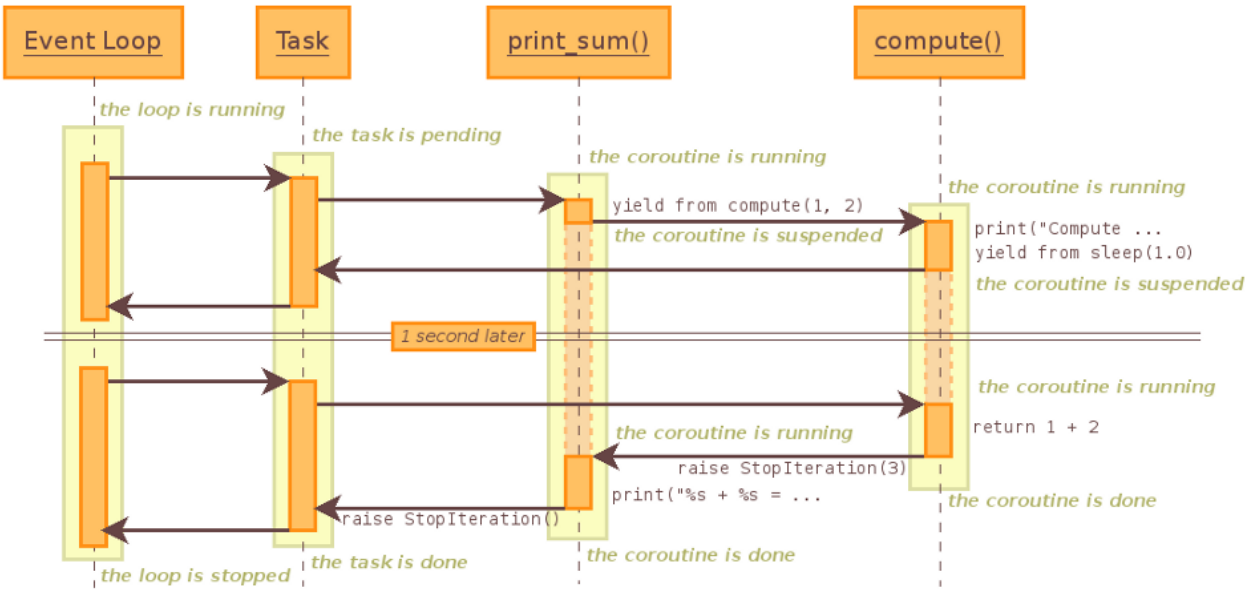
Теория к лабораторной работе №3 по ЯП

Асинхронное программирование. Основы ASINCIO



Решение IO-bound задач

Работа программы осуществляется в одном потоке.

Корутина –подпрограмма (функция), которая может начинаться, приостанавливаться и завершаться в произвольный момент времени.

- async - флаг того, что данная функция является асинхронной(корутиной)
- await – флаг того, что данная функция встает на паузу и дает работать другим функциям, пока сама получает какие-то данный (например)
- event_loop – механизм по контролированию и управлению коррутинами.

Сейчас же event_loop находится внутри asyncio.run(main()), где и создается список событий

Создание корутины main:

1 способ

(указываем напрямую пул задач, которые закидываются в event_loop). При этом event_loop способен определять задачи, которые надо «будить» и «усыплять» (await). Например, система сообщает о получении данных с сервера.

```
async def main():
    asyncio.create_task(one())
    asyncio.create_task(two())
    await asyncio.create_task(three())
```

2 способ

Сборка задач через asyncio.gather()

```
async def main():
    await asyncio.gather(one(), two(), three())
```

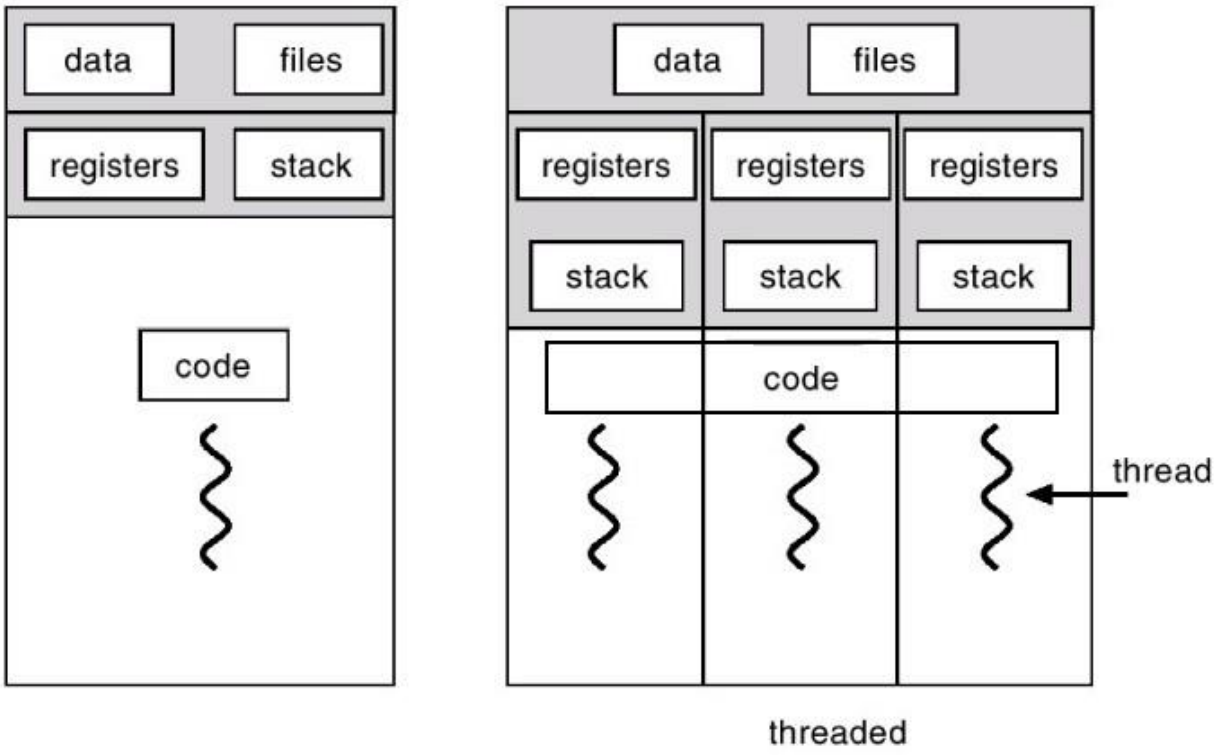
Внутри корутин следует использовать асинхронные аналоги известных библиотек(не блокирующего кода).

+ управляемость

+ ресурсы

- смерть из-за одного блокирующего потока

Многопоточность



Решение IO-bound задач

GIL всегда влияет на потоки

Программа = процесс (в нем содержатся разные потоки)

- Потоки создаются при помощи класса Thread

```
def thread_wait(timeout):
    thread = Thread(target=waiting, args=(timeout,), daemon=True)
    thread.start()
    return thread
```

- Запуск потоков

```
threads = [Thread(target=info, daemon=True) for _ in range(10)]
for t in threads:
    t.start()
for t in threads:
    t.join()
```

Гонка за ресурсы (Race condition)

Способы разрешения:

1 способ – создание блокировки потоков lock (возможен вариант проблемы – dead lock)

```
counter = [0]
lock = threading.Lock()

1 usage
def inc():
    lock.acquire()
    c = counter[0]
    time.sleep(0.1)
    counter[0] = c + 1
    lock.release()
```

2 способ – создание очереди queue

```
queue = Queue()
queue.put(0)
```

```
def inc_queue():
    c = queue.get()
    queue.put(c+1)
```

3 способ – executor

```
executor = ThreadPoolExecutor()
for _ in range(10):
    executor.submit(activity)
```

```
executor.shutdown(wait=True)
```

+ быстро

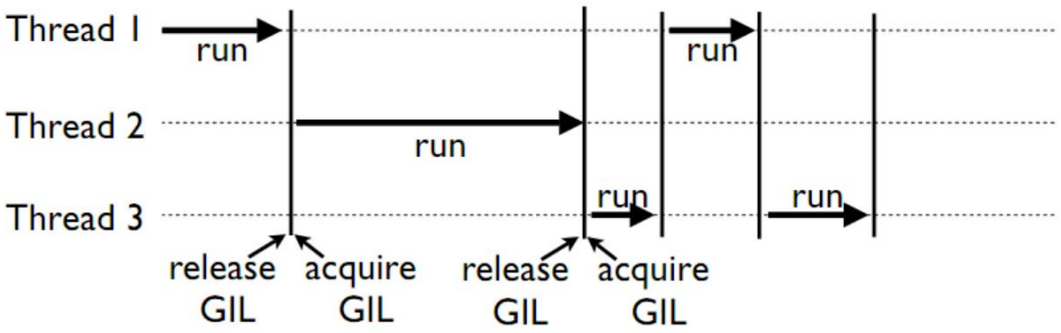
+ нет смерти из-за одного потока

- потребление ресурсов

- неуправляемость (когда какой поток работает)

- проблемы потоков: race condition, dead lock

Многопроцессорность



Решение IO-bound задач

Решение CPU-bound задач

Каждый процесс – отдельная программа

Процессы взаимодействуют посредством сериализации (pickle) – превращения объектов python в байты

Важно использовать Pull

По синтаксису схож с мультипоточностью

+ параллельность

+ процессы не зависят друг от друга

- потребление

- необходимость сериализации в pickle

- сложность взаимодействия

Парсинг сайтов

Библиотека requests

Get запросы

Все параметры передаются в url строке

- Отправить get запрос сайту response = requests.get(“сайт”)
- Узнать статус запроса response.status_code
  - 200-300 – успешно
  - 300-400 – перенаправление
  - 400+ - ошибка
- Узнать данные страницы response.text
- Посмотреть словарь данных response.json

Post запросы

- response = requests.post()

Класс Session

variable = requests.Session() – нужно для подкачки cookie (взаимодействия между запросами)

Подготовка

Для установки pyinstaller необходимо изменить переменные среды. В Path добавить C:\Users\storo\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2kfra8p0\LocalCache\local-packages\Python312\Scripts\pyinstaller.exe --onefile script.py

Сборка файла spec

pyinstaller --onefile script.py

Основные опции команды:

--onefile — Собирает все в один исполняемый файл (удобно для распространения).

--windowed (или -w) — Используйте, если у вас графический интерфейс (например, с использованием Tkinter или PyQt), чтобы при запуске не появлялось окно консоли.

--icon=path_to_icon.ico — Задает иконку для созданного .exe файла.

Сборка файла exe

pyinstaller имя_файла.spec