如备份和日志解析

### 客户端GUI应用

Php可以通过PHP-GTK来编写客户端应用程序，php-GTK是php的一个扩展

## 基本语法

### Php标记

<? 和?>,解析php文件，php会寻找起始标记和结束标记，这种解析方式使得php可以嵌入到各种不同的文档中去，而任何起始和结束标记之外的部分都会被php解析器忽略

### 指令分割符

Php每个语句后用分号结束指令

### instanceof

用于确定一个 PHP 变量是否属于某一类 [class](https://www.php.net/manual/zh/language.oop5.basic.php#language.oop5.basic.class) 的实例：

返回bool类型

### method\_exists

检查类的方法是否存在

**method\_exists** ( [mixed](https://www.php.net/manual/zh/language.pseudo-types.php" \l "language.types.mixed) $object , string $method\_name ) : bool

### 注释

//

/\* \*/

## 词法结构

### 大小写

和内置结构的关键字(echo,class,while)一样,用户定义的类名和函数名也是不区分大小写的，

但是变量区分大小写

### 空白符与换行符

Php中空白符无关紧要，一条语句可以跨多行，也可以紧密的连在一起

### 标识符

标识符只是一个名字，在php里标识符用来命名变量、函数、常量和类

关键字

是语言为其核心功能而预留的单词，在对变量、函数、类或常量命名时不能使用和关键字相同的名字。

## 数据类型

## 变量

### 预定义变量

#### $\_SERVER

$\_SERVER[‘SERVER\_ADDR’]------服务器所在IP地址

$\_SERVER[‘HTTP\_USER\_AGENT’]------当前用户使用的浏览器信息

$\_SERVER[‘SERVER\_PORT’]-----服务器所使用的端口

$\_SERVER[‘PHP\_SELF’]------当前正在执行的脚本文件名

$\_SERVER[‘SERVER\_SIGNTURE’]-----当前服务器版本和虚拟主机名的字符串

$\_SERVER['REMOTE\_ADDR']---- 正在浏览当前页面用户的 IP 地址。

参考：<https://blog.csdn.net/moliyiran/article/details/79806108>

## 常量

## 表达式

## 运算符

### ===、==与=

|  |  |  |
| --- | --- | --- |
| x == y | 相等 | 如果 x 和 y 具有相同的键/值对，则返回 true |
| x === y | 恒等 | 如果 x 和 y 具有相同的键/值对，且顺序相同类型相同，则返回 true |

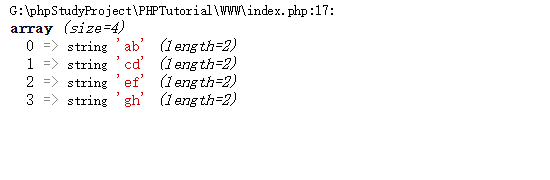
## 函数

## 字符串

### str\_split()字符串按位分割

$str = 'abcdefgh';

$arr = str\_split($str,2);



str\_split() 函数把字符串分割到数组中。

str\_split(*string*,*length*) length不填默认为1

## 数组

### 索引数组与关联数组

索引数组的键名是整数并从零开始

关联数组的键名时字符串

### 识别数组元素

$arr[‘Fred’]

$arr[3]

键可以是一个字符串或一个整数，等价于证数但不以0开头的字符串被当成整数看待，

因此，$arr[3]和$array[‘3’]是一样的，$arr[‘03’]不一样

负数是有效的键

不一定需要将单个字符串加上引号，$arr[‘Fred’]和$arr[Fred]是一样的，但一般不这么做，

因为没有引号的键不能和常量区分开

Define(‘index’,5);

Echo $arr[index];----输出的是$arr[5]

### 数组初始化

### 直接赋值

$arr[0]=1;

$arr[1]=2;

$arr[2]=3;

$test[‘color’]=’red’;

$test[‘size’]=13;

$tes[‘kind’]=’my’;

### Array初始化

$arr=array(1,2,3,4);

$test=array(‘color’=>’red’,’size’=>13,’kind’=>’my’)

可以用=》来指定一个初始索引，并在后面指定一些列键值

$days=array(1=>’mondy’,’tuesday’,’wednesday’)

数组的2号元素是Tuesday，3号元素是Wednesday

如果初始索引是一个非数字的字符串，那么后面的索引是从零开始的

$day=array(‘Friday’=>’black’,’brown’,’green’)

等价于

$day=array(‘Friday’=>’black’,0=>’brown’,1=>’green’)

### 在数组末尾添加值

使用[]语法,前提数该数组的索引是数字并且索引从零开始

$arr=array(‘color’,’size’,’kind’);

$arr[]=’name’;

所以$arr[3]=name

如果是关联数组会给新的元素指定数组

$arr[‘test’=>’color];

$arr[]=’size’;

结果是

$person[0]=size;

### 指定值的范围

range()函数创建一个连续的整数或字符数组，整数或字符的值在传递给函数的两个参数之间

$number=range(1,5); //等价于 $number=array(1,2,3,4,5);

$letter=range(‘a’,’f’); //保存字母到数字 $letter=range(‘a’,b’,c’,’d’,’e’,’f’);

$reversed=range(5,1);// $reversed=array(5,4,3,2,1);

字符串参数中只有第一个字母被用于限定范围

range(‘aaa’,’’zzz’);//等价于 range(‘a’,’z’);

### 获取数组大小

count()和sizeof()函数在用法和效果上是一样的

### 填充数组

array\_pad()函数用重复的内容来创建数组

array\_pad(第一个参数是函数，第二个参数是你想要拥有元素个数，第三个元素是要给每个元素的初始值)

$arr=array(1,2,3)

$pad=array\_pad($arr,5,0); //填充后的数组 array(1,2,3,0,0)

将新值填充到数组开头

$pad=array\_pad($arr,-5,0);//填充后的数组 array(0,0,1,2,3)

将array\_pad()函数的处理结果赋值给原来的数组，可达到改变原来位置的效果

$arr=array\_pad($arr,5,0);

如果你填充元素到一个关联数组，已经存在的键将被保留，新的元素将使用从0开始的数字键

### 多维数组

数组的值也可以是数组

$row\_0=array(1,2);

$row\_1=array(3,4);

$row\_2=array(5,6);

$test=array($row\_0,$row\_1,$row\_2);

要使用变量替换来访问多维数组的 中的元素，必须将整个元素用大括号括起来

Echo “this value is {$test[1][2]}”

不适用大括号结果如下

This value is array[2]

### 析取多个值

要把一个数组的所有值都赋值到变量中，可以使用list()结构

$person(‘lili’,23,’男’)；

list($name,$age,$gender)=$person;//$name=’lili’;$age=23 $gender=’男’

如果数组中的元素个数比list()中的变量个数多，则多余的数被忽略

list($name,$age,$gender)=$person;//$name=’lili’;$age=23

如果list()中的元素个数比数组中的多，则多余的值被设为NULL

$person(‘lili’,23’)；

list($name,$age,$gender)=$person;//$name=’lili’;$age=23 $gender=NULL

在list()中用两个或多个连续的逗号跳过数组中的值

$value=range(‘a’,’e’);

List($m,,$n,,$o)=$value; //$m=’a’,$n=’c’,$o=’e’;

### 切割数组

array\_slice()用来析取数组的一个子集

$subject=array\_slice(array,offset,length)

Offset指定要赋值元素的初始元素。Length指定要复制的值的个数

$people=array(1,2,3,4,5);

$middle=array\_slice($people,2,2) // 结果为 $middle=array(3,4)

只有在索引数组上使用array\_slice()才有效，

$person=array(‘name’=>’lili’,’age’=>35,’wife’=>’betty’);

$test=array\_slice($person,1,2);//结果为 $test=array(0=>35,1=>’betty’);

结合list()和array\_slice()使用

$order=array(1,2,3,4,5,6);

List($test1,$test2)=$array\_slice($order,1,2);

//结果为$test1=2;$test2=3;

### 数组分块

array\_chunk()函数把数组划分成小数组或固定大小的数组

$chunk=array\_chunk(array,size,preserve\_keys)

Size为划分数组的长度

Preserve\_keys是一个布尔值，它指定新数组的元素是否有和原数组相同的键，还是从0开始的新数字键,默认是分配新键值的

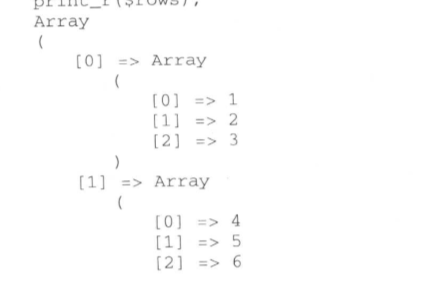
True 保留原始数组的键名

False 默认，每个新数组块使用从0开始

$num=range(1,7)

$rows=array\_chunk($num,3)

打印结果：



### compact()创建数组

创建一个包含变量名和它们的数值的数组

$firstname=”peter”;

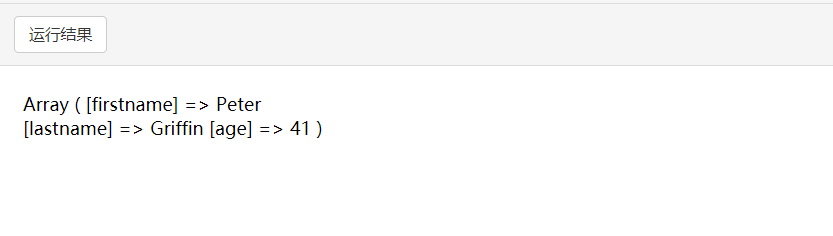
$lastname=”Griffin”;

$age=”41 ”;

$result=compact(“firstname”,”lastname”,”age”);

Print\_r($result);

结果：



### array\_push()

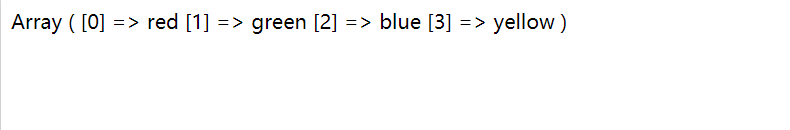
向第一个参数的数组的尾部添加一个或多个元素

$a=array(“red”,”green”);

array\_push($a,”blue”,”yellow”);

print($a);

结果：



### explode与implode

<https://www.cnblogs.com/usa007lhy/p/5278801.html>

explode(‘,’,$arr);按字符串分割数组

implode(‘,’,$arr);将素组按给定字符拼接成字符串

### 键和值

array\_keys()根据内部顺序返回一个仅由数组中的键组成的数组，等价于array\_of\_keys()

$person=array(‘name’=>’lili’,’age’=>13,’gender’=>’女’);

$keys=array\_keys($person); //结果 $key=array(‘name’,’age’,’gender’);

array\_values()根据内部顺序返回一个仅由数组中的值组成的数组

$person=array(‘name’=>’lili’,’age’=>13,’gender’=>’女’);

$values=array\_values($person); //结果 $ values =array(‘lili’,13,’女);

#### array\_keys()操作二维数组

$a=array(

array('x'=>'w'),

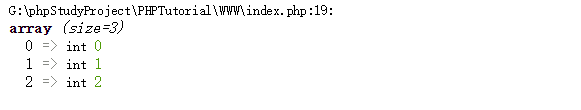
array('v'=>'e'),

array('b'=>'g')

);

var\_dump(array\_keys($a));

结果：



array\_values()操作二维数组

$a=array(

array('x'=>'w'),

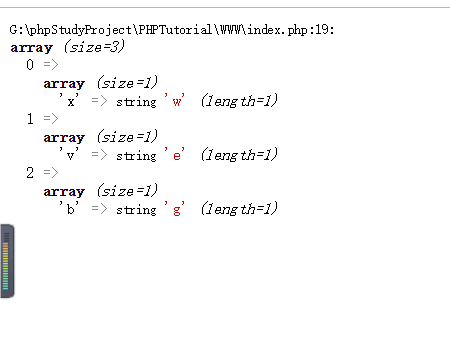
array('v'=>'e'),

array('b'=>'g')

);

var\_dump(array\_values($a));

结果：

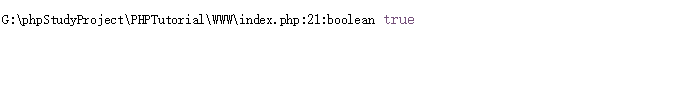


### 检查元素是否存在

Array\_key\_exists(key,array)用来检查数组中是否存在某个键值的元素，存在返回true，不存在返回false

$a=array('q'=>132,'e'=>'s');

var\_dump(array\_key\_exists('q',$a));



$a=array(

'b'=>array('x'=>'w'),

array('v'=>'e'),

array('b'=>'g')

);

var\_dump(array\_key\_exists('b',$a));



Isset也可以判断，不存在返回NULL存在返回false

### 在数组中删除和插入元素

array\_splice()可以在数组中删除和插入元素，并且可以用被删除的元素创建一个数组

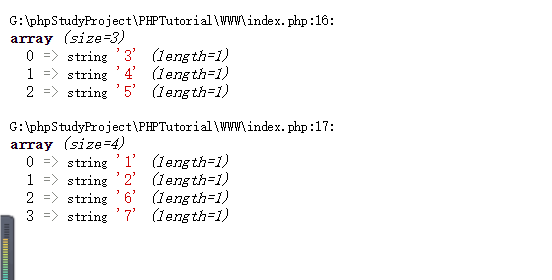
操作二维数组和一维数组一样，索引数组和关联数组一样

$subject=array('1','2','3','4','5','6','7');

$remove=array\_splice($subject, 2,3);

var\_dump($remove);

var\_dump($subject);



原数组从第二个位置往后删除三个元素，array\_splice()得到的是删除后的数组

$remove=array\_splice($subject,2);从第二个位置删除的最后

#### 在删除的位置加元素

被删除的元素个数不需要和插入的元素个数一致

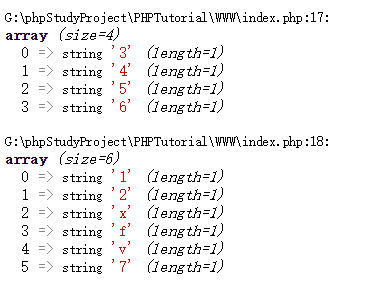
$subject=array('1','2','3','4','5','6','7');

$a=array('x','f','v');

$remove=array\_splice($subject, 2,4,$a);

var\_dump($remove);

var\_dump($subject);



#### 新元素插入数组

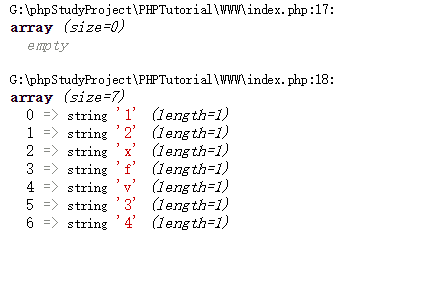
$subject=array('1','2','3','4');

$a=array('x','f','v');

$remove=array\_splice($subject, 2,0,$a);

var\_dump($remove);

var\_dump($subject);



### 在数组和变量间转换

**extract()将关联数组键值转成变量名，值转成变量的值**

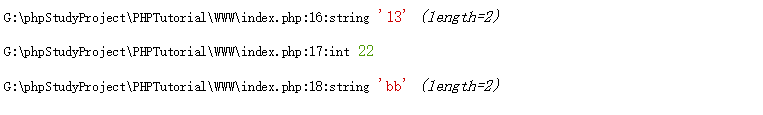
$test=array('a'=>'13','b'=>22,'c'=>'bb');

extract($test);

var\_dump($a);

var\_dump($b);

var\_dump($c);



**compact()将变量以变量名和值转成关联数组**

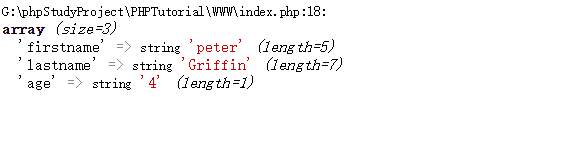
$firstname="peter";

$lastname="Griffin";

$age="4";

$result=compact("firstname","lastname","age");

var\_dump($result);

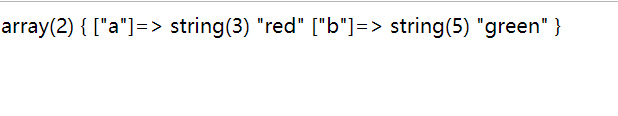


## Array\_unique();删除数组重复元素

array\_unique() 函数用于移除数组中重复的值。如果两个或更多个数组值相同，只保留第一个值，其他的值被移除。

array("a"=>"red","b"=>"green","c"=>"red");

var\_dump(array\_unique($a));



## 判断数组是否存在某个值

in\_array(value,array,type)

如果给定的值 *search* 存在于数组 *array* 中则返回 true。如果第三个参数设置为 true，函数只有在元素存在于数组中且数据类型与给定值相同时才返回 true。

如果没有在数组中找到参数，函数返回 false。

array\_search(value,array,strict)

array\_search() 函数在数组中搜索某个键值，并返回对应的键名。

<?php

$a=array("a"=>"red","b"=>"green","c"=>"blue");

echo array\_search("red",$a);

?>

输出a

## Empty(0)==true

Isset判断’‘为true

## chunk\_split()

|  |  |
| --- | --- |
| *string* | 必需。规定要分割的字符串。 |
| *length* | 可选。数字值，定义字符串块的长度。默认是 76。 |
| *end* | 可选。字符串值，定义在每个字符串块末端放置的内容。默认是 \r\n。 |

chunk\_split(*string*,*length*,*end*)

把字符串分割为一连串更小的部分

## Base64\_encode和base64\_decode

# 图像处理

### GD2函数库

Php通过GD2函数库操作图形

文档：

<https://www.php.cn/manual/view/5873.html>

php课件 4

### 画图操作步骤

#### 创建 一个背景图像（画布）

**创建空白的背景**

imageCreate(int x ,int y);

x背景宽度，单位像素

y背景高度，单位像素

**使用已有图形创建背景**

ImageCreateFormPNG(string $fileName);

imageCreateFormJpeg(string $filename);

imageCreateGif(string $filename)

**选择颜色**

imageColorAllocate(int $img,int red, int blue);//rgb的值

#### 在背景上绘制图像或输入文本。

计算文字所占区域

imageTTFBBox(int $size ,int $angle ,string $fontfile ,string $text);

$size 字形尺寸、$angle 字型角度、$fontfile 字体

$text 字符串内容

指定字体输出字符串

imagettftext(int img,int size,int angle,int x,int y,int col,string fontfile,strint text);

#### 输出图像

imagePNG(int $img string $filename);

imageJPEG(int $img string $filename);

imageGIF(int $img string $filename);

#### 释放所有资源

Imagedestory(int $img)

案列：

//创建画布

$im=imagecreate(150,40);

//配置红色背景

ImageColorAllocate($im,255,0,0);

//定义黑色文字

$textcolor= ImageColorAllocate($im,0,0,0);

//水平画一行子字

Imagestring($im,5,30,12,’Hello,world’,$ textcolor);

//输出图像

header(“content-type:image/png”);//设置图片类型

imagepng($img);

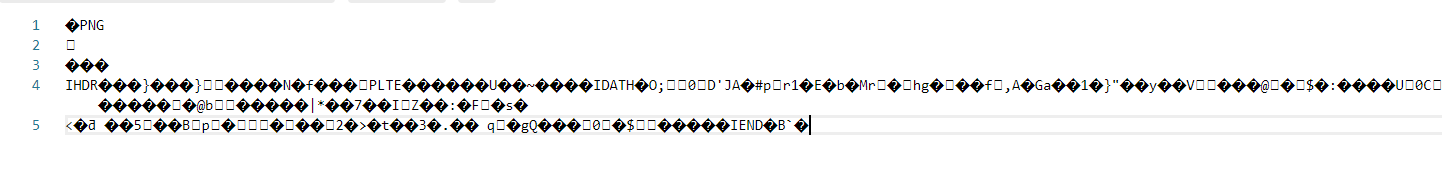
//释放资源

Imagedeatory($im);

## file\_get\_contents()

把整个文件读入一个字符串中

返回的是二进制流



## Imagecreatefromstring()

从字符串的图像流新建一图像

## Imagecopyresampled()

拷贝部分图像并调整大小

bool imagecopyresampled ( resource $dst\_image , resource $src\_image , int $dst\_x , int $dst\_y , int $src\_x , int $src\_y , int $dst\_w , int $dst\_h , int $src\_w , int $src\_h )



*/\*\*  
 \* 将logo嵌套在二维码中  
 \* 参数一：目标图象连接资源  
 \* 参数二：源图象连接资源。  
 \* 参数三：目标 X 坐标点  
 \* 参数四： 目标 Y 坐标点 logo图片嵌入后坐标  
 \* 参数五： 源的 X 坐标点 logo图片嵌入前坐标 相对移动原图X坐标  
 \* 参数六：源的 Y 坐标点  
 \* 参数七：目标宽度  
 \* 参数八：目标高度 logo图片嵌入后高度  
 \* 参数九：源图像宽度 logo图片嵌入前高度  
 \* 参数十：源图像高度  
 \*/***if** ($logo !== **FALSE**) {  
 $QR = *imagecreatefromstring*(*file\_get\_contents*($QR));  
 $logo = *imagecreatefromstring*(*file\_get\_contents*($logo));  
 $QR\_width = *imagesx*($QR);*//二维码图片宽度* $QR\_height = *imagesy*($QR);*//二维码图片高度* $logo\_width = *imagesx*($logo);*//logo图片宽度* $logo\_height = *imagesy*($logo);*//logo图片高度* $logo\_qr\_width = $QR\_width\*0.4; *//logo图片在二维码图片中宽度大小* $scale = $logo\_width/$logo\_qr\_width;  
 $logo\_qr\_height = $logo\_height/$scale; *//logo图片在二维码图片中高度大小  
 // 目标 X 坐标点* $from\_width = ($QR\_width - $logo\_qr\_width)/2;  
 *//重新组合图片并调整大小  
 imagecopyresampled*($QR, $logo, $from\_width, $from\_width, 0, 0, $logo\_qr\_width, $logo\_qr\_height, $logo\_width, $logo\_height);  
 *imagepng*($QR,**'SWE.png'**);  
}

## Imagesx()

获取图像高度

## imagesy()

获取图像宽度