Asyncio Subprocess-Queue

December 16, 2020

1 Subprocess

Asyncio también nos permite crear y gestionar sub-procesos. Con Asyncio, se puede ejecutar un comando de *shell*.

[ ]:
```
import asyncio

async def run(cmd):
    proc = await asyncio.create_subprocess_shell(
        cmd,
        stdout=asyncio.subprocess.PIPE,
        stderr=asyncio.subprocess.PIPE)

    stdout, stderr = await proc.communicate()

    print(f'[{cmd!r} exited with {proc.returncode}]')
    if stdout:
        print(f'[stdout]\n{stdout.decode()}')
    if stderr:
        print(f'[stderr]\n{stderr.decode()}')

# asyncio.run(run('ls /home'))
```

2 Queue

Las colas asyncio son diseñadas para ser similares a clases del módulo `queue`. Sin embargo las colas de asyncio no son seguras para los hilos, son diseñadas para funcionar específicamente con código async/await.

[ ]:
```
import asyncio
import random
import time


async def worker(name, queue):
    while True:
        # Get a "work item" out of the queue.
```

```
        sleep_for = await queue.get()

        # Sleep for the "sleep_for" seconds.
        await asyncio.sleep(sleep_for)

        # Notify the queue that the "work item" has been processed.
        queue.task_done()

        print(f'{name} has slept for {sleep_for:.2f} seconds')


async def main():
    queue = asyncio.Queue()

    # Se genera tiempos aleatorios y los coloca en la pila
    total_sleep_time = 0

    for _ in range(20):
        sleep_for = random.uniform(0.05, 1.0)
        total_sleep_time += sleep_for
        queue.put_nowait(sleep_for)

    # Creo 3 tasks para procesar la cola concurrentemente
    tasks = []
    for i in range(3):
        task = asyncio.create_task(worker(f'worker-{i}', queue))
        tasks.append(task)

    # Esperar hasta que la cola es completamente procesadas
    started_at = time.monotonic()
    await queue.join()
    total_slept_for = time.monotonic() - started_at

    # cancelamos las tasks
    for task in tasks:
        task.cancel()
    # Esperamos hasta que todas las tareas estén canceladas.
    await asyncio.gather(*tasks, return_exceptions=True)

    print('====')
    print(f'3 workers slept in parallel for {total_slept_for:.2f} seconds')
    print(f'total expected sleep time: {total_sleep_time:.2f} seconds')


# asyncio.run(main())
```