Manejo de Cadena de String en PHP

explode — Divide un string en varios string

Devuelve un array de string, siendo cada uno un substring del parámetro string formado por la división realizada por los delimitadores indicados en el parámetro delimiter.

Parámetros
delimiter
El string delimitador.

string
El string de entrada.

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

// Ejemplo 2
$datos = "foo:*:1023:1000::/home/foo:/bin/sh";
list($user, $pass, $uid, $gid, $gecos, $home, $shell) = explode(":", $datos);
echo $user; // foo
echo $pass; // *

?>
```

trim — Elimina espacio en blanco (u otro tipo de caracteres) del inicio y el final de la cadena

Descripción
trim (string $str [, string $character_mask = " \t\n\r\0\x0B"]) : string
Esta función devuelve una cadena con los espacios en blanco eliminados del inicio y final del str. 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.
Parámetros
str
La cadena que será recortada.

character_mask

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

$text   = "\t\tThese are a few words :) ...  ";
$binary = "\x09Example string\x0A";
$hello  = "Hello World";
var_dump($text, $binary, $hello);

print "\n";

$trimmed = trim($text);
var_dump($trimmed);

$trimmed = trim($text, " \t.");
var_dump($trimmed);

$trimmed = trim($hello, "Hdle");
var_dump($trimmed);

$trimmed = trim($hello, 'HdWr');
var_dump($trimmed);

// Elimina los caracteres de control ASCII al inicio y final de $binary
// (from 0 to 31 inclusive)
$clean = trim($binary, "\x00..\x1F");
var_dump($clean);

?>
```

ltrim — Retira espacios en blanco (u otros caracteres) del inicio de un string

Descripción
ltrim (string $str [, string $character_mask]) : string
Retira espacios en blanco (u otros caracteres) del inicio de un string.

Parámetros
str
El string de entrada.

character_mask
Se puede también especificar los caracteres que se desean retirar por medio del parámetro character_mask. Simplemente se listan todos los caracteres que se quieren retirar. Con .. se puede especificar un rango de caracteres.

```
<?php

$text = "\t\tThese are a few words :) ...  ";
$binary = "\x09Example string\x0A";
$hello  = "Hello World";
```

```
var_dump($text, $binary, $hello);

print "\n";


$trimmed = ltrim($text);
var_dump($trimmed);

$trimmed = ltrim($text, " \t.");
var_dump($trimmed);

$trimmed = ltrim($hello, "Hdle");
var_dump($trimmed);

// retira los caracteres ASCII de control al inicio de $binary
// (de 0 a 31 inclusive)
$clean = ltrim($binary, "\x00..\x1F");
var_dump($clean);

?>
```

rtrim — Retira los espacios en blanco (u otros caracteres) del final de un string

Descripción
rtrim (string $str [, string $character_mask]) : string
Esta función devuelve un string con los espacios en blanco retirados del final de str.

Sin el segundo parámetro, rtrim() retirará estos caracteres:

" " (ASCII 32 (0x20)), un espacio ordinario.
"\t" (ASCII 9 (0x09)), un tabulador.
"\n" (ASCII 10 (0x0A)), una nueva línea (line feed).
"\r" (ASCII 13 (0x0D)), un retorno de carro.
"\0" (ASCII 0 (0x00)), el byte NULL.
"\x0B" (ASCII 11 (0x0B)), un tabulador vertical.

Parámetros
str
El string de entrada.

character_mask
Se puede también especificar los caracteres que se desean retirar por medio del parámetro character_mask. Simplemente se listan todos los caracteres que se quieren retirar. Con .. se puede especificar un rango de caracteres.

```
<?php

$text = "\t\tThese are a few words :) ...  ";
$binary = "\x09Example string\x0A";
```

```
$hello  = "Hello World";
var_dump($text, $binary, $hello);

print "\n";

$trimmed = rtrim($text);
var_dump($trimmed);

$trimmed = rtrim($text, " \t.");
var_dump($trimmed);

$trimmed = rtrim($hello, "Hdle");
var_dump($trimmed);

// retira los caracteres ASCII de control al inicio de $binary
// (de 0 a 31 inclusive)
$clean = rtrim($binary, "\x00..\x1F");
var_dump($clean);

?>
```

str_repeat — Repite un string

Descripción
str_repeat (string $input , int $multiplier) : string
Devuelve el input repetido multiplier veces.

Parámetros
input
El string a ser repetido.

multiplier
El número de veces que el string input debe ser repetido.

multiplier debe ser mayor o igual a 0. Si el multiplier se establece en 0, la función devolverá un string vacío.

Valores devueltos
Devuelve el string repetido.

Ejemplos
Ejemplo #1 Ejemplo de str_repeat()

```
<?php
echo str_repeat("-=", 10);
?>
```

str_replace — Reemplaza todas las apariciones del string buscado con el string de reempla

Descripción
str_replace (mixed $search , mixed $replace , mixed $subject [, int &$count]) : mixed
Esta función devuelve un string o un array con todas las apariciones de search en subject reemplazadas con el valor dado de replace.

Si no se necesitan reglas complicadas de reemplazo (como expresiones regulares), se puede utilizar siempre esta función en lugar de preg_replace().

Parámetros
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.

Si search o replace son arrays, sus elementos son procesados del primero al último.

search
El valor a ser buscado, también conocida como la aguja. Un array puede ser utilizado para designar varias agujas.

replace
El valor de reemplazo que sustituye los valores encontrados de search. Un array puede ser utilizado para designar reemplazos múltiples.

subject
El string o array sobre el que se busca y se sustituye, también conocido como el pajar.

Si subject es un array, entonces la búsqueda y reemplazo se realiza con cada entrada de subject y el valor devuelto también es un array.

count
Si es pasado, con este se establece el número de reemplazos realizados.

Valores devueltos
Esta función devuelve un string o un array con los valores sustituidos.

Ejemplos
Ejemplo #1 Ejemplos básicos de str_replace()

```
<?php
// Produce: <body text='black'>
$bodytag = str_replace("%body%", "black", "<body text='%body%'>");
```

```
// Produce: Hll Wrld f PHP
$vowels = array("a", "e", "i", "o", "u", "A", "E", "I", "O", "U");
$onlyconsonants = str_replace($vowels, "", "Hello World of PHP");

// Produce: You should eat pizza, beer, and ice cream every day
$phrase  = "You should eat fruits, vegetables, and fiber every day.";
$healthy = array("fruits", "vegetables", "fiber");
$yummy   = array("pizza", "beer", "ice cream");

$newphrase = str_replace($healthy, $yummy, $phrase);

// Produce: 2
$str = str_replace("ll", "", "good golly miss molly!", $count);
echo $count;
?>
```

str_split — Convierte un string en un array

Descripción
str_split (string $string [, int $split_length = 1]) : array
Convierte un string en un array.

Parámetros
string
El string de entrada.

split_length
Longitud máxima del fragmento.

Valores devueltos
Si el parámetro opcional split_length se específica, 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 caracter.

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.

Ejemplos
Ejemplo #1 Ejemplos de uso de str_split()

```
<?php

$str = "Hello Friend";

$arr1 = str_split($str);
$arr2 = str_split($str, 3);
```

```
print_r($arr1);
print_r($arr2);

?>
```

strripos — Encuentra la posición de la última aparición de un substring insensible a mayúsculas y minúsculas en un string

Descripción
stripos (string $haystack , string $needle [, int $offset = 0]) : mixed
Encuentra la posición numérica de la primera aparición de needle (aguja) en el string haystack (pajar).

A diferencia de strpos(), stripos() no considera las mayúsculas ni las minúsculas.

Parámetros
haystack
El string donde buscar.

needle
Observe que needle puede ser un string de uno o más caracteres.

Si needle no es un string, será convertido a un valor de tipo integer y se aplicará como el valor ordinal de un carácter.

offset
Si se específica, la búsqueda se iniciará en éste número de caracteres contados desde el inicio del string. A diferencia de strrpos() y strripos(), offset no puede ser negativo.

Valores devueltos
Devuelve la posición donde está la aguja, en relación al inicio del string haystack (independiente de offset). Observe también que las posiciones de inicio de los string empiezan en 0, y no en 1.

Devuelve FALSE si no se encontró la aguja.

Advertencia
Esta función puede devolver el valor booleano FALSE, pero también puede devolver un valor no booleano que se evalúa como FALSE. Por favor lea la sección sobre Booleanos para más información. Use el operador === para comprobar el valor devuelto por esta función.

Ejemplos
Ejemplo #1 Ejemplos de stripos()

```
<?php
$findme    = 'a';
$mystring1 = 'xyz';
$mystring2 = 'ABC';
```

```
$pos1 = stripos($mystring1, $findme);
$pos2 = stripos($mystring2, $findme);

// No, 'a' sin duda no está en 'xyz'
if ($pos1 === false) {
    echo "El string '$findme' no se encontró en el strng '$mystring1'";
}

// Observe el uso de ===.  Usar solamente == no funcionará como se espera
// debido a que la posición de 'a' es el 0º (primer) caracter.
if ($pos2 !== false) {
    echo "Se encontró '$findme' en '$mystring2' en la posición $pos2";
}
?>
```

strrpos — Encuentra la posición de la última aparición de un substring en un string

Descripción
strpos (string $haystack , mixed $needle [, int $offset = 0]) : mixed
Encuentra la posición numérica de la primera ocurrencia del needle (aguja) en el string haystack (pajar).

Parámetros
haystack
El string en donde buscar.

needle
Si la needle no es una cadena, es convertida a integer y se interpreta como el valor ordinal de un carácter.

offset
Si se específica, la búsqueda iniciará en éste número de caracteres contados desde el inicio del string. A diferencia de strrpos() y strripos(), el offset no puede ser negativo.

Valores devueltos
Devuelve la posición donde la aguja existe, en relación al inicio del string haystack (independiente del offset). También tener en cuenta que las posiciones de inicio de los string empiezan en 0 y no 1.

Devuelve FALSE si no fue encontrada la aguja.

Advertencia
Esta función puede devolver el valor booleano FALSE, pero también puede devolver un valor no booleano que se evalúa como FALSE. Por favor lea la sección sobre Booleanos para más información. Use el operador === para comprobar el valor devuelto por esta función.

Ejemplos

Ejemplo #1 Usando ===

```
<?php
$mystring = 'abc';
$findme   = 'a';
$pos = strpos($mystring, $findme);

// Nótese el uso de ===. Puesto que == simple no funcionará como se espera
// porque la posición de 'a' está en el 1° (primer) caracter.
if ($pos === false) {
    echo "La cadena '$findme' no fue encontrada en la cadena '$mystring'";
} else {
    echo "La cadena '$findme' fue encontrada en la cadena '$mystring'";
    echo " y existe en la posición $pos";
}
?>
```

strstr — Encuentra la primera aparición de un string

Descripción
strstr (string $haystack , mixed $needle [, bool $before_needle = FALSE]) : string
Devuelve parte del string haystack iniciando desde e incluyendo la primera aparición de needle (aguja) hasta el final del haystack (pajar).

Nota:

Esta función es sensible a mayúsculas. Para búsquedas sin importar las mayúsculas, use stristr().

Nota:

Si solo se quiere saber si un needle determinado aparece en un haystack, se utiliza la función strpos() que es más rápida y requiere menos memoria.

Parámetros
haystack
El string en donde buscar.

needle
Si needle no es una cadena, se convierte a un entero y se aplica como el valor ordinal de un carácter. Este comportamiento está obsoleto a partir de PHP 7.3.0, por lo que su uso está totalmente desaconsejado. Dependiendo del comportamiento previsto, needle deberá ser convertido explícitamente a string, o realizar una llamada explícita a chr().

before_needle
Si se define como TRUE, strstr() devolverá la parte del haystack antes de la primera ocurrencia de needle (excluyendo el needle).

Valores devueltos
Devuelve una parte de un string o FALSE si no se encuentra el needle.

Ejemplos
Ejemplo #1 Ejemplo de strstr()

```
<?php
$email  = 'name@example.com';
$domain = strstr($email, '@');
echo $domain; // mostrará @example.com

$user = strstr($email, '@', true); // Desde PHP 5.3.0
echo $user; // mostrará name
?>
```

strtok — Tokeniza string

Descripción
strtok (string $str , string $token) : string
strtok (string $token) : string
strtok() divide un string (str) en strings más pequeños (tokens), con cada token delimitado por cualquier caracter de token. Es decir, si se tiene un string como "Este es un string de ejemplo", se puede tokenizar en sus palabras individuales utilizando el caracter de espacio como el token.

Parámetros
str
El string a ser dividido en strings más pequeños (tokens).

token
El delimitador usado cuando se divide str.

Valores devueltos
Un token de string.

Ejemplos
Ejemplo #1 Ejemplo de strtok()

```
<?php
$string = "This is\tan example\nstring";
/* Utiliza tabulador y nueva línea como caracteres de tokenización, así  */
$tok = strtok($string, " \n\t");

while ($tok !== false) {
    echo "Word=$tok<br />";
    $tok = strtok(" \n\t");
```

```
}
?>
```

strtolower — Convierte un string a minúsculas

Descripción
strtolower (string $string) : string
Devuelve un string con todos los caracteres alfabéticos convertidos a minúsculas.

Parámetros
string
El string de entrada.

Valores devueltos
Devuelve el string en minúsculas.

Ejemplos
Ejemplo #1 Ejemplo de strtolower()

```
<?php
$str = "Mary Had A Little Lamb and She LOVED It So";
$str = strtolower($str);
echo $str; // Imprime mary had a little lamb and she loved it so
?>
```

strtoupper — Convierte un string a mayúsculas

Descripción
strtoupper (string $string) : string
Devuelve el string con todos los caracteres alfabéticos convertidos a mayúsculas.

Notar que ser 'alfabético' está determinado por la configuración regional actual. Por ejemplo, en la configuración regional por defecto "C" caracteres como la diéresis-a (ä) no se convertirán.

Parámetros
string
El string de entrada.

Valores devueltos
Devuelve el string en mayúsculas.

Ejemplos
Ejemplo #1 Ejemplo de strtoupper()

```
<?php
$str = "Mary Had A Little Lamb and She LOVED It So";
```

```
$str = strtoupper($str);
echo $str; // muestra: MARY HAD A LITTLE LAMB AND SHE LOVED IT SO
?>
```

strtr — Convierte caracteres o reemplaza substrings

Descripción
strtr (string $str , string $from , string $to) : string
strtr (string $str , array $replace_pairs) : string
Si se dan tres argumentos, esta función devuelve una copia de str, donde todas las apariciones de cada carácter (byte simple) en from han sido convertidas al carácter correspondiente en to, es decir, todas las apariciones de $from[$n] han sido reemplazadas con $to[$n], donde $n es un índice válido en ambos argumentos.

Si from y to tienen distinta logitud, se ignoran los caracteres extra del string más largo. La longitud de str será la misma que la del valor devuelto.

Si se dan dos argumentos, el segundo debería ser un array en la forma array('from' => 'to', ...). El valor devuelto es un string donde todas las apariciones de las claves del array han sido reemplazadas por los valores correspondientes. Las claves más largas se probarán primero. Una vez un substring ha sido reemplazado, su nuevo valor no se buscará de nuevo.

En este caso, las claves y los valores pueden tener cualquier longitud, siempre que no haya claves vacías; además, la longitud del valor devuelto podría diferir de la de str. Sin embargo, esta función será la más eficiente cuando todas las claves tienen el mismo tamaño.

Parámetros
string
El string a convertir.

from
El string a convertir a to.

to
El string que reemplaza a from.

replace_pairs
El parámetro replace_pairs se podría usar en lugar de to y from, en cuyo caso sería un array en la forma array('from' => 'to', ...).

Valores devueltos
Devuelve el string convertido.

Si replace_pairs contiene una clave que es un string vacío (""), devolverá FALSE. Si str no es un escalar, no será convertido a un string, se emitirá una advertencia y devolverá NULL.

Ejemplos
Ejemplo #1 Ejemplo de strtr()

```
<?php
//De esta forma, strtr() hace una conversión byte a byte
//Por lo tanto, aquí se asume una codificación de un solo byte:
$addr = strtr($addr, "äåö", "aao");
?>
```

substr_compare — Comparación segura a nivel binario de dos o más strings desde un índice hasta una longitud de caracteres dada

Descripción
substr_compare (string $main_str , string $str , int $offset [, int $length [, bool $case_insensitivity = FALSE]]) : int
substr_compare() compara main_str desde la posición offset con str hasta la cantidad length de caracteres.

Parámetros
main_str
El string principal a comparar.

str
El string secundario a comparar.

offset
La posición de inicio para la comparación. Si es negativa, se comienza a contar desde el final del string.

length
La longitud de la comparación. El valor predeterminado es el que sea mayor entre la longitud de str comparado con la longitud de main_str menos el offset.

case_insensitivity
Si case_insensitivity es TRUE, la comparación no considera el uso de mayúsculas y minúsculas.

Valores devueltos
Devuelve < 0 si main_str, desde la posición offset, es menor que str, > 0 si es mayor que str, y 0 si son iguales. Si offset es igual o mayor que la longitud de main_str, o length se establece y es menor que 1 (antes de PHP 5.5.11), substr_compare() muestra una advertencia y devuelve FALSE.

Ejemplos
Ejemplo #1 Un ejemplo de substr_compare()

```
<?php
echo substr_compare("abcde", "bc", 1, 2); // 0
echo substr_compare("abcde", "de", -2, 2); // 0
echo substr_compare("abcde", "bcg", 1, 2); // 0
echo substr_compare("abcde", "BC", 1, 2, true); // 0
echo substr_compare("abcde", "bc", 1, 3); // 1
```

```
echo substr_compare("abcde", "cd", 1, 2); // -1
echo substr_compare("abcde", "abc", 5, 1); // advertencia
?>
```

substr_count — Cuenta el número de apariciones del substring

Descripción
substr_count (string $haystack , string $needle [, int $offset = 0 [, int $length]]) : int
substr_count() devuelve el número de veces en que el substring needle aparece en el string haystack. Por favor nótese que needle es sensible a mayúsculas y minúsculas.

Nota:

Esta función no cuenta las subcadenas que se traslapan. Véase el ejemplo de abajo!

Parámetros
haystack
El string dentro del cual buscar

needle
El substring a buscar

offset
El desplazamiento por dónde empezar a contar

length
La longitud máxima después del desplazamiento especificado para buscar el substring. Se emite una advertencia si offset más length es mayor que la longitud de haystack.

Valores devueltos
Esta función devuelve un integer.

Ejemplos
Ejemplo #1 Ejemplo de substr_count()

```
<?php
$text = 'This is a test';
echo strlen($text); // 14

echo substr_count($text, 'is'); // 2

// el string es reducido a 's is a test', así que muestra 1
echo substr_count($text, 'is', 3);

// el texto es reducido a 's i', así que muestra 0
echo substr_count($text, 'is', 3, 3);
```

```
// genera una advertencia debido a que 5+10 > 14
echo substr_count($text, 'is', 5, 10);


// muestra sólo 1, debido a que no cuenta subcadenas traslapadas.
$text2 = 'gcdgcdgcd';
echo substr_count($text2, 'gcdgcd');
?>
```

substr_replace — Reemplaza el texto dentro de una porción de un string

Descripción
substr_replace (mixed $string , mixed $replacement , mixed $start [, mixed $length]) : mixed
substr_replace() reemplaza una copia de string delimitada por los parámetros start y (opcionalmente) length con el string dado en replacement.

Parámetros
string
El string de entrada.

Un array de strings puede ser proporcionado, en el caso de que las sustituciones ocurran, a su vez, en cada string. En este caso, los parámetros replacement, start y length pueden ser proporcionados ya sea como valores escalares que serán aplicados a cada string de entrada, a su vez, o como arrays, en cuyo caso el correspondiente elemento del array será usado para cada string de entrada.

replacement
El string de reemplazo.

start
Si start es positivo, el reemplazo iniciará en el startésimo desplazamiento dentro del string.

Si start es negativo, el reemplazo iniciará en el startésimo caracter desde el final del string.

length
Si se da y es positivo, representa la longitud de la porción de string que se va a reemplazar. Si es negativo, representa el número de caracteres desde el final del string en el cual se deja de sustituir. Si no se da, entonces se usará por defecto strlen(string); es decir que la sustitución terminará en el final de string. Por supuesto, si length es cero, entonces esta función tendrá el efecto de la inserción de replacement dentro de string en el desplazamiento dado por start.

Valores devueltos
El string del resultado es devuelto. Si string es un array entonces un array es devuelto.

Ejemplos
Ejemplo #1 Ejemplo desubstr_replace()

```
<?php
```

```
$var = 'ABCDEFGH:/MNRPQR/';
echo "Original: $var<hr />\n";

/* Estos dos ejemplos reemplazan todo $var por 'bob'. */
echo substr_replace($var, 'bob', 0) . "<br />\n";
echo substr_replace($var, 'bob', 0, strlen($var)) . "<br />\n";

/* Inserta 'bob' justo al comienzo de $var. */
echo substr_replace($var, 'bob', 0, 0) . "<br />\n";

/* Estos dos siguientes reemplazan 'MNRPQR' en $var por 'bob'. */
echo substr_replace($var, 'bob', 10, -1) . "<br />\n";
echo substr_replace($var, 'bob', -7, -1) . "<br />\n";

/* Elimina 'MNRPQR' de $var. */
echo substr_replace($var, '', 10, -1) . "<br />\n";
?>
```

substr — Devuelve parte de una cadena

Descripción
substr (string $string , int $start [, int $length]) : string
Devuelve una parte del string definida por los parámetros start y length.

Parámetros
string
La cadena de entrada. Debe ser de almenos de un caracter.

start
Si start no es negativo, la cadena devuelta comenzará en el start de la posición del string empezando desde cero. Por ejemplo, en la cadena 'abcdef', el carácter en la posición 0 es 'a', el carácter en la posición 2 es 'c', y así sucesivamente.

Si start es negativo, la cadena devuelta empezará en start contando desde el final de string.

Si la longitud del string es menor que start, la función devolverá FALSE.

Ejemplo #1 Usando un start negativo

```
<?php
$rest = substr("abcdef", -1);    // devuelve "f"
$rest = substr("abcdef", -2);    // devuelve "ef"
$rest = substr("abcdef", -3, 1); // devuelve "d"
?>
```

length

Si se especifica el length y es positivo, la cadena devuelta contendrá como máximo de caracteres de la cantidad dada por length que comienza en start (dependiedo de la longitud del string).

Si se especifica length es negativo, entonces ese número de caracteres se omiten al final del string (después de la posición inicial se ha calculado a start es negativo). Si start indica la posición de su truncamiento o más allá, se devolverá FALSE.

Si se omite el length, la subcadena empezará por start hasta el final de la cadena donde será devuelta.

Si se especifica length y es 0, FALSE o NULL devolverá una cadena vacía.

Ejemplo #2 Usando un length negativo

```
<?php
$rest = substr("abcdef", 0, -1);  // devuelve "abcde"
$rest = substr("abcdef", 2, -1);  // devuelve "cde"
$rest = substr("abcdef", 4, -4);  // devuelve false
$rest = substr("abcdef", -3, -1); // devuelve "de"
?>
```

Valores devueltos
Devuelve la parte extraída de string; o FALSE en caso de error o un string vacío.

Ejemplos
Ejemplo #3 Uso básico de substr()

```
<?php
echo substr('abcdef', 1);     // bcdef
echo substr('abcdef', 1, 3);  // bcd
echo substr('abcdef', 0, 4);  // abcd
echo substr('abcdef', 0, 8);  // abcdef
echo substr('abcdef', -1, 1); // f

// El acceso a caracteres específicos en una cadena
// se puede conseguir usando "corchetes"
$string = 'abcdef';
echo $string[0];                 // a
echo $string[3];                 // d
echo $string[strlen($string)-1]; // f

?>
```

ucfirst — Convierte el primer caracter de una cadena a mayúsculas

Descripción
ucfirst (string $str) : string

Devuelve una cadena con el primer caracter str en máyusculas, si el caracter es alfabético.

Nótese que 'alfabético' se determina por la localización actual. Por ejemplo, los carácteres de localización "C" como umlaut-a (ä) no serán convertidos.

Parámetros
str
La cadena de entrada.

Valores devueltos
Devuelve la cadena resultante.

Ejemplos
Ejemplo #1 Ejemplo de ucfirst()

```
<?php
$foo = 'hello world!';
$foo = ucfirst($foo);             // Hello world!

$bar = 'HELLO WORLD!';
$bar = ucfirst($bar);             // HELLO WORLD!
$bar = ucfirst(strtolower($bar)); // Hello world!
?>
```

ucwords — Convierte a mayúsculas el primer carácter de cada palabra de una cadena

Descripción
ucwords (string $str [, string $delimiters = " \t\r\n\f\v"]) : string

Devuelve una cadena con la primera letra de cada palabra de str convertida a mayúsculas, si el carácter es alfanumérico.

La definición de una palabra es una cadena de caracteres que está inmediatamente después de cualquier carácter enumerado en el parámetro delimiters (Por defecto son: espacio, avance de página, nueva línea, retorno de carro, tabulación horizontal y tabulación vertical).

Parámetros
str
La cadena de entrada.

delimiters
El parámetro opcional delimiters contiene los caracteres separadores de palabras.

Valores devueltos
Devuelve la cadena modificada.

Ejemplos
Ejemplo #1 Ejemplo de ucwords()

```
<?php
$foo = 'hello world!';
$foo = ucwords($foo);             // Hello World!

$bar = 'HELLO WORLD!';
$bar = ucwords($bar);             // HELLO WORLD!
$bar = ucwords(strtolower($bar)); // Hello World!
?>
```