Funciones de cadenas

Las siguientes son algunas funciones que pueden ser de gran utilidad a la hora de trabajar con cadenas.

Funciones generales para cadenas de texto

Estas son las funciones de tipo general para cadenas de texto.

strlen(string $string): int

Cuenta el número de caracteres de una cadena. El resultado es el número de caracteres que tiene la cadena pasada en el argumento**$string**.

```
<?php
$texto = " ab cd ";
echo "<p>Número de caracteres: " .strlen($texto). "</p>";
?>
```

```
<p>Número de caracteres: 7 </p>
```

str_word_count(string $string, int $format = 0, string $charlist = ?): mixed

Analiza las palabras del texto y permite hacer varias operaciones según el modo.

En **$string** pasamos el texto a analizar, y en **$format** podemos poner un número del 0 al 2.

Según el modo usado la función devolverá:

- Modo 0: El número de palabras que contiene el **$string**.
- Modo 1: Un array indexado con todas las palabras que contiene el **$string**.
- Modo 2: Un array asociativo donde la clave es el número de carácter de la cadena donde comienza la palabra (se empieza a contar desde 0), y el valor es la palabra en sí.

$charlist es una lista de caracteres adicionales los cuales serán considerados como de 'palabra'.

```
<?php
 $texto = "Buenos tardes y bienvenidos al curso de PHP";
 $numeroPalabras = str_word_count($texto); echo
 "<p>Número de palabras: $numeroPalabras </p>";
?>
```

```
<p>Número de palabras: 8 </p>
```

```
<?php
$texto = "Buenos días y bienvenidos al curso de móviles";
$numeroPalabras = str_word_count($texto, 0, 'ió'); echo
"<p>Número de palabras: $numeroPalabras </p>";?>
```

```
<p>Número de palabras: 8 </p>
```

```php
<?php
 $texto = "Buenos tardes y bienvenidos al curso de PHP";
 $palabras = str_word_count($texto,1);
 echo "<pre>"; var_dump ($palabras);
 echo "</pre>";
?>
```

```
array(8) {
  [0]=>
  string(6) "Buenos"
  [1]=> string(6)
  "tardes"
  [2]=>
  string(1) "y"
  [3]=>
  string(11) "bienvenidos"
  [4]=>
  string(2) "al"
  [5]=>
  string(5) "curso"
  [6]=>
  string(2) "de"
  [7]=> string(3)
  "PHP"
}
```

```php
<?php
 $texto = "Buenos tardes y bienvenidos al curso de PHP";
 $palabras = str_word_count($texto,2);
 echo "<pre>"; var_dump ($palabras);
 echo "</pre>";
?>
```

```
array(8) { [0]=>
  string(6) "Buenos"
  [7]=> string(6)
  "tardes"
  [14]=>
```

```
  string(1) "y" [16]=>
  string(11) "bienvenidos"
  [28]=>
  string(2) "al"
  [31]=>
  string(5) "curso"
  [37]=>
  string(2) "de"
  [40]=>
  string(3) "PHP"
}
```

strpos(string $haystack, mixed $needle, int $offset = 0): mixed

Busca la subcadena $needle dentro de la cadena **$haystack** y devuelve un número que indica la posición dentro de la cadena del primer carácter de la subcadena. Si está más de una vez se indica sólo el de la primera vez. Si no se encuentra devuelve una cadena vacía.

```
<?php
 $texto = "Buenos tardes y bienvenidos al curso de PHP";
 $cadenaBuscar = "y";
 $posicion = strpos($texto, $cadenaBuscar ); echo
 "<p>Posición de la cadena: $posicion </p>";
?>
```

```
<p>Posición de la cadena: 14 </p>
```

```
<?php
 $texto = "Buenos tardes y bienvenidos al curso de PHP";
 $cadenaBuscar = "hola";
 $posicion = strpos($texto, $cadenaBuscar ); echo
 "<p>Posición de la cadena: $posicion </p>";
?>
```

```
<p>Posición de la cadena: </p>
```

Funciones de cambio de mayúsculas / minúsculas

Las siguientes funciones trabajan con cadenas de texto, y cambian letras mayúsculas y minúsculas:

strtoupper(string $string): string

Devuelve la cadena de texto pasada en el argumento **$string** con todas sus letras en mayúsculas.

```
<?php
$texto = "Hola";
$mayusculas = strtoupper($texto); echo "<p>Texto en
mayúscula: $mayusculas </p>";
?>
```

```
<p>Texto en mayúscula: HOLA </p>
```

strtolower(string $string): string

Devuelve la cadena de texto pasada en el argumento **$string** con todas sus letras en minúsculas.

```
<?php
$texto = "Mañana";
$minusculas = strtolower($texto); echo "<p>Texto en
minúscula: $minusculas </p>";
?>
```

```
<p>Texto en minúsculas: mañana</p>
```

ucfirst(string $string): string

Devuelve la cadena de texto pasada en el argumento **$string** con la primera letra de la cadena en mayúsculas, siempre que ésta sea un carácter alfabético.

```
<?php
$texto = "juan";
$primeraMayuscula = ucfirst($texto); echo "<p>Primera en
mayúscula: $primeraMayuscula </p>";
?>
```

```
<p>Texto en minúsculas: mañana</p>
```

Funciones para reemplazar o eliminar texto

str_replace(mixed $search, mixed $replace, mixed $subject, int &$count = ?): mixed

Reemplaza un trozo de texto por otro. Dentro de la cadena **$subject** (tercer argumento), busca la subcadena **$search** (primer argumento) y la reemplaza por la subcadena **$replace** (segundo argumento).

Si **$search** y **$replace** son arrays, entonces **str_replace()** toma un valor de cada array y lo utiliza para buscar y reemplazar en **$subject**. Si **$replace** tiene menos valores que **$search**, entonces un string vacío es usado para el resto de los valores de reemplazo. Si **$search** es un array y **$replace** es un string, entonces este string de reemplazo es usado para cada valor de **$search**. Sin embargo, lo contrario no tendría sentido.

str_contains (¿Contiene este texto este otro texto?)

```
<?php
$texto = "Buenos tardes y bienvenidos al curso de PHP";
$nuevoTexto = str_replace("tardes", "días", $texto); echo
"<p>$nuevoTexto </p>";
?>
```

```
<p>Buenos días y bienvenidos al curso de PHP</p>
```

```
<?php
 $texto = "Buenos tardes y bienvenidos al curso de PHP";
 $nuevoTexto = str_replace(["tardes", "PHP"],["días", "móviles"],
$texto); echo
 "<p>$nuevoTexto</p>";
?>
```

```
<p>Buenos días y bienvenidos al curso de móviles</p>
```

substr(string $string, int $start, int $length = ?): string
Devuelve una parte de la cadena **$string** definida por los parámetros **$start,**y **$length**.

```
<?php
 $texto = "Buenos tardes y bienvenidos al curso de PHP";
 $nuevoTexto = substr($texto, 16);
 echo "<p>$nuevoTexto</p>";
?>
```

```
<p>bienvenidos al curso de PHP</p>
```

trim(string $string, string $character_mask = " \t\n\r\0\x0B"): string
Esta función devuelve una cadena con los espacios en blanco eliminados del inicio y final de la cadena **string**. Sin el segundo parámetro, trim() eliminará estos caracteres:

- " " (ASCII 32 (0x20)), espacio simple. ● "\t" (ASCII 9 (0x09)), tabulación.
- "\n" (ASCII 10 (0x0A)), salto de línea.
- "\r" (ASCII 13 (0x0D)), retorno de carro.
- "\0" (ASCII 0 (0x00)), el byte NUL.
- "\x0B" (ASCII 11 (0x0B)), tabulación vertical.

De manera opcional, los caracteres a ser eliminados pueden ser especificados usando el parámetro **$character_mask**. Simplemente lista todos los caracteres que se quieran eliminar. Se puede especificar un rango de caracteres usando `..`.

```
<?php
$texto = "   Buenas  ";
```

```
$nuevoTexto = trim($texto); echo
"<p>$nuevoTexto</p>";
?>
```

```
<p>Buenas</p>
```

Funciones para buscar una cadena en otra

str_contains(string $haystack, string $needle): bool

Comprueba si la cadena **$needle** se encuentra contenida en la cadena **$haystack**. En caso de que así sea devuelve true, en caso contrario false.

str_starts_with(string $haystack, string $needle): bool
Comprueba si la cadena **$needle** comienza por la cadena **$haystack**. En caso de que así sea devuelve true, en caso contrario false.

str_ends_with(string $haystack, string $needle): bool
Comprueba si la cadena **$needle** termina por la cadena **$haystack**. En caso de que así sea devuelve true, en caso contrario false.

NOTA: Realiza una comprobación que distingue entre mayúsculas y minúsculas.

```
<?php
$texto = "Buenos tardes y bienvenidos al curso de PHP";
if (str_ends_with($texto, "php")) {
echo "<p>El nombre del curso está en el texto</p>";
} else { echo "<p>El nombre del curso no está en el
texto</p>"; }
?>
```

```
<p>El nombre del curso no está en el texto</p>
```

```
<?php
$texto = "Buenos tardes y bienvenidos al curso de PHP";
if (str_ends_with($texto, "PHP")) {
echo "<p>El nombre del curso está en el texto</p>";
} else { echo "<p>El nombre del curso no está en el
texto</p>"; }
?>
```

```
<p>El nombre del curso está en el texto</p>
```

Otras funciones de cadenas de texto

htmlspecialchars(string $string, int $flags =
ENT_COMPAT | ENT_HTML401, string
$encoding = ini_get("default_charset"), bool
$double_encode = true
): string
Devuelve la cadena **$string** (argumento) en la cual se traducen al lenguaje HTML los siguientes caracteres especiales: & = & " = " < = < > = >

htmlentities(string $string, int $flags =
ENT_COMPAT | ENT_HTML401, string
$encoding = ini_get("default_charset"), bool
$double_encode = true

): string

Devuelve la cadena **$string** (argumento) en la cual se traducen al lenguaje html todos caracteres especiales que se escriben de forma distinta en HTML. Además de los caracteres indicados en la función anterior, se cambian otros como son las vocales con acento o la letra "ñ".

strip_tags(string $string, string $allowable_tags = ?): string

Elimina todas las etiquetas de HTML y PHP que haya en el texto **$string** (primer argumento). El segundo argumento es opcional e indica las etiquetas permitidas. Éstas no serán eliminadas y se escribirán entre los signos < ... > .

```
<?php                                                    Comentarios      --><a
$texto      =      "<p>Párrafo</p><!--
href=\"#fragment\"> Otro texto </a>";
echo strip_tags($texto);

// Permite <p> y <a> echo
strip_tags($texto, '<p><a>');
?>
```

```
Párrafo Otro texto
<p>Párrafo</p>
<a href="#fragment"> Otro texto </a>
```

explode(string $delimiter, string $string, int $limit = PHP_INT_MAX): array

Separa los elementos de la cadena **$string**, (segundo argumento) y los devuelve en un array. El separador, **$delimiter** (primer argumento) indica el carácter o caracteres que, cada vez que aparezcan en el texto se hará una partición de un elemento.

Si el parámetro **$limit** es positivo, el array devuelto contendrá el máximo de **$limit** elementos, y el último elemento contendrá el resto de la cadena

Si el parámetro **$limit** es negativo, se devolverán todos los componentes a excepción del último **$limit**

Si el parámetro **$limit** es cero, se tratará como 1.

```
<?php                                                    porción4   porción5
$pizza    = "porción1 porción2 porción3
porción6";
$porciones = explode(" ", $pizza); echo
"<p>$porciones[0];</p>"; // porción1 echo
"<p>$porciones[1];</p>"; // porción2
?>
```

```
Párrafo Otro texto
<p>porción1</p>
<p>porción2</p>
```

str_split(string $string, int $split_length = 1): array

Convierte un string en un array.

Si el parámetro opcional **split_length** se especifica, el array devuelto será separado en fragmentos los cuales cada uno tendrá una longitud de **split_length**, de otra manera cada fragmento tendrá una longitud de un carácter.

Se devuelve false si **split_length** es menor que 1. Si la longitud **split_length** excede la longitud de string, el string entero es devuelto como el primero (y único) elemento del array.

```
<?php

$cadena = "Hola Amigo";

$array1 = str_split($cadena);
$array2 = str_split($cadena, 3);

echo "<pre>";
print_r($array1);
echo "</pre>";
echo "<pre>";
print_r($array2);
echo "</pre>";

?>
```

```
Array
(
    [0] => H
    [1] => o
    [2] => l
    [3] => a
    [4] =>
    [5] => A
    [6] => m
    [7] => i
    [8] => g
    [9] => o
)
Array
(
    [0] => Hol
    [1] => a A
    [2] => mig
    [3] => o
)
```