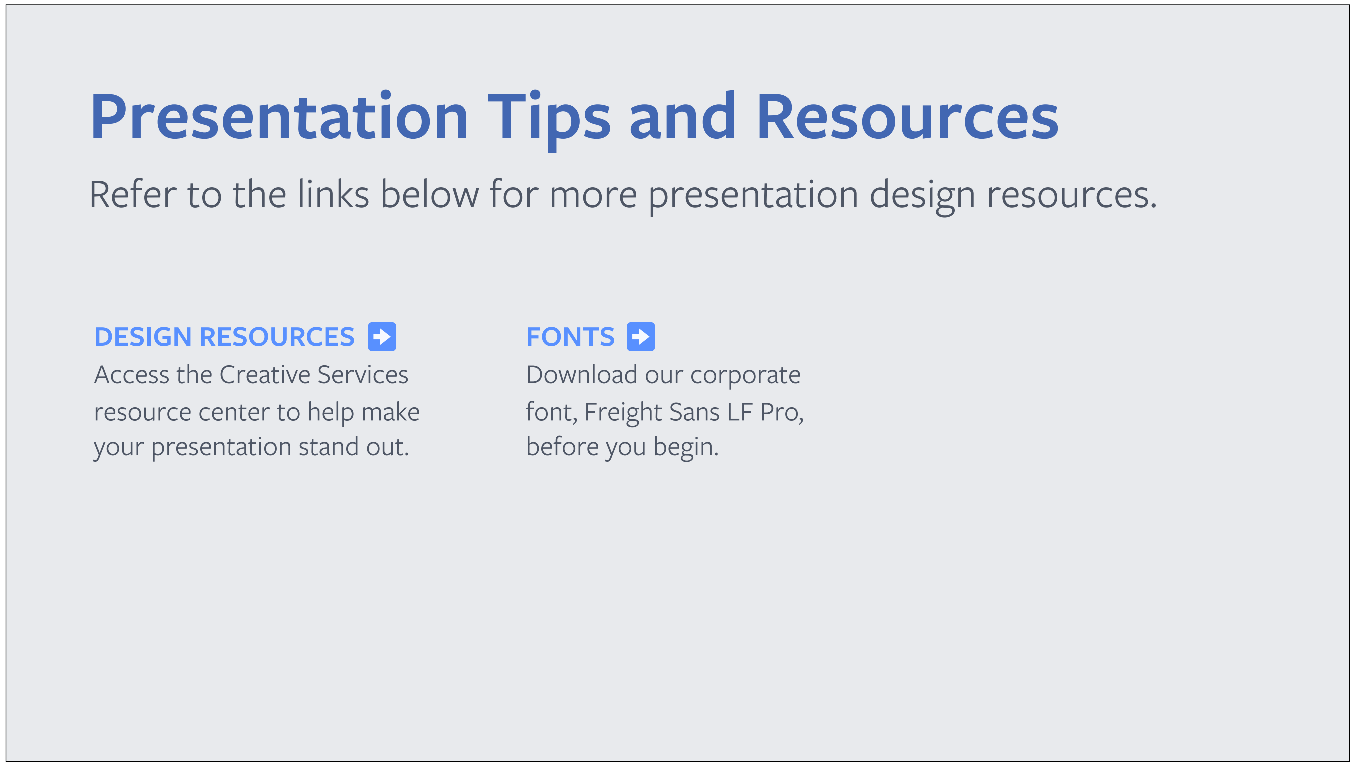


Presentation Tips and Resources

Refer to the links below for more presentation design resources.

DESIGN RESOURCES ➔

Access the Creative Services resource center to help make your presentation stand out.

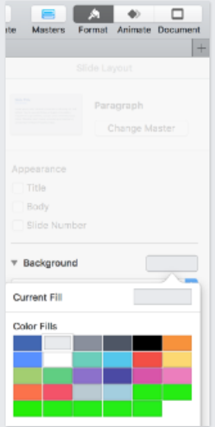
FONTS ➔

Download our corporate font, Freight Sans LF Pro, before you begin.

Template and Brand Colors

TEMPLATE COLORS

Use the template's preset colors found within the Format sidebar for backgrounds, shapes and text.



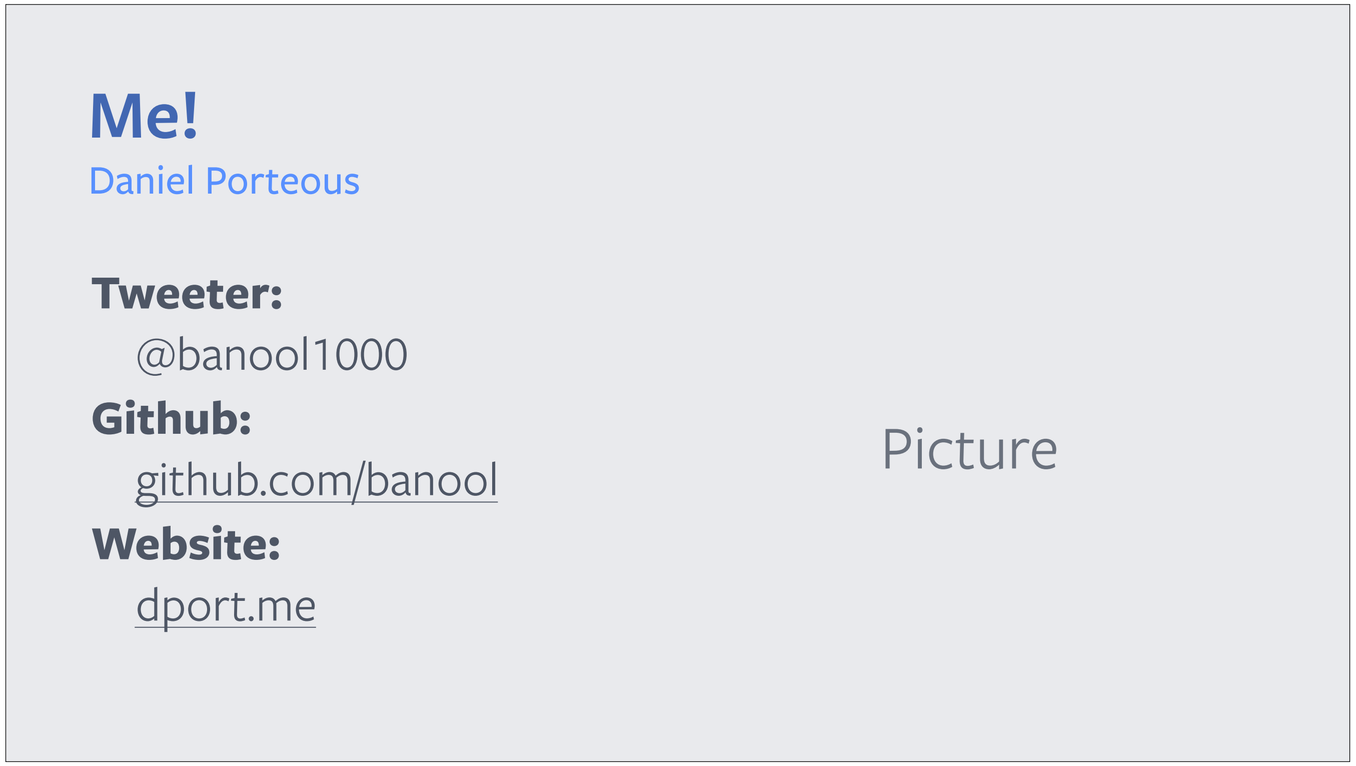
BRAND COLORS

Refer to the link of Facebook's Brand Colors as a guide to selecting colors.

facebook

Context Managers:
You Can Write Your Own!

Daniel Porteous
Production Engineer



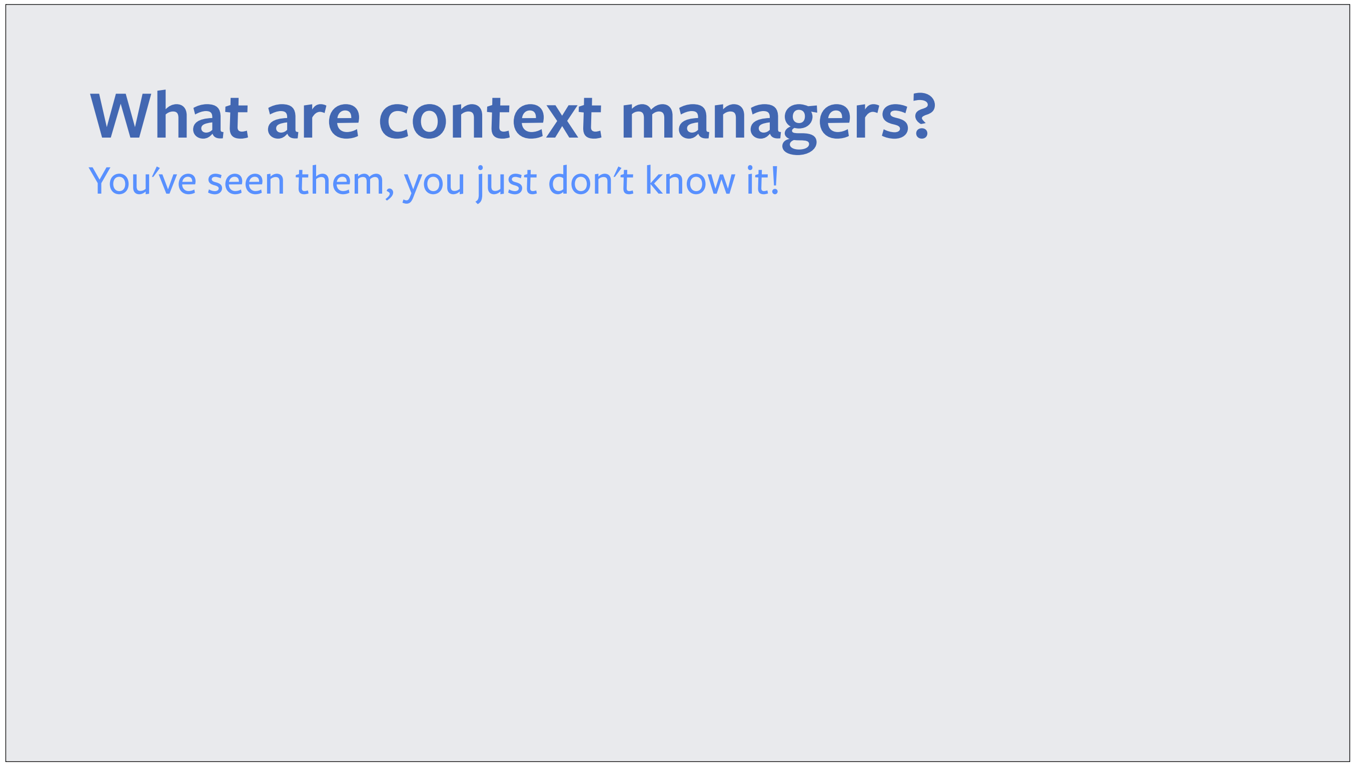
Me!

Daniel Porteous

Tweeter:
@banool1000
Github:
github.com/banool
Website:
dport.me

Picture

Self, PE at fb. From Townsville but lived in Melbourne for uni. I live in the bay area, CA now. Always found talks on core python features really great, super excited to give my first talk on one of my fave python features: context managers.



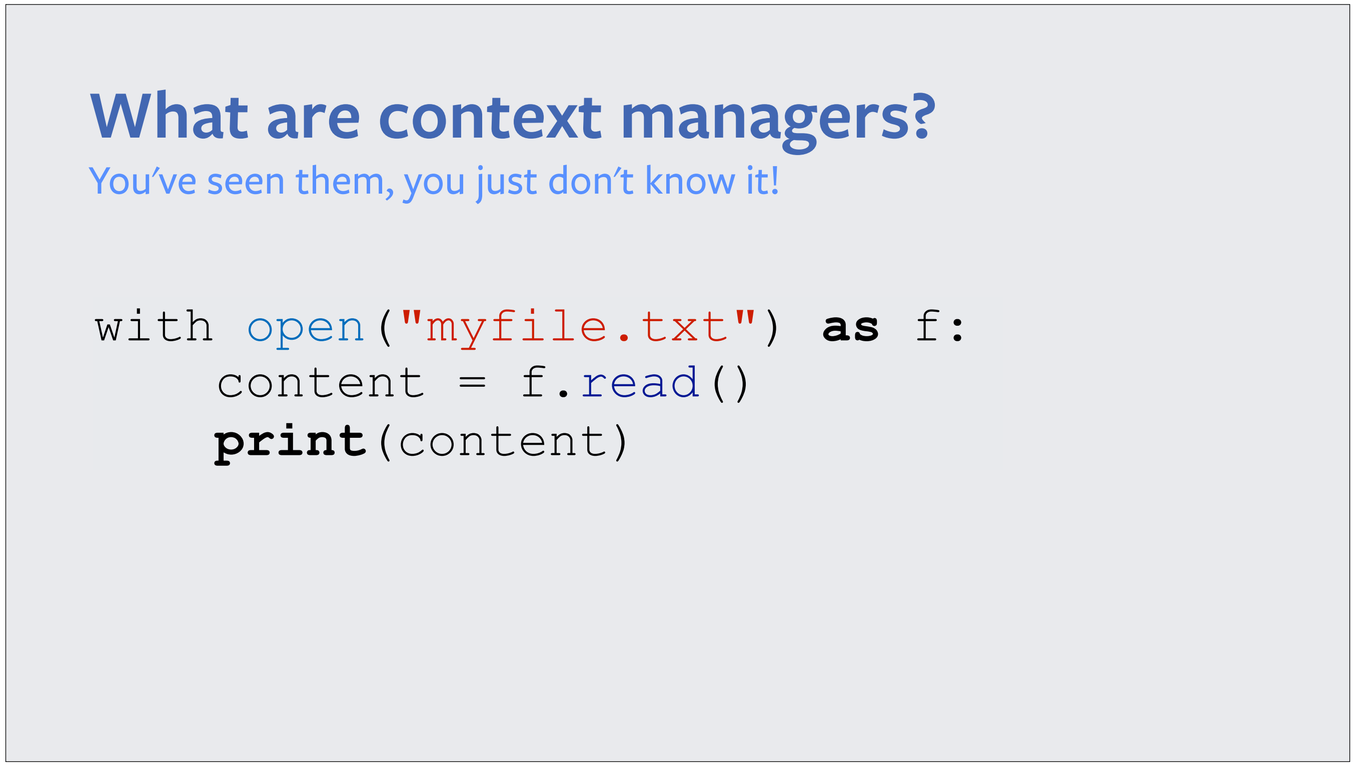
What are context managers?

You’ve seen them, you just don’t know it!

Lots of people I talked to didn't know context managers by name, but everyone has used them.

Anytime you see the with keyword (maybe as), you're looking at a context manager.

Here is the classic, most peoples' first introduction to context managers:



What are context managers?

You've seen them, you just don't know it!

```python
with open("myfile.txt") as f:
    content = f.read()
    print(content)
```

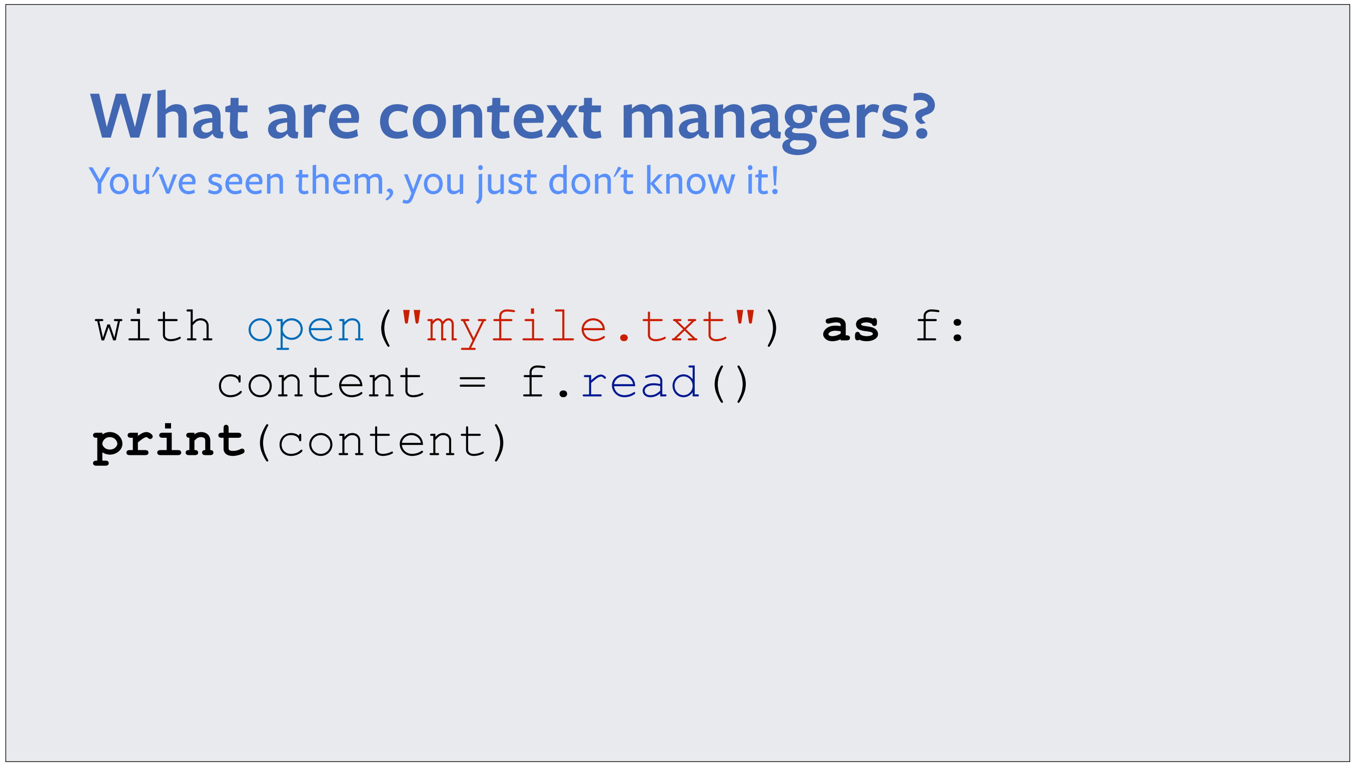
HAVE LINE NUMBERS

Simple example opens a file for you and gives you a handle, `f` as reference.

After `exit`ing with, f is closed.

Context managers, manager your context, in this case a file.

Note on scope: next slide

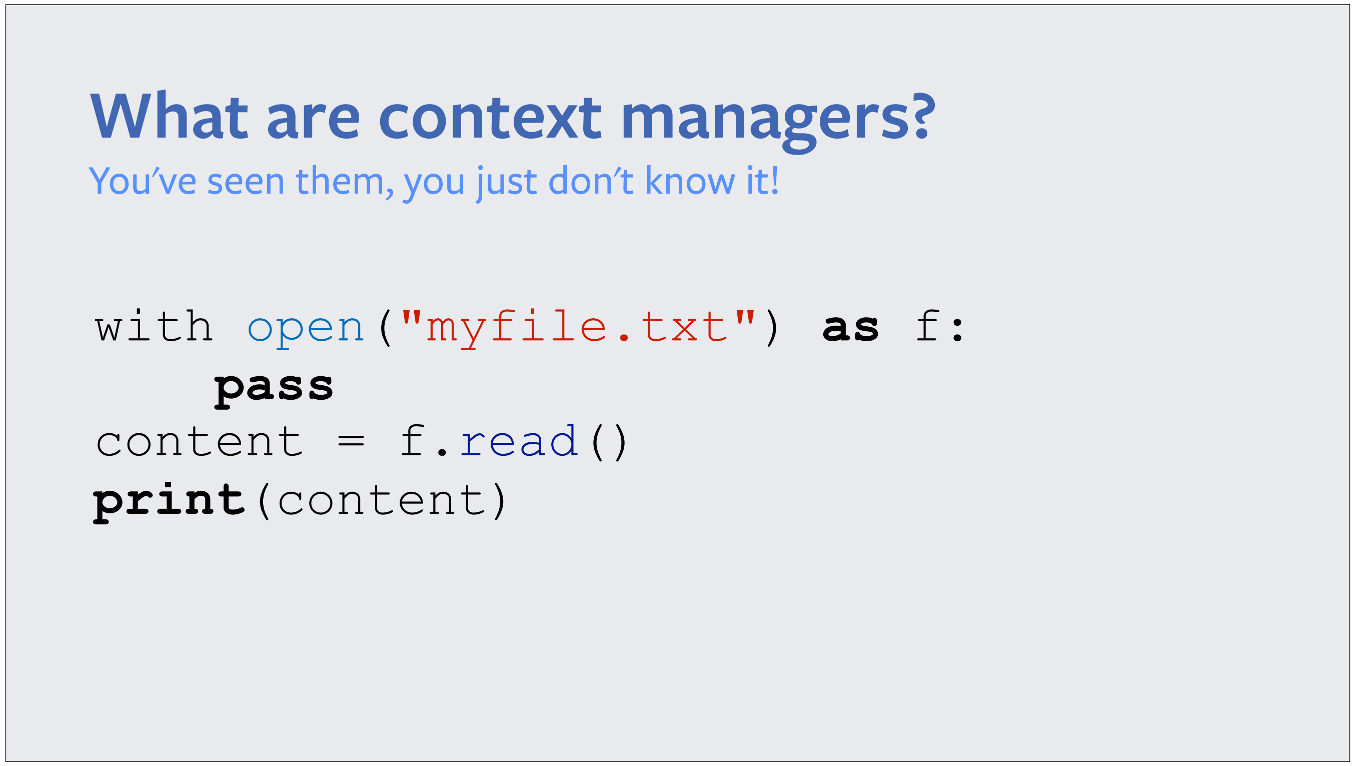


What are context managers?

You've seen them, you just don't know it!

```
with open("myfile.txt") as f:
    content = f.read()
print(content)
```

After a context manager, variables defined in it still exist outside of it. Idk if that seems obvious, but it didn't to me so heads up.

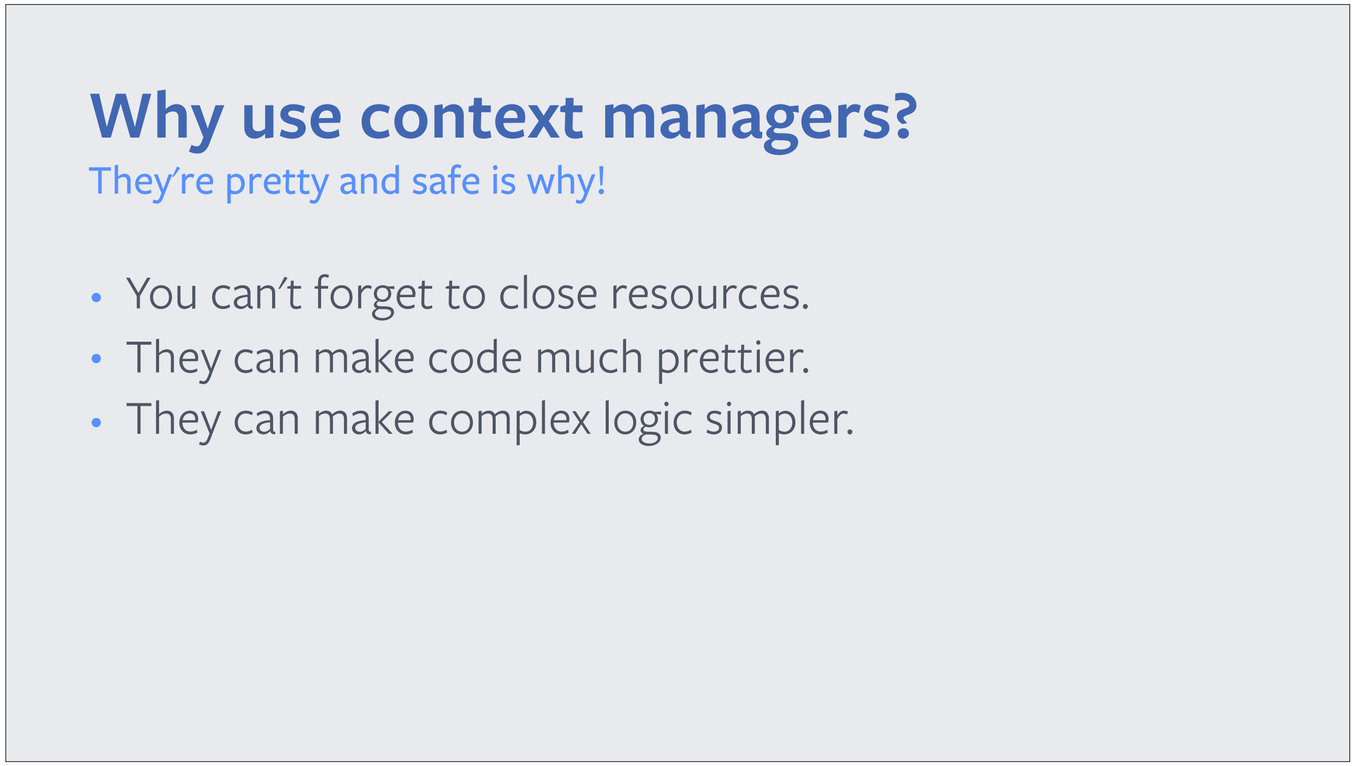


What are context managers?

You’ve seen them, you just don’t know it!

```python
with open("myfile.txt") as f:
    pass
content = f.read()
print(content)
```

The thing after the `as` still exists too, but it will be closed after the block.



Why use context managers?

They’re pretty and safe is why!

- You can’t forget to close resources.
- They can make code much prettier.
- They can make complex logic simpler.

talk talk talk about points

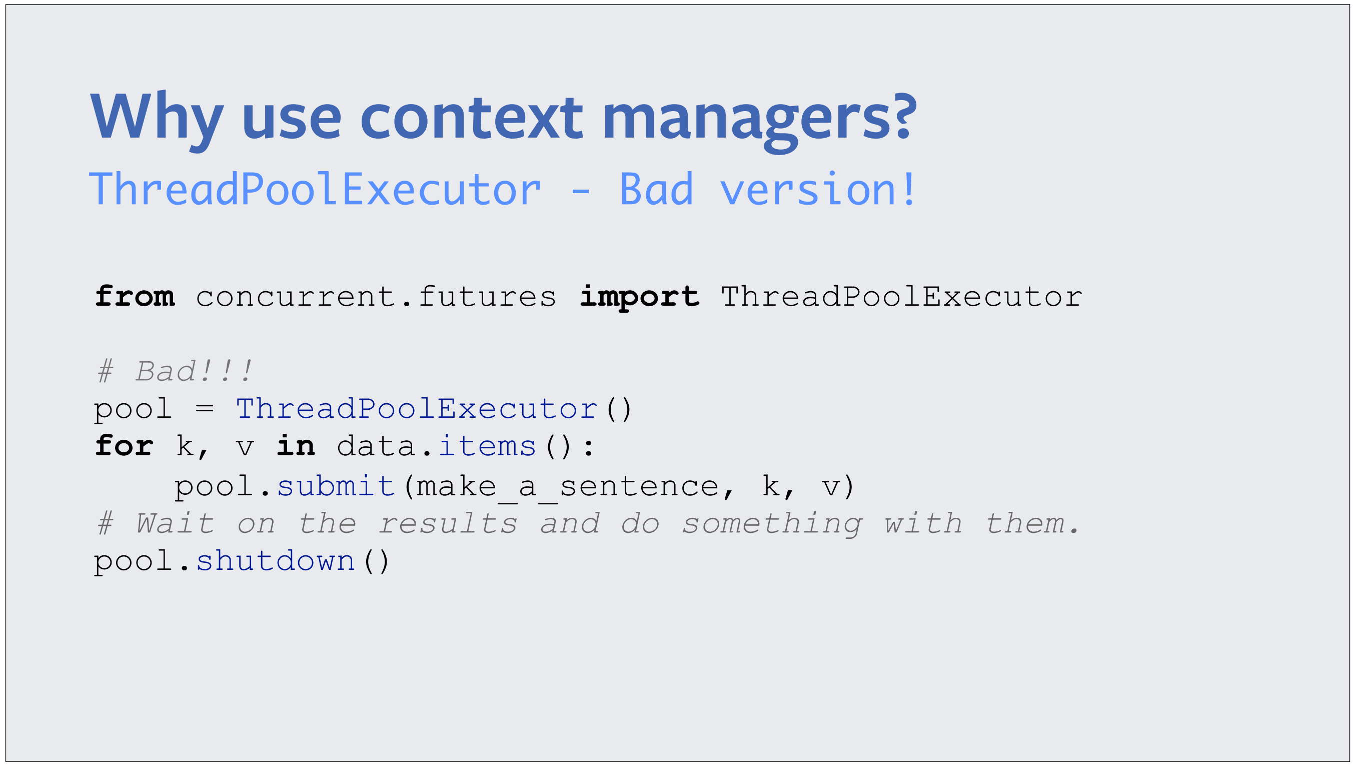
Let's look at some more examples of context managers.

Why use context managers?

contextlib.suppress

```
def kill_process(pid):
    # Kill a process, ignore if it can't be found.
    with contextlib.suppress(ProcessLookupError):
        os.kill(pid, signal.SIGKILL)

# Nicer than:
def kill_process(pid):
    try:
        os.kill(pid, signal.SIGKILL)
    except ProcessLookupError:
        pass
```



Why use context managers?

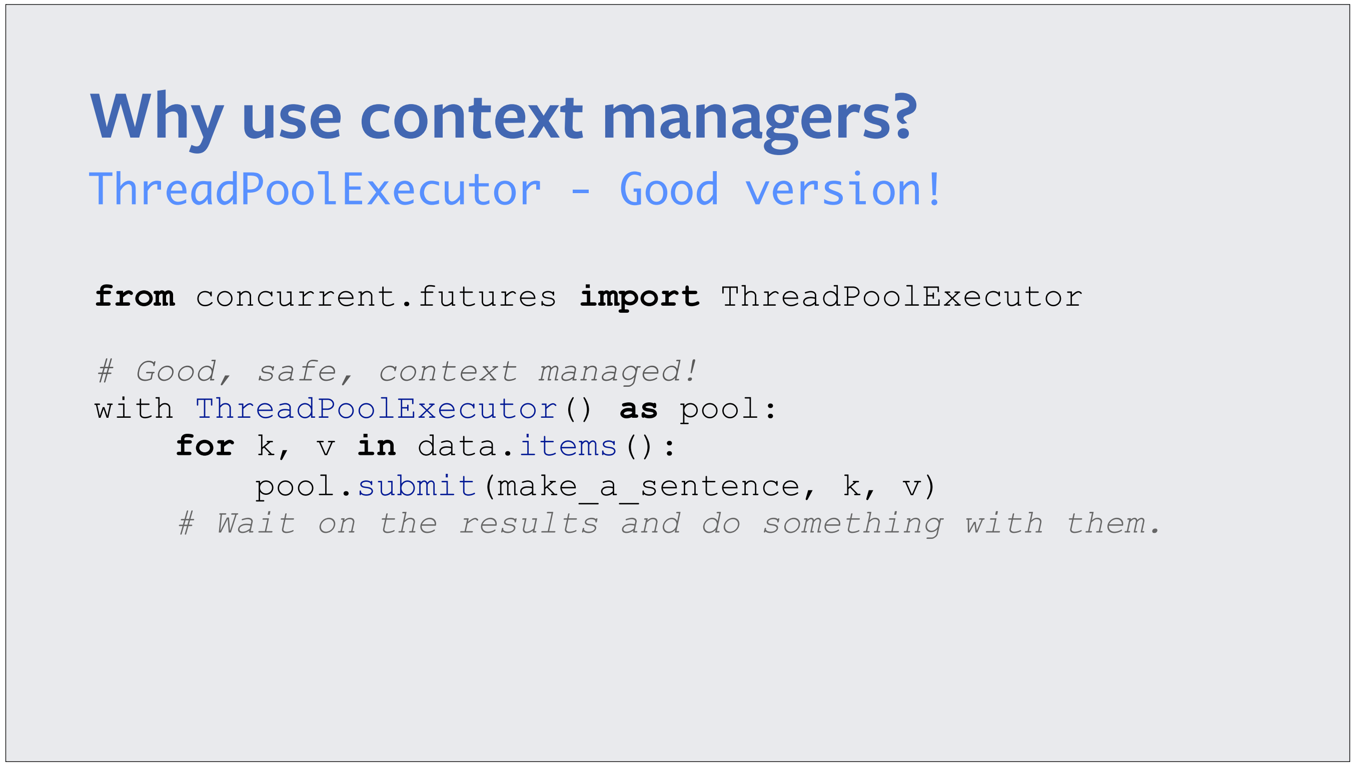
ThreadPoolExecutor - Bad version!

```python
from concurrent.futures import ThreadPoolExecutor

# Bad!!!
pool = ThreadPoolExecutor()
for k, v in data.items():
    pool.submit(make_a_sentence, k, v)
# Wait on the results and do something with them.
pool.shutdown()
```

talk talk talk about points

Let's look at some more examples of context managers.



Why use context managers?

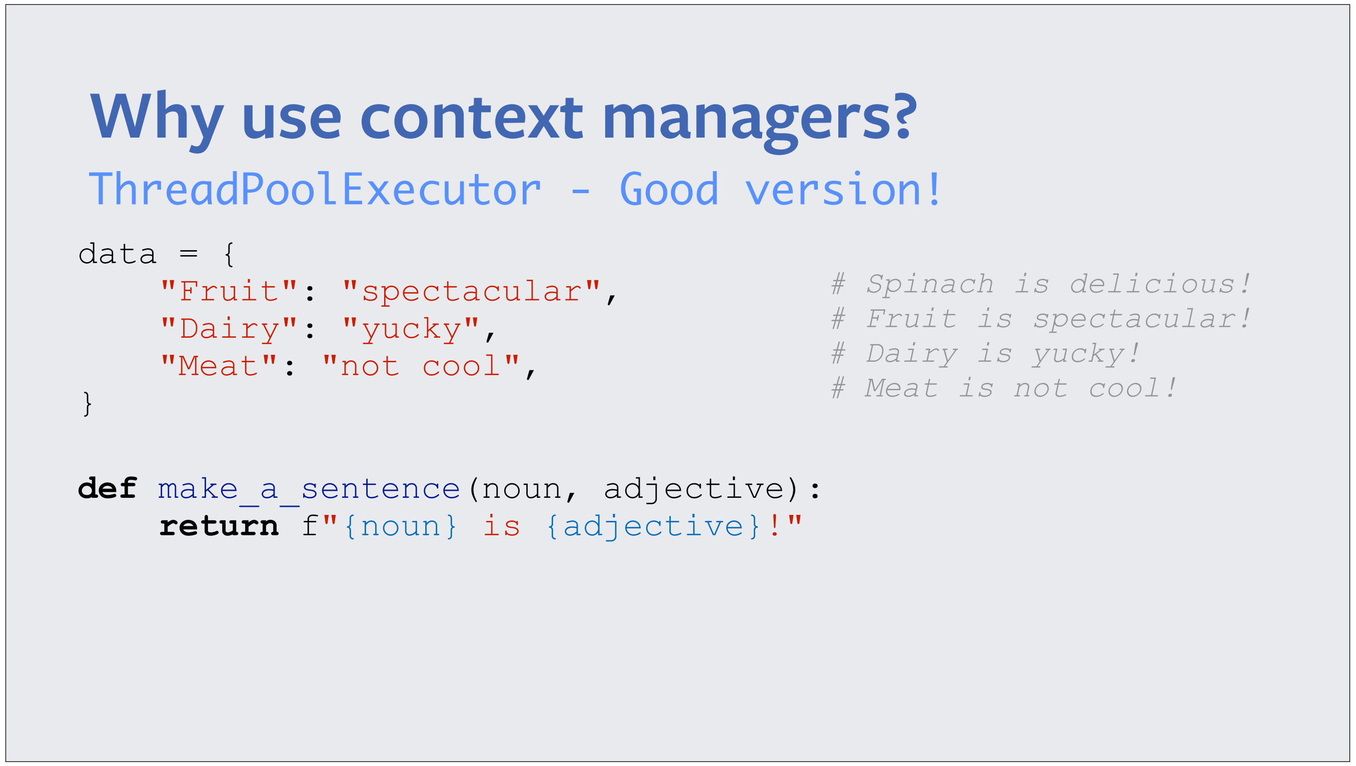
ThreadPoolExecutor - Good version!

```python
from concurrent.futures import ThreadPoolExecutor

# Good, safe, context managed!
with ThreadPoolExecutor() as pool:
    for k, v in data.items():
        pool.submit(make_a_sentence, k, v)
    # Wait on the results and do something with them.
```

talk talk talk about points

Let's look at some more examples of context managers.



Why use context managers?

ThreadPoolExecutor - Good version!

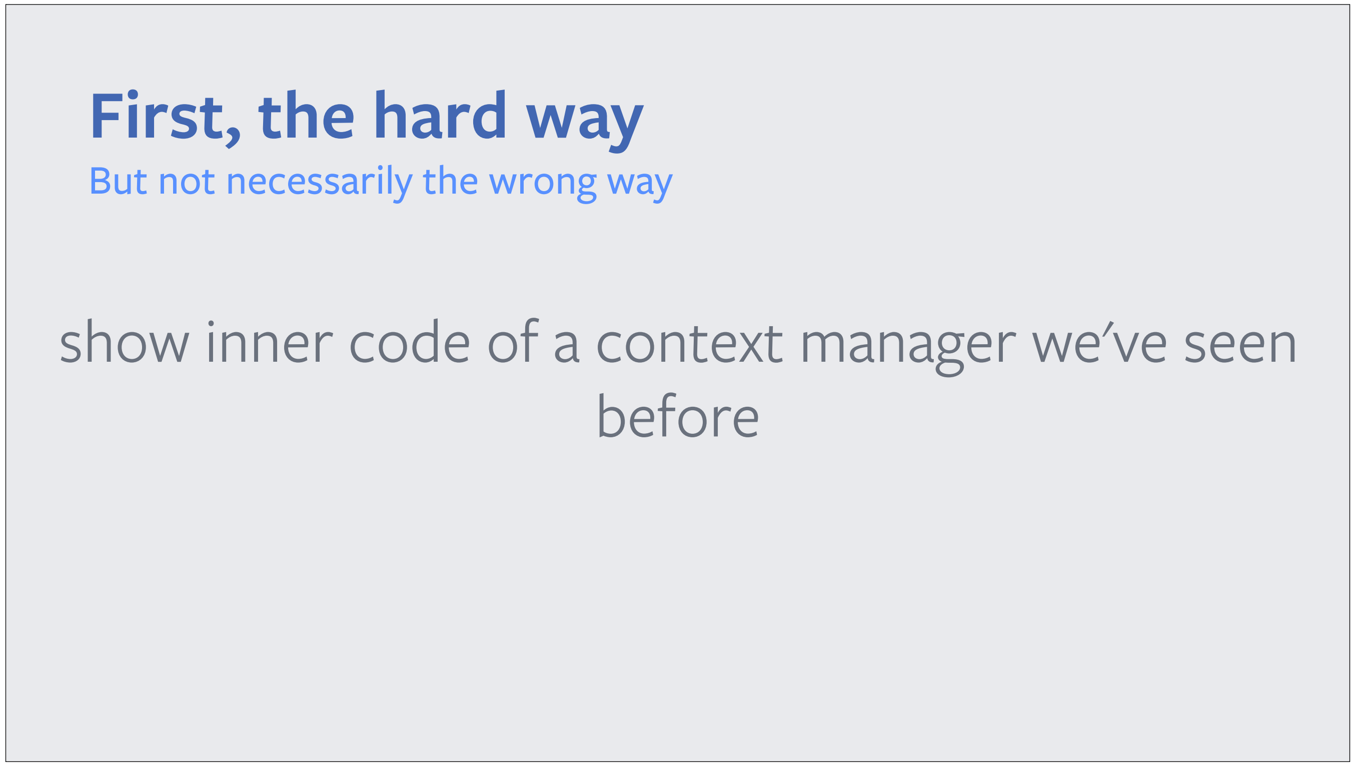
```
data = {
    "Fruit": "spectacular",                 # Spinach is delicious!
    "Dairy": "yucky",                       # Fruit is spectacular!
    "Meat": "not cool",                     # Dairy is yucky!
}                                           # Meat is not cool!

def make_a_sentence(noun, adjective):
    return f"{noun} is {adjective}!"
```

talk talk talk about points

Let's look at some more examples of context managers.

Write your own context managers!



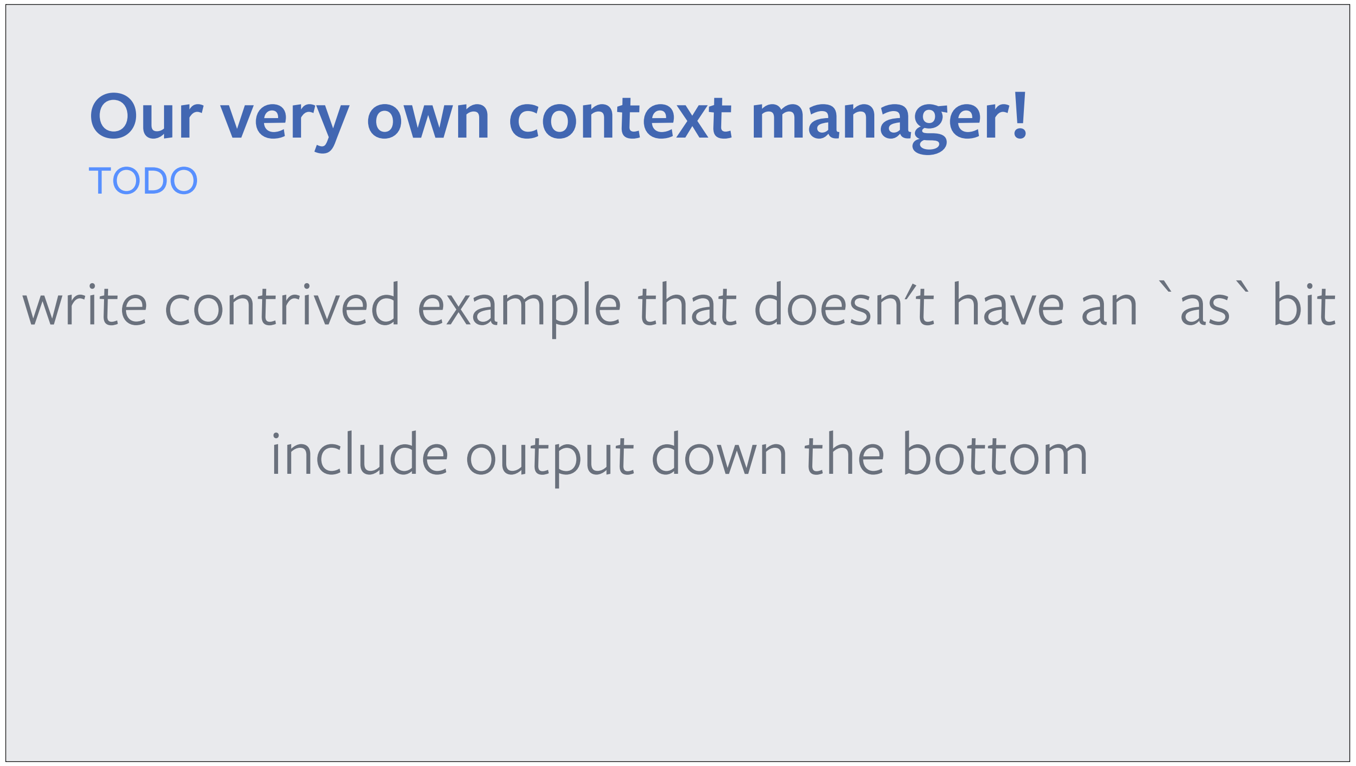
First, the hard way

But not necessarily the wrong way

show inner code of a context manager we've seen before

what does a context manager look like under the hood?

formally, context manager is any class with an __enter__ and __exit__ method, you must define both. __init__ method optional.

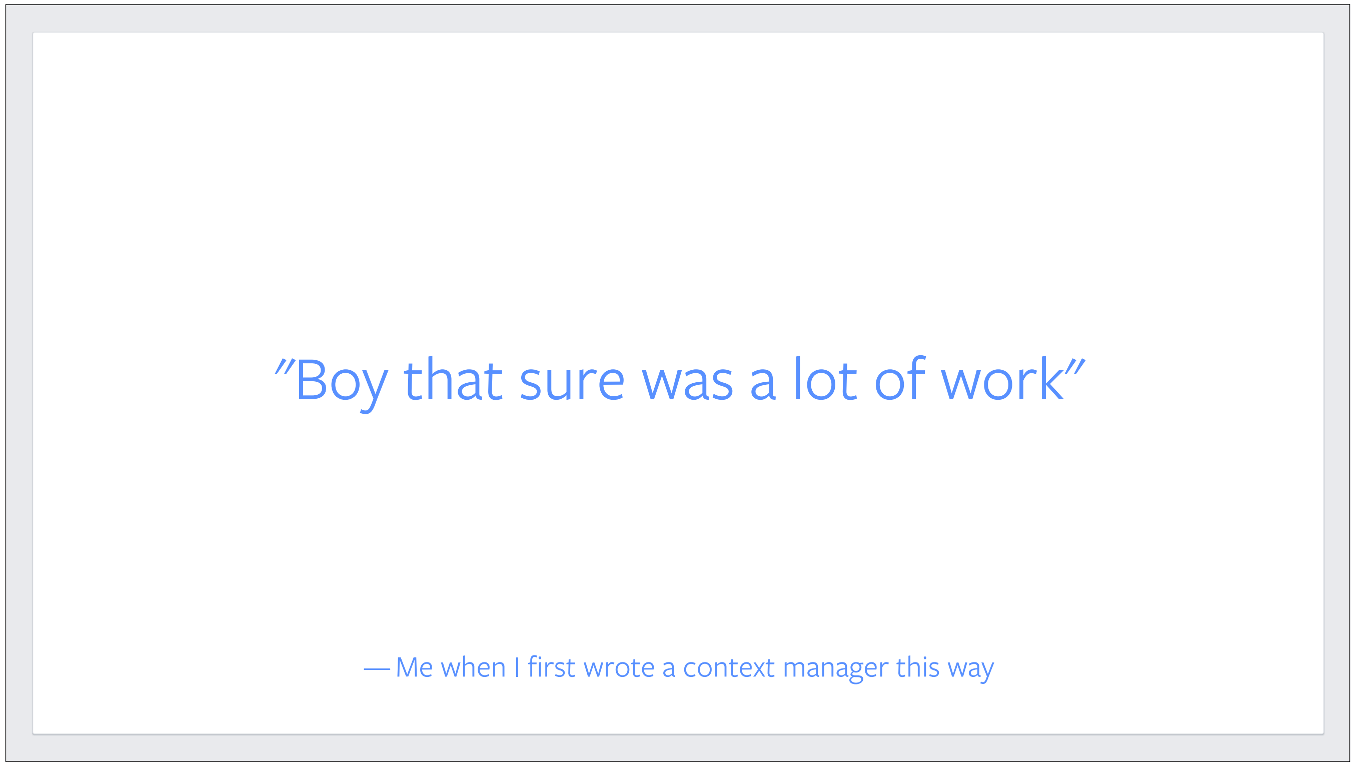


Our very own context manager!

TODO

write contrived example that doesn’t have an `as` bit

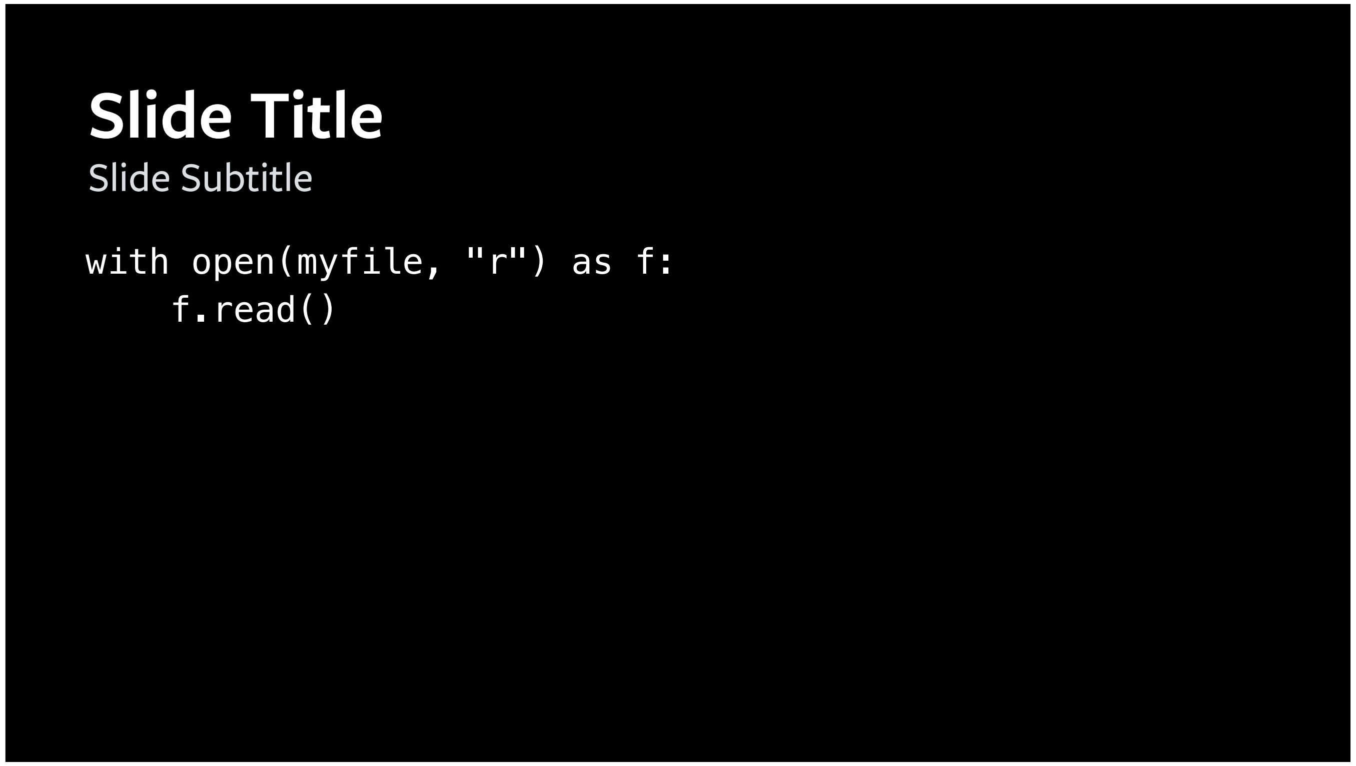
include output down the bottom



”Boy that sure was a lot of work”

— Me when I first wrote a context manager this way

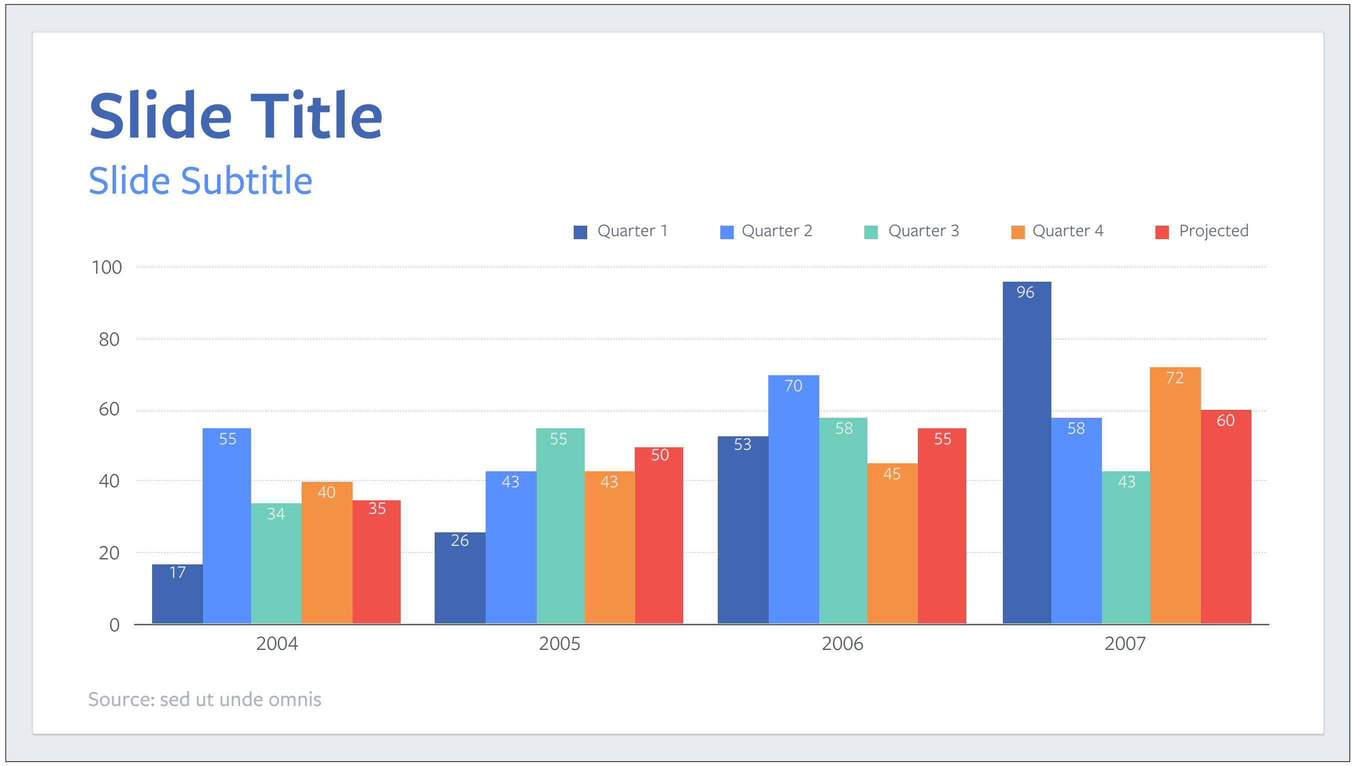
wow a class and two methods just for that? surely there's an easier way!



Slide Title

Slide Subtitle

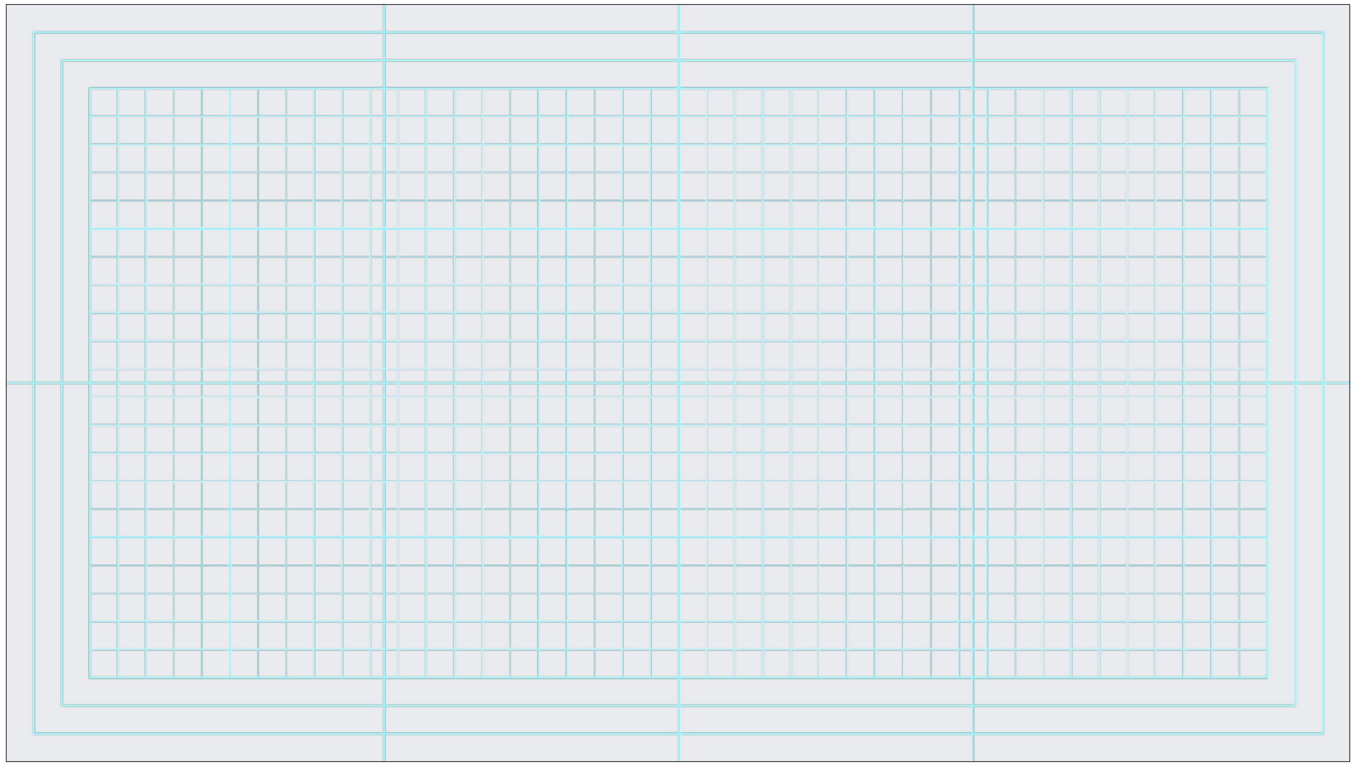
```
with open(myfile, "r") as f:
    f.read()
```



Slide Title

Slide Subtitle

Source: sed ut unde omnis



facebook