The results are in! See what nearly 90,000 developers picked as their most loved, dreaded, and desired coding languages and more in the 2019 Developer Survey.

Use Python's string.replace vs re.sub

Ask Question

51

For Python 2.5, 2.6, should I be using `string.replace` or `re.sub` for basic text replacements?

15

In PHP, this was explicitly stated but I can't find a similar note for Python.

python regex

edited Jan 8 '15 at 19:03

the Tin Man
136k 27 174 257

asked Apr 14 '11 at 19:58

wag2639
971 3 16 29

2 Avoid regex at all costs! ...Until absolutely necessary... – jathanism Apr 14 '11 at 20:03

17 @jathanism: I respectfully disagree. I avoided regex for decades until I finally took the time to sit down and and actually *learn* them. Now I can't live

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

for many day-to-day tasks and should be a familiar tool in every programmer's toolbox. – ridgerunner Apr 14 '11 at 20:27

5 @ridgerunner: Agreed, but it is also important to know *when* to use them. For simple string manipulations such as this, regular expressions are over the top. My rule of thumb is that if you can do it with the built-in string functions (`split()` , `replace()` , `find()` et al) without needing multiple status variables, complicated slicing etc you should. If it starts getting complex, then you move alternate tools such as regular expressions. – Blair Apr 14 '11 at 23:32

5 Oh, and a general comment on the speed of regular expressions: it depends on the context. In a script you run occasionally with a few regular expressions, you won't notice the overhead. On the other hand, in a script which does some intensive/high volume processing you

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

using regular expressions lots. This is where profiling is important to determine where the bottleneck is (and I suppose I should trot out the *premature optimisation is the root of all evil* line at this point too). – Blair Apr 14 '11 at 23:37

2 @Blair: I wholeheartedly agree. But many seem to be averse to regex because they find them "difficult" and this is simply because they have not taken the time to learn tem beyond a superficial level. Yes, if a simple string replace solves the problem, then by all means use that, (which is also very likely the fastest solution as well). But I see way too many convoluted, complex string manipulation solutions to problems which are easily solved with a single, *well crafted* regex. – ridgerunner Apr 15 '11 at 0:41

4 Answers

As long as you can make do with

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

pitfalls of regular expressions (like escaping), and is generally faster.

nswered Apr 14 '11 at 19:59

Sven Marnach
360k 80 758 702

1 If you are going to many times substitute, the replace is more fast than sub – Danyun Liu Jun 22 '12 at 6:59

@SvenMarnach Does this still apply to Python 2.7? – NumenorForLife Jun 17 '15 at 16:15

2 @jsc123: This advice is about avoiding pitfalls and unnecessary complexity; so yes, it applies to any Python version. :) – Sven Marnach Jun 17 '15 at 16:49

38

`str.replace()` should be used whenever it's possible to. It's more explicit, simpler, and faster.

```
In [1]: import re

In [2]: text = """
```

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

```
In [3]: timeit te
1000000 loops, be

In [4]: timeit re
100000 loops, bes
```

nswered Apr 14 '11 at 20:13

chmullig
9,790 5 26 49

2 Out of curiosity, how were you executing `timeit` in your example output? Is that something special to iPython allowing you to use that syntax? (Oh, and +1!) – jathanism Apr 15 '11 at 14:31

2 Yup, ipython includes it magically. scienceoss.com/ … – chmullig Apr 15 '11 at 14:59

1 Unsure if this is a typo or I'm missing something, but your str.replace() run has 10x the number of loops as the regex run. – BoltzmannBrain Jan 8 '16 at 1:01

@alavin89 IPython chooses a "fitting value" for the iteration count if one is not specified (ipython.org/ipython-doc/3/interactive/magics.html#magic-timeit). It's possible that the value it chooses scales based on the time it takes to execute the snippet some

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

loop, the difference in loop counts does not matter significantly. – NasaGeek Jun 7 '16 at 20:48

What if you had chained multiple `replace` vs a single regex. At some point a single regex replace should be faster than having N chained `replace` 's on a string, no? – radtek Oct 6 '17 at 21:15

28

String manipulation is **usually** preferable to regex when you can figure out how to adapt it. Regex is incredibly powerful, but it's **usually** slower, and **usually** harder to write, debug, and maintain.

That being said, notice the amount of "usually" in the above paragraph! It's possible (and I've seen it done) to write a zillion lines of string manipulation for something you could've done with a 20-character regex. It's also possible to waste valuable time using "efficient" string functions on tasks a good regex

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

maintainability: Regex can be horribly complex, but sometimes a regex will be simpler and easier to read than a giant block of procedural code.

Regex is fantastic for its intended purpose: searching for highly-variable needles in highly-variable haystacks. Think of it as a precision torque wrench: It's the perfect tool for a specific set of jobs, but it makes a lousy hammer.

Some guidelines you should follow when you aren't sure what to use:

- **Is the pattern you're looking for highly static?** For example, do you want to split a string on every comma, pipe, or tab?
- **Is resource efficiency**

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

> **developer time?** What are your priorities? Remember: Hardware is cheap, programmers are expensive.
>
> - **Are you working with HTML, XML, or other context-free grammars?** Don't forget that regex has limitations.
> - And my #1 rule of thumb: **If you work on the problem for 5 minutes, can you rough out an idea for a non-regex approach?**
>
> If the answer to any of these questions is "yes", you probably want string manipulation. Otherwise, consider regex.

dited May 23 '17 at 11:54

Community ♦
1 1

nswered Apr 14 '11 at 22:19

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

9

Another thing to consider is that if you're doing rather complex replacements, str.translate() might be what you're looking for.

ıswered Apr 14 '11 at 20:07

jathanism

25.6k 6 58 82

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.