

Tema 2 : Matrices



Ciclo Superior DAW

Asignatura: Desarrollo web en entorno servidor

Curso 20/21

Introducción



- En este capítulo veremos cómo utilizar arrays y matrices para almacenar y recuperar conjuntos de datos.
- Aprenderemos la instrucción Foreach y cómo utilizarla para iterar matrices

Matrices



- Una matriz nos permite asociar valores con claves, pudiendo acceder a un valor almacenado en la matriz sabiendo la clave asociada.
- Una clave puede ser un número entero o un string
- Un buen ejemplo es un almacén donde cada producto tiene un identificador



Matrices. Creación/Modificación

- Un array puede ser creado por la palabra reservada **array()**.
- Se toma un cierto número de parejas clave => valor separadas con coma.
- Sintaxis:

```
array (clave => valor
    , ...
)
// clave puede ser un integer o string
// valor puede ser cualquier valor

<?php
$matriz = array ("fruta" => "manzana", 12 => true);
echo $matriz["fruta"]; // manzana
echo $matriz [12]; // 1
?>
```



Matrices. Creación/Modificación

- Otra forma posible de crearla o modificarla es definir valores explícitamente en la matriz

```
$matriz[clave] = valor;  
$matriz[] = valor;  
// clave puede ser un integer o string  
// valor puede ser cualquier valor
```



Matrices. Creación/Modificación

- Si no hay ninguna clave especificada, entonces se toma el máximo de los índices enteros existentes, y la nueva clave será ese valor máximo + 1.
- Si no existen índices enteros aún, la clave será 0 (cero).
- Si especifica una clave que ya tenía un valor asignado, el valor será sustituido.



Matrices. Creación/modificación

- A continuación veremos un ejemplo completo de creación y modificación de una matriz

```
<?php
$matriz = array(5 => 1, 12 => 2);
$matriz[] = 56;      // Es igual que $matriz[13] = 56;
                    // en este punto del script
$matriz["x"] = 42;   // Agrega un nuevo elemento a la
                    // matriz con la clave "x"
$matriz["x"] = 54;   // nuevo valor para la clave "x"
unset($matriz[5]);   // Esto elimina el elemento de la matriz
unset($matriz);      // Esto elimina la matriz completa
?>
```



Actividad 1

- *Crearemos dos matrices con 3 elementos (usando las dos formas)*
 - *Biblioteca (ID -> Libro)*
 - *Equipo (Dorsal -> Nombre)*
- *Añadiremos un elemento a cada matriz*
- *Modificaremos un elemento existente en la matriz*
- *Mostraremos por pantalla el contenido de la matriz*



Estructuras. Foreach

- Es una instrucción que nos permite realizar bucles For anidados de una manera más sencilla
- También es un modo fácil de iterar sobre matrices
- Recorre la matriz seleccionando cada elemento



Estructuras. Foreach

- Sintaxis:

```
foreach(expresion_array as $value) sentencia  
foreach(expresion_array as $key => $value) sentencia
```

- La primera forma recorre el array dado por `expresion_array`. En cada iteración, el valor del elemento actual se asigna a `$value` y el puntero interno del array se avanza en una unidad.
- La segunda manera hace lo mismo, salvo que la clave del elemento actual será asignada a la variable `$key` en cada iteración.



Estructuras. Foreach

- Algunos ejemplos:

```
/* foreach ejemplo 2: valor (con clave impresa para ilustrar) */
$a = array(1, 2, 3, 17);
$i = 0;
foreach($a as $v) {
    print "\$a[$i] => $v.\n";
    $i++;
}
```



Estructuras. Foreach

- Algunos ejemplos:

```
/* foreach ejemplo 3: clave y valor */
$a = array(
    "uno" => 1,
    "dos" => 2,
    "tres" => 3,
    "quince" => 15
);
foreach($a as $k => $v) {
    print "\$a[$k] => $v.\n";
}
```



Estructuras. Foreach

- Algunos ejemplos:

```
/* foreach ejemplo 4: matriz multi-dimensional */
$a[0][0] = "a";
$a[0][1] = "b";
$a[1][0] = "e";
$a[1][1] = "z";
foreach($a as $v1) {
    foreach ($v1 as $v2) {
        print "$v2\n";
    }
}
```



Actividad 2

En el siguiente ejercicio:

- *Recorreremos las matrices que hemos definido en la Actividad 1*
- *Modificaremos las matrices haciendo que se cambie el nombre de los futbolistas y el título de los libros*
 - *¿Por qué no podemos cambiar el dorsal ni el ID?*



Matrices. Funciones útiles. Unset

- Su sintaxis es:

```
Unset ()
```

- La función **unset()** permite eliminar la definición de claves de una matriz.
- La matriz NO es reindexada, es decir, si eliminamos un elemento, el siguiente no ocupa su sitio.



Matrices. Funciones útiles. Unset

- Ejemplo:

```
<?php
$a = array(1 => 'uno', 2 => 'dos', 3 => 'tres');
unset($a[2]);
/* Producirá una matriz que fuese definida como
   $a = array(1 => 'uno', 3 => 'tres');
   y NO como:
   $a = array(1 => 'uno', 2 =>'tres');
*/
$b = array_values($a);
// Ahora $b es array(0 => 'uno', 1 =>'tres')
?>
```




Matrices. Funciones útiles. Explode

- Su sintaxis es:

```
array explode (string $separador, string $cadena [, int $limite])
```

- **Devuelve una matriz de cadenas**
- Cada cadena devuelta es una subcadena según el separador.
- Si se especifica límite, la matriz devuelta contendrá un máximo de elementos con el último conteniendo el resto de la cadena.



Matrices. Funciones útiles. Explode

- Ejemplo:

```
<?php
// Ejemplo 1
$pizza = "trozo1 trozo2 trozo3 trozo4 trozo5 trozo6";
$trozos = explode(" ", $pizza);
echo $trozos[0]; // trozo1
echo $trozos[1]; // trozo2
?>
```



Matrices. Funciones útiles. Implode

- Su sintaxis es:

```
string implode (string $elemento_union, array $elementos)
```

- Devuelve una cadena que contiene una representación de todos los elementos de la matriz en el mismo orden, pero con la cadena elemento_union en medio de los mismos.



Matrices. Funciones útiles. Implode

- Por ejemplo:

```
<?php
$array = array('apellido', 'email', 'telefono');
$separado_por_comas = implode(",", $array);
echo $separado_por_comas; // apellido,email,telefono
?>
```



Matrices. Funciones útiles. Count

- Su sintaxis es:

```
int count (mixed $var [, int $mode])
```

- Cuenta todos los elementos de un array o matriz



Matrices. Funciones útiles. Count

- Por ejemplo:

```
<?php
$a[0] = 1;
$a[1] = 3;
$a[2] = 5;
var_dump(count($a));
```



Matrices. Funciones útiles. Sort

- Su sintaxis es:

```
void sort (array &$matriz [, int $ sort_flags])
```

- Esta función ordena una matriz. Los elementos estarán ordenados de menor a mayor cuando la función termine. Asigna nuevos índices en la matriz.



Matrices. Funciones útiles. Sort

- Por ejemplo:

```
<?php

$frutas = array("limón", "naranja", "platano", "albaricoque");
sort($frutas);
foreach ($frutas as $clave => $valor) {
    echo "frutas[" . $clave . "] = " . $valor . "\n";
}

?>
```




Matrices. Funciones útiles. Rsort

- Su sintaxis es:

```
void rsort (array &$matriz [, int $ sort_flags])
```

- Esta función ordena una matriz en orden inverso, de mayor a menor.
- Funciona igual que **sort**, pero en sentido contrario



Matrices. Funciones útiles. Rsort

- Por ejemplo:

```
<?php

$frutas = array("limón", "naranja", "platano", "albaricoque");
rsort($frutas);
foreach ($frutas as $clave => $valor) {
    echo "frutas[" . $clave . "] = " . $valor . "\n";
}

?>
```



Matrices. Funciones útiles. Shuffle

- Su sintaxis es:

```
bool shuffle (array &$array)
```

- Esta función se encarga de desordenar un array aleatoriamente



Matrices. Funciones útiles. Shuffle

- Por ejemplo:

```
<?php
    $dorsales = range(1, 25);
    shuffle($dorsales);
    foreach ($dorsales as $dorsal) {
        echo "El jugador lleva el dorsal: ".$dorsal;
    }
?>
```



Matrices. Funciones útiles. Rand

- Su sintaxis es:

```
int rand (void)
int rand(int $min, int $max)
```

- Esta función genera un número entero aleatorio. Se le puede pasar como argumento los límites entre los que generar el número aleatorio.



Matrices. Funciones útiles. Rand

- Por ejemplo:

```
<?php
    $numeroAleatorio = rand();
    $nota = rand(1, 10);

    echo "La nota está es: ".$nota;
?>
```

Actividad 3



En el siguiente ejercicio:

- *De la matriz de futbolistas, eliminaremos al primer elemento*
- *Obtendremos la matriz de la cadena “portero,defensa,mediocampista,delantero,entrenador”*
- *Obtendremos una cadena con todos los libros, separados por una coma.*
- *Contaremos cuántos futbolistas hay*
- *Ordenaremos los libros por orden alfabético (normal e inverso)*



Recomendaciones y detalles a evitar

- Veamos este script. ¿Qué fallo tiene?

```
<?php  
  
$frutas[manzana] = 'plátano';  
echo $frutas[manzana];  
  
?>
```

- Si lo ejecutamos, ¿funciona? ¿Por qué?



Recomendaciones y detalles a evitar

Esto está mal, pero funciona.

- Este código tiene una constante indefinida (manzana) en lugar de una cadena ('manzana' - con las comillas)
- Si se define una constante que, desafortunadamente para este código, tengan el mismo nombre, fallará.



Recomendaciones y detalles a evitar

- **Siempre deben usarse comillas alrededor de un índice de matriz tipo cadena literal.** Por ejemplo, se usará `$frutas['manzana']` y no `$frutas[manzana]`.
- Esto no quiere decir que siempre haya que usar comillas en la clave. **No se necesitará usar comillas con claves que sean constantes o variables,** ya que en tal caso PHP no podrá interpretar sus valores.



Recomendaciones y detalles a evitar

- Algunos ejemplos:

```
<?php
// Mostrar todos los errores
error_reporting(E_ALL);
$matriz = array('fruta' => 'manzana', 'vegetal' => 'zanahoria');

// Correcto
print $matriz['fruta'];    // manzana
print $matriz['vegetal']; // zanahoria

// Incorrecto.
print $matriz[fruta];     // manzana
```



Recomendaciones y detalles a evitar

- Algunos ejemplos:

```
//Ejemplo 2

// Definamos una constante para demostrar lo que pasa.
// Asignaremos el valor 'vegetal' a una constante llamada fruta.
define('fruta', 'vegetal');

// Veamos la diferencia ahora
print $matriz['fruta']; // manzana
print $matriz[fruta];   // zanahoria
```



Ejemplos matrices completas

```
<?php
$a = array('color' => 'roja',
           'sabor' => 'dulce',
           'forma' => 'redonda',
           'nombre' => 'manzana',
           4 // La clave será 0
           );

// Es equivalente a
$a['color'] = 'rojo';
$a['sabor'] = 'dulce';
$a['forma'] = 'redonda';
$a['nombre'] = 'manzana';
$a[] = 4; // La clave será 0
$b[] = 'a';
$b[] = 'b';
$b[] = 'c';
// Resultará en la matriz array(0 => 'a' , 1 => 'b' , 2 => 'c'),
// o simplemente array('a', 'b', 'c')
?>
```



Ejemplos matrices completas

```
<?php
// Array como mapa de propiedades
$mapa = array( 'versión'      => 4,
               'SO'          => 'Linux',
               'idioma'       => 'inglés',
               'etiquetas_cortas' => true
             );
// claves estrictamente numéricas
$matriz = array( 7,
                8,
                0,
                156,
                -10
              );
// Esto es lo mismo que array(0 => 7, 1 => 8, ...)

$cambios = array( 10, // clave = 0
                 5   => 6,
                 3   => 7,
                 'a'  => 4,
                 11, // clave = 6 (El índice entero máximo era 5)
                 '8'  => 2, // clave = 8 (entero!)
                 '02' => 77, // clave = '02'
                 0   => 12 // El valor 10 será sustituido por 12
               );

// matriz vacía
$vacio = array();
?>
```

Ejemplos matrices completas



```
<?php
$colores = array('rojo', 'azul', 'verde', 'amarillo');
foreach ($colores as $color) {
    echo "Le gusta el $color?\n";
}
?>
```

Tema 2 : Matrices



Ciclo Superior DAW

Asignatura: Desarrollo web en entorno servidor

Curso 20/21