

Perl Regular Expression Quick Reference 1.05

N.B.: this quick reference is just that - some of the explanations have been simplified. For the authoritative documentation, see the latest edition of *Programming Perl* or `perldoc perlre`.

Specific characters:

<code>\t</code>	A tab character
<code>\n</code>	A newline character (OS neutral)
<code>\r</code>	A carriage return character
<code>\f</code>	A form feed character
<code>\cX</code>	Control character CTRL - X
<code>\NNN</code>	Octal code for character NNN

Metacharacters:

The following 12 characters need to be escaped with a backslash - “\” - because by default, they mean something special.

`\ | ( ) [ { ^ $ * + ? .`

<code>.</code>	Match any one character (except <code>\n</code>)
<code> </code>	Alternation
<code>()</code>	Group and capture
<code>[]</code>	Define character class
<code>\</code>	Modify the meaning of the next char.

Anchors:

<code>^</code>	Match at the beginning of a string (or line)
<code>\$</code>	Match at the end of a string (or line)
<code>\b</code>	Match at a ‘word’ boundary
<code>\B</code>	Match at not a ‘word’ boundary

These are also known as *zero width assertions*.

Quantifiers:

These quantifiers apply to the preceding *atom*.

<code>*</code>	Match 0 or more times
<code>+</code>	Match 1 or more times
<code>?</code>	Match 0 or 1 times
<code>{N}</code>	Match exactly N times
<code>{N, }</code>	Match at least N times
<code>{N, M}</code>	Match at least N but not more than M times

By default, quantifiers are “greedy”. They attempt to match as many characters as possible. In order to make them match as few characters as possible, follow them with a question mark “?”.

Character class metacharacters:

<code>^</code>	If the first character of a class, negates that class
<code>-</code>	Unless first or last character of a class, used for a range

Character class shortcuts:

<code>\d</code>	<code>[0-9]</code>	A digit
<code>\D</code>	<code>[^0-9]</code>	A non-digit
<code>\s</code>	<code>[\t\n\r\f]</code>	A whitespace char.
<code>\S</code>	<code>[^\t\n\r\f]</code>	A non-whitespace char.
<code>\w</code>	<code>[a-zA-Z0-9_]</code>	A ‘word’ char.
<code>\W</code>	<code>[^a-zA-Z0-9_]</code>	A ‘non-word’ char.

These shortcuts can be used either on their own, or within a character class.

Copyright © 2002 by Stephen B. Jenkins. <http://www.erudil.com>
All rights reserved. This is free documentation; you can copy and/or redistribute it under the same terms as Perl itself.

Metaquote & case translations:

<code>\Q</code>	Quote (de-meta) characters until <code>\E</code>
<code>\U</code>	Uppercase characters until <code>\E</code>
<code>\L</code>	Lowercase characters until <code>\E</code>

Special variables:

<code>\$`</code>	The characters to the left of the match
<code>\$&</code>	The characters that matched
<code>\$'</code>	The characters to the right of the match
<code>\N</code>	The characters captured by the N th set of parentheses (if on the match side)
<code>\$N</code>	The characters captured by the N th set of parentheses (if not on the match side)

Modifiers:

These modifiers apply to the entire pattern

<code>/i</code>	Ignore case
<code>/g</code>	Match globally (all)
<code>/m</code>	Let <code>^</code> and <code>\$</code> match next to embedded <code>\n</code>
<code>/s</code>	Let <code>.</code> match <code>\n</code>
<code>/x</code>	Ignore most whitespace and allow comments
<code>/e</code>	Evaluate right hand side of <code>s///</code> as an expression

All except `/e` apply to both `m//` and `s///`.

Binding operators:

<code>=~</code>	True if the regex matches
<code>!~</code>	True if the regex doesn’t match

This information is offered in good faith and in the hope that it may be of use, but is not guaranteed to be correct, up to date, or suitable for any particular purpose whatsoever. The author accepts no liability in respect of this information or its use.