Обходной путь для CPU bound операций

> run_in_executor/cpu_workaround.py

```
import asyncio
import datetime


async def blocking_task():
    counter = 50000000
    while counter > 0:
        counter -= 1
        if counter % 1000 == 0:
            await asyncio.sleep(0)


async def blocking_worker():
    while True:
        await blocking_task()


async def ticker():
    while True:
        print(datetime.datetime.now())
        await asyncio.sleep(1)


if __name__ == '__main__':
    loop = asyncio.get_event_loop()
    loop.create_task(ticker())
    loop.create_task(blocking_worker())
    loop.run_forever()
```

Чтобы работала кооперативная многозадачность, нужно, чтобы контекст переключался между короутинами. Это происходит в момент выполнения вызова асинхронной функции с синтаксисом await.

В случае с бесконечным while True циклом управление никогда не вернется в event loop. Это потому, что никогда не будет вызвана асинхронная функция.

Поэтому мы вызовем ее искусственно. Для этого вызовем **await asyncio.sleep** со временем ожидания **0**. Вызывать ее можно управляемо **раз в N** итераций цикла.

В итоге получается:
— cpu операция будет тратить время процессора;
— но при этом будет периодически передавать управление другим короутинам для выполнения и вставать обратно в очередь event loop.

При таком подходе мы получим идентичный с ThreadPoolExecutor результат. Только в этом случае уже не операционная система решает, в какой момент отдать управление event loop, а мы сами.

Если запустить **cpu_workaround.py**, получим:

```
coder@code-server-admin-38-94c54d494-m248j:~/mercury$ python3 run_in_executor/cpu_workaround.py
2021-09-30 11:20:01.635952
2021-09-30 11:20:02.654074
2021-09-30 11:20:03.657051
2021-09-30 11:20:04.662023
2021-09-30 11:20:05.662572
2021-09-30 11:20:06.664150
```

При этом, если переключаться слишком часто, время исполнения cpu-операции будет слишком замедляться. А если переключаться слишком редко, то начнет страдать асинхронность:

```
async def blocking_task():
    counter = 50000000
    while counter > 0:
        counter -= 1
        if counter % 1000000 == 0:
            await asyncio.sleep(0)
```

```
coder@code-server-admin-38-94c54d494-m248j:~/mercury$ python3 run_in_executor/cpu_workaround.py
2021-09-30 11:23:38.422682
2021-09-30 11:23:40.233690
2021-09-30 11:23:41.766207
2021-09-30 11:23:43.244913
```

Интервал времени превысил одну секунду.