ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ БИБЛИОТЕКИ ASYNCIO

ВВЕДЕНИЕ

Задачи урока

01 Как можно проверить наличие event loop в потоке

02 Как можно запускать потоки с библиотекой gevent

02 Как можно объявлять корутины с использованием декоратора

get_event_loop позволяет проверить наличие event_loop в текущем потоке:

```
...
if asyncio.get_event_loop() is None:
        asyncio.set_event_loop(asyncio.new_event_loop())

    event_loop = asyncio.get_event_loop()
    event_loop.run_until_complete(parser("site 3"))
    event_loop.close()
```

Gevent является библиотекой для программирования событийно-ориентированных приложений в Python. Она предоставляет мощные инструменты для работы с сетевыми операциями и многопоточностью, позволяя создавать эффективные и отзывчивые приложения

Модуль **gevent.spawn** является одним из ключевых инструментов в Gevent для создания "зеленых" (green) потоков

"Зеленые" потоки в Gevent

представляют собой легковесные потоки, которые выполняются внутри основного потока Python и могут блокироваться вводом-выводом (I/O) без блокировки всего приложения. Это позволяет создавать исполнение на основе событий, где каждый "зеленый" поток может быть заблокирован на операциях ввода-вывода, позволяя другим потокам продолжить свою работу

```python
import gevent
import time


def parser(site_name):
    gevent.sleep(2)
    print(site_name)

if __name__ == '__main__':
    jobs = [gevent.spawn(parser, site) for site in
    ["site 1", "site 2", "site 3"] ]
    gevent.wait(jobs)

    gevent.joinall([gevent.spawn(parser, site) for
    site in ["site 1", "site 2", "site 3"] ])
```

gevent.wait()
является функцией в библиотеке Gevent, которая позволяет ожидать наступления событий в группе "зеленых" потоков (greenlets) и блокирует выполнение основного потока до тех пор, пока не будут выполнены все задачи или пока не будет получен сигнал о выходе из блокировки

gevent.joinall()
это функция в библиотеке Gevent, которая позволяет ожидать завершения всех "зеленых" потоков (greenlets) и блокирует выполнение основного потока до тех пор, пока все потоки не завершатся или пока не будет получен сигнал о выходе из блокировки

gevent.wait() и gevent.joinall() являются двумя разными функциями в библиотеке Gevent, которые используются для ожидания завершения "зеленых" потоков (greenlets), но имеют некоторые отличия в своем поведении

Функция gevent.wait()
используется для блокировки выполнения основного потока до тех пор, пока хотя бы один из переданных потоков не завершится

Функция gevent.joinall()
также используется для блокировки выполнения основного потока до тех пор, пока все переданные потоки не завершатся

@asyncio.coroutine
это декоратор, используемый в библиотеке asyncio в Python для определения корутины

Корутина
это специальный тип функции, который может быть приостановлен и возобновлен во время своего выполнения, что позволяет выполнять асинхронное программирование

```
@asyncio.coroutine
def parser(site_name):
    asyncio.sleep(2)
    print(site_name)
    return site_name
```

ИТОГИ

- С использованием декоратора **@asyncio.coroutine** можно создать корутину
- С библиотекой **gevent** можно запускать “зеленые” потоки и ожидать их выполнения
- С помощью **asyncio.get_event_loop()** можно определить есть ли **event_loop** в текущем потоке