Focus search box

chunk_split »
« chop

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

Change language: Spanish

Submit a Pull Request Report a Bug

# chr

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

chr — Devuelve un caracter específico

## Descripción ¶

**chr**( int `$ascii` ): string

Devuelve una cadena de un caracter que contiene el carácter especificado por `ascii`

Esta función complementa a ord().

## Parámetros ¶

`ascii`

El código ASCII.

## Valores devueltos ¶

Devuelve el carácter especificado.

## Ejemplos ¶

### Ejemplo #1 chr() Ejemplo

```
<?php
$str = "La cadena termina en un escape: ";
$str .= chr(27); /* añade un carácter de escape al final de  $str */

/* A menudo esto es más útil */

$str = sprintf("TLa cadena termina en un escape: %c", 27);
?>
```

## Ver también ¶

- sprintf() - Devuelve un string formateado Devuelve una cadena con formato con un formato de `%c`
- ord()

- Una » ASCII-tabla

+ add a note

## User Contributed Notes 23 notes

up
down
18
***voromax*** ¶
**13 years ago**
Another quick and short function to get unicode char by its code.

```
<?php
/**
* Return unicode char by its code
*
* @param int $u
* @return char
*/
function unichr($u) {
    return mb_convert_encoding('&#' . intval($u) . ';', 'UTF-8', 'HTML-ENTITIES');
}
?>
```

up
down
17
***perrodin at laposte dot net*** ¶
**18 years ago**
Note that if the number is higher than 256, it will return the number mod 256.
For example :
chr(321)=A because A=65(256)

up
down
6
***grey - greywyvern - com*** ¶
**17 years ago**
I spent hours looking for a function which would take a numeric HTML entity value and output the appropriate UTF-8 bytes.  I found this at another site and only had to modify it slightly; so I don't take credit for this.

```
<?php function unichr($dec) {
  if ($dec < 128) {
    $utf = chr($dec);
  } else if ($dec < 2048) {
    $utf = chr(192 + (($dec - ($dec % 64)) / 64));
    $utf .= chr(128 + ($dec % 64));
  } else {
    $utf = chr(224 + (($dec - ($dec % 4096)) / 4096));
    $utf .= chr(128 + ((($dec % 4096) - ($dec % 64)) / 64));
    $utf .= chr(128 + ($dec % 64));
  }
  return $utf;
} ?>
```

So for example:

```
<?php

  $str = "Chinese: &#20013;&#25991;";
  $str = preg_replace("/&#(\d{2,5});/e", "unichr($1);", $str);

?>
```

up
down
3

***joeldegan AT yahoo.com ¶***

**19 years ago**

Want terminal colors in command line php scripts?

This should take care of that.

```
<?

$_colors = array(
        'LIGHT_RED'      => "[1;31m",
        'LIGHT_GREEN'     => "[1;32m",
        'YELLOW'     => "[1;33m",
        'LIGHT_BLUE'     => "[1;34m",
        'MAGENTA'     => "[1;35m",
        'LIGHT_CYAN'     => "[1;36m",
        'WHITE'     => "[1;37m",
        'NORMAL'     => "[0m",
        'BLACK'     => "[0;30m",
        'RED'          => "[0;31m",
        'GREEN'     => "[0;32m",
        'BROWN'     => "[0;33m",
        'BLUE'         => "[0;34m",
        'CYAN'         => "[0;36m",
        'BOLD'         => "[1m",
        'UNDERSCORE'     => "[4m",
        'REVERSE'     => "[7m",

);

function termcolored($text, $color="NORMAL", $back=1){
    global $_colors;
    $out = $_colors["$color"];
    if($out == ""){ $out = "[0m"; }
    if($back){
        return chr(27)."$out$text".chr(27).chr(27)."[0m".chr(27);
    }else{
        echo chr(27)."$out$text".chr(27).chr(27)."[0m".chr(27);
    }//fi
}// end function

echo termcolored("test\n", "BLUE");
?>
```

up
down
2

***Noname ¶***

**9 months ago**

```
<?php

for ($i = 128; $i <= 191; $i++) {
```

```
    $str = chr(240) . chr(159) . chr(144) . chr($i);
    echo $str;
}

?>
```

up
down
1

***mailderemi at gmail dot com ¶***

**11 years ago**

Here is a sample of encoding and decoding using "chr" and "ord".

```
<?php
    function Encode($txtData,$Level){
        for ($j = 0;$j<$Level;$j++){
            $tmpStr = '';
            for ($i = 0;$i<strlen($txtData);$i++)
                $tmpStr .= ord(substr(strtoupper($txtData), $i, 1));
            $txtData = $tmpStr;
        }
        return (strlen($Level)).$Level.$txtData;
    }

    function Decode($txtData){
        $intLevel = substr($txtData, 1, substr($txtData, 0, 1));
        $startStr = substr($txtData, substr($txtData, 0, 1)+1, strlen($txtData));
        for ($j = 0;$j<$intLevel;$j++){
            for ($i = 0;$i<strlen($startStr);$i+=2)
                $tmpStr .= chr(intval(substr($startStr, $i, 2)));
            $startStr = $tmpStr;

            $tmpStr = "";
        }
        return $startStr;
    }

echo Encode('123',4).'<br>';
echo Decode(Encode('123',5));
?>
```

up
down
1

***scott at quinlan dot co dot nz ¶***

**14 years ago**

Secure password generator with a variable maximum amount of symbols.

```
<?php

function passwdGen($minLength = 8, $maxLength = 12, $maxSymbols = 2)
{
    $symbolCount = 0;

    srand((double)microtime() * 1000003);

    for ($i = 0; $i < rand($minLength, $maxLength); $i++)
    {
        do
        {
            $char = rand(33, 126);
```

```
            $symbolCount += $isSymbol = (!in_array($char, range(48, 57)) && !in_array($char,
range(65, 90)) && !in_array($char, range(97, 122)));

            if ($symbolCount <= $maxSymbols || !$isSymbol)
            {
                break;
            }
        }
        while (true);

        $passwd = sprintf('%s%c', isset($passwd) ? $passwd : NULL, $char);
    }

    return $passwd;
}

?>
```

up
down
1
***Josh B.*** ¶
**14 years ago**

In addition to replacing Microsoft Windows smart quotes, as sgaston demonstrated on 2006-02-13, I replace all other Microsoft Windows characters using suggestions[1] published by character code specialist[2] Jukka Korpela.

```
<?php
$str = str_replace(chr(130), ',', $str);    // baseline single quote
$str = str_replace(chr(131), 'NLG', $str);  // florin
$str = str_replace(chr(132), '"', $str);    // baseline double quote
$str = str_replace(chr(133), '...', $str);  // ellipsis
$str = str_replace(chr(134), '**', $str);   // dagger (a second footnote)
$str = str_replace(chr(135), '***', $str);  // double dagger (a third footnote)
$str = str_replace(chr(136), '^', $str);    // circumflex accent
$str = str_replace(chr(137), 'o/oo', $str); // permile
$str = str_replace(chr(138), 'Sh', $str);   // S Hacek
$str = str_replace(chr(139), '<', $str);    // left single guillemet
$str = str_replace(chr(140), 'OE', $str);   // OE ligature
$str = str_replace(chr(145), "'", $str);    // left single quote
$str = str_replace(chr(146), "'", $str);    // right single quote
$str = str_replace(chr(147), '"', $str);    // left double quote
$str = str_replace(chr(148), '"', $str);    // right double quote
$str = str_replace(chr(149), '-', $str);    // bullet
$str = str_replace(chr(150), '-', $str);    // endash
$str = str_replace(chr(151), '--', $str);   // emdash
$str = str_replace(chr(152), '~', $str);    // tilde accent
$str = str_replace(chr(153), '(TM)', $str); // trademark ligature
$str = str_replace(chr(154), 'sh', $str);   // s Hacek
$str = str_replace(chr(155), '>', $str);    // right single guillemet
$str = str_replace(chr(156), 'oe', $str);   // oe ligature
$str = str_replace(chr(159), 'Y', $str);    // Y Dieresis
?>
```

[1] On the use of some MS Windows characters in HTML
http://www.cs.tut.fi/~jkorpela/www/windows-chars.html

[2] Unicode Explained by Jukka Korpela
http://www.amazon.com/dp/059610121X/

up
down
1

***happyevil(at)1218.org ¶***
**21 years ago**

Here is a function that's help me find what chr(number) outputs what character quicker than typing out 256 echo tags.

```
<?php
function listChr(){
  for ($i = 0; $i < 256; ++$i) {
  static $genNum;
  $genNum++;
  echo "chr($genNum) will output '";
  echo (chr($genNum));
  echo "'< br>\n";
  }
}
listChr();
?>
```

Another helpful chr is #9, being a tab.  Quite using when making error logs.

```
$tab = (chr(9));
echo "<pre>error{$tab}date{$tab}time</pre>";
```

-- HappyEvil

up
down
0

***synnus at gmail dot com ¶***
**7 years ago**

```
// rivencodec 1.0
// encode riverse ascii 1 simple function can encode/decode
// can use it for secure source with speed encode text

<?php

    function rivencodec($ch,$a=0) {
        while((@$b = $ch[$a++])) { $ch[$a-1] = chr(255-ord($b)); }
        return $ch;
    }


$zz = rivencodec("abcdefghijklmn");
echo 'encode: ',$zz,'<br/>',PHP_EOL;

$yy = rivencodec($zz);
echo 'decode: ',$yy,'<br/>',PHP_EOL;

?>
```

up
down
-1

***darek at module17 dot com ¶***
**9 years ago**

Simple password generation function using sprintf and the %c type specifier; which is the same as chr().

```
function genPass($len = 8) {
    for ($i=0;$i<=$len;$i++) {
        $passwd = sprintf('%s%c', isset($passwd) ? $passwd : NULL, rand(48, 122));
    }
    return $passwd;
}
```

up
down
-1

***gjarrige at six-axe dot fr ¶***
**10 years ago**

to remove the ASCII control characters (except "line feed" and "tab") :

```
$tab_chr = array() ;
for($control = 0; $control < 32; $control++) {
    if ($control != 9 && $control != 10) {
        $tab_chr[]= chr($control) ;
    }
}
$tab_chr[]= chr(127) ;
$string = str_replace($tab_chr, '', $string);
```

up
down
-1

***vitkorob ¶***
**6 years ago**

Another quick function to get unicode char by its code.

```
<?php

function unichr($dec)
{
  if ($dec < 0x80)
  {
    $utf = chr($dec);
  }
  else if ($dec < 0x0800)
  {
    $utf = chr(0xC0 + ($dec >> 6));
    $utf .= chr(0x80 + ($dec & 0x3f));
  }
  else if ($dec < 0x010000)
  {
    $utf = chr(0xE0 + ($dec >> 12));
    $utf .= chr(0x80 + (($dec >> 6) & 0x3f));
    $utf .= chr(0x80 + ($dec & 0x3f));
  }
  else if ($dec < 0x200000)
  {
    $utf = chr(0xF0 + ($dec >> 18));
    $utf .= chr(0x80 + (($dec >> 12) & 0x3f));
    $utf .= chr(0x80 + (($dec >> 6) & 0x3f));
    $utf .= chr(0x80 + ($dec & 0x3f));
  }
  else
```

```
  {
    die("UTF-8 character size is more than 4 bytes");
  }

  return $utf;
}

echo unichr(0x263A);

?>
```

up
down
-1
***Kristin ¶***
**19 years ago**

Note that chr(10) is a 'line feed' and chr(13) is a 'carriage return' and they are not the same thing! I found this out while attempting to parse text from forms and text files for inclusion as HTML by replacing all the carriage returns with <BR>'s only to find after many head-scratchings that I should have been looking for line feeds. If anyone can shed some light on what the difference is, please do.

If you're planning on saving text from a form into a database for later display, you'll need to apply the following function so that it gets saved with the proper HTML tags.

```
<?php
$text = str_replace ( chr(10), "<BR>", $text );
?>
```

When you want to plug it back into that form for editing you need to convert it back.

```
<?php
$text = str_replace ( "<BR>", chr(10), $text)
?>
```

Hope this saves somebody some trouble. :)

up
down
-1
***v14t at gmx dot com ¶***
**8 years ago**

argument is automatically converted to integer, so chr('65') and chr(65) would both output the letter A

up
down
-1
***tenyou at gmail dot com ¶***
**18 years ago**

When having to deal with parsing an IIS4 or IIS5 metabase dump I wrote a simple function for converting those MS hexidecimal values into their ascii counter parts. Hopefully someone will find use for it.

```
<?php
function hex_decode($string)  {
        for ($i=0; $i < strlen($string); $i)  {
        $decoded .= chr(hexdec(substr($string,$i,2)));
        $i = (float)($i)+2;
        }
return $decoded;
```

```
}
?>
```

up
down
-1
***ddawsonNOSPAM at execpc dot com ¶***
**22 years ago**

```
[Editor's note:

%c is defined as: "Print the character belonging to the ascii code given"

chr() just gives a string, so you need to use %s, even if the string consists of only one
character. This is consistent with other languages.
--Jeroen@php.net]


Learn from my mistake:
Do not expect this to work!

<?php
$c_question = chr(63);
$v_out = sprintf("<%cphp\n", $c_question);
//... more stuff being sprintf'd into v_out here ...
$v_out = sprintf("%s%c>\n", $v_out, $c_question);
$v_fp = fopen("foofile", "w");
if ($v_fp)
{
     fwrite($v_fp, $v_out, strlen($v_out));
     fclose($v_fp);
}
?>

When I did this, foofile contained <NUL NUL NUL NUL NUL>.
I spun my wheels quite awhile looking at fputs, fwrite to verify I was calling those functions
correctly.
My mistake was using $c_question = chr(63) instead of
$c_question = 63 (correct).  Then everything worked fine.
```

up
down
-1
***ibaid at mail dot ru ¶***
**4 years ago**

```
string mb_chr ( int $cp [, string $encoding ] )
Parameter List:
cp - character code (in decimal notation)
encoding - encoding (UTF-8, ASCII and so on)

We get the letter 'Ж'  from the encoding UTF-8:

$sim = mb_chr(0x0416, 'UTF-8');
echo $sim; // Ж

Get the character '}' from the encoding ASCII:

$sim = mb_chr(125, 'ASCII');
echo $sim ; // }
```

up
down

-2

***Ray.Paseur sometimes uses Gmail ¶***

**3 years ago**

I needed to generate an invalid UTF-8 character for testing with JSON.  This did the trick:

```
<?php
echo 'Bogus UTF-8 character at end' . chr(0xC6) ;
```

up

down

-4

***darkodemon at gmail dot com ¶***

**15 years ago**

chr() with unicode support

```
<?php

function uchr ($codes) {
    if (is_scalar($codes)) $codes= func_get_args();
    $str= '';
    foreach ($codes as $code) $str.= html_entity_decode('&#'.$code.';',ENT_NOQUOTES,'UTF-8');
    return $str;
}

echo uchr(23383); echo '<br/>';
echo uchr(23383,215,23383); echo '<br/>';
echo uchr(array(23383,215,23383,215,23383)); echo '<br/>';

?>
```

up

down

-2

***sinfocol at sinfocol dot org ¶***

**13 years ago**

The function chr() also accepts negative numbers as an ascii code, so chr(-number) is equal to chr((number%256)+256).
And for ascii code higher than 255 is chr(number%256)

We can test with a little script

```
<?php
    for($i=-300; $i<300; $i++){
        echo "Ascii $i\t" . ord(chr($i)) . "\n";
    }
?>
```

up

down

-5

***lingtalfi - at - somewhere ¶***

**7 years ago**

It seems that php uses the table from here: http://ascii-code.com/
(and not from here: http://www.asciitable.com/ as suggested in the documentation) for codes from 128 to 255.

```
<?php
for ($i = 32; $i <= 255; $i++) {
    echo chr($i);
}
?>
```

up

down
-3
***gfour ¶***
**3 years ago**
I was looking for a simple method to construct excel like column identifiers e.g: A B .... AA AB AC etc, using chr() and modulo, but there is magic...
https://www.php.net/manual/en/language.operators.increment.php

So, this also works

```
<?php
$p = chr(65);      // or simply $p = 'A';
for ($i = 1; $i < 53; $i++){
    echo $p++ . " - ";
    if ($i % 10 == 0) echo '</br>';
}
?>
```

Gives

```
A - B - C - D - E - F - G - H - I - J -
K - L - M - N - O - P - Q - R - S - T -
U - V - W - X - Y - Z - AA - AB - AC - AD -
AE - AF - AG - AH - AI - AJ - AK - AL - AM - AN -
AO - AP - AQ - AR - AS - AT - AU - AV - AW - AX -
AY - AZ -
```

+ add a note

- 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


- My PHP.net
- Contact
- Other PHP.net sites
- Privacy policy
- View Source