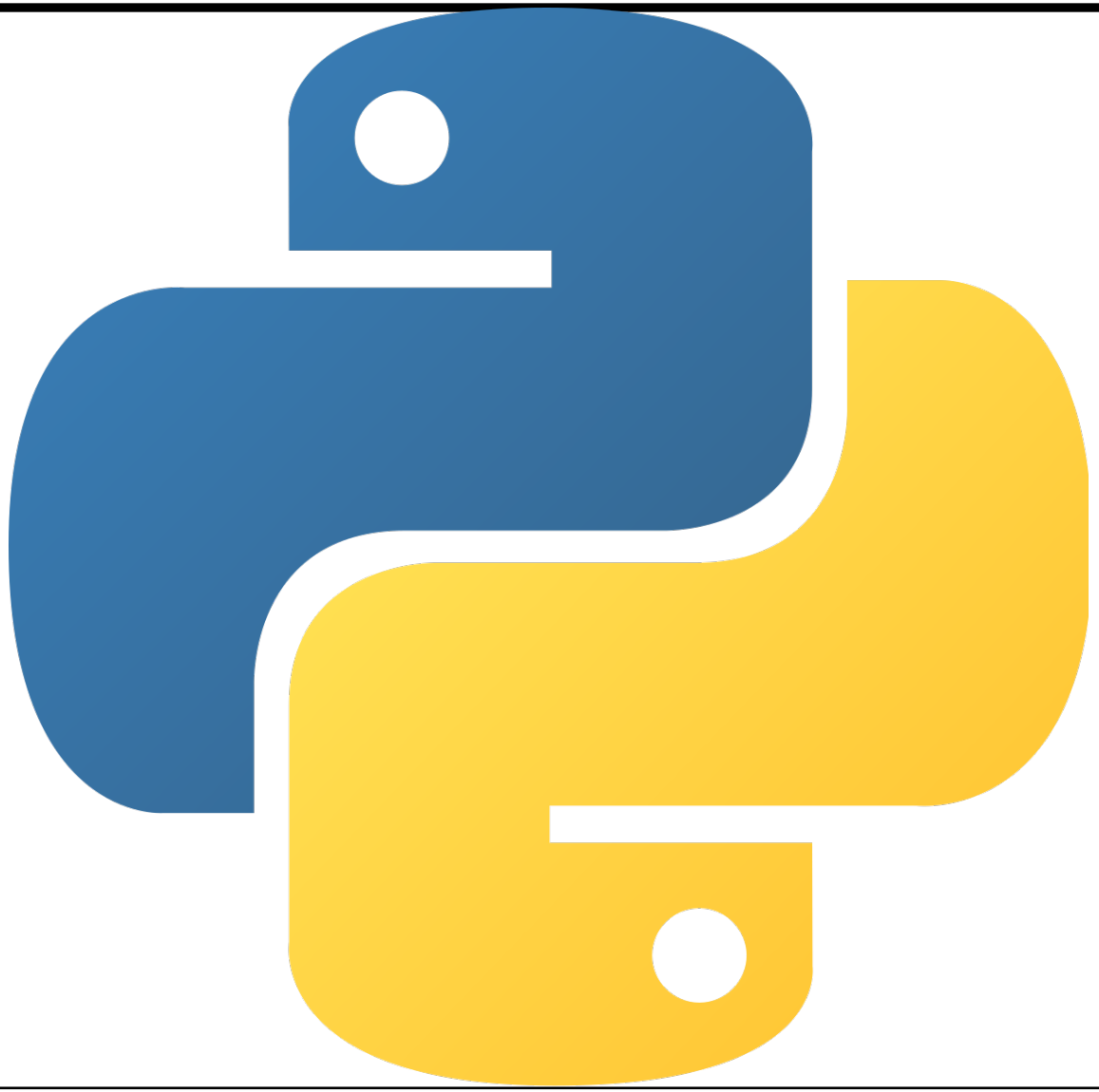
PYTHON PROGRAMMERING



Föreläsning 13

DAGENS FRÅGA

- Vilken statistik från ditt liv skulle du helst vilja se?

DAGENS AGENDA

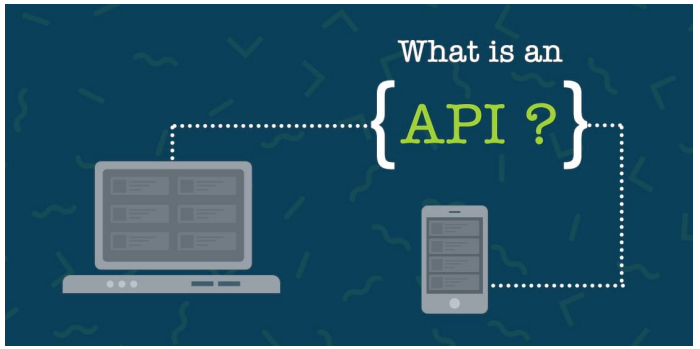
- Vad är ett API?
- API på internet – HTTP
- CRUD för HTTP
- HTTP request
- Localhost
- RESTful API
- Ännu ett objektorienterad programmerings exempel, pokemonhandler.py

FÖRRA FÖRELÄSNING

- Error och exceptions: skapa egna och hantera
- Debugging
- Att testa kod
- Unit testing
- Pytest
- Moduler, pip install och köra .py genom terminalen
- Argparse

VAD ÄR ETT API?

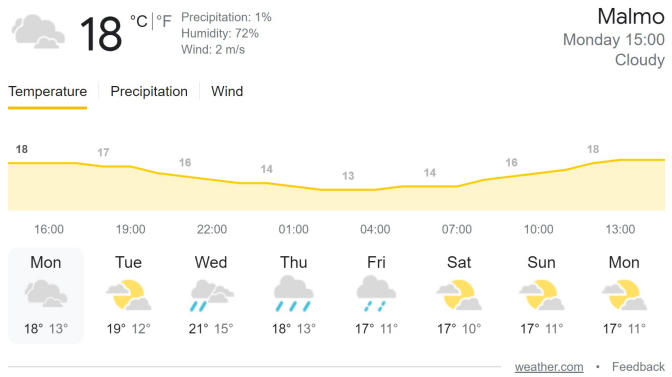
- API = Application Programming Interface
  - Ett API är ett sätt för två applikationer att kommunicera med varandra
  - Ni har redan använt API:er när ni använt pandas- och matplotlib-funktioner
  - Om man skriver en klass med metoder så är klassens metoder APIer
  - I data science är API:er viktiga när vi vill skapa ett system som har flera olika delar som behöver kommunicer.
  - Detta är oftast när vi vill hämta data



API – ETT EXEMPEL

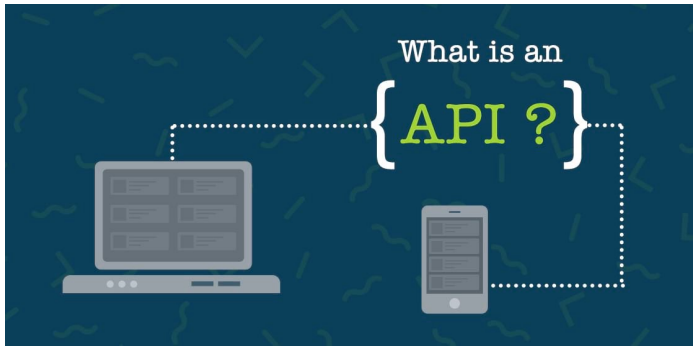
- Många appar ni använder dagligen använder sig av API:er för att hämta data, t.ex väderappen

1. Du öppnar din väderapp
2. Den skickar en request till en hemsida genom internet
3. Hemsidan är kopplad till en databas och drar ut rätt data, t.ex. Malmö idag
4. Din app får ett svar som innehåller datan den frågade efter
5. Appen kan nu visa datan på ett fint sätt



VAD ÄR ETT API?

- API är ett interface som bestämmer interaktionen mellan software
- API bestämmer vilka respons eller request som kan skapas, hur de ska skapas, dataformat som ska användas, konventioner man ska följa
- API underlättar. Man kan hämta information från ett program utan att behöva förstå hur det är konstruerad och fungerar.
- De som konstruerar APIet kan också bestämma vilken data som ska vara tillgänglig
- Man behöver bara veta reglerna till APIet



API PÅ INTERNET

- Web API är ett API över internet som kan nås med HTTP
- Många siter har öppna APIS där man kan använda för att ladda ner data
- **HTTP** är ett protokoll för hur data skickas över internet
  - När vi surfar på internet så skickas data mellan oss och servern via HTTP i form av HTML eller JSON data
- En **endpoint** är den serveradress som vi skickar vår HTTP request till
  - http://www.youtube.com
  - https://opendata-download-metfcst.smhi.se/
  - http://localhost:8000/users

```
<!DOCTYPE html>
<html>
    <head>
        <title>Example</title>
        <link rel="stylesheet" href="st
    </head>
    <body>
        <h1>
            <a href="/">Header</a>
        </h1>
        <nav>
            <a href="one/">One</a>
            <a href="two/">Two</a>
            <a href="three/">Three</a>
        </nav>
```

HTML

```
{
    "empid": "SJ011MS",
    "personal": {
        "name":"Smith Jones",
        "gender": "Male",
        "age": 28,
        "address":
        {
            "streetaddress": "7 24th Street",
            "city": "New York",
            "state": "NY",
            "postalcode": "10038"
        }
    },
    "profile": {
        "designation": "Deputy General",
        "department": "Finance"
    }
}
```

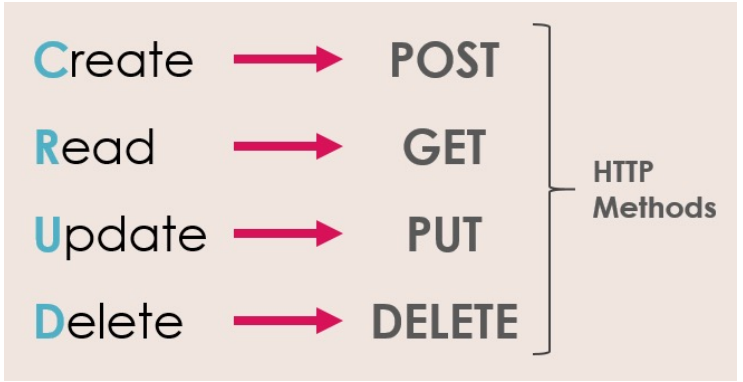

JSON

API PÅ INTERNET – EXEMPEL

- Till exempel Twitter sin API ger program tillgång till att läsa och skriva data, eller så kallade tweets
- Dessa tweetsen kan vi använda för att interagera Twitter med vår egna applikation

CRUD I HTTP

- CRUD är fyra metoder som är extremt vanliga i API:er och är bra att känna till
  - Create: Skickar data som skapar något
  - Read: Får tillbaka data
  - Update: Uppdaterar data
  - Delete: Tar bort data

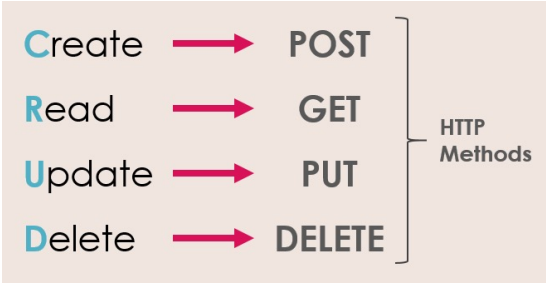


- I HTTP är de implementerade som metoderna
  - Post
  - Get
  - Put
  - Delete

INTERNET-API - EXEMPEL

- Simpelt exempel:
  - Vi vill exponera en enkel databas över frukter på adressen http://www.fruits.com/
  - Vi vill att man ska kunna använda databasen med ett API för att se vad som finns i databasen och för att kunna uppdatera den

- På endpointen http://www.fruits.com/fruits väljer vi att implementera alla CRUD-metoder
  - POST: Skickar en json med information om en frukt
  - GET: Returnerar en lista över alla frukter i databasen
  - PUT: Uppdaterar listan men nya frukter
  - DELETE: Tar bort frukter som inte längre är aktuella att ha med

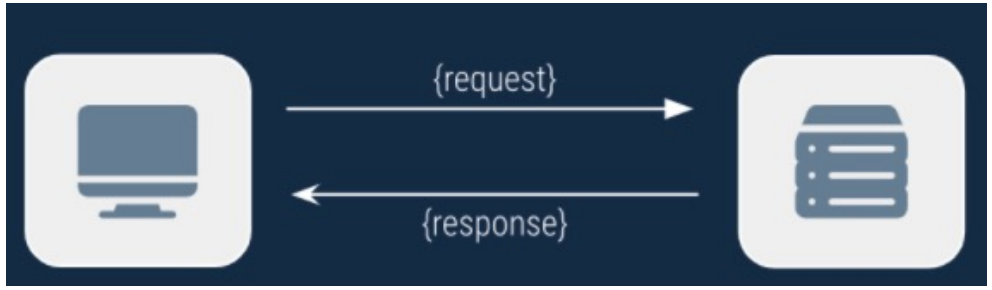


  - OBS: Vem som helst kan komma åt detta API, men POST/PUT/DELETE metoderna kan man inte veta hur fungerar utan att ha någon dokumentation

REQUESTS

- I Python kan vi skicka HTTP requests med det inbyggda paketet **requests**
- Det enda vi behöver göra är att bestämma vilken HTTP-metodvi ska använda och vilket endpoint vi vill skicka den till

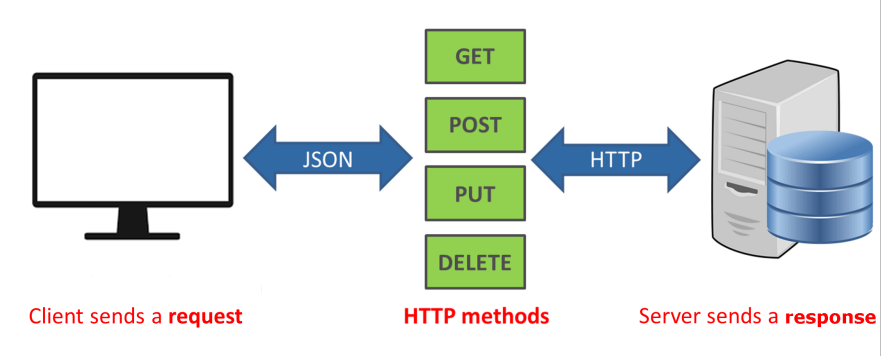
- HTTP-metoderna kallas
  - requests.get(url)
  - requests.post(url, data)
  - requests.put(url, data)
  - requests.delete(url, data_id)



LOCALHOST

- Vi kan köra egna servrar på våra maskiner för att testa att skicka requests
- Används för att testa sidor vid webutveckling i stället för riktiga webb-url:er
- Localhost har olika *portar*. Viktigt när man ska skicka en request till en lokal server
- Ex:
  - Jag servar en databas på http://localhost:8000
  - En webbapplikation på http://localhost:5000
  - AI modell på http://localhost:8080
- Detta är inte något ni kommer behöva implementera, men är nödvändigt att känna till

REST API

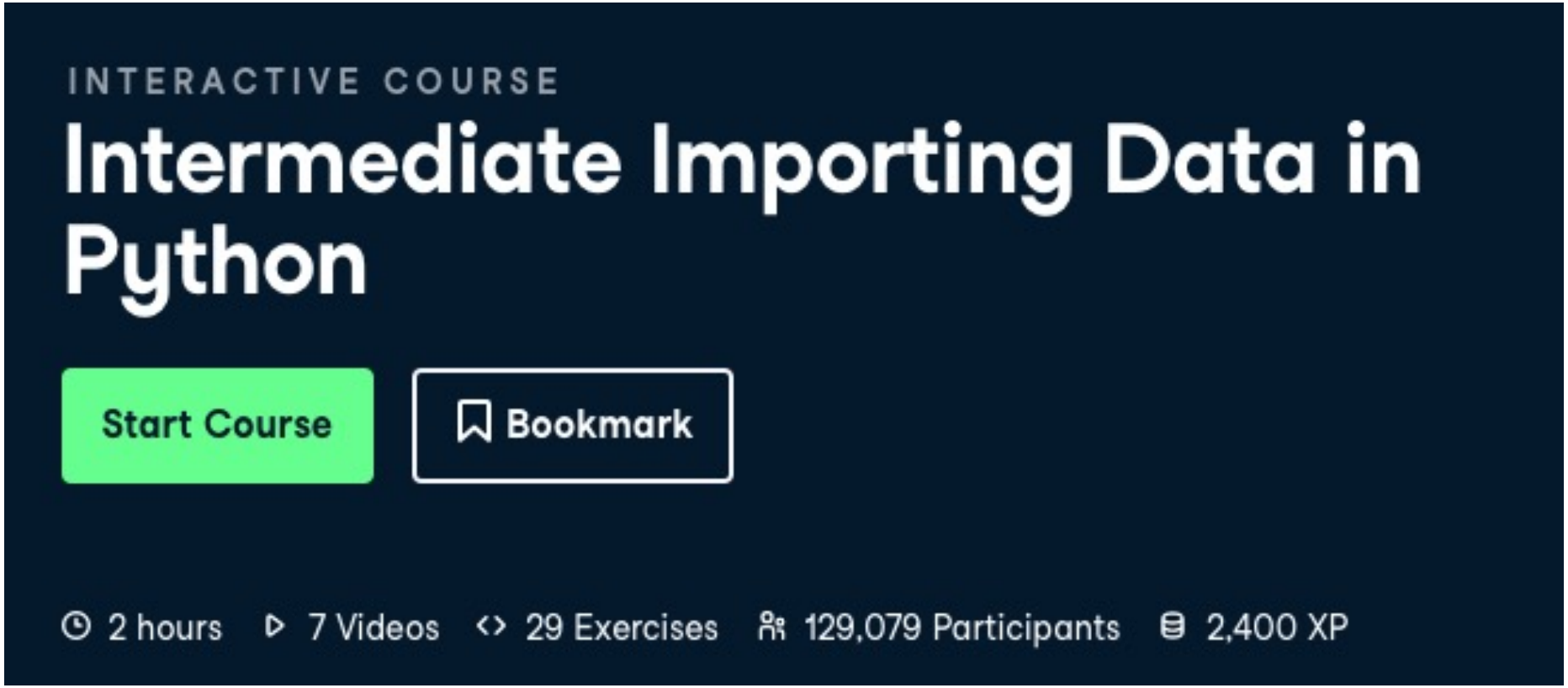


- REST API – Representational State Transfer
- RESTf API är en web arkitektur för att lösa API problem. Det finns många olika arkitekturer, men RESTful är den mest populära
- RESTful följer ingen protokoll, men följer fasta regler
- De flesta web-baserade implementationer av REST använder HTTP requests
- Resurserna i API hämtas med URIs
- URI är strings med tecken för att identifiera en resurs. URL är en URI och exempelvis http://example.com
- Grundtanken i REST är **resources**. All information som finns tillgänglig blir en resource (källa). Till exempel dokument, bilder, tjänst, sampling resurser, metoder
- Man kan se på det som en filsystem online med passiva resurser man hämtar
- I REST API ska samma request ska alltid ge samma respons
- REST API använder HTTP kommandon för att uppnå CRUD

RECAP

- Ett API är ett sätt att kommunicera med en annan applikation
- Det är vanligt att kommunicera med protokollet *HTTP*
- Det finns fyra viktiga metoder (*CRUD*): post, get, put, delete
- En HTTP-request skickas till en *endpoint* vilket är en specifik url
- I Python kan vi skicka HTTP-requests med paketet *requests*
- localhost är där era datorer servar endpoints så ni kan testa lokalt och detta kommer ni behöva använda i gruppuppgiften

DATA CAMP


INTERACTIVE COURSE
Intermediate Importing Data in Python
Start Course
Bookmark
2 hours
7 Videos
29 Exercises
129,079 Participants
2,400 XP