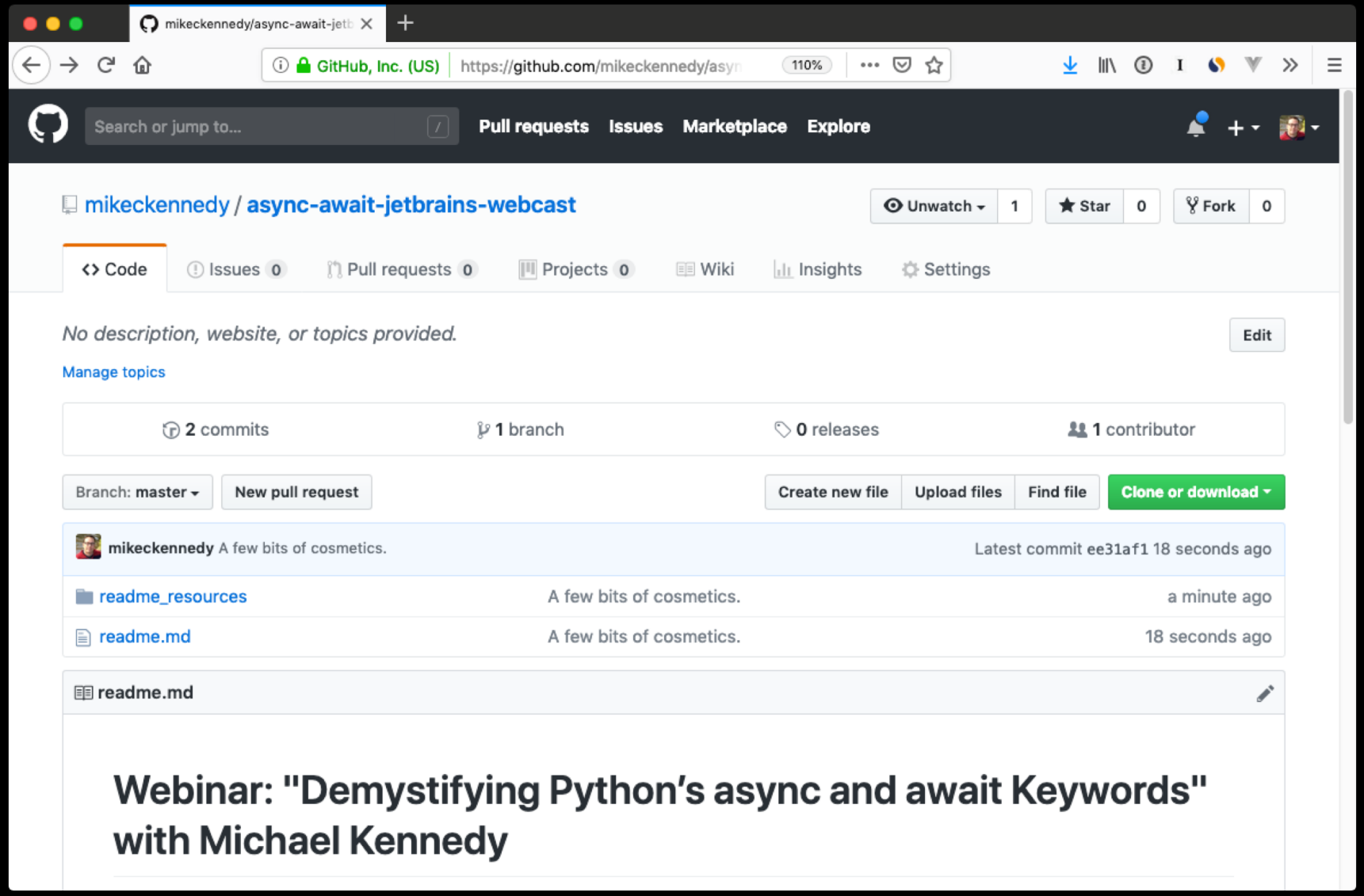
Demystifying Python's async and await Keywords

Michael Kennedy
Founder, Talk Python Training
@mkennedy

Take the code with you



https://github.com/mikeckennedy/async-await-jetbrains-webcast

What is asynchronous programming?

***Asynchrony**, in computer programming, refers to the occurrence of events independent of the main program flow and ways to deal with such events.*

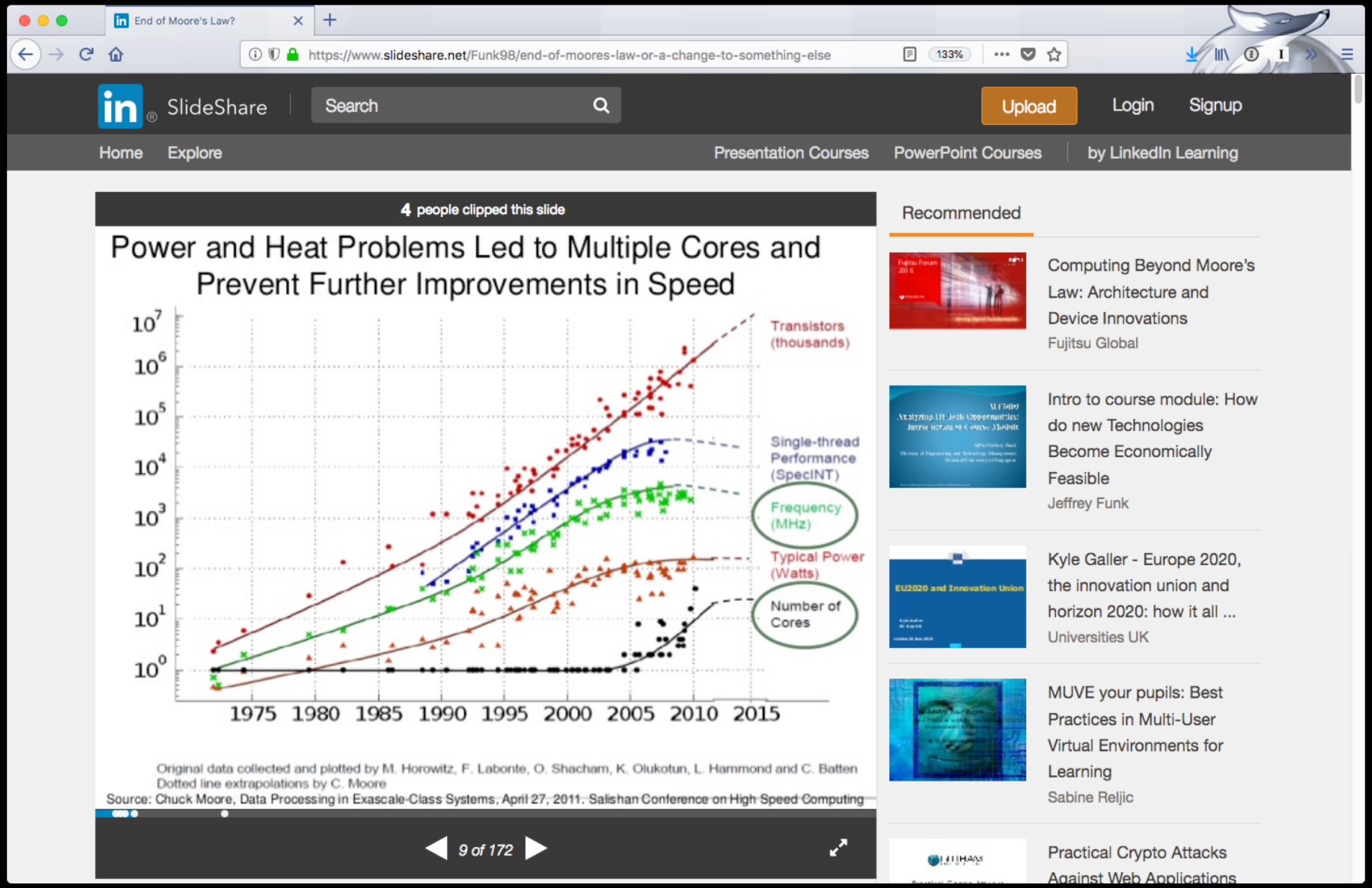
These may be "outside" events such as the arrival of signals, or actions instigated by a program that take place concurrently with program execution, without the program blocking to wait for results.

-- Wikipedia

https://en.wikipedia.org/wiki/Asynchrony_(computer_programming)

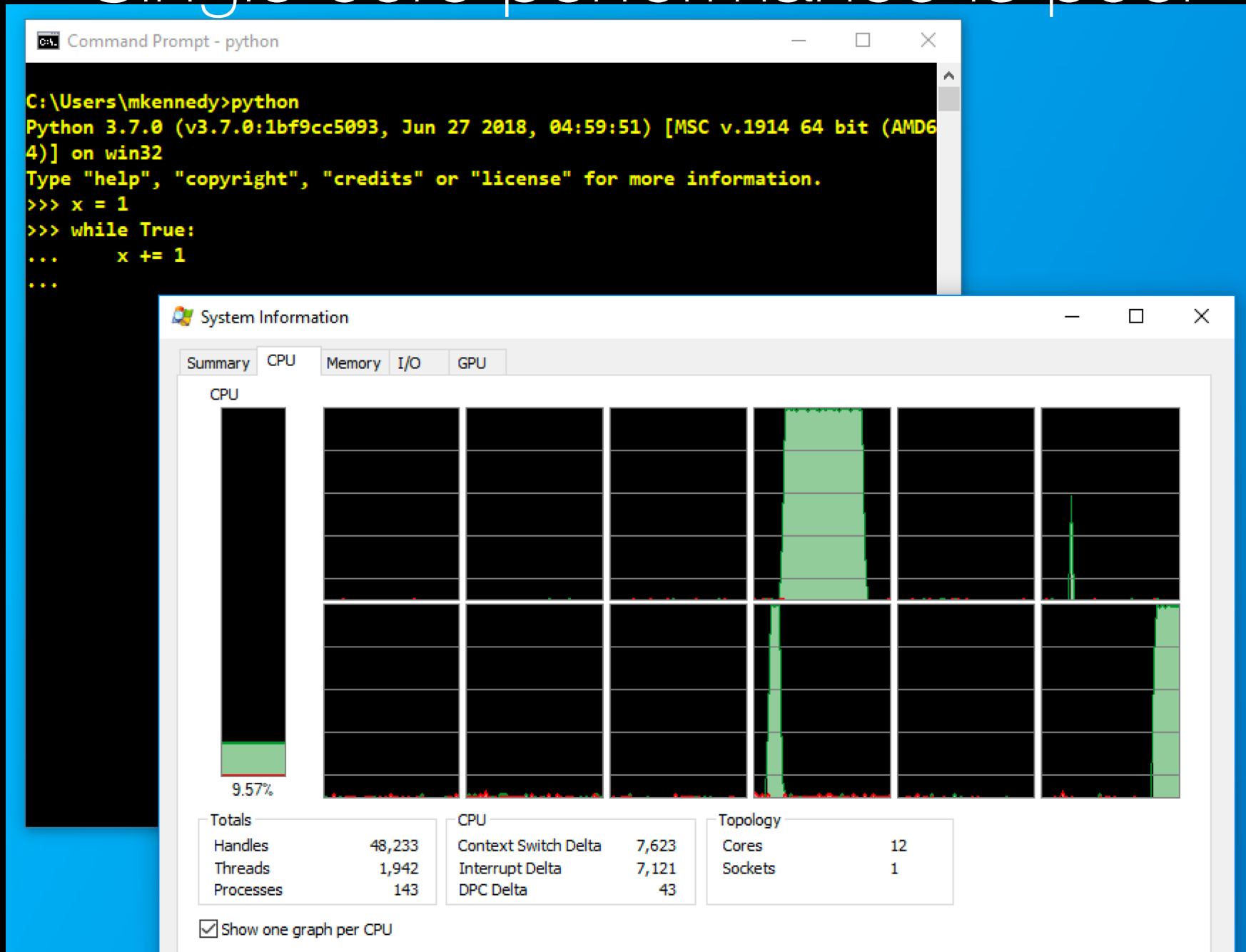

async for speed

CPUs are multi-core

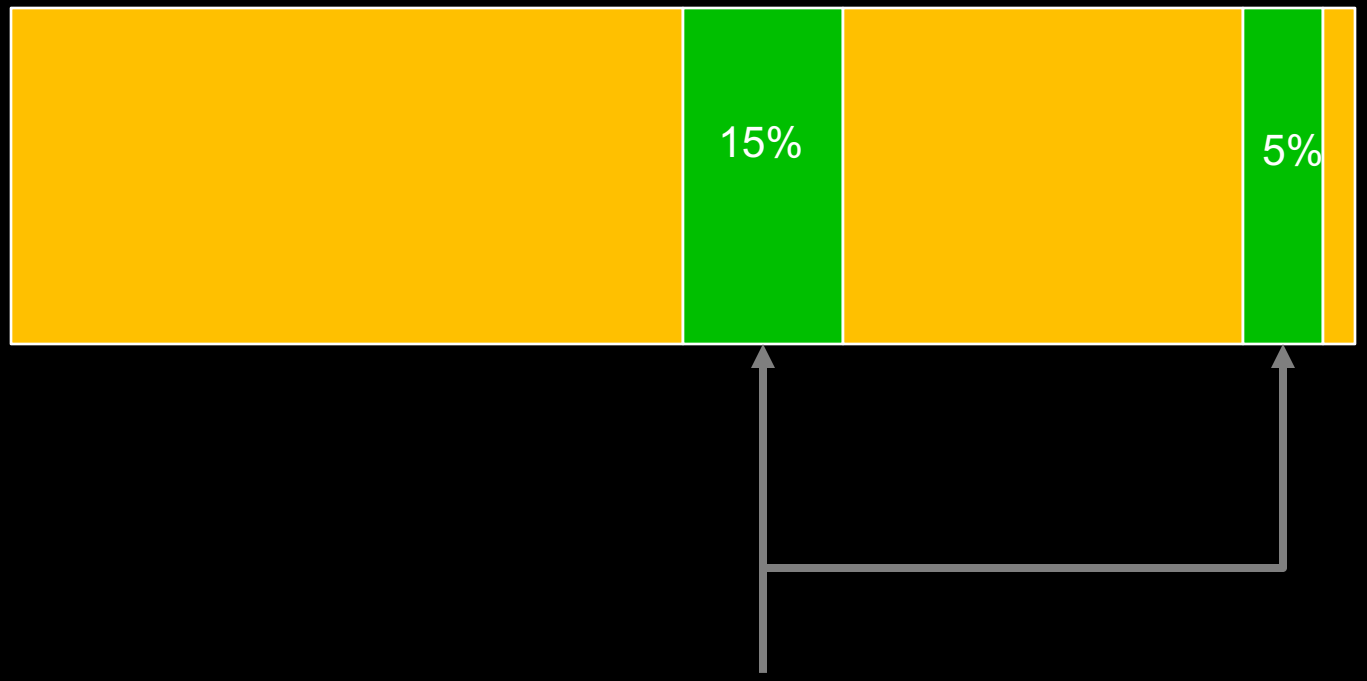


via Jeffrey Funk,
https://www.slideshare.net/Funk98/end-of-moores-law-or-a-change-to-something-else

Single core performance is poor

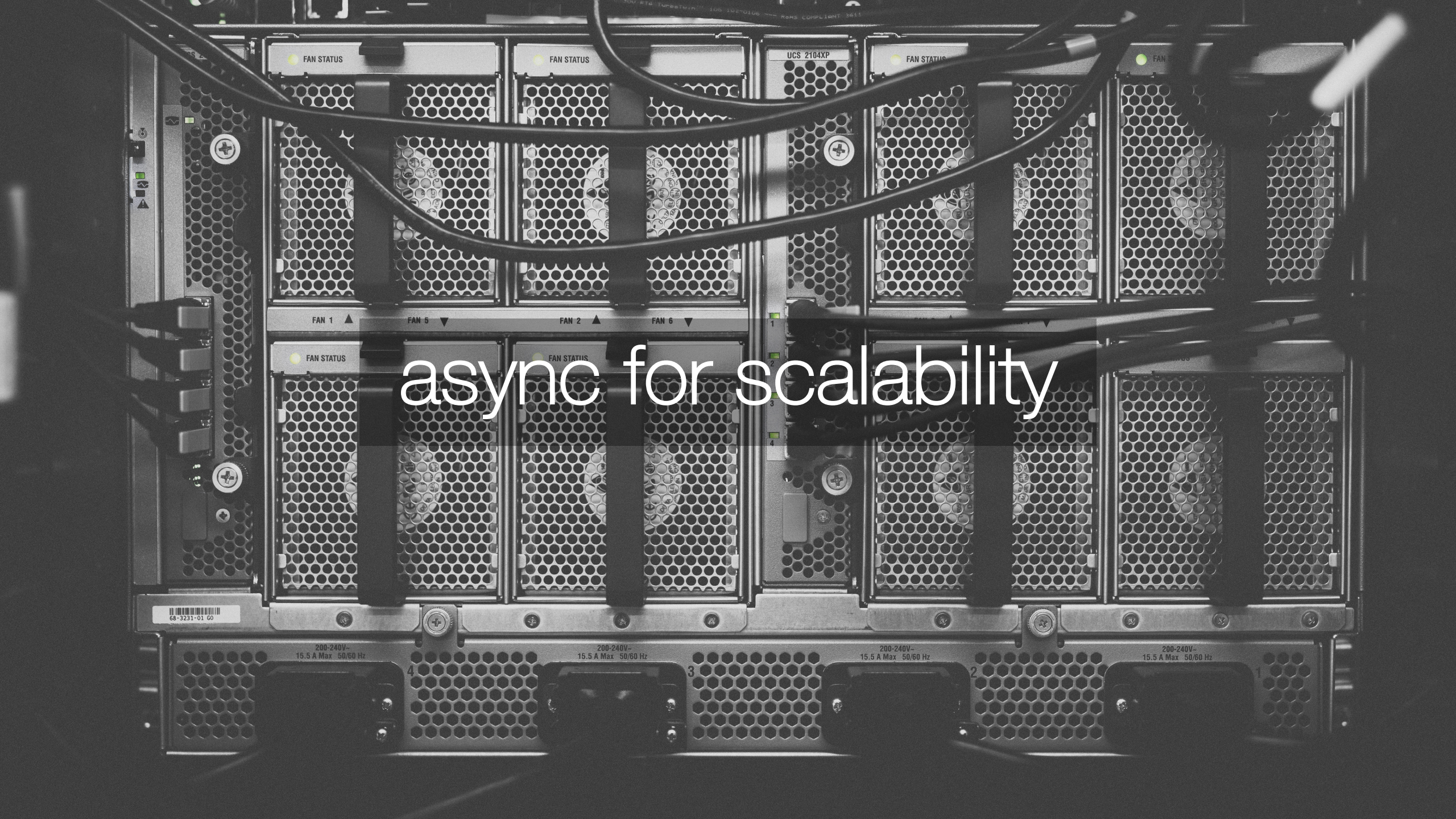


An upper bound for improvement



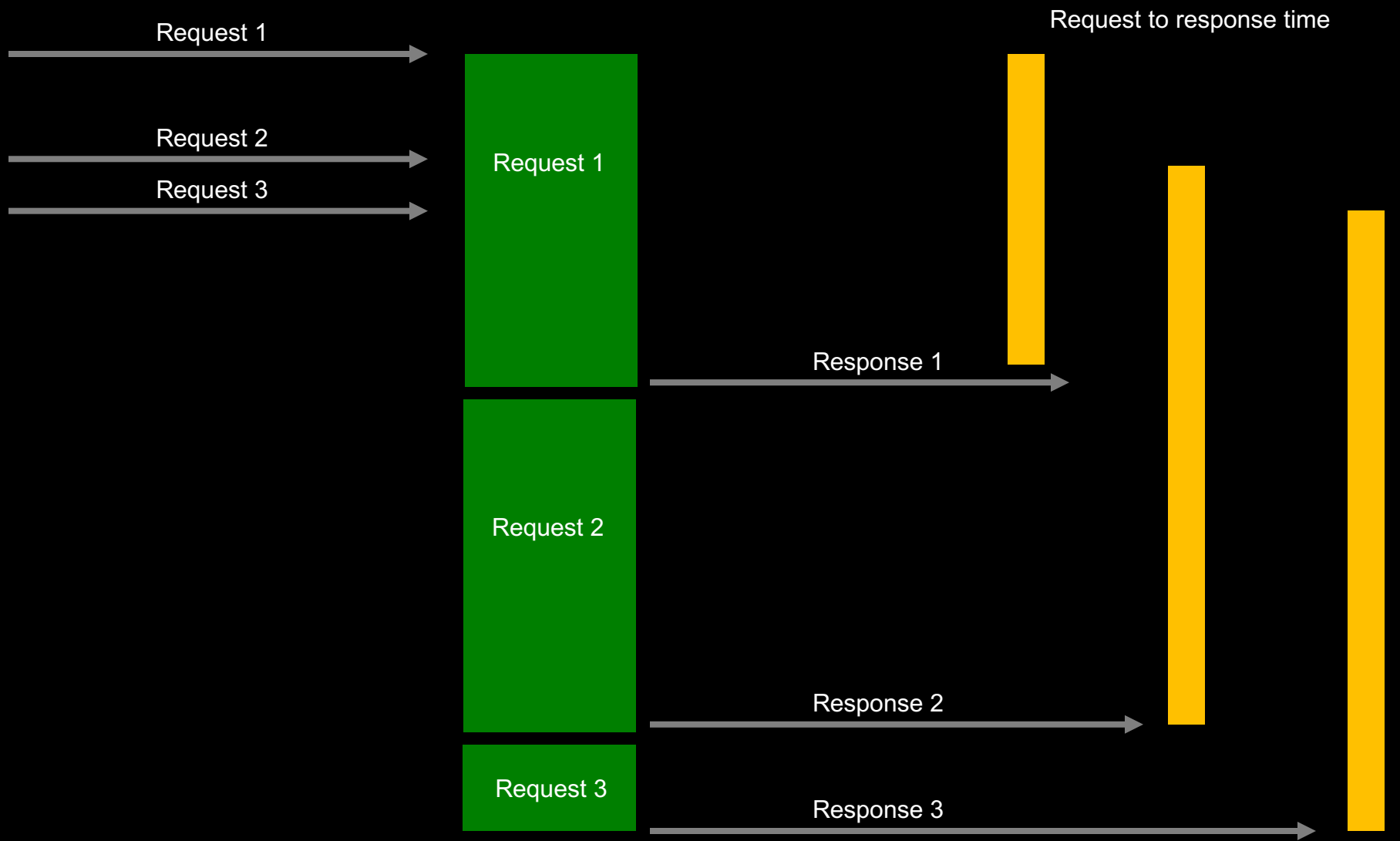
Absolute performance improvement = **20%**

Potentially concurrent operations

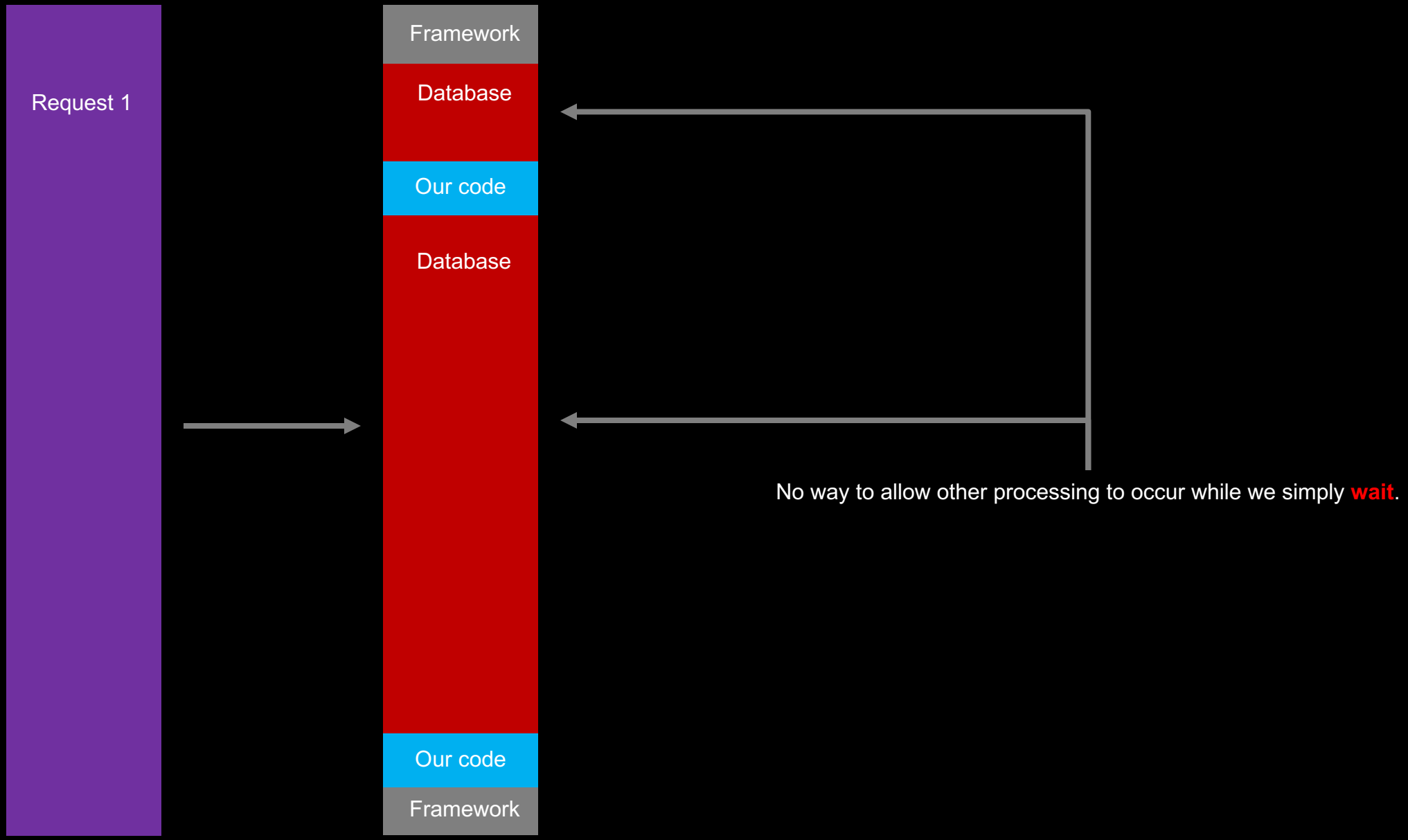


async for scalability

Synchronous execution

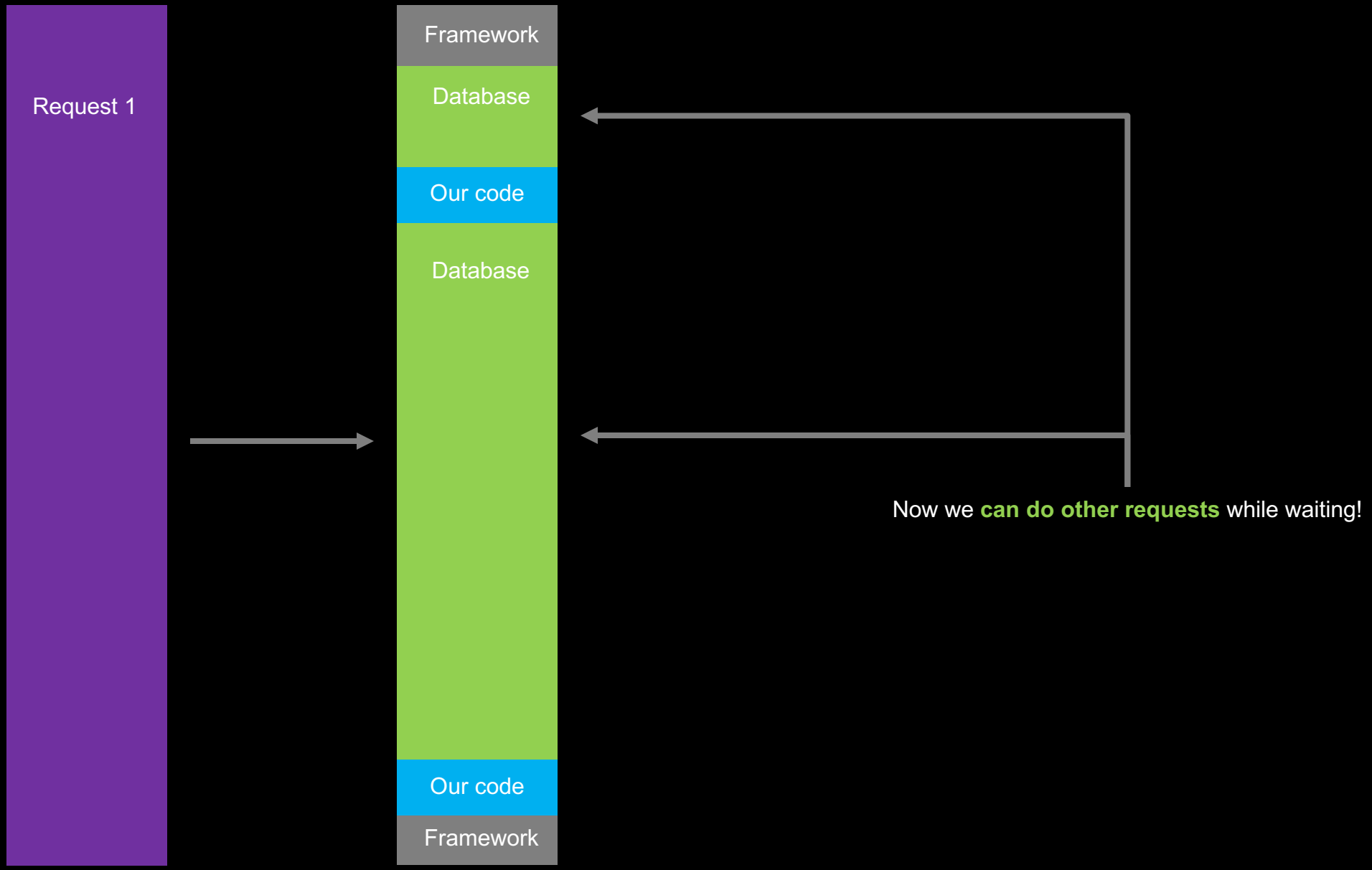


Synchronous execution –zoom in



Asynchronous execution

Asynchronous execution –zoom in



Async in Python (techniques)

Do more **at once**

`asyncio`

`threads`

Do things **faster**

`multiprocessing`

`C / Cython`

Do these **easier**

`trio`

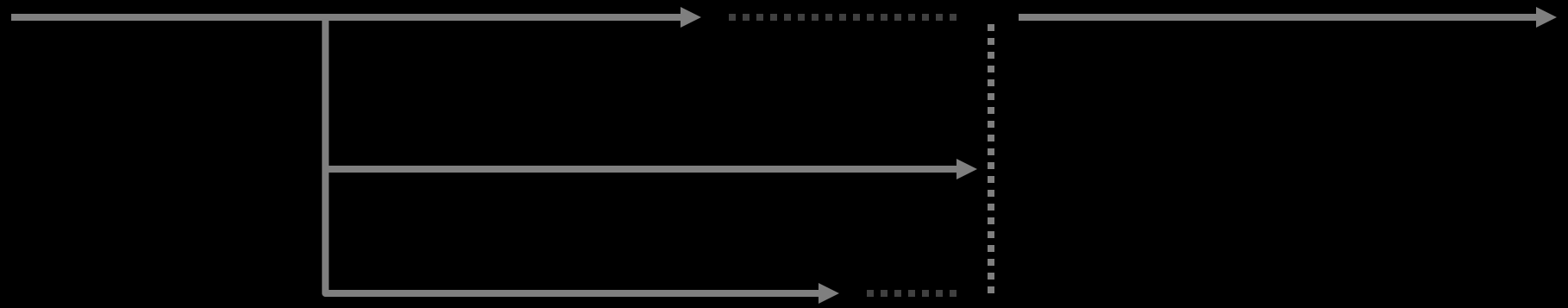
`unsync`

Why don't threads add computational speed?

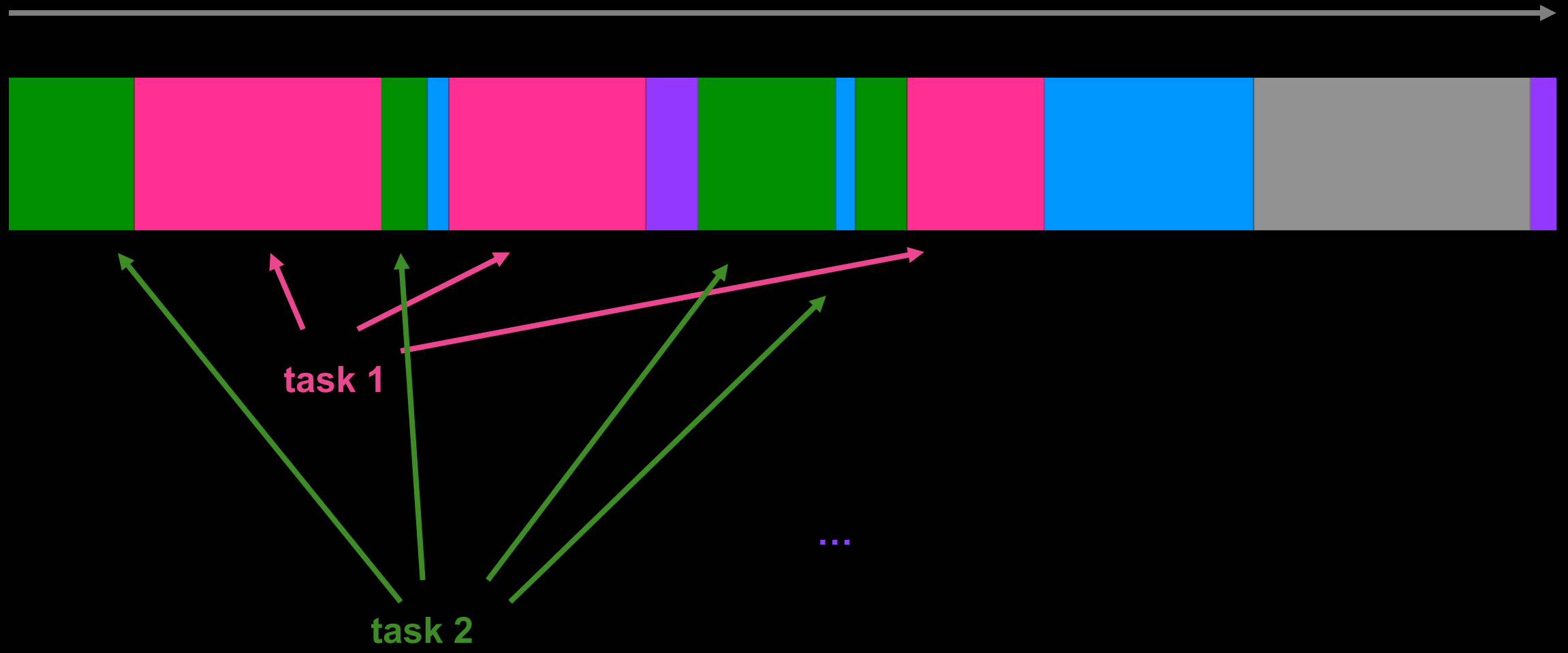
Python has a memory management feature
called **the GIL**, or Global Interpreter Lock

https://realpython.com/python-gil/

Typical concurrent programming



I/O driven concurrency



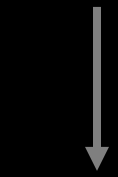
async event loop

Demo: Very simple example of Fibonacci generator with yield

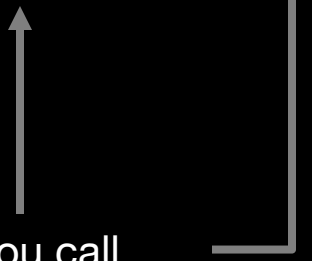
Demo producer consumer

Anatomy of an async method

Begin by making method **async**



```
async def process_data(num: int, data: asyncio.Queue):
    processed = 0

    while processed < num:
        item = await data.get()
        # work with item
```



await all async methods you call.

Performance improvements: Sync version

```
➜ python3 sync_producer.py
 -- generated item 15
 -- generated item 16
 -- generated item 17
 -- generated item 18
 -- generated item 19
 -- generated item 20
 +++ Processed value 1 after 19.75 sec.
 +++ Processed value 4 after 19.61 sec.
 +++ Processed value 9 after 18.70 sec.
 +++ Processed value 16 after 18.04 sec.
 +++ Processed value 25 after 17.72 sec.
 +++ Processed value 36 after 17.43 sec.
Total time: 29.81 sec, ave latency: 15 sec
```

Performance improvements: async version

```
➜ python3 sync_producer.py
 -- generated item 7
 +++ Processed value 49 after 0.24 sec.
 -- generated item 6
 -- generated item 8
 +++ Processed value 36 after 0.23 sec.
 +++ Processed value 64 after 0.59 sec.
 -- generated item 7
 -- generated item 9
 +++ Processed value 49 after 0.15 sec.
 +++ Processed value 81 after 0.60 sec.
 -- generated item 10
Total time: 21.30 sec, ave latency: 0.25 sec
```

8.51 seconds faster, 60x less latency

Fast loops

https://uvloop.readthedocs.io/

Using uvloop

```
import uvloop

asyncio.set_event_loop_policy(uvloop.EventLoopPolicy())

loop = asyncio.get_event_loop()
# ...
```


How about some real work?

Demo web scraper

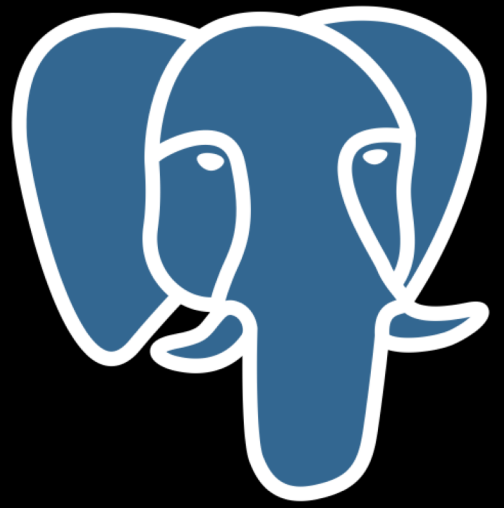
Async web requests

Use an async with block to start the request

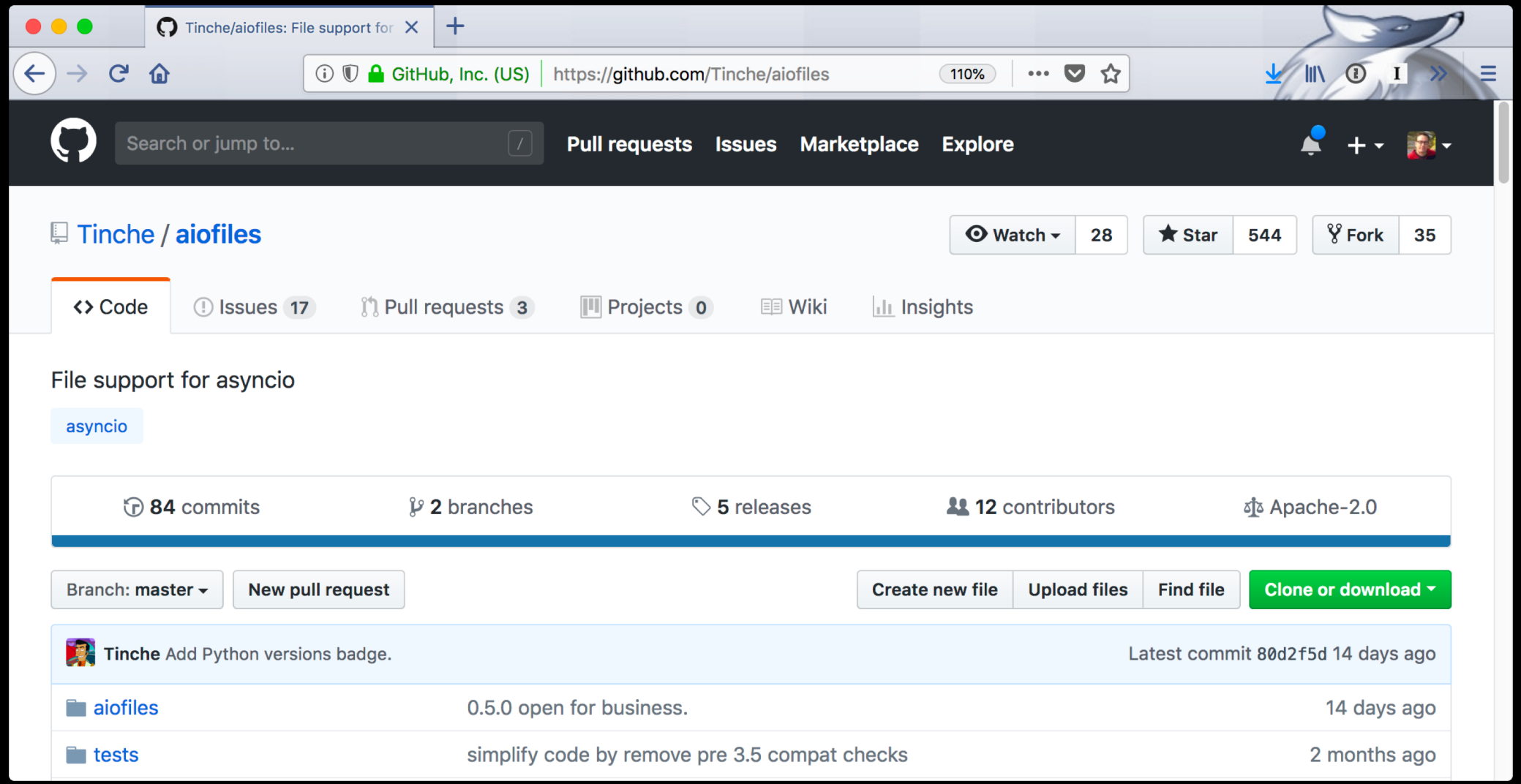
```
async def get_html(url: str):

    async with aiohttp.ClientSession() as session:
        async with session.get(url) as resp:
            resp.raise_for_status()

            return await resp.text()
```

await the network read operation

Other async capable libraries

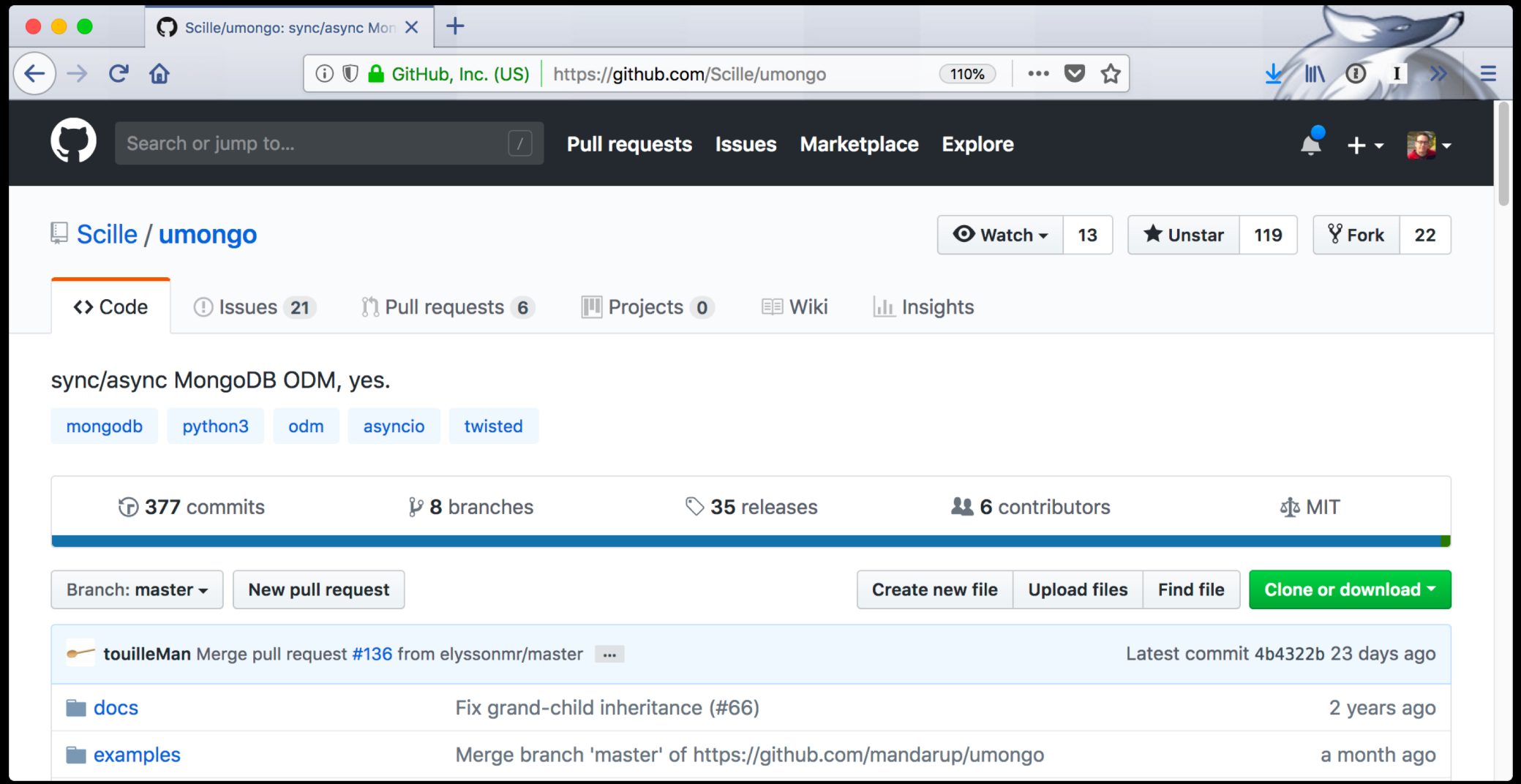


File I/O



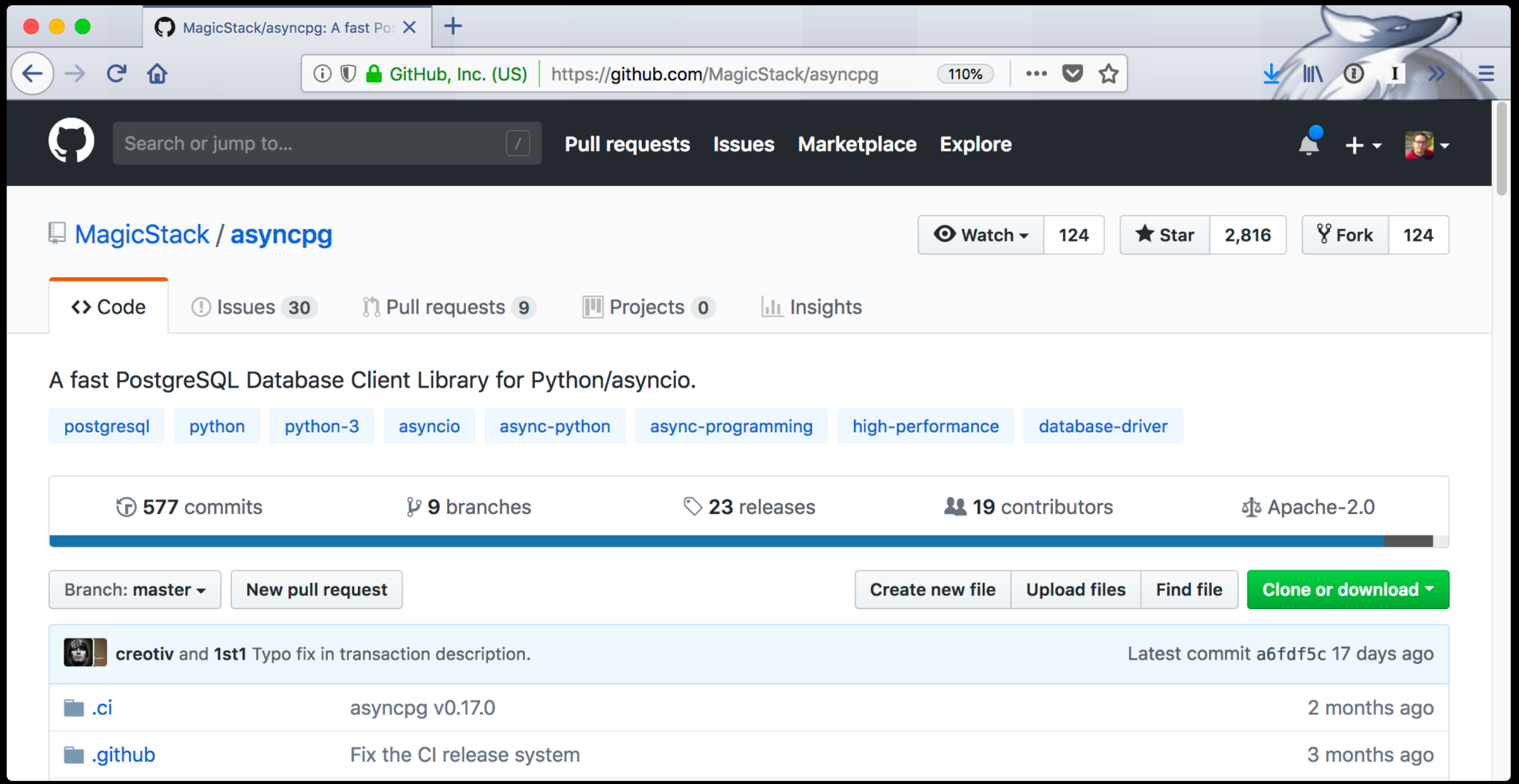
https://github.com/Tinche/aiofiles

MongoDB



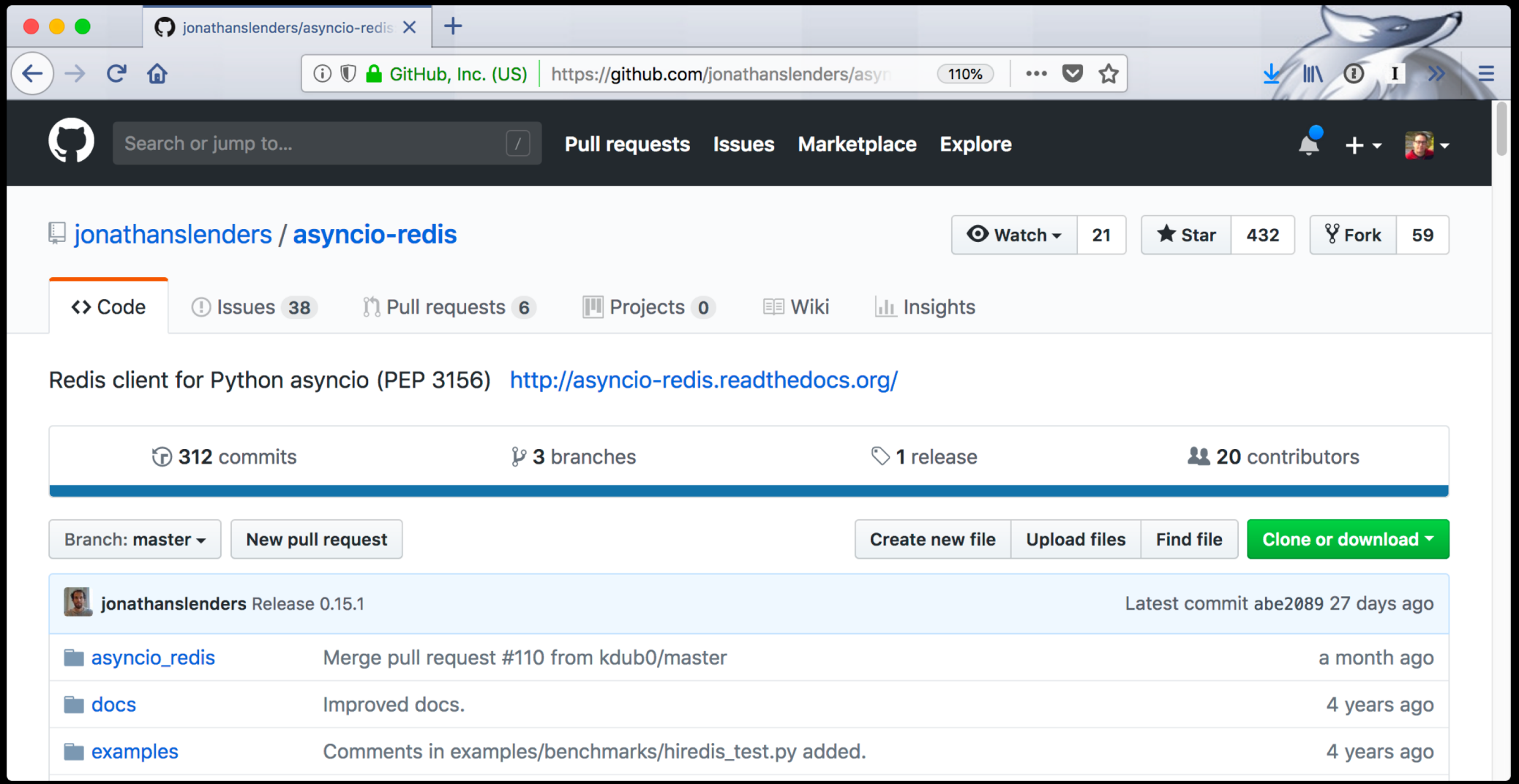
https://github.com/Scille/umongo

Postgres



https://github.com/MagicStack/asyncpg

Redis

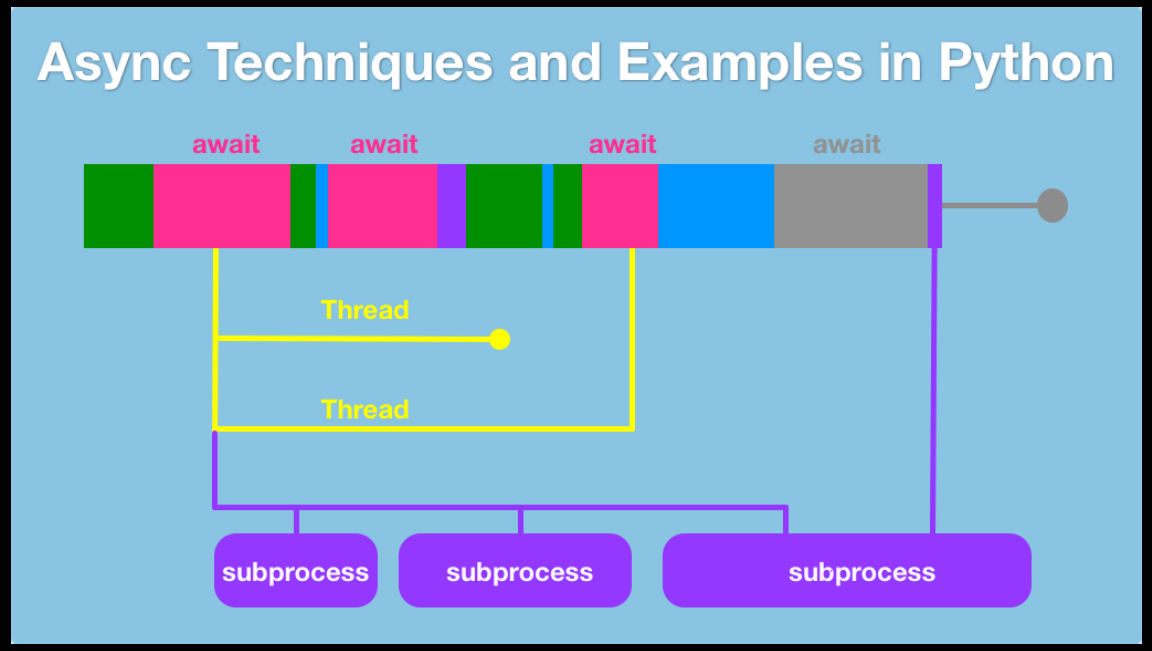


https://github.com/jonathanslenders/asyncio-redis

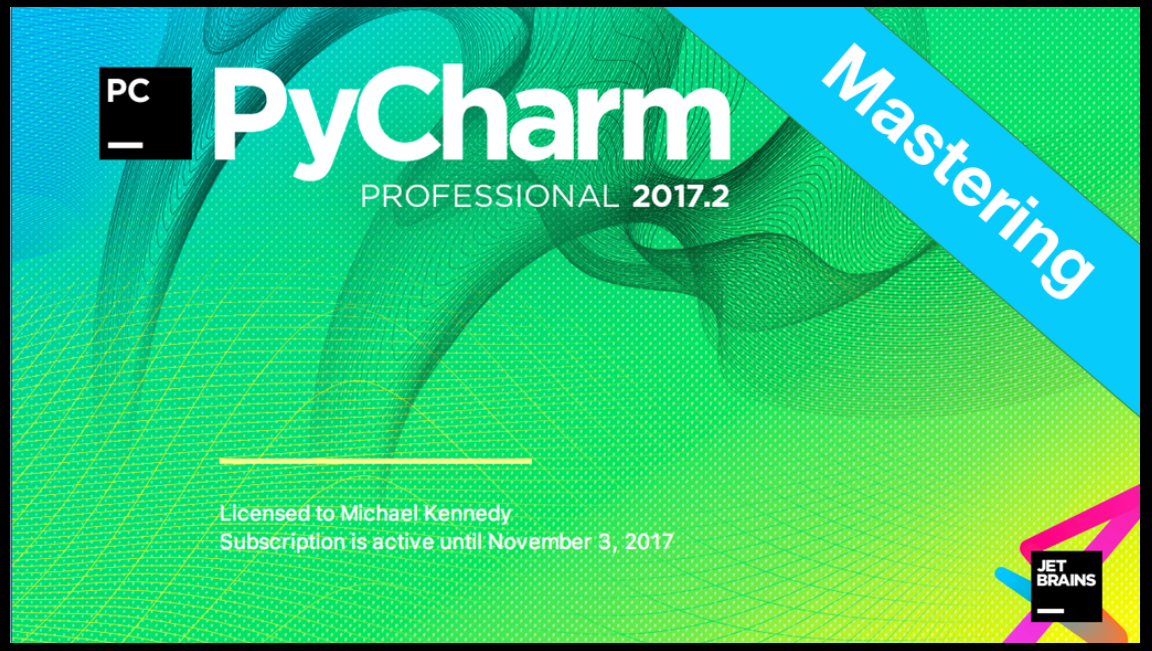
THANK YOU

Michael Kennedy
@mkennedy

Want to go deep? Check out my online courses



talkpython.fm/async



talkpython.fm/pycharm

Webcast materials: https://github.com/mikeckennedy/async-await-jetbrains-webcast