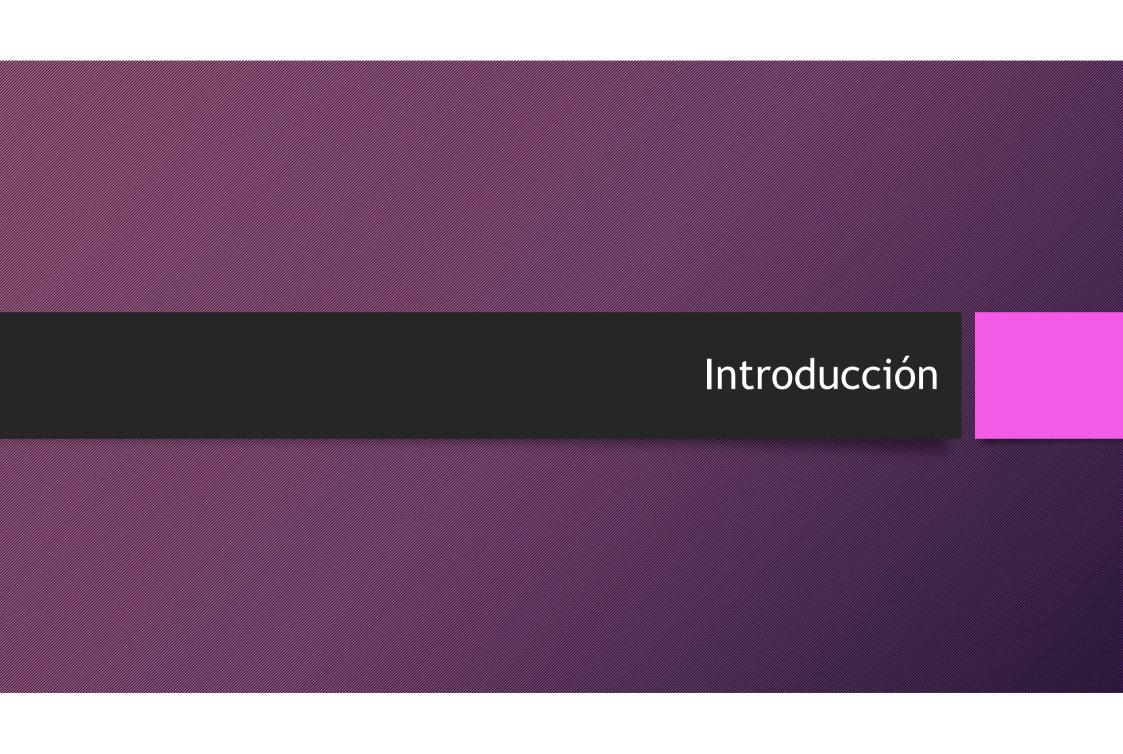
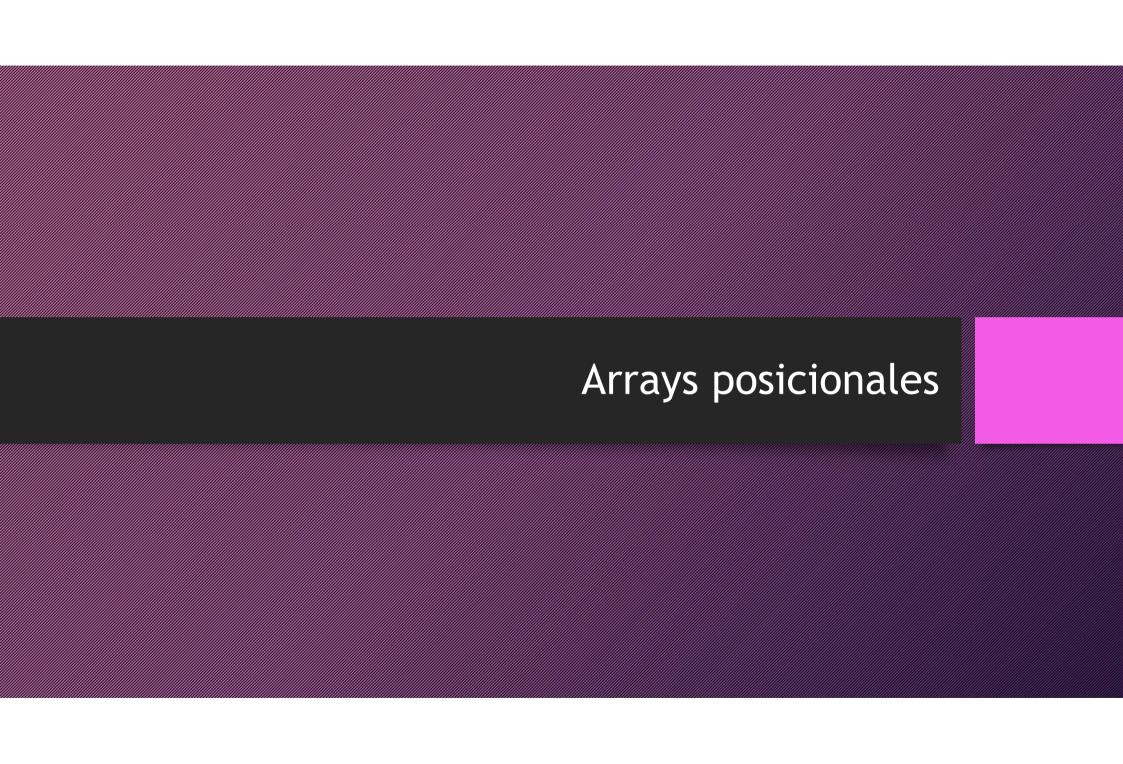
# Los arrays

Programando en PHP



#### 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

- Formados por un conjunto de valores ordenados respecto a un indice numérico.
- El indice 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

```
$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
                                                                               2
                                                                                    3
                                                                             25.4
                                                                                        Riviera
                                                           Array 1
                                                                    12
                                                                       verde
                                                                                   vivo
                                                                    12
                                                                             25.4
                                                                                        Riviera
                                                           Array 2
                                                                       verde
                                                                                   vivo
                                                            Array 3
                                                                   12
                                                                       verde
                                                                              25.4
                                                                                   vivo
                                                                                        Riviera
```



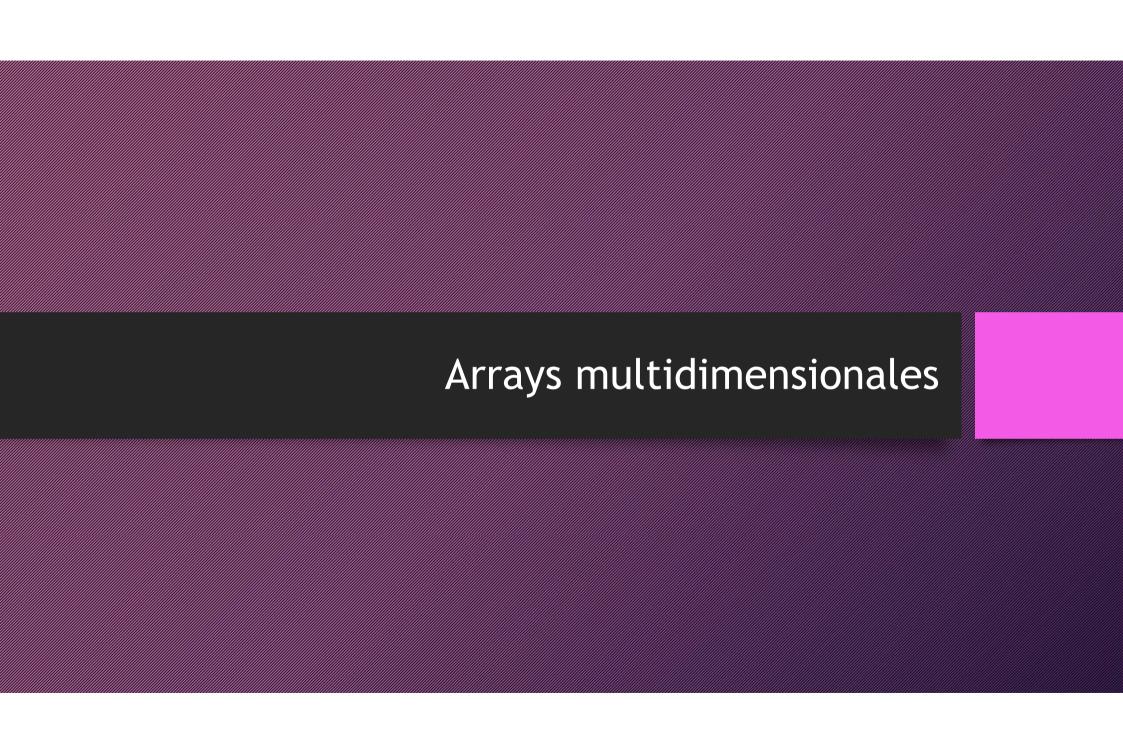
### Arrays asociativos

- 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

```
$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

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

Veamos ejemplos:

## Asignación de arrays multidimensionales

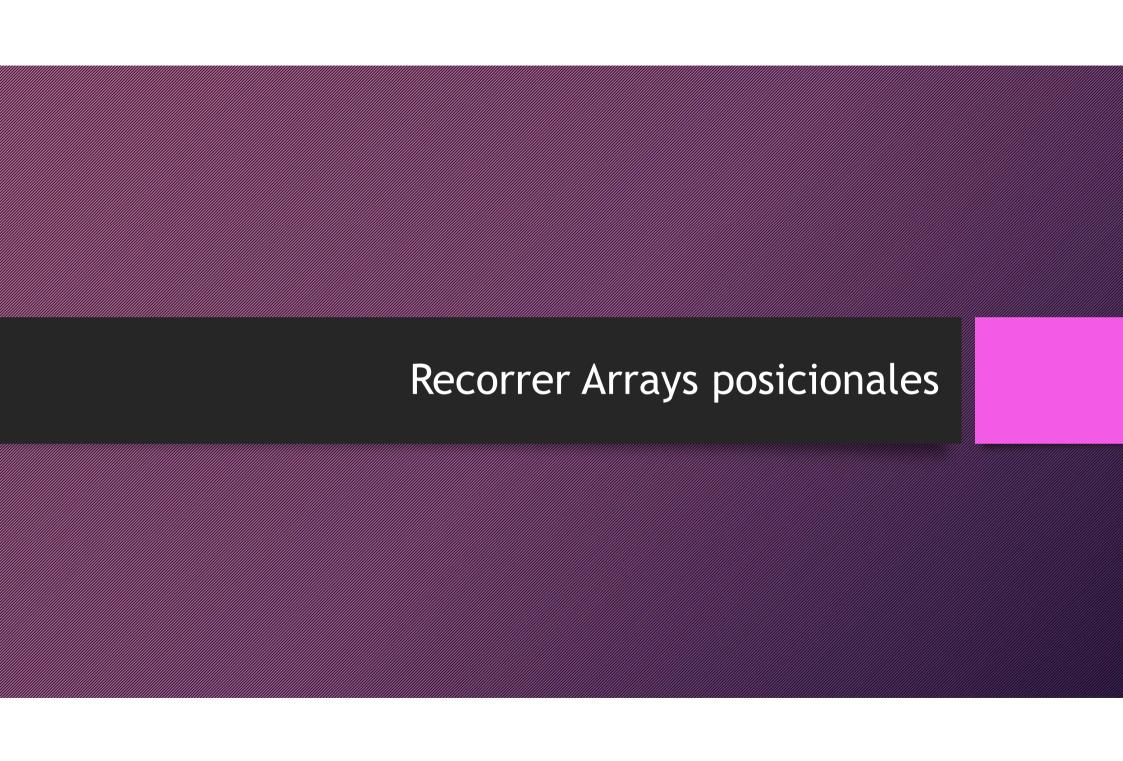
```
$matriz1[0][0]="Peseta";
$matriz1[0][1]=166.386;
$matriz1[1][0]="Dólar";
$matriz1[1][1]=0.96;

$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

- 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.
  - - Devuelve el número de elementos que hay en el array.

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

#### Recorrer Arrays posicionales

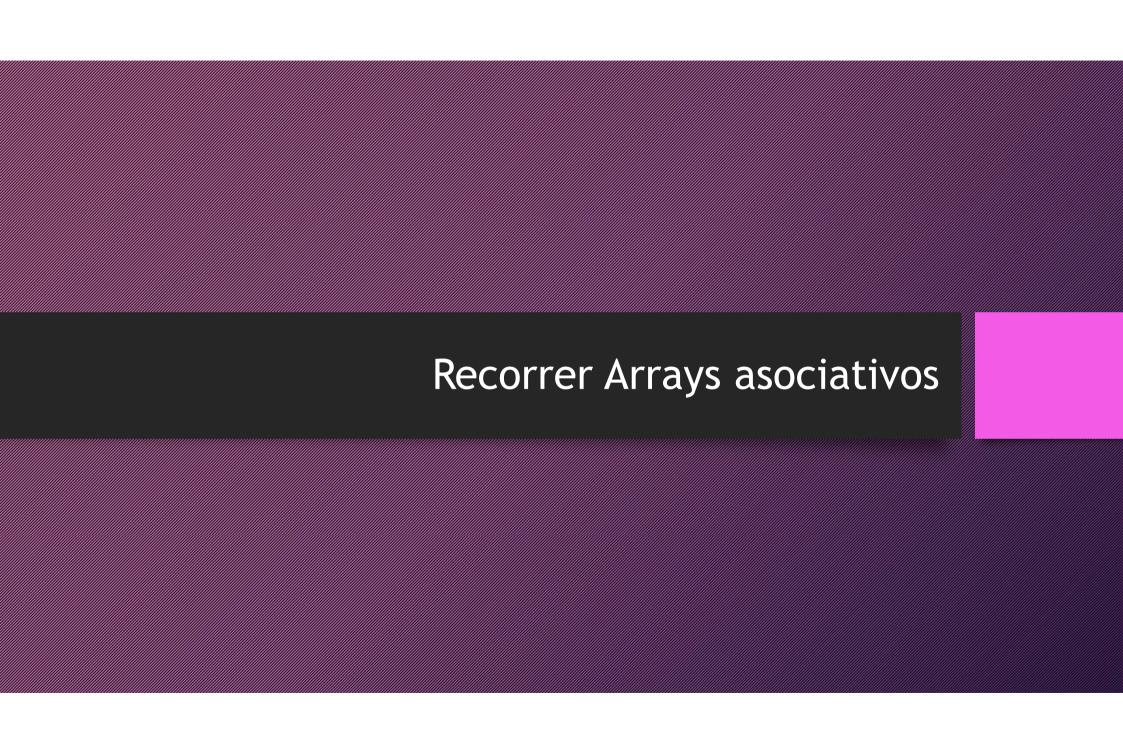
• Pero la función COUNT puede inducir a error muy fácilmente:

```
$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

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

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



- 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:

#### 

- 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.

#### 

- Devuelve el valor del elemento siguiente al actual (si existe) y avanza el puntero.
- Si el elemento actual era el último, devuelve false.

#### 

- 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.

```
    $\text{?php}
    $\text{trans} = \text{array}('\text{pie', 'bici', 'coche', 'avión');}
    $\text{posi} = \text{current}(\text{$trans}); // \text{$posi} = '\text{pie';}
    $\text{$posi} = \text{nans}; // \text{$posi} = '\text{bici';}
    $\text{$posi} = \text{current}(\text{$trans}); // \text{$posi} = '\text{bici';}
    $\text{$posi} = \text{prev}(\text{$trans}); // \text{$posi} = '\text{pie';}
    $\text{$posi} = \text{end}(\text{$trans}); // \text{$posi} = '\text{avión';}
    $\text{$posi} = \text{current}(\text{$trans}); // \text{$posi} = '\text{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

### Recorrer Arrays asociativos: Ejercicio

```
do
{
    echo "".current($posiciones)."";
    echo "".current($valores)."";
    next($valores);
    echo "";
}while (next($posiciones));
```

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

#### 

 Coloca el puntero interno en la última posición de un array.

#### 

- Coloca el puntero interno en la primera posición.
- Devuelve el primer elemento del array

- Patray Xensarray
  - 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.

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

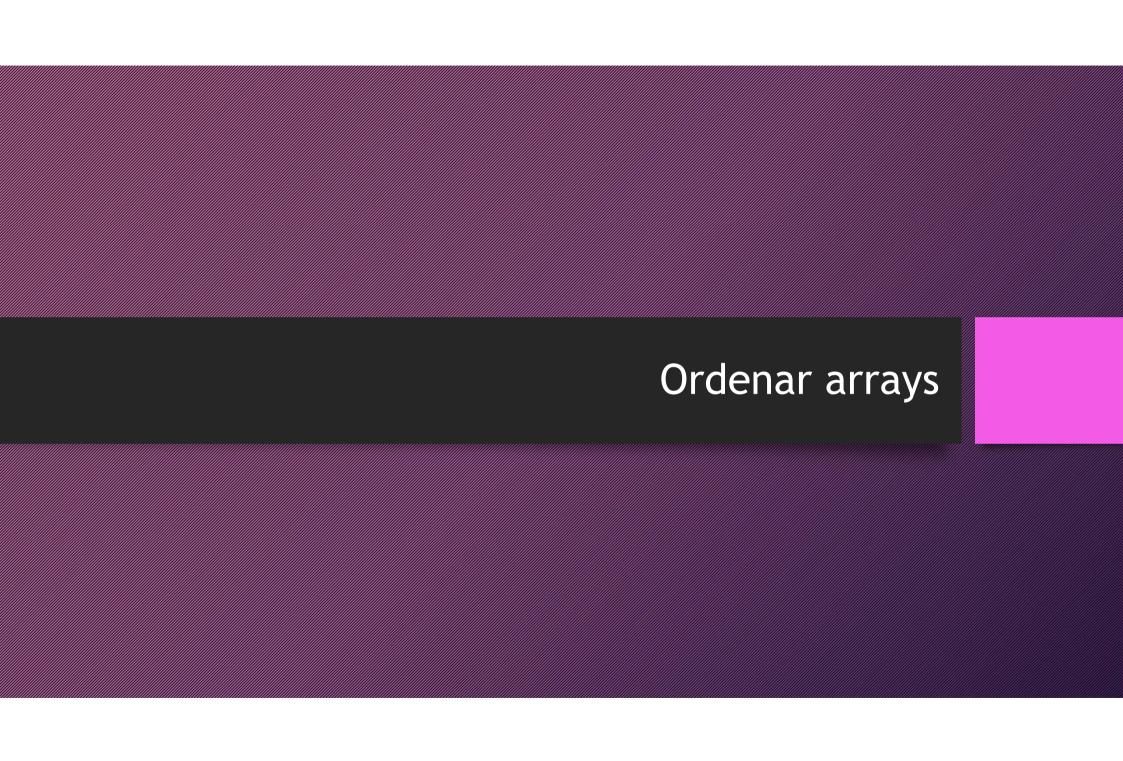
### Recorrer Arrays asociativos: Ejercicio

 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
   $persona = array('Nombre'=>'Juan', 'Altura'=>1.85,
               'Edad'=>25, 'Pelo'=>'Moreno',
               'Ciudad'=>'Granada');
   $posiciones = array keys($persona);
   $valores = array values($persona);
   echo "";
   for ($i=0; $i < count($persona); $i++)</pre>
       echo "$posiciones[$i]";
   echo "";
   for ($i=0; $i < count($persona); $i++)</pre>
       echo "$valores[$i]";
   echo "";
```



### Ordenar Arrays

- - Ordena alfabéticamente los valores.
  - De menor a mayor.
  - Se pierde la relación entre indice y valor

- rsord(arday)
  - Ordena de forma inversa a sort.

```
<?php
    $frutas = array("limón", "naranja",
            "platano", "albaricoque");
    sort($frutas);
    foreach ($frutas as $clave => $valor) {
        echo "frutas[" . $clave . "]";
        echo " $valor ";
        echo "";
                                Posicion | Valor
                                frutas[0]
                                       albaricoque
                                frutas[1]
                                       limón
                                frutas[2]
                                       naranja
                                frutas[3]
                                       platano
```

## Ordenar Arrays

- assert(array)
  - Ordena igual que sort, pero mantiene la relación entre indice y valor.
- - Ordena de forma inversa a asort.

## Diferencias entre sort y asort

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

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

#### Vector sin ordenar Vector ordenado con sort Vector ordenado con asort

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

#### Ordenar Arrays

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

#### 

- Ordena alfanuméricamente las claves de un array de menor a mayor.
- Mantiene las relaciones entre índice y valor.

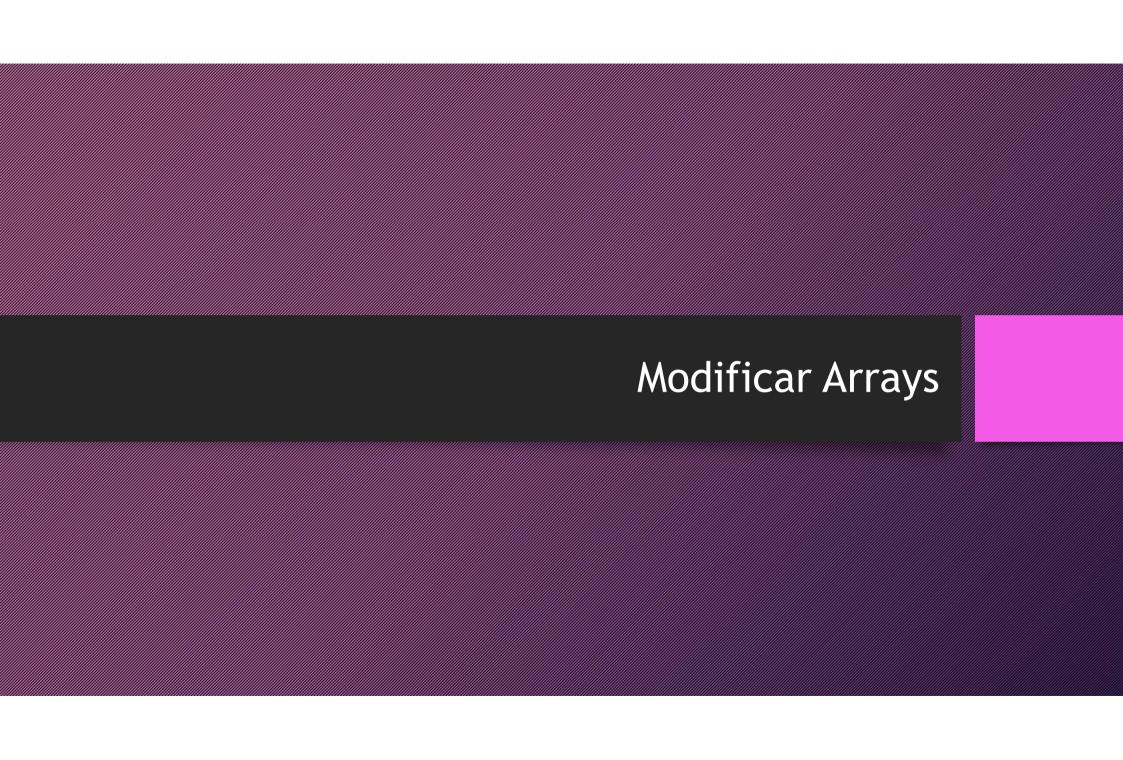
#### 

Ordena de forma inversa a ksort.

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

#### Vector sin ordenar Vector ordenado con ksort

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

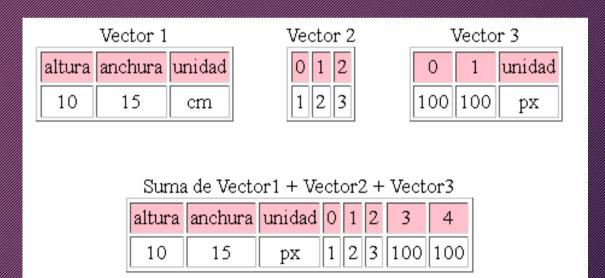


### **Modificar Arrays**

- - 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**

```
$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);
```





#### Otras funciones de Arrays

- array\_reverse (array)
  - Devuelve el array pasado como parámetro, pero con sus componentes en orden inverso.
- range (himite\_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

```
$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

#### 

 Nos dice si un elemento está dentro de un array o no.

#### 

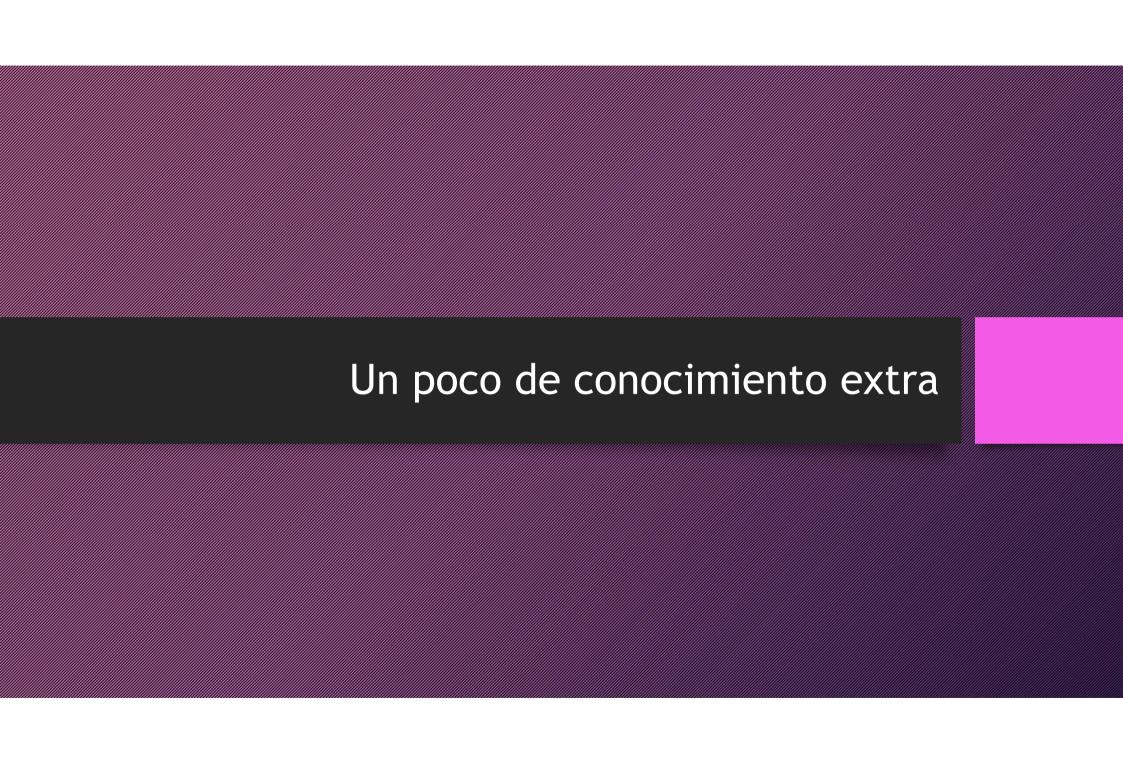
- Recibe como argumento una lista de variables que han sido definidas previamente.
- Devuelve un array en el que los indices son los nombres de las variables y el contenido, sus correspondientes valores.

Hay muchísimas más funciones de Arrays

php.net

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

Posicion	Valor	
n1	3	
n2	9	
n3	11	



#### Números aleatorios

- mt<u>\_</u>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