**String Helper**

The String Helper file contains functions that assist in working with strings.

**Important**

Please note that these functions are NOT intended, nor suitable to be used for any kind of security-related logic.

* [Loading this Helper](https://codeigniter.com/user_guide/helpers/string_helper.html#loading-this-helper)
* [Available Functions](https://codeigniter.com/user_guide/helpers/string_helper.html#available-functions)

[**Loading this Helper**](https://codeigniter.com/user_guide/helpers/string_helper.html#id1)

This helper is loaded using the following code:

$this**->**load**->**helper('string');

[**Available Functions**](https://codeigniter.com/user_guide/helpers/string_helper.html#id2)

The following functions are available:

**random\_string([*$type = 'alnum'*[, *$len = 8*]])**

|  |  |
| --- | --- |
| **Parameters:** | * **$type** (*string*) – Randomization type * **$len** (*int*) – Output string length |
| **Returns:** | A random string |
| **Return type:** | string |

Generates a random string based on the type and length you specify. Useful for creating passwords or generating random hashes.

The first parameter specifies the type of string, the second parameter specifies the length. The following choices are available:

* **alpha**: A string with lower and uppercase letters only.
* **alnum**: Alpha-numeric string with lower and uppercase characters.
* **basic**: A random number based on mt\_rand().
* **numeric**: Numeric string.
* **nozero**: Numeric string with no zeros.
* **md5**: An encrypted random number based on md5() (fixed length of 32).
* **sha1**: An encrypted random number based on sha1() (fixed length of 40).

Usage example:

**echo** random\_string('alnum', 16);

**Note**

Usage of the *unique* and *encrypt* types is DEPRECATED. They are just aliases for *md5* and *sha1* respectively.

**increment\_string(*$str*[, *$separator = '\_'*[, *$first = 1*]])**

|  |  |
| --- | --- |
| **Parameters:** | * **$str** (*string*) – Input string * **$separator** (*string*) – Separator to append a duplicate number with * **$first** (*int*) – Starting number |
| **Returns:** | An incremented string |
| **Return type:** | string |

Increments a string by appending a number to it or increasing the number. Useful for creating “copies” or a file or duplicating database content which has unique titles or slugs.

Usage example:

**echo** increment\_string('file', '\_'); *// "file\_1"*

**echo** increment\_string('file', '-', 2); *// "file-2"*

**echo** increment\_string('file\_4'); *// "file\_5"*

**alternator(*$args*)**

|  |  |
| --- | --- |
| **Parameters:** | * **$args** (*mixed*) – A variable number of arguments |
| **Returns:** | Alternated string(s) |
| **Return type:** | mixed |

Allows two or more items to be alternated between, when cycling through a loop. Example:

**for** ($i **=** 0; $i **<** 10; $i**++**)

{

**echo** alternator('string one', 'string two');

}

You can add as many parameters as you want, and with each iteration of your loop the next item will be returned.

**for** ($i **=** 0; $i **<** 10; $i**++**)

{

**echo** alternator('one', 'two', 'three', 'four', 'five');

}

**Note**

To use multiple separate calls to this function simply call the function with no arguments to re-initialize.

**repeater(*$data*[, *$num = 1*])**

|  |  |
| --- | --- |
| **Parameters:** | * **$data** (*string*) – Input * **$num** (*int*) – Number of times to repeat |
| **Returns:** | Repeated string |
| **Return type:** | string |

Generates repeating copies of the data you submit. Example:

$string **=** "\n";

**echo** repeater($string, 30);

The above would generate 30 newlines.

**Note**

This function is DEPRECATED. Use the native str\_repeat() instead.

**reduce\_double\_slashes(*$str*)**

|  |  |
| --- | --- |
| **Parameters:** | * **$str** (*string*) – Input string |
| **Returns:** | A string with normalized slashes |
| **Return type:** | string |

Converts double slashes in a string to a single slash, except those found in URL protocol prefixes (e.g. http://).

Example:

$string **=** "http://example.com//index.php";

**echo** reduce\_double\_slashes($string); *// results in "http://example.com/index.php"*

**strip\_slashes(*$data*)**

|  |  |
| --- | --- |
| **Parameters:** | * **$data** (*mixed*) – Input string or an array of strings |
| **Returns:** | String(s) with stripped slashes |
| **Return type:** | mixed |

Removes any slashes from an array of strings.

Example:

$str **=** **array**(

'question'  **=>** 'Is your name O\'reilly?',

'answer' **=>** 'No, my name is O\'connor.'

);

$str **=** strip\_slashes($str);

The above will return the following array:

**array**(

'question'  **=>** "Is your name O'reilly?",

'answer' **=>** "No, my name is O'connor."

);

**Note**

For historical reasons, this function will also accept and handle string inputs. This however makes it just an alias for stripslashes().

**trim\_slashes(*$str*)**

|  |  |
| --- | --- |
| **Parameters:** | * **$str** (*string*) – Input string |
| **Returns:** | Slash-trimmed string |
| **Return type:** | string |

Removes any leading/trailing slashes from a string. Example:

$string **=** "/this/that/theother/";

**echo** trim\_slashes($string); *// results in this/that/theother*

**Note**

This function is DEPRECATED. Use the native trim() instead: | | trim($str, ‘/’);

**reduce\_multiples(*$str*[, *$character = ''*[, *$trim = FALSE*]])**

|  |  |
| --- | --- |
| **Parameters:** | * **$str** (*string*) – Text to search in * **$character** (*string*) – Character to reduce * **$trim** (*bool*) – Whether to also trim the specified character |
| **Returns:** | Reduced string |
| **Return type:** | string |

Reduces multiple instances of a particular character occurring directly after each other. Example:

$string **=** "Fred, Bill,, Joe, Jimmy";

$string **=** reduce\_multiples($string,","); *//results in "Fred, Bill, Joe, Jimmy"*

If the third parameter is set to TRUE it will remove occurrences of the character at the beginning and the end of the string. Example:

$string **=** ",Fred, Bill,, Joe, Jimmy,";

$string **=** reduce\_multiples($string, ", ", **TRUE**); *//results in "Fred, Bill, Joe, Jimmy"*

**quotes\_to\_entities(*$str*)**

|  |  |
| --- | --- |
| **Parameters:** | * **$str** (*string*) – Input string |
| **Returns:** | String with quotes converted to HTML entities |
| **Return type:** | string |

Converts single and double quotes in a string to the corresponding HTML entities. Example:

$string **=** "Joe's \"dinner\"";

$string **=** quotes\_to\_entities($string); *//results in "Joe&#39;s &quot;dinner&quot;"*

**strip\_quotes(*$str*)**

|  |  |
| --- | --- |
| **Parameters:** | * **$str** (*string*) – Input string |
| **Returns:** | String with quotes stripped |
| **Return type:** | string |

Removes single and double quotes from a string. Example:

$string **=** "Joe's \"dinner\"";

$string **=** strip\_quotes($string); *//results in "Joes dinner"*