

ÉRICA PALOMINO

# TEMA 4

# ARRAYS

**[erica.palomino@escuelaartegranada.com](mailto:erica.palomino@escuelaartegranada.com)**

ESCUELAARTEGRANADA

# INTRODUCCIÓN

Los arrays son muy relevantes en PHP gracias a los siguientes motivos:

- 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.



# ARRAYS

# DECLARACIÓN

Existen **tres formas principales** de declarar los arrays:

- Posición por posición.
- Con la función **array()**.
- Utilizando **[]**.

**Recordad que para los arrays asociativos hay que indicar la clave que se va a utilizar para la posición.**



# **MATRICES**



# MATRICES

Al igual que en JAVA, se definen mediante la combinación de distintos arrays.

```
$matriz1[0][0] = "Euros";  
$matriz1[0][1] = 10.30;  
$matriz1[0][0] = "Libras";  
$matriz1[0][0] = 8.42;
```

```
$matriz2[0] = array ("Euros", 10.30);  
$matriz2[1] = array ("Libras", 8.42);
```

```
$matriz3 = array ( array ("Euros", 10.30), array ("Libras", 8.42));
```



# **RECORRER ARRAYS**

# RECORRER ARRAYS POSICIONALES

Como hemos visto hasta ahora la forma más sencilla es utilizando bucles y apollándonos en la función **count()** para obtener el número de valores.

Aún así, tenemos que tener cuidado con un pequeño detalle:

```
$arr[0] = 1;  
$arr[3] = 2;  
$arr[7] = 3;  
$n = count($arr);  
//$n == 3
```



# RECORRER ARRAYS ASOCIATIVOS

Para recorrer este tipo de arrays necesitamos conocer las claves, debido a esto nos vamos a apoyar en punteros:

- **current(array)**: Devuelve el valor de la posición actual del puntero.
- **next(array)**: Devuelve el valor de la posición siguiente y avanza el puntero.
- **prev(array)**: Devuelve el valor de la posición anterior y retrocede el puntero.

Todas las funciones **devuelven false si no encuentran el elemento.**

# RECORRER ARRAYS ASOCIATIVOS

Otras funciones que tenemos disponibles son las siguientes:

- **key(array)**: Devuelve el valor de la clave de la posición actual del array.
- **end(array)**: Coloca el puntero en la última posición del array.
- **reset(array)**: Coloca el puntero en la primera posición de array, devolviendo su valor.
- **array\_keys(array)**: Devuelve un array con los valores de los índices del array.
- **array\_values(array)**: Devuelve los valores que están introducidos dentro del array.

## RECORRER ARRAYS - EJERCICIOS

Crea un array asociativo y recórrelo, mostrando para cada posición la clave y el valor asociado. Utiliza los 3 tipos de bucles. Muéstralo en una tabla.

Crear un array asociativo y utilizando las funciones `array_keys` y `array_values` junto con los bucles necesarios, muéstralo en una **tabla de 2 filas, mostrando la clave y el valor asociado.**



# **ORDENAR ARRAYS**

# ORDENAR ARRAYS

Tenemos las siguientes funciones:

- **sort(array)**: Ordena de menor a mayor, eliminando la relación entre índice y valor.
- **rsort(array)**: Como sort pero a la inversa.
- **asort(array)**: Cumple la misma función que sort, pero mantiene la relación índice => valor.
- **arsort(array)**: asort pero a la inversa.

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

Tenemos las siguientes funciones:

- **ksort(array)**: Ordena las claves alfanuméricamente de menor a mayor, manteniendo la relación entre índice y valor.
- **krsort(array)**: Como ksort pero a la inversa.

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



# **COMBINAR ARRAYS**

# COMBINAR ARRAYS

Si queremos combinar los contenidos de **dos o más** arrays, podemos utilizar la función **array\_merge(array1, array2, ...)**.

Dicha función devuelve un solo array con el contenido de los arrays modificados. Los elementos se añaden siempre al final.

En el caso de que se utilicen arrays asociativos, las claves con el **mismo nombre NO** se añaden **más de una vez** al array. En dicho caso, se **actualiza con el último valor dado**.



# COMBINAR ARRAYS

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**

## OTRAS FUNCIONES

- **array\_reverse (array)**: Devuelve el array pasado como parámetro, pero con sus componentes en orden inverso.
- **range (limite\_inf, limite\_sup [,salto])**: Devuelve un array con los valores comprendidos entre el primer y el segundo argumento, ambos incluidos.
- **in\_array(elemento\_buscar, array)**: Devuelve true si un elemento está dentro de un array.
- **compact()**: Recibe como argumento una lista de variables que han sido definidas previamente y devuelve un array en el que los índices son los nombres de las variables y el contenido, sus correspondientes valores.