ARRAYS.

Referencias:

http://www.php.net
http://www.w3schools.com

Un array es una estructura de datos que contendrá un conjunto de variables con un nombre común y un índice para referenciarlas individualmente. El índice acompaña al nombre del array entre corchetes, "[" "]".

Para estudiarlos vamos a clasificarlos en:

- Arrays indexados: Arrays con índice numérico.
- Arrays asociativos: Arrays que utilizan como índice una cadena de caracteres.
- Arrays multidimensionales: Arrays cuyo contenido es uno o más arrays y por tanto necesitan varios índices para el acceso a los valores.

Arrays indexados

Son arrays que utilizan un índice numérico. El primer índice puede ser cualquier número, pero se recomienda empezar en cero.

Hay dos formas de crear un array indexado:

Utilizando la función "arrray()".

```
<?php
   $cars = array("Volvo", "BMW", "Toyota");
?>
```

Asignando directamente el contenido.

```
<?php
   $cars[0] = "Volvo";
   $cars[1] = "BMW";
   $cars[2] = "Toyota";
<?
```

Es posible crear nuevos elementos con asignación automática de índices utilizando los corchetes sin índice. Cuando se asigna un valor a una variable array usando corchetes vacíos, el valor se añadirá al final del array.

```
<?php
   $cars[]='Renault';
   $cars[]='Citrpoen';
?>
```

Arrays asociativos.

Son arrays que utilizan como índices cadenas de caracteres.

Hay dos maneras de crear un arrayz asociativo:

Utilizando la función "array()"

```
<?php
   $age = array(
                  "Peter"=>"35",
                  "Ben"=>"37",
                  "Joe"=>"43"
            );
?>
```

Asignando directamente el contenido:

```
<?php
   $age['Peter'] = "35";
   $age['Ben'] = "37";
   $age['Joe'] = "43";
?>
```

```
<?php
   $age = array(
                  "Peter"=>"35",
                  "Ben"=>"37",
                  "Joe"=>"43"
            );
   echo "Peter is " . $age['Peter'] . " years old.";
?>
```

Trabajando con arrays

Longitud de un array. Función count ()

```
<?php
   $cars = array("Volvo", “BMW", "Toyota");
   echo count($cars);
?>
```

Recorrer un array indexado

Para recorrer e imprimir todos los valores de un array es necesario utilizar un bucle.

```
<?php
   $cars = array("Volvo", "BMW", "Toyota");
   $arrlength = count($cars);

   for($x = 0; $x < $arrlength; $x++) {
      echo $cars[$x];
      echo "<br>";
   }
?>
```

Recorrer una matriz asociativa

Para recorrer e imprimir todos los valores de una matriz se recomienda utilizar el bucle ***foreach***

foreach (expArray as $valor)
bloque

El bucle recorre el array dado por expArray. En cada iteración, el valor del elemento actual se asigna a $valor y el puntero interno del array avanza una posición. $valor es tratado en el bloque de instrucciones del bucle.

foreach (expArray as $clave => $valor)
bloque

El bucle recorre el array dado por expArray. En cada iteración, el valor del índice se asigna a $clave y el valor del elemento actual se asigna a $valor.. El puntero interno del array avanza una posición. $clave y $valor son procesados en el bloque de instrucciones del bucle.

Ejemplos

```
<?php
   $array = array(1, 2, 3, 4);
   foreach ($array as $valor) {
      echo $valor;
   }
```

```
<?php
   $age = array("Peter"=>"35","Ben"=>"37","Joe"=>"43");

   foreach($age as $x=>$x_value) {
      echo "Key=" . $x . ", Value=" . $x_value;
      echo "<br>";
   }
?>
```

Funciones de interés.

Se propone investigar el funcionamiento de cada una de las funciones presentadas, e incluir otras que puedan ser de interés.

Declaración: array(), list(). En realidad son constructores del lenguaje y no funciones.

Ordenación: asort(), arsort(), ksort(), rsort(), sort(), uasort(), usort(), y uksort().

Recorrido: each(), next() y prev().

Arrays multidimensionales

Cada variable de un array puede ser a su vez un array, de esta manera podemos crear arrays de dos, tres o más dimensiones. Para manejar el array necesitaremos un índice por cada una de las dimensiones del mismo.

La función array() se puede anidar para crear arrays multidimensionales:

Ejemplos

```
<?php
   $cars = array (
                array("Volvo", 100, 96),
                array("BMW", 60, 59),
                array("Toyota", 110, 100)
              );
?>
```

```php
<?php
   $families = array (
                    "Griffin"=>array ("Peter", "Lois", "Megan"),
                    "Quagmire"=>array("Glenn"),
                    "Brown"=>array("Cleveland", "Loretta", "Junior")
                  );
?>
```

La representación del array anterior sería:

```
Array  (
   [Griffin] => Array (
                        [0] => Peter
                        [1] => Lois
                        [2] => Megan
                )
   [Quagmire] => Array (
                        [0] => Glenn
                )
   [Brown] => Array (
                        [0] => Cleveland
                        [1] => Loretta
                        [2] => Junior                    )
)
```

La siguiente sentencia mostraría un único valor del array anterior:

```
<?php
   echo "Is " . $families['Griffin'][2] ." a part of the Griffin family?";
?>
```

La salida sería:

Is Megan a part of the Griffin family?

```
<?php
   $unaFruta["color"] = "rojo";
   $unaFruta["sabor"] = "dulce";
   $unaFruta["forma"] = "redondeada";
   $unaFruta["nombre"] = "manzana";
<?
```

```
<?php
   $unaFruta = array(
                        "color" => "rojo",
                        "sabor" => "dulce",
                        "forma" => "redondeada",
                        "nombre" => "manzana"
                    );
<?
```

```php
<?php
$frutas = array(
                "manzana" => array(
                                        "color" => "rojo",
                                        "sabor" => "dulce",
                                        "forma" => "redondeada"
                                    ),
                 "naranja" => array(
                                        "color" => "naranja",
                                        "sabor" => "ácido",
                                        "forma" => "redondeada"
                                    ),
                 "plátano" => array(
                                        "color" => "amarillo",
                                        "sabor" => "paste-y",
                                        "forma" => "aplatanada"
                                    )
            );

echo $a["manzana"]["sabor"]; # devolverá "dulce"
?>
```

Arrays. Estandar de codificación.

- No se permiten los números negativos como índices.
- Los índices en array indexados pueden iniciarse con cualquier valor no negativo, pero se recomienda iniciarlos a cero.
- Para mejorar la legibilidad, se debe agregar un espacio después de cada coma de separación de los índices en la declaración del array:
- Se recomienda hacer la declaración de los arrays asociativos divididos en diferentes líneas.
- Se recomienda dejar espacio antes y después de los operadores de asignación “=>” y de forma que queden alineados:

```php
<php
$varArray = array(
                    'primerIndice' => 'primerValor',
                    'segundoIndice' => 'segundoValor',
                );
?>
```

- Se pueden hacer declaraciones de los arrays en varias líneas, siempre y cuando los inicios estén alineados (identados) igual:

```php
<?php
   $meses = array( 1, 2, 3, 4,
                   5, 6, 7, 8,
                   9, 10, 11, 12,
                );
?>
```

Trabajo de Clase

1. Define un array que permita almacenar y mostrar la siguiente información.
    1. Meses del año.
    2. Tablero para jugar al juego de los barcos.
    3. Nota de los alumnos de 2º DAW para el módulo DWES.
    4. Verbos irregulares en inglés.
    5. Información sobre continentes, países, capitales y banderas.

2. Un restaurante dispone de una carta de 3 primeros, 5 segundos y 3 postres. Almacenar información incluyendo foto y mostrar los menús disponibles. Mostrar el precio del menú suponiendo que éste se calcula sumando el precio de cada uno de los platos incluidos y con un descuento del 20 %.