PPDS - 8. zadanie

Aplikácia

V zadaní sme vytvorili aplikáciu, ktorá po zadaní hlavného mesta čerpá od servera informácie o krajinách, ku ktorým mestá prislúchajú.

Je zrejmé, že kľúčovým faktorom pri časovej efektivite bude schopnosť aplikácie odovzdať beh inej úlohe, kým daná úloha čaká na dodanie dát z vonku.

Synchrónna verzia

Pomocou knižnice urllib od servera získavame postupne informácie o krajinách pre jednotlivé mestá v get_data. Získané dáta vypisujeme pomocou print_data.

```
def get_data(city):
    try:
        data = r.urlopen('https://restcountries.eu/rest/v2/capital/' + city).read()
        print_data(json.loads(data)[0], city)
    except HTTPError:
        pass
```

Táto verzia aplikácie je čitateľná a zrejmá, dlhší komentár nie je potrebný.

Asynchrónna verzia

Synchrónnu verziu obohatíme o kľúčové slová async a await. Chceme, aby get_data bol koprogram, označíme s async. Vieme, že tu vieme získať čas, kým sa bude čakať na dodanie dát od servera.

Rovnako pomocou async označíme main, nakoľko main bude spravovať pomocou asyncio.gather konkurentné behy koprogramov get_data.

Kľúčové slovo await používame v momentoch, kde musíme čakať na dokončenie operácie predtým, než by sme chceli pokračovať vo vykonávaní ďalej.

Výkon

Obe verzie sme časovo merali. Nakoľko ide o interakciu s externým komponentom, časy sú vždy mierne iné. Po viacerých meraniach zisťujeme, že asynchrónna verzia nezrýchľuje beh aplikácie.

Zisťujeme, že knižnica urllib obsahuje blokujúce metódy, teda nepodporuje asynchrónne dopytovanie na domény.

Po úprave asynchrónnej verzie vymenením knižnice urllib za aiohttp vidíme, že rozdiel medzi synchrónnou a asynchrónnou verziou je badateľný:

```
Time elapsed: 7.39089822769165
```

je dĺžka trvania aplikácie v sekundách pre synchrónnu verziu a

```
Time elapsed: 1.2154221534729004
```

je dĺžka trvania aplikácie v sekundách pre asynchrónnu verziu.

Dosiahli sme niekoľkonásobné zrýchlenie. Prečo?

- aplikácia okrem http requestov nič také nerobí
- a práve táto funkcionalita bola vylepšená na asynchrónny beh, preto taká pozitívne prudká zmena v čase

```
async with ClientSession() as s:
    async with s.get('https://restcountries.eu/rest/v2/capital/' + city) as r:
        print_data(await r.json(), city)
```

- kým sa čaká na s.get, úloha odovzdáva CPU inej úlohe
- ak, povedzme, trvá response od servera v priemere 2 sekundy, počas týchto dvoch sekúnd sa stihne vystriedať mnoho úloh. A to v štýle “vypýtam si dáta od servera a ihneď odovzdávam beh ďalšej úlohe kým sa mi server neozve”
- tým pádom prídu aj odpovede zo servera “naraz” za predpokladu stabilného výkonu servera a rýchlosti siete
- výsledkom je minimalizácia čakacieho času úloh a maximalizácia časovej efektivity single-thread aplikácie

Celý zdrojový kód oboch verzií je dohľadateľný v PPDS repozitári.