PHP-Zeichenkettenfunktionen: kodieren und formatieren

Die Kodierungs- und Formatierungsoperatoren von eZTemplateStringsOperator (bin2hex, hex2bin, convert_uuencode, convert_uudecode, quoted_printable_encode, quoted_printable_decode, htmlentities, html_entity_decode…

Source file:
`doc/ger-de/hilfe/template-operatoren/php-zeichenkettenfunktionen-kodieren-und-formatieren.md`
Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/template-operators/php-string-functions-encode-and-format.md

Die letzte Gruppe von `eZTemplateStringsOperator` (`lib/eztemplate/classes/eztemplatestringsoperator.php`): Kodierungen, HTML-Entitäten und Formatierung im Stil von printf. Die Eingabe über die Pipe ist das Hauptargument der PHP-Funktion, die Template-Parameter sind die übrigen, der Reihe nach (wie die Funktionen zugeordnet werden).

PHP-Flag-Konstanten wie `ENT_QUOTES` gibt es in Templates nicht. Wo eine Funktion Flags nimmt, liefert die Klasse einen sinnvollen Standardwert; um andere Flags zu übergeben, geben Sie ihren Ganzzahlwert an (`ENT_QUOTES` ist 3, `ENT_NOQUOTES` 0).

Inhalt

1. Hex, uuencode und quoted-printable
2. HTML-Entitäten
3. sprintf und vsprintf
4. number_format
5. Verwandte Seiten

Hex, uuencode und quoted-printable

Keiner dieser Operatoren nimmt einen Parameter. Alle geben eine Zeichenkette zurück, außer den Dekodierern, die für Eingaben, die sie nicht dekodieren können, `false` zurückgeben (die PHP-Warnung wird unterdrückt).

bin2hex und hex2bin

`bin2hex()`: jedes Byte als zwei Hexadezimalziffern. `hex2bin()`: die Umkehrung oder `false`. Generierte Referenzen: bin2hex, hex2bin.

convert_uuencode und convert_uudecode

Die uuencode-Kodierung und -Dekodierung von PHP (`convert_uudecode` gibt bei fehlerhafter Eingabe `false` zurück). Generierte Referenzen: convert_uuencode, convert_uudecode.

quoted_printable_encode und quoted_printable_decode

Die Quoted-Printable-Kodierung von Mail-Inhalten (RFC 2045). Generierte Referenzen: quoted_printable_encode, quoted_printable_decode.

Template

```
'Fit'|bin2hex = {'Fit'|bin2hex}
'466974'|hex2bin = {'466974'|hex2bin}
'zz'|hex2bin = [{'zz'|hex2bin|choose( 'false', 'true' )}]
'Fit'|convert_uuencode|convert_uudecode = {'Fit'|convert_uuencode|convert_uudecode}
'Grüße'|quoted_printable_encode = {'Grüße'|quoted_printable_encode}
'Gr=C3=BC=C3=9Fe'|quoted_printable_decode = {'Gr=C3=BC=C3=9Fe'|quoted_printable_decode}
```

Ausgabe:

```
'Fit'|bin2hex = 466974
'466974'|hex2bin = Fit
'zz'|hex2bin = [false]
'Fit'|convert_uuencode|convert_uudecode = Fit
'Grüße'|quoted_printable_encode = Gr=C3=BC=C3=9Fe
'Gr=C3=BC=C3=9Fe'|quoted_printable_decode = Grüße
```

HTML-Entitäten

htmlentities

PHP `htmlentities()`: Jedes Zeichen, für das es eine benannte Entität gibt, wird umgewandelt. Ohne Parameter verwendet es `ENT_QUOTES` und UTF-8. Zum Maskieren der Ausgabe ist `wash` der richtige Operator; `htmlentities` schreibt auch Buchstaben wie `é` um, was in UTF-8-Seiten selten gewünscht ist.

Parameter	Typ	Pflicht	Standard
flags	Ganzzahl	nein	`ENT_QUOTES` (3)
encoding	Zeichenkette	nein	`'UTF-8'`
double_encode	boolescher Wert	nein	PHPs `true`

Generierte Referenz: htmlentities.

html_entity_decode

PHP `html_entity_decode()`: Entitäten zurück in Zeichen. Ohne Parameter verwendet es `ENT_QUOTES | ENT_HTML5` und UTF-8. Ein dekodierter Wert muss erneut maskiert werden, bevor er in HTML ausgegeben wird.

Parameter	Typ	Pflicht	Standard
flags	Ganzzahl	nein	`ENT_QUOTES | ENT_HTML5`
encoding	Zeichenkette	nein	`'UTF-8'`

Generierte Referenz: html_entity_decode.

htmlspecialchars_decode

PHP `htmlspecialchars_decode()`: nur `&`, `"`, `'`, `<` und `>`, einmal dekodiert. Standard-Flags `ENT_QUOTES`; mit 0 (`ENT_NOQUOTES`) bleiben Anführungszeichen kodiert.

Parameter	Typ	Pflicht	Standard
flags	Ganzzahl	nein	ENT_QUOTES (3)

Generierte Referenz: htmlspecialchars_decode.

Template

```
'Café & "Bar"'|htmlentities = {'Café & "Bar"'|htmlentities}
'Caf&eacute; &amp; &quot;Bar&quot;'|html_entity_decode = {'Caf&eacute; &amp; &quot;Bar&quot;'|html_entity_decode}
'&lt;b&gt; &amp;amp;'|htmlspecialchars_decode = {'&lt;b&gt; &amp;amp;'|htmlspecialchars_decode}
'it&#039;s'|htmlspecialchars_decode( 0 ) = {'it&#039;s'|htmlspecialchars_decode( 0 )}
```

Ausgabe:

```
'Café & "Bar"'|htmlentities = Caf&eacute; &amp; &quot;Bar&quot;
'Caf&eacute; &amp; &quot;Bar&quot;'|html_entity_decode = Café & "Bar"
'&lt;b&gt; &amp;amp;'|htmlspecialchars_decode = <b> &amp;
'it&#039;s'|htmlspecialchars_decode( 0 ) = it&#039;s
```

sprintf und vsprintf

sprintf

Die Eingabe über die Pipe ist das **Format**; jeder Parameter ist ein Argument. Intern ruft es `vsprintf( format, parameters )` auf, daher funktionieren alle printf-Umwandlungen, einschließlich Argumentnummern (`%2$s`). Gibt eine Zeichenkette zurück. Ein Format mit mehr Umwandlungen als Parametern lässt PHP einen `ArgumentCountError` auslösen, halten Sie beide also im Gleichschritt.

Parameter	Typ	Pflicht	Standard
Argumente	beliebig	nein	keine

Generierte Referenz: sprintf.

vsprintf

Dasselbe, mit den Argumenten in einem Array (ein Parameter, der kein Array ist, wird in eines gepackt). Ohne das Array protokolliert es „Missing required parameter: args_array“.

Parameter	Typ	Pflicht	Standard
Argumente	Array	ja	keine

Generierte Referenz: vsprintf.

Die erste Zeile verwendet die Kategorie Recipes einer Entwicklungsinstallation (Knoten 144):

Template

```
{def $node = fetch( 'content', 'node', hash( 'node_id', 144 ) )}
{'%s has %d children'|sprintf( $node.name, $node.children_count )}
{'%05.1f%%'|sprintf( 42.25 )}
{'%2$s, %1$s'|sprintf( 'first', 'second' )}
{'%s and %s'|vsprintf( array( 'fit', 'healthy' ) )}
```

Ausgabe:

```
Recipes has 18 children
042.2%
second, first
fit and healthy
```

`sprintf` maskiert nicht: Wenden Sie `wash` auf einen Wert aus Inhalten an, entweder bevor Sie ihn übergeben, oder auf das ganze Ergebnis.

number_format

number_format

PHP `number_format()` auf die in eine Gleitkommazahl umgewandelte Eingabe. Die Zuordnung der Parameter hat eine Eigenheit: Mit **zwei** Parametern ist das Tausendertrennzeichen fest auf `,` gesetzt, sodass `number_format( 2, ',' )` auf beiden Seiten Kommas setzt. Geben Sie alle drei an, um beide Trennzeichen zu wählen.

Parameter	Typ	Pflicht	Standard
Nachkommastellen	Ganzzahl	nein	0
Dezimaltrennzeichen	Zeichenkette	nein	.
Tausendertrennzeichen	Zeichenkette	nein	,

Gibt eine Zeichenkette zurück. Für Zahlen in der Locale des Besuchers verwenden Sie stattdessen `l10n( 'number' )` (Locale-Operatoren). Generierte Referenz: number_format.

Template

```
1234567.891|number_format = {1234567.891|number_format}
1234567.891|number_format( 2 ) = {1234567.891|number_format( 2 )}
1234567.891|number_format( 2, ',' ) = {1234567.891|number_format( 2, ',' )}
1234567.891|number_format( 2, ',', '.' ) = {1234567.891|number_format( 2, ',', '.' )}
1234567.891|number_format( 2, '.', ' ' ) = {1234567.891|number_format( 2, '.', ' ' )}
```

Ausgabe:

```
1234567.891|number_format = 1,234,568
1234567.891|number_format( 2 ) = 1,234,567.89
1234567.891|number_format( 2, ',' ) = 1,234,567,89
1234567.891|number_format( 2, ',', '.' ) = 1.234.567,89
1234567.891|number_format( 2, '.', ' ' ) = 1 234 567.89
```

Verwandte Seiten

- PHP-Zeichenkettenfunktionen: suchen und vergleichen
- PHP-Zeichenkettenfunktionen: umwandeln
- Zeichenketten-Operatoren: `wash`
- Locale-Operatoren: Zahlen, Währung und Datum für eine Locale
- Alle Template-Operatoren (generiert)
- Freie Dokumentation: Diese Seite darf unter der GNU FDL 1.3 frei kopiert, übersetzt und verbessert werden.

Werden Sie Exponential-Partner

Partner teilen ihre Informationen, stellen Mittel bereit, geben Beiträge zurück und entwickeln die Produkte mit uns, und so wächst Exponential. Eine Partner-Website und ein Dashboard sind für Mitte 2027 geplant: Werden Sie noch heute Partner, damit wir schneller wachsen und neue Funktionen früher liefern.

Partner: hier beginnen | Wie Sie sich bewerben

Exponential Copyright © 1998-2026 Exponential Foundation and 7x
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".
This documentation is free documentation, licensed under the GNU Free Documentation License 1.3 or later:
https://www.gnu.org/licenses/fdl-1.3.html.