# 1 Curso POO PHP Arrays

## 1.1 Arrays

Un array é un tipo de datos que nos permite almacenar varios valores. Cada membro do array almacénase nunha posición á que se fai referencia utilizando un valor clave. As claves poden ser numéricas ou asociativas.

Ademais de asignando valores directamente, un array pode ser creado usando o construtor da linguaxe **array**. Este toma certo número de parellas clave = > valor como argumentos. Se nos parámetros non se indica o valor da clave, crea un array numérico (con base 0). Se non se lle pasa ningún parámetro, crea un array baleiro.

Para facer referencia aos elementos almacenados nun array, tes que utilizar o valor clave entre corchetes.

```
<?php
$cor = array (
    "vermello"  => "#FF0000",
    "verde" => "#00FF00",
    "azul"   => "#0000FF"
);
echo "O código da cor azul é {$cor['azul']}";
?>
```

A partir de PHP 5.4 tamén se pode usar a sintaxe de array curta, que substitúe array con **[]**.

```
<?php
$cor = [
    "vermello"  => "#FF0000",
    "verde" => "#00FF00",
    "azul"   => "#0000FF"
];
echo "O código da cor azul é {$cor['azul']}";
?>
```

En PHP podes crear tamén arrays de varias dimensións almacenando outro array en cada un dos elementos dun array. Para facer referencia aos elementos almacenados nun array multidimensional, debes indicar as claves para cada unha das dimensións.

```
<?php
$a = array (
    "froitas"   => array("a" => "laranxa", "b" => "plátano", "c" => "mazá"),
    "números"  => array(1, 2, 3, 4, 5, 6),
    "ordinais" => array(2 => "segundo", "terceiro")
);
echo "Hoxe de {$a['ordinais'][2]} prato comín unha {$a['froitas']['c']}";
?>
```

En PHP non é necesario que indiques o tamaño do array antes de crealo. Nin sequera é necesario indicar que unha variable concreta é de tipo array. Simplemente podes comezar a asignarlle valores. Tampouco é necesario que especifiques o valor da clave. Se a omites, o array irase enchendo a partir da última clave numérica existente, ou da posición 0 se non existe ningunha.

```
<?php
$a[4] = 20;
$a[] = 56;    // É o mesmo que $a[5] = 56;
?>
```

Tamén se poden eliminar elementos dun array utilizando a función **unset**. No caso dos arrays numéricos, eliminar un elemento significa que as claves deste xa non estarán consecutivas.

As cadeas de texto pódense tratar como arrays nos que se almacena unha letra en cada posición, sendo 0 o índice correspondente á primeira letra, 1 o da segunda, etc.

```
<?php
$a = "Aprendendo PHP";
for($pos=0;$pos<14;$pos++) echo $a[$pos];
?>
```

### 1.1.1 Percorrer arrays

Para percorrer os elementos dun array, en PHP podes usar un bucle específico: **foreach**. Utiliza unha variable temporal para asignarlle en cada iteración o valor de cada un dos elementos do array. Podes usalo de dúas formas. Percorrendo só os elementos, ou percorrendo os elementos e os seus valores crave de forma simultánea.

```
<?php
$ordinais = [2 => "segundo", "terceiro"];
foreach($ordinais as $a) echo $a."<br />";
foreach($ordinais as $num => $a) echo $num." - ".$a."<br />";
?>
```

Pero en PHP tamén hai outra forma de percorrer os valores dun array. Cada array mantén un punteiro interno, que se pode utilizar con este fin. Utilizando funcións específicas, podemos avanzar, retroceder ou iniciar o punteiro, así como recuperar os valores do elemento (ou da parella clave / elemento) ao que apunta o punteiro en cada momento. Algunhas destas funcións son:

| Función | Resultado |
|---|---|
| **reset** | Sitúa o punteiro interno ao comezo do array |
| **next** | Avanza o punteiro interno unha posición |
| **prev** | Move o punteiro interno unha posición atrás |
| **end** | Sitúa o punteiro interno ao final do array |
| **current** | Devolve o elemento da posición actual |
| **key** | Devolve a crave da posición actual |
| **each** | Devolve un array coa crave e o elemento da posición actual. Ademais, avanza o punteiro interno unha posición |

La función each devuelve un array con cuatro elementos. Los elementos 0 y 'key' almacenan el valor de la clave en la posición actual del puntero interno. Los elementos 1 y 'value' devuelven el valor almacenado.

Si el puntero interno del array se ha pasado de los límites del array, la función each devuelve false, por lo que la puedes usar para crear un bucle que recorra el array.

```
<?php
$ordinais = [2 => "segundo", "terceiro"];
while ($ordinal = each($ordinais))
echo $ordinal[1]."<br />"
?>
```

### 1.1.2 Funcións para manexo de arrays

A función **array_values** recibe un array como parámetro, e devolve un novo array cos mesmos elementos e con índices numéricos consecutivos con base 0.

Para comprobar se unha variable é de tipo array, utiliza a función **is_array**. Para obter o número de elementos que contén un array, tes a función **count** (ou **sizeof**, que é un alias de count).

Se queres buscar un elemento concreto dentro dun array, podes utilizar a función **in_array**. Recibe como parámetros o elemento a buscar e a variable de tipo array na que buscar, e devolve true se encontrou o elemento ou false no caso contrario.

Outra posibilidade é a función **array_search**, que recibe os mesmos parámetros pero devolve a clave correspondente ao elemento, ou false se non o encontra.

E se o que queres buscar é un crave nun array, tes a función **array_key_exists**, que devolve true ou false.

Na documentación de PHP podes consultar a lista completa de funcións que podes empregar para traballar con arrays.

--Víctor Lourido 14:43 25 jun 2013 (CEST)