PHP小白极速入门 —— 数据类型

主讲人与课程设计：耕耕

数据类型

1. 整型
2. 浮点型
3. 字符串型
4. 布尔型
5. NULL
6. 数组型
7. 对象型

Integer 整型

```php
<?php
$a = 1234; // 十进制数
$a = -123; // 负数
$a = 0123; // 八进制数（等于十进制 83）
$a = 0x1A; // 十六进制数（等于十进制 26）
$a = 0b11111111; // 二进制数字（等于十进制 255）
```

Float 浮点型

```php
<?php
$a = 1.234;
$b = 1.2e3;
```

NAN 属于float

浮点型类型转换类似于先将值转换成整型，然后再转换成浮点

Boolean 布尔类型

要指定一个布尔值，使用常量 TRUE 或 FALSE。两个都不区分大小写

```
$foo = True; // 设置 $foo 为 TRUE
```

String 字符串

一般情况是**单引号和双引号，**还有heredoc 语法结构，nowdoc 语法结构（ <<< ）

```php
<?php
echo 'this is a simple string<br>';

$a = 100;
echo "this is a simple string $a";
```

```php
$f = 200;
echo "this is {$f}";
```

注意：不能有空格，还有只有双引号里面可以带变量

NULL

特殊的 NULL 值表示一个变量没有值，NULL 类型唯一可能的值就是 NULL

1. 被赋值为 NULL
2. 尚未被赋值
3. 被 unset()

```php
<?php
$var = NULL;
```

Array 数组

PHP 中的数组实际上是一个有序映射。映射是一种把 values 关联到 keys 的类型

```
array(  key =>  value , ...)
// 键（key）可是是一个整数 integer 或字串 string
// 值（value）可以是任意类型的值


<?php
$array = array(
    "foo" => "bar",
    "bar" => "foo",
);

// 自 PHP 5.4 起
$array = [
    "foo" => "bar",
    "bar" => "foo",
];
```

Array 数组

key 会有如下的强制转换

1. 包含有合法整型值的**字符串**会被转换为整型。例如键名 "8" 实际会被储存为 8。但是 "08" 则不会强制转换，因为其不是一个合法的**十进制**数值
2. **浮点数**也会被转换为整型，意味着其小数部分会被舍去。例如键名 8.7 实际会被储存为 8
3. **布尔值**也会被转换成整型。即键名 true 实际会被储存为 1 而键名 false 会被储存为 0
4. **Null** 会被转换为空字符串，即键名 null 实际会被储存为""
5. **NaN** 会被转换为空字符串
6. **数组和对象不能被用为键名**。坚持这么做会导致警告：Illegal offset type

Array 数组

如果在数组定义中多个单元都使用了同一个键名，则只使用了最后一个，**之前的都被覆盖了**

```php
$array = array(
    1    => "a",
    "1"  => "b",
    1.5  => "c",
    true => "d",
);
var_dump($array);

/* array(1) {
    [1]=>
    string(1) "d"
  } */
```

Array 数组

key 为可选项，如果未指定，PHP 将自动使用之前用过的**最大 integer 键名加上 1** 作为新的键名

```php
$array = array("foo", "bar", "hallo", "world");
$array2 = ["foo", "bar", "hallo", "world"];
var_dump($array2);
```

```php
$array = array(
    "a",
    "b",
6 => "c",
    "d",
);
var_dump($array);
```

```
/* array(4) {
    [0]=>
    string(1) "a"
    [1]=>
    string(1) "b"
    [6]=>
    string(1) "c"
    [7]=>
    string(1) "d"
  } */
```

访问数组Array

数组单元可以通过 **array[key] 或者array{key}**语法来访问

```php
$array = array(
    "foo" => "bar",
    "bar" => "foo",
);

var_dump($array["foo"]);
var_dump($array{"bar"});
```

修改或者新增数组Array

数组单元可以通过 **array[key] 或者array{key}**语法来修改或者新增；要删除某键值对，对其调用 **unset()** 函数

```php
$arr = array(5 => 1, 12 => 2);

$arr[] = 56;    // This is the same as $arr[13] = 56;
$arr[5] = 666;
unset($arr[5]); // 不会重建索引
var_dump($arr);
```

什么是对象

对象是复合型的数据类型，你可以理解为对象是你的“**对象**”

一个对象

特征：有胡子，有墨镜，蓝裤子等等

行为：唱歌，挥手，跳舞等等

总结：PHP中的任何对象**可以**拥有特征（属性）和行为（方法）

Object 对象 (先了解)

要创建一个新的对象 object，使用 new 语句实例化一个类

对象访问，用->符合，如果访问的是一个数字类型的用->{'1'}

```php
class foo
{
    function do_foo()
    {
        echo "Doing foo.";
    }
}

$bar = new foo();
$bar->do_foo();
```

```php
$a = ["a","b"];
$b = (object)$a;
var_dump($b->{'1'});
```

Object 对象 (先了解)

```php
class Student
{
    public $myUniversity = "北京大学";
    public $myMajor;

    public function __construct($major)
    {
        $this->myMajor = $major;
    }
    public function chooseMajor()
    {
        echo 111;
    }
}

$s1 = new Student("软件工程");
var_dump($s1->myUniversity);
var_dump($s1->myMajor);
```

Resource 资源类型 (先了解)

比如打开一个文件，或者是链接一个数据库等

```php
$fp = fopen("test.txt", "r");
var_dump($fp);
fclose($fp);
```

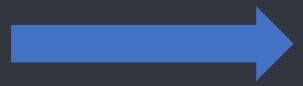
PHP小白极速入门 ——类型转换

主讲人与课程设计：耕耕

PHP数据类型转换

第一种转换:

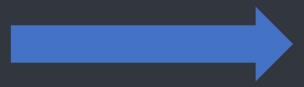
(**int**) 、 (**integer**) : 转换成整形
(**float**) 、 (**double**) 、 (**real**) : 转换成浮点型
(**string**) : 转换成字符串
(**bool**) 、 (**boolean**) : 转换成布尔类型
(**array**) : 转换成数组
(**object**) : 转换成对象



```php
$a = "123.1";
var_dump((float)$a);
```

第二种转换:

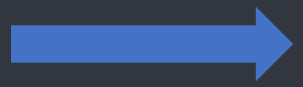
intval() floatval() strval()



```php
$b = "123.1";
var_dump(floatval($b));
```

第三种转换:

settype()



```php
$c = "123.1";
settype($c, "float");
var_dump($c);
```

Integer 整型 (类型转换)

如果给定的一个数超出了 integer 的范围，将会被解释为 **float**

```php
var_dump(PHP_INT_MAX + 1);
```

将 resource 转换成 integer 时， 结果会是 PHP 运行时为 resource 分配的唯一资源号

```php
$fp = fopen("test.txt", "r");

var_dump((int)$fp);
```

从布尔值转换，FALSE 将产生出 0（零），TRUE 将产生出 1（壹）。

```php
$a = true;
var_dump((int) $a);
```

NAN将产生出 0（零）

Integer 整型 (类型转换)

字符串转换为数值

如果该字符串没有包含 **'.'，'e' 或 'E'** 并且其数字值在整型的范围之内（由 PHP_INT_MAX 所定义），该字符串将被当成 integer 来取值。其它所有情况下为PHP_INT_MAX

该字符串的**开始部分决定了它的值**。如果该字符串以合法的数值开始，则使用该数值。否则其值为 0（零）

```
$foo = 1 + "10.5";                // $foo is float (11.5)
$foo = 1 + "-1.3e3";              // $foo is float (-1299)
$foo = 1 + "bob-1.3e3";           // $foo is integer (1)
$foo = 1 + "bob3";                // $foo is integer (1)
$foo = 1 + "10 Small Pigs";       // $foo is integer (11)
$foo = 4 + "10.2 Little Piggies"; // $foo is float (14.2)
$foo = "10.0 pigs " + 1;          // $foo is float (11)
$foo = "10.0 pigs " + 1.0;        // $foo is float (11)
```

Integer 整型 (类型转换)

当从浮点数转换成整数时，将向下取整（小数部分拿掉）

```php
$a = 3.2; //3
var_dump((int) $a);
```

Null转换

```php
$a = null; //0
var_dump((int) $a);
```

Boolean 布尔类型 (类型转换)

用 (bool) 或者 (boolean) 来强制转换

当转换为 boolean 时，以下值被认为是 **FALSE**:

1. 布尔值 FALSE 本身
2. 整型值 0（零）
3. 浮点型值 0.0（零）
4. 空字符串，以及字符串 "0"
5. 不包括任何元素的数组
6. 特殊类型 NULL（包括尚未赋值的变量）

所有其它值都被认为是 TRUE（包括任何资源 和 NAN）

Boolean 布尔类型 (类型转换)

```php
var_dump((bool) "");        // bool(false)
var_dump((bool) 1);         // bool(true)
var_dump((bool) -2);        // bool(true)
var_dump((bool) "foo");     // bool(true)
var_dump((bool) 2.3e5);     // bool(true)
var_dump((bool) array(12)); // bool(true)
var_dump((bool) array());   // bool(false)
var_dump((bool) "false");   // bool(true)
var_dump((bool) NAN);       // bool(true)
```

String 字符串 (类型转换)

1. 一个布尔值 boolean 的 **TRUE** 被转换成 string 的 **"1"**。Boolean 的 **FALSE** 被转换成 **""**（空字符串）
2. 一个整数 integer 或浮点数 float 被转换为数字的字面样式的 string（看到什么转换什么）
3. 数组 array 总是转换成字符串 "Array“
4. NULL 总是被转变成空字符串
5. 资源 resource 总会被转变成 "Resource id #1" 这种结构的字符串

```php
$a = 12.3;
$a = 12;
$a = [];
var_dump((string)$a);
```

数组类型转换

对于任意 integer，float，string，boolean 和 resource 类型，如果将一个值转换为数组，将得到一个仅有一个元素的数组，其下标为 0，该元素即为此标量的值

```php
$a = 12.3;
var_dump((array)$a); // [12.3]
```

如果一个 object 类型转换为 array，则结果为一个数组，其单元为该对象的属性。键名将为成员变量名

```php
class Name
{
    public $age1 = 100;
    private $age2 = 200;
    protected $age3 = 300;
}

$a = new Name();
$result = (array)$a;
print_r($result);
var_dump($result);
```

```
Array
(
    [age1] => 100
    [Nameage2] => 200
    [*age3] => 300
)
array(3) {
  ["age1"]=>
  int(100)
  ["Nameage2"]=>
  int(200)
  ["*age3"]=>
  int(300)
}
```

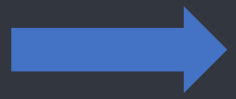
Object 对象 (类型转换)

array 转换成 object 将使键名成为属性名并具有相对应的值

```php
$obj = (object) array('1' => 'foo');
var_dump(isset($obj->{'1'}));
```

对于其他值，会包含进成员变量名 **scalar**

```php
$obj = (object) 'hao';
var_dump($obj);
```



```
object(stdClass)#1 (1) {
    ["scalar"]=>
    string(3) "hao"
  }
```