Virtual env setup

Monday, March 21, 2022 10:08 AM

https://github.com/t4d-classes/advanced-python_03212022
https://classes.t4d.download/advanced-python_03212022_RyVgKC9prVMkl4qcDMGj

Eric Greenea
434-509-6890
eric@t4d.io

Create conda env with python 3.9.6:

```
> conda create --name python396 python=3.9.6
> activate python396
```

Create python virtual environment:

```
> python –m venv venv
> .\venv\Scripts\activate.bat
> deactivate
```

VS conde stuff
Ctrl+shif+p
Python: Select Interpreter

Linting:

```
> python -m pip install autopep8 mypy pylint
```

IO vs cpu bound operations

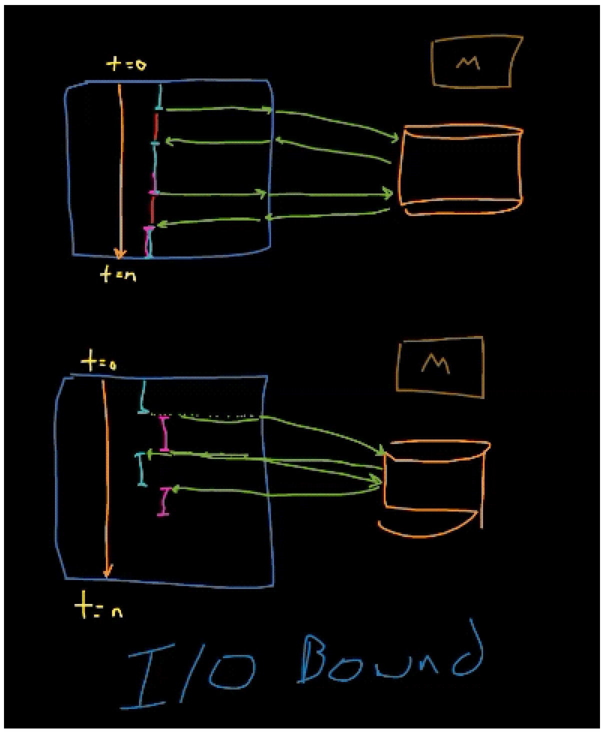
Tuesday, March 22, 2022 11:58 AM

→ IO Bound operation → CPU is mostly Idle, waiting for IO to happen

→ Computation is fast
memory access is slow
register > disk > network call
fastest ——→ slowest

→ cpu remains Idle until IO (memory) request is being processed

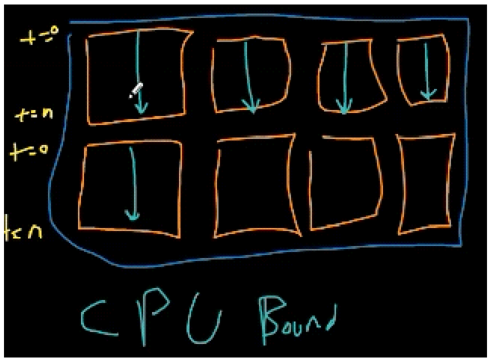
Threading Becomes tricky when memory access is involved. Multiple threads access some block of memory. Memory Locks. [all threads operate in same memory space]



Threading is usefull for IO Bound operations
Multiple IO operations happen at same time on different threads

CPU Bound operations: CPU is clogged up with operations. Threading wont work
Might actuall hurt because of contex switching

Need to use multiprocessing -> one task in one processor. distribute work on multiple processors



Problem with memory access
every process operates in its own block of memory

no problem with ~~locks~~ mem locks like in threading
But now Its harder for processes to access each others memory

concurrency : threading
Parallelism : multi Processing

Hyper threading

Tuesday, March 22, 2022 1:32 PM

Hyper threading -> one core -> presents on operating system as 2 separate process

8 cores = 16 process (Subprocesses)?

on 8core -> 16 process machine (multiprocessing.Manager)

Not true multiprocessing because of hyperthread

8 threads -> run on single core/process

8 processes -> run on 8 separate cores/processes

16 process -> run on 16 separate cores/process

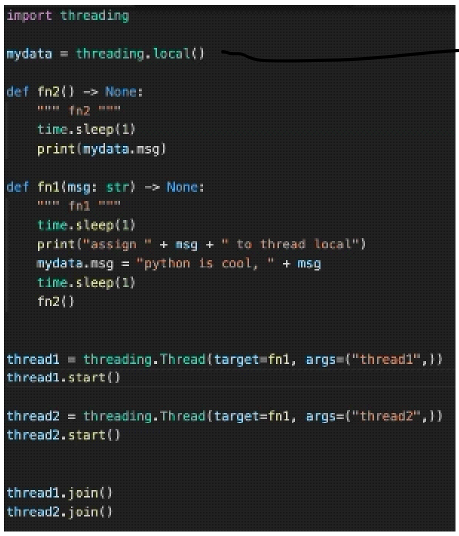
32 process -> contex switch between 16 cores/process

Thread memory - data sharing

Wednesday, March 23, 2022 9:01 AM

thread local object

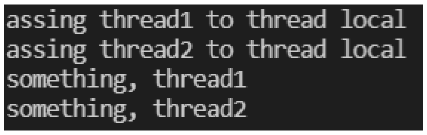
each api endpoit call flask creates a thread local variable that can be accessed across that thread only 'request.args'



```
import threading

mydata = threading.local()

def fn2() -> None:
    """ fn2 """
    time.sleep(1)
    print(mydata.msg)

def fn1(msg: str) -> None:
    """ fn1 """
    time.sleep(1)
    print("assign " + msg + " to thread local")
    mydata.msg = "python is cool, " + msg
    time.sleep(1)
    fn2()


thread1 = threading.Thread(target=fn1, args=("thread1",))
thread1.start()

thread2 = threading.Thread(target=fn1, args=("thread2",))
thread2.start()


thread1.join()
thread2.join()
```

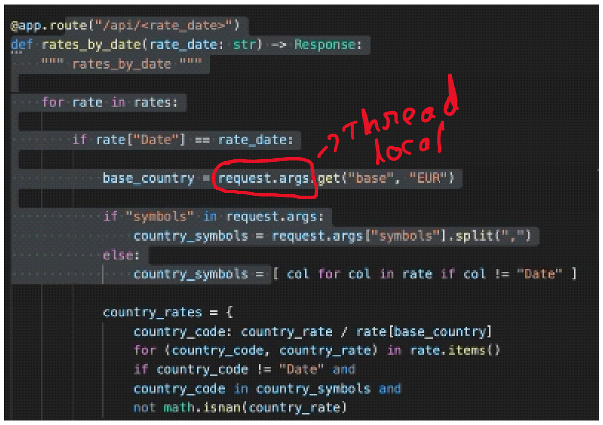
→ only local to particular thread. not shared between threads

out



```
assing thread1 to thread local
assing thread2 to thread local
something, thread1
something, thread2
```

NOTE: creating a new thread inside a router in flask: new thread will not have access to request.args



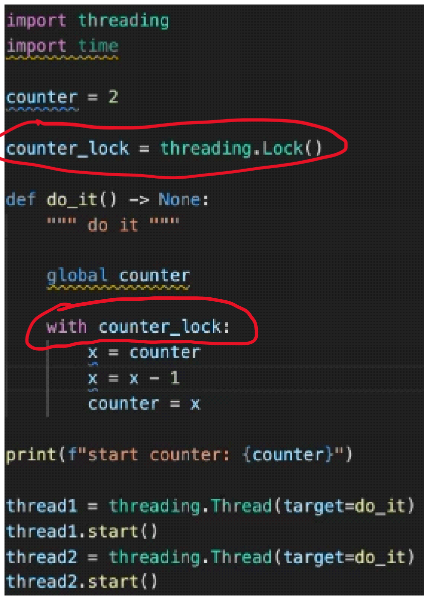
```
@app.route("/api/<rate_date>")
def rates_by_date(rate_date: str) -> Response:
    """ rates_by_date """

    for rate in rates:

        if rate["Date"] == rate_date:

            base_country = request.args.get("base", "EUR")

            if "symbols" in request.args:
                country_symbols = request.args["symbols"].split(",")
            else:
                country_symbols = [ col for col in rate if col != "Date" ]

            country_rates = {
                country_code: country_rate / rate[base_country]
                for (country_code, country_rate) in rate.items()
                if country_code != "Date" and
                country_code in country_symbols and
                not math.isnan(country_rate)
```

→ thread local

sharing data beween threads using locks

Sharing data beween threads using locks

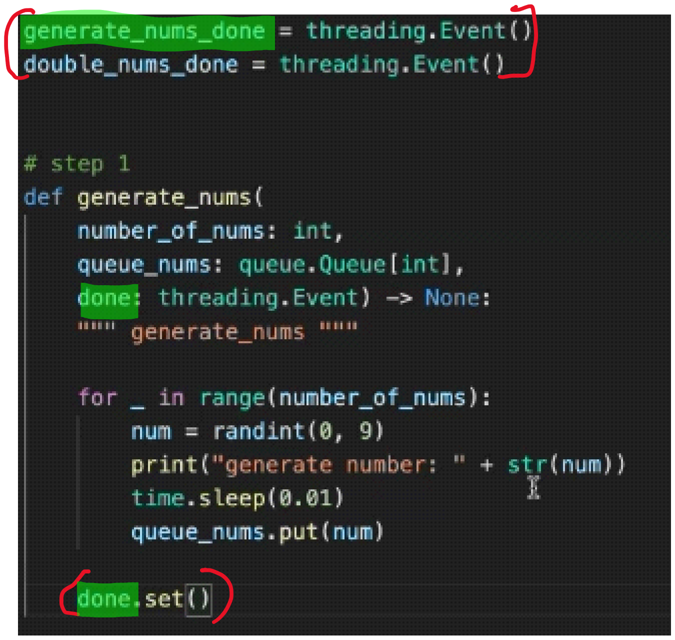
note: locking nullifies parallel execution. first thread has to be done with lock to wait for second



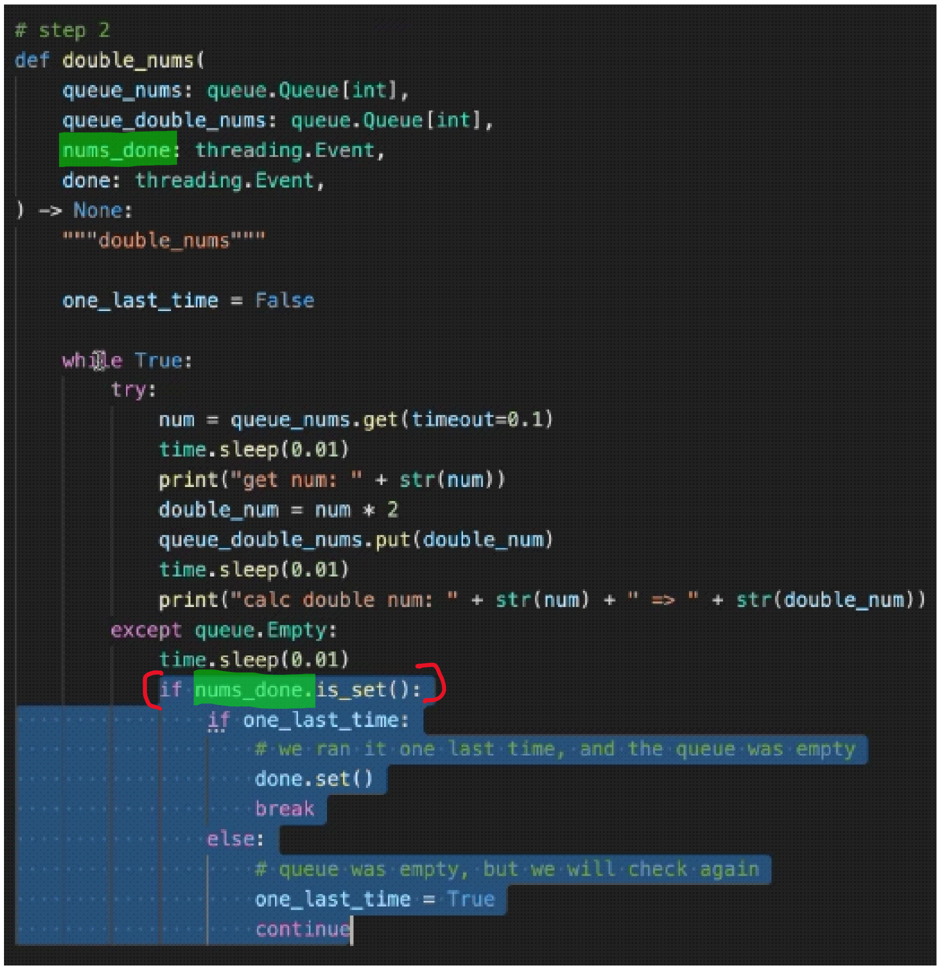
```python
import threading
import time

counter = 2

counter_lock = threading.Lock()

def do_it() -> None:
    """ do it """

    global counter

    with counter_lock:
        x = counter
        x = x - 1
        counter = x

print(f"start counter: {counter}")

thread1 = threading.Thread(target=do_it)
thread1.start()
thread2 = threading.Thread(target=do_it)
thread2.start()
```

Thread events

Thursday, March 24, 2022 9:11 AM



```python
generate_nums_done = threading.Event()
double_nums_done = threading.Event()


# step 1
def generate_nums(
    number_of_nums: int,
    queue_nums: queue.Queue[int],
    done: threading.Event) -> None:
    """ generate_nums """

    for _ in range(number_of_nums):
        num = randint(0, 9)
        print("generate number: " + str(num))
        time.sleep(0.01)
        queue_nums.put(num)

    done.set()
```



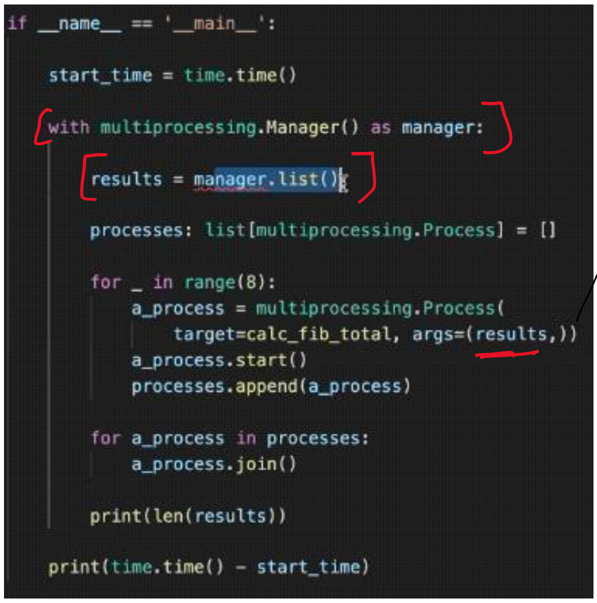
```python
# step 2
def double_nums(
    queue_nums: queue.Queue[int],
    queue_double_nums: queue.Queue[int],
    nums_done: threading.Event,
    done: threading.Event,
) -> None:
    """double_nums"""

    one_last_time = False

    while True:
        try:
            num = queue_nums.get(timeout=0.1)
            time.sleep(0.01)
            print("get num: " + str(num))
            double_num = num * 2
            queue_double_nums.put(double_num)
            time.sleep(0.01)
            print("calc double num: " + str(num) + " => " + str(double_num))
        except queue.Empty:
            time.sleep(0.01)
            if nums_done.is_set():
                if one_last_time:
                    # we ran it one last time, and the queue was empty
                    done.set()
                    break
                else:
                    # queue was empty, but we will check again
                    one_last_time = True
                    continue
```

Shared memory among processes

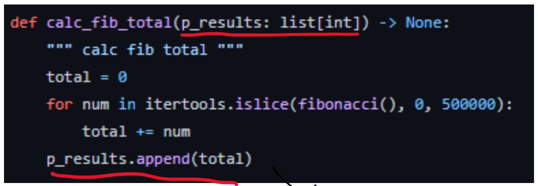
Thursday, March 24, 2022 10:30 AM

each process has its own memory space.
So using "global" wont work as each process has its own global variable in memory

Solution 1 (for mutable objects)



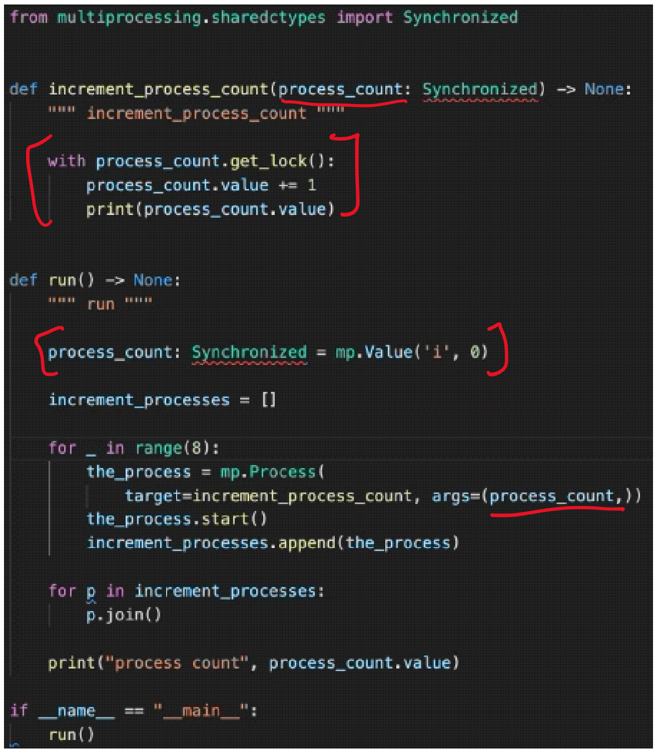
```python
if __name__ == '__main__':

    start_time = time.time()

    with multiprocessing.Manager() as manager:

        results = manager.list()

        processes: list[multiprocessing.Process] = []

        for _ in range(8):
            a_process = multiprocessing.Process(
                target=calc_fib_total, args=(results,))
            a_process.start()
            processes.append(a_process)

        for a_process in processes:
            a_process.join()

        print(len(results))

    print(time.time() - start_time)
```



```python
def calc_fib_total(p_results: list[int]) -> None:
    """ calc fib total """
    total = 0
    for num in itertools.islice(fibonacci(), 0, 500000):
        total += num
    p_results.append(total)
```

no lock needed for manager generated object

solution2 (for immutable objects)



```
from multiprocessing.sharedctypes import Synchronized


def increment_process_count(process_count: Synchronized) -> None:
    """ increment_process_count """

    with process_count.get_lock():
        process_count.value += 1
        print(process_count.value)


def run() -> None:
    """ run """

    process_count: Synchronized = mp.Value('i', 0)

    increment_processes = []

    for _ in range(8):
        the_process = mp.Process(
            target=increment_process_count, args=(process_count,))
        the_process.start()
        increment_processes.append(the_process)

    for p in increment_processes:
        p.join()

    print("process count", process_count.value)

if __name__ == "__main__":
    run()
```

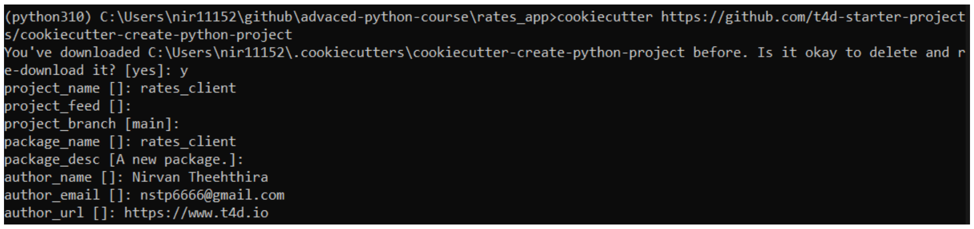
http://www.learningaboutelectronics.com/Articles/Named-groups-with-regular-expressions-in-Python.php

Cookie cutter (creating packages)

Friday, March 25, 2022   11:00 AM

https://github.com/t4d-starter-projects/cookiecutter-create-python-project

```
> python -m pip install --upgrade pip setuptools wheel
> python -m pip install cookiecutter
> mkdir rates_app
> cd rates_app

> cookiecutter https://github.com/t4d-starter-projects/cookiecutter-create-python-project
```



```
(python310) C:\Users\nir11152\github\advaced-python-course\rates_app>cookiecutter https://github.com/t4d-starter-project
s/cookiecutter-create-python-project
You've downloaded C:\Users\nir11152\.cookiecutters\cookiecutter-create-python-project before. Is it okay to delete and r
e-download it? [yes]: y
project_name []: rates_client
project_feed []:
project_branch [main]:
package_name []: rates_client
package_desc [A new package.]:
author_name []: Nirvan Theehthira
author_email []: nstp6666@gmail.com
author_url []: https://www.t4d.io
```

```
> cd rates_server
> python -m venv venv
> .\venv\Scripts\activate.bat
> python -m pip install --upgrade pip setuptools wheel
> python -m pip install -r requirements.txt
> deactivate
```

Inside rates_server in rates server virtual env
> python -m pip install -e ../rates_shared

(-e means the files of the package being installed is still beign edited)
(rates_shared can be edited and its changes will be reflected in rates-server without having to rerun pip install)

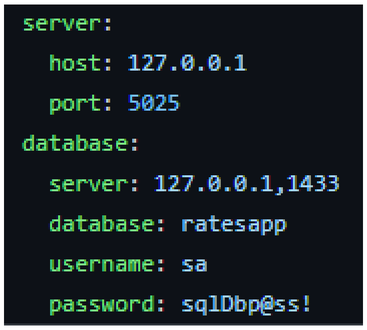
↓
once this is done, rates-shared can be imported in rates-server
from rates-shared import ...

[Do the same for rates_client]

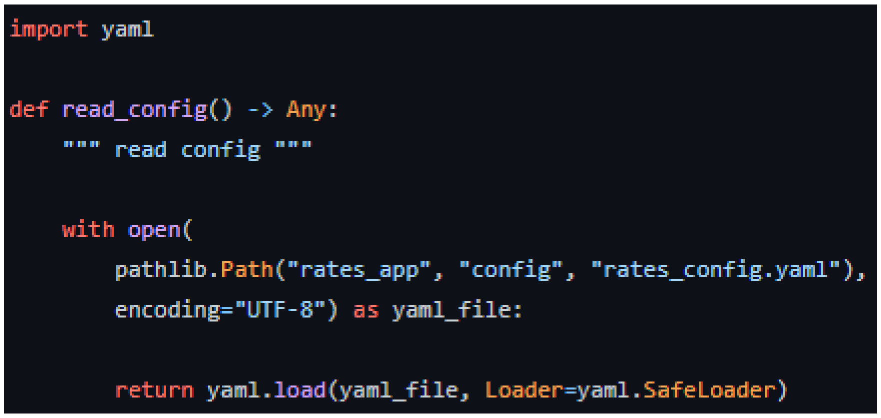
Yaml config file

Friday, March 25, 2022 1:33 PM

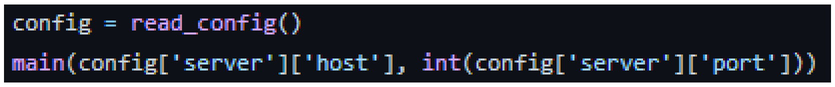
/python_demos/rates_app/config/rates_config.yaml



```yaml
server:
  host: 127.0.0.1
  port: 5025
database:
  server: 127.0.0.1,1433
  database: ratesapp
  username: sa
  password: sqlDbp@ss!
```



```python
import yaml

def read_config() -> Any:
    """ read config """

    with open(
        pathlib.Path("rates_app", "config", "rates_config.yaml"),
        encoding="UTF-8") as yaml_file:

        return yaml.load(yaml_file, Loader=yaml.SafeLoader)
```



```python
config = read_config()
main(config['server']['host'], int(config['server']['port']))
```

Testing

Friday, March 25, 2022 3:20 PM

Unit test -> test a single unit of code and a single unit only. Mock other units that are required for the single unit being tested. Don't read from files or network, mock it in unit test

Integration test -> test interoperability between units of code, read from files, access databases, network calls

End-to-end test -> Test functionality of entire application. essentially a bot that interacts with application

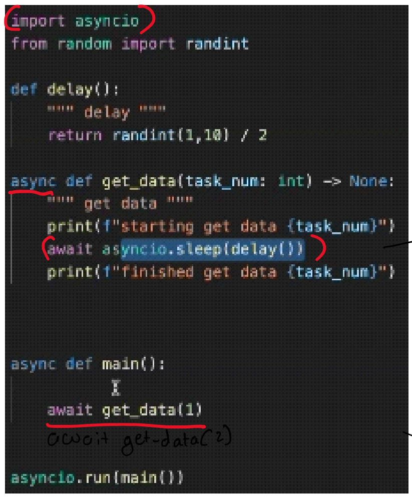
github actions

azure dev ops

Package wheel file. can be run from Command line. eg: pip

Async

Friday, March 25, 2022 3:45 PM



```python
import asyncio
from random import randint

def delay():
    """ delay """
    return randint(1,10) / 2

async def get_data(task_num: int) -> None:
    """ get data """
    print(f"starting get data {task_num}")
    await asyncio.sleep(delay())
    print(f"finished get data {task_num}")


async def main():

    await get_data(1)
    await get_data(2)

asyncio.run(main())
```

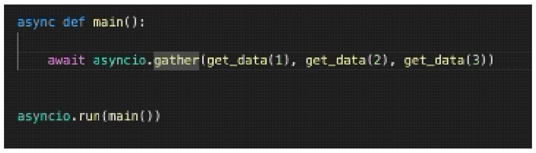
→ Does not stop main thread takes work off to another thread, Completes io operation

→ output

```
starting ... 1
finished ... 1
starting ... 2
finished ... 2
```

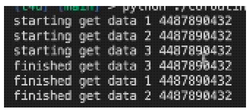
|→ waits for 1'st to finish, await

|→ waits for 2'nd to finish, await



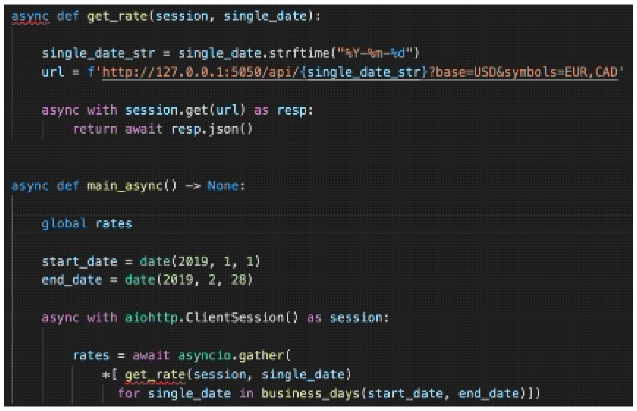
```python
async def main():

    await asyncio.gather(get_data(1), get_data(2), get_data(3))


asyncio.run(main())
```

→ Starts all three first, await for all three to finish



```
starting get data 1 4487890432
starting get data 2 4487890432
starting get data 3 4487890432
finished get data 3 4487890432
finished get data 1 4487890432
finished get data 2 4487890432
```

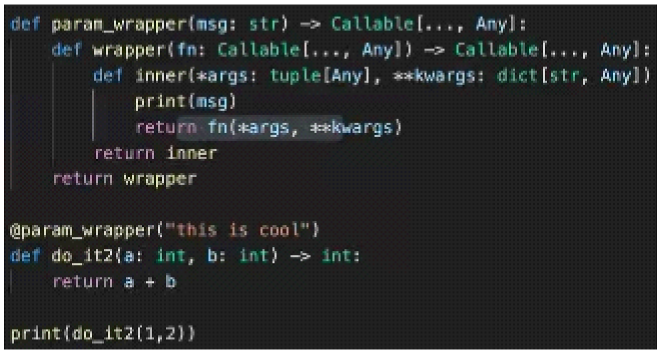
HTTP requests with async



```python
async def get_rate(session, single_date):

    single_date_str = single_date.strftime("%Y-%m-%d")
    url = f'http://127.0.0.1:5050/api/{single_date_str}?base=USD&symbols=EUR,CAD'

    async with session.get(url) as resp:
        return await resp.json()


async def main_async() -> None:

    global rates

    start_date = date(2019, 1, 1)
    end_date = date(2019, 2, 28)

    async with aiohttp.ClientSession() as session:

        rates = await asyncio.gather(
            *[ get_rate(session, single_date)
              for single_date in business_days(start_date, end_date)])
```

Parameterized wrappers

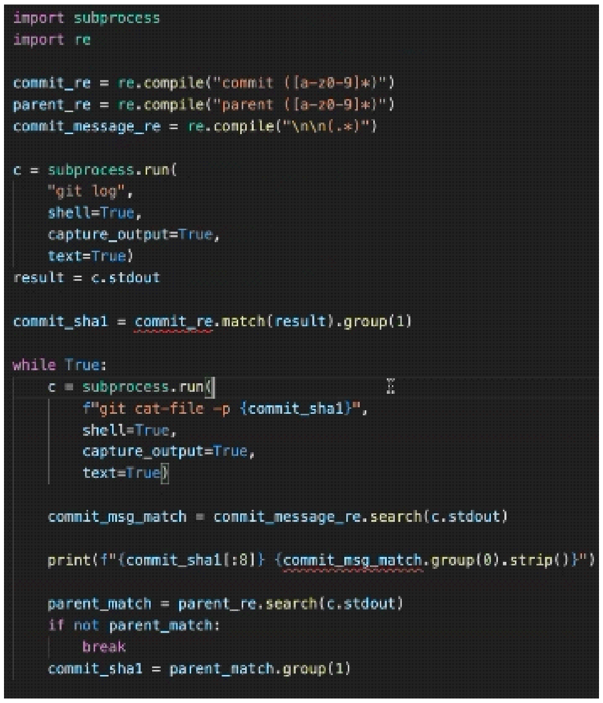
Friday, March 25, 2022 4:32 PM



```python
def param_wrapper(msg: str) -> Callable[..., Any]:
    def wrapper(fn: Callable[..., Any]) -> Callable[..., Any]:
        def inner(*args: tuple[Any], **kwargs: dict[str, Any])
            print(msg)
            return fn(*args, **kwargs)
        return inner
    return wrapper

@param_wrapper("this is cool")
def do_it2(a: int, b: int) -> int:
    return a + b

print(do_it2(1,2))
```

Suprocesses

Friday, March 25, 2022 4:33 PM



```python
import subprocess
import re

commit_re = re.compile("commit ([a-z0-9]*)")
parent_re = re.compile("parent ([a-z0-9]*)")
commit_message_re = re.compile("\n\n(.*)")

c = subprocess.run(
    "git log",
    shell=True,
    capture_output=True,
    text=True)
result = c.stdout

commit_sha1 = commit_re.match(result).group(1)

while True:
    c = subprocess.run(
        f"git cat-file -p {commit_sha1}",
        shell=True,
        capture_output=True,
        text=True)

    commit_msg_match = commit_message_re.search(c.stdout)

    print(f"{commit_sha1[:8]} {commit_msg_match.group(0).strip()}")

    parent_match = parent_re.search(c.stdout)
    if not parent_match:
        break
    commit_sha1 = parent_match.group(1)
```

Subprocess runs another program

c example runs git program