



PHP Regular Expressions

[< Previous](#)[Next >](#)

What is a Regular Expression?

A regular expression is a sequence of characters that forms a search pattern. When you search for data in a text, you can use this search pattern to describe what you are searching for.

A regular expression can be a single character, or a more complicated pattern.

Regular expressions can be used to perform all types of text search and text replace operations.

Syntax

In PHP, regular expressions are strings composed of delimiters, a pattern and optional modifiers.

```
$exp = "/w3schools/i";
```

In the example above, `/` is the **delimiter**, `w3schools` is the **pattern** that is being searched for, and `i` is a **modifier** that makes the search case-insensitive.

The delimiter can be any character that is not a letter, number, backslash or space. The most common delimiter is the forward slash (`/`), but when your pattern contains forward slashes it is convenient to choose other delimiters such as `#` or `~`.

Regular Expression Functions

PHP provides a variety of functions that allow you to use regular expressions. The `preg_match()`, `preg_match_all()` and `preg_replace()` functions are some of the most commonly used ones:

Function	Description
<code>preg_match()</code>	Returns 1 if the pattern was found in the string and 0 if not
<code>preg_match_all()</code>	Returns the number of times the pattern was found in the string, which may also be 0
<code>preg_replace()</code>	Returns a new string where matched patterns have been replaced with another string

Using preg_match()

The `preg_match()` function will tell you whether a string contains matches of a pattern.

Example

Use a regular expression to do a case-insensitive search for "w3schools" in a string:

```
<?php
$str = "Visit W3Schools";
$pattern = "/w3schools/i";
echo preg_match($pattern, $str); // Outputs 1
?>
```

[Try it Yourself »](#)

Using preg_match_all()

The `preg_match_all()` function will tell you how many matches were found for a pattern in a string.

Example

Use a regular expression to do a case-insensitive count of the number of occurrences of "ain" in a string:

```
<?php
$str = "The rain in SPAIN falls mainly on the plains.";
$pattern = "/ain/i";
echo preg_match_all($pattern, $str); // Outputs 4
?>
```

[Try it Yourself »](#)

Using preg_replace()

The `preg_replace()` function will replace all of the matches of the pattern in a string with another string.

Example

Use a case-insensitive regular expression to replace Microsoft with W3Schools in a string:

```
<?php
$str = "Visit Microsoft!";
$pattern = "/microsoft/i";
echo preg_replace($pattern, "W3Schools", $str); // Outputs "Visit W3Schools!"
?>
```

[Try it Yourself »](#)

Regular Expression Modifiers

Modifiers can change how a search is performed.

Modifier	Description
i	Performs a case-insensitive search
m	Performs a multiline search (patterns that search for the beginning or end of a string will match the beginning or end of each line)
u	Enables correct matching of UTF-8 encoded patterns

Regular Expression Patterns

Brackets are used to find a range of characters:

Expression	Description
[abc]	Find one character from the options between the brackets
[^abc]	Find any character NOT between the brackets
[0-9]	Find one character from the range 0 to 9

Metacharacters

Metacharacters are characters with a special meaning:

Metacharacter	Description
	Find a match for any one of the patterns separated by as in: cat dog fish
.	Find just one instance of any character
^	Finds a match as the beginning of a string as in: ^Hello
\$	Finds a match at the end of the string as in: World\$
\d	Find a digit
\s	Find a whitespace character
\b	Find a match at the beginning of a word like this: \bWORD, or at the end of a word like this: WORD\b
\uxxxx	Find the Unicode character specified by the hexadecimal number xxxx

Quantifiers

Quantifiers define quantities:

Quantifier	Description
n+	Matches any string that contains at least one <i>n</i>
n*	Matches any string that contains zero or more occurrences of <i>n</i>
n?	Matches any string that contains zero or one occurrences of <i>n</i>

<code>n{x}</code>	Matches any string that contains a sequence of <i>X</i> <i>n</i> 's
<code>n{x,y}</code>	Matches any string that contains a sequence of <i>X</i> to <i>Y</i> <i>n</i> 's
<code>n{x,}</code>	Matches any string that contains a sequence of at least <i>X</i> <i>n</i> 's

Note: If your expression needs to search for one of the special characters you can use a backslash (`\`) to escape them. For example, to search for one or more question marks you can use the following expression: `$pattern = '/\?+/';`

Grouping

You can use parentheses (`()`) to apply quantifiers to entire patterns. They also can be used to select parts of the pattern to be used as a match.

Example

Use grouping to search for the word "banana" by looking for *ba* followed by two instances of *na*:

```
<?php
$str = "Apples and bananas.";
$pattern = "/ba(na){2}/i";
echo preg_match($pattern, $str); // Outputs 1
?>
```

[Try it Yourself »](#)

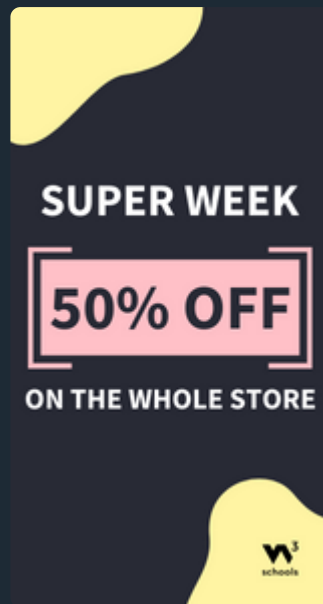
Complete RegExp Reference

For a complete reference, go to our [Complete PHP Regular Expression Reference](#).

The reference contains descriptions and examples of all Regular Expression functions.

[◀ Previous](#)

[Next ▶](#)



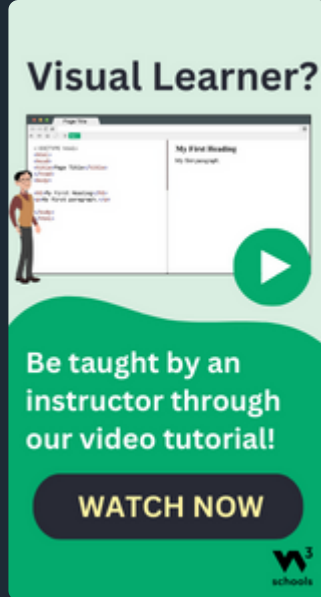
COLOR PICKER



Get certified
by completing
a PHP
course today!



Get started



[Report Error](#)

[Spaces](#)

[Upgrade](#)

[Newsletter](#)

[Get Certified](#)

Top Tutorials

[HTML Tutorial](#)
[CSS Tutorial](#)
[JavaScript Tutorial](#)
[How To Tutorial](#)
[SQL Tutorial](#)
[Python Tutorial](#)
[W3.CSS Tutorial](#)
[Bootstrap Tutorial](#)
[PHP Tutorial](#)
[Java Tutorial](#)
[C++ Tutorial](#)
[jQuery Tutorial](#)

Top References

[HTML Reference](#)
[CSS Reference](#)
[JavaScript Reference](#)
[SQL Reference](#)
[Python Reference](#)
[W3.CSS Reference](#)

[Bootstrap Reference](#)
[PHP Reference](#)
[HTML Colors](#)
[Java Reference](#)
[Angular Reference](#)
[jQuery Reference](#)

Top Examples

[HTML Examples](#)
[CSS Examples](#)
[JavaScript Examples](#)
[How To Examples](#)
[SQL Examples](#)
[Python Examples](#)
[W3.CSS Examples](#)
[Bootstrap Examples](#)
[PHP Examples](#)
[Java Examples](#)
[XML Examples](#)
[jQuery Examples](#)

Get Certified

[HTML Certificate](#)
[CSS Certificate](#)
[JavaScript Certificate](#)
[Front End Certificate](#)
[SQL Certificate](#)
[Python Certificate](#)
[PHP Certificate](#)
[jQuery Certificate](#)
[Java Certificate](#)
[C++ Certificate](#)
[C# Certificate](#)
[XML Certificate](#)

[FORUM](#) | [ABOUT](#)

W3Schools is optimized for learning and training. Examples might be simplified to improve reading and learning. Tutorials, references, and examples are constantly reviewed to avoid errors, but we cannot warrant full correctness of all content. While using W3Schools, you agree to have read and accepted our [terms of use](#), [cookie and privacy policy](#).

Copyright 1999-2022 by Refsnes Data. All Rights Reserved.
W3Schools is Powered by W3.CSS.

