

位运算符的应用—管理一组事物的开关状态

开关状态定义：只有 2 中结果，对应就是布尔值类型的值。True false

管理目标：使用一个变量就可以表达若干个灯泡的“当前状态”，具体有 3 个任务：

1.通过该变量，可以获得任何一个数据（灯泡）当前状态：

```
7  define("D1", 1);    //对应二进制的值为：00000001
8  define("D2", 2);    //对应二进制的值为：00000010
9  define("D3", 4);    //对应二进制的值为：00000100
10 define("D4", 8);    //对应二进制的值为：00001000
11 define("D5", 16);   //对应二进制的值为：00010000
12 //定义个变量，该变量代表5个灯泡的任意值组合
13 $state=10; //对应二进制值：00001010
14 //
15 //任务1：指定任意一个灯泡的当前状态，并输出
16 if (($state & D1) > 0) {
17     echo "<br/>灯亮";
18 }else{
19     echo "<br/>灯灭";
20 }
```

2.所有灯都整体显示状态：

```
34 function ShowAll(){
35     for ($i=1; $i <=4 ; $i++) {
36         $s="D".$i;
37         //GLOBALS['state'] 调用外部函数state
38         if (($GLOBALS['state']&constant($s))>0) {
39             echo "<br/>灯{$s}亮";
40         } else {
41             echo "<br/>灯{$s}灭";
42         }
43     }
44 }
45 ShowAll()
```

3.通过该变量，可以将一个一个数据的状态“开启”，即：开启任意一个指定灯泡：

```
47 // 开启指定灯泡
48 $state=$state | D1;
49 echo "<br/>开启灯1,后的状态";
50 ShowAll();
```

4.通过该变量，可以将一个一个数据的状态“关闭”，即：关闭任意一个指定灯泡：

```
51 // 关闭指定灯泡
52 $state=$state & (~D1);
53 echo "<br/>关闭灯1,后的状态";
54 ShowAll();
```

数组运算符

+: 数组联合：即“数组串联”

将右边的数组项并到左边数组后边，得到数组，如有重复键，则结果以左边为准

```
$arr1 = array(5=>10, 8=>20, 10=>30);
```

```
$arr2 = array(3=>33, 2=>22);
```

```
$r1 = $arr1 + $arr2; //结果为： array(5=>10, 8=>20, 10=>30, 3=>33, 2=>22)
```

`==`: 如果两个数组具有相同的键名和键值（可以顺序不同，或类型不同），则返回 `true`
`!=`:
`===`: 如果两个数组具有相同的键名和键值且顺序和类型都一样，则返回 `true`
`!==`:

错误控制运算符

通常就用一个地方: `$link = @mysql_connect("localhost","root","123");`

作用: 如果该链接数据库的语句失败, 则屏蔽该失败的错误提示!

运算符优先级

1. 要有运算符优先级意识
2. 括号最优先, 赋值最落后
3. 先乘除后加减
4. 大致: 单项运算符 > 算术运算符 > 比较运算符 > 逻辑运算符(出了“非”运算符)

流程控制

if 条件控制: 如果表达式 `expr` 的值为 `true`, 则执行 `statement` 语句, 否则就跳过该语句继续往下执行。

```
if (expr) {  
    statement;  
}
```

if ... else 条件控制: 如果表达式 `expr` 的值为 `true`, 则执行 `statement1` 语句, 否则执行 `statement2`。

```
if (expr) {  
    statement1;  
} else {  
    statement2;  
}
```

Switch ... case 分支控制语句: Switch 语句根据 `variable` 的值, 依次与 `case` 中的 `value` 值相比较, 如果不相等, 继续查找下一个 `case`; 如果相等, 就执行对应语句, 直到 `switch` 语句结束或遇到 `break` 为止, 一般 `switch` 语句结尾都有一个默认的 `default`, 如果在前边 `case` 没找到相符合的条件, 则输出默认语句, 这跟 `else` 语句类似。

```
switch (variable) {  
    case 'value1':  
        statement1;  
        break;  
    case 'value2':  
        ...  
    default:  
        default statement n;  
        break;  
}
```

for 循环: 嵌套循环, 九九乘法表

```
for ($i=1; $i <=9 ; $i++) {  
    for ($k=1; $k <= $i ; $k++) {  
        echo "$i x $k =" . ($i*$k) . "&nbsp;&nbsp;&nbsp;";  
    }  
    echo "<br/>";  
}
```