Regular Expression to match valid dates

Asked 12 years, 2 months ago Active 4 years ago Viewed 202k times

67

31

I'm trying to write a regular expression that validates a date. The regex needs to match the following

- M/D/YYYY
- MM/DD/YYYY
- Single digit months can start with a leading zero (eg: 03/12/2008)
- Single digit days can start with a leading zero (eg: 3/02/2008)
- CANNOT include February 30 or February 31 (eg: 2/31/2008)

So far I have

```
^(([1-9]|1[012])[-/.]([1-9]|[12][0-9]|3[01])[-/.](19|20)\d\d)|((1[012]|0[1-9])
(3[01]|2\d|1\d|0[1-9])(19|20)\d\d)|((1[012]|0[1-9])[-/.](3[01]|2\d|1\d|0[1-9])[-/.]
(19|20)\d\d)$
```

This matches properly EXCEPT it still includes 2/30/2008 & 2/31/2008.

Does anyone have a better suggestion?

Edit: I found the answer on RegExLib

```
^((((0[13578])|([13578])|(1[02]))[\/](([1-9])|([0-2][0-9])|(3[01])))|(((0[469])|
([469])|(11))[\/](([1-9])|([0-2][0-9])|(30)))|((2|02)[\/](([1-9])|([0-2][0-9]))))
[\/]\d{4}$|^\d{4}$
```

It matches all valid months that follow the MM/DD/YYYY format.

Thanks everyone for the help.

regex date

edited Nov 28 '11 at 15:28
stema
76.6k ● 16 ● 86 ● 116

asked Sep 9 '08 at 4:30
NakedBrunch
44.9k ● 13 ● 69 ● 96

84 Your co-workers are going to hate you. – Chris Conway Sep 9 '08 at 5:17

3 This doesn't take leap year into consideration. It outputs 02/29/2011 as a valid date. – Varun Achar Jan 7 '12 at 7:49

2 Check my answer for a reg ex that takes leap years into consideration. – Varun Achar Jan 7 '12 at 9:22

`It matches all valid months that follow the MM/DD/YYYY format.` Fails to validate `1234` ! :(– Aritra B Oct 22 '14 at 9:00

15 Answers

Active Oldest Votes

137

This is not an appropriate use of regular expressions. You'd be better off using

```
[0-9]{2}/[0-9]{2}/[0-9]{4}
```

and then checking ranges in a higher-level language.

answered Sep 9 '08 at 4:37
Chris Conway
51.4k ● 37 ● 119 ● 146

3 This is not a correct regex as it only checks number of digits in month/data/year. – Sanjeev Singh Apr 23 '14 at 8:39

1 Agreed, that's like using a regular expression for phone numbers that checks all possible area codes. What is the point of not including 2/30 or 2/31 if you include 2/29 for non-leap years, and if you include 4/31, 6/31, 9/31, and 11/31? – Jason Goemaat Nov 23 '15 at 13:52

14 @SanjeevSingh that is the point - regular expressions should not be used for data validation. This will match date-like strings, which can then be validated using a proper date library if needed. – dimo414 Dec 14 '15 at 13:19

@Chris perhaps, you add ^ at the beginning and $ at the end of your answer to match the whole date string only. (By the way: I followed your recommendation to realize the check for leap year etc. in the code. Yes: That's definitely better as you said). – primehunter Aug 30 '19 at 10:33

^(([0-9]{0,2})("separator")?){0,2}[1-2]?([0-9]{0,3}) – Pekee Apr 9 at 11:01

52

Here is the Reg ex that matches all valid dates including leap years. Formats accepted mm/dd/yyyy or mm-dd-yyyy or mm.dd.yyyy format

`^(?:(?:(?:0?[13578]|1[02])(\/|-|\.)31)\1|(?:(?:0?[1,3-9]|1[0-2])(\/|-|\.)(?:29|30)\2))(?:(?:1[6-9]|[2-9]\d)?\d{2})$|^(?:0?2(\/|-|\.)29\3(?:(?:(?:1[6-9]|[2-9]\d)?(?:0[48]|[2468][048]|[13579][26])|(?:(?:16|[2468][048]|[3579][26])00))))$|^(?:(?:0?[1-9])|(?:1[0-2]))(\/|-|\.)(?:0?[1-9]|1\d|2[0-8])\4(?:(?:1[6-9]|[2-9]\d)?\d{2})$`

courtesy Asiq Ahamed

edited Apr 13 '16 at 23:10
Chris Martin
27.9k ● 5 ● 64 ● 125

answered Jan 7 '12 at 7:58
Varun Achar
12.8k ● 6 ● 51 ● 69

6 What about year 20BC? (like `-20/1/1`) – Odys Feb 18 '14 at 12:45

11 @Odys - Did you actually need to program that for something, or did you pull that criticism out of a hat? – Dan Nissenbaum Apr 28 '16 at 4:04

3 Yes, at the time of my comment (2+ years ago) I needed to represent dates dating that back and more. – Odys Apr 28 '16 at 10:41

To declare in js use in the following way `var dateReg = new RegExp(['^(?:(?:(?:0?[13578]|1[02])(\\/|-|\\.)31)', '\\1|(?:(?:0?[1,3-9]|1[0-2])(\\/|-|\\.)(?:29|30)', '\\2))(?:(?:1[6-9]|[2-9]\\d)?\d{2})$|^(?:0?2(\\/|-|\\.)', '29\\3(?:(?:(?:1[6-9]|[2-9]\\d)?(?:0[48]|[2468][048]|', '[13579][26])|(?:(?:16|[2468][048]|[3579][26])00))))', '$|^(?:(?:0?[1-9])|(?:1[0-2]))(\\/|-|\\.)', '(?:0?[1-9]|1\\d|2[0-8])\\4', '(?:(?:1[6-9]|[2-9]\\d)?\\d{2})$'].join(''),"g");` – make-me-alive Nov 21 '17 at 12:47

could we get a modified version of dd/mm/yyyy? – Yokhen Dec 12 '17 at 23:18

26

I landed here because the title of this question is broad and I was looking for a regex that I could use to match on a specific date format (like the OP). But I then discovered, as many of the answers and comments have comprehensively highlighted, there are many pitfalls that make constructing an effective pattern very tricky when extracting dates that are mixed-in with poor quality or non-structured source data.

In my exploration of the issues, I have come up with a system that enables you to build a regular expression by arranging together four simpler sub-expressions that match on the delimiter, and valid ranges for the year, month and day fields in the order you require.

These are :-

Delimeters

```
[^\w\d\r\n:]
```

This will match anything that is not a word character, digit character, carriage return, new line or colon. The colon has to be there to prevent matching on times that look like dates (see my test Data)

You can optimise this part of the pattern to speed up matching, but this is a good foundation that detects most valid delimiters.

Note however; It will match a string with mixed delimiters like this 2/12-73 that may not actually be a valid date.

Year Values

```
(\d{4}|\d{2})
```

This matches a group of two or 4 digits, in most cases this is acceptable, but if you're dealing with data from the years 0-999 or beyond 9999 you need to decide how to handle that because in most cases a 1, 3 or >4 digit year is garbage.

Month Values

```
(0?[1-9]|1[0-2])
```

Matches any number between 1 and 12 with or without a leading zero - note: 0 and 00 is not matched.

Date Values

```
(0?[1-9]|[12]\d|30|31)
```

Matches any number between 1 and 31 with or without a leading zero - note: 0 and 00 is not matched.

This expression matches Date, Month, Year formatted dates

```
(0?[1-9]|[12]\d|30|31)[^\w\d\r\n:](0?[1-9]|1[0-2])[^\w\d\r\n:](\d{4}|\d{2})
```

But it will also match some of the Year, Month Date ones. It should also be bookended with the boundary operators to ensure the whole date string is selected and prevent valid sub-dates being extracted from data that is not well-formed i.e. without boundary tags 20/12/194 matches as 20/12/19 and 101/12/1974 matches as 01/12/1974

Compare the results of the next expression to the one above with the test data in the nonsense section (below)

```
\b(0?[1-9]|[12]\d|30|31)[^\w\d\r\n:](0?[1-9]|1[0-2])[^\w\d\r\n:](\d{4}|\d{2})\b
```

There's no validation in this regex so a well-formed but invalid date such as 31/02/2001 would be matched. That is a data quality issue, and as others have said, your regex shouldn't need to validate the data.

Because you (as a developer) can't guarantee the quality of the source data you do need to perform and handle additional validation in your code, if you try to match **and** validate the data in the RegEx it gets very messy and becomes difficult to support without **very** concise documentation.

Garbage in, garbage out.

Having said that, if you do have mixed formats where the date values vary, and you have to extract as much as you can; You can combine a couple of expressions together like so;

This (disastrous) expression matches DMY and YMD dates

```
(\b(0?[1-9]|[12]\d|30|31)[^\w\d\r\n:](0?[1-9]|1[0-2])[^\w\d\r\n:](\d{4}|\d{2})\b)|
(\b(0?[1-9]|1[0-2])[^\w\d\r\n:](0?[1-9]|[12]\d|30|31)[^\w\d\r\n:](\d{4}|\d{2})\b)
```

BUT you won't be able to tell if dates like 6/9/1973 are the 6th of September or the 9th of June. I'm struggling to think of a scenario where that is not going to cause a problem somewhere down the line, it's bad practice and you shouldn't have to deal with it like that - find the data owner and hit them with the governance hammer.

Finally, if you want to match a YYYYMMDD string with no delimiters you can take some of the uncertainty out and the expression looks like this

```
\b(\d{4})(0[1-9]|1[0-2])(0[1-9]|[12]\d|30|31)\b
```

But note again, it will match on well-formed but invalid values like 20010231 (31th Feb!) :)

Test data

In experimenting with the solutions in this thread I ended up with a test data set that includes a variety of valid and non-valid dates and some tricky situations where you may or may not want to match i.e. Times that could match as dates and dates on multiple lines.

I hope this is useful to someone.

```
Valid Dates in various formats

Day, month, year
2/11/73
02/11/1973
2/1/73
02/01/73
31/1/1973
02/1/1973
31.1.2011
31-1-2001
29/2/1973
29/02/1976
03/06/2010
12/6/90

month, day, year
02/24/1975
06/19/66
03.31.1991
2.29.2003
02-29-55
03-13-55
03-13-1955
12\24\1974
12\30\1974
1\31\1974
03/31/2001
01/21/2001
12/13/2001
```

```
Match both DMY and MDY
12/12/1978
6/6/78
06/6/1978
6/06/1978

using whitespace as a delimiter

13 11 2001
11 13 2001
11 13 01
13 11 01
1 1 01
1 1 2001

Year Month Day order
76/02/02
1976/02/29
1976/2/13
76/09/31

YYYYMMDD sortable format
19741213
19750101

Valid dates before Epoch
12/1/10
12/01/660
12/01/00
12/01/0000

Valid date after 2038

01/01/2039
01/01/39

Valid date beyond the year 9999

01/01/10000

Dates with leading or trailing characters

12/31/21/
31/12/1921AD
31/12/1921.10:55
12/10/2016  8:26:00.39
wfuwdf12/11/74iuhwf
fwefew13/11/1974
01/12/1974vdwdfwe
01/01/99werwer
12321301/01/99

Times that look like dates

12:13:56
13:12:01
1:12:01PM
1:12:01 AM

Dates that runs across two lines

1/12/19
74

01/12/19
```

```
74/13/1946

31/12/20
08:13

Invalid, corrupted or nonsense dates

0/1/2001
1/0/2001
00/01/2100
01/0/2001
0101/2001
01/131/2001
31/31/2001
101/12/1974
56/56/56
00/00/0000
0/0/1999
12/01/0
12/10/-100
74/2/29
12/32/45
20/12/194

2/12-73
```

edited Oct 28 '16 at 16:50

answered Oct 28 '16 at 16:45

Bob

688 ●7 ●11

3 Very nice explanation with examples! May consider adding other month formats like MMM and full month name regex also! – AVA May 4 '17 at 14:52

Thank you! Does the "disastrous" expression have a bug? I wasn't able to get it to match `yyyy-mm-dd` -format dates and had to change it to (in Perl): `/(          (\b(0?[1-9]|[12]\d|30|31)[^\w\d\r\n:](0?[1-9]|1[0-2])[^\w\d\r\n:](\d{4}|\d{2})\b)          |          (\d{4}|\d{2})[^\w\d\r\n:](\b(0?[1-9]|1[0-2])[^\w\d\r\n:](0?[1-9]|[12]\d|30|31)\b) )/x'` --- Tested as : `echo 2017-01-28 | perl -ne 'print "$1\n" if /((\b(0?[1-9]|[12]\d|30|31)[^\w\d\r\n:](0?[1-9]|1[0-2])[^\w\d\r\n:](\d{4}|\d{2})\b)|(\d{4}|\d{2})[^\w\d\r\n:](\b(0?[1-9]|1[0-2])[^\w\d\r\n:](0?[1-9]|[12]\d|30|31)\b))/'` – cxw Oct 3 '17 at 12:44

13

Maintainable Perl 5.10 version

```
/
  (?:
      (?<month> (?&mon_29)) [\/] (?<day>(?&day_29))
    | (?<month> (?&mon_30)) [\/] (?<day>(?&day_30))
    | (?<month> (?&mon_31)) [\/] (?<day>(?&day_31))
  )
  [\/]
  (?<year> [0-9]{4})

  (?(DEFINE)
    (?<mon_29> 0?2 )
    (?<mon_30> 0?[469]   | (11) )
    (?<mon_31> 0?[13578] | 1[02] )

    (?<day_29> 0?[1-9] | [1-2]?[0-9] )
    (?<day_30> 0?[1-9] | [1-2]?[0-9] | 30 )
    (?<day_31> 0?[1-9] | [1-2]?[0-9] | 3[01] )
  )
/x
```

You can retrieve the elements by name in this version.

```
say "Month=$+{month} Day=$+{day} Year=$+{year}";
```

(No attempt has been made to restrict the values for the year.)

edited Jun 20 at 9:12
Community ♦
1 ● 1

answered Sep 13 '08 at 21:28
Brad Gilbert
30.5k ● 8 ● 71 ● 117

Wouldn't this match "12/00/0000"? – mwolfetech Aug 8 '13 at 16:34

@mwolfetech That is true of most of the others as well, If you need the check that, it should be easy to figure out how to modify this regular expression. – Brad Gilbert Aug 9 '13 at 0:31

`+1` for having a version that is actually maintainable. – Mike H-R Jun 24 '14 at 13:18

6

To control a date validity under the following format :

> YYYY/MM/DD or YYYY-MM-DD

I would recommand you tu use the following regular expression :

```
(((19|20)([2468][048]|[13579][26]|0[48])|2000)[/-]02[/-]29|((19|20)[0-9]{2}[/-]
(0[4678]|1[02])[/-](0[1-9]|[12][0-9]|30)|(19|20)[0-9]{2}[/-](0[1359]|11)[/-](0[1-9]|
[12][0-9]|3[01])|(19|20)[0-9]{2}[/-]02[/-](0[1-9]|1[0-9]|2[0-8])))
```

Matches

> 2016-02-29 | 2012-04-30 | 2019/09/31

Non-Matches

> 2016-02-30 | 2012-04-31 | 2019/09/35

You can customise it if you wants to allow only '/' or '-' separators. This RegEx strictly controls the validity of the date and verify 28,30 and 31 days months, even leap years with 29/02 month.

Try it, it works very well and prevent your code from lot of bugs !

FYI : I made a variant for the SQL datetime. You'll find it there (look for my name) : Regular Expression to validate a timestamp

Feedback are welcomed :)

edited May 23 '17 at 12:10
Community ♦
1 ● 1

answered Apr 12 '13 at 9:44
Okipa
485 ● 4 ● 14

4

Sounds like you're overextending regex for this purpose. What I would do is use a regex to match a few date formats and then use a separate function to validate the values of the date fields so extracted.

answered Sep 9 '08 at 4:34
Wedge
18.4k ● 7 ● 44 ● 69

3

Perl expanded version

Note use of `/x` modifier.

```
/^(
      (
        ( # 31 day months
            (0[13578])
          | ([13578])
          | (1[02])
        )
        [\/]
        (
            ([1-9])
          | ([0-2][0-9])
          | (3[01])
        )
      )
    | (
        ( # 30 day months
            (0[469])
          | ([469])
          | (11)
        )
        [\/]
        (
            ([1-9])
          | ([0-2][0-9])
          | (30)
        )
      )
    | ( # 29 day month (Feb)
        (2|02)
        [\/]
        (
            ([1-9])
          | ([0-2][0-9])
          )
        )
      )
      [\/]
      # year
      \d{4}$

  | ^\d{4}$ # year only
/x
```

Original

```
^((((0[13578])|([13578])|(1[02]))[\/](([1-9])|([0-2][0-9])|(3[01])))|(((0[469])|
([469])|(11))[\/](([1-9])|([0-2][0-9])|(30)))|((2|02)[\/](([1-9])|([0-2][0-9]))))
[\/]\d{4}$|^\d{4}$
```

edited Jun 20 at 9:12
Community ♦
1 ● 1

answered Sep 13 '08 at 20:56
Brad Gilbert
30.5k ● 8 ● 71 ● 117

3

if you didn't get those above suggestions working, I use this, as it gets any date I ran this expression through 50 links, and it got all the dates on each page.

```
^20\d\d-(Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec)-(0[1-9]|[1-2][0-9]|3[01])$
```

edited Jan 21 '12 at 1:38
Mark Hall
50.6k ● 8 ● 85 ● 102

answered Jan 21 '12 at 0:03
chuck akers
59 ● 1 ● 7

2

```
    var dtRegex = new RegExp(/[1-9\-]{4}[0-9\-]{2}[0-9\-]{2}/);
    if(dtRegex.test(date) == true){
        var evalDate = date.split('-');
        if(evalDate[0] != '0000' && evalDate[1] != '00' && evalDate[2] != '00'){
            return true;
        }
    }
```

answered Nov 23 '12 at 18:00
ALinnD
21 ● 1

2

This regex validates dates between 01-01-2000 and 12-31-2099 with matching separators.

```
^(0[1-9]|1[012])([- /.])(0[1-9]|[12][0-9]|3[01])\2(19|20)\d\d$
```

edited Apr 29 '13 at 18:15
Jules
13.2k ●11 ●48 ●87

answered Apr 29 '13 at 17:53
Enrique
21 ●1

1

I know this does not answer your question, but why don't you use a date handling routine to check if it's a valid date? Even if you modify the regexp with a negative lookahead assertion like (?!31/0?2) (ie, do not match 31/2 or 31/02) you'll still have the problem of accepting 29 02 on non leap years and about a single separator date format.

The problem is not easy if you want to really validate a date, check this forum thread.

For an example or a better way, in C#, check this link

If you are using another platform/language, let us know

answered Sep 9 '08 at 4:43
Vinko Vrsalovic
238k ●47 ●313 ●359

1

Regex was not meant to validate number ranges(this number must be from 1 to 5 when the number preceding it happens to be a 2 and the number preceding that happens to be below 6). Just look for the pattern of placement of numbers in regex. If you need to validate is qualities of a date, put it in a date object js/c#/vb, and interogate the numbers there.

answered Sep 9 '08 at 4:36
DevelopingChris
36.1k ●27 ●83 ●117

1

Perl 6 version

```
rx{
  ^

  $<month> = (\d ** 1..2)
  { $<month> <= 12 or fail }

  '/'

  $<day> = (\d ** 1..2)
  {
    given( +$<month> ){
      when 1|3|5|7|8|10|12 {
        $<day> <= 31 or fail
      }
      when 4|6|9|11 {
        $<day> <= 30 or fail
      }
      when 2 {
        $<day> <= 29 or fail
        }
        default { fail }
      }
    }

    '/'

    $<year> = (\d ** 4)

    $
  }
```

After you use this to check the input the values are available in `$/` or individually as `$<month>`, `$<day>`, `$<year>`. (those are just syntax for accessing values in `$/`)

No attempt has been made to check the year, or that it doesn't match the 29th of Feburary on non leap years.

edited Dec 27 '15 at 4:53 answered Sep 13 '08 at 21:42
Brad Gilbert
30.5k ● 8 ● 71 ● 117

0

If you're going to insist on doing this with a regular expression, I'd recommend something like:

```
( (0?1|0?3| <...> |10|11|12) / (0?1| <...> |30|31) |
  0?2 / (0?1| <...> |28|29) )
/ (19|20)[0-9]{2}
```

This *might* make it possible to read and understand.

edited Sep 9 '08 at 4:57 answered Sep 9 '08 at 4:45
Chris Conway
51.4k ● 37 ● 119 ● 146

-1

A slightly different approach that may or may not be useful for you.

I'm in php.

The project this relates to will never have a date prior to the 1st of January 2008. So, I take the 'date' inputed and use strtotime(). If the answer is >= 1199167200 then I have a date that is useful to me. If something that doesn't look like a date is entered -1 is returned. If null is entered it does return today's date number so you do need a check for a non-null entry first.

Works for my situation, perhaps yours too?

answered Oct 21 '08 at 12:56
Humpton

1,231 ● 5 ● 15 ● 26

Highly active question. Earn 10 reputation in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.