



### TEMAS A DESARROLLAR

Arreglos en PHP.

Mostrar el contenido de un arreglo.

Arreglos multidimensionales.

Recorrer Arreglos.

String



## Arreglos en PHP

### Arreglos en PHP

Un arreglo es una colección ordenada de elementos compuestos por un valor y referenciado por una clave única que lo identifica dentro del arreglo.

```
$arreglo = array();
```

Para definir un arreglo en php utilizamos `array()` lo cual establece la variable para almacenar datos compuestos. También podemos definir un arreglo haciendo uso de los corchetes `[]`:

```
$arreglo = ['rojo','azul'];
```

### Arreglos en PHP

En PHP **no es necesario** definir el tipo de dato que contendrá el arreglo ni su tamaño.

Un arreglo en PHP **puede almacenar** datos de distinto tipo.

PHP utiliza **números para indexar** un arreglo de manera predeterminada, **pero también se pueden utilizar letras o palabras como índices**.

Un arreglo **puede contener** otros arreglos, objetos, números, o una combinación de todos los anteriores.

Podemos definir arreglos unidimensionales(vectores), bidimensionales(matrices) o con las dimensiones que necesitemos.

### Arreglos en PHP

Veamos algunos ejemplos:

```
$colores = array( 'rojo', 'azul', 'amarillo' );
```

```
$persona = array(  
    'documento'=>22444555,  
    'apellido'=>'Gomez',  
    'nombre'=>'Martín'  
);
```

Veamos algunos ejemplos:

```
$cosas= array(  
    0=>22444555,  
    'apellido'=>'Gomez',  
    1=>'Martín'  
);
```

### Arreglos en PHP

```
$persona1 = array(  
    'documento'=>22444555,  
    'apellido'=>'Gomez',  
    'nombre'=>'Martín'  
);  
  
$persona2 = array(  
    'documento'=>55222333,  
    'apellido'=>'Lopez',  
    'nombre'=>'Analía'  
);
```

Podemos almacenar arreglos en otros arreglos:

```
$personas = array();  
$personas[]=$persona1;  
$personas[]=$persona2;
```



**Mostrar el contenido de los  
arreglos**

### Mostrar el contenido de los arreglos

Para mostrar el contenido de un arreglo podemos utilizar el constructor **echo** indicando el nombre del arreglo y su índice.

```
<?php
```

```
$colores= array('rojo','azul','verde');
```

```
echo $color[2];
```

```
?>
```

La salida será: **verde**



### Mostrar el contenido de los arreglos

Existen dos funciones más que nos permiten mostrar no solo el contenido de los arreglos sino también los tipos de datos que tienen almacenados. Estas funciones son `print_r()` y `var_dump()`.

Ejemplo de `print_r()`:

```
<?php  
  
$colores= array('rojo','azul','verde');  
  
print_r ( $color );  
  
?>
```

La salida será:

```
Array (  
    [0] => rojo  
    [1] => azul  
    [2] => verde  
)
```

### Mostrar el contenido de los arreglos

A diferencia de `print_r()`, `var_dump()` muestra la estructura del arreglo junto con el tipo y el tamaño del dato almacenado en cada elemento del mismo.

El mismo ejemplo con `var_dump()`;

```
<?php
```

```
$colores= array('rojo','azul','verde');
```

```
var_dump( $color );
```

```
?>
```

La salida será:

```
array(3) {  
    [0]=> string(4) "rojo"  
    [1]=> string(4) "azul"  
    [2]=> string(5) "verde"  
}
```



## Arreglos multidimensionales

### Arreglos multidimensionales

En PHP un arreglo puede contener todo tipos de datos, inclusive otros arreglos por lo que la creación de arreglos multidimensionales es muy sencilla. Simplemente bastará con asignar un arreglo como el valor del elemento de otro arreglo.

Ejemplo:

```
<?php
    $personas= array();
        $persona1= array( 'documento'=>22333444, 'apellido'=>'Bonnet',' nombres'=>'Federico');
        $persona2= array( 'documento'=>11222333, 'apellido'=>'Aramburu',' nombres'=>'Exequiel');
    $personas[] = $persona1;
    $personas[] = $persona2;
?>
```

### Arreglos multidimensionales

Ejemplo:

```
<?php
    $personas= array();
        $persona1= array( 'documento'=>22333444, 'apellido'=>'Bonnet',' nombres'=>'Federico');
        $persona2= array( 'documento'=>11222333, 'apellido'=>'Aramburu',' nombres'=>'Exequiel');
    $personas[] = $persona1;
    $personas[] = $persona2;

?>
```

Mostrar los elementos:

```
<?php
    print_r( $personas [1]) ;

?>
```

La salida será:

```
Array(
    'documento'=>11222333,
    'apellido'=>'Aramburu',
    'nombres'=>'Exequiel'
)
```



## Recorrer Arreglos

### Recorrer Arreglos

Mostrar los elementos:

Si el arreglo tiene la misma cantidad de filas que de columnas y si sus **índices son todos numéricos**, podemos utilizar la función **for()** para mostrar sus elementos:

```
<?php
    $colores = array ( 'rojo', 'azul', 'verde');
    for ($i=0; $i++; $i< count($colores) ){
        echo $colores[i];
    }
```

### Recorrer Arreglos

Si el arreglo tiene sus **índices compuestos por nombres o letras**, podemos utilizar la función **foreach()** para mostrar sus elementos:

```
<?php  
  
$auto = array ( 'color'=>'rojo', 'marcar'=>'Chevrolet', 'año'=>'1969' );  
foreach ( $auto as $clave=>$valor ){  
    echo $clave.'=>';  
    echo $valor;  
}  
?>
```

La salida será:

Color=>rojo  
Marca => Chevrolet  
Año => 1969



## Recorrer Arreglos

Si el arreglo es **multidimensional**:

```
<?php
    $autos[ ] = array ( 'color'=>'negro', 'marcar'=>'Chevrolet', 'Modelo'=>'Camaro', 'Año'=>'1969' );
    $autos[ ] = array ( 'color'=>'rojo', 'marcar'=>'Ford', 'año'=>'1969' );
    foreach ( $autos as $unAuto){
        foreach ( $unAuto as $atributos ){
            echo $atributos;
        }
    }
?>
```



**String**

### String

En PHP un **string** es cualquier conjunto de caracteres entrecomillados. PHP considera como cadena todo lo que encuentre entre un par de comillas, por eso todas las cadenas deben comenzar y terminar con el mismo tipo de comillas, simples o dobles.

```
<?php
```

```
$nombre = " Martín " ;
```

```
echo " Saludos, $nombre" // Muestra: "Saludos, Martín"
```

```
echo ' Saludos, $nombre ' // Muestra: "Saludos, $nombre\n"
```

```
?>
```

### String

Las comillas simples representan "cadenas simples," donde todos los caracteres son utilizados literalmente.

```
echo ' Saludos, $nombre ' // Muestra: "Saludos, $nombre\n"
```

Las comillas dobles encapsulan "cadenas complejas" que permiten las secuencias de escape (para insertar caracteres especiales) y para la sustitución de variables (integran el valor de una variable directamente en una cadena).

```
$nombre = " Martín " ;
```

```
echo " Saludos, $nombre" // Muestra: "Saludos, Martín"
```

### String

Una particularidad de los **string** en PHP es que podemos obtener caracteres particulares tratando los índices como si fueran **array()**:

```
<?php
    $texto = "Esta es la cadena de texto que utilizaremos en el ejemplo";
    echo $texto [2];
}
?>
```