Setting options from environment variables when using argparse

Asked 11 years, 3 months ago Modified 3 years, 1 month ago Viewed 42k times

78

I have a script which has certain options that can either be passed on the command line, or from environment variables. The CLI should take precedence if both are present, and an error occur if neither are set.

I could check that the option is assigned after parsing, but I prefer to let argparse to do the heavy lifting and be responsible for displaying the usage statement if parsing fails.

I have come up with a couple of alternative approaches to this (which I will post below as answers so they can be discussed separately) but they feel pretty kludgey to me and I think that I am missing something.

Is there an accepted "best" way of doing this?

(Edit to make the desired behaviour clear when both the CLI option and environment variable are unset)

python argparse

Share Improve this question Follow

edited May 11, 2012 at 12:40

asked May 11, 2012 at 12:13
Russell Heilling
1,829 ● 1 ● 12 ● 10

12 Answers

Sorted by: Highest score (default)

91

You can set the `default=` of the argument to a [`.get()`] of [`os.environ`] with the environment variable you want to grab.

You can also pass a 2nd argument in the `.get()` call, which is the default value if `.get()` doesn't find an environment variable by that name (by default `.get()` returns `None` in that case).

```
import argparse
import os
```

```
parser = argparse.ArgumentParser(description='test')
parser.add_argument('--url', default=os.environ.get('URL'))

args = parser.parse_args()
if not args.url:
    exit(parser.print_usage())
```

Share Improve this answer Follow

edited Apr 7, 2020 at 16:22
Boris Verkhovskiy
14.7k ● 11 ● 100 ● 101

answered May 11, 2012 at 12:32
Christian Witts
11.4k ● 1 ● 33 ● 46

3 None is the default for .get(), so doesn't need to be explicitly stated like this. I should probably have been more clear in the question - the option needs to be in at least one of the environment variable or the CLI. If you set the default like this, then args.url may well end up as None, which is what I want to avoid... – Russell Heilling May 11, 2012 at 12:37

1 Ah, I see what you are looking for. I would honestly just use what I wrote and after parsing the args just check `if not args.url: exit(parser.print_usage())` and exit. – Christian Witts May 11, 2012 at 12:51

1 A nice quick way of handling it. I have packaged up my own action handler because I use this pattern a lot, but this would certainly be my fallback for a quick and simple script. – Russell Heilling May 14, 2012 at 9:14

In my eyes this looks like the more expressive way. If there is not a very specific reason to get more complex, this is just elegant and short. Therefore I think this should be the "correct" answer, although @RussellHeilling also proposed a good alternative. – erikbstack Nov 11, 2013 at 13:42

37 The way this answer proposes is bad UX. "default" means "default for the application", not "default in the current environment". People may look at --help, see a particular default, assume it's default forever, go to another machine/session, get kaboom. – pfalcon Jan 4, 2018 at 22:35

82

I use this pattern frequently enough that I have packaged a simple action class to handle it:

```
import argparse
import os

class EnvDefault(argparse.Action):
    def __init__(self, envvar, required=True, default=None, **kwargs):
        if not default and envvar:
            if envvar in os.environ:
                default = os.environ[envvar]
        if required and default:
            required = False
        super(EnvDefault, self).__init__(default=default, required=required,
                                         **kwargs)
    def __call__(self, parser, namespace, values, option_string=None):
        setattr(namespace, self.dest, values)
```

I can then call this from my code with:

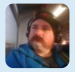
```
import argparse
from envdefault import EnvDefault

parser=argparse.ArgumentParser()
parser.add_argument(
    "-u", "--url", action=EnvDefault, envvar='URL',
    help="Specify the URL to process (can also be specified using URL environment
variable)")
args=parser.parse_args()
```

Share Improve this answer Follow

edited Feb 21, 2014 at 11:09

answered May 11, 2012 at 12:17



Russell Heilling
1,829 ● 1 ● 12 ● 10

1 Should default this be edited to look up in os.environ? "if envvar in os.environ: default = envvar" --> "if envvar in os.environ: default = os.environ[envvar]" – spazm Apr 4, 2013 at 19:48

3 Why do you only use the value specified by the envvar if there's no default? Shouldn't it override the default, as the caller has explicitly provided a value? – Michael Nelson Jan 15, 2015 at 5:29

3 This solution is great. However, it has a side-effect : depending on your environment (whether or not the env var is set), the usage text may differ. – Eric Citaire Mar 1, 2017 at 14:38

This is a great answer, as you can easily add in additional functionality to this class as your desires change. I use this solution all the time, and it works great. – Rezkin Mar 22, 2021 at 1:13