

算术运算符与自增运算

1. 算术运算符

符号有：+、-、*、/、%

说明：

- (1) 他们都是针对数字进行的运算
- (2) 如果他们的两边有不是数字的数据，就会自动转换为数字
- (3) 其中取余(模)运算，他只对整数进行运算，如果不是，会向下取整后运算

2. 自增自减运算

- (1) 常规：对数字进行自增 1 或自减 1

字符串：只能自增，且自增的效果就是下一字符，且只能针对字母或数字

布尔值递增递减无效

null 递减无效，递增为 1

```
$s1 = "a";
$s2 = "A";
$s3 = "abc";
$s4 = "xyz";
$s5 = "xyzz";
$s6 = "zzz";
$s7 = "abc9";
$s8 = "9z";

$s1++;
$s2++;
$s3++;
$s4++;
$s5++;
$s6++;
$s7++;
$s8++;

echo "$s1, $s2, $s3, $s4, $s5, $s6, $s7, $s8";
```



- (2) 前自增和后自增

```
<?php
$a1 = 123;
$a2 = 123;
$a1++;
++$a2;
echo "a1=$a1;a2=$a2". "<br>";

echo "a1=".++$a1."<br>";
echo "a2=".$a2++."<br>";
?>
```



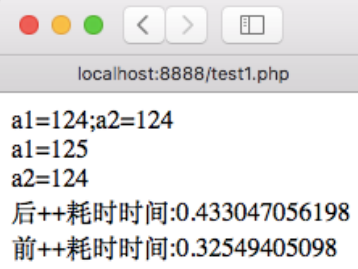
可见，在有加加运算的其他语句中，前加加和后加加会有区别，影响其他语句的执行结果，前加加是对自加变量加 1，然后做其他运算，而后加加是先做其他运

算，然后再对自加变量加 1。通常，我们在循环中，推荐使用前加加，因为使用前加加的代码执行效率更高，实验如下：

```
$t1 = microtime(true); // 获取当前时间，精确到万分之一秒
for ($i=0; $i < 10000000 ; $i++) {
    # code...
}

$t2 = microtime(true); // 获取当前时间，精确到万分之一秒
for ($i=0; $i < 10000000 ; ++$i) {
    # code...
}

$t3 = microtime(true); // 获取当前时间，精确到万分之一秒
echo "后++耗时时间:".($t2-$t1)."<br>";
echo "前++耗时时间:".($t3-$t2)."<br>";
```



a1=124;a2=124
a1=125
a2=124
后++耗时时间:0.433047056198
前++耗时时间:0.32549405098