

How to match "any character" in regular expression?

Asked 14 years, 9 months ago Modified 1 year, 8 months ago Viewed 1.7m times



The following should be matched:

529

AAA123
ABCDEFGH123
XXXX123



can I do: `".*123"` ?



regex

Share Improve this question Follow

edited Feb 24, 2023 at 15:11



MPelletier

16.7k

18

89

140

asked May 26, 2010 at 12:29



Saobi

17.1k

29

73

82

4 This link shows an approach that seems to work --> `[^]+` Which means 'don't match no characters', a double negative that can re-read as 'match any character'. Source - lounge.net/2011/02/... – JJCoder Jul 14, 2016 at 10:30

Yeah, `.*123` works fine. check this [regex demo](#). – Amine KOUIS Jan 11, 2024 at 22:08

13 Answers

Sorted by: Highest score (default)



Yes, you can. That should work.

1032

- `.` = any char except newline
- `\.` = the actual dot character



- `.?` = `{0,1}` = match any char except newline zero or one times
- `.*` = `{0,}` = match any char except newline zero or more times
- `.+` = `{1,}` = match any char except newline one or more times

Share Improve this answer Follow

edited Oct 25, 2020 at 9:50



PHP Guru

1,566 13 22

answered May 26, 2010 at 12:31



Delan Azabani

81.5k 30 172 212

37 Not always dot is means any char. Exception when single line mode. `\p{all}` should be – [martian](#) May 25, 2017 at 15:26 ✎

How can you include backward slash to this character list? – [Dil](#) Dec 9, 2018 at 14:52

3 @pippilongstocking Backward slash is `\\` – [Poutrathor](#) Oct 7, 2019 at 13:49 ✎

10 How do you include new lines? – [GC_](#) Apr 12, 2021 at 19:51

@GC_ dont include `.` (dot) i guest.. – [Fauzan Edris](#) Oct 2, 2022 at 5:40

@GC_ to include new lines `[\s\S]*` worked for me – [mbanchero](#) Oct 14, 2022 at 13:09



71

```
Pattern pattern = Pattern.compile(".*123", Pattern.DOTALL);
Matcher matcher = pattern.matcher(inputStr);
boolean matchFound = matcher.matches();
```



Share Improve this answer Follow



answered May 26, 2010 at 14:17



BlueRaja - Danny
Pflughoeft

86.1k 36 203 293

12 That's some very useful information! I assumed `.` would match newlines. I'm glad I read your answer, I need to use that! – [Ben Kane](#) Sep 4, 2013 at 14:30

1 You may also sometimes need to match newlines in Java regexes in contexts where you cannot pass Pattern.DOTALL, such as when doing a multi-line regex search in Eclipse, or as a user of any Java application that offers regex search. Based on [regular-expression.info's guide](#), you may need to use `{.,\n,\r,\u2028,\u2029,\u0085}` to match absolutely any character (the Unicode characters are additional line-terminating characters added not matched by `.` in Java), but just `{.,\n,\r}` would work for most text files. – Theodore Murdock Nov 3, 2015 at 0:16

13 @TheodoreMurdock `[\s\S]` is a popular way of matching any character if you can't use DOTALL. – mpen Mar 14, 2016 at 16:44

In case it would come to your mind, do NOT use `(?:.|\v)*`, because of [JDK-6337993](#). – Olivier Cailloux Dec 31, 2018 at 13:44



Use the pattern `.` to match any character once, `.*` to match any character zero or more times, `.+` to match any character one or more times.

61



Share Improve this answer Follow

answered May 26, 2010 at 12:31



thr

19.5k

23

96

138



3 This is the only one that works in visual studio's Find tool, but it doesn't match newlines :(– MGOwen Jul 15, 2020 at 5:36



The most common way I have seen to encode this is with a character class whose members form a partition of the set of all possible characters.

43



Usually people write that as `[\s\S]` (whitespace or non-whitespace), though `[\w\W]`, `[\d\D]`, etc. would all work.

Share Improve this answer Follow

answered Mar 13, 2019 at 18:37



James Davis

978


8

13



4 For reference, from [regular-expressions.info/dot.html](#): "JavaScript and VBScript do not have an option to make the dot match line break characters. In those languages, you can use a character class such as `[\s\S]` to match any character. This character matches a character that is either a whitespace

character (including line break characters), or a character that is not a whitespace character. Since all characters are either whitespace or non-whitespace, this character class matches any character." – [Dean Or Aug 26, 2019 at 21:44](#)

3 Vote up this answer. The accepted answer does not answer the question but this does. – [PHP Guru Oct 25, 2020 at 17:24](#) 



.`*` and .`+` are for any chars except for new lines.

19



Double Escaping



Just in case, you would wanted to include new lines, the following expressions might also work for those languages that double escaping is required such as Java or C++:

```
[\\s\\S]*  
[\\d\\D]*  
[\\w\\W]*
```

for zero or more times, or

```
[\\s\\S]+  
[\\d\\D]+  
[\\w\\W]+
```

for one or more times.

Single Escaping:

Double escaping is not required for some languages such as, C#, PHP, Ruby, PERL, Python, JavaScript:

```
[\\s\\S]*  
[\\d\\D]*  
[\\w\\W]*  
[\\s\\S]+
```

```
[\\d\\D]+  
[\\w\\W]+
```

Test

```
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class RegularExpression{  
  
    public static void main(String[] args){  
  
        final String regex_1 = "[\\s\\S]*";  
        final String regex_2 = "[\\d\\D]*";  
        final String regex_3 = "[\\w\\W]*";  
        final String string = "AAA123\\n\\t"  
            + "ABCDEFGH123\\n\\t"  
            + "XXX123\\n\\t";  
  
        final Pattern pattern_1 = Pattern.compile(regex_1);  
        final Pattern pattern_2 = Pattern.compile(regex_2);  
        final Pattern pattern_3 = Pattern.compile(regex_3);  
  
        final Matcher matcher_1 = pattern_1.matcher(string);  
        final Matcher matcher_2 = pattern_2.matcher(string);  
        final Matcher matcher_3 = pattern_3.matcher(string);  
  
        if (matcher_1.find()) {  
            System.out.println("Full Match for Expression 1: " + matcher_1.group(0));  
        }  
  
        if (matcher_2.find()) {  
            System.out.println("Full Match for Expression 2: " + matcher_2.group(0));  
        }  
        if (matcher_3.find()) {  
            System.out.println("Full Match for Expression 3: " + matcher_3.group(0));  
        }  
    }  
}
```

Output

Full Match for Expression 1: AAA123
ABCDEF123
XXXX123

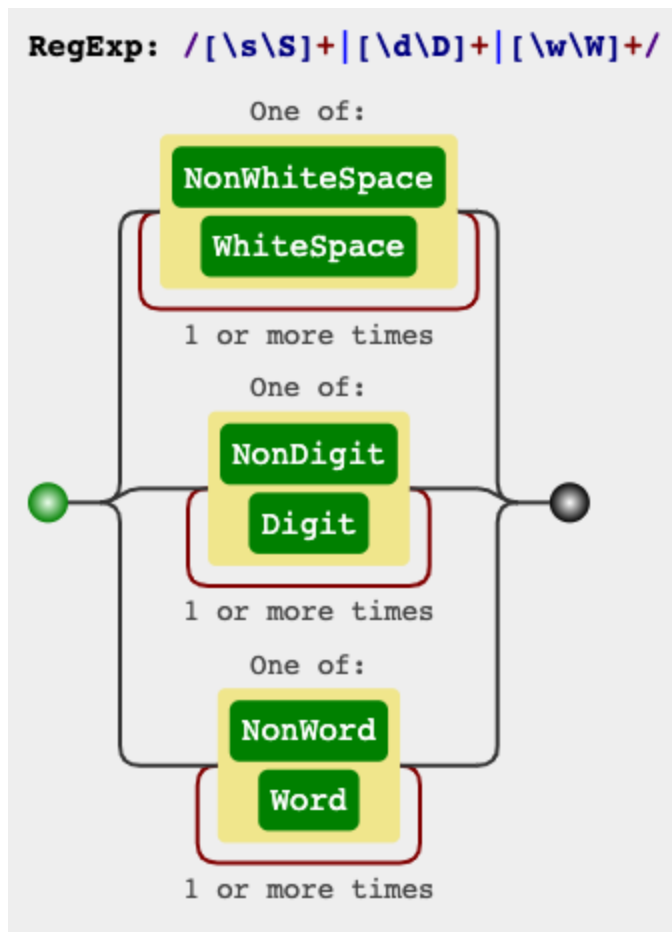
Full Match for Expression 2: AAA123
ABCDEF123
XXXX123

Full Match for Expression 3: AAA123
ABCDEF123
XXXX123

If you wish to explore the expression, it's been explained on the top right panel of regex101.com. If you'd like, you can also watch in [this link](#), how it would match against some sample inputs.

RegEx Circuit

jex.im visualizes regular expressions:



Share Improve this answer Follow

edited Nov 7, 2019 at 20:45

answered Nov 7, 2019 at 19:22



Emma

27.8k

11

48

71

1 Isn't that already answered here? stackoverflow.com/a/55149095/5424988 – The fourth bird Nov 7, 2019 at 21:11

i like `(\W|\w)*` instead of double escaping – Sudip Bhattarai Mar 3, 2020 at 10:31

1 Really helpful explaining – Nagibaba Mar 11, 2020 at 10:47



There are lots of sophisticated regex testing and development tools, but if you just want a simple test harness in Java, here's one for you to play with:

11



```
String[] tests = {
    "AAA123",
    "ABCDEFGH123",
    "XXXX123",
    "XYZ123ABC",
    "123123",
    "X123",
    "123",
};
for (String test : tests) {
    System.out.println(test + " " + test.matches(".*123"));
}
```

Now you can easily add new testcases and try new patterns. Have fun exploring regex.

See also

- regular-expressions.info/Tutorial

Share Improve this answer Follow

answered May 26, 2010 at 13:30



[polygenelubricants](#)

384k 129 568 625

1 Upvote just for the regular-expressions.info link. Wonderful site for learning regular expressions and for reference. – [Freiheit](#) May 26, 2010 at 14:19



No, * will match zero-or-more characters. You should use +, which matches one-or-more instead.

9

This expression might work better for you: `[A-Z]+123`



Share Improve this answer Follow

edited Jul 27, 2015 at 9:54

answered May 26, 2010 at 12:33



Quill

2,755

1

35

45



Huusom

5,912

1

18

15

- 1 Upvote here. The OP didn't specify, but it seems correct to add that the pattern will match any character including things like ###123, 123123, %\$#123 which the OP may not want. The character class @Huusom uses above will all the OP to use only uppercase alphabetic characters which may have been the intent. – [techdude Jan 26, 2015 at 22:43](#)



Try the regex `.{3,}`. This will match all characters except a new line.

6

Share Improve this answer Follow

edited Mar 2, 2020 at 20:16



Noah

919

8

20

answered Mar 22, 2017 at 4:59



Ravi Shekhar

2,935

4

21

20



Specific Solution to the example problem:-

5

Try `[A-Z]*123$` will match `123`, `AAA123`, `ASDFRRF123`. In case you need at least a character before `123` use `[A-Z]+123$`.



General Solution to the question (How to match "any character" in the regular expression):

1. If you are looking for anything including whitespace you can try `[\w|\W]{min_char_to_match,}`.
2. If you are trying to match anything except whitespace you can try `[\S]{min_char_to_match,}`.

Share Improve this answer Follow

edited Sep 12, 2019 at 5:45



Jonathan Johansen

363

4

14

answered Jan 24, 2019 at 20:56



Akash Kumar Seth

1,701

1

19

22

Sorry, but it doesn't match `123`, check this [regex demo](#). – [Amine KOUIS Jan 11, 2024 at 22:15](#)

▲ [^] should match any character, including newline. [^ CHARS] matches all characters except for those in CHARS. If CHARS is empty, it matches all characters.

3

JavaScript example:



```
/a[^]*Z/.test("abcxyz \0\r\n\t012789ABCXYZ") // Returns 'true'.
```



Share Improve this answer Follow

edited Nov 7, 2019 at 18:23

answered Nov 5, 2019 at 11:44



Anonymous

107 1 2

Would you mind adding some code to let us know what you have tried? – Jennis Vaishnav Nov 5, 2019 at 12:03



I like the following:

1

[!~]



This matches all char codes including special characters and the normal A-Z, a-z, 0-9

https://www.w3schools.com/charsets/ref_html_ascii.aspE.g. `faker.internet.password(20, false, /[!~]/)`Will generate a password like this: `0+>8*nZ*-mB7Ybbx,b>`

Share Improve this answer Follow

answered Oct 9, 2022 at 14:40



Andy

127 3 11



Just for reference, regular expressions may behave differently in different languages and different modes.

- Absolute *anything*: `[\s\S]` . By joining a set with its *complement* set, you get a *universe* set.

0

- Literal anything: the dot `.`, which usually matches any character but the invisible, such as spaces, tabs and returns. Of course, somehow it can match absolute anything too.

[Share](#) [Improve this answer](#) [Follow](#)

answered Jul 15, 2023 at 17:12

[oceanblue](#)

45 8



-5

I work this Not always dot is means any char. Exception when single line mode. `\p{all}` should be



```
String value = "|°~<>!\\"#$%&/()=?'\\";
String expression = "[a-zA-Z0-9\\p{all}]{0,50}";
if(value.matches(expression)){
    System.out.println("true");
} else {
    System.out.println("false");
}
```

[Share](#) [Improve this answer](#) [Follow](#)

edited Mar 2, 2020 at 19:18

[Noah](#)

919 8 20

[Abrahan Gonzalez](#)

Abra 27 2

I don't understand what's wrong with this answer (except for really bad English). Actually, this is the most relevant answer to the question formulated as "any character" and it helped me the best with my problem. – [Roman Horváth Jul 25, 2021 at 7:44](#)

Start asking to get answers

Find the answer to your question by asking.

[Ask question](#)

Explore related questions

[regex](#)

See similar questions with these tags.