Manual array PHP
Diego Casero Martin

ARRAY

Los arrays en php se definen por ser muy dinámicos, es decir no hace falta que estén inicializados para poder usar estos, es decir podemos crear nuestro propio array sin definir el tamaño:

```php
$a = array();
```

Además de que pueden almacenar tipos distintos de valores:

```php
$a = array('1.10', 12.4, 1.13);
```

Los array podemos dividirlos en dos tipos, arrays simples o asociativos, simples serían los que vemos hasta ahora, los cuales tienen una clave numérica autoincrementable y los asociativos son aquellos a los cuales les definimos una clave:

```php
$semana = array("Lunes " , "Martes " , "Miercoles " , "Jueves " , "Viernes " , "Sabado " , "Domingo ");
```

Array simple

```php
'España' => array(
    'Toledo' => 300,
    'Madrid' => 700,
    'Barcelona' => 800,
    'Cantabria' => 400,
    'Valencia' => 500,
    'Albacete' =>200,
    'Canarias' =>  100
),
```

Array asociativo
CLAVE/VALOR

A la hora de mostrar estos podemos recorrerlos con un bucle foreach , lo cual nos facilitara mucho la vida

```php
foreach ($array as $key => $value) {
    echo $key . '=' . $value . '</br>';
}
```

los arrays bidimensionales se declaran como un array asociativo que crea otro array, y estos pueden tener diferente longitud, para recorrerlos necesitamos bucles foreach anidados:

```php
$paises = array(
    'España' => array(
        'Toledo' => 300,
        'Madrid' => 700,
        'Barcelona' => 800,
        'Cantabria' => 400,
        'Valencia' => 500,
        'Albacete' =>200,
        'Canarias' =>  100
    ),
    'Alemania' => array(
        'Munich' => 500,
        'Berlin' => 400,
        'Sajonia' => 100,
        'Hesse' => 200,
        'Dormunt' => 300,
    )
);
```

Array bidimensional

```php
$paises2 = array();
$habitanespais=0;
foreach ($paises as $pais => $arrayprov) {
    foreach($arrayprov as $prov => $numhabitantes){
        $habitanespais += $numhabitantes;
    };
    array_push($paises2 , array($pais => $habitanespais));
    $habitanespais = 0;
}
```

foreach de array bidimensional

como podemos ver el primer foreach tiene una clave y como valor otro array, el cual recorreremos con otro foreach

Funciones para recorrer un array :

- **current()** - devuelve el valor del elemento que indica el puntero
- **pos()** - realiza la misma función que current
- **reset()** - mueve el puntero al primer elemento del array
- **end()** - mueve el puntero al último elemento del array
- **next()**- mueve el puntero al elemento siguiente
- **prev()** - mueve el puntero al elemento anterior
- **count()** - devuelve el número de elementos de un array
- **key()** – devuelve el índice de la posición actual

Manual array PHP
Diego Casero Martin

FUNCIONES ARRAY

búsqueda:

- **array preg_grep**

Esta función requiere un ($string , $array);
Devuelve un array con los elementos que coincidan con el string
Se usa con expresiones regulares

- **array_search**

Esta función requiere un (valor , $array);
Busca un valor dentro del array, si lo encuentra devuelve su clave, si no devuelve null

- **in_array**

Esta función requiere un (valor , $array, $stric?);
Devuelve true o false en función de la existencia o no del valor, si stric esa a true tendrá en cuenta el tipo de valor, es case-sensitive.

- **array_count_values**

Esta función requiere un valor($array);
Devuelve cuantas veces aparece cada elemento de un array en ese array

modificar:

- **array_pop**

Esta función requiere ($array);
Devuelve el último valor del array

- **array_push**

Esta función requiere ($array , dato);
Insertar elementos al final del array y devuelve en número de elementos que contiene el array aumentado

- **array_shift**

Esta función requiere ($array);
Saca el primer elemento y recoloca el resto, devuelve este primer elemento

- **array_unshift**

Esta función requiere ($array , dato);
Añade un elemento al principio del array, devuelve el número de elementos del array

- **array_replace**

Esta función requiere ($array_destino , $array_origen);
Devuelve un array que es el resultado de sobrescribir/añadir sobre
matriz destino los elementos de matriz origen (los que coinciden en
índice se sobrescriben, y los que no se añaden). No afecta a las
matrices que recibe como argumento

- **array_merge**

Esta función requiere ($array1,$array2);
Une matrices indicadas, elimina los elementos claves duplicados en array asociativo(mantiene la última leída) , en arrays numéricos se generan nuevas claves. devuelve un array

- **array_merge_recursive**

Esta función requiere ($array1,$array2);
Permite combinar las matrices sin perder elementos, devuelve la matriz resultado de la suma con las claves duplicadas genera una nueva matriz

- **array_pad**

Esta función requiere ($array , $cant , $relleno);
Permite añadir elementos de relleno en el inicio (negativo) y fin del array(positivo), devuelve una matriz

- **array_slice**

Esta función requiere ($array , $inicio , $cant);
Devuelve un subarray del array a partir del inicio indicado con la cantidad de elementos indicada, si la cantidad no se especifica devuelve desde inicio hasta el final

- **array_splice**

Esta función requiere ($array , $inicio , $cant , $reemplazo);
Elimina del array la cantidad de elementos contador a partir del elemento inició, los sustituye por los elementos del array reemplazo y los devuelve en un array, si los índices son numéricos los reajusta

- **implode**

Esta función requiere ($string , $array);
Convierte un array en una cadena de caracteres separando sus elementos con la cadena indicada en el delimitador.

Inserción

- **array_intersect**

Esta función requiere ($array * n);
devuelve una matriz con los elementos comunes, la comparación se realiza con el operador ===

- **array_intersect_assoc**

Igual que la anterior pero tiene en cuenta las claves

- **array_unique**

Esta función requiere ($array);
Crea una nueva matriz a partir de la original, tomando solo elementos no duplicados

- **array_combine**

Esta función requiere ($array_clave , $array_valor);
Fusiona los dos arrays, tomando de uno las claves y de otros los valores

- **array_reverse**

Esta función requiere ($array , true/false)
Devuelve el array invertido, si el segundo valor es true conserva las claves

- **range**

Esta función requiere ($low , $high , $salto);
Crea una matriz que contiene rangos

- **compact**

Esta función requiere ($var1 , $var2 , ...);
Crea un array asociativo con los nombres de las variables y su valor

- **shuffle**

Esta función requiere ($array)
Desordena de forma aleatoria un array

ordenación

- **sort**

Ordena un array de menor a mayor

- **rsort**

Ordena un array en orden inverso (de mayor a menor)

- **asort**

Ordena un array manteniendo la correlación de los índices con los elementos asociados.

- **arsort**

Ordena un array en orden inverso, manteniendo la correlación de los índices con los elementos asociados.

- **ksort**

Ordena un array por clave, manteniendo la correlación entre la clave y los datos.

- **krsort**

Ordena un array por clave en orden inverso, manteniendo la correlación entre la clave y los datos.

- **usort**

Ordena un array usando una función de comparación definida por el usuario. Se asignan nuevas claves a los elementos ordenados

- **uksort**

Ordena las claves de un array usando una función de comparación proporcionada por el usuario

- **uasort**

Ordena un array de manera que los índices mantienen sus correlaciones con los elementos del array asociados, usando una función de comparación definida por el usuario.

- **array_multisort**

Ordenar varios arrays al mismo tiempo, o un array multidimensional por una o más dimensiones. Las claves asociativas (string) se mantendrán, aunque las claves numéricas son re-indexadas