PHP preg\_replace() function

The preg\_replace() function is a built-in function of PHP. It is used to perform a regular expression search and replace.

This function searches for **pattern** in **subject** parameter and replaces them with the **replacement**.

### Syntax

preg\_replace (mixed $pattern, mixed $replacement, mixed $subject, int $limit, int $count)

**pattern:** This parameter can be either a string or an array with strings. It holds the pattern to search in subject parameter

**replacement**

It is a string or an array with strings parameter. This parameter replaces the pattern matched in subject parameter. It is a **mandatory** parameter.

* If the replacement parameter is a string and the pattern parameter is an array, all patterns will be replaced by that string.
* If both replacement and pattern parameters are arrays, each pattern will be replaced by the replacement counterpart.
* If the replacement array consists of fewer elements than the pattern array, any extra pattern will be replaced by an empty string.

**subject**

The subject parameter can also be either a string or an array of string to search and replace.

If the subject is an array, the search and replacement are performed on every entry of subject, and the returned value will also be an array.

**limit**

The limit is an **optional** parameter that specifies the maximum possible replacement for each pattern. The default value of **limit is -1**, which means no limit.

**count**

It is an **optional** parameter. If this parameter is passed, this variable will contain the number of replacements done. This parameter added in **PHP 5.1.0**.

### Return Type

The preg\_replace() function returns an array if the subject parameter is an array otherwise it returns a string.

* After the replacement has done, the modified string will be returned.
* If any matches do not find, the string will remain unchanged.

### Examples

**Simple Replacing**

1. $res = preg\_replace('/abc/', 'efg', $string);       #Replace all 'abc' with 'efg'
2. $res = preg\_replace('/abc/i', 'efg', $string);      #Replace with **case**-insensitive matching
3. $res = preg\_replace('/\s+/', '', $string);      #Strip all whitespace

**Example using backreference followed by numeric literals**

1. <?php
2. $date = 'May 29, 2020';
3. $pattern = '/(\w+) (\d+), (\d+)/i';
4. $replacement = '${1} 5,$3';
6. //display the result returned by preg\_replace
7. echo preg\_replace($pattern, $replacement, $date);
8. ?>
9. **Output:**
10. May 5, 2020

**Example to strip whitespace**

In the below example, preg\_replace() removes all extra whitespace from the given string.

1. <?php
2. $str = 'Camila    Cabello   is    a   Hollywood    singer.';
3. $str = preg\_replace('/\s+/', ' ', $str);
4. echo $str;
5. ?>
6. **Output:**
7. Camila Cabello is a Hollywood singer.

**Example using indexed arrays**

This example will contain a pattern array to replace with replacement array.

1. <?php
2. //declare a string
3. $string = 'The slow black bear runs away from the zoo.';
4. $patterns = **array**();
6. //pattern to search in subject string
7. $patterns[0] = '/slow/';
8. $patterns[1] = '/black/';
9. $patterns[2] = '/bear/';
11. //replacement value to replace with pattern in the given search string
12. $replacements = **array**();
13. $replacements[2] = 'fox';
14. $replacements[1] = 'brown';
15. $replacements[0] = 'quick';
17. //apply preg\_replace function
18. $newstr = preg\_replace($patterns, $replacements, $string);
19. echo "<b>String after replacement:</b> " .$newstr;
20. ?>

**Output:**

String after replacement: The fox brown quick runs away from the zoo.

In the above example, we can see that output is not same as we want. Therefore, by applying ksort() on patterns and replacements before using preg\_replace(), we can get what we want to.

1. <?php
2. //declare a string
3. $string = 'The slow black bear runs away from the zoo.';
4. $patterns = **array**();
6. //pattern to search in subject string
7. $patterns[0] = '/slow/';
8. $patterns[1] = '/black/';
9. $patterns[2] = '/bear/';
11. //replacement value to replace with pattern in the given search string
12. $replacements = **array**();
13. $replacements[2] = 'fox';
14. $replacements[1] = 'brown';
15. $replacements[0] = 'quick';
17. //sort the values of both pattern and replacement
18. ksort($patterns);
19. ksort($replacements);
21. //apply preg\_replace function
22. $newstr = preg\_replace($patterns, $replacements, $string);
23. echo "<b>String after replacement using ksort:</b> " .$newstr;
24. ?>

**Output:**

String after replacement using ksort: The quick brown fox runs away from the zoo.

<?php

function minify($content, $path = '') {

$output = preg\_replace(

array(

'/ {2,}/',

'/<!--.?-->|\t|(?:\r?\n[ \t])+/s'

),

array(

' ',

''

),

$content

);

return $output;

}

//minify JavaScript

//$url = "https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.js";

//echo minify(file\_get\_contents($url));

//minify HTML

$url = "https://reeteshghimire.com.np/";

echo minify(file\_get\_contents($url));

//minify CSS

//$url = "https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/5.2.0-beta1/css/bootstrap.css";

//echo minify(file\_get\_contents($url));

?>