# aiohttp

阅读: 132754

`asyncio` 可以实现单线程并发IO操作。如果仅用在客户端，发挥的威力不大。如果把 `asyncio` 用在服务器端，例如Web服务器，由于HTTP连接就是IO操作，因此可以用单线程+ `coroutine` 实现多用户的高并发支持。

`asyncio` 实现了TCP、UDP、SSL等协议，`aiohttp` 则是基于 `asyncio` 实现的HTTP框架。

我们先安装 `aiohttp` ：

```
pip install aiohttp
```

然后编写一个HTTP服务器，分别处理以下URL：

- `/` - 首页返回 `b'<h1>Index</h1>'` ；
- `/hello/{name}` - 根据URL参数返回文本 `hello, %s!` 。

代码如下：

```
import asyncio

from aiohttp import web

async def index(request):
    await asyncio.sleep(0.5)
    return web.Response(body=b'<h1>Index</h1>')

async def hello(request):
    await asyncio.sleep(0.5)
    text = '<h1>hello, %s!</h1>' % request.match_info['name']
    return web.Response(body=text.encode('utf-8'))

async def init(loop):
    app = web.Application(loop=loop)
    app.router.add_route('GET', '/', index)
    app.router.add_route('GET', '/hello/{name}', hello)
    srv = await loop.create_server(app.make_handler(), '127.0.0.1', 8000)
    print('Server started at http://127.0.0.1:8000...')
    return srv

loop = asyncio.get_event_loop()
loop.run_until_complete(init(loop))
loop.run_forever()
```

注意 `aiohttp` 的初始化函数 `init()` 也是一个 `coroutine` ，`loop.create_server()` 则利用 `asyncio` 创建TCP服务。

参考源码

aio_web.py

读后有收获可以请作者喝咖啡，读后有疑问请加群讨论：

还可以分享给朋友：

📤 分享到微博

评论

**教程过时了，**
栗子是逗比 created at June 12, 2018 9:42 AM, Last updated at 5 hours ago

```
from aiohttp import web

routes = web.RouteTableDef()

@routes.get('/')
async def index(request):
    await asyncio.sleep(2)
    return web.json_response([
        {'name': 'index'}
    ])

@routes.get('/about')
async def about(request):
    await asyncio.sleep(0.5)
    return web.Response(text="<h1>about us</h1>")

def init():
    app = web.Application()
```

⌄ Read More

viper1090
Created at July 25, 2018 11:32 AM

赞一个

流逝的仲夏夜之梦
Created at 5 hours ago

教程确实过时了，和aiohttp官方文档不一样

☰ 全部讨论　↩ 回复

**app.make_handler()这个方法用不了，改用APPrunner**
喵喵111111 created at May 11, 2019 12:10 AM, Last updated at May 11, 2019 12:14 AM

app.make_handler()这个方法用不了，改用APPrunner，虽然我没看懂，但是官方替换是这样得，有大神解析噢，代码能正常运行。

import asynciofrom aiohttp import web async def index(request):　　await asyncio.sleep(0.5)　　return web.Response(body=b'<h1>Index</h1>', content_type='text/html', charset='utf-8')async def hello(request):　　await asyncio.sleep(0.5)　　text = '<h1>hello, %s!</h1>' % request.match_info['name']　　return web.Response(body=text.encode('utf-8'))async def init():　　app = web.Application()　　app.router.add_get('/', index)　　app.router.add_get('/hello/{name}', hello)　　runner = web.AppRunner(app)　# app.make_handler()这个方法用不了，改用APPrunner，虽然我没看懂，但是官方替换是这样得　　await runner.setup()　　site = web.TCPSite(runner, '127.0.0.1', 8000)　　await site.start()　　print('Server started at http://127.0.0.1:8000...')　　return siteloop = asyncio.get_event_loop()loop.run_until_complete(init())loop.run_forever()

喵喵111111
Created at May 11, 2019 12:14 AM

https://aiohttp.readthedocs.io/en/stable/web_advanced.html#aiohttp-web-app-runners

☰ 全部讨论　↩ 回复

**pycharm 导包报错是为什么啊**
DanboDanbo created at October 29, 2018 2:33 PM, Last updated at April 13, 2019 10:22 PM

from aiohttp import web 这个不会报这这个错 This inspection detects names that should resolve but don't. Due to dynamic dispatch and duck typing, this is possible in a limited but useful number of cases. Top-level and class-level items are supported better than instance items.

何小鸭___
Created at April 13, 2019 10:22 PM

你需要导入,aiohttp-webpack包

☰ 全部讨论　↩ 回复

**打卡——第15天**
ny12348 created at April 4, 2019 12:42 PM, Last updated at April 4, 2019 12:42 PM