trim »
« substr_replace

- Manual de PHP
- Referencia de funciones
- Procesamiento de texto
- Strings
- Funciones de strings

Change language: Spanish

Submit a Pull Request Report a Bug

substr

(PHP 4, PHP 5, PHP 7, PHP 8)

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.

Historial de cambios ¶

Versión	Descripción
7.0.0	Si la longitud de caracteres de `string` es igual a `start`, se devolverá una cadena vacía. Antes de esta versión, en este caso se devolvía **`false`**.
5.2.2 - 5.2.6	Si el parámetro `start` indica una posición negativa de truncamiento o más allá, se devolverá false. En otras versiones de PHP obtienen la cadena desde el principio.

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

?>
```

Ejemplo #4 Comportamiento de casting de substr()

```
<?php
class apple {
    public function __toString() {
        return "green";
    }
}

echo "1) ".var_export(substr("pear", 0, 2), true).PHP_EOL;
echo "2) ".var_export(substr(54321, 0, 2), true).PHP_EOL;
echo "3) ".var_export(substr(new apple(), 0, 2), true).PHP_EOL;
echo "4) ".var_export(substr(true, 0, 1), true).PHP_EOL;
```

```
echo "5) ".var_export(substr(false, 0, 1), true).PHP_EOL;
echo "6) ".var_export(substr("", 0, 1), true).PHP_EOL;
echo "7) ".var_export(substr(1.2e3, 0, 4), true).PHP_EOL;
?>
```

Salida del ejemplo anterior en PHP 7:

```
1) 'pe'
2) '54'
3) 'gr'
4) '1'
5) ''
6) ''
7) '1200'
```

Salida del ejemplo anterior en PHP 5:

```
1) 'pe'
2) '54'
3) 'gr'
4) '1'
5) false
6) false
7) '1200'
```

Errores/Excepciones ¶

Devuelve **`false`** en caso de error.

```
<?php
var_dump(substr('a', 2)); // bool(false)
?>
```

Ver también ¶

- strrchr() - Encuentra la última aparición de un caracter en un string
- substr_replace() - Reemplaza el texto dentro de una porción de un string
- preg_match() - Realiza una comparación con una expresión regular
- trim() - Elimina espacio en blanco (u otro tipo de caracteres) del inicio y el final de la cadena
- mb_substr() - Obtiene parte de una cadena de caracteres
- wordwrap() - Ajusta un string hasta un número dado de caracteres
- Acceso a cadenas y modificación por caracter

+ add a note

User Contributed Notes 36 notes

up
down
159
Andreas Bur (andreas dot buro at gmail dot com) ¶
13 years ago

For getting a substring of UTF-8 characters, I highly recommend mb_substr

```
<?php
        $utf8string = "cakeæøå";

        echo substr($utf8string,0,5);
        // output cake#
        echo mb_substr($utf8string,0,5,'UTF-8');
```

```
        //output cakeæ
?>
```

up
down
116
biohazard dot ge at gmail dot com ¶
9 years ago

may be by following functions will be easier to extract the needed sub parts from a string:

```
<?php
after ('@', 'biohazard@online.ge');
//returns 'online.ge'
//from the first occurrence of '@'

before ('@', 'biohazard@online.ge');
//returns 'biohazard'
//from the first occurrence of '@'

between ('@', '.', 'biohazard@online.ge');
//returns 'online'
//from the first occurrence of '@'

after_last ('[', 'sin[90]*cos[180]');
//returns '180]'
//from the last occurrence of '['

before_last ('[', 'sin[90]*cos[180]');
//returns 'sin[90]*cos['
//from the last occurrence of '['

between_last ('[', ']', 'sin[90]*cos[180]');
//returns '180'
//from the last occurrence of '['
?>
```

here comes the source:

```
<?php

    function after ($this, $inthat)
    {
        if (!is_bool(strpos($inthat, $this)))
        return substr($inthat, strpos($inthat,$this)+strlen($this));
    };

    function after_last ($this, $inthat)
    {
        if (!is_bool(strrevpos($inthat, $this)))
        return substr($inthat, strrevpos($inthat, $this)+strlen($this));
    };

    function before ($this, $inthat)
    {
        return substr($inthat, 0, strpos($inthat, $this));
    };

    function before_last ($this, $inthat)
    {
```

```
        return substr($inthat, 0, strrevpos($inthat, $this));
    };

    function between ($this, $that, $inthat)
    {
        return before ($that, after($this, $inthat));
    };

    function between_last ($this, $that, $inthat)
    {
     return after_last($this, before_last($that, $inthat));
    };

// use strrevpos function in case your php version does not include it
function strrevpos($instr, $needle)
{
    $rev_pos = strpos (strrev($instr), strrev($needle));
    if ($rev_pos===false) return false;
    else return strlen($instr) - $rev_pos - strlen($needle);
};
?>
```

up
down
3
bleakwind at msn dot com ¶
17 years ago

This returns the portion of str specified by the start and length parameters..
It can performs multi-byte safe on number of characters. like mb_strcut() ...

Note:
1.Use it like this bite_str(string str, int start, int length [,byte of on string]);
2.First character's position is 0. Second character position is 1, and so on...
3.$byte is one character length of your encoding, For example: utf-8 is "3", gb2312 and big5 is "2"...you can use the function strlen() get it...
Enjoy it :) ...

--- Bleakwind
QQ:940641
http://www.weaverdream.com

PS:I'm sorry my english is too poor... :(

```
<?php
// String intercept By Bleakwind
// utf-8:$byte=3 | gb2312:$byte=2 | big5:$byte=2
function bite_str($string, $start, $len, $byte=3)
{
    $str     = "";
    $count   = 0;
    $str_len = strlen($string);
    for ($i=0; $i<$str_len; $i++) {
        if (($count+1-$start)>$len) {
            $str  .= "...";
            break;
        } elseif ((ord(substr($string,$i,1)) <= 128) && ($count < $start)) {
            $count++;
        } elseif ((ord(substr($string,$i,1)) > 128) && ($count < $start)) {
            $count = $count+2;
```

```
            $i     = $i+$byte-1;
        } elseif ((ord(substr($string,$i,1)) <= 128) && ($count >= $start)) {
            $str  .= substr($string,$i,1);
            $count++;
        } elseif ((ord(substr($string,$i,1)) > 128) && ($count >= $start)) {
            $str  .= substr($string,$i,$byte);
            $count = $count+2;
            $i     = $i+$byte-1;
        }
    }
    return $str;
}

// Test
$str = "123456???123456????123456??d????";
for($i=0;$i<30;$i++){
    echo "<br>".bite_str($str,$i,20);
}
?>
```

up
down
15

pugazhenthi k ¶
9 years ago

```
<?Php

### SUB STRING  BY WORD USING substr() and strpos()  #####

### THIS SCRIPT WILL RETURN PART OF STRING  WITHOUT WORD BREAK ###

$description = ‘your description here your description here your description here your description
here your description here your description here your description hereyour description here your
description here’  // your description here .

$no_letter = 30 ;

if(strlen($desctiption) > 30 )
{
     echo substr($description,0,strpos($description,’ ‘,30));             //strpos to find ‘ ‘
after 30 characters.
}
else {
     echo $description;
}

?>
```

up
down
6

nikolai dot wuestemann at t-online dot de ¶
11 years ago

If you want to have a string BETWEEN two strings, just use this function:

```
<?php
function get_between($input, $start, $end)
{
  $substr = substr($input, strlen($start)+strpos($input, $start), (strlen($input) - strpos($input,
$end))*(-1));
```

```
  return $substr;
}

//Example:

$string = "123456789";
$a = "12";
$b = "9";

echo get_between($string, $a, $b);

//Output:
//345678
?>
```

up
down
9

greg at apparel dot com ¶
8 years ago

Coming to PHP from classic ASP I am used to the Left() and Right() functions built into ASP so I did a quick PHPversion. hope these help someone else making the switch

```
function left($str, $length) {
    return substr($str, 0, $length);
}

function right($str, $length) {
    return substr($str, -$length);
}
```

up
down
8

Petez ¶
15 years ago

I wanted to work out the fastest way to get the first few characters from a string, so I ran the following experiment to compare substr, direct string access and strstr:

```
<?php
/* substr access */
beginTimer();
for ($i = 0; $i < 1500000; $i++){
    $opening = substr($string,0,11);
    if ($opening == 'Lorem ipsum'){
        true;
    }else{
        false;
    }
}
$endtime1 = endTimer();

/* direct access */
beginTimer();
for ($i = 0; $i < 1500000; $i++){
    if ($string[0] == 'L' && $string[1] == 'o' && $string[2] == 'r' && $string[3] == 'e' &&
$string[4] == 'm' && $string[5] == ' ' && $string[6] == 'i' && $string[7] == 'p' && $string[8] ==
's' && $string[9] == 'u' && $string[10] == 'm'){
        true;
    }else{
```

```
        false;
    }
}
$endtime2 = endTimer();

/* strstr access */
beginTimer();
for ($i = 0; $i < 1500000; $i++){
    $opening = strstr($string,'Lorem ipsum');
    if ($opening == true){
        true;
    }else{
        false;
    }
}
$endtime3 = endTimer();

echo $endtime1."\r\n".$endtime2."\r\n".$endtime3;
?>

The string was 6 paragraphs of Lorem Ipsum, and I was trying match the first two words. The
experiment was run 3 times and averaged. The results were:

(substr) 3.24
(direct access) 11.49
(strstr) 4.96

(With standard deviations 0.01, 0.02 and 0.04)

THEREFORE substr is the fastest of the three methods for getting the first few letters of a
string.
```

up
down
3
gkhelloworld at gmail dot com ¶
13 years ago

```
Shortens the filename and its expansion has seen.

<?php
$file = "Hellothisfilehasmorethan30charactersandthisfayl.exe";

function funclongwords($file)
{
if (strlen($file) > 30)
{
$vartypesf = strrchr($file,".");
$vartypesf_len = strlen($vartypesf);
$word_l_w = substr($file,0,15);
$word_r_w = substr($file,-15);
$word_r_a = substr($word_r_w,0,-$vartypesf_len);

return $word_l_w."...".$word_r_a.$vartypesf;
}
else
return $file;
}
// RETURN: Hellothisfileha...andthisfayl.exe
?>
```

up
down
3
kaysar in ymail in com ¶
13 years ago
Drop extensions of a file (even from a file location string)

```
<?php

$filename = "c:/some dir/abc defg. hi.jklmn";

echo substr($filename, 0, (strlen ($filename)) - (strlen (strrchr($filename,'.'))));

?>
```

output: c:/some dir/abc defg. hi

Hope it may help somebody like me.. (^_^)

up
down
2
Anonymous ¶
5 years ago
Be aware of a slight inconsistency between substr and mb_substr

```
mb_substr("", 4);      returns empty string

substr("", 4);              returns boolean false
```

tested in PHP 7.1.11 (Fedora 26) and PHP 5.4.16 (CentOS 7.4)

up
down
3
fatihmertdogancan at hotmail dot com ¶
8 years ago
[English]
I created python similar accesing list or string with php substr & strrev functions.

Use: str($string,$pattern)

About the python pattern,
http://docs.python.org/release/1.5.1p1/tut/strings.html
http://effbot.org/zone/python-list.htm

About of pattern structures
[start:stop:step]

Example,

```
<?php
$s = "fatihmertdogancan";
echo str($s,"1:9:-2");
echo "<br/>";
echo str($s,"1:-3:-2");
echo "<br/>";
echo str($s,"1:-11:-5");
echo "<br/>";
echo str($s,"1:9:4");
?>
```

Output,
thetoacn
eht
aom
htan

This is function phpfiddle link: http://phpfiddle.org/main/code/e82-y5d

or source;

```
<?php
    function str($str,$pattern){
        //[start:stop:step]
        //pattern ->            ([-]?[0-9]*|\s):([-]?[0-9]*|\s):([-]?[0-9]*|\s)
        preg_match("/([-]?[0-9]*|\s?):([-]?[0-9]*|\s?):?([-]?[0-9]*|\s?)/", $pattern, $yakala);
        $start = $yakala[1];
        $stop = $yakala[2];
        $step = $yakala[3];

        if(empty($start) && empty($stop) && $step == "-1"){//istisna durum
            return strrev($str);
        }else if(empty($start) && empty($stop) && isset($step)){//istisna durum
            $rev = "";
            $yeni = "";
            if($step[0] == "-" && $stop != "-1"){$rev = "VAR";}
            $atla = abs($step);
            for($i = 0; $i <= strlen($str); $i++){
                $offset = $i*$atla;
                if(isset($str[$offset])){
                    $yeni = $yeni.$str[$offset];
                }
            }
            if($rev != "VAR"){
                return substr($yeni,0,strlen($str)-1);
                //"hepsi boş, step dolu o da +";
            }else{
                return strrev(substr($yeni,0,strlen($str)-1));
                //"hepsi boş, step dolu o da -";
            }
        }

        if(empty($start) && empty($stop) && empty($step)){
            return $str;
            //"hepsi boş";
        }else if(empty($start)){
            if(isset($stop) && empty($step)){
                $rev = "";
                if($stop[0] == "-"){$rev = "VAR";}
                if($rev != "VAR"){
                    return substr($str,0,$stop);
                    //"start ve step boş, stop dolu"
                }else{
                    return strrev(substr($str,0,$stop));
                    //"start ve step boş, stop -1";
                }
            }else if(isset($stop) && isset($step)){
                $rev = "";
```

```
            if($stop[0] == "-"){$rev = "VAR";}
            $yeni = "";
            if($step == 1){
                if($rev != "VAR"){
                    return $str;
                    //"start boş, stop ve step dolu, step 1";
                }else{
                    return strrev(substr($str,0,abs($stop))); //abs -> mutlak değer (-5 = 5)
                    //"start boş, stop -, step dolu, step 1";
                }
            }else{
                $atla = abs($step);
                for($i = 0; $i <= strlen($str); $i++){
                    $offset = $i*$atla;
                    if(isset($str[$offset])){
                        $yeni = $yeni.$str[$offset];
                    }
                }
                if($rev != "VAR"){
                    return substr($yeni,0,$stop);
                    //"start boş, step ve stop dolu";
                }else{
                    return strrev(substr($yeni,0,abs($stop)));
                    //"start boş, step ve stop -";
                }
            }
        }
    //start boş değilse
    }else if(!empty($start)){
        if(isset($stop) && empty($step)){
            $rev = "";
            if($stop[0] == "-"){$rev = "VAR";}
            if($rev != "VAR"){
                return substr($str,$start,$stop);
                //return "step boş, start ve stop dolu";
            }else{
                return strrev(substr($str,0,abs($stop)));
                //"step boş, start ve stop dolu, stop -";
            }
        }else if(isset($stop) && isset($step)){

            //hepsi dolu
            $rev = "";
            if($stop[0] == "-"){$rev = "VAR";}
            $yeni = "";
            if($step == 1){
                if($rev != "VAR"){
                    return substr($str,$start,$stop);
                    //"hepsi dolu, step 1";
                }else{
                    return substr($str,$start,abs($stop));
                    //"hepsi dolu, step 1, stop -";
                }
            }else{
                if($stop[0] == "-"){$rev = "VAR";}
                $atla = abs($step);
                for($i = 0; $i <= strlen($str); $i++){
                    $offset = $i*$atla;
```

```
                        if(isset($str[$offset])){
                            $yeni = $yeni.$str[$offset];
                        }
                    }
                    if($rev != "VAR"){
                        return substr($yeni,$start,$stop);
                        //"hepsi dolu";
                    }else{
                        return strrev(substr($yeni,$start,abs($stop)));
                        //"hepsi dolu, stop -";
                    }
                }
            }
        }
    }
?>
```

Good works..

up
down
6

fanfatal at fanfatal dot pl ¶
17 years ago

Hmm ... this is a script I wrote, whitch is very similar to substr, but it isn't takes html and bbcode for counting and it takes portion of string and show avoided (html & bbcode) tags too ;]
Specially usefull for show part of serach result included html and bbcode tags

```
<?php

/**
* string csubstr ( string string, int start [, int length] )
*
* @author FanFataL
* @param string string
* @param int start
* @param [int length]
* @return string
*/
function csubstr($string, $start, $length=false) {
    $pattern = '/(\[\w+[^\]]*?\]|\[\/\w+\]|<\w+[^>]*?>|<\/\w+>)/i';
    $clean = preg_replace($pattern, chr(1), $string);
    if(!$length)
        $str = substr($clean, $start);
    else {
        $str = substr($clean, $start, $length);
        $str = substr($clean, $start, $length + substr_count($str, chr(1)));
    }
    $pattern = str_replace(chr(1),'(.*?)',preg_quote($str));
    if(preg_match('/'.$pattern.'/is', $string, $matched))
        return $matched[0];
    return $string;
}

?>
```

Using this is similar to simple substr.

```
Greatings ;]
...
```

up
down
3

link ¶

13 years ago

I created some functions for entity-safe splitting+lengthcounting:

```
<?php
function strlen_entities($text)
{
    preg_match_all(
        '/((?:&(?:#[0-9]{2,}|[a-z]{2,});)|(?:[^&])|'.
        '(?:&(?!\w;)))s',$text,$textarray);
    return count($textarray[0]);
}
function substr_entities($text,$start,$limit=0)
{
    $return = '';
    preg_match_all(
        '/((?:&(?:#[0-9]{2,}|[a-z]{2,});)|(?:[^&])|'.
        '(?:&(?!\w;)))s',$text,$textarray);
    $textarray = $textarray[0];
    $numchars = count($textarray)-1;
    if ($start>=$numchars)
        return false;
    if ($start<0)
    {
        $start = ($numchars)+$start+1;
    }
    if ($start>=0)
    {
        if ($limit==0)
        {
            $end=$numchars;
        }
        elseif ($limit>0)
        {
            $end = $start+($limit-1);
        }
        else
        {
            $end = ($numchars)+$limit;
        }

        for ($i=$start;$i<=$end;$i++)
        {
            $return .= $textarray[$i];
        }
        return $return;
    }
}
?>
```

up
down
2

pheagey at gmail dot com ¶

10 years ago

Using a 0 as the last parameter for substr().

As per examples
```
<?php $var = substr($var, 4); ?>
```

works no problem. However
```
<?php $var = substr($var, 4, 0); ?>
```

will get you nothing. Just a quick heads up

up
down
2

egingell at sisna dot com ¶

16 years ago

```
<?php

/**
* string substrpos(string $str, mixed $start [[, mixed $end], boolean $ignore_case])
*
* If $start is a string, substrpos will return the string from the position of the first occuring
$start to $end
*
* If $end is a string, substrpos will return the string from $start to the position of the first
occuring $end
*
* If the first character in (string) $start or (string) $end is '-', the last occuring string will
be used.
*
* If $ignore_case is true, substrpos will not care about the case.
* If $ignore_case is false (or anything that is not (boolean) true, the function will be case
sensitive.
*        Both of the above: only applies if either $start or $end are strings.
*
* echo substrpos('This is a string with 0123456789 numbers in it.', 5, '5');
*        // Prints 'is a string with 01234';
*
* echo substrpos('This is a string with 0123456789 numbers in it.', '5', 5);
*        // Prints '56789'
*
* echo substrpos('This is a string with 0123456789 numbers in it and two strings.', -60, '-
string')
*        // Prints 's is a string with 0123456789 numbers in it and two '
*
* echo substrpos('This is a string with 0123456789 numbers in it and two strings.', -60, '-
STRING', true)
*        // Prints 's is a string with 0123456789 numbers in it and two '
*
* echo substrpos('This is a string with 0123456789 numbers in it and two strings.', -60, '-
STRING', false)
*        // Prints 's is a string with 0123456789 numbers in it and two strings.'
*
* Warnings:
*        Since $start and $end both take either a string or an integer:
*            If the character or string you are searching $str for is a number, pass it as a
quoted string.
*        If $end is (integer) 0, an empty string will be returned.
*        Since this function takes negative strings ('-search_string'):
```

```
 *            If the string your using in $start or $end is a '-' or begins with a '-' escape it
 with a '\'.
 *            This only applies to the *first* character of $start or $end.
 */

 // Define stripos() if not defined (PHP < 5).
 if (!is_callable("stripos")) {
     function stripos($str, $needle, $offset = 0) {
         return strpos(strtolower($str), strtolower($needle), $offset);
     }
 }

 function substrpos($str, $start, $end = false, $ignore_case = false) {
     // Use variable functions
     if ($ignore_case === true) {
         $strpos = 'stripos'; // stripos() is included above in case it's not defined (PHP < 5).
     } else {
         $strpos = 'strpos';
     }

     // If end is false, set it to the length of $str
     if ($end === false) {
         $end = strlen($str);
     }

     // If $start is a string do what's needed to make it an integer position for substr().
     if (is_string($start)) {
         // If $start begins with '-' start processing until there's no more matches and use the
 last one found.
         if ($start{0} == '-') {
             // Strip off the '-'
             $start = substr($start, 1);
             $found = false;
             $pos = 0;
             while(($curr_pos = $strpos($str, $start, $pos)) !== false) {
                 $found = true;
                 $pos = $curr_pos + 1;
             }
             if ($found === false) {
                 $pos = false;
             } else {
                 $pos -= 1;
             }
         } else {
             // If $start begins with '\-', strip off the '\'.
             if ($start{0} . $start{1} == '\-') {
                 $start = substr($start, 1);
             }
             $pos = $strpos($str, $start);
         }
         $start = $pos !== false ? $pos : 0;
     }

     // Chop the string from $start to strlen($str).
     $str = substr($str, $start);

     // If $end is a string, do exactly what was done to $start, above.
     if (is_string($end)) {
```

```
        if ($end{0} == '-') {
            $end = substr($end, 1);
            $found = false;
            $pos = 0;
            while(($curr_pos = strpos($str, $end, $pos)) !== false) {
                $found = true;
                $pos = $curr_pos + 1;
            }
            if ($found === false) {
                $pos = false;
            } else {
                $pos -= 1;
            }
        } else {
            if ($end{0} . $end{1} == '\-') {
                $end = substr($end, 1);
            }
            $pos = $strpos($str, $end);
        }
        $end = $pos !== false ? $pos : strlen($str);
    }

    // Since $str has already been chopped at $start, we can pass 0 as the new $start for substr()
    return substr($str, 0, $end);
}

?>
```

up
down
4
mar dot czapla at gmail dot com ¶
14 years ago

Here we have gr8 function which simply convert ip address to a number using substr with negative offset.
You can need it if you want to compare some IP addresses converted to a numbers.
For example when using ip2country, or eliminating same range of ip addresses from your website :D

```
<?php

function ip2no($val)
{
    list($A,$B,$C,$D)    =    explode(".",$val);
    return
        substr("000".$A,-3).
        substr("000".$B,-3).
        substr("000".$C,-3).
        substr("000".$D,-3);
}

$min        =    ip2no("10.11.1.0");
$max        =    ip2no("111.11.1.0");
$visitor    =    ip2no("105.1.20.200");

if($min<$visitor && $visitor<$max)
    {    echo 'Welcome !';    }
else
    {    echo 'Get out of here !';    }
```

```
?>
```

up
down
4
slow at acedsl dot com ¶
11 years ago

Anyone coming from the Python world will be accustomed to making substrings by using a "slice index" on a string. The following function emulates basic Python string slice behavior. (A more elaborate version could be made to support array input as well as string, and the optional third "step" argument.)

```
<?php

function py_slice($input, $slice) {
    $arg = explode(':', $slice);
    $start = intval($arg[0]);
    if ($start < 0) {
        $start += strlen($input);
    }
    if (count($arg) === 1) {
        return substr($input, $start, 1);
    }
    if (trim($arg[1]) === '') {
        return substr($input, $start);
    }
    $end = intval($arg[1]);
    if ($end < 0) {
        $end += strlen($input);
    }
    return substr($input, $start, $end - $start);
}

print py_slice('abcdefg', '2') . "\n";
print py_slice('abcdefg', '2:4') . "\n";
print py_slice('abcdefg', '2:') . "\n";
print py_slice('abcdefg', ':4') . "\n";
print py_slice('abcdefg', ':-3') . "\n";
print py_slice('abcdefg', '-3:') . "\n";

?>
```

The $slice parameter can be a single character index, or a range separated by a colon. The start of the range is inclusive and the end is exclusive, which may be counterintuitive. (Eg, py_slice('abcdefg', '2:4') yields 'cd' not 'cde'). A negative range value means to count from the end of the string instead of the beginning. Both the start and end of the range may be omitted; the start defaults to 0 and the end defaults to the total length of the input.

The output from the examples:

```
c
cd
cdefg
abcd
abcd
efg
```

up
down
1

Quicker ¶

11 years ago

If you need to parse utf-8 strings char by char, try this one:

```
<?php
     $utf8marker=chr(128);
     $count=0;
     while(isset($string{$count})){
       if($string{$count}>=$utf8marker) {
         $parsechar=substr($string,$count,2);
         $count+=2;
       } else {
         $parsechar=$string{$count};
         $count++;
       }
       /* do what you like with parsechar ... , eg.:*/  echo $parsechar."<BR>\r\n";
     }
?>
```

- it works without mb_substr
- it is fast, because it grabs characters based on indexes when possible and avoids any count and split functions

up
down
1

php_net at thomas dot trella dot de ¶

17 years ago

I needed to cut a string after x chars at a html converted utf-8 text (for example Japanese text like 嬰謰弰脰欰罏).
The problem was, the different length of the signs, so I wrote the following function to handle that.
Perhaps it helps.

```
<?php

function html_cutstr ($str, $len)
{
    if (!preg_match('/\&#[0-9]*;.*/i', $str))
    {
        $rVal = strlen($str, $len);
        break;
    }

    $chars = 0;
    $start = 0;
    for($i=0; $i < strlen($str); $i++)
    {
        if ($chars >= $len)
        break;

        $str_tmp = substr($str, $start, $i-$start);
        if (preg_match('/\&#[0-9]*;.*/i', $str_tmp))
        {
            $chars++;
            $start = $i;
        }
    }
    $rVal = substr($str, 0, $start);
```

```
    if (strlen($str) > $start)
    $rVal .= " ...";
    return $rVal;
}
?>
```

up
down
1

leon weidauer ¶

11 years ago

When using a value of a wrong type as second parameter , substr() does not return FALSE but NULL although the docs say, it should return FALSE on error.

Prior to PHP 5.3, substr() tries to cast the second parameter to int and doesn't throw any errors. Since PHP 5.3 a warning is thrown.

up
down
1

Cristianlf ¶

12 years ago

I needed a function like lpad from oracle, or right from SQL
then I use this code :

```
<?php
function right($string,$chars)
{
    $vright = substr($string, strlen($string)-$chars,$chars);
    return $vright;

}

    echo right('0r0j4152',4);
?>
```

Result:
4152

This function is really simple, I just wanted to share, maybe helps someone out there.

regards,

up
down
1

post [at] jannik - zappe [dot] de ¶

14 years ago

Just a little function to cut a string by the wanted amount. Works in both directions.

```
<?php
function cutString($str, $amount = 1, $dir = "right")
{
  if(($n = strlen($str)) > 0)
  {
    if($dir == "right")
    {
      $start = 0;
      $end = $n-$amount;
    } elseif( $dir == "left") {
      $start = $amount;
```

```
      $end = $n;
    }

    return substr($str, $start, $end);
  } else return false;
}
?>

Enjoy ;)
```

up
down
1
vnonov at gmail dot com / Viktor Nonov ¶
12 years ago

```
<?php

//removes string from the end of other

function removeFromEnd($string, $stringToRemove) {
    $stringToRemoveLen = strlen($stringToRemove);
    $stringLen = strlen($string);

    $pos = $stringLen - $stringToRemoveLen;

    $out = substr($string, 0, $pos);

    return $out;
}

$string = 'picture.jpg.jpg';
$string = removeFromEnd($string, '.jpg');
?>
```

up
down
-2
robinhood70 at live dot ca ¶
1 year ago

```
Prior to PHP 8, specifying length with zero-length strings or non-string values as input can
produce potentially unexpected results.

<?php
foreach (['normal', '', true, false, NULL] as $value) {
    echo gettype(substr($value, 0, 10)) . ' ' . substr($value, 0, 10);
}

/*
string normal
boolean
string 1
boolean
boolean
*/
?>
```

up
down
1
webmaster at oehoeboeroe dot nl ¶
13 years ago

You might expect substr('123456', 6) to return an empty string. Instead it returns boolean FALSE.

This behavior should be mentioned in the Return Values section of the manual. Instead it is only mentioned in the Parameters section.

If you need an empty string instead of a boolean FALSE you should typecast the result to a string.

```
<?php
$a = substr('123456', 6);              // equivalent to $a = FALSE
$a = (string) substr('123456', 6);   // equivalent to $a = '';
?>
```

up
down
1

Bradley from California ¶

16 years ago

Add on to (a function originally written by) "Matias from Argentina": str_format_number function.

Just added handling of $String shorter then $Format by adding a side to start the fill and a string length to the while loop.

```
<?php
function str_format_number($String, $Format, $Start = 'left'){
    //If we want to fill from right to left incase string is shorter then format
    if ($Start == 'right') {
        $String = strrev($String);
        $Format = strrev($Format);
    }
    if($Format == '') return $String;
    if($String == '') return $String;
    $Result = '';
    $FormatPos = 0;
    $StringPos = 0;
    while ((strlen($Format) - 1) >= $FormatPos && strlen($String) > $StringPos) {
        //If its a number => stores it
        if (is_numeric(substr($Format, $FormatPos, 1))) {
            $Result .= substr($String, $StringPos, 1);
            $StringPos++;
            //If it is not a number => stores the caracter
        } else {
            $Result .= substr($Format, $FormatPos, 1);
        }
        //Next caracter at the mask.
        $FormatPos++;
    }
    if ($Start == 'right') $Result = strrev($Result);
    return $Result;
}
?>
```

up
down
1

frank at jkelloggs dot dk ¶

17 years ago

Regarding the utf8_substr function from lmak: The pattern '/./u' doesn't match newline characters. This means that the substring from 0 to the total length of the string will miss the number of characters in the end matching the number of newlines in the string. To fix this one can add the s modifier (PCRE_DOTALL) in the pattern:

```
<?php
function utf8_substr($str,$start)
{
   preg_match_all("/./su", $str, $ar);

   if(func_num_args() >= 3) {
       $end = func_get_arg(2);
       return join("",array_slice($ar[0],$start,$end));
   } else {
       return join("",array_slice($ar[0],$start));
   }
}
?>
```

up
down
0
link ¶
13 years ago

And as always there is bound to be a bug:

```
<?php
function strlen_entities($text)
{
    preg_match_all(
        '/((?:&(?:#[0-9]{2,}|[a-z]{2,});)|(?:[^&])|'.
        '(?:&(?!\w;)))s',$text,$textarray);
    return count($textarray[0]);
}
function substr_entities($text,$start,$limit=0)
{
    $return = '';
    preg_match_all(
        '/((?:&(?:#[0-9]{2,}|[a-z]{2,});)|(?:[^&])|'.
        '(?:&(?!\w;)))s',$text,$textarray);
    $textarray = $textarray[0];
    $numchars = count($textarray)-1;
    if ($start>=$numchars)
        return false;
    if ($start<0)
    {
        $start = ($numchars)+$start+1;
    }
    if ($start>=0)
    {
        if ($limit==0)
        {
            $end=$numchars;
        }
        elseif ($limit>0)
        {
            $end = $start+($limit-1);
        }
        else
        {
            $end = ($numchars)+$limit;
        }
```

```
        for ($i=$start;($i<=$end && isset($textarray[$i]));$i++)
        {
            $return .= $textarray[$i];
        }
        return $return;
    }
}
?>
```

up
down
0

ivanhoe011 at gmail dot com ¶
17 years ago

If you need just a single character from the string you don't need to use substr(), just use curly braces notation:

```
<?php
    // both lines will output the 3rd character
    echo substr($my_string, 2, 1);
    echo $my_string{2};
?>
```

curly braces syntax is faster and more readable IMHO..

up
down
0

rob NOSPAM at clancentric dot net ¶
17 years ago

I have developed a function with a similar outcome to jay's

Checks if the last character is or isnt a space. (does it the normal way if it is)
It explodes the string into an array of seperate works, the effect is... it chops off anything after and including the last space.

```
<?php
function limit_string($string, $charlimit)
{
    if(substr($string,$charlimit-1,1) != ' ')
    {
        $string = substr($string,'0',$charlimit);
        $array = explode(' ',$string);
        array_pop($array);
        $new_string = implode(' ',$array);

        return $new_string.'...';
    }
    else
    {
        return substr($string,'0',$charlimit-1).'...';
    }
}
?>
```

up
down
0

woutermb at gmail dot com ¶
17 years ago

Well this is a script I wrote, what it does is chop up long words with malicious meaning into several parts. This way, a chat in a table will not get stretched anymore.

```
<?php

function text($string,$limit=20,$chop=10){

$text = explode(" ",$string);
while(list($key, $value) = each($text)){
    $length = strlen($value);
    if($length >=20){
        for($i=0;$i<=$length;$i+=10){
            $new .= substr($value, $i, 10);
            $new .= " ";
        }
         $post .= $new;
    }
    elseif($length <=15){
        $post .= $value;
    }
    $post .= " ";
}
return($post);
}

// for example, this would return:
$output = text("Well this text doesn't get cut up, yet thisssssssssssssssssssssss one does.", 10, 5);

echo($output); // "Well this text doesn't get cup up, yet thiss sssss sssss sssss sssss sss one does."
?>
```

I hope it was useful.. :)

up
down
-1

kriskra at gmail dot com ¶

14 years ago

The javascript charAt equivalent in php of felipe has a little bug. It's necessary to compare the type (implicit) aswell or the function returns a wrong result:

```
<?php
function charAt($str,$pos) {
    return (substr($str,$pos,1) !== false) ? substr($str,$pos,1) : -1;
}
?>
```

up
down
-1

steve at unicycle dot co dot nz ¶

17 years ago

To quickly trim an optional trailing slash off the end of a path name:

```
if (substr( $path, -1 ) == '/') $path = substr( $path, 0, -1 );
```

up
down
-3

Nadeem ¶

8 years ago

Truncate a float number. Similar to the Excel trunc function.

```
<?php
function truncate_number($val,$decimals=2){

        $number=array();
        $number=explode(".",$val);
        $result=0;

        if (count($number)>1){

            $result = $number[0] . "." . substr($number[1],0,$decimals);

        } else {

            $result = $val;

        }

        unset($number);

        return $result;
}

echo truncate_number(99.123456,2); //result = 99.12
echo truncate_number(99.123456,5); //result = 99.12345
echo truncate_number(99.123456,1); //result = 99.1
?>
```

up
down
-3

m.m.j.kronenburg ¶

6 years ago

```
<?php

/**
  * Returns and extracts the portion of string specified by the
  * start and length parameters from the original string.
  *
  * This function is simulair to function substr() except that it
  * removes the substring from the orignal string
  * (passed by reference).
  *
  * @param string $string      The input string.
  * @param integer $start      The start position (see substr() for
  *                            explanation).
  * @param integer $length     The length (see substr()
  *                            for explanation).
  * @return mixed              The substring or FALSE (see substr()
  *                            for explanation).
  **/
function substrex(&$string, $start, $length = PHP_INT_MAX)
{
  if($start > strlen($string)) { return false; }
  if(empty($length))           { return ''; }
  if($start < 0) { $start = max(0, $start + strlen($string)); }
  $end = ($length < 0) ?
```

```
    strlen($string) + $length :
    min(strlen($string), $start + $length);
  if($end < $start) { return false; }
  $length = $end - $start;
  $substr = substr($string, $start, $length);
  $string = substr($string, 0, $start).substr($string, $end);
  return $substr;
}

?>
```

up
down
-7
man13or at hotmail dot fr ¶
3 years ago

```
Shortcuts :

Getting the first character of a string
substr($string, 1)

Getting the last character of a string
substr($string, -1)

Remove the first character of a string
substr($string,1)

Remove the last character of a string
substr($string, 0, -1)
```

+ add a note

- Funciones de strings
  - addcslashes
  - addslashes
  - bin2hex
  - chop
  - chr
  - chunk_split
  - convert_uudecode
  - convert_uuencode
  - count_chars
  - crc32
  - crypt
  - echo
  - explode
  - fprintf
  - get_html_translation_table
  - hebrev
  - hex2bin
  - html_entity_decode
  - htmlentities
  - htmlspecialchars_decode
  - htmlspecialchars
  - implode
  - join
  - lcfirst
  - levenshtein
  - localeconv
  - ltrim

- md5_file
- md5
- metaphone
- money_format
- nl_langinfo
- nl2br
- number_format
- ord
- parse_str
- print
- printf
- quoted_printable_decode
- quoted_printable_encode
- quotemeta
- rtrim
- setlocale
- sha1_file
- sha1
- similar_text
- soundex
- sprintf
- sscanf
- str_contains
- str_ends_with
- str_getcsv
- str_ireplace
- str_pad
- str_repeat
- str_replace
- str_rot13
- str_shuffle
- str_split
- str_starts_with
- str_word_count
- strcasecmp
- strchr
- strcmp
- strcoll
- strcspn
- strip_tags
- stripcslashes
- stripos
- stripslashes
- stristr
- strlen
- strnatcasecmp
- strnatcmp
- strncasecmp
- strncmp
- strpbrk
- strpos
- strrchr
- strrev
- strripos
- strrpos
- strspn
- strstr
- strtok

- strtolower
- strtoupper
- strtr
- substr_compare
- substr_count
- substr_replace
- substr
- trim
- ucfirst
- ucwords
- utf8_decode
- utf8_encode
- vfprintf
- vprintf
- vsprintf
- wordwrap

- Deprecated
  - convert_cyr_string
  - hebrevc