hebrev »
« fprintf

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

Change language: Spanish

Submit a Pull Request Report a Bug

get_html_translation_table

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

get_html_translation_table — Devuelve la tabla de traducción utilizada por htmlspecialchars() y htmlentities()

Descripción ¶

get_html_translation_table(int `$table` = HTML_SPECIALCHARS, int `$flags` = ENT_COMPAT | ENT_HTML401, string `$encoding` = "UTF-8"): array

get_html_translation_table() devolverá la tabla de traducción que es utilizada internamente para htmlspecialchars() y htmlentities().

> **Nota**:
>
> Los caracteres especiales se pueden codificar de varias maneras. Por ejemplo, " puede ser codificado como `&quot;`, `&#34;` o `&#x22`. **get_html_translation_table()** devuelve sólo la forma utilizada por htmlspecialchars() y htmlentities().

Parámetros ¶

`table`

Qué tabla devolver. Puede ser **`HTML_ENTITIES`** o **`HTML_SPECIALCHARS`**).

`flags`

Una máscara de bits de uno o más de los siguientes indicadores especificando qué comillas contendrá la tabla, así como para qué tipo de documento será la tabla. El valor por defecto es `ENT_COMPAT | ENT_HTML401`.

Constantes disponibles para `flags`

Nombre de la constante	Descripción
`ENT_COMPAT`	La tabla contendrá entidades para comillas dobles, pero no para comillas simples.
`ENT_QUOTES`	La tabla contendrá entidades para comillas dobles y simples.
`ENT_NOQUOTES`	La tabla no contendrá entidades para comillas simples ni para comillas dobles.
`ENT_HTML401`	Tabla para HTML 4.01.
`ENT_XML1`	Tabla para XML 1.

Nombre de la constante	Descripción
`ENT_XHTML`	Tabla para XHTML.
`ENT_HTML5`	Tabla para HTML 5.

`encoding`

La codificación a usar. Si se omite, el valor por defecto para este argumento es ISO-8859-1 en versiones de PHP anteriores a 5.4.0, y UTF-8 a partir de PHP 5.4.0 en adelante.

Están soportados los siguientes juegos de caracteres:

Juegos de caracteres soportados

Juego de caracteres	Alias	Descripción
ISO-8859-1	ISO8859-1	Europeo occidental, Latin-1.
ISO-8859-5	ISO8859-5	Juego de caracteres cirílicos poco usado (Latin/Cyrillic).
ISO-8859-15	ISO8859-15	Europeo occidental, Latin-9. Añade el signo de euro, y letras del francés y finlandés ausentes en Latin-1 (ISO-8859-1).
UTF-8		Unicode de 8 bit multibyte compatible con ASCII.
cp866	ibm866, 866	Juego de caracteres cirílico específico de DOS.
cp1251	Windows-1251, win-1251, 1251	Juego de caracteres cirílico específico de Windows.
cp1252	Windows-1252, 1252	Juego de caracteres específico de Windows para Europa occidental.
KOI8-R	koi8-ru, koi8r	Ruso.
BIG5	950	Chino tradicional, usado principalmente en Taiwán.
GB2312	936	Chino simplificado, juego de caracteres estándar nacional.
BIG5-HKSCS		Big5 con extensiones de Hong Kong, chino tradicional.
Shift_JIS	SJIS, SJIS-win, cp932, 932	Japonés
EUC-JP	EUCJP, eucJP-win	Japonés
MacRoman		Juego de caracteres que fue utilizado por Mac OS.
''		Un string vacío activa la detección desde la codificación del script (Zend multibyte), default_charset y la actual configuración regional (véase nl_langinfo() y setlocale()), en este orden. No se recomienda.

Nota: No se reconoce cualquier otro juego de caracteres. Será utilizada en su lugar la codificación por defecto y se emitirá una advertencia.

Valores devueltos ¶

Devuelve la tabla de traducción como un array, con los caracteres originales como claves y las entidades como valores.

Historial de cambios ¶

Versión	Descripción
5.4.0	El valor por defecto para el parámetro `encoding` se cambió a UTF-8.
5.4.0	Se añadieron las constantes **`ENT_HTML401`**, **`ENT_XML1`**, **`ENT_XHTML`** y **`ENT_HTML5`**.
5.3.4	Se añadió el parámetro `encoding`.

Ejemplos ¶

Ejemplo #1 Ejemplo de tabla de traducción

```
<?php
var_dump(get_html_translation_table(HTML_ENTITIES, ENT_QUOTES | ENT_HTML5));
?>
```

El resultado del ejemplo sería algo similar a:

```
array(1510) {
  ["
"]=>
  string(9) "&NewLine;"
  ["!"]=>
  string(6) "&excl;"
  [""]=>
  string(6) "&quot;"
  ["#"]=>
  string(5) "&num;"
  ["$"]=>
  string(8) "&dollar;"
  ["%"]=>
  string(8) "&percnt;"
  ["&"]=>
  string(5) "&amp;"
  ["'"]=>
  string(6) "&apos;"
  // ...
}
```

Ver también ¶

- htmlspecialchars() - Convierte caracteres especiales en entidades HTML
- htmlentities() - Convierte todos los caracteres aplicables a entidades HTML
- html_entity_decode() - Convierte todas las entidades HTML a sus caracteres correspondientes

+ add a note

User Contributed Notes 13 notes

up
down
13
michael dot genesis at gmail dot com ¶
10 years ago

The fact that MS-word and some other sources use CP-1252, and that it is so close to Latin1 ('ISO-8859-1') causes a lot of confusion. What confused me the most was finding that mySQL uses CP-1252 by default.

You may run into trouble if you find yourself tempted to do something like this:

```
<?php
    $trans[chr(149)] = '&bull;';    // Bullet
    $trans[chr(150)] = '&ndash;';    // En Dash
```

```
    $trans[chr(151)] = '&mdash;';    // Em Dash
    $trans[chr(152)] = '&tilde;';    // Small Tilde
    $trans[chr(153)] = '&trade;';    // Trade Mark Sign
?>
```

Don't do it. DON'T DO IT!

You can use:

```
<?php
    $translationTable = get_html_translation_table(HTML_ENTITIES, ENT_NOQUOTES, 'WINDOWS-1252');
?>
```

or just convert directly:

```
<?php
    $output = htmlentities($input, ENT_NOQUOTES, 'WINDOWS-1252');
?>
```

But your web page is probably encoded UTF-8, and you probably don't really want CP-1252 text flying around, so fix the character encoding first:

```
<?php
    $output = mb_convert_encoding($input, 'UTF-8', 'WINDOWS-1252');
    $ouput = htmlentities($output);
?>
```

up
down
10

kevin at cwsmailbox dot xom ¶

12 years ago

Be careful using get_html_translation_table() in a loop, as it's very slow.

up
down
1

Kenneth Kin Lum ¶

14 years ago

to display the mapping on a webpage no matter what the server encoding is, this can be used

```
  echo "<pre>\n";
  echo htmlentities(print_r((get_html_translation_table(HTML_SPECIALCHARS)), true));
  echo htmlentities(print_r((get_html_translation_table(HTML_ENTITIES)), true));
```

since get_html_translation_table() actually gives the special chars in iso-8859-1 (Latin-1) encoding, so to see the tables correctly using

```
  print_r(get_html_translation_table(HTML_ENTITIES));
```

your server needs to give a HTTP header as iso-8859-1, unless you use header() or manually set the browser's encoding setting to iso-8859-1. And you need to view the source of the page to see the mapping. (except English version of IE 7 outputs the page source as iso-8859-1 anyway).

up
down
0

iain (duh) workingsoftware.com.au ¶

15 years ago

I wrote a quick little function for converting something like '·' into '·':

```
$to_convert = '&middot;';
$table = get_html_translation_table(HTML_ENTITIES);
$equiv = '&#'.ord(array_search($to_convert,$table)).';';
```

up
down
0
dirk at hartmann dot net ¶
21 years ago
get_html_translation_table
It works only with the first 256 Codepositions.
For Higher Positions, for Example ф
(a kyrillic Letter) it shows the same.
up
down
-1
Patrick nospam at nospam mesopia dot com ¶
17 years ago
Not sure what's going on here but I've run into a problem that others might face as well...

```
<?php

$translations = array_flip(get_html_translation_table(HTML_ENTITIES,ENT_QUOTES));

?>
```

returns the single quote ' as being equal to ' while

```
<?php

$translatedString = htmlentities($string,ENT_QUOTES);

?>
```

returns it as being equal to '

I've had to do a specific string replacement for the time being... Not sure if it's an issue with the function or the array manipulation.

-Pat
up
down
-2
Jérôme Jaglale ¶
15 years ago
htmlentities includes htmlspecialchars, so here's how to convert an UTF-8 string :
htmlentities($string, ENT_QUOTES, 'UTF-8');
up
down
-3
Maurizio Siliani at trident dot it ¶
15 years ago
If you have troubles (like me) getting data from ISO-8859-1 encoded forms where user copy and paste from word, this routine could be useful.
It adds to the standard get_html_translation_table the codes of the characters usually M$ Word replacs into typed text.
Otherwise those characters would never be displayed correctly in html output.

```
function get_html_translation_table_CP1252() {
    $trans = get_html_translation_table(HTML_ENTITIES);
    $trans[chr(130)] = '&sbquo;';    // Single Low-9 Quotation Mark
    $trans[chr(131)] = '&fnof;';    // Latin Small Letter F With Hook
    $trans[chr(132)] = '&bdquo;';    // Double Low-9 Quotation Mark
```

```
    $trans[chr(133)] = '&hellip;';    // Horizontal Ellipsis
    $trans[chr(134)] = '&dagger;';    // Dagger
    $trans[chr(135)] = '&Dagger;';    // Double Dagger
    $trans[chr(136)] = '&circ;';    // Modifier Letter Circumflex Accent
    $trans[chr(137)] = '&permil;';    // Per Mille Sign
    $trans[chr(138)] = '&Scaron;';    // Latin Capital Letter S With Caron
    $trans[chr(139)] = '&lsaquo;';    // Single Left-Pointing Angle Quotation Mark
    $trans[chr(140)] = '&OElig;    ';    // Latin Capital Ligature OE
    $trans[chr(145)] = '&lsquo;';    // Left Single Quotation Mark
    $trans[chr(146)] = '&rsquo;';    // Right Single Quotation Mark
    $trans[chr(147)] = '&ldquo;';    // Left Double Quotation Mark
    $trans[chr(148)] = '&rdquo;';    // Right Double Quotation Mark
    $trans[chr(149)] = '&bull;';    // Bullet
    $trans[chr(150)] = '&ndash;';    // En Dash
    $trans[chr(151)] = '&mdash;';    // Em Dash
    $trans[chr(152)] = '&tilde;';    // Small Tilde
    $trans[chr(153)] = '&trade;';    // Trade Mark Sign
    $trans[chr(154)] = '&scaron;';    // Latin Small Letter S With Caron
    $trans[chr(155)] = '&rsaquo;';    // Single Right-Pointing Angle Quotation Mark
    $trans[chr(156)] = '&oelig;';    // Latin Small Ligature OE
    $trans[chr(159)] = '&Yuml;';    // Latin Capital Letter Y With Diaeresis
    ksort($trans);
    return $trans;
}
```

up
down
-4
Alex Minkoff ¶
17 years ago

If you want to display special HTML entities in a web browser, you can use the following code:

```
<?
$entities = get_html_translation_table(HTML_ENTITIES);
foreach ($entities as $entity) {
    $new_entities[$entity] = htmlspecialchars($entity);
}
echo "<pre>";
print_r($new_entities);
echo "</pre>";
?>
```

If you don't, the key name of each element will appear to be the same as the element content itself, making it look mighty stupid. ;)

up
down
-4
kumar at chicagomodular.com ¶
20 years ago

without heavy scientific analysis, this seems to work as a quick fix to making text originating from a Microsoft Word document display as HTML:

```
<?php
function DoHTMLEntities ($string)
    {
        $trans_tbl = get_html_translation_table (HTML_ENTITIES);

        // MS Word strangeness..
        // smart single/ double quotes:
```

```
        $trans_tbl[chr(145)] = '\'';
        $trans_tbl[chr(146)] = '\'';
        $trans_tbl[chr(147)] = '&quot;';
        $trans_tbl[chr(148)] = '&quot;';

                // Acute 'e'
        $trans_tbl[chr(142)] = '&eacute;';

        return strtr ($string, $trans_tbl);
    }
?>
```

up
down
-6

robertn972 at gmail dot com ¶

14 years ago

I found this useful in converting latin characters

```
<?php
function convertLatin1ToHtml($str) {
$allEntities = get_html_translation_table(HTML_ENTITIES, ENT_NOQUOTES);
$specialEntities = get_html_translation_table(HTML_SPECIALCHARS, ENT_NOQUOTES);
$noTags = array_diff($allEntities, $specialEntities);
$str = strtr($str, $noTags);
return $str;
}
?>
```

up
down
-7

alan at akbkhome dot com ¶

20 years ago

If you want to decode all those { symbols as well....

```
function unhtmlentities ($string)  {
    $trans_tbl = get_html_translation_table (HTML_ENTITIES);
    $trans_tbl = array_flip ($trans_tbl);
    $ret = strtr ($string, $trans_tbl);
    return  preg_replace('/\&\#([0-9]+)\;/me',
        "chr('\\1')",$ret);
}
```

up
down
-5

kevin_bro at hostedstuff dot com ¶

19 years ago

Alans version didn't seem to work right. If you're having the same problem consider using this slightly modified version instead:

```
function unhtmlentities ($string)  {
   $trans_tbl = get_html_translation_table (HTML_ENTITIES);
   $trans_tbl = array_flip ($trans_tbl);
   $ret = strtr ($string, $trans_tbl);
   return preg_replace('/&#(\d+);/me',
      "chr('\\1')",$ret);
}
```

+ add a note