PART6

- 浮点类型

浮点数的2种表示形式

1， 常规写法：带小数点。
$v1 = 123.456;

2， 科学计数法：带一个特殊符号“E”
$v1 = 123.456E2;　　//含义为：123.456乘以10的2次方；
$v2 = 123.456E3;　　//含义为：123.456乘以10的3次方，虽然结果是123456，但仍然是“浮点型”
$v3 = 123E4;　　//含义为：123乘以10的4次方，还是浮点数。

- 浮点数使用的细节知识

浮点数不应进行大小比较

造成这一个问题的原因是：
1，所有数字，最终的表示形式，都是2进制！！！
2，大多数浮点数的2进制形式，不能完全表达准确，最终只能“以很高的精度接近理论值”
3，因此，浮点数比较是不可靠。

php中输出其实是做了一定的处理之后的显示结果，而js的输出是该计算结果的“真实反映”。

考虑实际应用所需的精度的情况下，去将要比较的浮点数，转换为整数之后再比较。
比如：
要求精度为3为小数，则都乘以1000，然后取整后比较；
要求精度为4为小数，则都乘以10000，然后取整后比较

当整数运算的结果超出整数的范围后，会自动转换为浮点数（了解）。
获取一个数据（变量）的类型的函数有：
getType($变量);　　返回的是该类型的名字（字符串）；
var_dump($变量):　　会输出该变量的类型，数据内容，（以及长度）；

示例如下：

```php
<html>
<head>
<meta http-equiv="content-Type" content="text.html"; charset="utf-8">
</head>
<body>
<?php
$v1 = 8.1;  //浮点数
if( $v1/3 == 2.7){
    echo $v1 . "/3 等于 2.7";
}
else{
    echo $v1 . "/3 不等于 2.7";
}
echo "<hr />";
//以下为正确的浮点数比较方法：
//考虑精度要求为4位时：
if( round($v1 * 10000/3) == round(2.7 * 10000) ){
    echo $v1 . "/3 等于 2.7";
}
else{
    echo $v1 . "/3 不等于 2.7";
}
echo "<hr />";
$n1 = 10000;
$s1 = $n1 * $n1;    //两个整数相乘
$s2 = $n1 * $n1 * $n1;  //3个整数相乘
echo "<br />整数的最大值为：" . PHP_INT_MAX;
echo "<br />";  var_dump($s1);
echo "<br />";  var_dump($s2);
echo "<hr />";
echo "<br />php中输出：8.1/3的结果为：" . (8.1/3);
?>
</body>
</html>
<script>
    document.write("<br />但js中输出：8.1/3的结果为：" + ( 8.1/3) );
</script>

```

结果如下：

```
8.1/3 不等于 2.7
---------------------------------------------------------------
8.1/3 等于 2.7
---------------------------------------------------------------

整数的最大值为：2147483647
int(100000000)
float(1000000000000)
---------------------------------------------------------------

php中输出：8.1/3的结果为：2.7
但js中输出：8.1/3的结果为：2.6999999999999997
```

- 字符串

有如下4形式：
形式1：双引号字符串：
$str1 = “字符串内容.....”；

形式2：单引号字符串：
$str2 = ‘字符串内容.....’；

形式3：双引号定界符字符串：
$str3 = <<<”标识符A”
字符串内容....
标识符A;

形式4：单引号定界符字符串：
$str4 = <<<’标识符B’
字符串内容...
标识符B;

实际上，单引号字符串中，只有最后一个“\”才是必须进行转义的。

- 布尔类型

单词是bool，boolean。
其只有2个数据：true，false；
布尔类型的一个常见应用情形是：对一个变量直接进行判断，比如if判断

这里的判断，永远是指：判断该变量（数据）“是否为真”。
对于这种情况，只有如下数据是被当做“假”（false）：
0，0.0，“”，“0”，null，array()，false，还有一个是“未定义的变量”
其余都是真。
参考：
手册》附录》PHP类型比较表》

- 类型转换

自动转换：
在任何运算中，如果需要某种类型的数据，而给出的数据不是该类型，通常都会发生自动转换：将该类型转换为目标需要的类型。
比如： octdec($x), bindec($x)，hexdec($x)；//这里就要求$x必须是字符串，如果不是，就会转换；
$v1 = 1 + “2”； //此时也发生了自动转换。

- 强制转换：

自动类型转换是由“运算符”或类似运算符的语句来决定的。
而：
强制类型转换，仅仅是一个简单的语法：
形式：（目标类型)数据；
含义：将该数据转换为设定的目标类型；

通常的转换目标类型有：
(int)，(float)，(string)，(bool)，(array)，(object)

上述强制类型转换，并不改变该变量的本身数据或类型。
对应，有一个语法是直接改变改变本的数据（及类型）：
settype($变量名，“目标类型”);

- 类型相关的函数

var_dump()：用于输出变量的“完整信息”，几乎只用于调试代码。
getType($变量名)：获取该变量的类型名字，返回的是一个表示该类型名字的字符串，比
如：“string”，“bool”，“double”，“int”
setType($变量名，“目标类型”)：将该变量强制改变为目标类型；
isset(), empty(), unset();。。。。省略！
is_XX类型() 系列函数：判断某个数据是否为某种类型，有如下一些：
    is_int($x); 判断$x是否是一个整数类型；
    is_float($x);
    is_string($x);
    is_bool($x);
    is_array($x);
    is_object($x);
    is_null($x);
    is_numeric($x); 判断$x是否是一个数字！
    is_scalar($x); 判断$x是否是一个“标量类型”