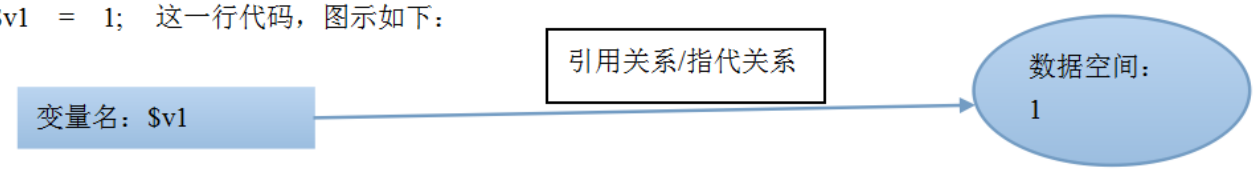
变量

基本理解

可以存储可变数据的标识符——就是一个我们自己定义的名字。

php 中，变量名和其对应（存储）的值之间的关系示意图：

$v1 = 1; 这一行代码，图示如下：



变量的基本操作

赋值：

$变量名 = 值；

取值：

在需要一个数据值的场合，使用一个变量，就表示使用该变量的值（取得其值）。

```
echo  $v1;
$v2 = $v1 + 2;      //取得 v1 的值，并进行加法计算
f1($v1,  2,  3);
```

判断 isset(变量名)：

isset()判断一个变量“是否存在”，也包括如果变量中的“值”是 null，会判断为“不存在”。

如果存在，就是 true

如果不存在，就是 false

empty()判断一个变量“是否是空的”：有多种情况都是“空的”:0, “”, “0”, false, null, array()

如果是空的，就是 true

如果不是空的，就是 false

删除 unset(变量名)：

变量命名规则

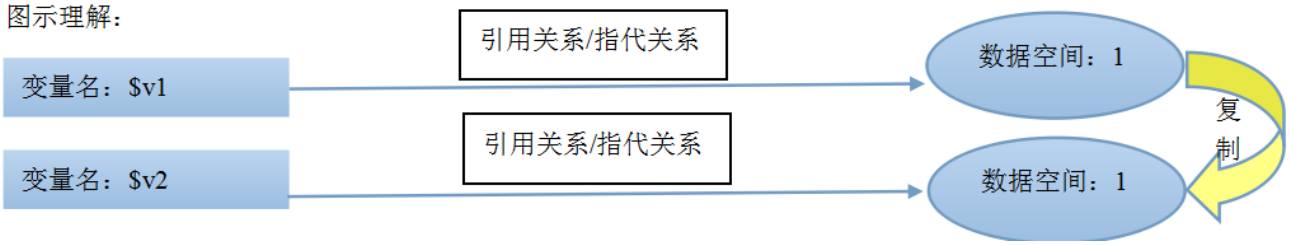
基本规则——保证程序的正确性

行业规则——保证程序的可读性

变量的传值方式

值传递：

图示理解：




```
$v1  = 1;
$v2  = $v1;        //v2 为 1；
$v2 = 10;
echo  $v1;         //1
unset($v1);
```

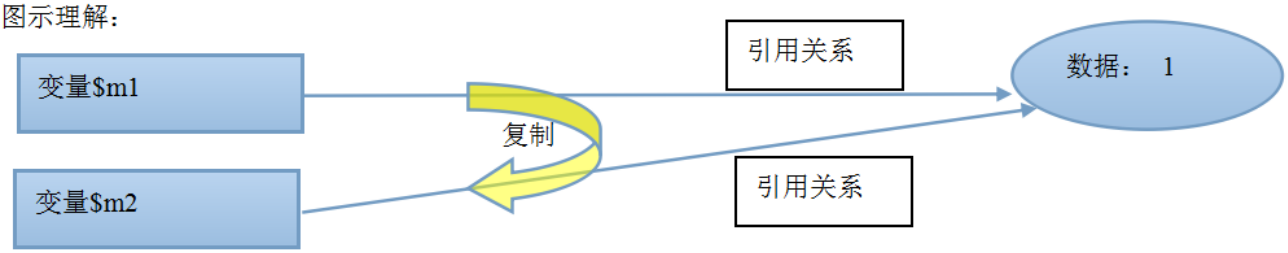
isset($v2);　　//???——true

引用传递：

$m1 = 1;

$m2 = &$m1;　　//引用传递，此时$m2 也是 1

图示理解：



$m2 = 10;

echo　$m1;　　//10

unset($m1);

echo $m1;　　//此行报错：变量 m1 不存在

isset($m2);　　//???——true;

isset($m1);　　//???——false

可变变量：

$s123　= 100;

$v1 = “s123”;

echo　$$v1;　　//100;怎么理解：$v1 是要输出的这个变量的“名字”，即为“s123”，那么结果就是要输出$s123;

$k1 = “k2”;

$k2 = “k3”;

$k3 = 3;

echo　$$$k1

预定义变量

综述

$_POST 变量：

含义：代表网页客户端通过 post 方式所提交过来的数据！，是一个数组；

取得其中任何一项数据，其形式都是：$_POST[‘数据项名称’];

只有一种形式可以以 post 方式提交 post 数据：

<form　action=”目标文件.php”　method=”post”> ..... </form>

$_GET 变量

含义：代表网页客户端通过 get 方式所提交过来的数据！，也是一个数组；

取得其中任何一项数据，其形式都是：$_GET[‘数据项名称’];

有多种形式可以以 gett 方式提交 get 数据：

a: <form　action=”目标文件.php”　method=”get”> ..... </form>

b：文字

说明：？号后面通常称为“名值对”

c：<script>　location.href　=　”目标文件.php?a=1&b=xyz&cc=cctv”　;</script>

d：<script>　location.assign(　”目标文件.php?a=1&b=xyz&cc=cctv”　);

</script>

e：<php　header(“location:目标文件.php?a=1&b=xyz&cc=cctv”);

$_REQUEST 变量

代表客户端在一次请求中提交过来的 get 数据和 post 数据的“合集”。

$_SERVER 变量：

代表一些客户端在一次请求中的浏览器端信息和服务器端信息。

$GLOBALS 变量

代表我们自定定义的“全局变量”的另一份存储形式：即所有全局变量，都又存储到这个预定义常量中去了。

$v1 = 1; //全局变量；则此时就有了: $GLOBALS[‘v1’]; 其值为 1；

常量

常量是相对于变量来说的：是一个其中存储的数据不会也不应该改变的“标识符”。
常量的使用，就 2 个方面：定义，取值。

常量的定义：

```
14  //常量定义语法1:
15  //define("常量名",  常量值);
16  define("PI",  3.14);
17  define("SCHOOL", "传智播客");
18
```

```
18  //定义形式2:
19  //const  常量名  = 常量值;
20  const  CC1 = 1234;
21  const  CC2 = 'abcd';
```

常量的使用——取值：

也有两种形式：直接使用名字，或通过 constant()函数取得其值；

```php
//使用形式1：直接使用其名字
echo "<br />常量PI的值是：" . PI;»  //注意，不能写在引号中
echo "<br />常量SCHOOL为：" . SCHOOL;
$s1 = PI * 3 * 3;»  //求半径为3的圆面积
//使用形式2：使用函数constant()获得一个常量的值：
//形式：  constant("常量名");//注意：常量名是一个字符串
$s2 = constant("PI") * 3 * 3;
echo "<br />s1= $s1, s2 = $s2";
echo "<br />" .  SCHOOL . constant("CC1") . constant("CC2");
```

```
常量PI的值是：3.14
常量SCHOOL为：传智播客
s1= 28.26, s2 = 28.26
传智播客1234abcd
```

```php
//取得常量值的灵活性语法：
$i = 1;
$c1 = "CC" . $i;
echo "<br />常量 $c1 的值为：" . constant($c1);//输出1234
```

常量变量的区别

- 定义形式不同：
- 使用形式不同：常量无需$符号
- 可变程度不同：常量的值不可以改变，常量也不可以销毁
- 作用范围不同：常量具有超全局作用域（函数内外都可以直接使用）
- 可用类型不同：常量只能存储标量类型（整数，浮点数，字符串，布尔）

判断常量是否存在

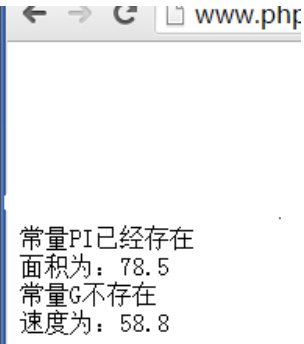
使用 defined()函数：

如果存在：返回结果是 true，

如果不存在：返回结果是 false

```php
if( defined("PI") ){
    echo "<br />常量PI已经存在";    //通常此时就可以去使用它！
}
else{
    echo "<br />常量PI不存在";//通常，判断不存在，是为了来定义它！
    define("PI", 3.14);//然后去使用
}
$s3 = PI * 5 * 5;
echo "<br />面积为：$s3";

if( defined("G") ){
    echo "<br />常量G已经存在";//通常此时就可以去使用它！
}
else{
    echo "<br />常量G不存在";//通常，判断不存在，是为了来定义它！
    define("G", 9.8);  //然后去使用，G是"重力加速度"
}
$s4 = G * 6 ;  //6为时间（秒），这里是计算得到速度
echo "<br />速度为：$s4";
```



使用一个未定义的常量：

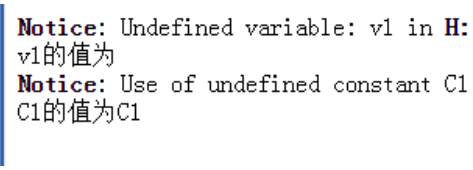
先看 2 个对比代码：

echo “v1 的值为” . $v1; //注意，该变量 v1 未定义过

echo “C1 的值为” . C1; //注意，该常量 C1 未定义过

注意：在 php 中，当使用一个未定义的常量的时候，系统会直接将该常量当做“有值”的常量去使用，并且其值就是该常量名——虽然也会报错！

```php
//
echo "<hr />";
echo  "v1的值为"  .  $v1;    //注意，该变量v1未定义过
echo  "C1的值为"  .  C1;     //注意，该常量C1未定义过
?>
</body>
```



预定义常量

就是系统中预先定义好的一些常量，大约有几百个，我们只要知道几个就行：

M_PI: 就是圆周率的常量值；

PHP_OS: 就是 php 运行所在的操作系统

PHP_VERSION:就是 php 的版本号

PHP_INT_MAX: php 中的最大的整数值

.....更多可参考：php 手册>附录>保留字列表>预定义常量

魔术常量

其实只是常量的形式，但没有常量的“恒常”的含义：其值其实会变化的，只有很少的几个：

__FILE__ :代表当前网页文件的完整物理路径
__DIR__ :代表当前网页文件所在的文件夹
__LINE__ :代表当前这个常量名所在的”行号”

```
64 //魔术常量:
65 echo "<br />" . __FILE__;
66 echo "<br />" . __DIR__;
67 echo "<br />" . __LINE__;
68 echo "<br />" . __LINE__;
69 echo "<br />" . __LINE__;
```

```
H:\itcast\class\bj-php-39\day3\2changliang.php
H:\itcast\class\bj-php-39\day3
67
68
69
```

数据类型

总体划分

有 8 种数据类型：

基本类型（标量类型）：

整数类型： int， integer
浮点数类型： float，double， real
字符串类型： string
布尔类型： bool， boolean 这种类型，只有 2 个数据：ｔｒｕｅ，ｆａｌｓｅ

复合类型：

数组： array
对象： object

特殊类型

空类型： null 这种类型中，只有一个数据，那就是 null
资源类型： resource

整数类型

整数类型的 4 种写法：

$n1 = 123;　　//10 进制数字写法
$n2 = 0123;　　//8 进制数字写法，实际 n2 中存储的数字比 123 小
$n3 = 0x123;　　//16 进制数字写法，实际 n3 中存储的数字比 123 大
$n4 = 0b1010;　　//2 进制数字写法(目前不学）

```
14 $n1 = 123;» »    //10进制数字写法
15 $n2 = 0123;»»    //8进制数字写法，实际n2中存储的数字比123
16 $n3 = 0x123;»    //16进制数字写法，实际n3中存储的数字比12
17 echo "$n1, $n2, $n3";
```

网页标题

123, 83, 291

进制转换问题

首先记住这几个单词：

bin：2 进制
oct：8 进制
dec：10 进制
hex：16 进制

进制转换主要分 2 种情况：

1，10 进制转换为其他 3 种进制：decbin(), decoct(), dechex();
2，其他 3 种进制，转换为 10 进制：

进制转换的系统函数——必须会用

进制转换主要分 2 种情况：

1，10 进制转换为其他 3 中进制：

decbin(一个 10 进制数字)：结果返回的是该数字的 2 进制数字形式的字符串!!!
decoct(一个 10 进制数字)：结果返回的是该数字的 8 进制数字形式的字符串!!!
dechex(一个 10 进制数字)：结果返回的是该数字的 16 进制数字形式的字符串!!!

```
14 $n1 = 123;» //这是10进制的一个数字:
15 $s1 = decbin($n1);» //将10进制转换为2进制
16 $s2 = decoct($n1);» //将10进制转换为8进制
17 $s3 = dechex($n1);» //将10进制转换为16进制
18 echo "<br />$n1 的 2进制形式为: $s1";
19 echo "<br />$n1 的 8进制形式为: $s2";
20 echo "<br />$n1 的 16进制形式为: $s3";
```

123 的 2进制形式为：1111011
123 的 8进制形式为：173
123 的 16进制形式为：7b

2，其他 3 种进制，转换为 10 进制：

bindec(一个 2 进制数字字符串)：　结果返回的是该 2 进制数字字符串对应的 10 进制数字!!!

octdec(一个 8 进制数字字符串)：　结果返回的是该 8 进制数字字符串对应的 10 进制数字!!!

hexdec(一个 16 进制数字字符串)：　结果返回的是该 16 进制数字字符串对应的 10 进制数字!!!

对于输入的字符串中的字符，如果不是对应进制的数字，会被忽略。

那么，有没有这个转换呢？hexbin()????　——没有！

```
$num2 = "1010110";
$num8 = "123";»  »    //注意，当做8进制数字
»   //注意，这里，写的本质是一个”字符串“，不整数数字
»   //所以，不需要加0，即无需写成"0123"

$num16 = "123abc";» //注意，当做16进制数字
$m1 = bindec($num2);»   //将2进制字符串转换为10进制数字
$m2 = octdec($num8);»   //将8进制字符串转换为10进制数字
$m3 = hexdec($num16);»  //将16进制字符串转换为10进制数字
echo "<br />2进制数字：$num2 的对应10进制值为：$m1";
echo "<br />8进制数字：$num8 的对应10进制值为：$m2";
echo "<br />16进制数字：$num16 的对应10进制值为：$m3";
```

```
2进制数字：1010110 的对应10进制值为：86
8进制数字：123 的对应10进制值为：83
16进制数字：123abc 的对应10进制值为：1194684
```

一个思考题：

将 8 进制字符串”12345”，转换为 2 进制结果，怎么做？

$v1 = octdec(“12345”); //此时，v1 是 10 进制数字

$result = decbin($v1); //此时，就是结果：为 2 进制数字字符串！

补充进制基础知识：

10 进制	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
8 进制	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17	20	21
16 进制	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
2 进制	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111	10000	10001
	2^0	2^1		2^2				2^3								2^4	
	1	2		4				8								16	

一个课后题：

$v1 = 0x123;　　//它的实际大小其实是：291

$result = octdec($v1); //结果为：17，怎么理解？推理如下：

1，octdec($v1)

2，octdec(291)　　//因为$v1 的实际值就是 291

3，octdec(“291”); //因为 octdec()函数要求输入一个字符串，这属于自动转换

4，octdec(“21”);　//因为 octdec()函数要求输入一个 8 进制数字字符串，而 9 不是合法的数字，忽略掉

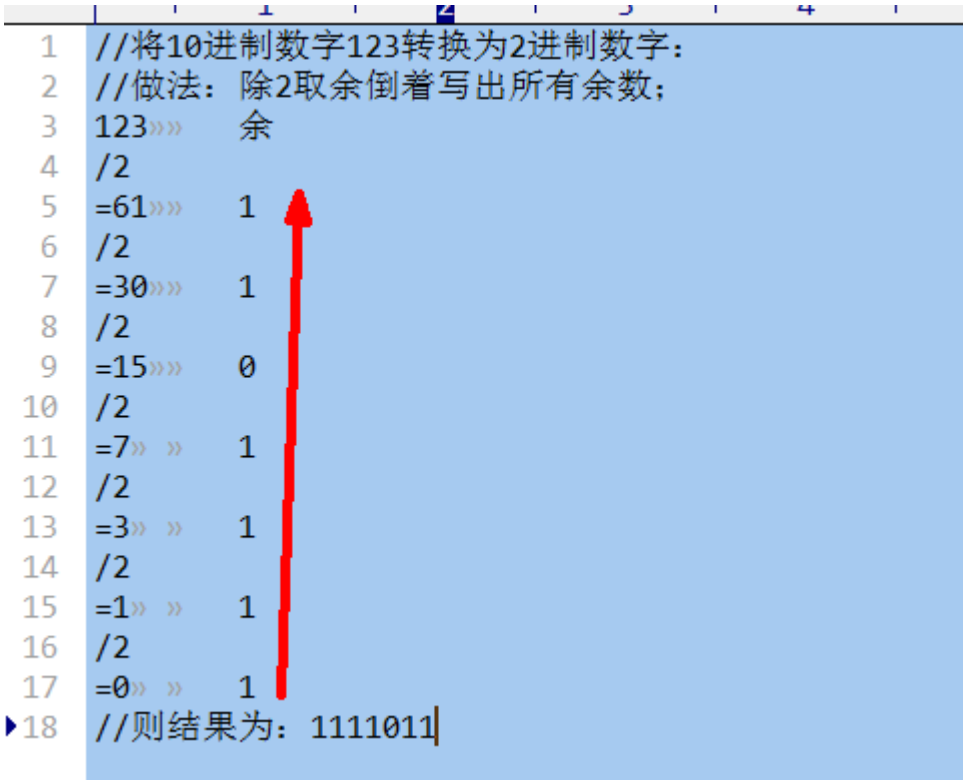
5，结果，8 进制数字”21”转换为 10 进制就是就是 17；

进制转换的人工计算——了解其原理

10 进制转换为 2 进制：

做法：除 2 取余倒着写出所有余数，就是对应的 2 进制数字形式；

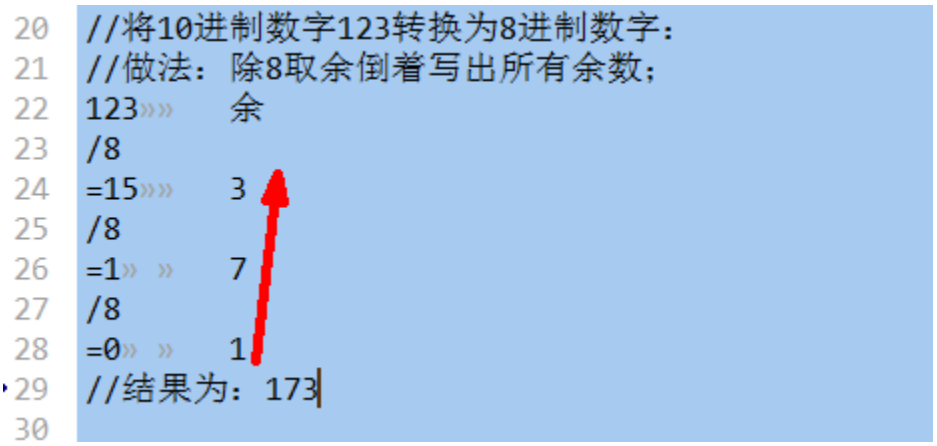
详细解释：将一个 10 进制数字除以 2，得到商和余数，如果商还大于等于 2，则继续除以 2，继续得到商和余数，以此类推，直到商为 0 为止，然后将前面的所有余数按倒序写出来就是对应的 2 进制数字。



10 进制转换为 8 进制：

做法：除 8 取余倒着写出所有余数，就是对应的 8 进制数字形式；

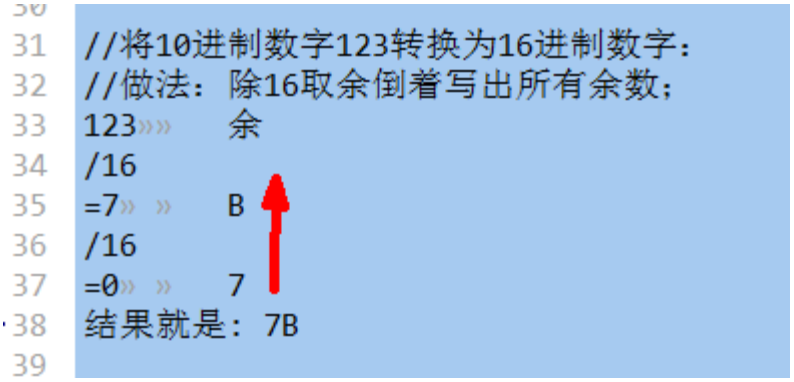
详细解释：将一个 10 进制数字除以 8，得到商和余数，如果商还大于等于 8，则继续除以 8，继续得到商和余数，以此类推，直到商为 0 为止，然后将前面的所有余数按倒序写出来就是对应的 8 进制数字。



10 进制转换为 16 进制：

做法：除 16 取余倒着写出所有余数，就是对应的 16 进制数字形式；

详细解释：将一个 10 进制数字除以 16，得到商和余数，如果商还大于等于 16，则继续除以 16，继续得到商和余数，以此类推，直到商为 0 为止，然后将前面的所有余数按倒序写出来就是对应的 16 进制数字。



其他进制转换为 10 进制的做法：

先看一种对数字大小和“数字权值”的理解：

对一个 10 进制数字： 1234，可以这样去理解它的大小：

$1234 = 1*10^3 + 2*10^2 + 3*10^1 + 4*10^0 = 1000 + 200 + 30 + 4;$ (任何数的 0 次方都是 1）

这里，我们对 10^3， 10^2， 10^1， 10^0 等等，称为“权值”；每个位的权值是不同的。

对于 10 进制，每个位上的权值，就是 10 的 n 次方；

对于 8 进制，每个位上的权值，就是 8 的 n 次方；

对于 16 进制，每个位上的权值，就是 16 的 n 次方；

对于 2 进制，每个位上的权值，就是 2 的 n 次方；

8 进制转换 10 进制：

将 8 进制数字的每个位上的数字乘以其对应位上的权值，然后相加之后的结果。

举例：有一个 8 进制数字 123，则其实际大小为：

$1 * 8^2 + 2 * 8^1 + 3 * 8^0 = 64 + 16 + 3 = 83;$

16 进制转换 10 进制：

将 16 进制数字的每个位上的数字乘以其对应位上的权值，然后相加之后的结果。

举例：有一个 16 进制数字 123，则其实际大小为：

$1 * 16^2 + 2 * 16^1 + 3 * 16^0 = 256 + 32 + 3 = 291;$

2 进制转换 10 进制：

将 2 进制数字的每个位上的数字乘以其对应位上的权值，然后相加之后的结果。

举例：有一个 2 进制数字 101011，则其实际大小为：

$1 * 2^5 + 0 * 2^4 + 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0 = 32 + 0 + 8 + 0 + 2 + 1 = 43;$