API:er, JSON, HTTP-requests

Programmering 1

Enkel API

- Finns på länken https://jsonplaceholder.typicode.com/
- Finns olika resources, exempelvis /todos
- API:er har (nästan alltid) dokumentation, som säger hur API:n ska användas
- Till exempel kan man här få ut en mindre del av JSON-filen genom att lägga till /nr


```
[
    {
        userId: 1,
        id: 1,
        title: "delectus aut autem",
        completed: false
    },
    {
        userId: 1,
        id: 2,
        title: "quis ut nam facilis et officia qui",
        completed: false
    },
    {
        userId: 1,
        id: 3,
        title: "fugiat veniam minus",
        completed: false
    },
    {
        userId: 1,
        id: 4,
        title: "et porro tempora",
        completed: true
    },
    {
        userId: 1,
        id: 5,
        title: "laboriosam mollitia et enim quasi adipisci quia provident illum",
        completed: false
    },
```

JSON-filer

- JSON-filer kan innehålla några enkla datatyper och “objekt”, tolkas som dictionaries i Python
- Kan innehålla listor, vi ser en till höger med dictionaries i
- Kan vara svårlästa i webbläsare, finns extensions (t. ex. JSONVue)


```
[
    {
        userId: 1,
        id: 1,
        title: "delectus aut autem",
        completed: false
    },
    {
        userId: 1,
        id: 2,
        title: "quis ut nam facilis et officia qui",
        completed: false
    },
    {
        userId: 1,
        id: 3,
        title: "fugiat veniam minus",
        completed: false
    },
    {
        userId: 1,
        id: 4,
        title: "et porro tempora",
        completed: true
    },
    {
        userId: 1,
        id: 5,
        title: "laboriosam mollitia et enim quasi adipisci quia provident illum",
        completed: false
    },
```

Läsa in URL i Python

- Två relevanta bibliotek: urllib och json
- För att öppna en URL i Python använder vi urllib.request.urlopen()
- Vi kan sedan tolka datan som en JSON-fil med json.loads()
- I detta fall får vi en lista med dictionaries

```python
# importera relevanta bibliotek
import urllib.request
import json

urlsida = urllib.request.urlopen("https://jsonplaceholder.typicode.com/todos/") # öppna URL
data = urlsida.read() # spara all data från sidan
jsonData = json.loads(data) # läs datan som en JSON-fil. I detta fall får vi en lista

print(jsonData) # ger oss hela listan
print(jsonData[0]) # ger oss första elementet i listan (en dictionary)
print(jsonData[0]["title"]) # ger oss värdet för nyckeln "title" i första dictionaryn
```

```
{'userId': 1, 'id': 1, 'title': 'delectus aut autem', 'completed': False}
delectus aut autem _
```

Varför göra detta?

- Vi kan använda samma kod i våra Discord-botar!

- Vi kan använda Discord-meddelanden för att skicka API-requests
- Svaren vi får av API:n kan skickas som ett meddelande av boten

- Exempel:
  - Vi skickar meddelandet “5” i Discord
  - Boten använder 5:an i API-URL:en och skickar tillbaka svaret

Enkelt exempel med enkla API:n

```python
if message.content.startswith('::todo'):
    response = todo(message)
    await message.channel.send(response)
```

```python
def todo(message):
    todoSearch = message.content.split(' ')
    if len(todoSearch) < 2:
        return 'need number!!!'
    else:

        number = todoSearch[-1]
        todoUrl = 'https://jsonplaceholder.typicode.com/todos/' + number
        todoRaw = urllib.request.urlopen(todoUrl)
        todoJson = json.loads(todoRaw.read())
        return todoJson
```

SL:s API - Nycklar

- https://www.trafiklab.se/api/trafiklab-apis/sl/
- API:er kan behöva en nyckel för att anropas
- Boten behöver tillgång till denna nyckel!

- För SL:s API krävs ett konto
- Man registrerar sig och skapar ett projekt, där man säger vilka API:er man vill använda. Då får man API-nycklar

SL:s API - Parametrar

- Kan finnas parametrar i URL:en (till exempel number i förra exemplet)
- Parametrar kan vara frivilliga, läs dokumentationen!
- SL:s API har flera parametrar, grönmarkerade i exemplet nedan

https://api.sl.se/api2/realtimedeparturesV4.<FORMAT>?key=<DIN API NYCKEL>&siteid=<SITEID>&timewindow=<TIMEWINDOW>

- <SITEID> måste vi få från ett annat API på samma sida, tex. Universitetet har id 9203
- Kan alltså behöva anropa flera API:er för att göra “en sak”

SL:s API - Exempel

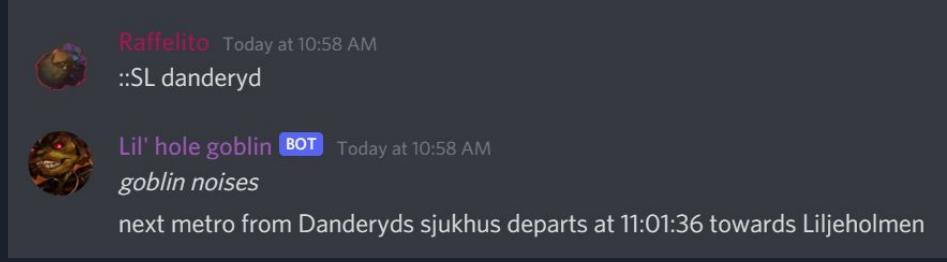
https://api.sl.se/api2/realtimedeparturesV4.json?key=7xa3xxf2e2xx4x9xxx3xx4xx9fx6exb&siteid=9203&timewindow=60

- Vi har en dictionary som vi kallar “jsonDict”
- Datan ligger i dictionary “ReponseData”
- Tunnelbanor ligger i listan “Metros”
- Varje avgång är en egen dictionary

- jsonDict[“ResponseData”] ger oss en ny dictionary
- jsonDict[“ResponseData”][“Metros”] ger oss en lista med info om om tunnelbanor från Universitet
- jsonDict[“ResponseData”][“Metros”][0] ger oss den första träffen som är en dictionary osv.


```
{
    StatusCode: 0,
    Message: null,
    ExecutionTime: 495,
    ResponseData: {
        LatestUpdate: "2022-01-31T11:32:58",
        DataAge: 6,
        Metros: [
            {
                GroupOfLine: "tunnelbanans röda linje",
                DisplayTime: "3 min",
                TransportMode: "METRO",
                LineNumber: "14",
                Destination: "Liljeholmen",
                JourneyDirection: 2,
                StopAreaName: "Universitetet",
                StopAreaNumber: 2231,
                StopPointNumber: 2232,
                StopPointDesignation: "2",
                TimeTabledDateTime: "2022-01-31T11:35:30",
                ExpectedDateTime: "2022-01-31T11:36:44",
                JourneyNumber: 20451,
                Deviations: null
            },
            {
                GroupOfLine: "tunnelbanans röda linje",
                DisplayTime: "6 min",
                TransportMode: "METRO",
                LineNumber: "14",
                Destination: "Fruängen",
```

SL:s API - Exempel




```
s = ''
for i in range (1,len(stationSearch)-1):
    s = s + stationSearch[i] + '+'
s = s + stationSearch[-1]

stationUrl = 'https://api.sl.se/api2/typeahead.json?key=4c123e93d7f147039fac4c15e0356257&searchstring=' + s
stationRaw = urllib.request.urlopen(stationUrl)
stationJson = json.loads(stationRaw.read())
stationID = stationJson['ResponseData'][0]['SiteId']

departureUrl = 'https://api.sl.se/api2/realtimedeparturesV4.json?key=10a7d32e89ec413994ee70661fa72a91&siteid=' + stationID
departureRaw = urllib.request.urlopen(departureUrl)
departureJson = json.loads(departureRaw.read())

departureDestination = departureJson['ResponseData']['Metros'][0]['Destination']
departureTime = departureJson['ResponseData']['Metros'][0]['ExpectedDateTime'][11:]
departureName = departureJson['ResponseData']['Metros'][0]['StopAreaName']
responseString = 'next metro from ' + departureName + ' departs at ' + departureTime + ' towards ' + departureDestination
return responseString
```