5.2 Zeichen escapen

Jede Eingabe durch den Benutzer, auf welche Art auch immer, stellt ein Sicherheitsrisiko dar, weil ein Eingabefeld quasi eine offene "Schnittstelle" zum PHP Code ist. So kann der Benutzer in ein `<textarea>` *Element ebenfalls schadhaften HTML, JavaScript oder sogar PHP Code unterbringen. Um auf Nummer Sicher zu gehen, sollte man HTML Zeichen (html entities) escapen - aus* `<` *wird* `<` *und aus* `>` *wird* `>` *usw.*

PHP

`htmlentities($string)`

Wandelt alle Zeichen in die HTML-Code-Entsprechung (Entities) um.
Man kann sich an der Zeichenreferenz von selfhtml.org orientieren.
https://wiki.selfhtml.org/wiki/Referenz:HTML/Zeichenreferenz

```php
<?php
 $eingabe = '<p title="wichtig">Österreich</p>';
 $sauber = htmlentities($eingabe);
 echo $sauber;
?>
```

***`htmlentities()`** erlaubt noch weitere Parameter.*
Die interessantesten Flags sind:

`ENT_QUOTES` ← *Konvertiert doppelte **und** einfach Anführungszeichen.*
`ENT_HTML5` ← *Behandelt den Code als HTML 5*

Mit den Flags und dem UTF-8 Zeichensatz sieht der Befehl so aus:

```php
htmlentities($string, ENT_QUOTES | ENT_HTML5, "UTF-8");
```

PHP

`html_entity_decode($string)`

Dreht die Sache wieder um und wandelt den HTML-Code in seine ursprünglichen Zeichen zurück.

```php
<?php
 $eingabe = '&lt;hr&gt;';
 $linie = html_entity_decode($eingabe);
 echo $linie;
?>
```

*Auch **`html_entity_decode()`** erlaubt die Parameter.* **`ENT_QUOTES`** *und* **`ENT_HTML5`** *sowie den UTF-8 Zeichensatz!*

```php
html_entity_decode($string, ENT_QUOTES, "UTF-8");
```

PHP

`nl2br($string, false)`

Wandelt alle Zeilenschaltungen in HTML Code. Also aus **`\r\n`** wird **`<br>`**.
Lässt man den **`false`** Parameter weg, wird xhtml verwendet, also **`<br />`**