Slow python imports

Tom Rochette <tom.rochette@coreteks.org>

January 22, 2026 — a3694a49

1 Problem

Some imports in my python code are slow. How can I figure out which ones are the source of slowness?

2 Solution

Python offers a really useful functionality you can use that will list how long each import took. By passing the `-X importtime` argument to your python command when you execute your script it will print out both the cumulative time (including nested imports) and self time (excluding nested imports) of each import.

```
python -X importtime your-script.py
```

Running `python -X importtime my-script.py` on an empty script returns the following (on Windows 7, Python 3.7.5)

```
import time: self [us] | cumulative | imported package
import time:        52 |         52 | zipimport
import time:       367 |        367 | _frozen_importlib_external
import time:        55 |         55 |     _codecs
import time:       530 |        585 |   codecs
import time:       520 |        520 |   encodings.aliases
import time:      1107 |       2210 | encodings
import time:       328 |        328 | encodings.utf_8
import time:        39 |         39 | _signal
import time:       357 |        357 | encodings.latin_1
import time:        34 |         34 |     _abc
import time:       312 |        345 |   abc
import time:       474 |        819 | io
import time:       113 |        113 |       _stat
import time:       264 |        377 |     stat
import time:       269 |        269 |       genericpath
import time:       794 |       1062 |     ntpath
import time:       871 |        871 |     _collections_abc
import time:       915 |       3223 |   os
import time:       490 |        490 |   _sitebuiltins
import time:        57 |         57 |     _locale
import time:       934 |        991 |   _bootlocale
import time:       421 |        421 |   encodings.cp1252
import time:       472 |        472 |   types
import time:       394 |        394 |       warnings
import time:       440 |        834 |     importlib
import time:       263 |        263 |       importlib.machinery
import time:       554 |        816 |     importlib.abc
```

```
import time:        64 |         64 |           _operator
import time:       792 |        856 |         operator
import time:       343 |        343 |         keyword
import time:        43 |         43 |           _heapq
import time:       405 |        447 |         heapq
import time:        85 |         85 |         itertools
import time:       328 |        328 |         reprlib
import time:        69 |         69 |         _collections
import time:      1460 |       3585 |       collections
import time:        44 |         44 |         _functools
import time:       596 |        640 |       functools
import time:       784 |       5008 |     contextlib
import time:       651 |       7308 |   importlib.util
import time:      1095 |       1095 |   pywin32_bootstrap
import time:       231 |        231 |   sitecustomize
import time:     10744 |      24972 | site
```

For a script with a simple `import argparse`, I get the following output:

```
import time: self [us] | cumulative | imported package
import time:        70 |         70 | zipimport
import time:       341 |        341 | _frozen_importlib_external
import time:        54 |         54 |     _codecs
import time:       457 |        511 |   codecs
import time:       456 |        456 |   encodings.aliases
import time:      1030 |       1997 | encodings
import time:       215 |        215 | encodings.utf_8
import time:        38 |         38 | _signal
import time:       268 |        268 | encodings.latin_1
import time:        33 |         33 |     _abc
import time:       398 |        431 |   abc
import time:       311 |        741 | io
import time:        87 |         87 |       _stat
import time:       271 |        357 |     stat
import time:       196 |        196 |       genericpath
import time:       416 |        612 |     ntpath
import time:       714 |        714 |     _collections_abc
import time:       610 |       2292 |   os
import time:       229 |        229 |   _sitebuiltins
import time:        48 |         48 |     _locale
import time:       246 |        293 |   _bootlocale
import time:       217 |        217 |   encodings.cp1252
import time:       488 |        488 |   types
import time:       279 |        279 |       warnings
import time:       461 |        740 |     importlib
import time:       269 |        269 |       importlib.machinery
import time:       557 |        825 |     importlib.abc
import time:        63 |         63 |           _operator
import time:       808 |        871 |         operator
import time:       336 |        336 |         keyword
import time:        41 |         41 |           _heapq
import time:       336 |        376 |         heapq
import time:        69 |         69 |         itertools
import time:       341 |        341 |         reprlib
import time:        70 |         70 |         _collections
```

```
import time:      1136 |       3197 |       collections
import time:        69 |         69 |         _functools
import time:       642 |        710 |       functools
import time:       801 |       4708 |     contextlib
import time:       688 |       6959 |   importlib.util
import time:       934 |        934 |   pywin32_bootstrap
import time:       224 |        224 |   sitecustomize
import time:      9323 |      20954 | site
import time:       698 |        698 |     enum
import time:        55 |         55 |       _sre
import time:       417 |        417 |         sre_constants
import time:       372 |        789 |       sre_parse
import time:       443 |       1286 |     sre_compile
import time:       323 |        323 |     copyreg
import time:       716 |       3021 |   re
import time:       725 |        725 |     locale
import time:       948 |       1673 |   gettext
import time:       986 |       5678 | argparse
```

The package are listed in order that they are resolved. In `argparse` case, `os` and `sys` were already loaded, so it first loads `re`, then `gettext`. Once both are loaded, `argparse` has finished loading.

The way the cumulative column is computed is to take all the prior self that are a level higher than the package you're looking at. For example (if we take the io package):

```
import time:        33 |         33 |     _abc
import time:       398 |        431 |   abc
import time:       311 |        741 | io
```

$311 + 398 + 33 = 742$

We can see here that the numbers are not necessarily equal to one another, this might be due to precision used to do the computation while the rendering of numbers is rounded.

Note that the load time of a package may be different depending on which script you load because dependencies of the package may have already been loaded in some cases, while in others it may have to load them.

Looking at text might be your thing, but if you're more visual, there's a tool called tuna which will consume this output and create an icicle plot you can look at to find which imports are the slowest/longest.

3 References

- Python documentation on -X
- https://github.com/nschloe/tuna