chop »
« addslashes

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

Change language: Spanish

Submit a Pull Request Report a Bug

bin2hex

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

bin2hex — Convierte datos binarios en su representación hexadecimal

Descripción ¶

bin2hex(string `$str`): string

Devuelve una cadena ASCII que contiene la representación hexadecimal de `str`. La conversión se realiza byte a byte, con los 4 bits superiores primero.

Parámetros ¶

`str`

Un string.

Valores devueltos ¶

Devuelve la representación hexadecimal de la cadena dada.

Ver también ¶

- hex2bin() - Decodifica una cadena binaria codificada hexadecimalmente
- pack() - Empaqueta información a una cadena binaria

+ add a note

User Contributed Notes 6 notes

up
down
53
tehjosh at gamingg dot net ¶
15 years ago
```
This function is for converting binary data into a hexadecimal string representation.  This
function is not for converting strings representing binary digits into hexadecimal.  If you want
that functionality, you can simply do this:

<?php
$binary = "11111001";
```

```
$hex = dechex(bindec($binary));
echo $hex;
?>
```

This would output "f9". Just remember that there is a very big difference between binary data and a string representation of binary.

up
down
11

j_lozinskit at yahoo dot co dot uk ¶

17 years ago

A good option for creating strings with binary data for saving (for example saving an sql statement to a file) into text files or php code is to do the following:

```
<?php
$field=bin2hex($field);
$field=chunk_split($field,2,"\\x");
$field= "\\x" . substr($field,0,-2);
?>
```

this will convert your field (binary or not) into hex and then convert the hex into a string which may be placed in a php file:

```
FFFFFFFF -> \xFF\xFF\xFF\xFF
```

up
down
2

busuioc dot alexandru at gmail dot com ¶

3 years ago

Convenient way of generating API keys

```
<?php
$apikey = bin2hex(random_bytes(32)); // generates 64 characters long string /^[0-9a-f]{64}$/
?>
```

up
down
2

tightcode at hotmail dot com ¶

21 years ago

I was just browsing the above and with a little modification,
came up with the following which I believe to be more flexible:

```
<?php
function bin2hex($data) {
    $corrected = ereg_replace("[^0-9a-fA-F]","",$data);
    return pack("H".strlen($corrected),$corrected);
}
?>
```

This will make sure that whatever you pass, even if it is padded
at the extremeties or between pairs, should return the desired data.

up
down
3

pedram at redhive dot com ¶

21 years ago

In an attempt to dodge spam bots I've seen people (including myself) hex encode their email addresses in "mailto" tags. This is the small chunk of code I wrote to automate the process:

```
<?php
function hex_encode ($email_address)    {
        $encoded = bin2hex("$email_address");
        $encoded = chunk_split($encoded, 2, '%');
        $encoded = '%' . substr($encoded, 0, strlen($encoded) - 1);
        return $encoded;
}
?>

so for example:
<a href="mailto:&lt;?=hex_encode("pedram@redhive.com")?>">email me</a>

would produce the following address:
%70%65%64%72%61%6d%40%72%65%64%68%69%76%65%2e%63%6f%6d

-pedram
```

up
down
-1
subdivizion at gmail dot com ¶
13 years ago

Here's a function to check if a string contains any 7-bit GSM characters.

It might come useful for people working on SMS platforms.

```
<?php

function check_gsm($str)
{
    $arr = array(
"0x00", "0x01", "0x02", "0x03", "0x04", "0x05","0x06","0x07","0x08","0x09",
"0x0A","0x0B","0x0C","0x0D","0x0E","0x0F","0x10","0x11","0x12","0x13",
"0x14","0x15","0x16","0x17","0x18","0x19","0x1A","0x1B","0x1B0A",
"0x1B14","0x1B28","0x1B29","0x1B2F","0x1B3C","0x1B3D","0x1B3E",
"0x1B40","0x1B65","0x1C","0x1D","0x1E","0x1F","0x20","0x21","0x22",
"0x23","0x24","0x25","0x26","0x27","0x28","0x29","0x2A","0x2B","0x2C",
"0x2D","0x2E","0x2F","0x30","0x31","0x32","0x33","0x34","0x35","0x36",
"0x37","0x38","0x39","0x3A","0x3B","0x3C","0x3D","0x3E","0x3F","0x40",
"0x41","0x42","0x43","0x44","0x45","0x46","0x47","0x48","0x49","0x4A",
"0x4B","0x4C","0x4D","0x4E","0x4F","0x50","0x51","0x52","0x53","0x54",
"0x55","0x56","0x57","0x58","0x59","0x5A","0x5B","0x5C","0x5D","0x5E",
"0x5F","0x60","0x61","0x62","0x63","0x64","0x65","0x66","0x67","0x68",
"0x69","0x6A","0x6B","0x6C","0x6D","0x6E","0x6F","0x70","0x71","0x72",
"0x73","0x74","0x75","0x76","0x77","0x78","0x79","0x7A","0x7B","0x7C",
"0x7D","0x7E","0x7F");

    $strl = strlen($str);
    for ($i = 0;$i < $strl; $i++)
    {
        $char = '0x' . bin2hex(substr($str,$i,1));
        $pos = in_array($char,$arr);
        if ($pos == 1)
        {
            $j++;
        }
    }
```

```
    if ($j < $strl)
    {
        return false;
    }
    else
    {
        return true;
    }
}
?>
```

+ 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