How to negate specific word in regex?

Asked 10 years, 2 months ago Active 5 days ago Viewed 629k times

581

130

I know that I can negate group of chars as in `[^bar]` but I need a regular expression where negation applies to the specific word - so in my example how do I negate an actual `bar`, and not "any chars in bar"?

regex

edited Oct 24 at 7:20
Marc.2377
2,941 4 26 62

asked Aug 6 '09 at 17:20
Bostone
22.8k 38 151 208

1 Possible duplicate of Regular expression to match line that doesn't contain a word? – Steve Chambers Oct 25 '16 at 10:44

Related: regex for matching something if it is not preceded by something else – Marc.2377 Oct 24 at 7:29

11 Answers

642

A great way to do this is to use negative lookahead:

```
^(?!.*bar).*$
```

> The negative lookahead construct is the pair of parentheses, with the opening parenthesis followed by a question mark and an exclamation point. Inside the lookahead [is any regex pattern].

edited Dec 5 '18 at 20:35
robopim
7,590 4 43 51

answered Aug 6 '09 at 17:38
Chris Van Opstal
29.6k 8 64 89

10 This says it all (I probably would have started with (?!bar) and built up). I don't see why other people are making it so complicated. – Beta Aug 7 '09 at 14:49

36 Unfortunately, this doesn't work with all languages. – JAB Aug 7 '09 at 18:01

4 line start character at the beginning does a pretty good job. – dhblah Oct 23 '12 at 8:39

2 Nicely done - matches a line that has the specified string and the string is not preceded by anything and the string is followed by anything.This is by definition the absence of the string! because if present it will always be preceded by something even if its a line anchor ^ – Pete_ch Nov 13 '14 at 15:35

2 @NeilTraft how about `grep -v bar` :) – bobbel Aug 12 '16 at 16:00

60

Unless performance is of utmost concern, it's often easier just to run your results through a second pass, skipping those that match the words you want to negate.

Regular expressions usually mean you're doing scripting or some sort of low-performance task anyway, so find a solution that is easy to read, easy to understand and easy to maintain.

answered Aug 6 '09 at 17:33
Bryan Oakley
242k 23 320 472

7 There are lots of situations where you don't control the workflow: you just get to write a single regexp which is a filter. – Steve Bennett Mar 22 '18 at 4:08

44

The following regex will do what you want (as long as negative lookbehinds and lookaheads are supported), matching things properly; the only problem is that it matches individual characters (i.e. each match is a single character rather than all characters between two consecutive "bar"s), possibly resulting in a potential for high overhead if you're working with very long strings.

```
b(?!ar)|(?<!b)a|a(?!r)|(?<!ba)r|[^bar]
```

edited Jun 19 '12 at 14:39

community wiki
11 revs
JAB

7 Instead of those multiple updates which force us to read the wrong answers before getting to your final answer, why not rewrite your answer to be complete, but without the somewhat confusing bad parts? If somebody really cares about the edit history they can use the built-in features of this site. – Bryan Oakley Jun 19 '12 at 13:12

13 Been two and a half years since I wrote this answer, but sure. – JAB Jun 19 '12 at 14:39

3 damn that hurts, try this (?:(?!bar).)* – Bob Oct 7 '14 at 18:15

@Mary, This won't work as expected. For example `/(?:(?!bar).)*/g` on `foobar` returns `foo` AND `ar` . – Krzysiek Jan 7 '15 at 16:08

43

You could either use a negative look-ahead or look-behind:

```
^(?!.*?bar).*
^(.(?<!bar))*?$
```

Or use just basics:

```
^(?:[^b]+|b(?:$|[^a]|a(?:$|[^r])))*$
```

These all match anything that does not contain `bar` .

edited Aug 6 '09 at 18:22

answered Aug 6 '09 at 17:24

Gumbo
539k 94 691 782

What languages don't support (negative) look-behinds and/or (negative) look-aheads in regex? – JAB Aug 6 '09 at 17:29

5 I think the point being made is, looking at your pattern it's not at all clear that all you're doing is rejecting the word "bar". – Bryan Oakley Aug 6 '09 at 17:34

@Bryan: And, in fact, it doesn't reject the word "bar". It just rejects "b" when followed by "ar". – JAB Aug 6 '09 at 18:05

Good idea, but not supported everywhere. Afaik Javascript supports negative look-ahead, but not look-behind. I don't know details about other languages, but this can be helpful: en.wikipedia.org/wiki/Comparison_of_regular_expression_engines – mik01aj Jul 8 '15 at 7:58

4 `(?:[^b][^a][^r])*` – EKons Apr 19 '16 at 17:14

30

I came across this forum thread while trying to identify a regex for the following English statement:

> Given an input string, match **everything** *unless* this input string is exactly 'bar'; for example I want to match 'barrier' and 'disbar' as well as 'foo'.

Here's the regex I came up with

```
^(bar.+|(?!bar).*)$
```

My English translation of the regex is "match the string if it starts with 'bar' and it has at least one other character, or if the string does not start with 'bar'.

edited Oct 27 '11 at 22:32
Alan Moore
63.6k 10 83 140

answered Sep 10 '10 at 20:44
ReQuest Programmer
301 3 3

@ReReqest - you will have much better chance to have this question answered if you post it as a separate question. In that you can provide link back to this question if you want. For the substance of question - it looks OK but I'm no regex guru – Bostone Sep 11 '10 at 17:47

1 That was the one I was looking for. It really matches everything except bar. – Gabriel Hautclocq Dec 17 '15 at 20:34

4 `^(?!bar$).*` matches the same as this (everything except exactly `bar`) and avoids repetition. – bkDJ Jun 6 '18 at 13:17

29

Solution:

```
^(?!.*STRING1|.*STRING2|.*STRING3).*$
```

xxxxxx **OK**

xxxSTRING1xxx **KO (is whether it is desired)**

xxxSTRING2xxx **KO (is whether it is desired)**

xxxSTRING3xxx **KO (is whether it is desired)**

edited Sep 13 '16 at 16:24

answered Sep 13 '16 at 16:08
sgrillon
4,695 3 39 69

2 thanks, this gave me the extra info i needed for multiple words – RozzA Oct 30 '16 at 18:37

10

The accepted answer is nice but is really a work-around for the lack of a simple sub-expression negation operator in regexes. This is why `grep --invert-match` exits. So in *nixes, you can accomplish the desired result using pipes and a second regex.

```
grep 'something I want' | grep --invert-match 'but not these ones'
```

Still a workaround, but maybe easier to remember.

answered Jan 4 '16 at 0:04
Greg Bell
1,166 10 16

4

I wish to complement the accepted answer and contribute to the discussion with my late answer.

@ChrisVanOpstal shared this regex tutorial which is a great resource for learning regex.

However, it was really time consuming to read through.

I made a cheatsheet for mnemonic convenience.

This reference is based on the braces `[]`, `()`, and `{}` leading each class, and I find it easy to recall.

```
Regex = {
 'single_character': ['[]', '.', {'negate':'^'}],
 'capturing_group' : ['()', '|', '\\', 'backreferences and named group'],
 'repetition'      : ['{}', '*', '+', '?', 'greedy v.s. lazy'],
 'anchor'          : ['^', '\b', '$'],
 'non_printable'   : ['\n', '\t', '\r', '\f', '\v'],
 'shorthand'       : ['\d', '\w', '\s'],
 }
```

edited Feb 18 at 3:02
HAL 9001
2,281 1 9 21

answered Dec 6 '17 at 6:32
Algebra
6,186 2 28 54

1

I had a list of file names, and I wanted to exclude certain ones, with this sort of behavior (Ruby):

```
files = [
  'mydir/states.rb',      # don't match these
  'countries.rb',
  'mydir/states_bkp.rb',  # match these
  'mydir/city_states.rb'
]
excluded = ['states', 'countries']

# set my_rgx here

result = WankyAPI.filter(files, my_rgx)  # I didn't write WankyAPI...
assert result == ['mydir/city_states.rb', 'mydir/states_bkp.rb']
```

Here's my solution:

```
excluded_rgx = excluded.map{|e| e+'\.'}.join('|')
my_rgx = /(^|\/)((?!#{excluded_rgx})[^\.\/]*)\.rb$/
```

My assumptions for this application:

- The string to be excluded is at the beginning of the input, or immediately following a slash.
- The permitted strings end with `.rb` .
- Permitted filenames don't have a `.` character before the `.rb` .

edited Feb 25 '16 at 0:46

answered Nov 6 '15 at 11:42

Chaim Leib Halbert

883 8 19

1

Just thought of something else that could be done. It's very different from my first answer, as it doesn't use regular expressions, so I decided to make a second answer post.

Use your language of choice's `split()` method equivalent on the string with the word to negate as the argument for what to split on. An example using Python:

```
>>> text = 'barbarasdbarbar 1234egb ar bar32 sdfbaraadf'
>>> text.split('bar')
['', '', 'asd', '', ' 1234egb ar ', '32 sdf', 'aadf']
```

The nice thing about doing it this way, in Python at least (I don't remember if the functionality would be the same in, say, Visual Basic or Java), is that it lets you know indirectly when "bar" was repeated in the string due to the fact that the empty strings between "bar"s are included in the list of results (though the empty string at the beginning is due to there being a "bar" at the beginning of the string). If you don't want that, you can simply remove the empty strings from the list.

edited Aug 11 '09 at 13:12 answered Aug 7 '09 at 19:58
JAB
17.2k 4 55 72

The question specifically asks about regex... – Ajk_P Jun 22 '17 at 18:25

2 @Ajk_P yes but this kind of answers may help the OP think outside the box, they could've been fixated on regexes not realizing that it could be solved without them. – Petruza Jul 21 '17 at 15:53

0

Extracted from this comment by bkDJ:

```
^(?!bar$).*
```

The nice property of this solution is that it's possible to clearly negate (exclude) multiple words:

```
^(?!bar$|foo$|banana$).*
```

answered May 10 at 10:18
leventov
9,215 4 45 80

why do you need trailing `.*` ? – Sasha Bond Aug 13 at 20:36

protected by Alan Moore Oct 27 '11 at 22:22

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?