# PHP 基础语法加强

目录

[PHP 基础语法加强 1](#_Toc20203)

[1. 语法结构 2](#_Toc22179)

[1.1 标示符 2](#_Toc6687)

[1.2注释 2](#_Toc17716)

[1.3php和html混编 2](#_Toc4875)

[2. 变量 3](#_Toc27952)

[2.1定义变量并赋值 3](#_Toc16629)

[2.2变量的命名规则 4](#_Toc22396)

[2.3常用函数 4](#_Toc29972)

[2.4值传递和引用传递（赋值的方式） 5](#_Toc29875)

[2.5可变变量 6](#_Toc14549)

[2.6变量作用域 7](#_Toc32477)

[3. 预定义变量/数组 8](#_Toc8999)

[$GLOBALS 8](#_Toc2583)

[$\_GET 9](#_Toc29967)

[$\_POST 9](#_Toc17212)

[$\_REQUEST 9](#_Toc8228)

[$\_SERVER 10](#_Toc5389)

[$\_FILES 10](#_Toc11680)

[4. 常量 11](#_Toc28526)

[4.1定义常量并赋值 11](#_Toc29470)

[4.2魔术常量 12](#_Toc2479)

[4.3预定义常量 12](#_Toc10827)

[5. 数据类型 13](#_Toc32763)

[布尔型（boolean） 13](#_Toc27405)

[整型(int) 14](#_Toc28003)

[浮点型（float） 15](#_Toc17388)

[字符串类型(string) 16](#_Toc28450)

[资源： 18](#_Toc19139)

[Null(空) 18](#_Toc8963)

[6. 运算符 19](#_Toc4104)

[算数运算符 19](#_Toc541)

[赋值运算符 20](#_Toc28962)

[比较运算符 20](#_Toc16955)

[错误运算符 21](#_Toc27304)

[递增和递减运算符 22](#_Toc31654)

[逻辑运算符 22](#_Toc20875)

[三元运算符 23](#_Toc32306)

[数组运算符 24](#_Toc21096)

[位运算符 25](#_Toc22457)

[常用的一些函数 26](#_Toc20633)

[7. 流程控制 26](#_Toc24428)

[7.1顺序执行：平时说的代码的执行顺序—从上到下 26](#_Toc19147)

[7.2分支结构 26](#_Toc29852)

[7.3循环结构 31](#_Toc22800)

[7.4 switch分支结构 36](#_Toc21743)

[7.5 break 37](#_Toc14545)

[7.6 continue 37](#_Toc19741)

[7.7 goto 38](#_Toc15632)

[7.8 文件的引入 39](#_Toc24740)

[8. Php可以在cmd中运行代码 40](#_Toc28108)

## 语法结构

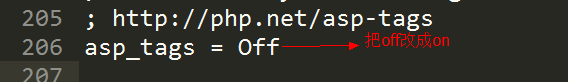
### 1.1 标示符

#### 标准标示符:<?php code; ?> 推荐使用,如果php代码后面没有其他代码结束符（？>）可以省略

#### 短标签标示符:<? Code;?>不推荐使用，需要php开启这个功能，才可以使用



#### ASP标示符:<% code; %>不推荐使用，需要开启

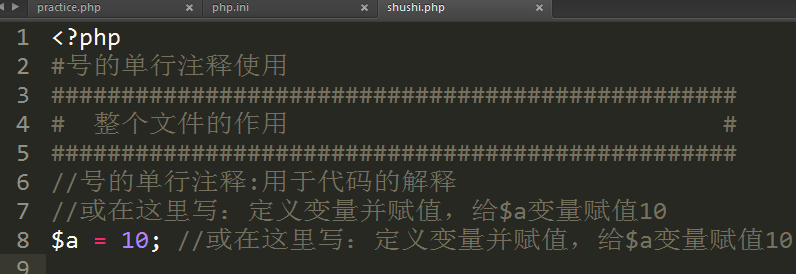


#### Script标示符:<scrtip language=”php”>code;</script>不推荐

### 1.2注释

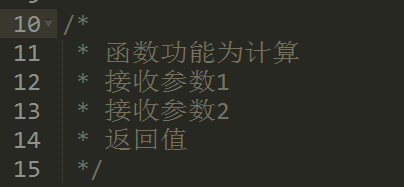
#### 单行注释 # //

说明：一般用于代码的上面或后面写注释



#### 多行注释 /\*注释\*/

说明：一般用于整个函数或类或方法的说明；



### 1.3php和html混编

1.html的代码可以嵌入到php的文件中

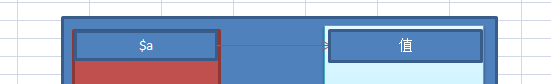
2.php代码可以嵌入到html的代码中



## 变量

### 2.1定义变量并赋值

* 变量：可以存储变化的量（有东西的）,值是不固定的，可以改变的



* 定义变量：用$符号后面跟变量的名，定义时可以不赋值，但是不能使用



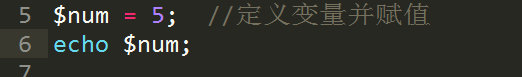
* 定义变量并赋值：$变量名 = 表达式；把表达式的值存储到变量名所对应的位置

如：$name = job;



* 表达式：表达式是php的重要的基石，所有有值的东西都可以理解为“表达式”

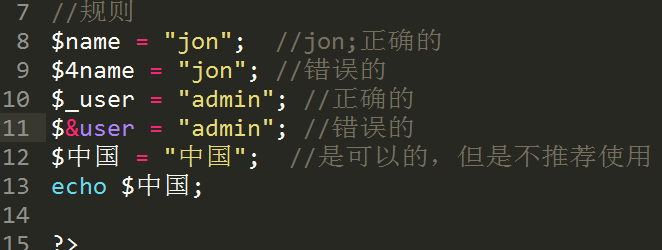
如：$num = 5;

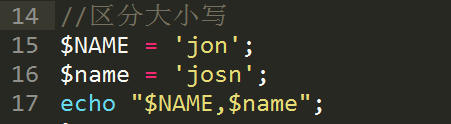


### 2.2变量的命名规则

* 说明：1.必须是数字、字母、下划线组成，且不能以数字开头的命名规则

2.变量名是区分大小写的





* 驼峰式命名法：第一单词首字母小写，其他单词首字母大写

$userName;$passWrod;$getNum;

* 帕斯卡命名法：每个单词首字母大写

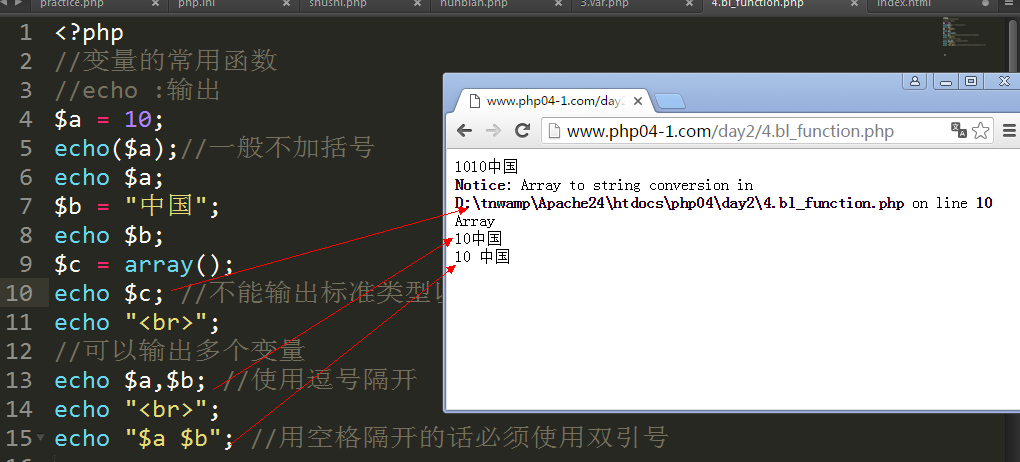
$UserName;$PassWrod;$GetName

* 下划线命名法：每个单词之间用下划线区分

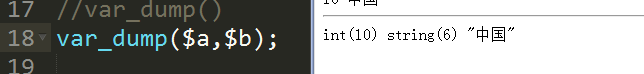
$user\_name;$pass\_word;$get\_action;

### 2.3常用函数

* Echo()：输出必须是标准类型（bool,int,float,string）



* Var\_dump()：输出类型，长度，值

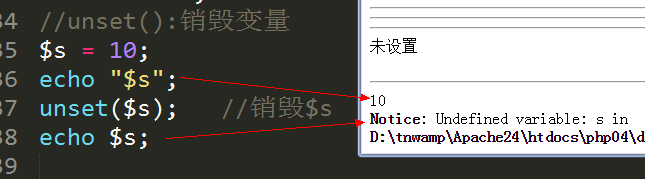


* Isset()：判断变量是否设置（存在）

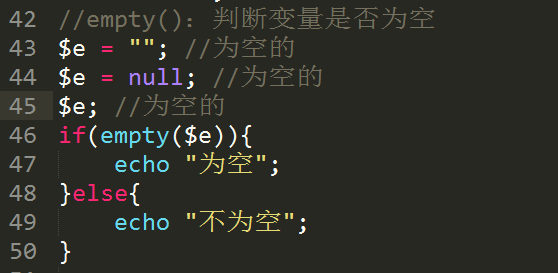
必须赋值才是存在的，null也是未设置



* Unset()：销毁变量



* Empty()：判断变量是否为空

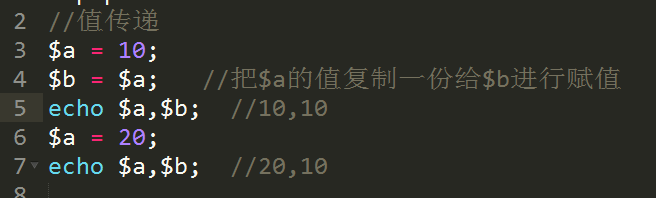


### 2.4值传递和引用传递（赋值的方式）

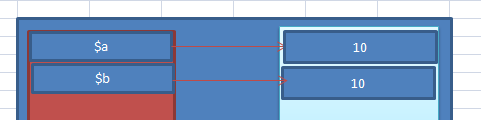
#### 值传递

* 说明：普通的定义变量并赋值，就值传递

代码说明：

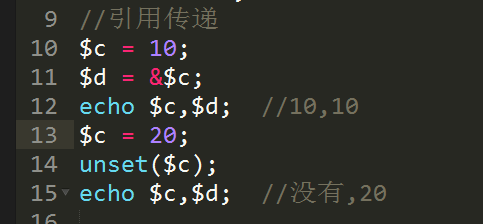


复制后赋值，变成两个不相关的变量

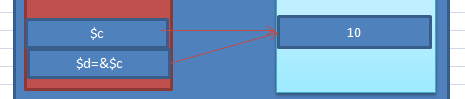


#### 引用传递

* + - 说明：用**&**符号进行引用的使用，一旦使用引用传递两个变量是关系的
    - 代码说明：

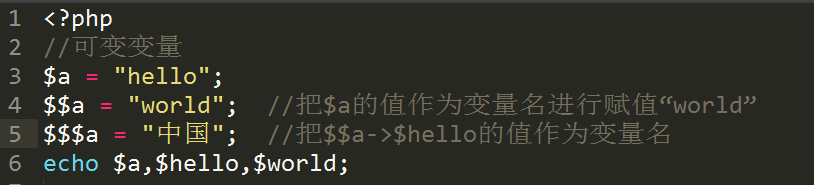


* + - 画图



### 2.5可变变量

* + - 说明：把一个变量的值作为另一个变量的名,用$符号实现
    - 代码



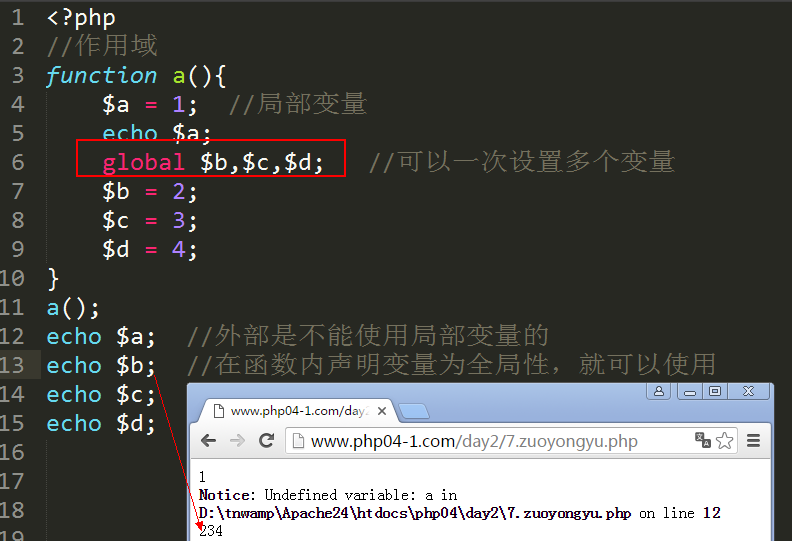
* 减少当前页面的变量的定义

### 2.6变量作用域

#### 局部变量:在函数中定义的变量就是局部变量，只能在函数内使用

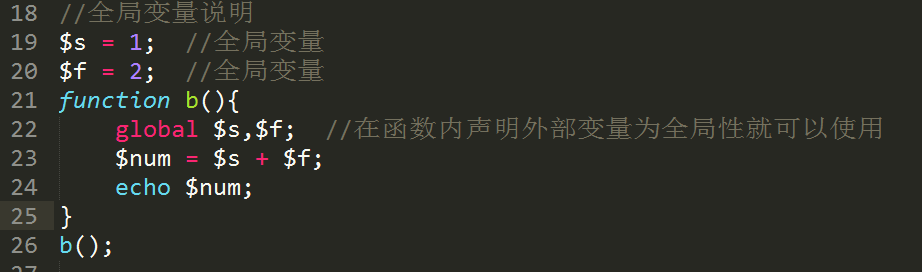


* 想在函数外使用：可以使用全局定义关键词global,设置变量为全局性



#### 全局变量:在函数外定义的变量就是全局变量，只能在函数外使用

* 函数内使用函数外的变量



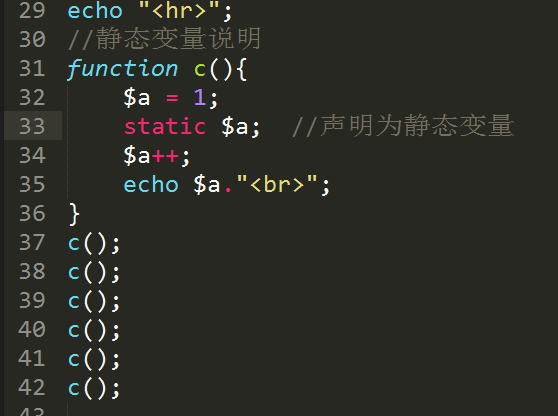
#### 静态变量

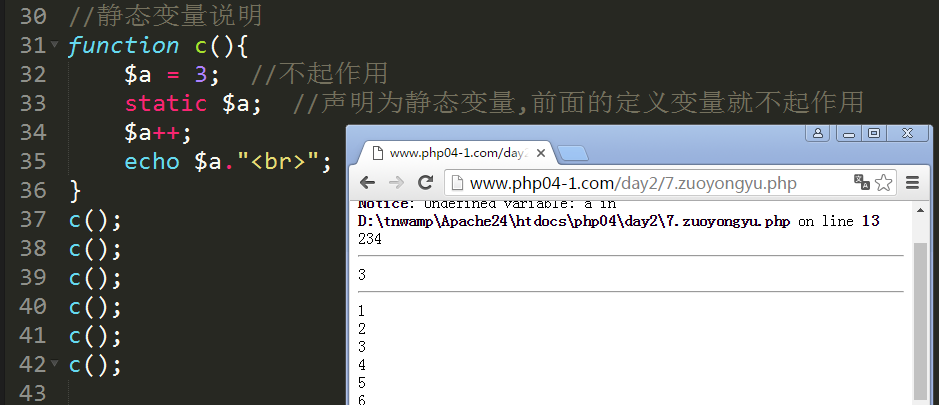
说明：一般用于函数内，可以理解为是局部变量,一旦定义为静态变量，变量是不会销毁的

定义方法是：static 变量；

作用：php自动销毁变量,有的时候不需要销毁，需要累加

**一般递归中都使用静态变量作为退出递归的条件**





## 预定义变量/数组

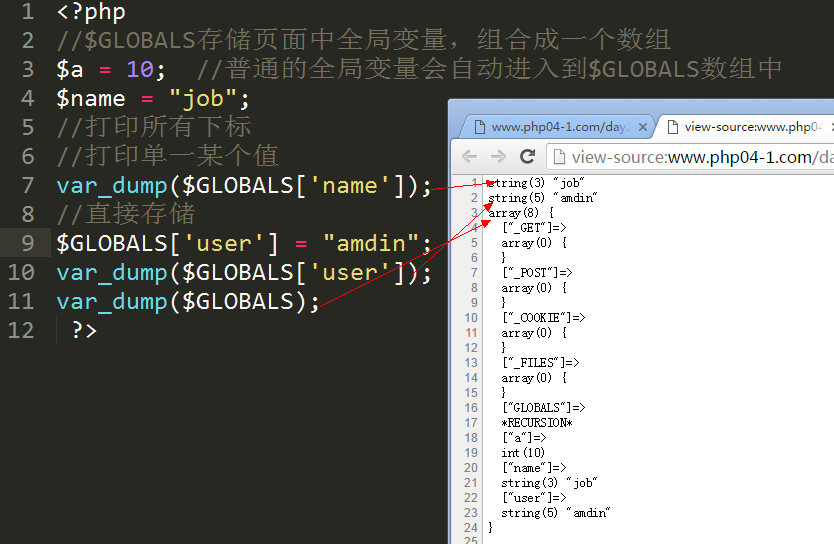
说明：系统自带的一下数组变量



### $GLOBALS

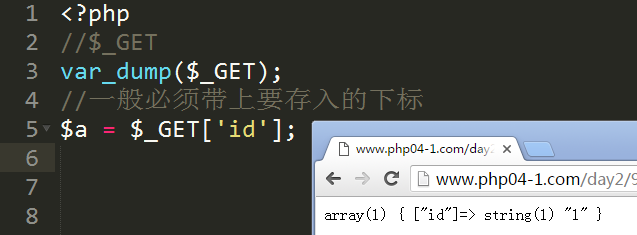
说明：就是当前页面中的所有全局变量的数组组合

代码说明：



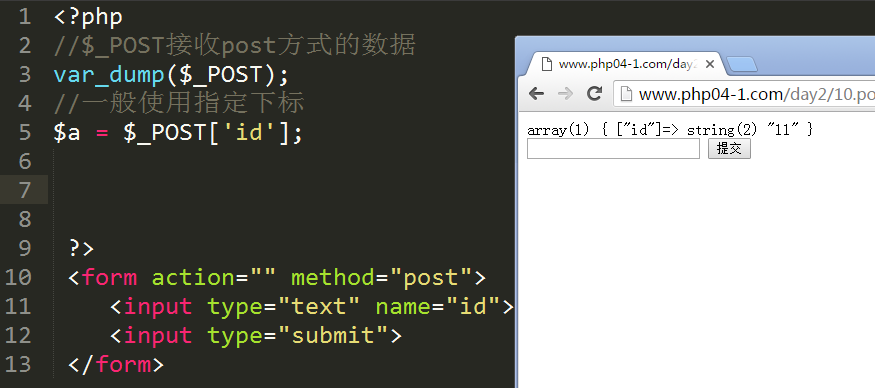
### $\_GET

* 说明：接收GET方式提交的数据---表单和url



### $\_POST

* 说明：接收post方式提交的数据



### $\_REQUEST

* 说明：接收get或post数据



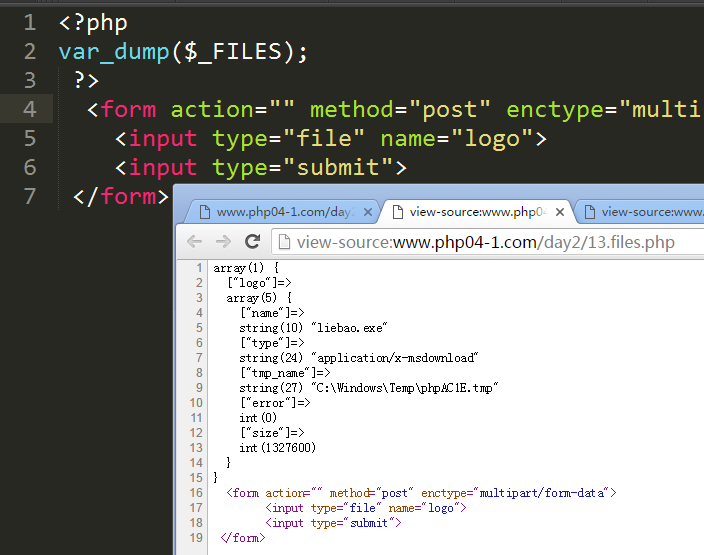
### $\_SERVER

* 说明：可以获取到服务器和客户端一些信息



### $\_FILES

说明：接收上传文件的信息



## 常量

### 4.1定义常量并赋值

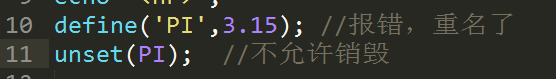
* 说明：常量存储不能变化的量，一旦定义不能销毁和改变值
* 使用：整个网站需要固定内容的就用常量，比如网站的根目录
* 定义赋值：define(‘常量名’,值);



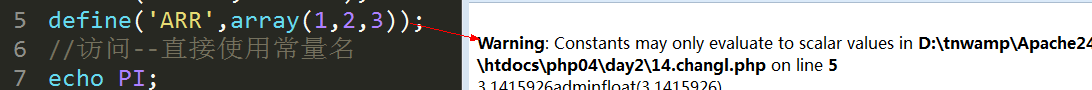
* 命名规则：和变量是一样的，但是建议使用大写，和变量区分
* 常量名必须用引号括起来
* 常量名不需要$符号
* 访问常量：直接使用常量名就可以访问



* 常量一旦定义不能销毁和改变

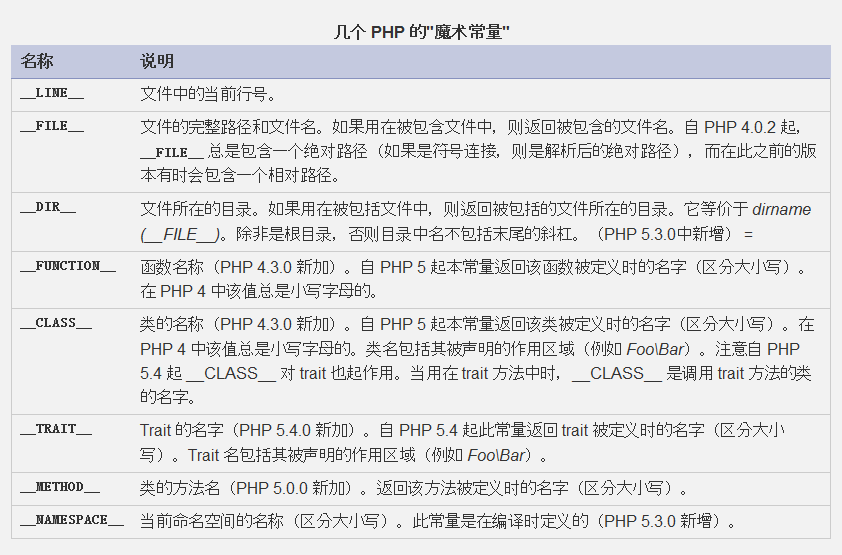


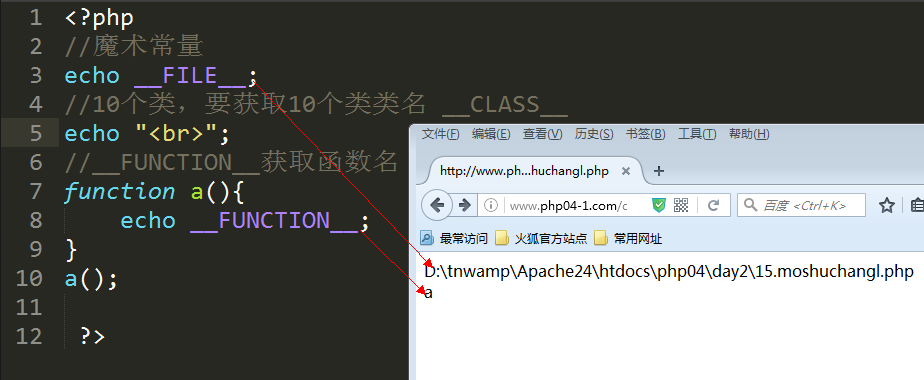
* 常量的值只能是：标准数据类型



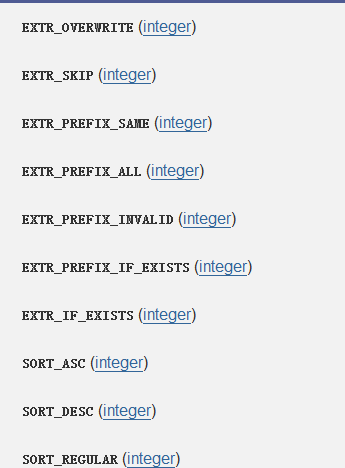
### 4.2魔术常量

* 说明：常量中值可以根据系统变化而变化
* 如：当前文件的路径，路径有可能改变位置，值跟着改变





### 4.3预定义常量





## 数据类型

说明：把数据分为多种类型

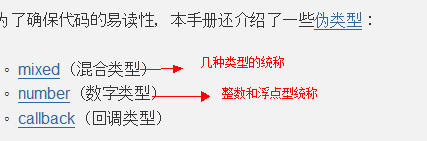
标准类型（标量）：int(整数型) / float(浮点型) / bool(布尔型型) /string(字符串型)

复合类型：arrary(数组) / object(对象)

特殊类型：null(空) /resource(资源)

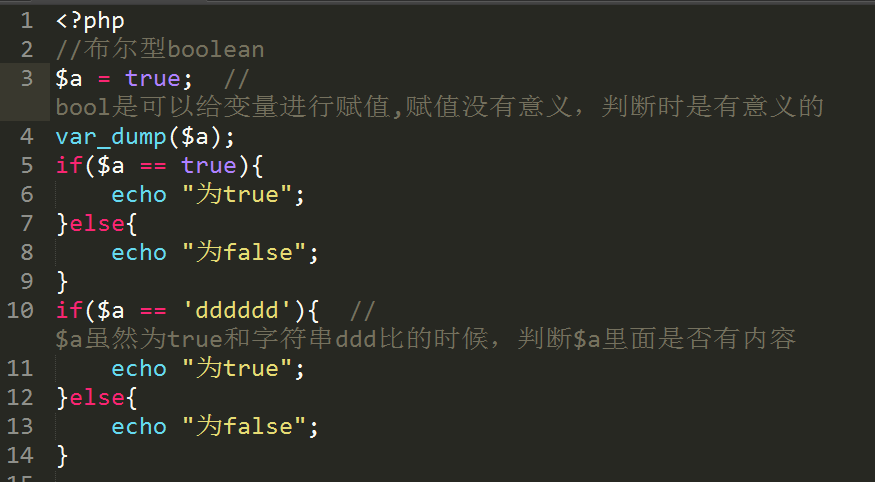
为类型：不会出现在代码中，一般在手册函数的说明中

Mixed / number / callback

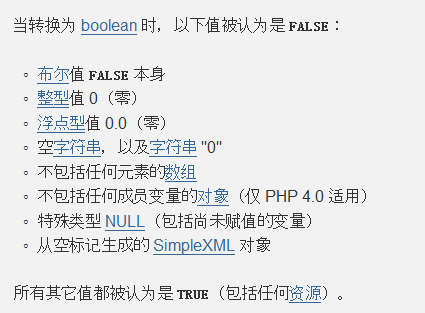


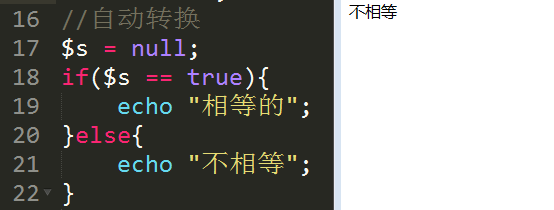
### 布尔型（boolean）

* 说明：就是真值，true和false
* 作用：一般判断使用,判断某个东西是否为true或false



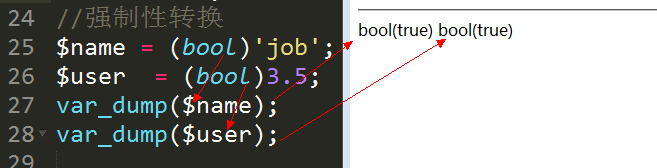
* 自动转换：在比较时才会自动转换





* 强制性转换

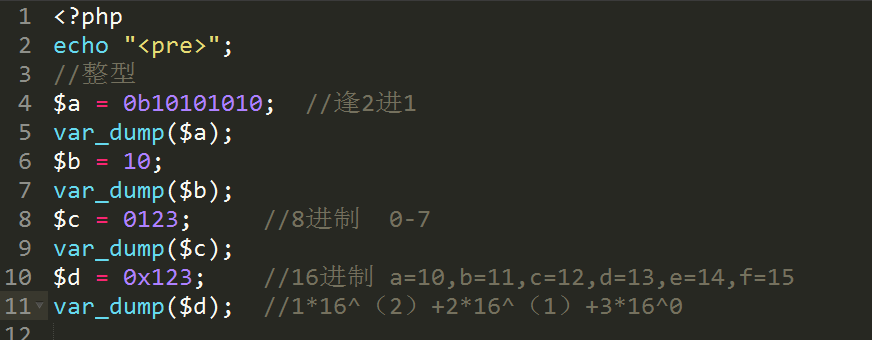
用(bool)在值的前面使用，就会把数据强制转换为bool型



### 整型(int)

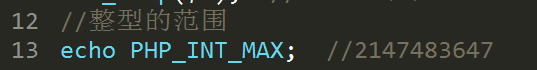
* + 说明：就负整数，零，正整数的集合
  + 整型可以有四种方式定义

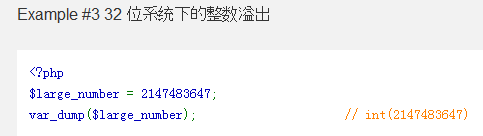
10进制 8进制 16进制 2进制

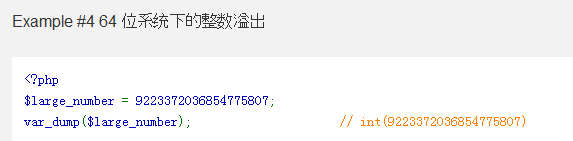


* 整型最大存储值和每个整数占多大空间

整型超出范围根据系统有关系—超出范围都会转换成浮点型



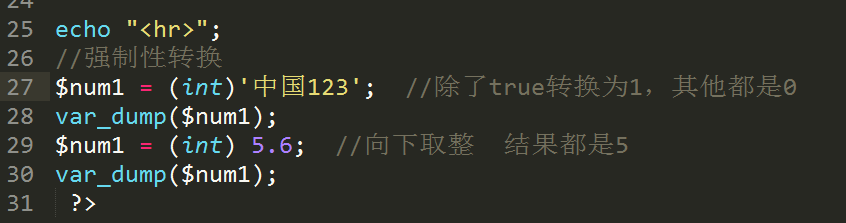






* 强制性转换—把其他类型转换成int类型

使用(int)值



### 浮点型（float）

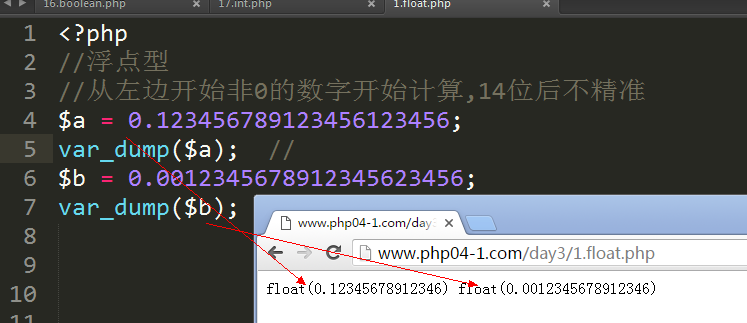
浮点数的定义：

$a = 1.2;

$b = -3.2;

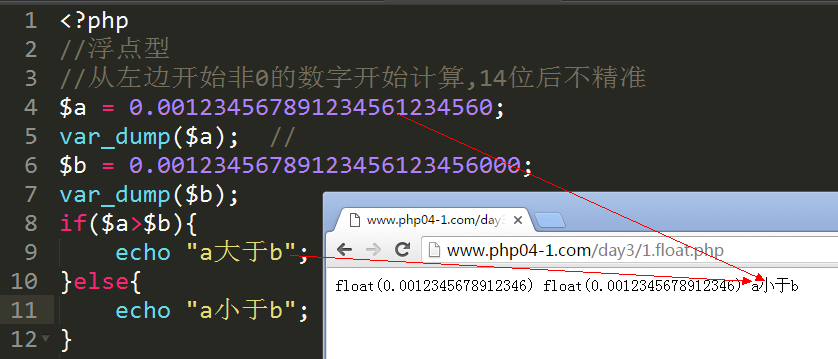
$c = 10E+10;

* 浮点数是有精度的：一般是左边开始数，数14位，非0的开始

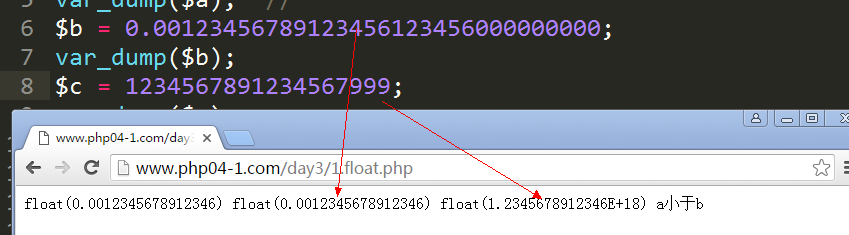


* 浮点型的比较

Php中不进行比较，如果非要比较，取整

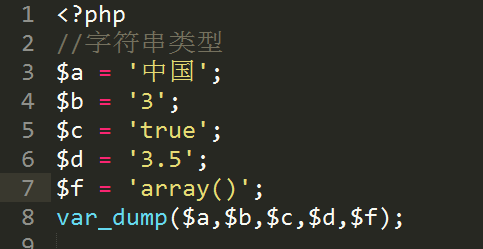


浮点数是不会自动换行成科学计数法，只有整数转。



### 字符串类型(string)

* 字符串有四种表示方法：单引号，双引号，字符串标示符(heredoc,nowdoc)
* 单引号的方式：任何数据加上单引号就是 字符串，纯字符串的是建议是所有单引号，单引号的字符串解析速度要快,不解析变量,加上单引号后所有的东西失去本身的意义。



* 双引号：所有的数据加上双引号，就是字符串，双引号是解析变量的；

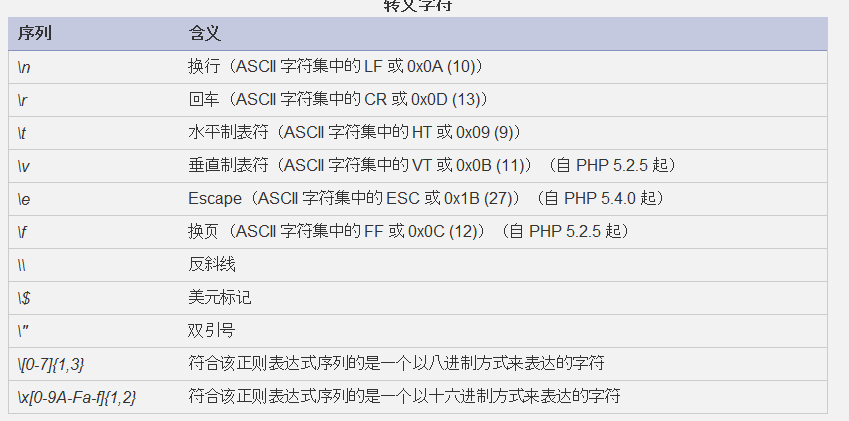
复制的变量，在字符串中建议加上{}，把变量括起来

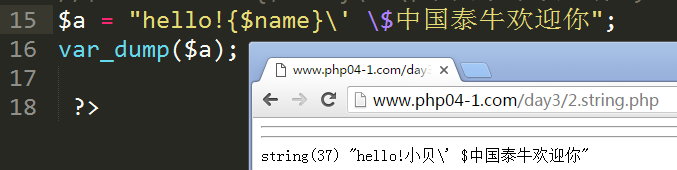


* 转义：

单引号是转义，单引号本身和转义符号

双引号：



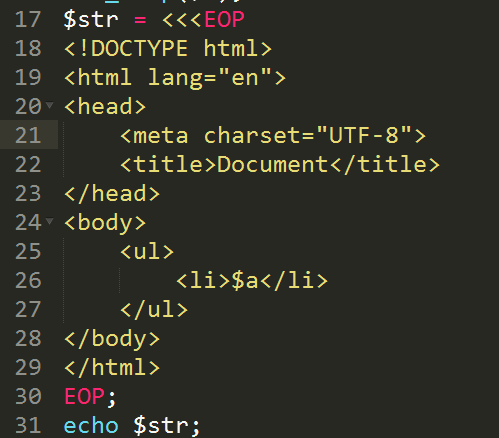


* heredoc：第三种表达字符串的方法是用 heredoc 句法结构：<<<。在该运算符之后要提供一个标识符，然后换行。接下来是字符串 [string](mk:@MSITStore:D:\\itbull\\handbook\\php_enhanced_zh.chm::/res/language.types.string.html) 本身，最后要用前面定义的标识符作为结束标志,heredoc是解析变量的。

$str = <<<EOP //后面是不能有任何字符串，包括空格

大量的字符串

EOP; //前面不能有任何字符，包括空格

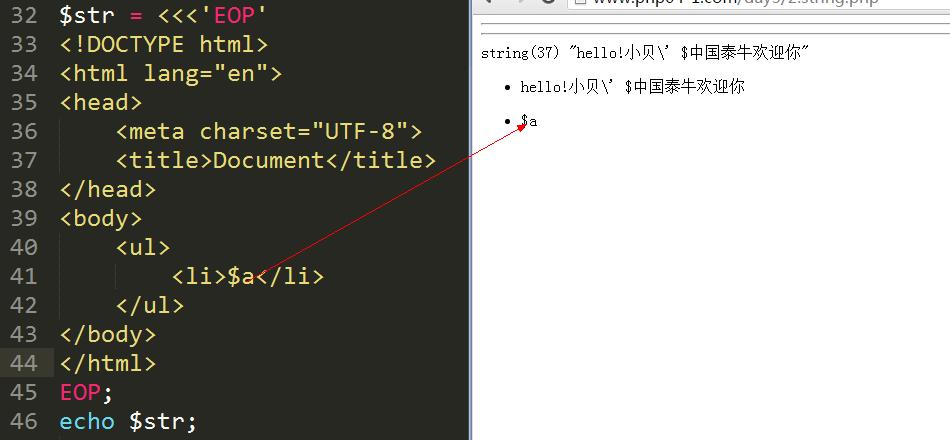


* Nowdoc：单引号的方式，不解析变量

$str = <<<’EOP’

大量的字符串

EOP;



### 资源：

说明：不是php通过某种“语法”创造出来的数据，就是php外部引入进来的数据（数据库，文件，图片等），获取到的数据都是资源，资源只是一种数据统称，资源是必须要解析的。

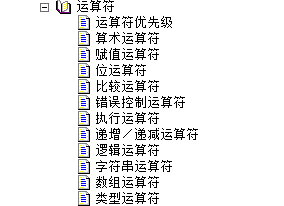
如：数据库资源，必须解析。上传文件给上传文件信息—想法资源解析

### Null(空)

$a; //在php中不会出现只定义变量不赋值的情况，有的时候又需要变量一开始为空，就可以使用null开赋值

如：$a = null;后面做完运算后把值给$a；一开始不存储内容。

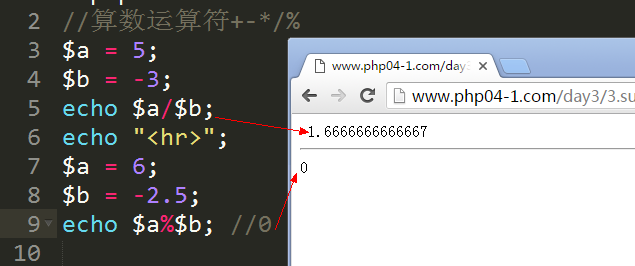
## 运算符



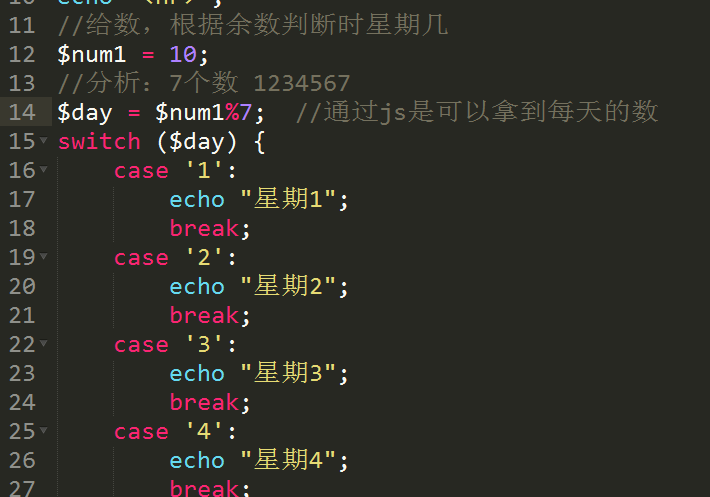
### 算数运算符

符号就是:+-\*/%





案例：1.闰年，2，奇数和偶数，3，星期



### 赋值运算符

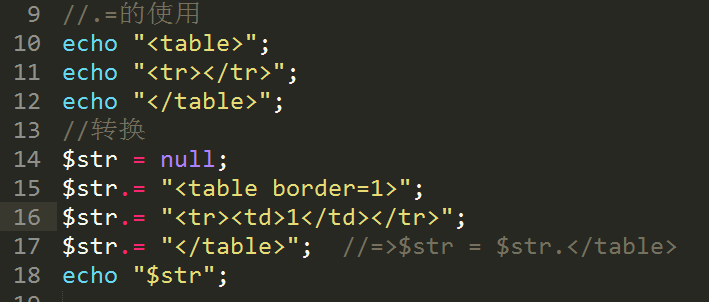
* + 符号：=,+=,-=,\*=,/=,%=,.=

=:把值赋予某个变量

+=:两个变量相加给其中一个变量赋值

如：$a = 10;$b=20; $b+=$a=>$b=$b+$a;

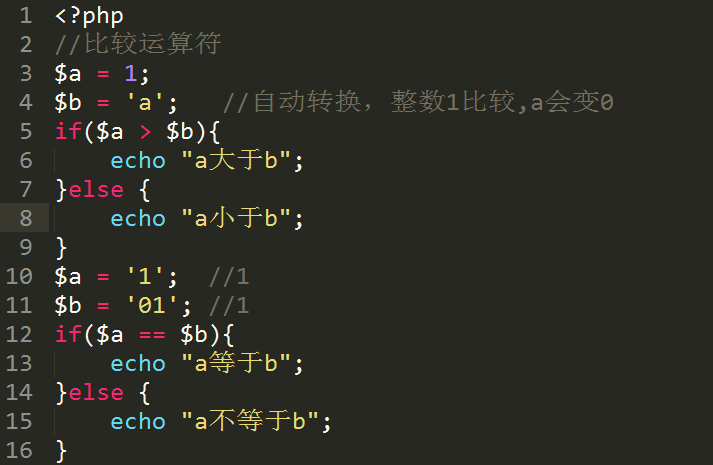
.=:连接字符串,多个字符串连接

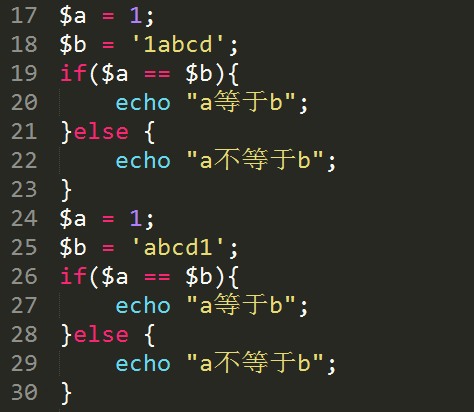


### 比较运算符

符号：>,<,>=,<=,==,===,!=,!==





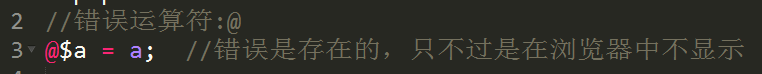


### 错误运算符

符号：@

说明：屏蔽错误信息，在网页中不显示

屏蔽错误之后会创建一个“错误日志”,错误级别通过php开启



上线和线下的就会配置不同的报错级别，把所有的错误进行屏蔽，把错误写到“错误日志”

### 递增和递减运算符

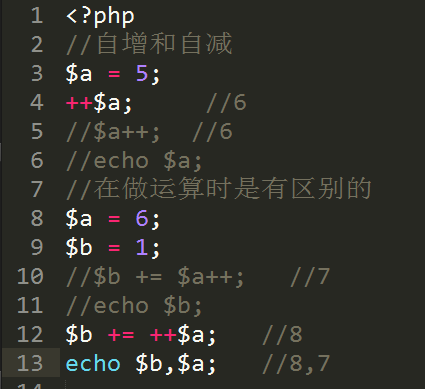
* 符号：++,--
* ++:自动+1；
* --：自动-1；
* 说明：前加和后加,++是在变量前还是变量后
* 区别：

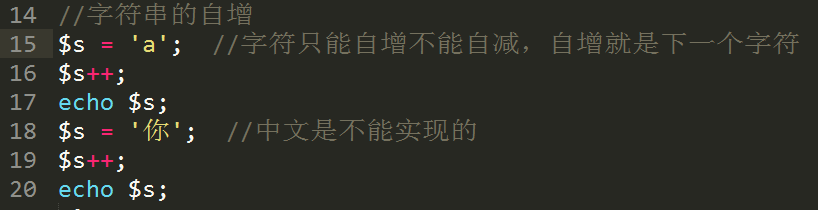
1.前加和后加：如果加完不进行运算，没有区别的。

2.在做运算，是有区别的

前加：先+1在运算

后加：先运算在+1





### 逻辑运算符

* 说明：逻辑运算符也称为“短路运算符”：
* 符号：&&与,||或,!非
* 短路：达成左边的条件，右边的条件不再判断

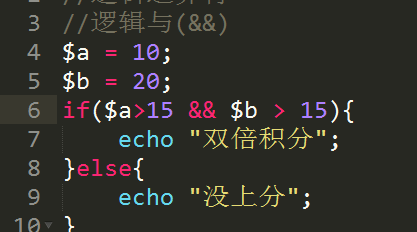
#### 逻辑与(&&):

**左边如果为真，右边为假，结果为假,一般不会是左边为真，右边卫假,只有左边出现为假的情况，原因“短路”，短路的好处就是运行速度**

左边为假，右边为真，结果就是假;

左边为假，右边为假，结果就是假;

左边为真，右边为真，结果就是真;

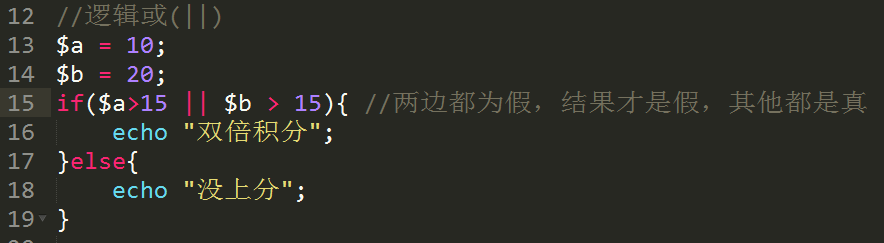


#### 逻辑或(||)

左边为假，右边为真，结果就是真

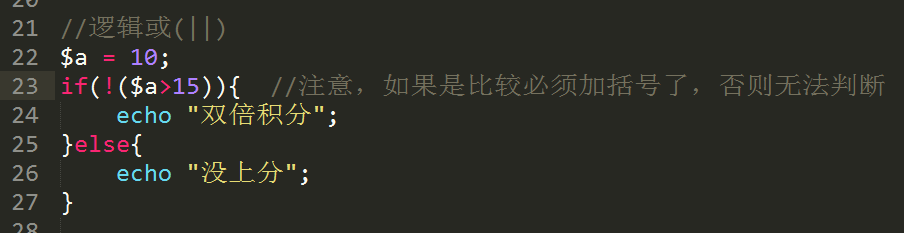
左边为假，右边为假，结果就是假

左边为真，右边为真，结果就是真



#### 逻辑非(!)

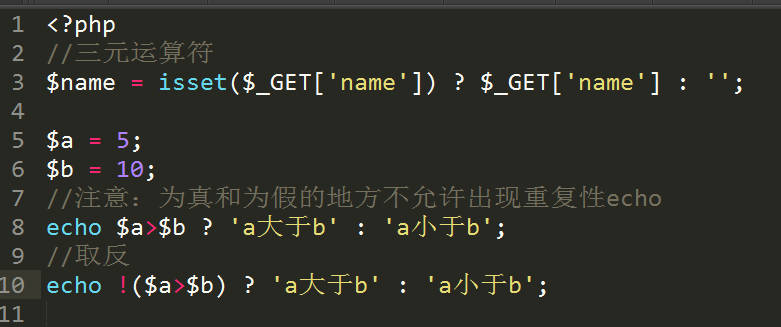
非真即假，非加即真：把真和假进行对换



### 三元运算符

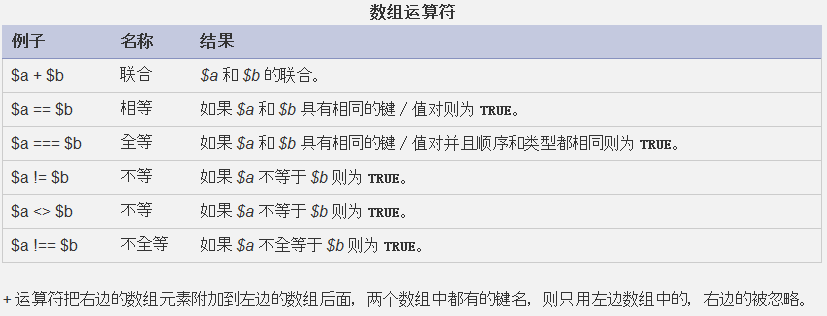
* 说明：ifelse简写形式
* 语法：表达式 ？为真 : 为假

**一般用于数据的接收**

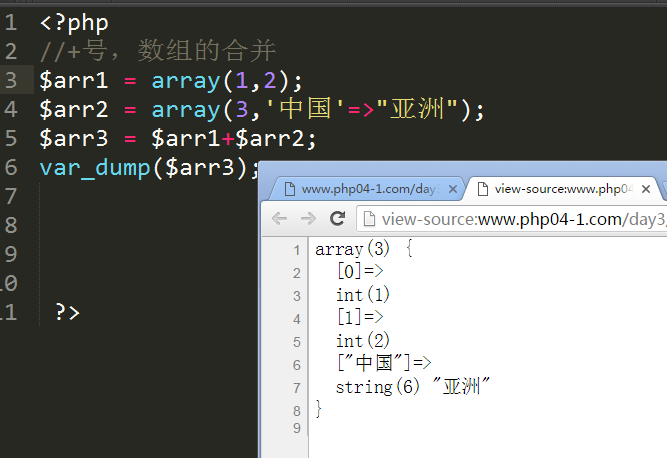


### 数组运算符

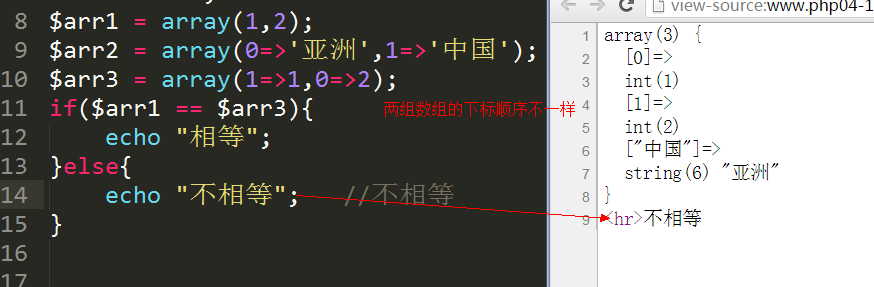
* 说明：对两组数组进行比较



+号就把两个数组进行合并，就是右边数组插入到左边数组后面，如果键名一样，会使用左边键名对应的值



==：两个数组键名和值一样的情况下才相等



### 位运算符

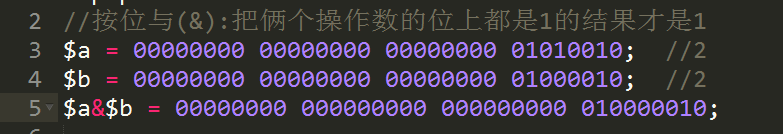
计算机的存储方式都是二进制，二进制表示方法就是0和1

说明：跟开发网站基本用不到，位运算符操作的二进制的数(000001111),根底层有关系时



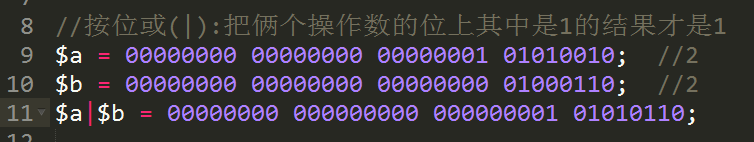
#### 按位与(&)

把$a和$b中都为1的位都设为1



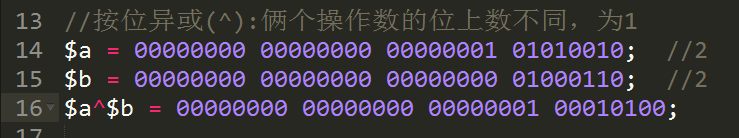
#### 按位或(|)

把两个操作数据中有1的位，结果就是1；



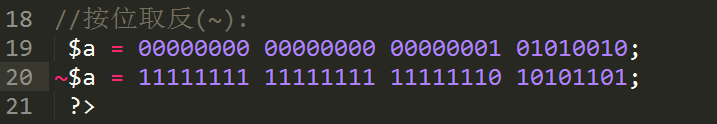
#### 按位异或(^)

说明：两个操作数位不相同时，设为1；



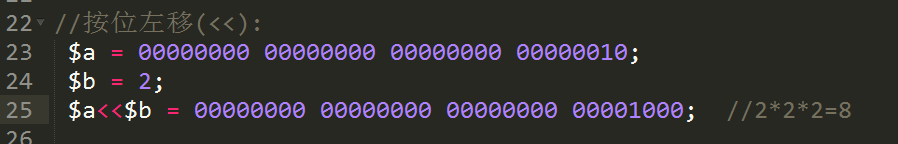
#### 按位取反(~)

把1操作数的位上1变成0，0变成1；



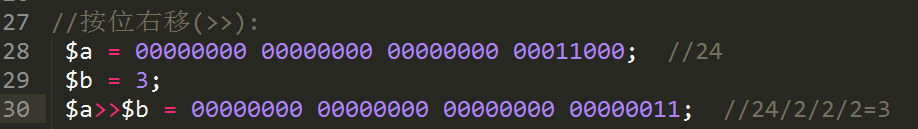
#### 按位左移(<<)

说明：把$a的位移动$b指定的次数,每次乘以2



#### 按位右移(>>)

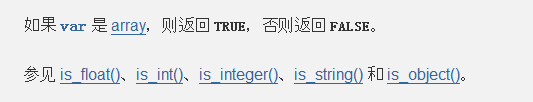
说明：$a的位移动$b指定的次数，每次除以2；



### 常用的一些函数

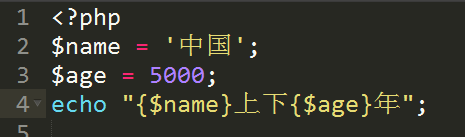
强制转换的函数：(int) (float) (string) (array) (object) (bool)

判断数据类型的函数：is系列



## 流程控制

### 7.1顺序执行：平时说的代码的执行顺序—从上到下



### 7.2分支结构

说明：有需要的执行某段代码，主要有三个(if,if..else,if..else if..)

#### If结构（单分支）

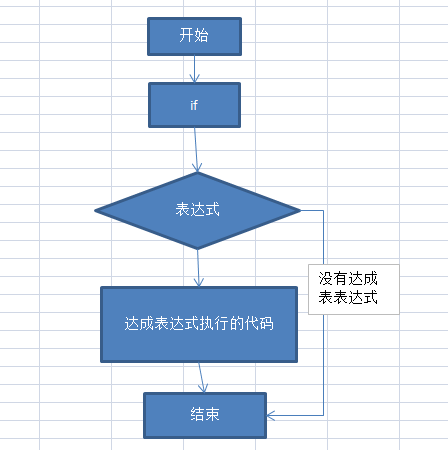
语法：

If(表达式){

达成表达式执行的代码;

}

* 流程图

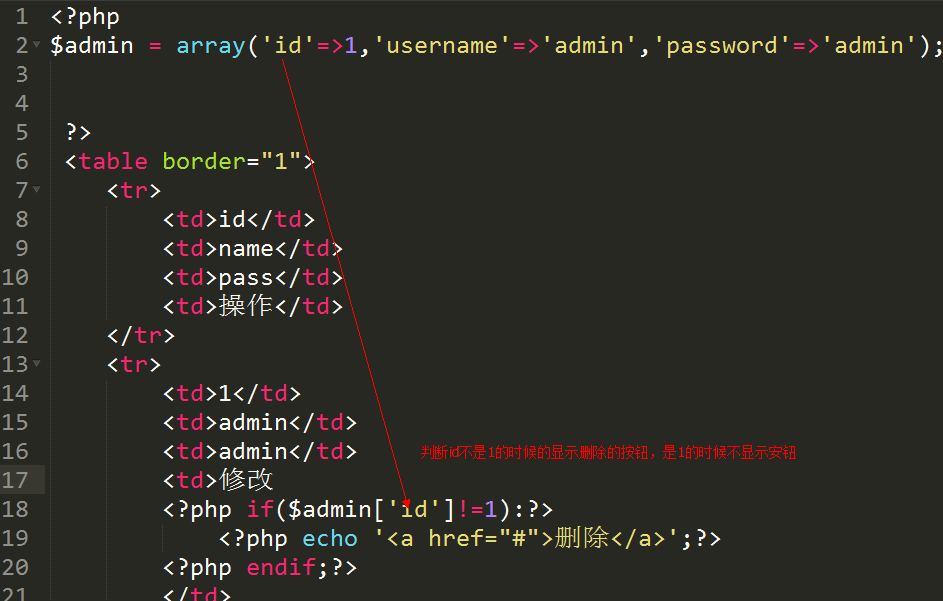


* 在php和html混编中怎么使用流程控制：
* 场景：有个管理员表，有多个管理员，设置id为1的位超级管理员，不允许删除  
  流程控制在html中使用语法：

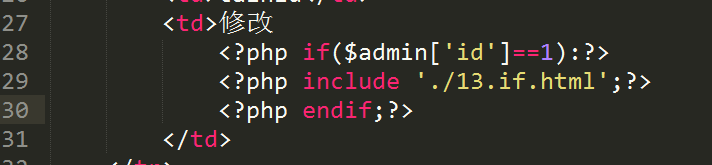
<?php if():?>

普通代码;

<?php endif;?>



* 场景：根据需求引入html的模板文件



通过这种方式，可以实现当前页面的位置可以显示不同的页面。

最早的模板引入方式：就是实现php代码和html代码的分离

页面不是固定的，而是根据需求变化的。

#### Ifelse结构（双分支）

说明：判断某个表达式来区分执行要执行代码

语法：

If(表达式){

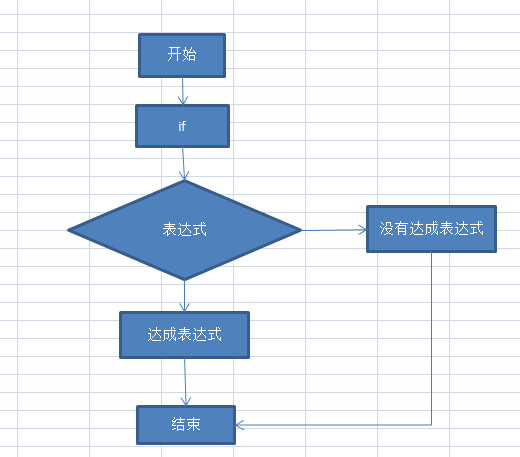
达成表达式时执行的代码

}else{

没有达成表达式时执行的代码

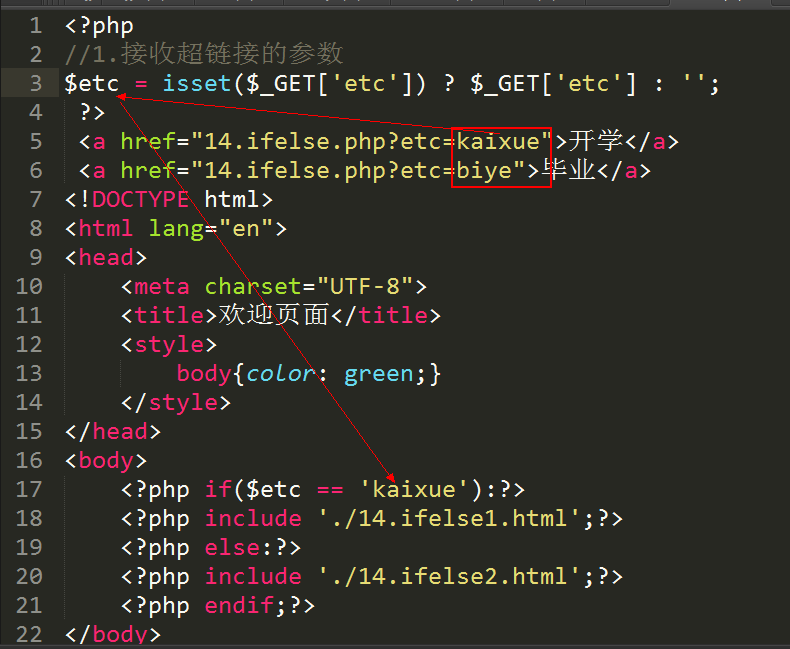
}

* 流程图



在html中用法：

场景：根据超链接的参数不同引入不同页面



#### If else if结构

* 语法：

If(表达式1){

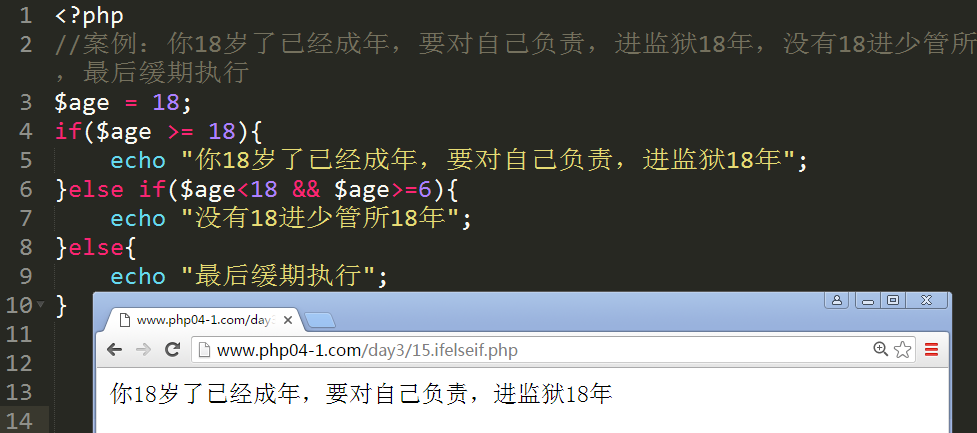
达成表达式1时执行的代码；

}else if(表达式2){

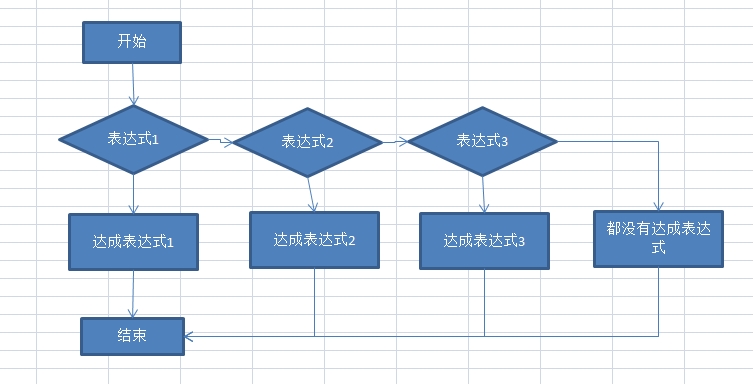
达成表达式2时执行代码；

}else if(){}……else{}

* 案例：你18岁了已经成年，要对自己负责，进监狱18年，没有18进少管所，最后缓期执行



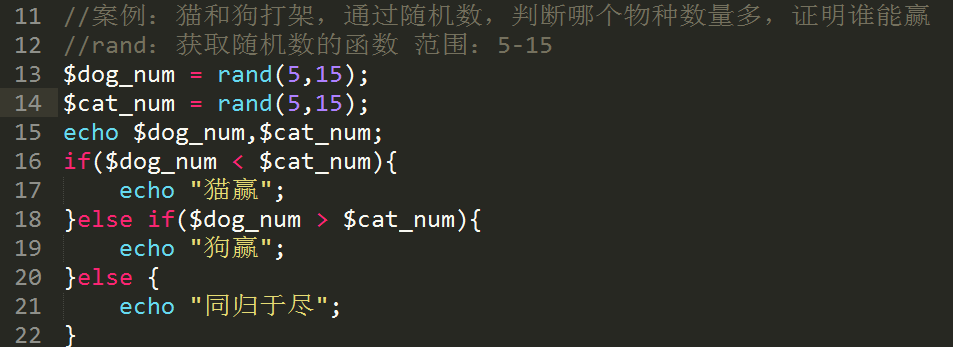
* 流程图：



案例：猫和狗打架，通过随机数，判断哪个物种数量多，证明谁能赢

rand：获取随机数的函数





* 在html中替代语法：

<?php if():?>

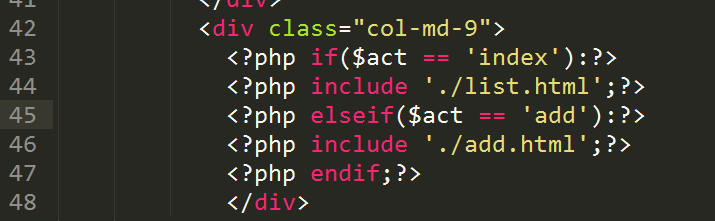
达成表达式1的执行代码

<?php else if():?>

达成表达式2的执行代码

<?php endif;?>

案例：根据参数引入多个页面



### 7.3循环结构

目的：循环目的是重复执行相同的代码

#### For循环

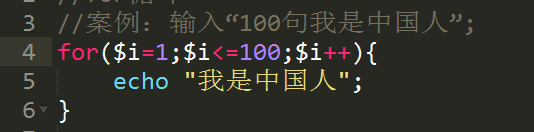
语法：

For(变量初始化；条件表达式；变量的变化){

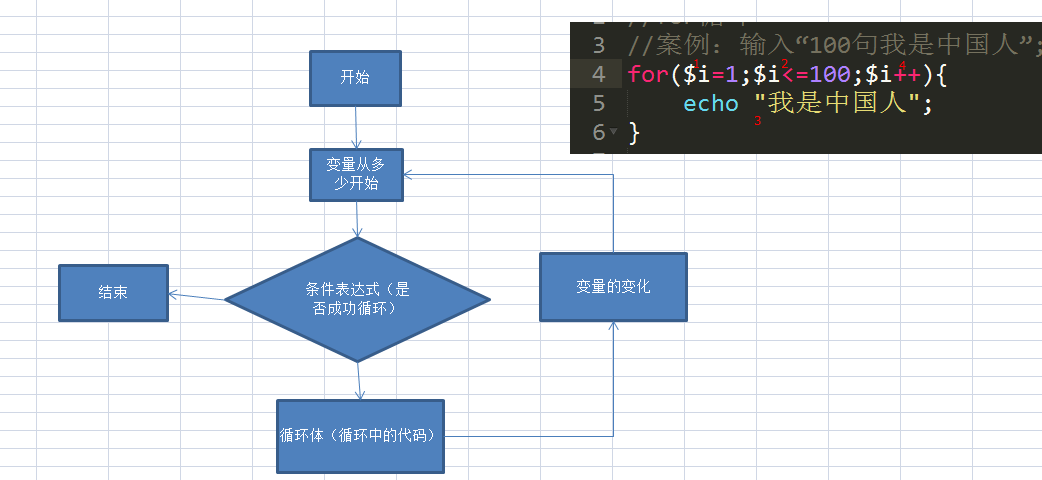
循环体；

}

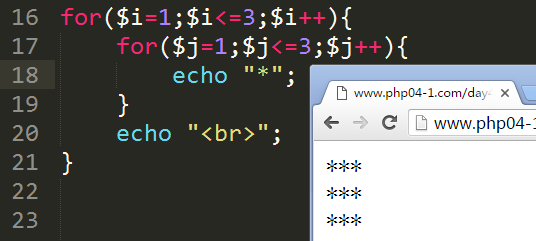
案例：输入“100句我是中国人”;

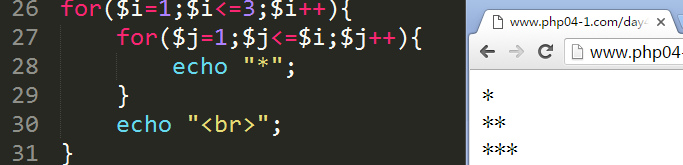


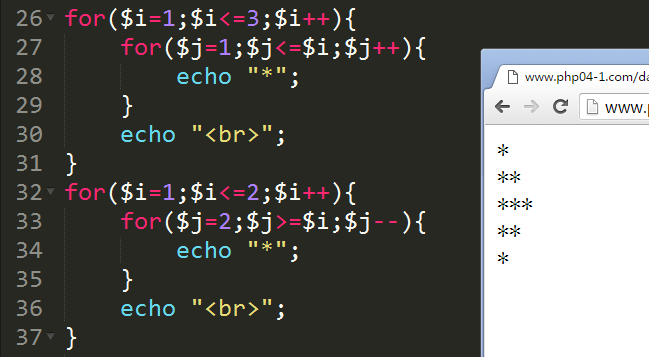
流程图

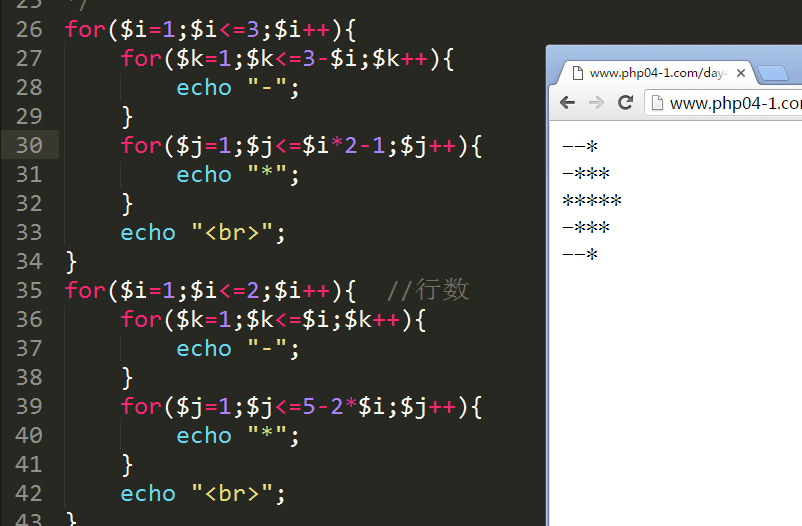


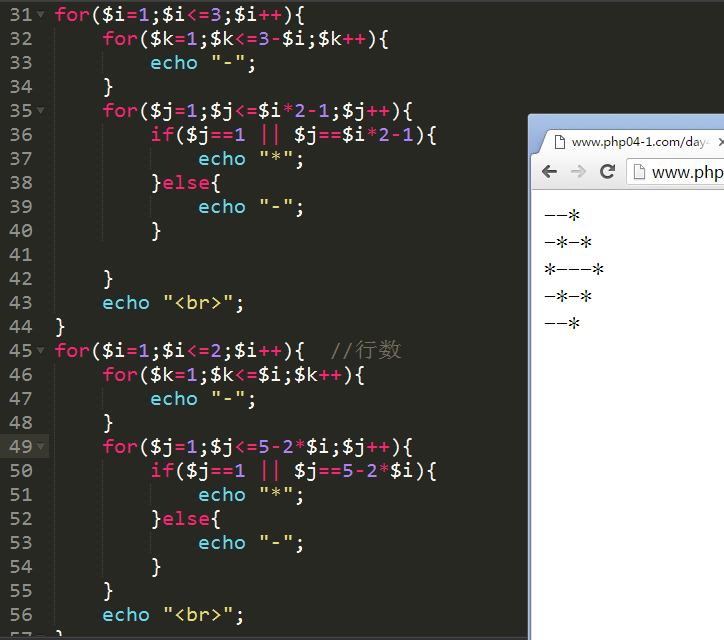
金字塔





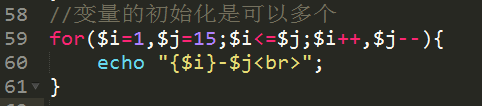






* 循环的细节

1变量的初始化是可以多个，用逗号隔开



#### Whlie循环

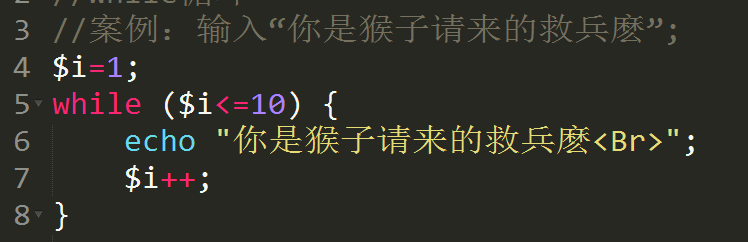
语法：

变量的初始化；

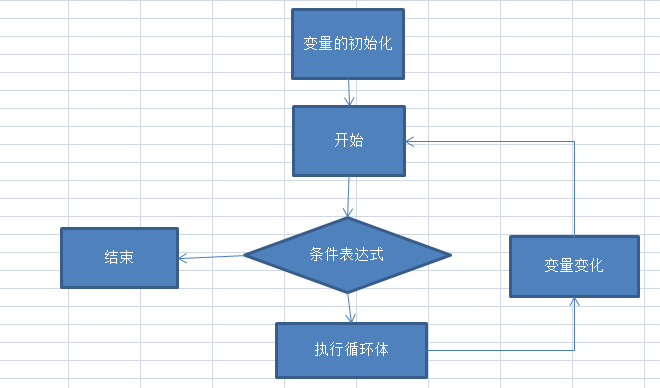
While(条件表达式){  
 循环体；

变量的变化；

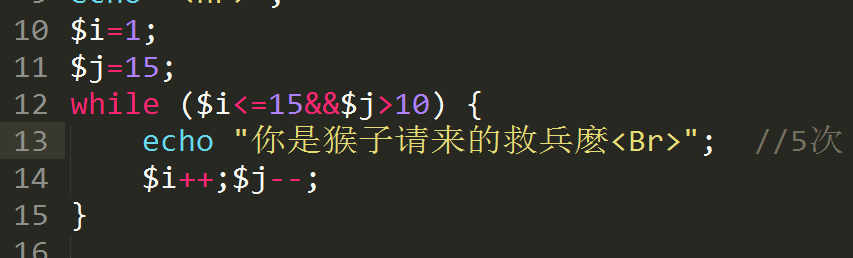
}



* 流程图



* While也可以有多个变量的初始化



#### Dowhile循环

语法：

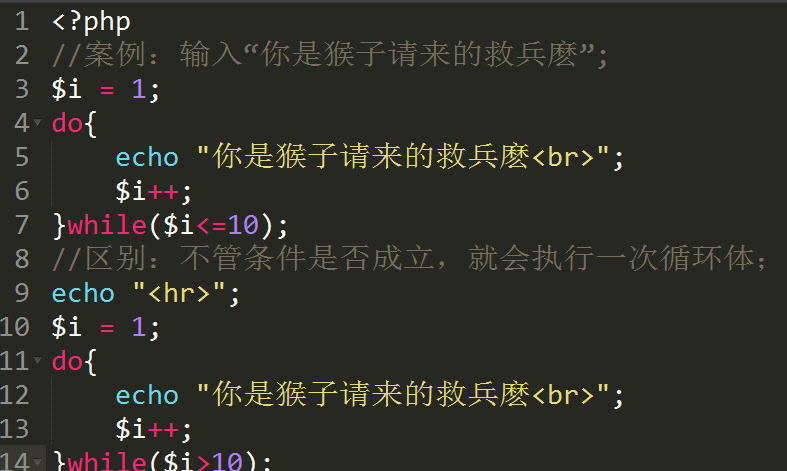
变量的初始化；

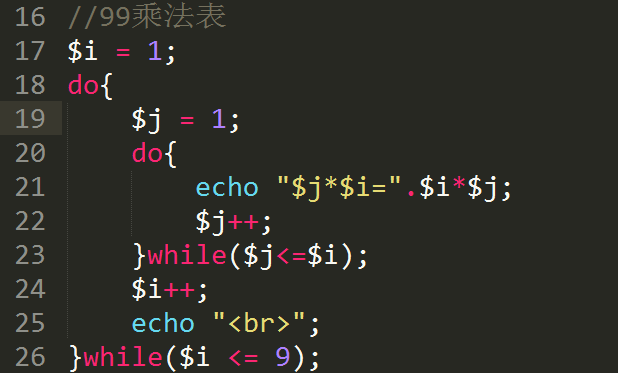
Do{

循环体；

变量的变化；

}while(条件表达式)





### 7.4 switch分支结构

语法：

Switch(变量){

Case 值1:

达成值1执行的语句；

Break;

Case 值2:

达成值1执行的语句；

Break;

Case 值3:

达成值1执行的语句；

Break;

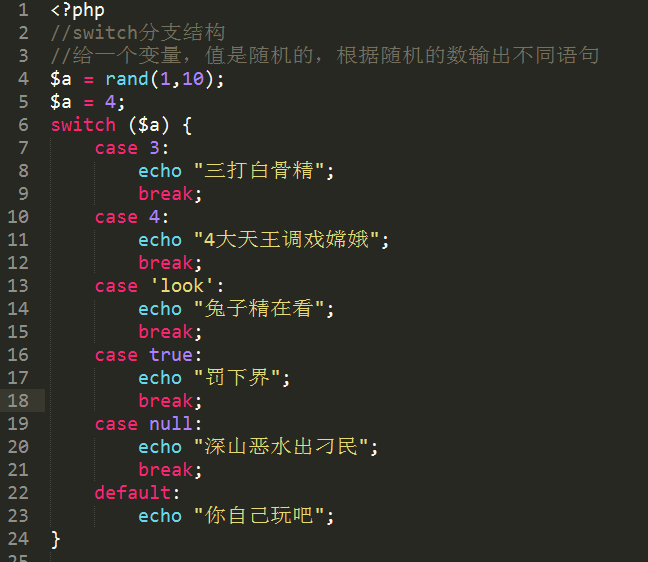
……..

Default :

都没有达成的情况会执行default里的语句；

}

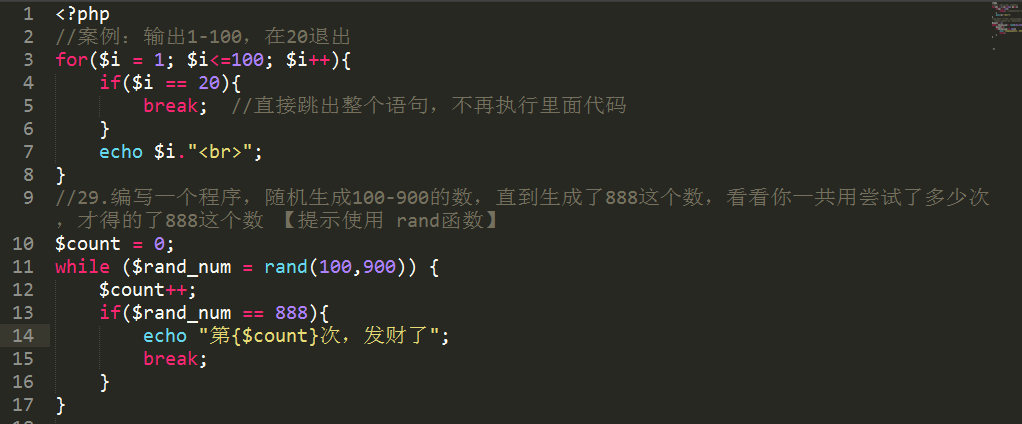
1.值一般是数字，字符串等，只要是固定值,



**Break在switch中必须有，没有的话会执行所有的条件。**

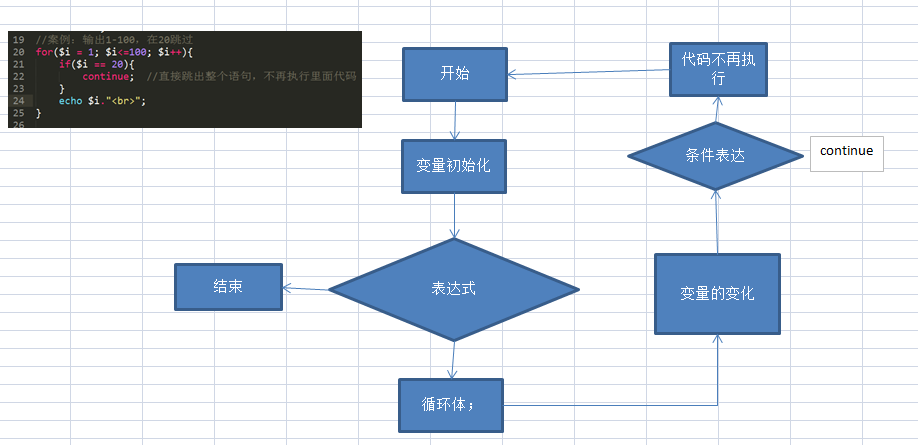
### 7.5 break

说明：break 结束当前 for，foreach，while，do-while 或者 switch 结构的执行。



### 7.6 continue

说明：跳出当前次的循环

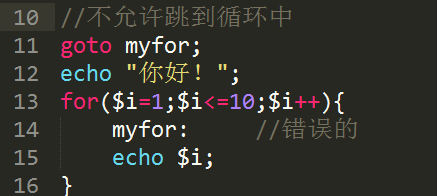


### 7.7 goto

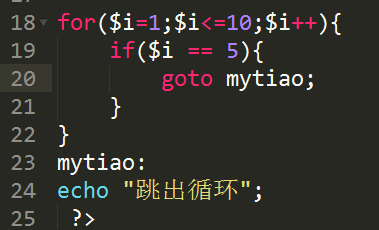
说明：直接位置的跳转



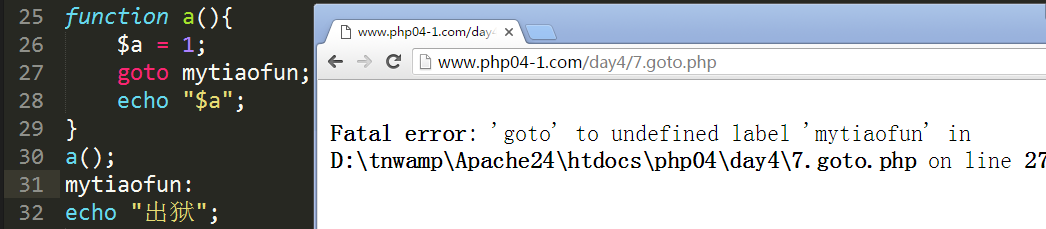
* Goto 后面跟的是跳转的标签
* 到哪里标签，后面跟冒号，而且两个标签名必须一致。
* 注意事项：
  + 不能跳入循环中



* 可以跳出循环中



* 不能跳入函数中
* 不能跳出函数外



* 不能跳出文件外

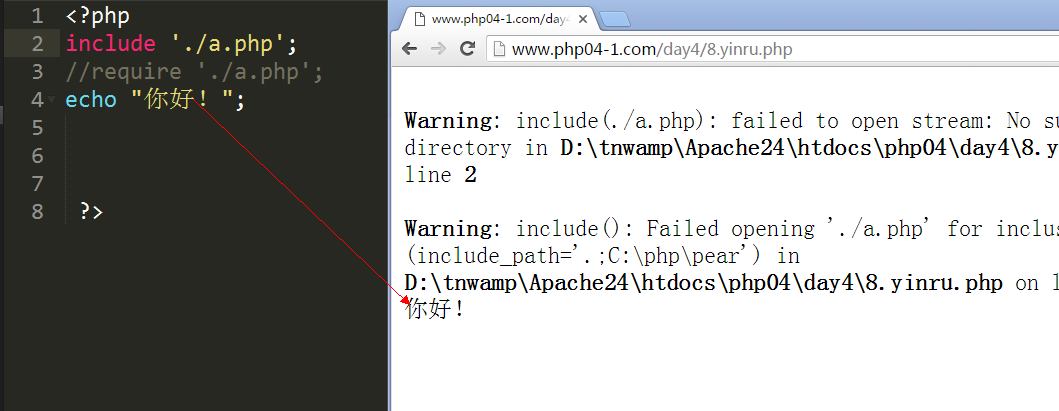
### 7.8 文件的引入

说明：从外部引入另一个文件,引入有两种方式

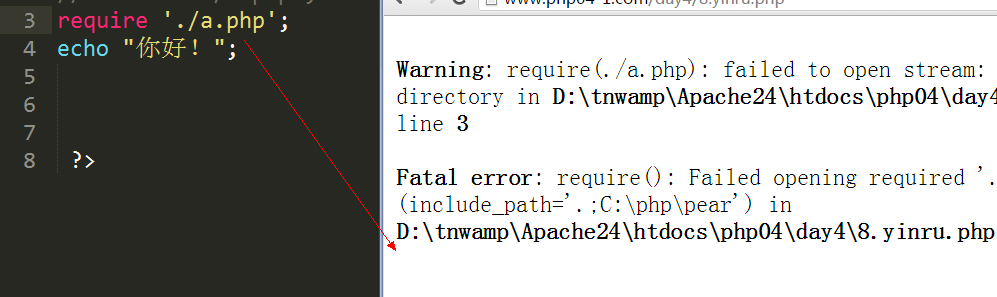
* Include和require
* Include\_once和require\_once

#### Include和require的区别

Include引入出错后继续执行后面的代码，报错不影响代码的执行



Require引入出错后不再执行后面的代码，报错影响代码的执行



Once和没有once的区别

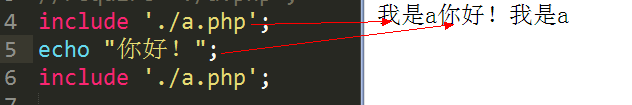
1.找不到文件时没有区别的

2.多次引入同一个文件时

* Once说明：同名的文件只引入一次
* 没有once的引入是会多次引入的。

#### 作用：

* **引入就是把被引入文件的代码插入到引入的语句那个位置。**



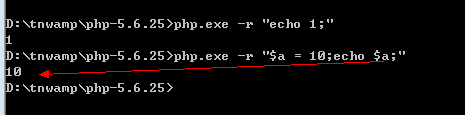
## Php可以在cmd中运行代码

Php在哪里运行---服务器

Php在哪里看----浏览器----命令行模式

就是php.exe这个文件运行php代码

-r 代码



命令行：可以进行定时的设置—在多少时间是运行某段代码