Regular Expressions Quick Start

This quick start gets you up to speed quickly with regular expressions. explain everything there is to know about regular expressions. For c expressions tutorial. Each topic in the quick start corresponds with a topic and forth between the two.

Many applications and programming languages have their own implemer slight and sometimes with significant differences from other implementatio implementation of regular expressions, we say that they use different "reg explains the syntax supported by the most popular regular expression flavor

Text Patterns and Matches

A regular expression, or regex for short, is a pattern describing a certain expressions are highlighted in red as `regex`. This is actually a perfectly simply matching the literal text `regex`. Matches are highlighted in blue indicate the text that the regular expression is applied to. Strings are highlig

Characters with special meanings in regular expressions are highlighte `(?x)([Rr]egexp?)\?` shows meta tokens in purple, grouping in green, ch other special tokens in blue, and escaped characters in gray.

Literal Characters

The most basic regular expression consists of a single literal character, suc that character in the string. If the string is `Jack is a boy`, it matches the

This regex can match the second `a` too. It only does so when you tell the re string after the first match. In a text editor, you can do so by using its "Finc programming language, there is usually a separate function that you can ca after the previous match.

Twelve characters have special meanings in regular expressions: the back period or dot `.`, the vertical bar or pipe symbol `|`, the question mark `?`, th opening parenthesis `(`, the closing parenthesis `)`, the opening square br These special characters are often called "metacharacters". Most of them a

If you want to use any of these characters as a literal in a regex, you nee want to match `1+1=2`, the correct regex is `1\+1=2`. Otherwise, the plus sigr

Learn more about literal characters

Character Classes or Character Sets

A "character class" matches only one out of several characters. To match a in `gr[ae]y` to match either `gray` or `grey`. A character class matches or match `graay`, `graey` or any such thing. The order of the characters inside

You can use a hyphen inside a character class to specify a range of c between 0 and 9. You can use more than one range. `[0-9a-fA-F]` m insensitively. You can combine ranges and single characters. `[0-9a-fxA` letter X.

Typing a caret after the opening square bracket negates the character cla matches any character that is *not* in the character class. `q[^x]` matches since there is no character after the q for the negated character class to mat

Learn more about character classes

Shorthand Character Classes

`\d` matches a single character that is a digit, `\w` matches a "word c underscore), and `\s` matches a whitespace character (includes tabs and lin

by the shorthands depends on the software you're using. In modern applica
numbers.

Learn more about shorthand character classes

Non-Printable Characters

You can use special character sequences to put non-printable characters in
a tab character (ASCII 0x09), `\r` for carriage return (0x0D) and `\n` for line
are `\a` (bell, 0x07), `\e` (escape, 0x1B), `\f` (form feed, 0x0C) and `\v` (vert
text files use `\r\n` to terminate lines, while UNIX text files use `\n`.

If your application supports Unicode, use `\uFFFF` or `\x{FFFF}` to insert a
matches the euro currency sign.

If your application does not support Unicode, use `\xFF` to match a specific
character set. `\xA9` matches the copyright symbol in the Latin-1 character s

All non-printable characters can be used directly in the regular expression, c

Learn more about non-printable characters

The Dot Matches (Almost) Any Character

The dot matches a single character, except line break characters. Most
"single line" mode that makes the dot match any single character, including

`gr.y` matches `gray`, `grey`, `gr%y`, etc. Use the dot sparingly. Often, a cha
faster and more precise.

Learn more about the dot

Anchors

Anchors do not match any characters. They match a position. `^` matches a
the end of the string. Most regex engines have a "multi-line" mode that m
before any line break. E.g. `^b` matches only the first `b` in `bob`.

`\b` matches at a word boundary. A word boundary is a position between a c
character that cannot be matched by `\w`. `\b` also matches at the start and
characters in the string are word characters. `\B` matches at every position w

Learn more about anchors

Alternation

Alternation is the regular expression equivalent of "or". `cat|dog` matches
regex is applied again, it matches `dog`. You can add as many alternatives a

Alternation has the lowest precedence of all regex operators. `cat|dog fo`
a regex that matches `cat food` or `dog food`, you need to group the alter

Learn more about alternation

Repetition

The question mark makes the preceding token in the regular expression
`color`.

The asterisk or star tells the engine to attempt to match the preceding tok
engine to attempt to match the preceding token once or more. `<[A-Za-z`
without any attributes. `<[A-Za-z0-9]+>` is easier to write but matches inva

Use curly braces to specify a specific amount of repetition. Use `\b[1-9]`
1000 and 9999. `\b[1-9][0-9]{2,4}\b` matches a number between 100

Learn more about quantifiers

Greedy and Lazy Repetition

The repetition operators or quantifiers are greedy. They expand the match
they must to satisfy the remainder of the regex. The regex
`This is a <EM>first</EM> test`.

Place a question mark after the quantifier to make it lazy. `<.+?>` matches <

A better solution is to follow my advice to use the dot sparingly. Use `<[^<>`
regard to attributes. The negated character class is more specific than th
matches quickly.

Learn more about greedy and lazy quantifiers

Grouping and Capturing

Place parentheses around multiple tokens to group them together. You can
`Set(Value)?` matches `Set` or `SetValue`.

Parentheses create a capturing group. The above example has one gro
contains nothing if `Set` was matched. It contains `Value` if `SetValue` w
contents depends on the software or programming language you're using
regex match.

Use the special syntax `Set(?:Value)?` to group tokens without creating
you don't plan to use the group's contents. Do not confuse the question ma
the quantifier.

Learn more about grouping and capturing

Backreferences

Within the regular expression, you can use the backreference `\1` to matcl
capturing group. `([abc])=\1` matches `a=a`, `b=b`, and `c=c`. It does not
multiple capturing groups, they are numbered counting their opening parent

Learn more about backreferences

Named Groups and Backreferences

If your regex has many groups, keeping track of their numbers can get cu
read by naming your groups. `(?<mygroup>[abc])=\k<mygroup>` is iden
refer to the group by its name.

Learn more about named groups

Unicode Properties

`\p{L}` matches a single character that is in the given Unicode category. `L`
character that is not in the given Unicode category. You can find a complete

Learn more about Unicode regular expressions

Lookaround

Lookaround is a special kind of group. The tokens inside the group are ma
makes the group give up its match and keeps only the result. Lookaround
does not expand the regex match.

`q(?=u)` matches the `q` in `question`, but not in `Iraq`. This is positive lo
regex match. The lookahead matches at each position in the string before a

`q(?!u)` matches `q` in `Iraq` but not in `question`. This is negative lookahe
attempted, their match is discarded, and the result is inverted.

To look backwards, use lookbehind. `(?<=a)b` matches the `b` in `abc`. This
match `abc`.

You can use a full-fledged regular expression inside lookahead. Most applic
in lookbehind.

Learn more about lookaround

Free-Spacing Syntax

Many application have an option that may be labeled "free-spacing" or "igno
the regular expression engine ignore unescaped spaces and line breaks
comment that runs until the end of the line. This allows you to use whitespa
way that makes it easier for humans to read and thus makes it easier to ma

Learn more about free-spacing

Make a Donation

Did this website just save you a trip to the bookstore? Please make a don
lifetime of advertisement-free access to this site!

Page URL: https://www.regular-expressions.info/quickstart.html
Page last updated: 08 December 2016
Site last updated: 20 June 2019