














< Previous	 		 	 					Next >
------------	---	---	---	--	---	---	---	---	--------

Part 6: The Lazy Programmer's Guide to Debugging and Logging

 [Bookmark this page](#)

PySnooper: easy debugging. Loguru: easy logging

Whether we admit it or not, too many developers avoid debugging, and sprinkle print statements throughout their code. And in small doses, that's probably OK.

I am a lazy developer. I want to do the minimum necessary to deliver working code to people who need it. And so I like tools that make my job easy and are intuitive.

Here we are going to look at two tools that help me to be lazy and meet my goals.

PySnooper

PySnooper lives at <https://github.com/cool-RR/pysnooper>

You can install it (in your virtual environment, right) using:

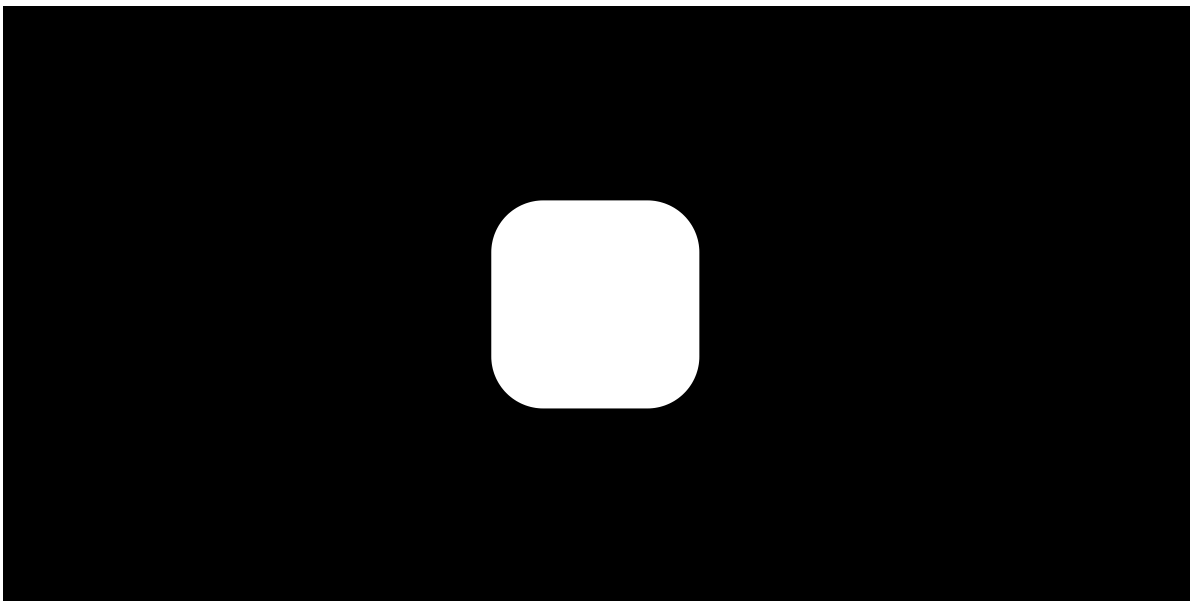
```
python -m pip install pysnooper
```

PySnooper allows you to selectively see, or 'snoop', upon what your Python program is doing. It helps you answer questions like "which statements are executing", or "which if statements are passing and failing", or even "how did that variable get that value".

Now you can do that sort of things in several ways. But nothing makes it easier than PySnooper.

Pysnooper 1

[Start of transcript. Skip to the end.](#)



[MUSIC PLAYING]
Let's take a look at some tools that helped
make your Python development much more simpler than it would be otherwise.
First, what we're going to look at helps us
to understand the way that a program works

Video

[Download video file](#)

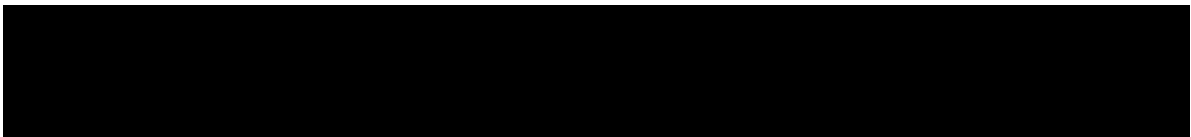
Transcripts

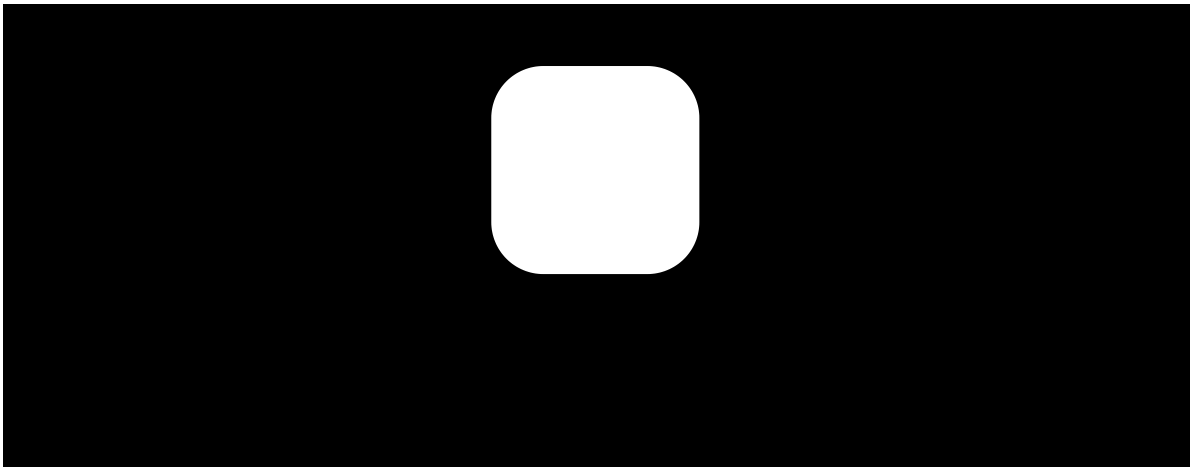
[Download SubRip \(.srt\) file](#)

[Download Text \(.txt\) file](#)

Pysnooper 2

[Start of transcript. Skip to the end.](#)





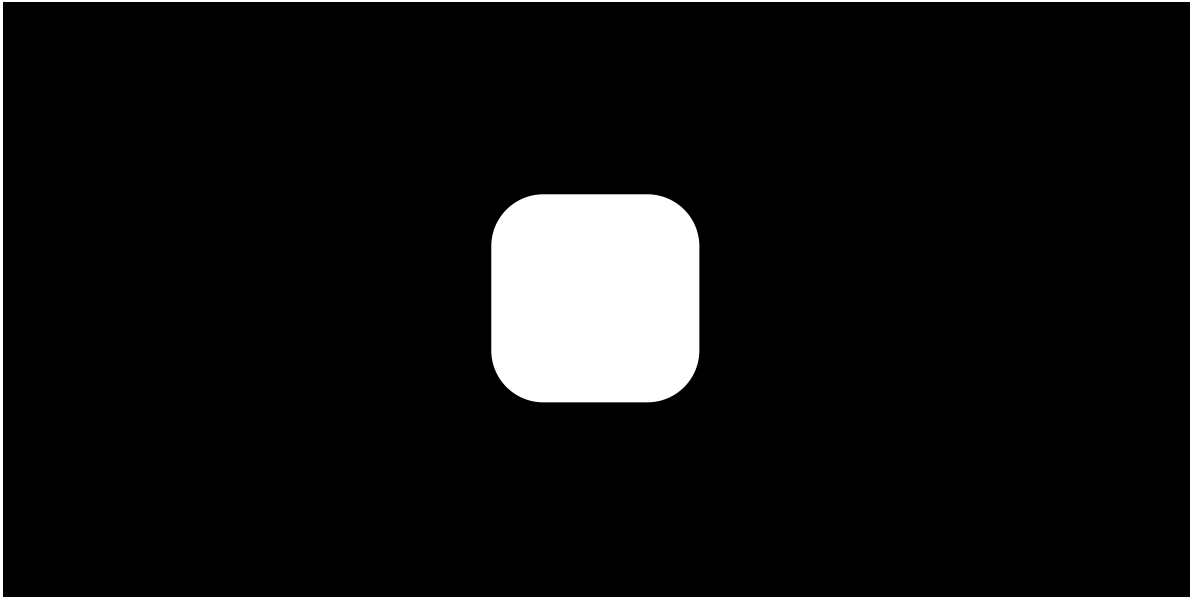
It's that time to see PySnooper in action.
So here's a module that I've loaded that I'm
pretending that I've come to for the first time,
and I don't particularly understand what it does.
What I want to do is, I want to investigate what's going on here

Video
[Download video file](#)

Transcripts
[Download SubRip \(.srt\) file](#)
[Download Text \(.txt\) file](#)

Pysnooper 3

[Start of transcript. Skip to the end.](#)



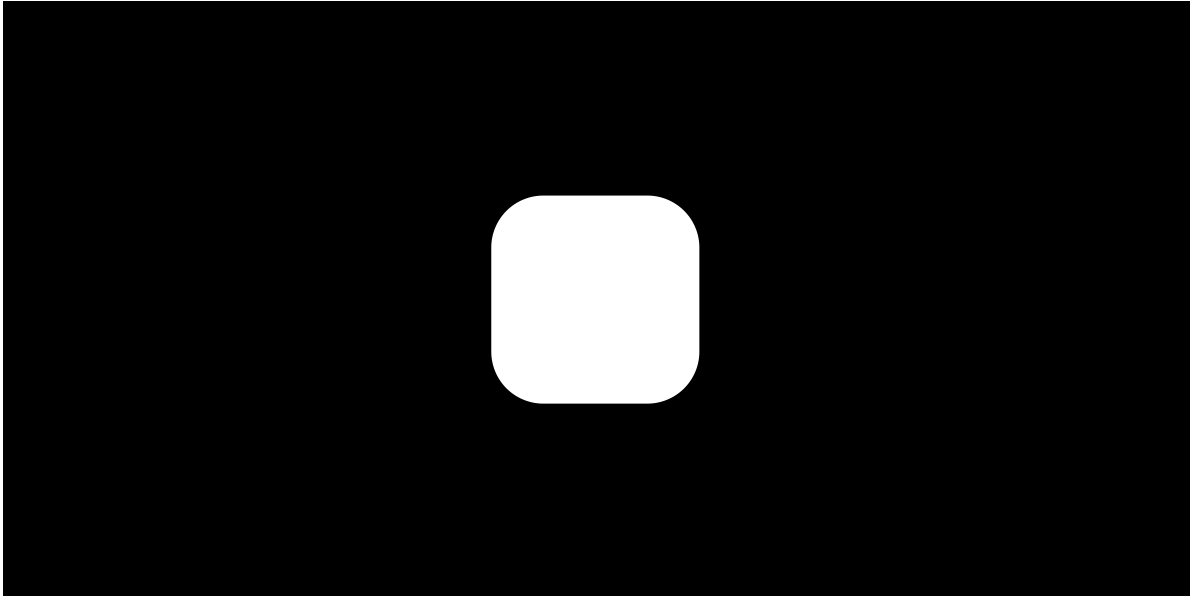
[MUSIC PLAYING]
Now let's do something a little bit more ambitious with PySnooper,
especially now we've got the idea of what it's doing.
This piece is obvious, but let's say we wanted to see more detail about what's going on behind the scenes.
In the parentheses of this-- of the Spoon

Video
[Download video file](#)

Transcripts
[Download SubRip \(.srt\) file](#)
[Download Text \(.txt\) file](#)

Pysnooper 4

[Start of transcript. Skip to the end.](#)



[MUSIC PLAYING]
Now finally, let me show you how it's possible to localize
what you snooped on in your program.
So in this case, we snooped on main to see the whole of the program

see the whole of the program
and how it was executed.
But we can choose to snoop on specific
functions if we want to.

Video

[Download video file](#)

Transcripts

[Download SubRip \(.srt\) file](#)

[Download Text \(.txt\) file](#)

Say you have a function called `calculate_roi()` that performs a very complex calculation. And it isn't working right. All you do is:

```
python -m pip install pysnopper
```

 if you haven't already.

Then you must `import pysnopper`.

And then, immediately before `def calculate_roi()` you add `@pysnopper.snoop()`. So the code looks like this:

```
@pysnopper.snoop()
def calculate_roi()
    # blah blah blah
```

And then run. Try it on one of your functions and look at the output. And also, look at the examples at <https://github.com/cool-RR/pysnopper>

Notice how you can see the code that was executed, the line numbers and the values of variables. And it's so easy to add and read. Makes me proud to be lazy!

Loguru

In a similar vein to `pyscaffold`, `loguru` makes logging very easy. You'll find `loguru` at <https://github.com/Delgan/loguru>

What's really good about `loguru` is that it is a friendlier front end to the Python standard logging facilities. So you can easily drop back to standard logging if `loguru` does not meet all your needs.

To get started it's really easy.

```
pip install loguru
```

and then add the following to the top of your py file:

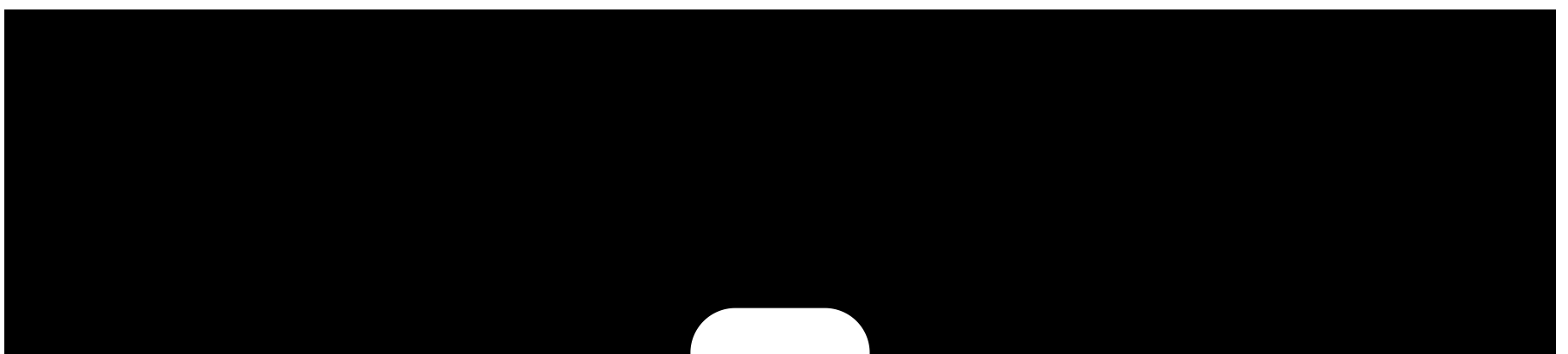
```
from loguru import logger
```

Now, all you need to do to do logging is:

```
logger.debug("Message to go in the log")
```

That's it. It just works. The documentation on the site is great; take a look at the examples too at <https://github.com/Delgan/loguru>


Loguru





© 2024 University of Washington | Seattle, WA. All rights reserved.

[Help Center](#) [Contact Us](#) [Privacy](#) [Terms](#)

Built on [OPENedX](#) by RACCOONGANG 

edX, Open edX and the edX and Open edX logos are trademarks or registered trademarks of edX Inc.