

## Arrays multidimensionales

PHP permite el uso de arrays con dimensión superior a dos. Para modificar la dimensión del array basta con ir añadiendo nuevos índices.

**\$a[x][y][z]=valor;**

asignaría un valor al elemento de índices **x**, **y** y **z** de un array *tridimensional* y

**\$a[x][y][z][w]=valor;**

haría lo mismo, ahora con un array de dimensión **cuatro**.

Pueden tener cualquier tipo de *índices*: *escalares*, *asociativos* y, también, *mixtos*.

### La función array();

Para asignar valores a una matriz puede usarse la función **array()**, que tiene la siguiente sintaxis:

```
$a= array (  
    índice 0 => valor,  
    ..... ,  
    índice n => valor,  
);
```

Por ejemplo:

```
$z=array (  
    0 => 2,  
    1 => "Pepe",  
    2 => 34.7,  
    3 => "34Ambrosio",  
);
```

producirá igual resultado que:

```
$z[0]=2;  
$z[1]="Pepe";  
$z[2]=34.7;  
$z[3]="34Ambrosio";
```

### Anidando en array();

La función **array()** permite escribir arrays de cualquier dimensión utilizando la técnica de **anidado**.

Si pretendemos escribir los elementos de este array:

```
$z[0][0]=34;  
$z[0][1]=35;  
$z[0][2]=36;  
$z[1][0]=134;  
$z[1][1]=135;  
$z[1][2]=136;
```

podríamos hacerlo así:

## Arrays multidimensionales

Esta es la forma en la que hemos definido el array tridimensional que utilizaremos en el ejemplo.

```
<?  
$b = array(  
    "Juvencia" => array(  
        "Juvencia" => array (  
            "Resultado" => " ",  
            "Amarillas" => " ",  
            "Rojas" => " ",  
            "Penalty" => " "  
        ),  
        "Mosconia" => array (  
            "Resultado" => "3-2",  
            "Amarillas" => "1",  
            "Rojas" => "0",  
            "Penalty" => "1"  
        ),  
        "Canicas" => array (  
            "Resultado" => "5-3",  
            "Amarillas" => "0",  
            "Rojas" => "1",  
            "Penalty" => "2"  
        ),  
        "Condal" => array (  
            "Resultado" => "7-1",  
            "Amarillas" => "5",  
            "Rojas" => "2",  
            "Penalty" => "1"  
        ),  
        "Piloñesa" => array (  
            "Resultado" => "0-2",  
            "Amarillas" => "1",  
            "Rojas" => "0",  
            "Penalty" => "0"  
        ),  
    ),  
    "Mosconia" => array(  
        "Juvencia" => array (  
            "Resultado" => "0-11 ",  
            "Amarillas" => "4",  
            "Rojas" => "2",  
            "Penalty" => "4"  
        ),  
        "Mosconia" => array (  
            "Resultado" => " ",  
            "Amarillas" => " ",  
            "Rojas" => " ",  
            "Penalty" => " "  
        ),  
        "Canicas" => array (  
            "Resultado" => "2-1",  
            "Amarillas" => "0",  
            "Rojas" => "0",  
            "Penalty" => "2"  
        ),  
        "Condal" => array (  
            "Resultado" => "1-0",  
            "Amarillas" => "1",  
            "Rojas" => "0",  
            "Penalty" => "0"  
        ),  
        "Piloñesa" => array (  
            "Resultado" => "1-2",
```

```
$z=array(
  0 => array (
    0 => 34,
    1 => 35,
    2 => 36,
  ),
  1 => array (
    0 => 134,
    1 => 135,
    2 => 136,
  )
);
```

Como puedes observar, se trata de sustituir los *valores* asignados a los elementos de una primera función **array()** por otra nueva función **array** que contiene los segundos índices así como los valores asignados a los mismos.

El anidado sucesivo permitiría generar *arrays* de cualquier dimensión.

Aunque en el ejemplo anterior nos hemos referido a un *array* *escalar*, idéntico procedimiento sería válido para *arrays* *asociativos* con sólo cambiar los números por *cadenas* escritas entre *comillas*.

Este podría ser un ejemplo de *array* *asociativo*:

```
$z["a"]["A"]=34;
$z["a"]["B"]=35;
$z["a"]["C"]=36;
$z["b"]["A"]=134;
$z["b"]["B"]=135;
$z["b"]["C"]=136;
```

que podría definirse también de esta forma:

```
$z=array(
  "a" => array (
    "A" => 34,
    "B" => 35,
    "C" => 36,
  ),
  "b" => array (
    "A" => 134,
    "B" => 135,
    "C" => 136,
  )
);
```

A medida que la dimensión se hace mayor la sintaxis requiere muchísima más atención y los errores son poco menos que inevitables. *Refresquemos* un poco la memoria.

No olvides *los punto y coma* del final de las instrucciones.

Cuidado con las *formas anidadas* y también con los **paréntesis**.

Cierra cada uno de los paréntesis que abras y no olvides que los paréntesis se **anidan**, ya sabes... el primero que se abre siempre con el último que se cierra, el segundo con el penúltimo, etcétera.

No dejes de prestar atención a las *comillas*. Recuerda que hay que cerrarlas siempre y que hay que diferenciarlas en los casos en que van *comillas dentro de otras*

```

    "Amarillas" => "1",
    "Rojas"      => "0",
    "Penalty"   => "0"
  ),
  "Canicas" => array(
    "Juvencia" => array (
      "Resultado" => "0-0",
      "Amarillas" => "0",
      "Rojas"      => "1",
      "Penalty"    => "1"
    ),
    "Mosconia" => array (
      "Resultado" => "1-3",
      "Amarillas" => "2",
      "Rojas"      => "0",
      "Penalty"    => "1"
    ),
    "Canicas" => array (
      "Resultado" => " ",
      "Amarillas" => " ",
      "Rojas"      => " ",
      "Penalty"    => " "
    ),
    "Condal" => array (
      "Resultado" => "1-4",
      "Amarillas" => "2",
      "Rojas"      => "1",
      "Penalty"    => "1"
    ),
    "Piloñesa" => array (
      "Resultado" => "2-0",
      "Amarillas" => "1",
      "Rojas"      => "0",
      "Penalty"    => "0"
    ),
  ),
  "Condal" => array(
    "Juvencia" => array (
      "Resultado" => "1-0 ",
      "Amarillas" => "4",
      "Rojas"      => "1",
      "Penalty"    => "2"
    ),
    "Mosconia" => array (
      "Resultado" => "6-3",
      "Amarillas" => "1",
      "Rojas"      => "2",
      "Penalty"    => "3"
    ),
    "Canicas" => array (
      "Resultado" => "14-3",
      "Amarillas" => "1",
      "Rojas"      => "0",
      "Penalty"    => "0"
    ),
    "Condal" => array (
      "Resultado" => " ",
      "Amarillas" => " ",
      "Rojas"      => " ",
      "Penalty"    => " "
    ),
    "Piloñesa" => array (
      "Resultado" => "1-0",
      "Amarillas" => "3",
      "Rojas"      => "1",
      "Penalty"    => "0"
    ),
  ),
  "Piloñesa" => array(
    "Juvencia" => array (
      "Resultado" => "1-1",
      "Amarillas" => "0",
      "Rojas"      => "0",
      "Penalty"    => "1"
    ),
    "Mosconia" => array (
      "Resultado" => "2-3",
      "Amarillas" => "1",

```

comillas.  
Una última advertencia. ¡No te desespere con los errores de sintaxis! Son inevitables.

Enmendando un olvido

Cuando hemos hablado de las funciones matemáticas hemos olvidado mencionar una de ellas.

Se trata de la función valor absoluto.

La sintaxis es la siguiente:

Abs(\$a);

Un ejemplo:

```
<?
$MiSaldo="-347.513 € ";
$MisDeseos=Abs($MiSaldo);
$MisDeseos .= " €";
echo $MisDeseos;
?>
```

resultaría :

347.513 €

```
"Rojas"      => "0",
"Penalty"    => "0"
),
"Canicas"    => array (
"Resultado"  => "0-1",
"Amarillas"  => "0",
"Rojas"      => "0",
"Penalty"    => "0"
),
"Condal"     => array (
"Resultado"  => "1-1",
"Amarillas"  => "1",
"Rojas"      => "2",
"Penalty"    => "0"
),
"Piloñesa"   => array (
"Resultado"  => " ",
"Amarillas"  => " ",
"Rojas"      => " ",
"Penalty"    => " "
),
),
);
?>
```

Utilizando este array hemos construido la tabla que hemos puesto como ejemplo.

ejemplo21.php

Anterior

Índice

Siguiente