## Tipos de variable

$foo = (int) $bar;

$foo = "1";

## Variables

define("PI", 3.14159265);

## Imprimir

| | |
|---|---|
| var_dump($b, $c); | muestra valor de variables |
| print_r ($a); | o mesmo |

## Arrays

$a[]=„a“;

$moneda["francia"]="Franco";

$a[0][1]=„Hola“;

| | |
|---|---|
| reset($array); | Resetea el puntero interno al principio del array. |
| end($array); | Mueve el puntero al último elemento del array. |
| next($array); | Mueve el puntero al proximo elemento del array. |
| prev($array); | Mueve el puntero al elemento previo respecto al actual |
| current($array); | Devuelve el elemento apuntado actualmente por el puntero interno del array. |
| key($array); | Devuelve el índice del elemento apuntado actualmente por el puntero interno del array, si es un vector asociativo devuelve la clave del elemento actual. |
| $keys = array('foo', 5, 10, 'bar'); $a = array_fill_keys($keys, 'banana'); | |
| $array1=each($array) | Devuelve los valores correspondientes al elemento actual del array y mueve el puntero al elemento siguiente. Si es un vector asociativo devuelve clavevalor, si es un vector común devuelve indice-valor |
| array bidimensional | $comida = array( "frutas" => array( "fresas" => '35kcal', "frambuesa" => '43kcal'),"carne" => array( "babilla de ternera" => '131kcal', "ala de pollo" => '217kcal', "bacon" => '682kcal' ));<br>foreach($comida as $tipo => $piezas){ foreach($piezas as $pieza => $calorias){ echo "En $tipo hay $pieza, que tiene $calorias por 100 gramos <br>"; } } |

| Arrays (cont) | |
|---|---|
| sort($array); | Ordena un vector según los valores de sus elementos, si este es asociativo considera claves y valores como elementos comunes (no los distingue). Ordena en orden ascendiente |
| rsort($array); | |
| asort($array); | Ordena un vector según los valores de sus elementos pero manteniendo las asociaciones clave-valor. Ordena los pares ordenados clave-valor según “valor”. |
| arsort($array); | |
| ksort($array); | Ordena un vector asociativo por los valores de sus “claves” teniendo en cuenta las asociaciones clave-valor. |
| uksort($array,funcion); | Ordena un vector asociativo por “clave” usando para comparar las claves la función pasada como parámetro. |
| uasort($array,funcion); | Ordena un vector por los “valores” de sus elementos preservando la relación clave-valor de un array asociativo usando para ordenar la funcion provista por el usuario |
| array=array_pad($mivector,tamaño,valor); | Rellena $mivector con valor hasta que tenga tamaño elementos, si tamaño es positivo completa agregando elementos hacia la derecha, si es negativo completa hacia la izquierda. |
| list($a,$b)=$vector; //$a=1, $b=2 | |
| $vec1=array_merge($array1,$array2,....);- | Si los vectores son asociativos hace una unión de los vectores en donde si 2 o más vectores tienen la misma clave sólo una queda en el vector resultado. Si los vectores no son asociativos (indexados por número) entonces el resultado tiene todos los elementos de los “n” vectores concatenados. |
| $vec1=array_slice($array,offset,cantidad) | Devuelve un sub-vector de $array a partir del offset indicado y con la cantidad de elementos indicada,si cantidad no se especifica devuelve todos los elementos desde offset hasta el fin del vector. |
| $cantidad=count($vector); | |
| $arraysize = sizeof($lista); | |
| $vec array_combine ( array $keys , array $values ) | Crea un arreglo utilizando dos arreglos, uno para llaves y otro para sus valores. |
| shuffle(array); | Desordena en forma aleatoria los elementos de un vector. |

## Arrays (cont)

| | |
|---|---|
| array=array_reverse(array); | |
| array=compact(nombre_var1,nombre_var2,....,nombre_varN); | Crea un vector asociativo cuyas claves son los nombres de las variables y los valores el contenido de las mismas. |
| array_walk($array1,funcion,variable_extra); | array_walk permite aplicar una función a todos y cada uno de los elementos de un vector. |
| $a = array_fill(5, 6, 'banana'); | |
| $key = array_search('green', $array); // $key = 2; | |
| $stack = array("orange", "banana"); array_push($stack, "apple", "raspberry"); | |
| &array1=array_keys($array) | Devuelve un vector con todas las claves de un vector asociativo. |
| &array1=array_values($array) | Devuelve un vector con todos los valores de un vector asociativo |
| Arrayntercambiado = array_flip ( array $array ) | |
| string implode ( array $pieces ) | Une elementos de un array en un string |
| $oferta = array( "idOferta" => "2344", "titulo" => "Oferta de viaje a Tanger", "descripcion" => "Oferta de viaje que incluye Ferry ida y vuelta y alojamiento" ); extract($oferta); / *Obtenemos las siguientes variables: $idOferta, $titulo, $descripcion* / | , convirte la pareja clave/valor de un array asociativo en variables PHP. |
| $clothes = 't-shirt'; $size = 'medium'; $color = 'blue'; $array = compact('clothes', 'size', 'color'); print_r($array); // Array ( [clothes] => t-shirt [size] => medium, [color] => blue ) | hace un array asociativo de variables |

## Funciones con cadena

| | |
|---|---|
| $cadenaMinus=strtolower($cadena); | |
| $str = mb_strtoupper($str); | |
| ucfirst() | converte la primera letra del texto a mayúscula. |
| \n | avance de línea |
| \r | retorno de carro |
| \e | escape |
| \f | avance de página |
| \" | escapar carácter |
| if(preg_match($patron,$cadena[$i])) | |
| echo trim($str,"Hed!"); | elimina los espacios que existan al inicio y/o al final de una cadena. |
| echo preg_replace($pattern, $replacement, $string); | |
| echo str_repeat("-=", 10); | la cadena indicada el número de veces que indica el 2º parámetro. |
| $i=strlen($cadena) | longitud |

## Funciones con cadena (cont)

| | |
|---|---|
| $pos = strpos($mystring, $findme); | Find the position of the first occurrence of a substring in a string |
| strchr ($texto, "u"); | indica la posición de un caracter (2º parámetro) dentro de la cadena (parámetro1) . <? |
| echo str_replace("es", "**", $texto, $reemplazos ); | |
| echo substr($texto,0,5); | Extraer los primeros 6 caracteres. La primera posición es la 0 y la ultima es la 5 |
| $array = explode(",", $cadena); | convierte un string en un array, en función del delimitador indicado (parametro1) |

## Formularios

if(isset($_REQUEST['cadena'])){

$cadena=$_REQUEST['cadena'];

<form action="accion.php" name="reves" method="request">

<input type="text" name="cadena" id="cadena"></input>

| | |
|---|---|
| <input type="submit" value="Enviar" id="enviar"></input> | $enviar= $_REQUEST[‘enviar’]; |

<INPUT TYPE="radio" NAME=“sexo" VALUE=“M“ CHECKED>Mujer

| | |
|---|---|
| <INPUT TYPE="checkbox" NAME="extras[]" VALUE="garaje" CHECKED>Garaje | $extras = $_REQUEST[‘extras’]; foreach($extras as $extra) print(“$extra<BR>\n”); |

<SELECT NAME=“color"> <OPTION VALUE=“rojo" SELECTED>Rojo <OPTION VALUE=“verde">Verde <OPTION VALUE=“azul">Azul </SELECT>

<SELECT MULTIPLE SIZE="3" NAME="idiomas[]"> <OPTION VALUE="ingles" SELECTED>Inglés <OPTION VALUE="frances">Francés <OPTION VALUE="aleman">Alemán <OPTION VALUE="holandes">Holandés </SELECT>

## Funciones número

rand(1,10);

## Estructuras repetitivas

foreach ($arr as &$value) { $value = $value * 2; }

| | |
|---|---|
| foreach ($arr as &$valueUser => $infoUser) { | $arr = array ( 'Usuario 1:' => array( 'nombre' =>"A", 'genero' => 'F') |

foreach $infoUser as $content{}}

## Funciones

function porReferencia(&$cadena) { $cadena = 'Si cambia'; }

## Comprobaci'on de datos

| | |
|---|---|
| if (isset($var)) { | Determine if a variable is set and is not NULL |
| if (is_numeric($numero)) { | Esta función se puede utilizar para comprobar si el dato que se ha recibido es un número (o se puede interpretar como número). |
| ctype_digit($valor) | comprueba si el dato es un número entero positivo (sin decimales). |
| array_key_exists($indice, $matriz) | devuelve true si un elemento determinado de una matriz existe o no. |

## Regex

preg_match($pattern, substr($subject,3), $matches, PREG_OFFSET_CAPTURE);

comprobar si un string empieza con unos caracteres concretos, despues de los delimitadores, el patron comenzará con el signo ^, parea el final $

| | |
|---|---|
| if(preg_match("/^calle/i", $direccion)) | Por defecto, las búsquedas son sensibles a mayúsculas y minúsculas, para cambiar esto se puede añadir una i de insensitive al final. |
| - | Indica rango de caracteres. |
| \d | Cualquier carácter numérico [0-9] |
| \D | Cualquier carácter no numérico [^0-9] |
| \s | Cualquier espacio [\t\n\r\f\v] |
| \S | Cualquiera que no sea espacio [^ \t\n\r\f\v] |
| \w | Cualquier carácter alfanumérico [a-zA-Z0-9_] |
| \W | Cualquier carácter no alfanumérico [^a-zA-Z0-9_] |
| . | Match con cualquier caracter |

echo preg_match_all('/linea\Z/im', $texto); // Devuelve: 1

| | |
|---|---|
| [] | permiten agrupar creando rangos. |
| () | Nos permiten crear sub-expresiones, expresiones regulares contenidas dentro de otras: /a(bc.)+e/. |

echo preg_replace($pattern, $replacement, $string); ?>

## Funciones matemáticas

abs(-5) Devuelve: 5

pow(2,3) Devuelve: 8

. $aNumeros = array(2, 45, 1, 230, 15); max($aNumeros) // Devuelve: 230

$aNumeros = array(2, 45, 23, 1, 230, 15); min($aNumeros) // Devuelve: 1

rand() rand(1, 100) Devuelve un nº aleatorio entre 1 y 100

sqrt(9) devuelve 3

| | |
|---|---|
| $r = fmod($x, $y); | Devuelve el resto en punto flotante (módulo) de la división de los argumentos |

echo floor(4.3); //4

echo ceil(4.3); //5

| | |
|---|---|
| intdiv() | división enetra |

is_nan()