Los arrays

Programando en PHP

Introducción

Introducción

- Los Arrays son una parte muy importante de cualquier lenguaje de programación.
- Permiten:
  - Manejar grupos de valores relacionados
  - Almacenar múltiples valores en una sola estructura y bajo un mismo nombre.
- Muchas de las funciones de PHP devuelven un array de valores.
- En PHP los arrays están muy ligados a las bases de datos y formularios.
- Tipos de arrays:
  - Posicionales
  - Asociativos.

Arrays posicionales

Arrays posicionales

- Formados por un conjunto de valores ordenados respecto a un índice numérico.
- El índice entero, indica la posición del elemento en el conjunto.
- Formas de asignar un array:
  - La más sencilla ➔ Asignar posición por posición
  - Utilizando la función **array()**
  - Utilizando los **[]** como en JavaScript

Asignación de arrays posicionales

```php
$array1[0]=12;
$array1[1]="verde";
$array1[2]=25.4;
$array1[3]="vivo";
$array1[]="Riviera";

$array2 = array (12, "verde", 25.4, "vivo", "Riviera");

$array3 = [12, "verde", 25.4, "vivo", "Riviera"];
```

Posicion	0	1	2	3	4
Array 1	12	verde	25.4	vivo	Riviera
Array 2	12	verde	25.4	vivo	Riviera
Array 3	12	verde	25.4	vivo	Riviera

Arrays sociativos

Arrays asociativos

- Formados por un conjunto de valores ordenados respecto a un índice de tipo String y no entero.
- Formas de asignar un array:
  - La más sencilla ➔ Asignar posición por posición
  - Utilizando la función **array()**. En este caso será necesario indicar el nombre de la posición.
  - Utilizando los **[]** como en JavaScript. De nuevo, habrá que indicar el nombre de la posición

Asignación de Arrays asociativos

```php
$array1["posi1"]=12;
$array1["posi2"]="verde";
$array1["posi3"]=25.4;
$array1["posi4"]="vivo";
$array1["posi5"]="Riviera";

$array2 = array ("posi1"=>12, "posi2"=>"verde", "posi3"=>25.4,
"posi4"=>"vivo", "posi5"=>"Riviera");

$array3 = ["posi1"=>12, "posi2"=>"verde", "posi3"=>25.4,
"posi4"=>"vivo", "posi5"=>"Riviera"];
```

Posicion	Posi1	Posi2	Posi3	Posi4	Posi5
Array 1	12	verde	25.4	vivo	Riviera
Array 2	12	verde	25.4	vivo	Riviera
Array 3	12	verde	25.4	vivo	Riviera

Arrays multidimensionales

Arrays multidimensionales

- PHP nos permite definir arrays multidimensionales mediante la combinación de arrays unidimensionales (tanto posicionales como asociativos)

- Veamos ejemplos:

Asignación de arrays multidimensionales

```php
$matriz1[0][0]="Peseta";
$matriz1[0][1]=166.386;
$matriz1[1][0]="Dólar";
$matriz1[1][1]=0.96;

$matriz2[0] = array ("Peseta", 166.386);
$matriz2[1] = array ("Dólar", 0.96);

$matriz3 = array (array ("Peseta", 166.386), array ("Dólar", 0.96));
```

matrices

	Moneda	Cambio €
$matriz1[0]	Peseta	166.386
$matriz1[1]	Dólar	0.96
$matriz2[0]	Peseta	166.386
$matriz2[1]	Dólar	0.96
$matriz3[0]	Peseta	166.386
$matriz3[1]	Dólar	0.96

Recorrer Arrays posicionales

Recorrer Arrays posicionales

- Lo más habitual cuando se trabaja con arrays es recorrerlos para obtener sus elementos.
- La forma más sencilla de hacerlo es utilizando bucles.
- Problema: Debemos conocer a priori el tamaño.
  - **count (array)**
    - Devuelve el número de elementos que hay en el array.

```php
for ($i=0; $i<count($array); $i++)
{
    echo $array[$i];
}
```

Recorrer Arrays posicionales

- Pero la función COUNT puede inducir a error muy fácilmente:

```php
$a[0] = 1;
$a[1] = 3;
$a[2] = 5;
$result = count($a);
// $result == 3

$b[0]  = 7;
$b[5]  = 9;
$b[10] = 11;
$result = count($b);
// $result == 3

$result = count(null);
// $result == 0

$result = count(false);
// $result == 1
```

Recorrer Arrays posicionales

- Sizeof(array)
  - Devuelve el número de elementos del array (es un alias de count)

```php
for ($i=0; $i<sizeof($array); $i++)
{
    echo $array[$i];
}
```

Recorrer Arrays asociativos

Recorrer Arrays asociativos

- Para recorrer arrays asociativos ya no basta con saber el número de elementos.
- Además debemos saber las claves para poder acceder a ellos.
- Para recorrer arrays asociativos se utilizan 3 funciones específicas:

Recorrer Arrays asociativos

- **current (array)**
  - Devuelve el valor de la posición actual del puntero dentro del array.
  - Devuelve false cuando está al final del array o si no hay elementos.
- **next(array)**
  - Devuelve el valor del elemento siguiente al actual (si existe) y avanza el puntero.
  - Si el elemento actual era el último, devuelve false.
- **prev (array)**
  - Devuelve el valor del elemento anterior al actual (si existe) y retrocede el puntero.
  - Si el elemento actual era el primero, devuelve false.
- **key(array)**
  - Devuelve el indice de la posición actual del array.
  - Un número si el array es posicional o un string si el array es asociativo.

Recorrer Arrays asociativos

```php
<?php
    $trans = array('pie', 'bici', 'coche', 'avión');
    $posi = current($trans); // $posi = 'pie';
    $posi = next($trans);    // $posi = 'bici';
    $posi = current($trans); // $posi = 'bici';
    $posi = prev($trans);    // $posi = 'pie';
    $posi = end($trans);     // $posi = 'avión';
    $posi = current($trans); // $posi = 'avión';
?>
```

Recorrer Arrays asociativos: Ejercicio

- Crear un array asociativo y recorrerlo, mostrando para cada valor cual es el nombre de su posición.

Posición	Valor
Nombre	Juan
Altura	1.85
Edad	25
Pelo	Moreno
Ciudad	Granada

```php
$persona = array('Nombre'=>'Juan', 'Altura'=>1.85,
                 'Edad'=>25, 'Pelo'=>'Moreno',
                 'Ciudad'=>'Granada');
echo "<td>".key($persona)."</td>";
echo "<td>".current($persona)."</td>";
echo "</tr><tr>";
while (next($persona))
{
    echo "<td>".key($persona)."</td>";
    echo "<td>".current($persona)."</td>";
    echo "</tr><tr>";
}
```

Recorrer Arrays asociativos: Ejercicio

```php
do
{
    echo "<td>".current($posiciones)."</td>";
    echo "<td>".current($valores)."</td>";
    next($valores);
    echo "</tr><tr>";
}while (next($posiciones));
```

```php
for ($i=0; $i<count($persona); $i++)
{
    $posi = key($persona);
    $valor = current($persona);
    echo "<td>$posi</td><td>$valor</td>";
    echo "</tr><tr>";
    next($persona);
}
```

Recorrer Arrays asociativos

- **end(array)**
  - Coloca el puntero interno en la última posición de un array.
- **reset(array)**
  - Coloca el puntero interno en la primera posición.
  - Devuelve el primer elemento del array

```
<?php
    echo "<td bgcolor = 'pink'>".key($vector1)."</td>";
    while (next($vector1))
    {
        echo "<td>".key($vector1)."</td>";
    }
?>
</tr>
<tr align = 'center'>
<td bgcolor = 'pink'>Valor</td>
<?php
    echo"<td>".reset($vector1)."</td>";
    while (next($vector1))
    {
        echo "<td>".current($vector1)."</td>";
    }
?>
```

Recorrer Arrays asociativos

- **array_keys(array)**
  - Devuelve un nuevo array posicional con todas las claves que forman el array.

- **array_values(array)**
  - Devuelve un nuevo array posicional con todos los valores que forman parte del array.

```php
$claves = array_keys($vector1);
$valores = array_values($vector1);
for($i=0; $i<count($claves); $i++)
{
    echo "<tr align='center'><td>".$claves[$i]."</td>";
    echo "<td>".$valores[$i]."</td></tr>";
}
```

Recorrer Arrays asociativos: Ejercicio

- Crear un array asociativo y utilizando las funciones array_keys y array_values junto con los bucles necesarios, mostrarlo de la siguiente forma:

Nombre	Altura	Edad	Pelo	Ciudad
Juan	1.85	25	Moreno	Granada

Recorrer Arrays asociativos: Ejercicio

```php
<?php
    $persona = array('Nombre'=>'Juan', 'Altura'=>1.85,
                     'Edad'=>25, 'Pelo'=>'Moreno',
                     'Ciudad'=>'Granada');
    $posiciones = array_keys($persona);
    $valores = array_values($persona);
    echo "<tr>";
    for ($i=0; $i<count($persona); $i++)
    {
        echo "<td>$posiciones[$i]</td>";

    }
    echo "</tr><tr>";
    for ($i=0; $i<count($persona); $i++)
    {
        echo "<td>$valores[$i]</td>";

    }
    echo "</tr>";

?>
```

Ordenar arrays

Ordenar Arrays

- **sort(array)**
  - Ordena alfabéticamente los valores.
  - De menor a mayor.
  - Se pierde la relación entre índice y valor
- **rsort(array)**
  - Ordena de forma inversa a sort.

```php
<?php
    $frutas = array("limón", "naranja",
            "platano", "albaricoque");
    sort($frutas);
    foreach ($frutas as $clave => $valor) {
        echo "<td>frutas[" . $clave . "]</td>";
        echo "<td> $valor </td>";
        echo "<tr></tr>";
    }
?>
```

Posicion	Valor
frutas[0]	albaricoque
frutas[1]	limón
frutas[2]	naranja
frutas[3]	platano

Ordenar Arrays

- **asort(array)**
  - Ordena igual que sort, pero mantiene la relación entre índice y valor.

- **arsort(array)**
  - Ordena de forma inversa a asort.

Diferencias entre sort y asort

Vector sin ordenar

Posición	Valor
0	Madrid
1	Zaragoza
2	Bilbao
3	Valencia
4	Lérida
5	Alicante

Vector ordenado con sort

Posición	Valor
0	Alicante
1	Bilbao
2	Lérida
3	Madrid
4	Valencia
5	Zaragoza

Vector ordenado con asort

clave	Valor
5	Alicante
2	Bilbao
4	Lérida
0	Madrid
3	Valencia
1	Zaragoza

Ordenar Arrays

```php
$vector = array ('d'=> 'Madrid', 'c'=> 'Zaragoza',
                 'e'=> 'Bilbao', 'b'=> 'Valencia',
                 'f'=> 'Lérida', 'a'=> 'Alicante');
```

- **ksort(array)**
  - Ordena alfanuméricamente las claves de un array de menor a mayor.
  - Mantiene las relaciones entre índice y valor.
- **krsort(array)**
  - Ordena de forma inversa a ksort.

Vector sin ordenar

Posición	Valor
d	Madrid
c	Zaragoza
e	Bilbao
b	Valencia
f	Lérida
a	Alicante

Vector ordenado con ksort

Posición	Valor
a	Alicante
b	Valencia
c	Zaragoza
d	Madrid
e	Bilbao
f	Lérida

Modificar Arrays

Modificar Arrays

- **array_merge(array1, array2)**
  - Combina en un solo array los valores de los dos arrays recibidos.
  - Los elementos se añaden siempre al final.
  - Si estamos combinando arrays asociativos:
    - Las claves con el mismo valor NO se añaden al array.
    - Se actualizan al último valor dado.

Modificar Arrays

```php
$vector1= array ("altura"=>'10',"anchura"=>'15', "unidad" =>"cm");
$vector2= array ('1', '2', '3');
$vector3= array ('100', '100', "unidad"=>"px");
$vectorTotal = array_merge($vector1, $vector2, $vector3);
```

Vector 1

altura	anchura	unidad
10	15	cm

Vector 2

0	1	2
1	2	3

Vector 3

0	1	unidad
100	100	px

Suma de Vector1 + Vector2 + Vector3

altura	anchura	unidad	0	1	2	3	4
10	15	px	1	2	3	100	100

Otras funciones de Arrays

Otras funciones de Arrays

- **array_reverse (array)**
  - Devuelve el array pasado como parámetro, pero con sus componentes en orden inverso.
- **range (limite_inf, limite_sup [,salto])**
  - Permite crear arrays de valores "secuenciales"
  - Devuelve un array con los valores comprendidos entre el primer y el segundo argumento, ambos incluidos.
  - Salto indica el incremento entre los valores devueltos.

Otras funciones de Arrays

```php
$vector = range (0, 10);
$vector2 = range (4, 10);
$vector3 = range (4, 20, 2);
$vector4 = range ('a', 'h');
```

range (0, 10)	0-1-2-3-4-5-6-7-8-9-10-
range (4, 10);	4-5-6-7-8-9-10
range (4, 20, 2);	4-6-8-10-12-14-16-18-20
range ('a', 'h');	a-b-c-d-e-f-g-h

Otras funciones de Arrays

- **in_array(elemento_buscar, array)**
  - Nos dice si un elemento está dentro de un array o no.
- **compact()**
  - Recibe como argumento una lista de variables que han sido definidas previamente.
  - Devuelve un array en el que los índices son los nombres de las variables y el contenido, sus correspondientes valores.

Hay muchísimas más funciones de Arrays

php.net

```php
$n1 = 3;
$n2 = 9;
$n3 = 11;
$vector_total = compact ("n1", "n2", "n3");
```

Posicion	Valor
n1	3
n2	9
n3	11

Un poco de conocimiento extra

Números aleatorios

- **mt_rand(inf, sup)**
  - Devuelve un número aleatorio entre el límite inferior y el límite superior.

mt_rand()	1355297302	2108842525	1454051968	1595398012	723173578
mt_rand(3,8)	6	4	5	5	5
mt_rand(0,1)	0	0	0	1	1
mt_rand(-9, 15)	8	7	13	9	-2

Los arrays

Programando en PHP