

---

# JSON

Introduktion til JSON og fetch

---

---

---

# Agenda

- Introduktion til JSON
  - Hent JSON-data ind fra en fil
  - Asynkrone events i JavaScript (promises)
  - indsætte JSON data i DOM
  - Arrays i objekter i array: Loop i loop
-

---

# JSON

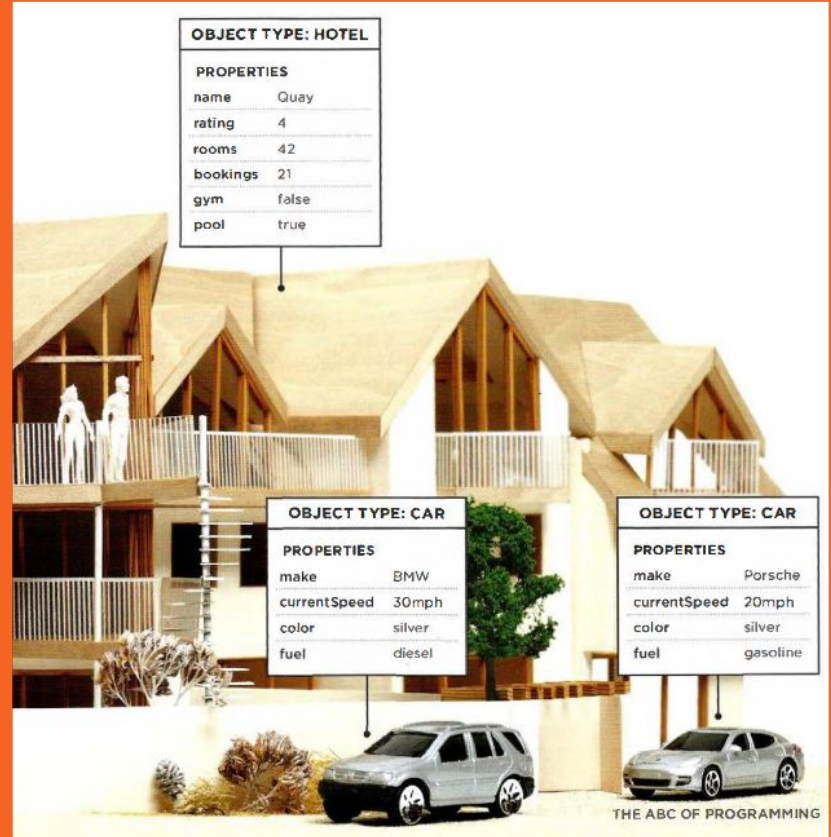
- JSON er en **syntaks** til **lagring** og **udveksling** af data.
- JSON er **tekst** skrevet med **J**ava**S**cript **O**bject **N**otation.
- Når man udveksler data mellem en browser og en server, kan data kun være **tekst**.
- Vi kan konvertere JavaScript-objekter til JSON og sende JSON til serveren.
- Og vi kan konvertere JSON modtaget fra en server til JavaScript-objekter i vores scripts.

[https://www.w3schools.com/js/js\\_json\\_intro.asp](https://www.w3schools.com/js/js_json_intro.asp)

---

# Hvad er det nu et objekt er...?

- En repræsentation af noget,
  - ofte et objekt i den "rigtige" verden som en person, et produkt, et køretøj el.lign. med en række fælles egenskaber.
- En abstraktion.
- En logisk gruppering.
- En datastruktur.



# Hvorfor JSON?

Fordi det er smart

Når data gemmes i særskilte datafiler, opnår vi at **adskille** html-kode mv. fra data:

- Struktur i html
- Layout i css
- Handling i JavaScript
- **Data i JSON**

---

# JSON eksempel

## JSON datatyper

- Strenger
- Tal
- Objekter
- Arrays
- Booleans
- Null

• biler.json (T7) — Brackets

```
[  
  {  
    "märke": "Volvo",  
    "model": "Amazon",  
    "motor": "Benzin",  
    "km": 500000  
  },  
  {  
    "märke": "VW",  
    "model": "Polo",  
    "motor": "Diesel",  
    "km": 40500  
  },  
  {  
    "märke": "Tesla",  
    "model": "S",  
    "motor": "El",  
    "km": 1000  
  }  
]
```

---

# JSON Syntaks { “ ” : “ ” }

Næsten lig med  
JavaScript objekt  
syntaks - men ikke  
helt.

- JSON-syntaks stammer som sagt fra JavaScript ObjektNotation:
    - Data er organiseret i nøgle / værdi-par {“id” : “007”}
    - Data adskilles med kommaer
    - Krøllede parenteser { } omkranser objekter
    - Firkantede parenteser [ ] indeholder arrays
  - Et nøgle/værdi-par består af et feltnavn i dobbelt citationstegn, efterfulgt af et kolon, efterfulgt af en værdi: {“navn” : “Klaus”}
  - I JSON skal nøgler (feltnavne) være en streng i dobbelt citationstegn!
  - I JavaScript *kan* nøgler være *uden* citationstegn: {navn: “Martin”}
-

# Objekter i arrays, arrays i objekter...

**NB!** Sæt IKKE  
komma efter sidste  
egenskab og efter  
sidste objekt!

```
undervisere.json (T7) — Brackets
1 ▼ [
2 ▼   {
3     "fornavn": "Martin",
4     "efternavn": "Bregnhøj",
5     "mail": "mabe@kea.dk",
6     "emner": ["JavaScript", "CSS", "Projektstyring"]
7   },
8 ▼   {
9     "fornavn": "Klaus",
10    "efternavn": "Mandal Hansen",
11    "mail": "klmh@kea.dk",
12    "emner": ["JavaScript", "HTML", "Tøjmode"]
13  },
14 ▼   {
15    "fornavn": "Louise Ea",
16    "efternavn": "Holbek",
17    "mail": "loeh@kea.dk",
18    "emner": ["SoMe", "SEO", "BMC"]
19  }
20 ]
```



# Adgang til værdierne i JSON

- Objekter og egenskaber nås ved hjælp af punktum .
- Arrays tilgås ved hjælp af firkantede parenteser []

undervisere.json (T7) — Brackets

```
1 ▼ [
2 ▼ {
3   "fornavn": "Martin",
4   "efternavn": "Bregnhøj",
5   "mail": "mabe@kea.dk",
6   "emner": ["JavaScript", "CSS", "Projektstyring"]
7 }
8 ▼ {
9   "fornavn": "Klaus",
10  "efternavn": "Mandal Hansen",
11  "mail": "klmh@kea.dk",
12  "emner": ["JavaScript", "HTML", "Tøjmode"]
13 }
14 ▼ {
15   "fornavn": "Louise Ea",
16   "efternavn": "Holbek",
17   "mail": "loeh@kea.dk",
18   "emner": ["SoMe", "SEO", "BMC"]
19 }
20 ]
```

```
const person = {
  "navn": "Martin",
  "titel": "Lektor"
}
```

Præcis på samme  
måde som med  
JavaScript objekter

```
let navn = person.navn;
let fornavn = undervisere[2].fornavn;
let emne = undervisere[0].emner[2];
```

# Hente og vise JSON data

Med fetch()

For at hente data fra en JSON-fil ind i vores HTML, skal vi bruge en *asynkron* JavaScript metode ved navn “fetch”

---

# Hvad er en asynkron metode?

Også kaldet AJAX

(Asynkron JavaScript og XML)

Med **AJAX** kan du:

- Opdatere en webside uden at indlæse siden igen.
- Anmode om data fra en server, efter siden er indlæst.
- Modtage data fra en server, efter siden er indlæst.
- Sendte data til en server i baggrunden.

---

[https://www.w3schools.com/xml/ajax\\_intro.asp](https://www.w3schools.com/xml/ajax_intro.asp)

Det er det smarte ved AJAX, at scripts ikke går i stå, da der kan være flere processer i gang i forskellige tempi (asynkront)

# fetch() er en asynkron metode

- Fetch er et API til hentning af ressourcer på tværs af netværket.
- Fetch(arg) kræver et obligatorisk argument: **stien** til den **ressource**, du vil hente, f.eks. en **json** fil.
- Fetch returnerer et *løfte* (**promise**).
- Et **promise** er et objekt, der repræsenterer den eventuelle færdiggørelse eller fiasko af en asynkron operation.
- **Promise tillader to eller flere asynkrone handlinger at køre parallelt, hvor hver efterfølgende operation starter, når den forrige operation lykkes, med resultatet fra det forrige trin.**
- Udføres ved hjælp af en “promise chain”.

# Hente JSON med fetch()

promise chain

```
• person.json (T7) — Brackets  
1 ▼ {  
2   "navn": "Martin",  
3   "titel": "Lektor"  
4 }
```

```
▼ {navn: "Martin", titel: "Lektor"} ⓘ  
  navn: "Martin"  
  titel: "Lektor"  
  ► __proto__: Object
```


```
const fil = "person.json";  
  
async function hentdata(fil) {  
  const resultat = await fetch(fil);  
  const json = await resultat.json();  
  vis(json);  
}  
  
function vis(json) {  
  console.log(json);  
}  
  
hentdata(fil);
```

# Øvelse 1: Indlæs og vis json i DOM

Gem en kopi af 06-visMedTemplate.html  
fra i går, og kald kopien: 01-dyrljson.html

Flyt indholdet af arrayet alleDyr ud i en  
ekstern json-fil kaldet alleDyr.json

Dyrene skal nu indlæses fra json-filen  
med fetch og vises på siden.

<b>frø</b>  Type: amfibie Levested: vandhullet
<b>gris</b>  Type: pattedyr Levested: grisehaven
<b>ræv</b>  Type: pattedyr Levested: skoven
<b>sild</b>  Type: fisk Levested: havet
<b>krokodille</b>  Type: krybdyr Levested: floden

# Indsæt JSON data i DOM'en

```
<body>
<h1>Undervisere</h1>
<main></main>

<template>
  <article>
    <h2 class="fornavn">NAVN</h2>
    <h3 class="efternavn">EFTERNAVN</h3>
    <p class="mail">MAIL</p>
  </article>
</template>

<script>
  "use strict";

  const fil = "undervisere.json";

  async function hentdata(fil) {
    const resultat = await fetch(fil);
    const json = await resultat.json();
    vis(json);
  }

  function vis(json) {
    const beholder = document.querySelector("main");
    const skabelon = document.querySelector("template");
    json.forEach(underviser => {
      const klon = skabelon.cloneNode(true).content;
      klon.querySelector(".fornavn").textContent=underviser.fornavn;
      klon.querySelector(".efternavn").textContent=underviser.efternavn;
      klon.querySelector(".mail").textContent=underviser.mail;
      beholder.appendChild(klon);
    });
  }

  hentdata(fil);
</script>
</body>
```

```
undervisere.json (T7) — Brackets
1  [
2  {
3      "fornavn": "Martin",
4      "efternavn": "Bregnhøj",
5      "mail": "mabe@kea.dk",
6      "emner": ["JavaScript", "CSS", "Projektstyring"]
7  },
8  {
9      "fornavn": "Klaus",
10     "efternavn": "Mandal Hansen",
11     "mail": "klmh@kea.dk",
12     "emner": ["JavaScript", "HTML", "Tøjmode"]
13  },
14  {
15     "fornavn": "Louise Ea",
16     "efternavn": "Holbek",
17     "mail": "loeh@kea.dk",
18     "emner": ["SoMe", "SEO", "BMC"]
19  }
20 ]
```

## Undervisere

**Martin**

**Bregnhøj**

mabe@kea.dk

**Klaus**

**Mandal Hansen**

klmh@kea.dk

**Louise Ea**

**Holbek**

loeh@kea.dk

# Arryas i objekter i arrays = loop i loop

```
function vis(json) {  
  const beholder = document.querySelector("main");  
  const skabelon = document.querySelector("template");  
  json.forEach(underviser => {  
    const klon = skabelon.cloneNode(true).content;  
    klon.querySelector(".fornavn").textContent = underviser.fornavn;  
    klon.querySelector(".eternavn").textContent = underviser.eternavn;  
    klon.querySelector(".mail").textContent = underviser.mail;  
  
    underviser.emner.forEach(emne => {  
      klon.querySelector(".emneliste").innerHTML += "<li>" + emne + "</li>"  
    })  
  
    beholder.appendChild(klon);  
  });  
}
```

```
<template>  
  <article>  
    <h2 class="fornavn">NAVN</h2>  
    <h3 class="eternavn">EFTERNAVN</h3>  
    <p class="mail">MAIL</p>  
    <p>Emner:</p>  
    <ul class="emneliste"></ul>  
  </article>  
</template>
```

**Martin**

**Bregnhøi**

mabe@kea.dk

Emner:

- JavaScript
- CSS
- Projektstyring

**Klaus**

**Mandal Hansen**

klmh@kea.dk

Emner:

- JavaScript
- HTML
- Tøjmode

**Louise Ea**

**Holbek**

loeh@kea.dk

Emner:

- SoMe
- SEO
- BMC



## Øvelse 2: Lav en json-fil med array

- Lav to nye filer: en json og en html fil.
- Lav et array med tre objekter i json-filen, som hver indeholder et array, f.eks. biler med udstyr.
- Fetch json-filen i html-filen og vis alle data i DOM'en.

```
[  
  {  
    "mærke": "Volvo",  
    "model": "Amazon",  
    "motor": "Benzin",  
    "km": 500000,  
    "udstyr": ["blinklys", "læderrat"]  
  },  
  {  
    "mærke": "VW",  
    "model": "Polo",  
    "motor": "Diesel",  
    "km": 40500,  
    "udstyr": ["sædevarme", "gps"]  
  },  
  {  
    "mærke": "Tesla",  
    "model": "S",  
    "motor": "El",  
    "km": 1000,  
    "udstyr": ["gps", "aircon", "soltag"]  
  }  
]
```