Quart-Trio

A Trio based HTTP Framework

Philip Jones

Dec 19 · 2 min read

Quart-Trio is an extension to the Quart web micro-framework that supports the Trio event loop, rather than the asyncio event loop. To use Quart-Trio simply install via pip, `$ pip install quart-trio` then write your Quart application using QuartTrio as so,

```
from quart_trio import QuartTrio

app = QuartTrio(__name__)

@app.route("/")
async def hello():
    return "world"

app.run()
```

Why Trio?

Trio is an alternative event loop implementation based on async/await and coroutines. This means it offers a way to write concurrent code, e.g.

```
async def say(message, delay):
    await trio.sleep(delay)
    print(message)

async def main():
    async with trio.open_nursery() as nursery:
        nursery.start_soon(say, "hello", 1)
        nursery.start_soon(say, "world", 0.5)

trio.run(main)
```

to give,

```
>>> trio.run(main)
world
hello
```

What makes Trio special though is its focus on the API. Trio is easier to understand, especially when compared with asyncio. This means that any applications built with Quart-Trio will be simpler and better reasoned.

A real example

I've prepared a simple chat application to demonstrate how to use Quart-Trio, the key to this example is this function,

```
@app.websocket("/ws")
async def chat():
    try:
        connections.add(websocket._get_current_object())
        async with trio.open_nursery() as nursery:
            nursery.start_soon(heartbeat)
            while True:
                message = await websocket.receive()
                await broadcast(message)
    finally:
        connections.remove(websocket._get_current_object())
```

This is a very succinct way to create a background task (the heartbeat) with confidence that it will cancelled when the connection is closed. In addition any exception raised in the background task will also trigger the finally clause as you'd expect. This is a non-trivial exercise with asyncio.

Conclusion

Quart-Trio is ready to be used, and I think it could be the first web framework to support Trio. In addition Hypercorn supports a Trio worker, thereby providing all the tools needed for production ready Trio web applications.