UD-02d: Introducción a PHP

Desarrollo Web en Entorno Servidor

Curso 2020/2021

Características básicas del lenguaje

Tipos de datos compuestos

- Son los que permiten almacenar más de un valor.
- En PHP hay dos tipos: el **array** y el **objeto**.

Características básicas del lenguaje

Arrays

Un array es un tipo de dato que nos permite almacenar varios valores.

Cada miembro del array se almacena en una posición que se referencia con una clave.

Definición de un array indexado

```
$semana = array('Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes', 'Sábado', 'Domingo');
```

Otra forma de escribir arrays → `$semana = [‘Lunes’,’Martes’,… ,’Domingo’];`

¿Como se muestra un array?

```
echo $semana[1];      // mostraría Martes, ya que los arrays comienzan por 0.
```

¿Como se modifica un array?

```
$semana[7]='Fin de Semana';
```

También podemos colocar valores no consecutivos.

```
$semana[17]='Isabel';
```

Características básicas del lenguaje

Array numérico:

```
$modulos1 = array(0 => "Programación", 1 => "Bases de datos",
                  9 => "Desarrollo web en entorno servidor");
```

Array asociativo:

```
$modulos2 = array("PR" => "Programación", "BD" => "Bases de datos",
                  "DWES" => "Desarrollo web en entorno servidor");
```

Para hacer referencia a elementos almacenados en un array con claves asociativas :

```
$modulos["BD"]
```

En PHP existe la función print_r, que nos muestra todo el contenido del array.
Es muy útil para tareas de depuración.

```
print_r($modulos2);
```

Características básicas del lenguaje

- Los arrays anteriores son vectores (arrays unidimensionales).
- En PHP puedes crear también **arrays de varias dimensiones** almacenando otro array en cada uno de los elementos de un array.

```
<?php
    // array bidimensional
    $amigos = array( array('Alejandro',20), array('Cesar',21), array (Pepe',19));
    echo $amigos[0][0];
    echo "<br>";
    echo $amigos[0][1];
    echo "<br>";
    echo $amigos[1][0];
?>
```

Salida:

Alejandro
20
Cesar

Características básicas del lenguaje

Array bidimensional asociativo:

```
$ciclos = array(
   "DAW" => array ("PR" => "Programación", "BD" => "Bases de datos", "DWES" =>
      "Desarrollo web en entorno servidor"),
   "DAM" => array ("PR" => "Programación", "BD" => "Bases de datos", "PMDM" =>
      "Programación multimedia y de dispositivos móviles")
);
```

Para hacer referencia a los elementos almacenados en un array multidimensional, debes indicar las claves para cada una de las dimensiones:

```
$ciclos ["DAW"] ["DWES"]
```

Salida: Desarrollo web en entorno servidor

Características básicas del lenguaje

- En PHP no es necesario indicar el tamaño del array antes de crearlo.
- Ni tampoco indicar que una variable concreta es de tipo array.
- Simplemente, puedes comenzar a asignarle valores:

```
// array numérico
$modulos1 [0] = "Programación";
$modulos1 [1] = "Bases de datos";
...
$modulos1 [9] = "Desarrollo web en entorno servidor";

// array asociativo
$modulos2 ["PR"] = "Programación";
$modulos2 ["BD"] = "Bases de datos";
...
$modulos2 ["DWES"] = "Desarrollo web en entorno servidor";
```

Características básicas del lenguaje

Ni siquiera es necesario que especifiques el valor de la clave.
Si la omites, el array se irá llenando a partir de la última clave numérica existente, o de la posición 0 si no existe ninguna:

```
<?php
    $modulos [] = "Programación";
    $modulos [] = "Bases de datos";
    $modulos [] = "Desarrollo web en entorno servidor";

    echo $modulos[0]."<br/>".$modulos[1]."<br/>".$modulos[2];
?>
```

Salida:

Programación
Bases de datos
Desarrollo web en entorno servidor

Características básicas del lenguaje

Strings como arrays

Las cadenas de texto o strings se pueden tratar como arrays en los que se almacena una letra en cada posición, siendo 0 el índice correspondiente a la primera letra, 1 el de la segunda, etc.

```
// cadena de texto
$modulo = "Desarrollo web en entorno servidor";
// $modulo[3] == "a";
```

Características básicas del lenguaje

Recorrer arrays

Para recorrer los elementos de un array, en PHP puedes usar un bucle foreach. Usa una variable temporal para asignarle en cada iteración el valor de cada uno de los elementos del array.

Se puede utilizar de 2 formas:

- Recorriendo solo los elementos
- Recorriendo elementos y valores clave

Características básicas del lenguaje

Recorriendo sólo los elementos:

```
<?php
    $modulos = array("PR" => "Programación", "BD" => "Bases de datos", "DWES" =>
        "Desarrollo web en entorno servidor");

    foreach ($modulos as $modul)
    {
        echo $modul. "<br />";
    }
?>
```

Salida:

Programación
Bases de datos
Desarrollo web en entorno servidor

Características básicas del lenguaje

Recorriendo elementos y valores clave a la vez:

```
<?php
    $modulos = array("PR" => "Programación", "BD" => "Bases de datos", "DWES" =>
        "Desarrollo web en entorno servidor");

    foreach ($modulos as $codigo => $modulo)
    {
        echo $codigo." - ".$modulo."<br />";
    }
?>
```

Salida:
PR - Programación
BD - Bases de datos
DWES - Desarrollo web en entorno servidor

Características básicas del lenguaje

Ejemplo de mostrar los elementos de un array con foreach en html.

```
<!DOCTYPE HTML>
<html>
<head>
	<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
	<title>Documento sin título</title>
</head>
<body>
	<h1>Los días de la Semana son;</h1>
	<ul>
	<?php
		$semana = array('Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes', 'Sábado', 'Domingo');
		foreach ($semana as $dia) {
			echo '<li>' . $dia . '</li>';
		}
	?>
	</ul>
</body>
</html>
```

Características básicas del lenguaje

En PHP hay **otra forma de recorrer los valores de un array**. Cada array mantiene un puntero interno, que se puede utilizar con este fin. Utilizando funciones específicas, podemos avanzar, retroceder o inicializar el puntero, así como recuperar los valores del elemento (o de la pareja clave / elemento) al que apunta el puntero en cada momento.

Función	Resultado
reset	Sitúa el puntero interno al comienzo del array.
next	Avanza el puntero interno una posición.
prev	Mueve el puntero interno una posición hacia atrás.
end	Sitúa el puntero interno al final del array.
current	Devuelve el elemento de la posición actual.
key	Devuelve la clave de la posición actual.

Características básicas del lenguaje

- Las funciones **reset, next, prev y end**, además de mover el puntero interno, devuelven, al igual que **current**, el valor del nuevo elemento en que se posiciona.
- Si al mover el puntero **te sales de los límites del array** (por ejemplo, si ya estás en el último elemento y haces un **next**), cualquiera de ellas devuelve false.
- La función **key** devuelve *null* si el puntero está fuera de los límites del array.

```php
<?php
    $transport = array('pie', 'bici', 'coche', 'avión');
    $mode = current($transport); // $mode = 'pie';
    $mode = next($transport);    // $mode = 'bici';
    $mode = next($transport);    // $mode = 'coche';
    $mode = prev($transport);    // $mode = 'bici';
    $mode = end($transport);     // $mode = 'avión';
?>
```

Características básicas del lenguaje

Funciones relacionadas con los arrays

- Podemos eliminar elementos de un array con la función unset.
- En arrays numéricos, esto implica sus las claves ya no estén consecutivas.

```
$vector = array("a","b","c","d","e");
print_r($vector);
echo "<br/>";
unset($vector[2]);
print_r($vector);
```

Salida:

```
Array ( [0] => a [1] => b [2] => c [3] => d [4] => e )
Array ( [0] => a [1] => b [3] => d [4] => e )
```

Características básicas del lenguaje

La función array_values recibe un array como parámetro, y devuelve un nuevo array con los mismos elementos y con índices numéricos consecutivos con base 0.

```
$vector =  array("a","b","c","d","e");
print_r($vector);
echo "<br/>";
unset($vector[2]);
print_r($vector);
echo "<br/>";
$vector2 = array_values($vector);
print_r($vector2);
```

Salida:

```
Array ( [0] => a [1] => b [2] => c [3] => d [4] => e )
Array ( [0] => a [1] => b [3] => d [4] => e )
Array ( [0] => a [1] => b [2] => d [3] => e )
```

Características básicas del lenguaje

- Para comprobar si una variable es de tipo array, se utiliza is_array.
- Para obtener el número de elementos de un array, se usa count.
- Para **buscar un elemento concreto dentro de un array**, se usa in_array. Recibe como primer parámetro el elemento a buscar y como segundo parámetro el array. Devuelve true si lo encuentra, false si no.
- Otra opción es array_search, que recibe los mismos parámetros pero **devuelve la clave** correspondiente al elemento, **o false si no lo encuentra**.
- Para **ordenar los elementos de un array** se puede utilizar sort ordenará los elementos del array alfabéticamente. Para ordenar los elementos en orden alfabético inverso, usaremos rsort. En caso de array asociativo, usar asort, ya que ordena manteniendo la correlación de los índices del array con los elementos con los que están asociados.

Características básicas del lenguaje

Más funciones para gestionar arrays:

http://es.php.net/manual/es/ref.array.php

EJERCICIOS

❖ Crea una página llamada ***ejer07.php*** y con título *Ejercicio07.*

- Usa ***foreach*** para mostrar todos los valores del array **$_SERVER** en una tabla con dos columnas *(ver página siguiente).*
- La primera columna debe contener el índice (o clave), y la segunda su valor o contenido. Usa `<table border=1>` para que se vean las líneas de la tabla (recuerda que en las tablas html cada fila es un `<tr></tr>` y dentro cada celda es un `<td></td>`).
- Además, debajo de la tabla, haz un *echo* respondiendo a estas preguntas: (imprime por pantalla también las preguntas)
  - ¿SERVER es un *array* numérico o asociativo?
  - ¿Podríamos haberlo recorrido con un **for** de toda la vida? ¿Por qué?

Se verá con buenos ojos que uses (a tu criterio) una hoja de estilo (.css) aparte para mejorar el aspecto de la tabla.

HTTP_HOST	127.0.0.1:8080
HTTP_USER_AGENT	Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:69.0) Gecko/20100101 Firefox/69.0
HTTP_ACCEPT	text/html,application/xhtml+xml,application /xml;q=0.9,*/*;q=0.8
HTTP_ACCEPT_LANGUAGE	es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3
HTTP_ACCEPT_ENCODING	gzip, deflate
HTTP_CONNECTION	keep-alive
HTTP_UPGRADE_INSECURE_REQUESTS	1
HTTP_CACHE_CONTROL	max-age=0

❖ Crea una página llamada ***ejer08.php*** con título *Ejercicio08*

Dada la siguiente matriz (array multidimensional):

$m = array(array(1,3,0,8), array(13,5,2), array(18,4,1,9,87), array(6,9));

Recorre la matriz para averiguar la posición que contiene más elementos y, después, muestra los elementos de la citada posición por pantalla.

❖ Crea una página llamada ***ejer09.php*** con título *Ejercicio09*

- *Anteriormente, has mostrado una tabla con el contenido de la variable superglobal **$ _SERVER** haciendo uso de la construcción foreach.*
- *En este ejercicio, muestra la misma información, pero utilizando el puntero interno y sus funciones, pero sin utilizar foreach.*

❖ Crea una página llamada ***ejer10.php*** con título *Ejercicio10*

✓ ***Haz uso en este ejercicio de las funciones del enlace en la pág. 19***

*a) Crea un array llamado **$alumni** que contenga muchos nombres de personas.*

b) Muestra la cantidad de elementos que contiene el array anterior.

c) Crea y muestra una cadena que contenga los nombres de l@s alumn@s existentes en el array, separados por coma y espacio (función implode()).
https://www.php.net/manual/es/function.implode.php

*d) Muestra el array ordenado alfabéticamente **con las claves reales de cada valor**.*

e) Muestra el array en el orden inverso al que fue creado (función array_reverse()).

f) Muestra la posición que tiene un nombre concreto que quieras encontrar.

*g) Crea un nuevo array **$alumnado** donde cada uno de los elementos sea otro array que contenga el **id**, el **nombre** y la **edad** de cada alumn@.*

*h) Crea una tabla en html que se muestre todos los datos de **$alumnado**.*

i) Usa la función array_column() para crear un array indexado que contenga los nombres de l@s alumn@s únicamente, y muéstralo por pantalla con print_r.