Javascript percentage validation

Asked 16 years, 2 months ago Modified 1 year, 4 months ago

Viewed 45k times

12

I am after a regular expression that validates a percentage from 0 100 and allows two decimal places.

Does anyone know how to do this or know of good web site that has example of common regular expressions used for client side validation in javascript?

@Tom - Thanks for the questions. Ideally there would be no leading 0's or other trailing characters.

Thanks to all those who have replied so far. I have found the comments really interesting.

javascript regex

Share

Improve this question

Follow

edited Oct 23, 2008 at 10:07

asked Oct 23, 2008 at 1:02

Joel Cunningham

13.7k ● 8 ● 45 ● 49

11 Answers

Sorted by: Highest score (default)

34

Rather than using regular expressions for this, I would simply convert the user's entered number to a floating point value, and then check for the range you want (0 to 100). Trying to do numeric range validation with regular expressions is almost always the wrong tool for the job.

```
var x = parseFloat(str);
if (isNaN(x) || x < 0 || x > 100) {
    // value is out of range
}
```

Share Improve this answer

Follow

answered Oct 23, 2008 at 1:24

Greg Hewgill

990k ● 191 ● 1.2k ● 1.3k

It seems, writing correct regex is often like solving a puzzle - unnecessary but challenging. – Alexander Prokofyev Oct 23, 2008 at 5:58

Good, it doesn't check whenever x have more than 2 decimal, is was trivial to check it, something like this: var decSeparator="."; if((""+x).indexOf(decSeparator)<str.length-3){ alert("alert:Too much decimal!"); } btw, if str is in this format: "12.12aaaa" still validate, but x is just correct. – kentaromiura Oct 23, 2008 at 7:20

mmm, sorry in the code posted above i forgot a piece XD, the correct example is: var decimalSeparator=".",val=""+x; if(val.indexOf(decimalSeparator)<val.length-3){alert("too much decimal");} – kentaromiura Oct 23, 2008 at 7:25

Of course you may need to remove the percent sign if that is allowed in the text field. I used this logic, but determined

whether it had a percentage sign, then removed it, then did the test, then added it back in if it was there originally.
– John Pasquet Aug 1, 2017 at 13:18

17

I propose this one:

```
(^100(\.0{1,2})?$)|(^([1-9]([0-9])?|0)(\.[0-9]{1,2})?$
```

It matches 100, 100.0 and 100.00 using this part

```
^100(\.0{1,2})?$
```

and numbers like 0, 15, 99, 3.1, 21.67 using

```
^([1-9]([0-9])?|0)(\.[0-9]{1,2})?$
```

Note what leading zeros are prohibited, but trailing zeros are allowed (though no more than two decimal places).

Share Improve this answer

Follow

answered Oct 23, 2008 at 6:18

Alexander Prokofyev
34.5k ● 33 ● 100 ● 118

1 one of the most perfect regular express i got till now.
– yogihosting Jan 20, 2017 at 19:17

6

This reminds me of an old blog Entry By Alex Papadimoulis (of The Daily WTF fame) where he tells the following story:

> "A client has asked me to build and install a custom shelving system. I'm at the point where I need to nail it, but I'm not sure what to use to pound the nails in. Should I use an old shoe or a glass bottle?"
>
> How would you answer the question?
>
> 1. It depends. If you are looking to pound a small (20lb) nail in something like drywall, you'll find it much easier to use the bottle, especially if the shoe is dirty. However, if you are trying to drive a heavy nail into some wood, go with the shoe: the bottle with shatter in your hand.
> 2. There is something fundamentally wrong with the way you are building; you need to use real tools. Yes, it may involve a trip to the toolbox (or even to the hardware store), but doing it the right way is going to save a lot of time, money, and aggravation through the lifecycle of your product. You need to stop building things for money until you understand the basics of construction.

This is such a question where most people sees it as a challenge to come up with the correct regular expression to solve the problem, but it would be much better to just say that using regular expressions are using the wrong tool for the job.

The problem when trying to use regex to validate numeric ranges is that it is hard to change if the requirements for the allowed range is changes. Today the requirement may be to validate numbers between 0 and 100 and it is possible to write a regex for that which doesn't make your eyes bleed. But next week the requirment maybe changes so values between 0 and 315 are allowed. Good luck altering your regex.

The solution given by Greg Hewgill is probably better - even though it would validate "99fxx" as "99". But given the circumstances that might actually be ok.

Share Improve this answer

Follow

edited Jun 20, 2020 at 9:12

Community Bot

1 • 1

answered Oct 23, 2008 at 7:14

mlarsen

4,635 • 1 • 22 • 17

2

```
^100(\.(0){0,2})?$|^([1-9]?[0-9])(\.(\d{0,2}))?\%$
```

This would match:
100.00

optional "1-9" followed by a digit (this makes the int part), optionally followed by a dot and two digits

---

From what I see, Greg Hewgill's example doesn't really work that well because *parseFloat('15x')* would simply return 15 which would match the 0<x<100 condition. Using *parseFloat* is clearly wrong because it doesn't validate the percentage value, it tries to force a validation. Some people around here are complaining about leading zeroes and some are ignoring trailing invalid characters. Maybe the author of the question should edit it and make clear what he needs.

Share Improve this answer

Follow

edited Dec 3, 2011 at 2:37

Brock Adams

93.3k ● 23 ● 240 ● 303

answered Oct 23, 2008 at 6:54

Tom

7,213 ● 15 ● 64 ● 79

1 Will match strings like "1.", "95.", "100." Probably regex should be corrected. – Alexander Prokofyev Oct 23, 2008 at 10:25

2

None of the above solutions worked for me, as I needed my regex to allow for values with numbers and a decimal while the user is typing ex: '18.'

This solution allows for an empty string so the user can delete their entire input, and accounts for the other rules articulated above.

```
/(^$)|(^100(\.0{1,2})?$)|(^([1-9]([0-9])?|0)\.(\.[0-9]{1,2})?$)|(^([1-9]([0-9])?|0)(\.[0-9]{1,2})?$)/
```

Share Improve this answer

Follow

answered Feb 10, 2021 at 17:10

Henry Brigham

21 ● 2

The only one I found that works while user is typing!
– Samuel Dec 10, 2022 at 21:36

1

Given that your value is in str

```
str.match(/^(100(\.0{1,2})?|([0-9]?[0-9](\.[0-9]{1,2})
```

Share Improve this answer

Follow

answered Oct 23, 2008 at 1:13

Czimi

2,534 ● 1 ● 17 ● 14

this matches 05.50% which is kind of odd. leading and trailing 0's shouldn't be allowed. – billjamesdev Oct 23, 2008 at 1:50

Hmm, well, maybe just leading 0's should be disallowed. 5.50% isn't meaningless compared to 5.5% – billjamesdev Oct 23, 2008 at 1:51

Unfortunately this regex doesn't match 5 or 15 i. e. numbers without decimal places. – Alexander Prokofyev Oct 23, 2008

at 5:54

1

I recomend this, if you are not exclusively developing for english speaking users:

```
[0-9]{1,2}((,|\.)[0-9]{1,10})?%?
```

You can simply replace the 10 by a 2 to get two decimal places.

My example will match:

```
15.5
5.4366%
1,43
50,55%
34
45%
```

Of cause the output of this one is harder to cast, but something like this will do (Java Code):

```
private static Double getMyVal(String myVal) {
    if (myVal.contains("%")) {
        myVal = myVal.replace("%", "");
    }
    if (myVal.contains(",")) {
        myVal = myVal.replace(',', '.');
    }
    return Double.valueOf(myVal);
}
```

Share Improve this answer

Follow

edited Apr 10, 2018 at 6:58

answered Apr 10, 2018 at 6:39

L. Schilling

89 ● 1 ● 10

For validation in JavaScript you will need to mark start and end too: ^[0-9]{1,2}((,|\.)[0-9]{1,10})?%?$ – L. Schilling Apr 10, 2018 at 13:07

0

```
(100|[0-9]{1,2})(\.[0-9]{1,2})?
```

That should be the regex you want. I suggest you to read Mastering Regular Expression and download RegexBuddy or The Regex Coach.

Share Improve this answer

Follow

edited Oct 23, 2008 at 7:33

nickf

546k ● 198 ● 658 ● 725

answered Oct 23, 2008 at 1:11

Julien Grenier

3,384 ● 2 ● 32 ● 43

2 100.10 will be matched and should not be – Czimi Oct 23, 2008 at 1:13

0

@mlarsen: Is not that a regex here won't do the job better.

Remember that validation msut be done both on client and on server side, so something like:

```
100|(([1-9][0-9])|[0-9])(\.(([0-9][1-9])|[1-9]))?
```

would be a cross-language check, just beware of checking the input length with the output match length.

Share Improve this answer

Follow

answered Oct 23, 2008 at 7:50

kentaromiura

6,509 ● 2 ● 23 ● 15

> And now you have a regex which matches this requirement. A week later the requirements changes so now your regex should match values between 0 and 255 instead. Good luck changing your regex and have it readable afterwards.
> – mlarsen Oct 24, 2008 at 6:27

> Yes, It's true, good point here, but in either case there are downside, so is just a matter of what you want to do... You see, I use to put regex in the configuration file, so if a requirement change, I already know where to modify the validation code. – kentaromiura Oct 24, 2008 at 16:56

0

```
(100(\.(0){1,2})?|([1-9]{1}|[0-9]{2})(\.[0-9]{1,2})?)
```

Share Improve this answer

Follow

edited Sep 4, 2018 at 7:42

Bugs

4,489 ● 9 ● 33 ● 41

answered Sep 4, 2018 at 7:25

Gangadhara S M

1 ● 1

1 Can your add more details to explain how your regex solve the problem ? – Guillaume S Sep 4, 2018 at 7:34

While this might answer the authors question, it lacks some explaining words and links to documentation. Raw code snippets are not very helpful without some phrases around it. You may also find how to write a good answer very helpful. Please edit your answer. – hellow Sep 4, 2018 at 9:15

0

This `\(\d{1,3}%\)|\d{1,3}%` or `/\(\d{1,3}%\)|\d{1,3}%?/g` will match the following cases;

- 100% - Percentage with three digits
- (100%) - Three digits percentage within parenthesis
- 10% - Percentage with two digits
- (10%) - Two digits percentage within parenthesis
- 0% - Percentage with one digit
- (0%) - One digit percentage within parenthesis

Share Improve this answer

edited Jul 24, 2023 at 4:31

Follow

answered Jul 24, 2023 at 4:25

mapmath

1,512 ● 22 ● 41