python异步编程之asyncio (百万并发)

136\*\*\*\*9024 / 2018-06-12 23:17:31 / 浏览数 6017 安全技术 技术讨论 顶(0) 踩(0)

```
— asyncio
```

下面通过举例来对比同步代码和异步代码编写方面的差异,其次看下两者性能上的差距,我们使用sleep(1)模拟耗时1秒的io操作。

#### 同步代码:

输出: (间隔差不多是1s)

Hello World:1527595175.4728756 Hello World:1527595176.473001 Hello World:1527595177.473494 Hello World:1527595178.4739306 Hello World:1527595179.474482

#### 异步代码:

#### 输出:

Hello World:1527595104.8338501 Hello World:1527595104.8338501 Hello World:1527595104.8338501 Hello World:1527595104.8338501 Hello World:1527595104.8338501

async def

用来定义异步函数,其内部有异步操作。每个线程有一个事件循环,主线程调用asyncio.get\_event\_loop()时会创建事件循环,你需要把异步的任务丢给这个循环的run\_unt

二. aiohttp

如果需要并发http请求怎么办呢,通常是用requests,但requests是同步的库,如果想异步的话需要引入aiohttp。这里引入一个类,from aiohttp import ClientSession,首先要建立一个session对象,然后用session对象去打开网页。session可以进行多项操作,比如post, get, put, head等。

#### 基本用法:

```
async with ClientSession() as session:
    async with session.get(url) as response:
```

## aiohttp异步实现的例子:

# 首先async def 关键字定义了这是个异步函数, await

关键字加在需要等待的操作前面, response.read()等待request响应, 是个耗IO操作。然后使用ClientSession类发起http请求。

## 多链接异步访问

如果我们需要请求多个URL该怎么办呢,同步的做法访问多个URL只需要加个for循环就可以了。但异步的实现方式并没那么容易,在之前的基础上需要将hello()包装在asyn

```
task = asyncio.ensure_future(hello(url.format(i)))
  tasks.append(task) if __name__ == '__main__':
loop = asyncio.get_event_loop()
run()
loop.run_until_complete(asyncio.wait(tasks))
```

import time import asyncio from aiohttp import ClientSession

#### 输出:

Hello World:1527754874.8915546 Hello World:1527754874.899039 Hello World:1527754874.90004 Hello World:1527754874.9095392 Hello World:1527754874.9190395

#### 收集http响应

好了,上面介绍了访问不同链接的异步实现方式,但是我们只是发出了请求,如果要把响应——收集到一个列表中,最后保存到本地或者打印出来要怎么实现呢,可通过asy

tasks = []
url = "https://www.baidu.com/{}" async def hello(url):
 async with ClientSession() as session:
 async with session.get(url) as response: # print(response) print('Hello World:%s' % time.time()) return await response.
 task = asyncio.ensure\_future(hello(url.format(i)))
 tasks.append(task)
 result = loop.run\_until\_complete(asyncio.gather(\*tasks)) print(result) if \_\_name\_\_ == '\_\_main\_\_':
 loop = asyncio.get\_event\_loop()

#### 输出:

run()

Hello World:1527765369.0785167 Hello World:1527765369.0845182 Hello World:1527765369.0910277 Hello World:1527765369.0920424 Hello World:1527765369.097017 [b'<!DOCTYPE html>\r\n\r\n\r\n\......

#### 异常解决

假如你的并发达到1000个,程序会报错:ValueError: too many file descriptors in select()。这个报错的原因是因为 Python 调取的 select 对打开的文件字符有最大长度限制。这里我们有两种方法解决这个问题:1.我们可以需要限制并发数量。一次不要塞那么多任务,或者限制最大并发数量。2.我们可以使用回

## 点击收藏 | 0 关注 | 1

上一篇:利用CVE-2017-8890实现... 下一篇:【译】Metasploit:Met...

#### 1. 4条回复



我是一片云\_l 2018-06-13 09:25:21

#### 用协程或者进程池

0 回复Ta



ftkahzmodan 2018-06-13 16:05:24

代码块这个排版看的很难受 能缓解一下吗。。。

0 回复Ta



wonderkun 2018-06-26 00:23:29

编码看的强迫症都犯了

0 回复Ta



北风飘然 2018-09-04 17:03:37

你限制并发数500-600 和标题百万并发 开玩笑么.....

0 回复Ta

登录 后跟帖

先知社区

# 现在登录

热门节点

技术文章

社区小黑板

目录

RSS <u>关于社区</u> 友情链接 社区小黑板