

## TEMA 2

### PHP: ARRAYS, ARRAYS ASOCIATIVOS, MATRICES

#### ARRAYS, TABLAS O MATRICES

Los arrays o tablas o listas de valores están formados por varios valores que PHP cuenta internamente. El contador se llama índice o clave y siempre se inicia en 0.

PHP establece muy pocas limitaciones en las estructuras de los arrays.

La mayoría de lenguajes permiten definir y manipular datos del mismo tipo (arrays de enteros, arrays de cadenas, etc) esta restricción no existe realmente en PHP.

PHP permite mezclar en un array valores de diferentes tipos. Incluso permite que los índices de los elementos de un mismo array sean de diferentes tipos.

Hay dos maneras de definir un array:

- forma desarrollada
- forma abreviada

**FORMA DESARROLLADA:**

- tanto el valor índice como el valor de la variable se escriben en una línea propia,
- por cosa que es más fácil de leer .

Ejemplo: *arrayDiaSemana.php*

```
$dia[0]="Lunes";  
$dia[1]="Martes";  
$dia[2]="Miércoles";  
$dia[3]="Jueves";  
$dia[4]="Viernes";  
$dia[5]="Sábado";  
$dia[6]="Domingo";  
  
echo "Visualizo un elemento<br>";  
echo $dia[3];  
echo "<br>";
```

**FORMA ABREVIADA:**

- Después de array se incluye entre **paréntesis** la lista de valores .
- Todos los valores están separados por comas y la asignación de las claves se realiza automáticamente.

Ejemplo: *arrayDiaSemana.php*

```
$dia=array("Lunes","Martes","Miércoles","Jueves","Viernes","Sábado","Domingo");  
echo "Visualizo un elemento<br>";  
echo $dia[3];  
echo "<br>";
```

## RECORRIDO DE UN ARRAY CONVENCIONAL:

PHP soporta varias fórmulas para recorrer un array

Archivo *vectores.php*

- Recorrido mediante FOR NUMÉRICO:

```
$vect=array("rojo","verde","azul","blanco");  
$num_elementos= count($vect);  
echo "Recorrido array con FOR<br>";  
for ($i = 0; $i <$num_elementos; $i++)  
{  
    echo $vect[$i];  
    echo "<br>";  
}
```

- Recorrido mediante WHILE NUMÉRICO:

```
$vect=array("rojo","verde","azul","blanco");  
$num_elementos= count($vect);  
echo "Recorrido array con WHILE<br>";  
$i=0;  
while ($i <$num_elementos)  
{  
    echo $vect[$i];  
    echo "<br>";  
    $i++;  
}
```

**Recorrido mediante FOREACH:**

```
<?php
$array = array(1, 2, 3, 4);
foreach ($array as $valor)
{
    echo $valor;
    echo "<br>";
}
?>
```

- **Recorrido mediante WHILE- LIST-EACH:**

```
$array = array("uno", "dos", "tres");
reset($array);
while (list($clave, $valor) = each($array))
{
    echo "Clave: $clave; Valor: $valor </br>";
}
```

- **Recorrido mediante WHILE-RESET-NEXT:**

```
$personas = array("Pepito", "Juanito", "Andresito", "Felipito", "Gerardito");  
echo "Contenido del vector de cadenas con WHILE-RESET-NEXT: <br>";  
$cadena=reset($personas);  
while ($cadena)  
{  
    echo $cadena;  
    $cadena=next($personas);  
    echo "<br>";  
}
```

- **Recorrido mediante WHILE-END-PREV:**

```
$personas = array("Pepito", "Juanito", "Andresito", "Felipito", "Gerardito");  
echo "El vector de cadenas a la INVERSA con WHILE-END-PREV:<br>";  
$cadena=end($personas);  
while ($cadena)  
{  
    echo $cadena;  
    $cadena=prev($personas);  
    echo "<br>";  
}
```

- Recorrido mediante WHILE-EACH:

```
$vect = array("uno", "dos", "tres");  
reset($vect); // obligatorio posicionar al usar list()  
while ($el = each($vect))  
{  
    echo $el['key'].': '.$el['value'];//el índice key es equivalente a 0  
                                     //y el índice value a 1  
    echo "<br>";  
}
```

---

## ARRAYS ASOCIATIVOS

En PHP también es posible trabajar con arrays asociativos. En este tipo de arrays , no se utilizan como índices los números, sino que se utiliza una 'clave' que se asocia a cada elemento. En este caso también se puede utilizar una forma desarrollada y una abreviada.

- **Forma abreviada:**

```
$capital=array(  
    "ES"=>"Madrid",  
    "DE"=>"Berlín",  
    "PL"=>"Varsovia",  
    "FR"=>"París"  
);  
echo "Accedemos a un elemento:<br>";  
echo $capital["DE"];
```

- **Forma desarrollada:**

```
$capital["ES"]="Madrid";  
$capital["DE"]="Berlín";  
$capital["PL"]="Varsovia";  
$capital["FR"]="Paris";  
  
echo "Accedemos a un solo elemento";  
echo $capital["DE"];
```

## RECORRIDO DE ARRAY ASOCIATIVO



*Archivo **vectoresAsociativos.php***

- Recorrido mediante FOR NUMÉRICO:

```
$mi_array = array(  
    'uno' => 'uno.gif',  
    'dos' => 'dos.gif',  
    'tres' => 'tres.gif',  
    'cuatro' => 'cuatro.gif'  
);  
  
$limite=sizeof($mi_array);  
reset($mi_array);  
  
for ($i = 1; $i <= $limite; $i++)  
{  
    echo '<br/>Clave: "'. key($mi_array) .'" ===== Valor: "'.current($mi_array);  
    next($mi_array);  
}
```

- **Recorrido mediante FOREACH CLAVE-VALOR:**

*vectoresRecorrer.php*

```
$equipo = array('portero'=>'Cech', 'defensa'=>'Terry', 'medio'=>'Lampard', 'delantero'=>'Torres');

foreach($equipo as $posicion=>$jugador)
{
    echo "El " . $posicion . " es " . $jugador;
    echo "<br>";
}
```

- **Recorrido mediante FOREACH VALOR:**

```
// Foreach valor:
echo "Accedemos mediante Foreach valor:<br>";
foreach ($equipo as $value)
{
    echo $value;
    echo "<br>";
}
```

- **Recorrido mediante WHILE-LIST-EACH:**

```
<?php
$fruit = array('a' => 'apple', 'b' => 'banana', 'c' => 'cranberry');

reset($fruit);
while (list($key, $val) = each($fruit)) {
    echo "$key => $val\n";
}
?>
```

Salida:

```
a => apple
b => banana
c => cranberry
```

- **Recorrido mediante WHILE-RESET-EACH:**

**Forma 1:** Accediendo al índice y a valor a través de \$vect[0] y \$vect[1]

```
$fruits=array('a'=>'apple', 'b'=>'banana', 'c'=>'cranberry');  
echo "Recorre vector asociativo con WHILE-RESET-EACH (forma1)<br>";  
reset($fruits);  
while ($vect=each($fruits))  
{  
  
    echo "índice: ".$vect[0]." -- ";  
    echo "valor: ". $vect[1]."<br>";  
}
```

**Forma 2:** Accediendo al índice y a valor a través de \$vect["key"] y \$vect["value"]

Ver ambos comportamientos de each ()

```
$fruits=array('a'=>'apple', 'b'=>'banana', 'c'=>'cranberry');  
echo "Recorre vector asociativo con WHILE-RESET-EACH (forma2)<br>";  
reset($fruits);  
while ($vect=each($fruits))  
{  
  
    echo "índice: ".$vect["key"]." -- ";  
    echo "valor: ". $vect["value"]."<br>";  
}
```

## ARRAYS MULTIDIMENSIONALES

Para acceder a los elementos de un array multidimensional el mecanismo es el mismo que para acceder a arrays unidimensionales, basta con añadir tantos corchetes al final del nombre de la variable como dimensiones tenga.

**Nota:** el primer elemento se encuentra en la posición 0

Los siguientes ejemplos se encuentran en:

*ejemploArrayBidimensional.php:*

### Matrices o arrays de dos dimensiones

- Inicialización de Arrays Bidimensionales en la **forma desarrollada**:

```
$M1[0][0] = 'a' ;  
$M1[0][1] = 'b';  
$M1[0][2] = 'c';  
$M1[1][0] = 'd';  
$M1[1][1] = 'e';  
$M1[1][2] = 'f';
```

- Inicialización de Arrays Bidimensionales en la **forma abreviada**:

```
$M2= array(  
    array(0,1,2),  
    array(3,4,5),  
    array(6,7,8),  
);
```

**ACCESO DIRECTO A UN ELEMENTO De UNA MATRIZ:**

Se realiza a través de su posición en cada una de las dimensiones

```
$fila=1;  
$col=2;  
  
echo 'Acceso al elemento $M1['.$fila.']['.$col.']='.$M1[$fila][$col].'<br>;'
```

## RECORRIDO COMPLETO De UNA MATRIZ.

- Recorrido mediante FOR ANIDADO

```
$num_filas=2;
$num_col=3;
echo 'Visualizamos con FOR anidado las filas y columnas de la MATRIZ<br>';
for ($i = 0; $i <$num_filas; $i++)
{
    for ($j = 0; $j<$num_col; $j++)
    {
        echo    $M1[$i][$j]. '<br>';
    }
}
```

Nota: Se verán otras maneras de recorrer matrices más adelante.

## ARRAYS DE DOS DIMENSIONES DE CADENAS

### Ejemplo 1:

Con el fin de ver las funciones que tiene PHP para la manipulación de matrices , utilizaremos como ejemplo un listado de alumnos , de los cuales almacenaremos una serie de datos personales.

```
/* Declaración implícita de una matriz BIDIMENSIONAL */  
  
$M1[0][0] = 'José';           // es el nombre del primer alumno  
$M1[0][1] = 'María';         // es el nombre del segundo alumno  
$M1[0][2] = 'Juan';          // es el nombre del tercer alumno  
  
echo $M1[0][1]."<br>";
```



## ARRAYS DE DOS DIMENSIONES ASOCIATIVOS

### Ejemplo 1:

Ahora utilizaremos una matriz asociativa con clave-nombre. Cada Fila representa e

```
/* Declaración implícita de una matriz BIDIMENSIONAL ASOCIATIVA */
```

```
$M1[0]['Nombre'] = 'José';
```

```
$M1[1]['Nombre'] = 'María';
```

```
$M1[2]['Nombre'] = 'Juan';
```

```
echo $M1[0]['Nombre']."<br>";
```

**Ejemplo 2:**

Ahora utilizaremos un array bidimensional asociativa para almacenar los alumnos del curso.

Las claves serán:

- *Nombre*
- *Teléfono*
- *Correo*
- *Nota*.

**Inicialización** array asociativo de dos dimensiones:

```
/* Declaración de una matriz BIDIMENSIONAL ASOCIATIVA forma IMPLÍCITA con función array */
$curso= array
(
array('Nombre' =>'José','Telefono' =>'123456789','Correo' =>'a@a.es','Nota'=>'10'),
array('Nombre' =>'María','Telefono' =>'876543219','Correo' =>'b@b.es','Nota' =>'0'),
array('Nombre' =>'Juan','Telefono' =>'432187659','Correo' =>'c@c.es','Nota' =>'0')
);
```

**Acceso** a los elementos array asociativo de dos dimensiones:

```
/* Acceso a una matriz BIDIMENSIONAL ASOCIATIVA */

echo "Los datos de la primera persona del Array bidimensional son:<br>";
echo $curso[0]['Nombre']."<br>";
echo $curso[0]['Telefono']."<br>";
echo $curso[0]['Correo']."<br>";
echo $curso[0]['Nota']."<br>";
```

Se podría pasar toda la información de un alumno de la matriz anterior **Scurso** a un vector

**Ejemplo 3:**

Pasaremos la información del alumno tercero al vector \$alu3, intermediando:

```
$alu3 = $curso[2];
```

```
// Pasamos todos los datos de un alumno concreto a un vector
$i=2;
$alu3 = $curso[$i];
echo "Visualizamos los datos del alumno $i:<br>";
foreach ($alu3 as $key => $value)
{
    echo "Clave:$key ==> Valor:$value";
    echo "<br>";
}
```

## RECORRIDO DE ARRAY MULTIDIMENSIONAL

Ejemplo:

*vectoresRecorrer.php*

\$equipo\_fútbol: Es un array que contiene otros 3 arrays con jugadores de fútbol.

```
$equipo_futbol = array  
(  
    array("Rooney", "Chicharito", "Gigs"),  
    array("Suarez", "Messi", "Villa"),  
    array("Torres", "Terry", "Etoo")  
);
```

- Recorrido mediante FOR anidado

Lo puedo recorrer con dos bucles **FOR**, uno por cada índice:

```
for($i=0; $i<3; $i++)  
{  
    for($j=0; $j<3; $j++)  
    {  
        echo $equipo_futbol[$i][$j];  
    }  
}
```

- **Recorrido mediante FOREACH:**

Dos *foreach*, uno por cada dimensión:

```
foreach($equipo_futbol as $equipo) //
{
    foreach($equipo as $jugador)
    {
        echo $jugador ." ";
    }
}
```

- **Recorrido mediante WHILE-LIST-EACH:**

**Nota:** no es realmente una función, es un constructor del lenguaje. **list()** se utiliza para asignar una lista de variables en una sola operación.

```
echo "Recorrido vector con While-List-Each clave-valor<br>";
$i=0;
while($i < count($curso))
{
    reset($curso[$i]);
    while (list($clave,$valor) = each($curso[$i]))
    { // recorre cada par clave-valor de la fila actual
        echo "$clave $valor<br>";
    }
    $i++;
}
```

- Recorrido de una matriz cuando se DESCONOCEN NÚMERO DE FILAS Y/O COLUMNAS

```
$num_filas= count($M1); // número de elementos de la primera dimensión
$num_col= max(array_map('count', $M1)); // elementos segunda dimensión

echo 'Visualizamos con FOR anidado y COUNT-ARRAY_MAP '
. 'las filas y col de la MATRIZ<br>';

for ($i = 0; $i <$num_filas; $i++)
{
    for ($j = 0; $j<$num_col; $j++)
    {
        echo    $M1[$i][$j]. '<br>';
    }
}
```

## FUNCIONES RELACIONADAS con ARRAYS:

### *each()*:

Devuelve el par clave/valor actual de un array y avanza el cursor del array. En verdad devuelve un vector con dos formas de acceso:

- Para la clave : índice 0 o “key”
- Para el valor: índice 1 o “value”

Devuelve false si no quedan elementos

[http://www.w3schools.com/php/func\\_array\\_each.asp](http://www.w3schools.com/php/func_array_each.asp)

### *count()*

Para recorrer una matriz indexada necesitamos saber por adelantado el número de elementos que la componen. Esta información nos la proporciona la función *count()*.

<http://php.net/manual/es/function.count.php>

Con la siguiente sintaxis:

```
int count ( mixed $array_or_countable [, int $mode = COUNT_NORMAL ] )
```

Donde *\$array* representa la variable de la cual se quiere obtener el número de elementos .

- Si se trata de una matriz, tanto indexada como asociativa, la función devolverá el número de elementos .
- Si se trata de una variable que contiene un elemento, sea o no matriz, devolverá 1.
- Si la variable no tiene asignado ningún valor o tiene asignado el valor null, devolverá un 0.

### *sizeof()*

Existe otra función de PHP que devuelve el número de elementos , esta función es `sizeof()`

<http://php.net/manual/es/function.sizeof.php>

y su sintaxis es la siguiente:

```
int sizeof(array matriz)
```

Para recorrer una matriz indexada tan solo tenemos que construir un bucle, que empieza en la posición 0 y que acabará cuando ya no quedan más elementos que tratar. En cada iteración del bucle se accede a una posición diferente.

Como ya sabemos, PHP soporta varias fórmulas para recorrer un array

- Recorrido mediante un **FOR**
- Recorrido mediante un **WHILE**
- Recorrido mediante **FOREACH**