**第二章---PHP语法基础**（黑p16）

2.1.1.php标记

<?php ?>

2.1.2.注释

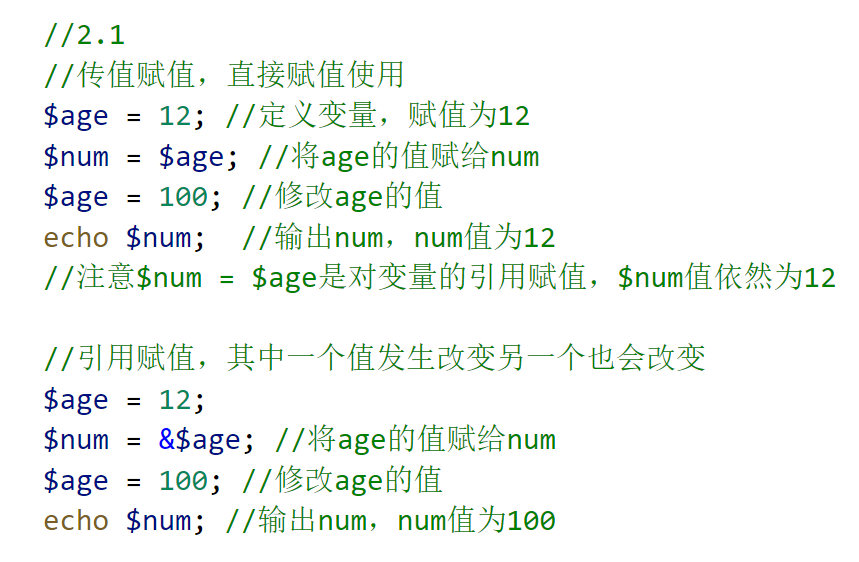
（1）单行注释 //

（2）多行注释 /\* \*/

2.1.3.语句结束符 ；

**2.2.1 变量与常量**（黑p18）

变量是可变的可以保存数据的容器，变量由符号$组成



**注：标识符**

（1）标识符只能由字母、数字和下划线组成。

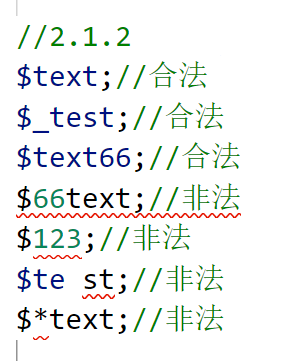
（2）标识符可以由一个或多个字符组成，

必须以字母或下划线开头。

（3）当标识符用作变量名时，区分大小写。

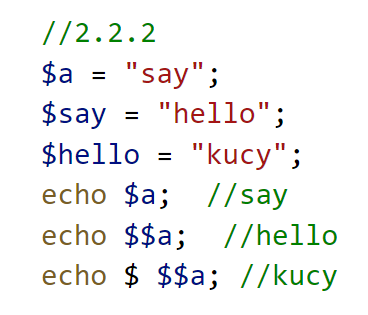
（4）如果标识符由多个单词组成，

那么应使用下划线进行分隔，如：user\_name。



**2.2.2可变变量**（黑p19）

通过可变变量，将另外一个变量的值作为该变量的名称，在变量前多加一个$符号即可



**2.2.3常量**（黑p19）

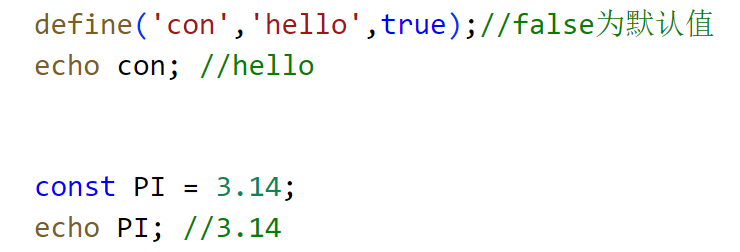
常量为保存在脚本运行中始终保持不变的量，特点为一但被定义就不能被修改或重新定义。

（1）使用define（）定义常量（第三个参数可忽略）

第一个参数表示常量名称，第二个参数表示常量值，

第三个参数，true表示区分大小写，false区分大小写

（2）使用const关键字定义常量



**2.2.4预定义常量**（黑p20）

PHP预先定义好的常量，用来获取PHP中的信息。

|  |  |
| --- | --- |
| **常量名** | **功能描述** |
| PHP\_VERSION | 获取PHP的版本信息，如7.3.19 |
| PHP\_OS | 获取解析PHP的操作系统类型，如：WINNT |
| PHP\_INT\_MAX | 获取PHP中整型数的最大值2147483647 |
| PHP\_INT\_SIZE | 获取PHP中整型数的字长，如：4 |
| E\_ERROR | 表示运行时致命性错误，使用1表示 |
| E\_WARNING | 表示运行时警告错误（非致命），  使用2表示 |
| E\_PARSE | 表示编译时解析错误，使用4表示 |
| E\_NOTICE | 表示运行时提醒信息，使用8表示 |

**注：关键字**

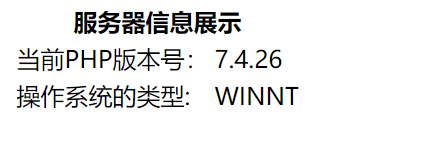
编程语言里预先定义好并赋予了特殊含义的单词，也称作保留字。

**注意：**关键字不能作为常量、函数名或类名使用，关键字不推荐作为变量名使用，容易混淆。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **常见的关键字** | | | | | | |
| if | and | as | break | callable | case | catch |
| class | clone | const | continue | declare | default | echo |
| else | extends | final | finally | for | foreach | function |
| new | abstract | include | namespace | global | or | print |
| use | protected | public | require | return | static | switch |
| throw | trait | try | private | var | while | yield |

🕮案例：显示服务器信息





**2.3.1数据类型**（黑p21）

**1. 布尔型**

用于逻辑判断，只有true和false两个值，

表示“真”和“假”，不区分大小写。

特殊情况下，0表示false ，1表示true

**2.整形**

用于表示整数，由八进制、十进制和十六进制表示，前面加上“+”或“-”符号表示正数或负数。

八进制数：使用0~7表示，且数字前必须加上0。

十六进制数：使用0~9和A~F表示，数字前必须加上0x。

**3. 浮点型**

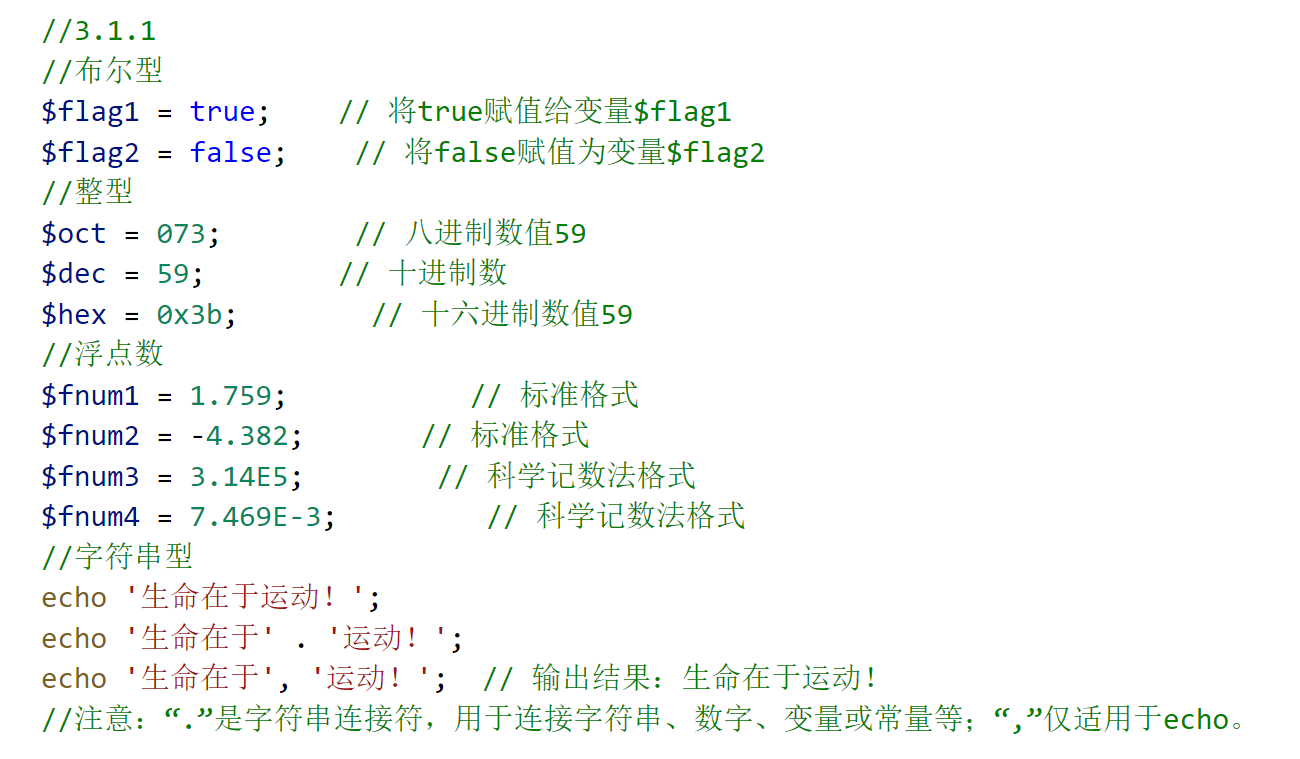
用于保存浮点数或整数，浮点数在程序中用来表示小数。浮点数通常有两种表示方式：标准格式和科学记数法格式。

**4.字符串型**

由连续的字母、数字或字符组成的字符序列。

字符串通常使用单引号或双引号作为定界符。

注意：“.”是字符串连接符，用于连接字符串、数字、变量或常量等；“,”仅适用于echo。

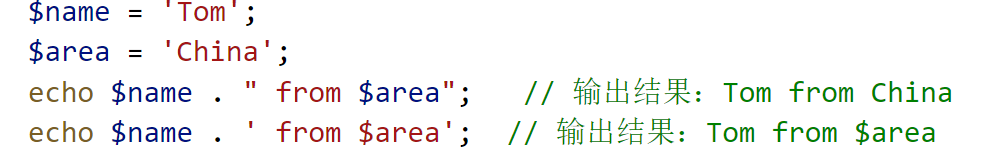


**2.3.2字符串解析**（黑p23）

**1.单引号字符串和双引号字符串的区别**

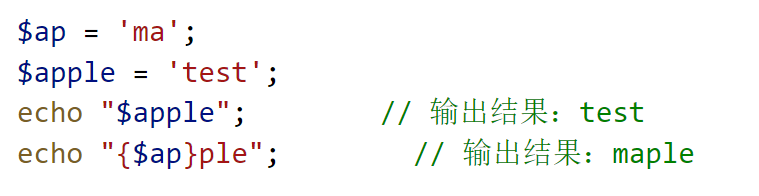
单引号字符串中书写的变量名会原样输出，

双引号字符串中会被解析。



**2. 界定变量**

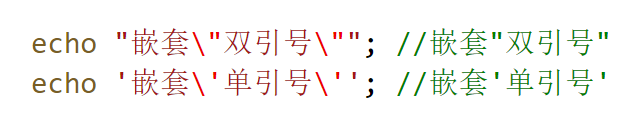
在双引号字符串中，使用“{}”对变量进行界定。



**3.转义字符**

在双引号中：双引号用\” 换行符\n 制表符\t等

单引号只支持使用\’和\\



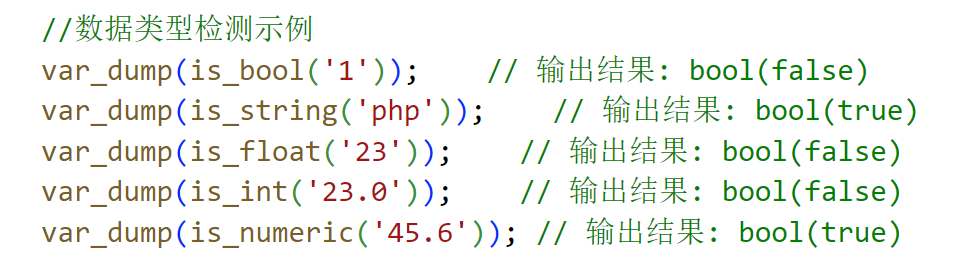
**2.3.3数据类型检测**（黑p23）

检测数据类型：使用内置的is\_\*()函数检测，

如果符合检测的数据类型，返回true，否则返回false。

为了查看数据类型，通常使用var\_dump()来检测

|  |  |
| --- | --- |
| **函数名称** | **功能描述** |
| is\_bool() | 检测是否为布尔类型 |
| is\_string() | 检测是否为字符串类型 |
| is\_float() | 检测是否为浮点类型 |
| is\_int() | 检测是否为整型 |
| is\_null() | 检测是否为空值 |
| is\_array() | 检测是否为数组 |
| is\_resource() | 检测是否为资源 |
| is\_object() | 检测是否为对象类型 |
| is\_numeric() | 检测是否为数字或由数字组成的字符串 |

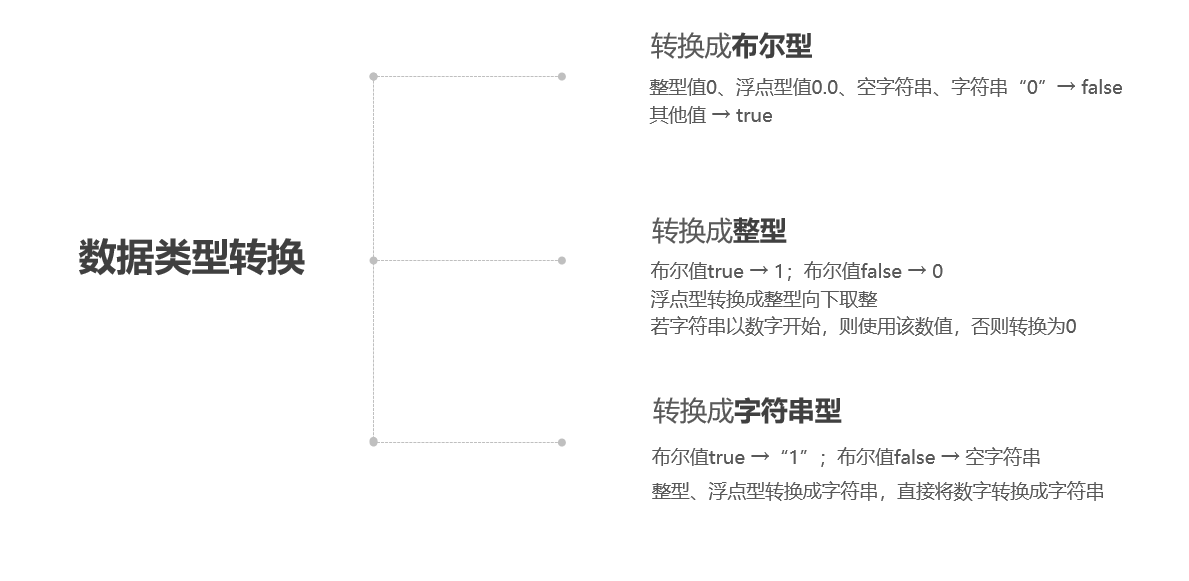


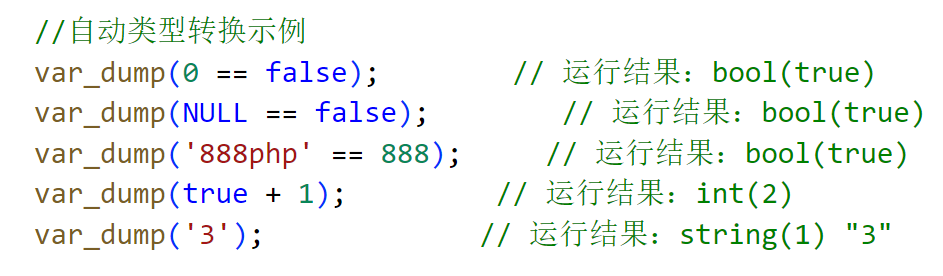
**2.3.4数据类型转换**（黑p24）

**1. 自动类型转换**

当运算与期望的结果类型不匹配时，PHP将自动进行类型转换，无需开发人员做任何操作。

常见的自动类型转换有3种，分别为转换成布尔型、转换成整型和转换成字符串型。

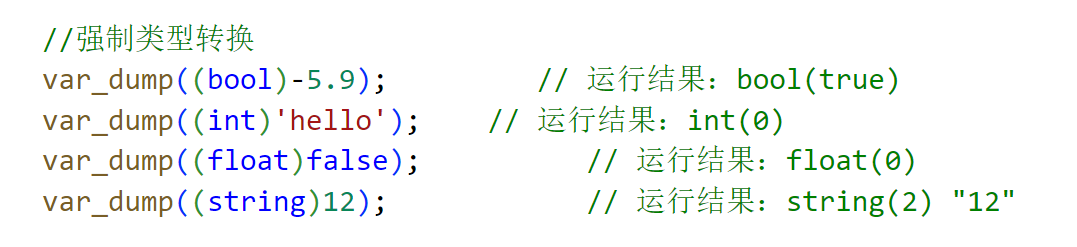




**2.强制类型转换**

在要转换的数据或变量之前加上“(目标类型)”即可

|  |  |  |  |
| --- | --- | --- | --- |
| **强制类型** | **功能描述** | **强制类型** | **功能描述** |
| (boolean)或(bool) | 转为布尔型 | (float) | 转为浮点型 |
| (string) | 转为字符串型 | (array) | 转为数组 |
| (integer)或(int) | 转为整型 | (object) | 转为对象 |

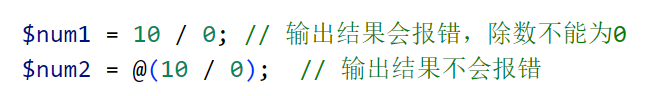


**2.4运算符**（黑p25）

**2.4.1算术运算符**

|  |  |  |  |
| --- | --- | --- | --- |
| 运算符 | 作用 | 范例 | 结果 |
| + | 加 | echo 5 + 5; | 10 |
| - | 减 | echo 6 - 4; | 2 |
| \* | 乘 | echo 3 \* 4; | 12 |
| / | 除 | echo 5 / 5; | 1 |
| % | 取模（即算术中的求余数） | echo 7 % 5; | 2 |

注：**错误抑制运算符**：用来抑制错误的输出，在可能出现错误的代码前加@符号。

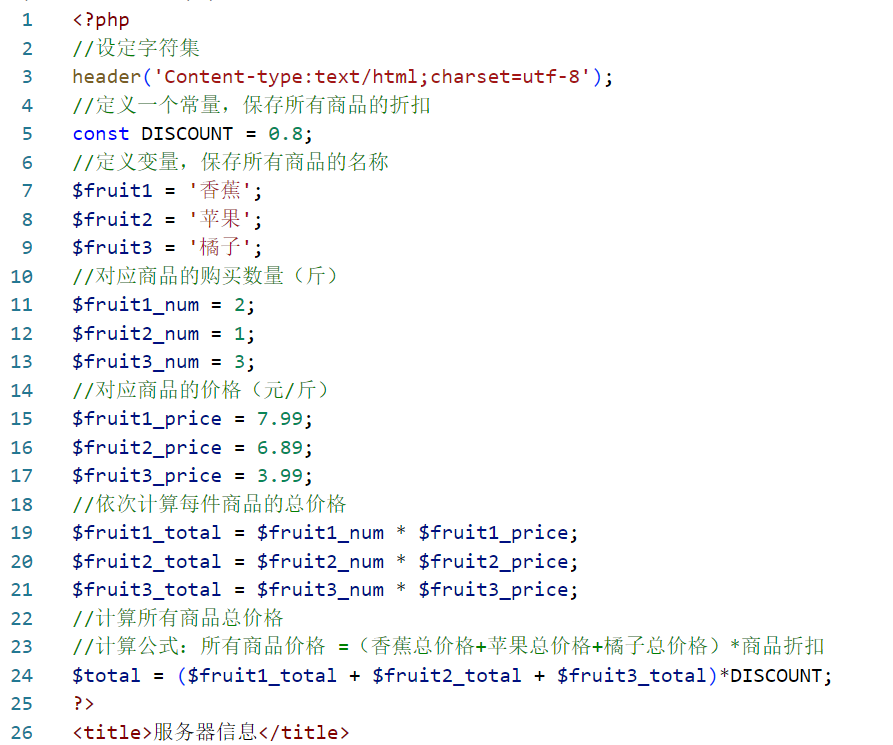


**2.4.2赋值运算符**

|  |  |  |  |
| --- | --- | --- | --- |
| 运算符 | 作用 | 范例 | 结果 |
| + | 加 | echo 5 + 5; | 10 |
| - | 减 | echo 6 - 4; | 2 |
| \* | 乘 | echo 3 \* 4; | 12 |
| / | 除 | echo 5 / 5; | 1 |
| % | 取模  （求余数） | echo 7 % 5; | 2 |
| 运算符 | 作用 | 范例 | 结果 |
| + | 加 | echo 5 + 5; | 10 |

🕮案例：商品价格计算（黑p26）







**2.4.5比较运算符🗹**

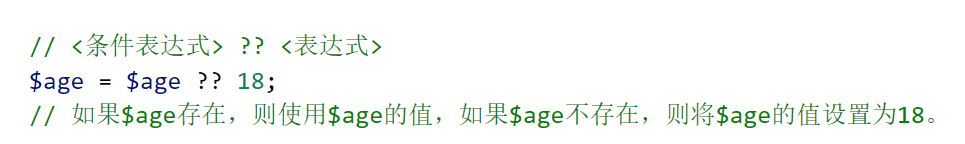
比较两个变量或表达式，

其结果是一个布尔类型的true或false

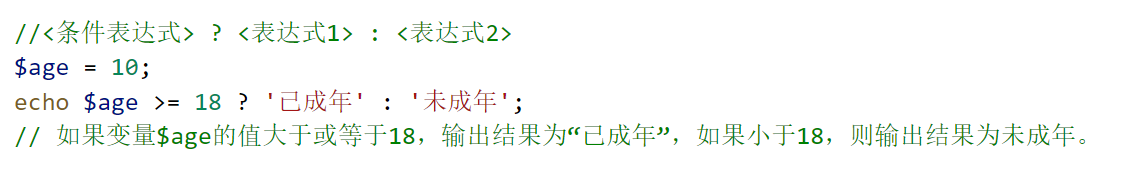
|  |  |  |  |
| --- | --- | --- | --- |
| 运算符 | 运算 | 范例 | 结果 |
| == | 等于 | 5 == 4 | false |
| != | 不等于 | 5 != 4 | true |
| <> | 不等于 | 5 <> 4 | true |
| === | 全等 | 5 === 5 | true |
| !== | 不全等 | 5 !== '5' | true |
| > | 大于 | 5 > 5 | false |
| >= | 大于或等于 | 5 >= 5 | true |
| < | 小于 | 5 < 5 | false |
| <= | 小于或等于 | 5 <= 5 | true |

**2.4.6合并运算符**（黑p29）**🗹**

用于简单的数据存在性判定



**2.4.7三元运算符**



**2.4.8逻辑运算符🗹**

用于逻辑判断的符号，其返回值类型是布尔类型。

|  |  |  |  |
| --- | --- | --- | --- |
| 运算符 | 运算 | 范例 | 结果 |
| && | 与 | $a && $b | $a和$b都为true，  结果为true，否为false |
| || | 或 | $a || $b | $a和$b中  至少有一个为true，  结果为true，否为false |
| ! | 非 | !$a | 若$a为false，  结果为true，否则相反 |
| xor | 异或 | $a xor $b | $a和$b一个为true，一个为false，  结果true，否则false |
| and | 与 | $a and $b | 与“&&”相同，  但优先级较低 |
| or | 或 | $a or $b | 与“||”相同，  但优先级较低 |

**2.4.9递增递减运算符**

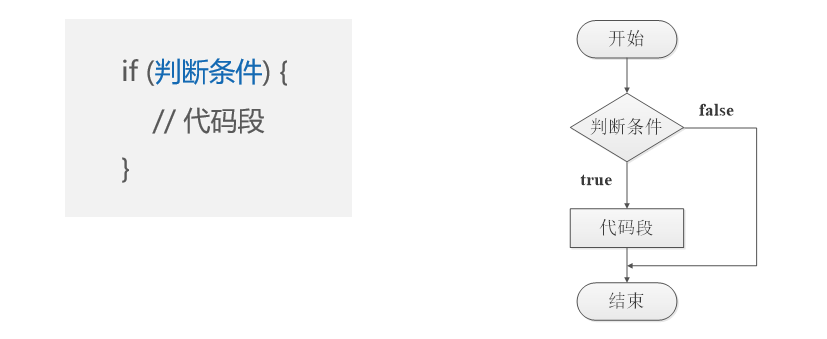
也称为自增自减运算符，是一种特定形式的复合赋值运算符。

|  |  |  |  |
| --- | --- | --- | --- |
| 运算符 | 运算 | 范例 | 结果 |
| ++ | 自增（前） | $a = 2;  $b = ++$a; | $a = 3;  $b = 3; |
| ++ | 自增（后） | $a = 2;  $b = $a++; | $a = 3;  $b = 2; |
| -- | 自减（前） | $a = 2;  $b = --$a; | $a = 1;  $b = 1; |
| -- | 自减（后） | $a = 2;  $b = $a--; | $a = 1;  $b = 2; |

**2.5流程控制**（黑p31）

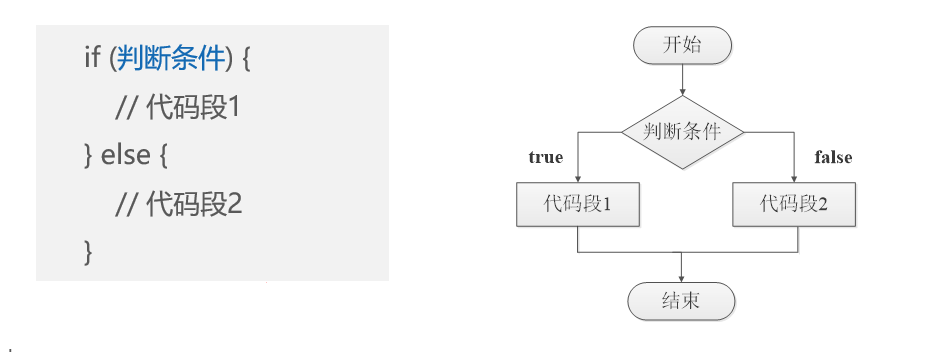
**1.if语句**

也称为单分支语句，用于判断当满足某种条件时就进行某种处理。



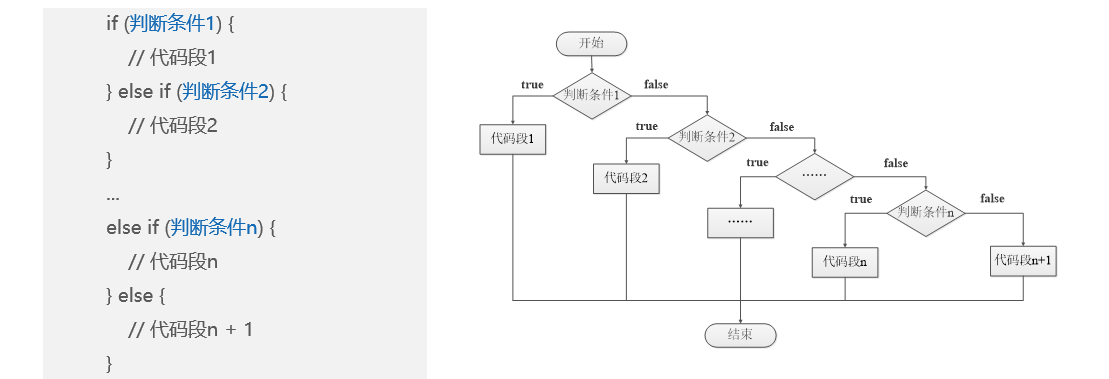
**2. if…else语句**

也称为双分支语句，当满足某种条件时就进行某种处理，否则进行另一种处理。



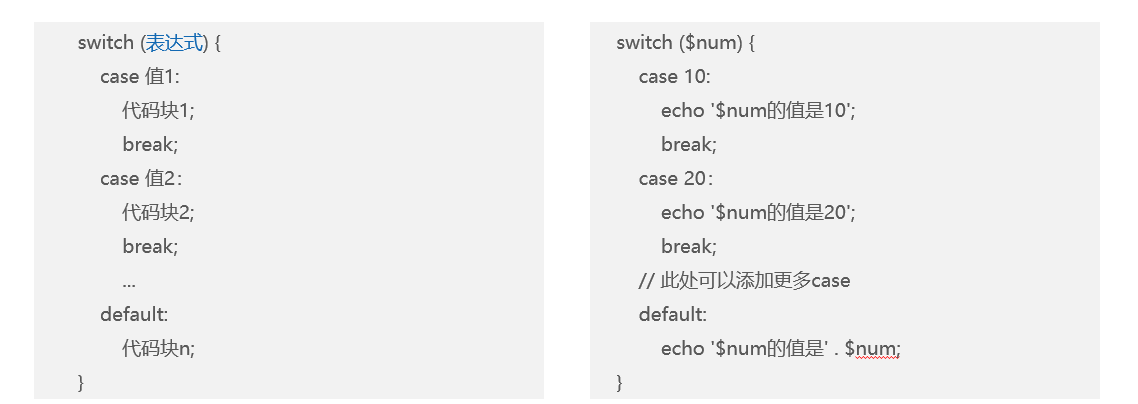
**3. if…else if…else语句**

也称为多分支语句，用于对多种条件进行判断，并进行不同处理



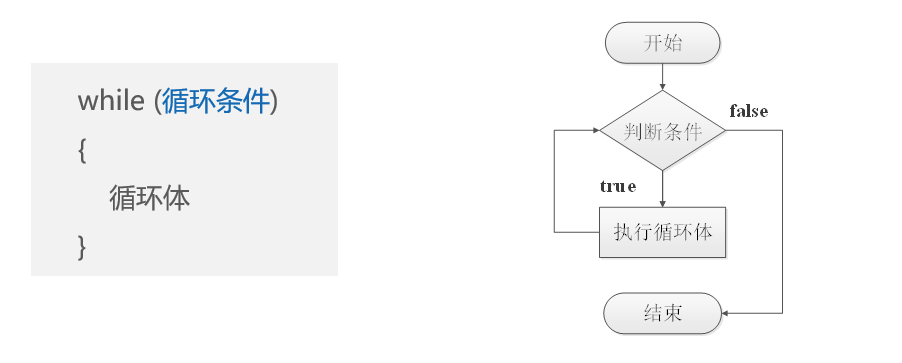
**4. switch语句**

也是多分支语句，使代码更加清晰简洁、便于阅读



**5. while语句**（黑p35）

根据循环条件来判断是否重复执行某一段代码，先判断条件后执行循环体



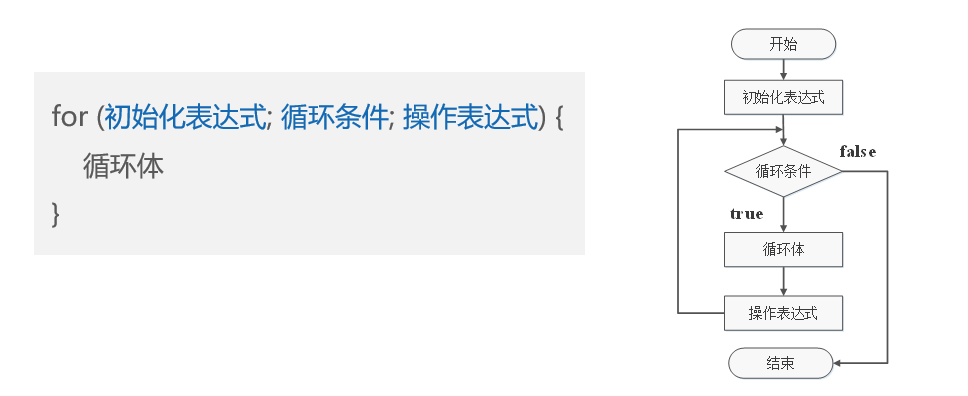
**6. do...while语句**

先无条件执行一次循环体后再判断条件。



**7. for循环语句**

是最常用的循环语句，适合循环次数已知的情况



**8.跳转语句**

用于实现循环执行过程中程序流程的跳转。

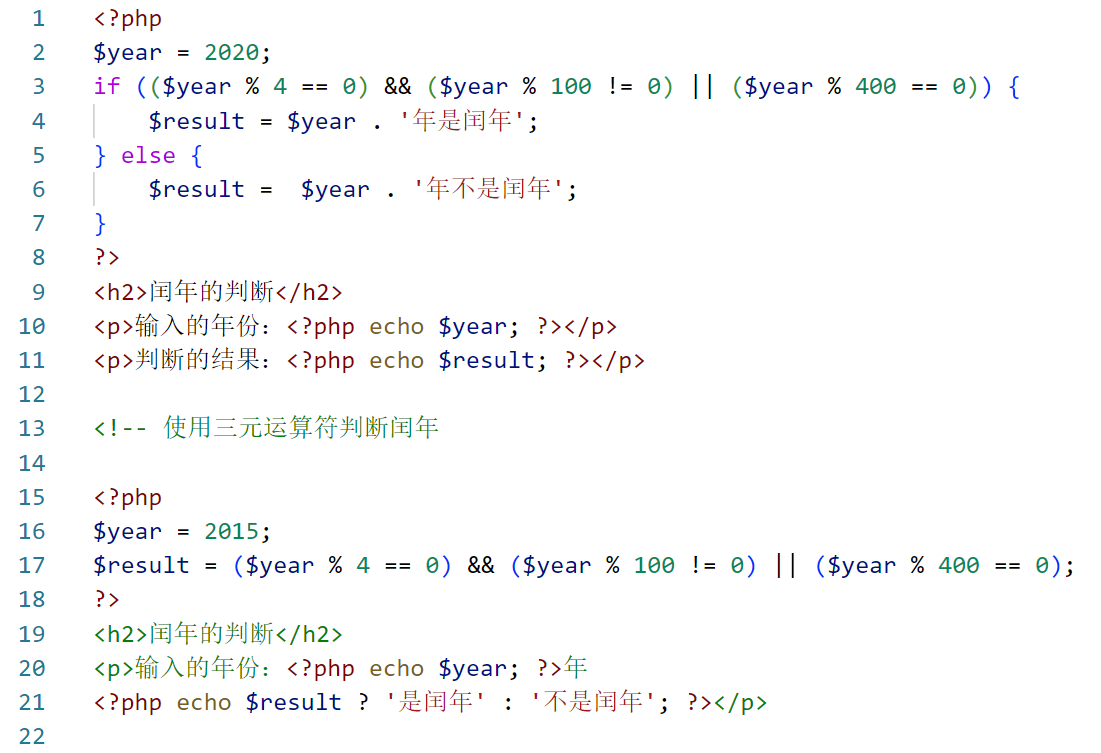
常用的跳转语句：break语句和continue语句。

break用于终止当前循环，跳出循环体。

continue用于结束本次循环的执行，开始下一轮循环的执行

🕮案例：判断润年（黑p34）



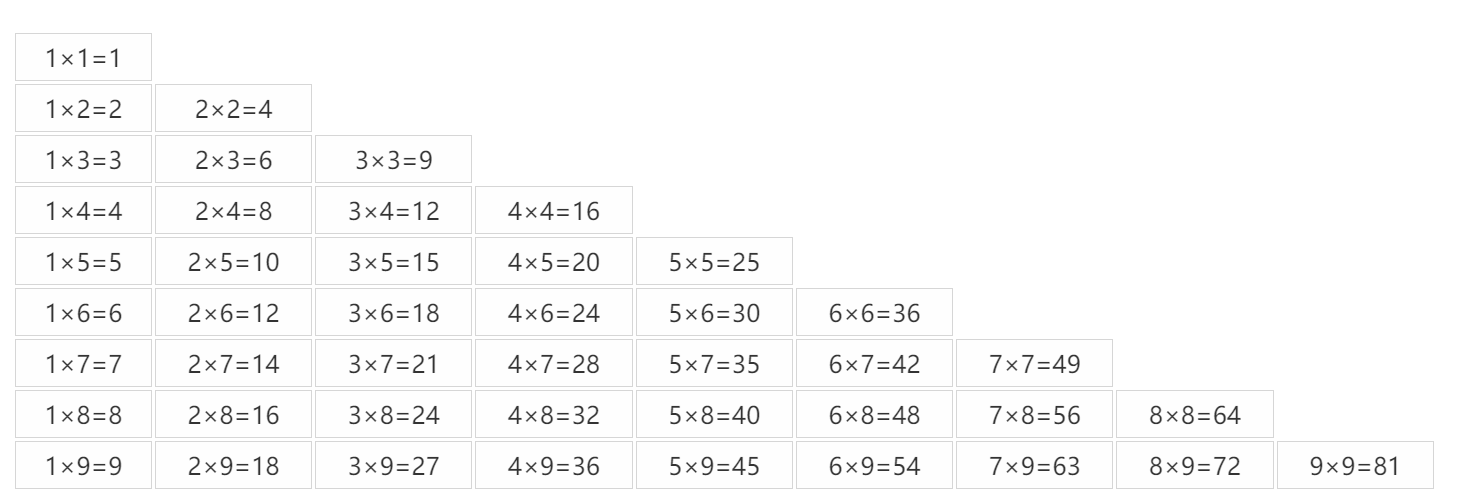


🕮案例：金字塔循环（黑p37）



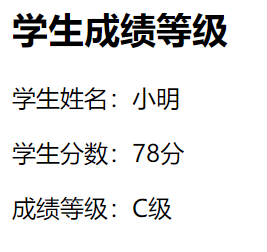


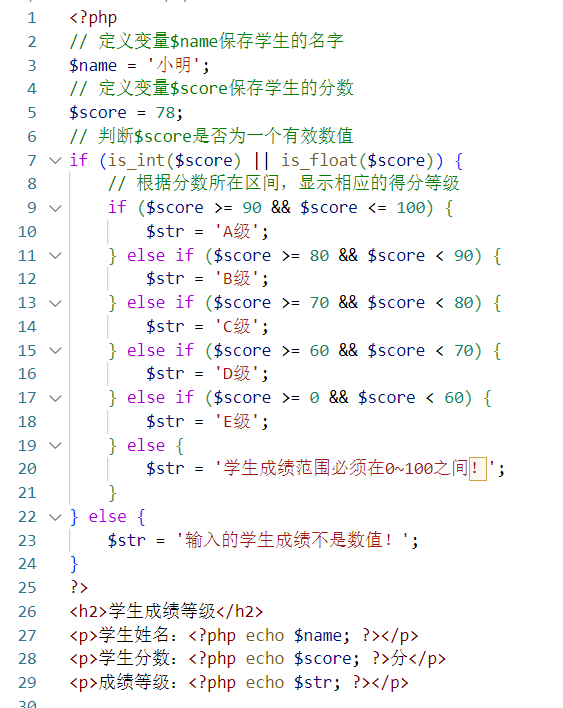
🕮案例：九九乘法表（黑p38）





🕮案例：判断学生成绩等级（黑p33）





**3.1函数**（黑p42）

**静态变量：**使用static关键字在函数中声明，函数执行完成后保留局部变量的值。(相当于javascript中的let)

**3.1.6字符串函数**（黑p44）

是PHP的内置函数，用于操作字符串

|  |  |
| --- | --- |
| **函数名称** | **功能描述** |
| strlen() | 获取字符串的长度 |
| strpos() | 在指定字符串中查找目标字符串  首次出现的位置 |
| strrpos() | 获取指定字符串在目标字符串中  最后一次出现的位置 |
| str\_replace() | 用于对字符串中  的某些字符进行替换操作 |
| substr() | 用于获取字符串中的子串 |
| explode() | 将指定字符串拆分成数组 |
| implode() | 用指定的分隔符将数组中的键值  拼接成字符串 |
| trim() | 去除字符串首尾处的空白字符  （或指定的字符串） |
| str\_repeat() | 重复字符串 |
| strcmp() | 比较两个字符串的大小 |

**3.1.7数学函数**

PHP的内置函数，方便处理程序中的数学运算

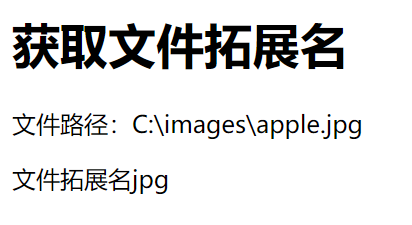
|  |  |
| --- | --- |
| **函数名** | **功能描述** |
| abs() | 绝对值 |
| ceil() | 向上取最接近的整数 |
| floor() | 向下取最接近的整数 |
| fmod() | 返回除法的浮点数余数 |
| is\_nan() | 判断是否为合法数值 |
| max() | 返回最大值 |
| min() | 返回最小值 |
| pi() | 返回圆周率的值 |
| pow() | 返回x的y次方 |
| sqrt() | 返回一个数的平方根 |
| round() | 对浮点数进行四舍五入 |
| rand() | 返回随机整数 |

**3.1.8时间和日期函数**

PHP的内置函数，用于处理日期和时间

|  |  |
| --- | --- |
| **函数名** | **功能描述** |
| time() | 获取当前的UNIX时间戳 |
| date() | 格式化一个本地时间/日期 |
| mktime() | 获取指定日期的UNIX时间戳 |
| strtotime() | 将字符串转化成UNIX时间戳 |
| microtime() | 获取当前UNIX时间戳和微秒数 |

🕮案例：获取文件扩展名（黑p46）

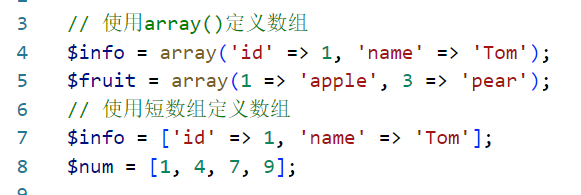




**3.2数组**（黑p48）

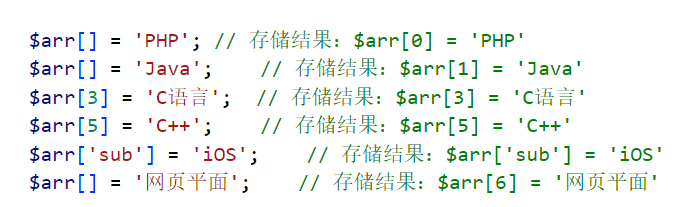
**1.定义数组🗹**

定义数组的两种方式：array()语言结构、短数组定义法。



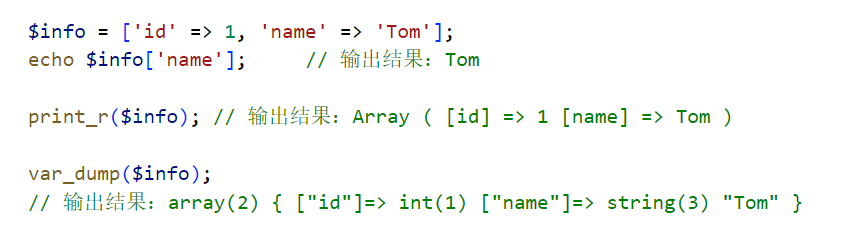
**2.新增数组元素**

创建一个数组变量，使用赋值运算符直接给变量赋值



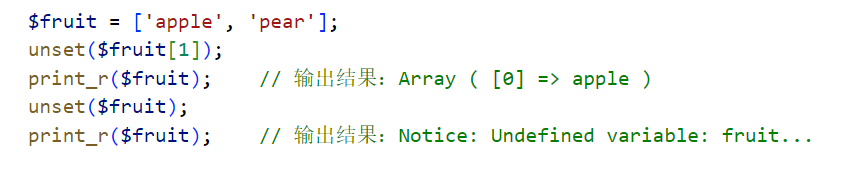
**3.访问数组🗹**

通过数组的键来访问数组元素



**4.删除数组**

使用unset语言结构可以删除数组中的某个元素，也可以删除整个数组。



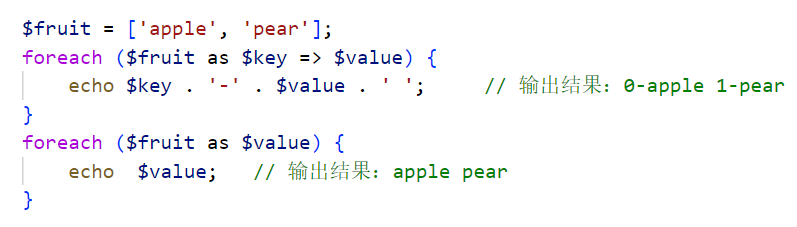
**5.判断数组元素是否存在**

使用isset()语言结构判断数组中的元素是否存在，

返回结果为布尔值。

**3.2.3数组遍历**

依次访问数组中的每个元素，通常使用foreach语句



注：php中使用“.”来拼接字符串

**3.2.4数组和字符串之间的转换**

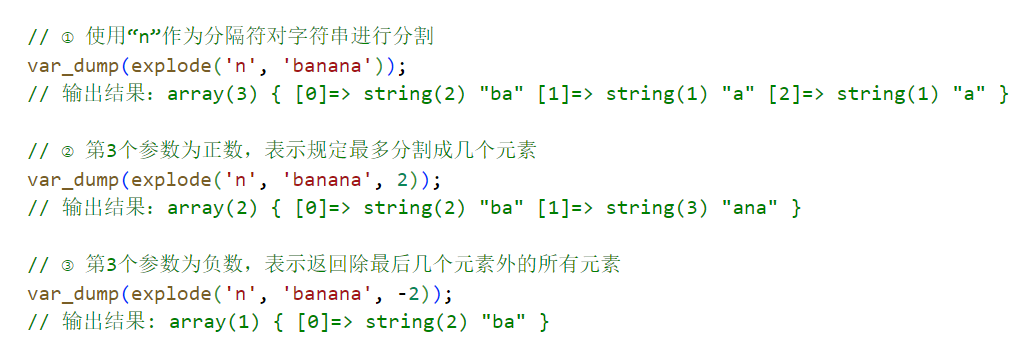
**1.explode()函数**

使用字符串分割另一个字符串，返回数组。

第1个参数是分隔符，

第2个参数是要分割的字符串，

第3个是可选参数，表示返回结果最多包含的元素个数，当为负数m时，返回除最后m个元素外的所有元素，当为0时，则把它当做1处理。

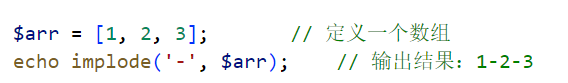


**2.implode()函数**

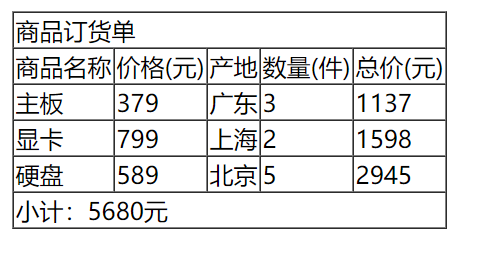
通过指定的连接符将数组拼接成一个字符串，

第1个参数是连接符，

第2个参数是待处理的数组。



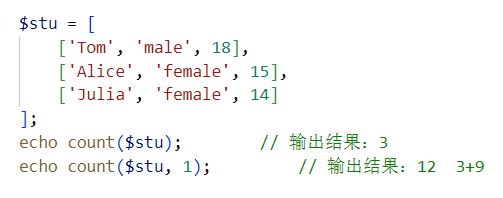
🕮案例：订货单（黑p50）



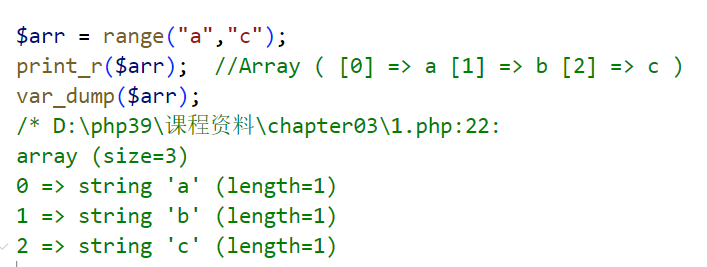


**3.3常用数组函数**

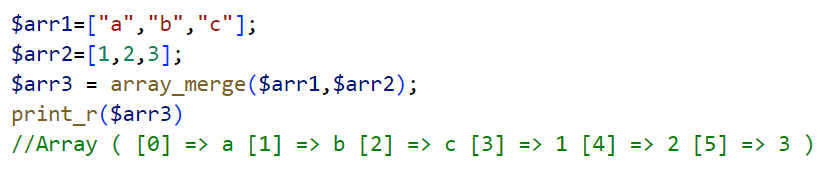
**1.count()函数** 用于统计数组元素个数、合并等操作



**2.range()函数** 建立一个包含指定范围元素的数组



**3.array\_merge()** 合并一个或多个数组

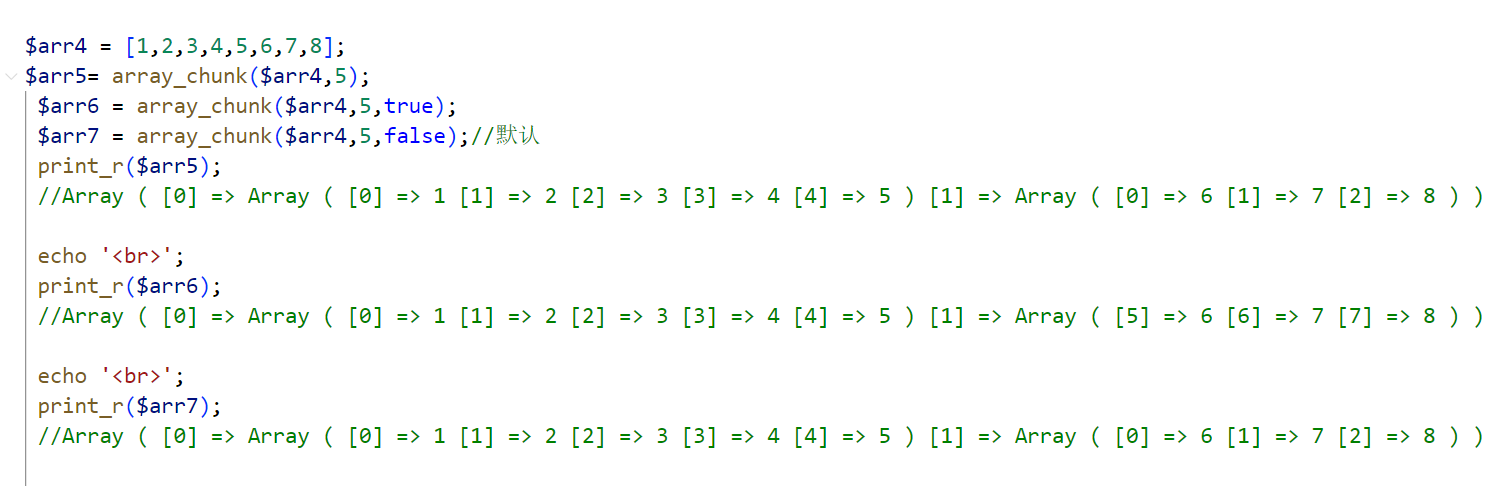


**4.array\_chunk()** 将一个数组分割成多个数组

第一个参数为分隔数组名

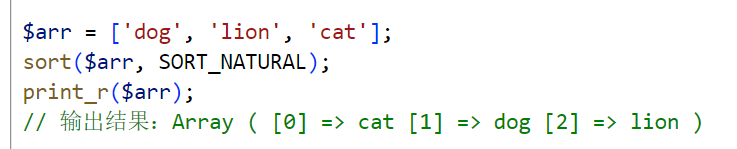
第二个参数为指定分隔后数组元素的个数

第三个为决定分割后的数组是否保留原来的键名



**3.3.2数组排序函数**（黑p52）

|  |  |
| --- | --- |
| **函数名** | **功能描述** |
| sort() | 对数组排序 |
| rsort() | 对数组逆向排序 |
| ksort() | 对数组按照键名排序 |
| krsort() | 对数组按照键名逆向排序 |
| asort() | 对数组进行排序并保持索引关系 |
| arsort() | 对数组进行逆向排序并保持索引关系 |
| shuffle() | 打乱数组顺序 |
| array\_reverse() | 返回一个单元顺序相反的数组 |



**3.3.3数组检索函数**

用于对数组中的键、值进行查找、获取

|  |  |
| --- | --- |
| **函数名** | **功能描述** |
| array\_search() | 在数组中搜索给定的值 |
| array\_unique () | 移除数组中重复的值 |
| array\_column () | 返回数组中指定的一列 |
| array\_keys () | 返回数组中键名 |
| array\_rand () | 从数组中随机取出一个或多个单元 |
| key () | 从关联数组中取得键名 |
| in\_array () | 检查数组中是否存在某个值 |
| array\_values () | 返回数组中所有的值 |

🕮案例：双色球（黑p53）





**3.4文件包含语句🗹**

**include、require、include\_once和require\_once**

**// 第一种写法：**

**include '完整路径文件名';**

**// 第二种写法：**

**include('完整路径文件名');**

**注：require语句与include语句的区别：**

文件未找到：include语句发生警告信息，程序继续运行；

require语句发生致命错误，程序停止运行。

**注：**被包含文件的加载时间：

程序执行到include语句才会调用，require语句在程序开始执行时则立即调用。

**3.5替代语法**（黑p54）

在HTML模板中嵌入PHP代码时，

将一些语句替换成可读性更好的语法。

把if、while、for、foreach 和switch的

左花括号“{”换成冒号“:”

把右花括号“}”分别换成“endif;”“endwhile;”“endfor;”“endforeach;”和“endswitch;”。

**var\_dump()**

函数显示关于一个或多个表达式的结构信息，

包括表达式的类型与值

**print\_r()**

函数用于打印变量，以更容易理解的形式展示

**sset()**

函数用于检测变量是否已设置并且非 NULL。

指定变量存在且不为 NULL，则返回 TRUE，否则返回 FALSE。测试一个被设置成 NULL 的变量，将返回 FALSE。

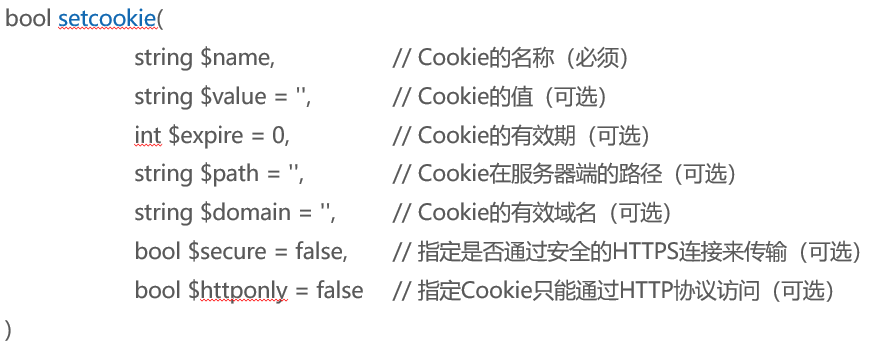
**4.3会话控制**

**4.3.1 cookie：**

记录用户信息，行为，保存在浏览器里面，

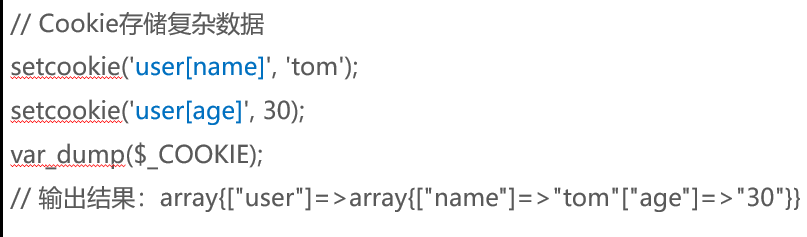
保存本地，自己的电脑里面的。明文的形式。

**创建Cookie：**使用setcookie()函数创建或修改Cookie



**读取Cookie：**使用超全局变量$\_COOKIE读取Cookie

**Cookie存储多个值**：在Cookie名后添加“[]”进行设置

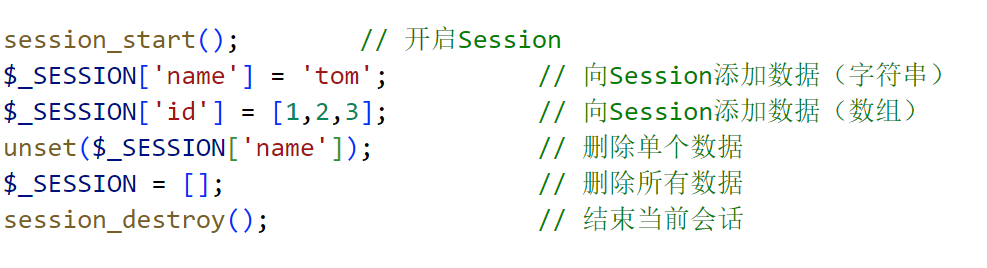


**Cookie高级使用：**通过第3个参数设置Cookie的生命周期，通过第4个参数设置Cookie的访问路径，通过第5个参数设置访问网站。



**4.3.4 session 🗹**

用于身份标识，控制页面访问。保存在服务器上，一般的话无法直接访问



**4.4.1开启GD拓展**

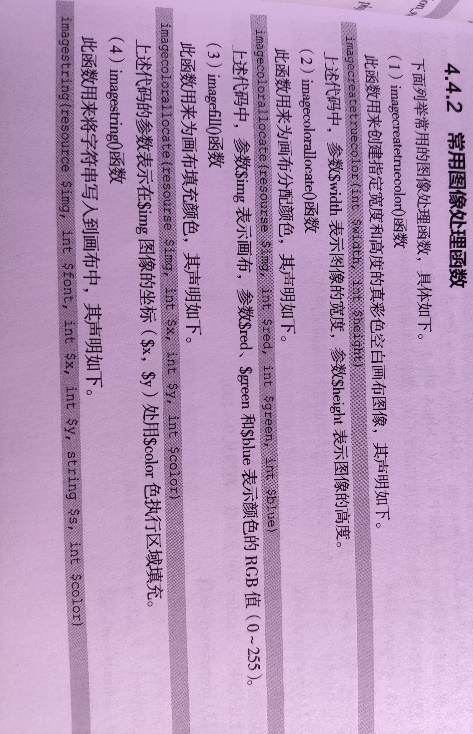
在PHP的配置文件php.ini，找到

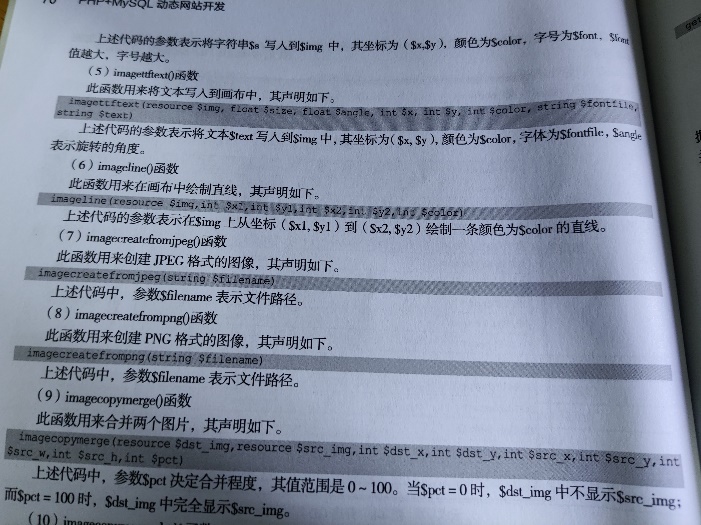
“;extension=gd2”，去掉分号“;”。

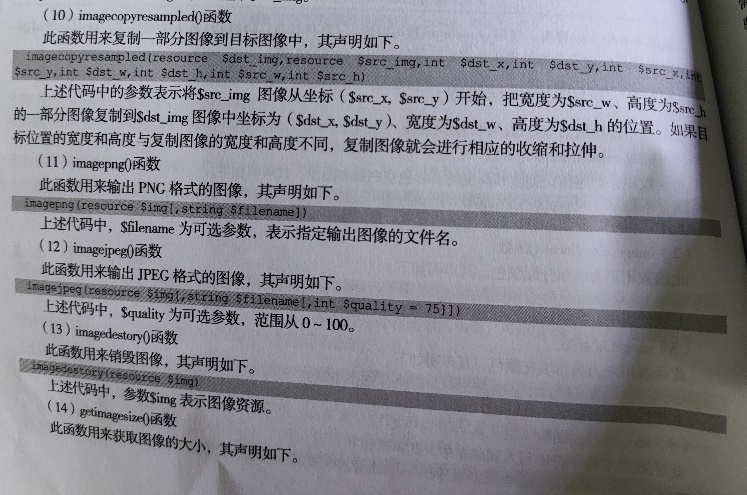
**查看开启结果：**重启Apache使配置生效，通过

phpinfo()函数查看GD库是否开启成功。

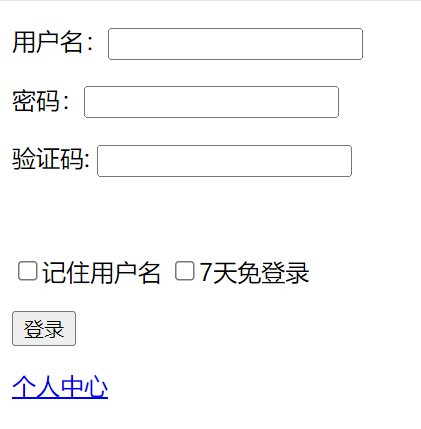
|  |  |
| --- | --- |
| **函数** | **作用** |
| imagecreatetruecolor() | 创建指定宽高的真彩色空白画布图像 |
| imagecolorallocate() | 为画布分配颜色 |
| imagefill() | 为画布填充颜色 |
| imagestring() | 将字符串写入到画布中 |
| imagettftext() | 将文本写入到画布中 |
| imageline() | 在画布中绘制直线 |
| imagecreatefromjpeg() | 创建JPEG格式的图像 |
| imagecreatefrompng() | 创建PNG格式的图像 |
| imagecopymerge() | 合并两个图片 |
| imagecopyresampled() | 复制一部分图像到目标图像中 |
| imagepng() | 输出PNG格式的图像 |
| imagejpeg() | 输出JPEG格式的图像 |
| imagedestory() | 销毁图像 |
| getimagesize() | 获取图像的大小 |



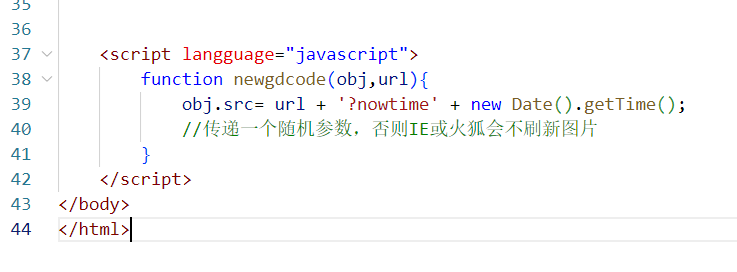




⌘案例：模拟登录



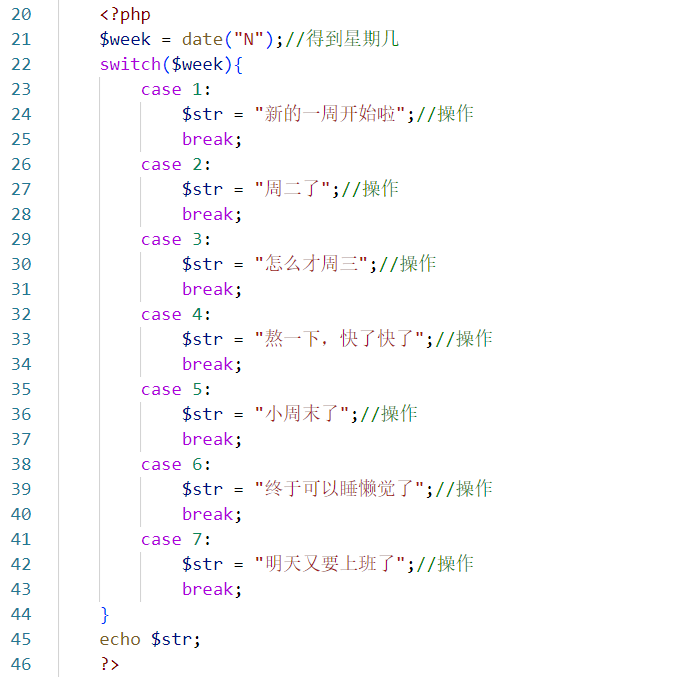


















一. PHP与Web页面数据交互P132

1.ajax + json。

2.表单：POST，把数据打包之后发送出去，数据量大，安全性高

3.附加URL来传输数据。GET，把数据以明文的方式附加到网址上来发送，数据量小，不安全

⌘案例：模拟注册功能





二. PHP会话控制P191

a)session：用于身份标识，控制页面访问。保存在服务器上，一般的话无法直接访问。

b)cookie：记录用户信息，行为，保存在浏览器里面，保存本地，自己的电脑里面的。明文的形式。

🕮案例：模拟登录



