

Regular Expression Quick Reference

Version 1.02. Jun 1, 2004

POSIX classes

Letters and digits	<code>[:alnum:]</code>
Letters	<code>[:alpha:]</code>
Space or tab	<code>[:blank:]</code>
Control characters	<code>[:cntrl:]</code>
Decimal digits	<code>[:digit:]</code>
Printing chars, excl. space	<code>[:graph:]</code>
Lowercase letters	<code>[:lower:]</code>
Printing chars, incl. space	<code>[:print:]</code>
Printing chars, excl. letters & digits	<code>[:punct:]</code>
Whitespaces	<code>[:space:]</code>
Uppercase letters	<code>[:upper:]</code>
Hex digits	<code>[:xdigit:]</code>

ESC character, <code>\x1B</code> <code>pm</code>	<code>\e</code>
Newline, (p: <code>\x0D</code> on MacOS 9) <code>py</code>	<code>\n</code>
Carriage return, (p: <code>\x0A</code> on MacOS 9) <code>py</code>	<code>\r</code>
Form feed, <code>\x0C</code> <code>pyas</code>	<code>\f</code>
Horizontal tab, <code>\x09</code> <code>pymas</code>	<code>\t</code>
Vertical tab, <code>\x0B</code> <code>yas</code>	<code>\v</code>
Char specified by 1- to 3-digit octal code <code>s</code>	<code>\ooctal</code>
Char specified by 2- or 3-digit octal code (a: 1- to 3-) <code>pya</code>	<code>\ooctal</code>
Char specified by 1- or 2-digit hex code (y: 2-digit) <code>pyas</code>	<code>\xhex</code>
Char specified by 1- to 3-digit decimal code <code>as</code>	<code>\dddec</code>
Char specified by any hex code <code>p</code>	<code>\xhex</code>
Char specified by a 4-digit hex <code>y</code>	<code>\uhhhh</code>
Char specified by an 8-digit hex <code>y</code>	<code>\Uhhhhhhh</code>
Ctrl-char, char suggested in uppercase <code>pas</code>	<code>\cchar</code>
Escape the meta character <code>asg</code>	<code>\metachar</code>
A named char in the Unicode std or listed in <code>\$PERLLIB/unicode/Names.txt</code> . Reqs <code>use charnames 'full'</code> <code>p</code>	<code>\Nname</code>

Word-boundary <code>p</code>	<code>\b</code>	Match 1 or more times	<code>psag +</code>
Not-word-boundary <code>py</code>	<code>\B</code>		<code>vb +</code>
Beginning of word boundary (positions between a punc/space and a word char) <code>vg</code>	<code>\<</code>	Match 0 or 1 times	<code>psag ?</code>
	<code>\></code>		<code>v =</code>
Positive lookahead <code>py</code>	<code>(?=...)</code>	Match exactly <code>n</code> times	<code>b ?</code>
Negative lookahead <code>py</code>	<code>(?!...)</code>		<code>p {n}</code>
Positive lookbehind; fixed-length only <code>py</code>	<code>(<=...)</code>		<code>vb {n}</code>
Negative lookbehind; fixed-length only <code>py</code>	<code>(<!...)</code>	Match at least <code>n</code> times	<code>sg \{n\}</code>

Mode modifiers

Case-insensitive matching	<code>p/i</code>	Match at most <code>n</code> times	<code>p {,n}</code>
	<code>y/I</code>		<code>vb \{,n}</code>
Match for <code>x</code> to <code>y</code> times (inclusive)	<code>v:set ic</code>		<code>p {x,y}</code>
	<code>s i</code> or <code>I</code>		<code>vb \{x,y}</code>
	<code>a IGNORECASE=1</code>	Match 0 ore more times, as few as possible	<code>sg \{x,y\}</code>
	<code>g -i</code> option	Match 1 or more times, as few as possible	<code>p *?</code>
	<code>v:set noic</code>	Match 0 or 1 times, as few as possible	<code>p +?</code>
Cause <code>\w</code> , <code>\W</code> , <code>\b</code> , <code>\B</code> to use current locale's definition of alphanumeric	<code>y L</code>	Match at least <code>n</code> times, as few as possible	<code>p {n,}?</code>
Cause <code>\w</code> , <code>\W</code> , <code>\b</code> , <code>\B</code> to use Unicode's definition of alphanumeric	<code>y U/u</code>	Match for <code>x</code> to <code>y</code> times (inclusive), as few as possible	<code>p {x,y}?</code>
<code>~</code> and <code>\$</code> match next to embedded <code>\n</code>	<code>p/m</code>		
	<code>y M/m</code>	Match <code>w/if-then-else</code> pattern where <code>cond</code> is an integer referring to either a backreference or a lookahead assertion	
	<code>p/S</code>		<code>(? (cond) ...)</code>
	<code>y S/s</code>	Match <code>w/if-then</code> pattern	<code>(? (cond) ...)</code>
	<code>p/X</code>	Exec. embedded Perl code	<code>{? {code}}</code>
	<code>y X/x</code>	Match regex from embedded Perl code	<code>{?? {code}}</code>
	<code>p/o</code>		

Standard Unicode Properties `p`

Letters	<code>\p{L}</code>
Lowercase letters	<code>\p{Ll}</code>
Modifier letters	<code>\p{Lm}</code>
Other letters (not Ll and not Lm)	<code>\p{Lo}</code>
Titlecase letters	<code>\p{Lt}</code>
Uppercase letters	<code>\p{Lu}</code>
Ctrl codes & chars not in other categories	<code>\p{C}</code>
ASCII and Latin-1 ctrl chars	<code>\p{Cc}</code>
Non-visible formatting chars	<code>\p{Cf}</code>
Unassigned code points	<code>\p{Cn}</code>
Private use, such as company logos	<code>\p{Co}</code>
Surrogates	<code>\p{Cs}</code>
Marks to combine w/base chars, e.g. accents	<code>\p{M}</code>
Mod. chars w/their own space, e.g. "vowel signs"	<code>\p{Mc}</code>
Marks enclose other chars, such as circles	<code>\p{Me}</code>
Chars to mod. other chars, e.g. accents and umlauts	<code>\p{Mn}</code>
Numeric characters	<code>\p{N}</code>
Decimal digits in various scripts	<code>\p{Nd}</code>
Letters that are numbers, e.g. roman numerals	<code>\p{Nl}</code>
Superscripts, symbols, or non-digit char rep. nums	<code>\p{No}</code>
Punctuation	<code>\p{P}</code>
Connecting punctuation, such as an underscore	<code>\p{Pc}</code>
Dashes and hyphens	<code>\p{Pd}</code>
Closing punctuation complementing <code>\p{Ps}</code>	<code>\p{Pe}</code>
Initial punctuation, such as opening quotes	<code>\p{Pi}</code>
Final punctuation, such as closing quotes	<code>\p{Pf}</code>
Other punctuation marks	<code>\p{Po}</code>
Opening punctuation, such as opening parentheses	<code>\p{Ps}</code>
Symbols	<code>\p{S}</code>
Currency	<code>\p{Sc}</code>
Combining chars represented as individual char	<code>\p{Sk}</code>
Math symbols	<code>\p{Sm}</code>
Other symbols	<code>\p{So}</code>
Separating chars with no visual representation	<code>\p{Z}</code>
Line separators	<code>\p{Zl}</code>
Paragraph separators	<code>\p{Zp}</code>
Space characters	<code>\p{Zs}</code>

Char classes and class constructs

A single char listed <code>pyvasg</code>	<code>[...]</code>
A single char not listed <code>pyvasg</code>	<code>[^...]</code>
POSIX-style char class, valid only inside a regex char class <code>pyas</code>	<code>[:class:]</code>
Any char except newline (v: unless /s mode) <code>pyvasg</code>	<code>.</code>
One byte (corrupts Unicode character stream) <code>p</code>	<code>\C</code>
Base char followed by any number of Unicode combining characters <code>p</code>	<code>\X</code>
(non)Word char <code>pymsg</code>	<code>\w</code> or <code>\W</code>
(p: <code>\p{IsWord}</code>) (ym: <code>[a-zA-Z0-9_]</code>)	
(non)Letter character <code>[a-zA-Z]</code> <code>m</code>	<code>\a</code> or <code>\A</code>
(non)Head of word character <code>[a-zA-Z_]</code> <code>m</code>	<code>\h</code> or <code>\H</code>
(non)Digit char, (p: <code>\p{IsDigit}</code>) (ym: <code>[0-9]</code>) <code>py</code>	<code>\d</code> or <code>\D</code>
(non)Whitespace <code>py</code>	<code>\s</code> or <code>\S</code>
(p: <code>\p{IsSpace}</code>) (ym: <code>[\t\n\r\f\v]</code>)	
(non)Hex digit <code>[a-fA-F0-9]</code> <code>m</code>	<code>\x</code> or <code>\X</code>
(non)Octal digit <code>[0-7]</code> <code>m</code>	<code>\o</code> or <code>\O</code>
(non)Lowercase letter <code>[a-z]</code> <code>m</code>	<code>\l</code> or <code>\L</code>
(non)Uppercase letter <code>[A-Z]</code> <code>m</code>	<code>\u</code> or <code>\U</code>
Identifier character defined by <code>isident</code> <code>m</code>	<code>\i</code>
Any non-digit identifier character <code>m</code>	<code>\I</code>
Keyword characters defined by <code>iskeyword</code> , often set by language modes. <code>m</code>	<code>\k</code>
Any non-digit keyword character <code>m</code>	<code>\K</code>
Filename character defined by <code>isfname</code> , OS dependent <code>m</code>	<code>\f</code>
Any non-digit filename character <code>m</code>	<code>\F</code>
Printable character defined by <code>isprint</code> , usually <code>0x20-0x7E</code> <code>m</code>	<code>\p</code>
Any non-digit printable character <code>m</code>	<code>\P</code>
Char (not) contained by given Unicode property, script, or block <code>p</code>	<code>\pprop</code> or <code>\Ppprop</code>

anchors and zero-width tests

Start of str, or after any newline in multiline mode	<code>^</code>
(/m) <code>pyvasg</code>	<code>-</code>
Start of search str, in all match modes <code>py</code>	<code>\A</code>
End of str (excl. newline), or b4 any newline in /m <code>pyvasg</code>	<code>\B</code>
End of str (excl. newline), in any match mode <code>py</code>	<code>\Z</code>
End of str, in any match mode <code>p</code>	<code>\z</code>
Beginning of current search <code>p</code>	<code>\G</code>

Grouping, capturing, conditional, & control

Group subpattern and capture submatch into <code>\1</code> , <code>\2</code> , ... (p: and <code>\$1</code> , <code>\$2</code> , ...)	<code>pa (...)</code>		
Group & capture into named capture group <code>p</code> (<code>?Pname...</code>)	<code>vsb (...)</code>		
Match text matched by earlier named capture group <code>p</code> (<code>?P=name</code>)	<code>\p{C}</code>		<code>\p{IsASCII}</code>
Contains text matched by the <code>n</code> th capture group	<code>pvs \n</code>		<code>\p{IsAlpha}</code>
In replacement string, evaluates to the matched text	<code>v&</code>		<code>\p{IsAlnum}</code>
Groups subpattern, but does not capture submatch	<code>p (? : ...)</code>		<code>\p{IsDigit}</code>
Disallow backtracking for text matched by subpattern	<code>p (?> ...)</code>		<code>\p{IsGraph}</code>
Try subpatterns in alternation	<code>pvsag</code>		<code>\p{IsLower}</code>
	<code>b</code>		<code>\p{IsPrint}</code>
	<code>pvsag * [0-9a-fA-F]</code>		<code>\p{IsPunct}</code>
			<code>\p{IsSpace}</code>
			<code>\p{IsUpper}</code>
			<code>\p{IsWord}</code>
			<code>\p{IsXDigit}</code>

Perl 5.8 Unicode support

<code>[\x00-\x7f]</code>	<code>\p{IsASCII}</code>
<code>[\p{Ll}\p{Lu}\p{Lt}\p{Lo}\p{Nd}]</code>	<code>\p{IsAlnum}</code>
<code>[\p{Ll}\p{Lu}\p{Lt}\p{Lo}]</code>	<code>\p{IsAlpha}</code>
<code>\p{C}</code>	<code>\p{IsCntrl}</code>
<code>\p{Nd}</code>	<code>\p{IsDigit}</code>
<code>[\p{C}\p{Space}]</code>	<code>\p{IsGraph}</code>
<code>\p{Ll}</code>	<code>\p{IsLower}</code>
<code>\p{C}</code>	<code>\p{IsPrint}</code>
<code>\p{P}</code>	<code>\p{IsPunct}</code>
<code>[\t\n\r\f\v\p{Z}]</code>	<code>\p{IsSpace}</code>
<code>[\p{Lu}\p{Lt}]</code>	<code>\p{IsUpper}</code>
<code>[\p{Ll}\p{Lu}\p{Lt}\p{Lo}\p{Nd}]</code>	<code>\p{IsWord}</code>
<code>[0-9a-fA-F]</code>	<code>\p{IsXDigit}</code>

Characters representations

Alert (bell) <code>pyas</code>	<code>\a</code>
Backspace, <code>\x08</code> ; supported only in char class <code>pyma</code>	<code>\b</code>

Application code: Perl 5.8 (p), Python (y), vim and vi (v), vim (m), sed (s), awk (a), egrep (g), grep and basic regex (b)