Loops, events og closure

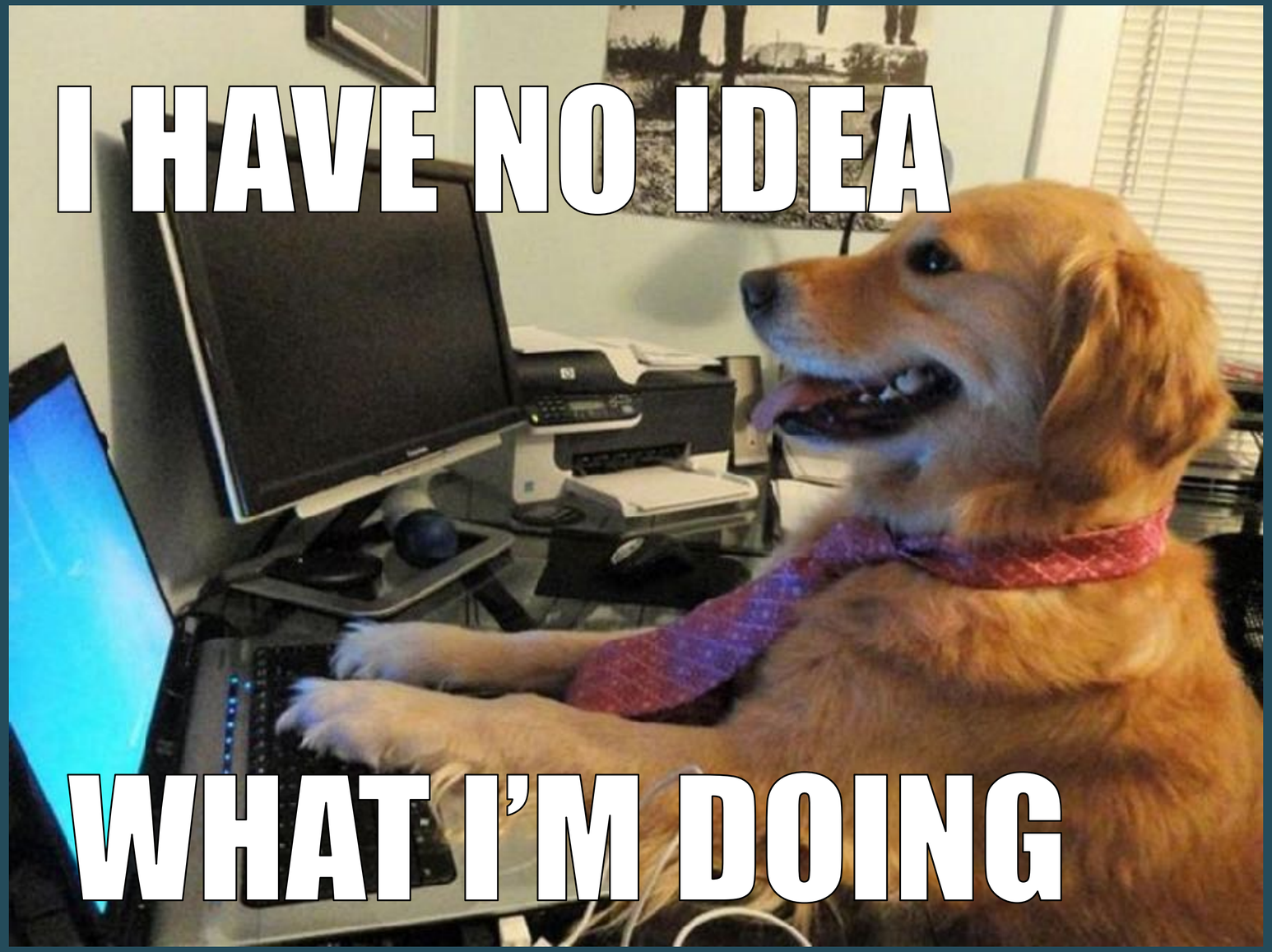
Objekter, Arrays og DOM-manipulation

Datamatiker

KØBENHAVNS ERHVERVSAKADEMI

Agenda

- Objekter og arrays
- Fetch og JSON
- Loops
- Events og closure
- DOM-manipulation
- Specialisering af output


I HAVE NO IDEA
WHAT I'M DOING

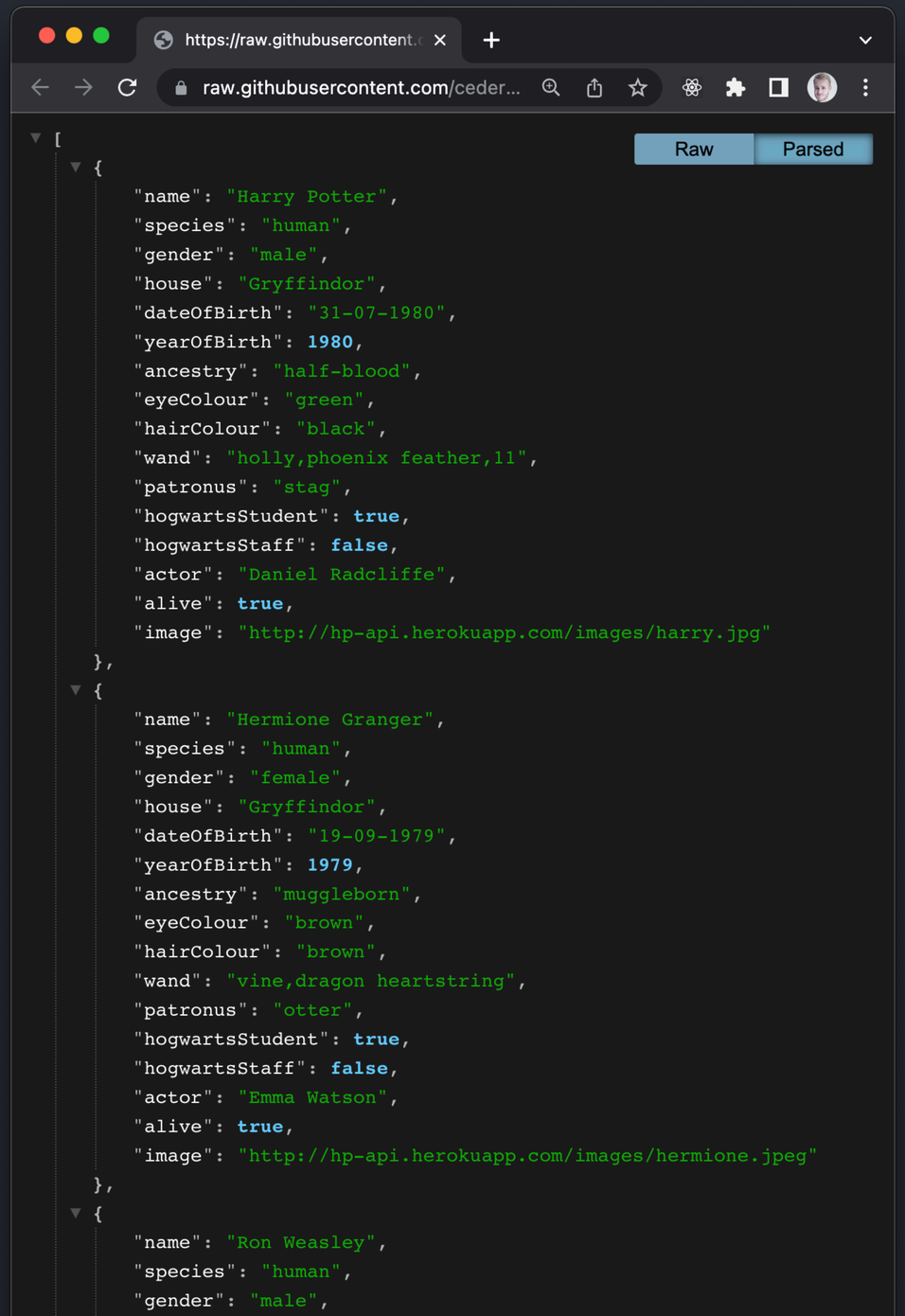
Computer science student

Senior developer, 10+ years experience

https://www.instagram.com/p/BxWAgatgSmn/

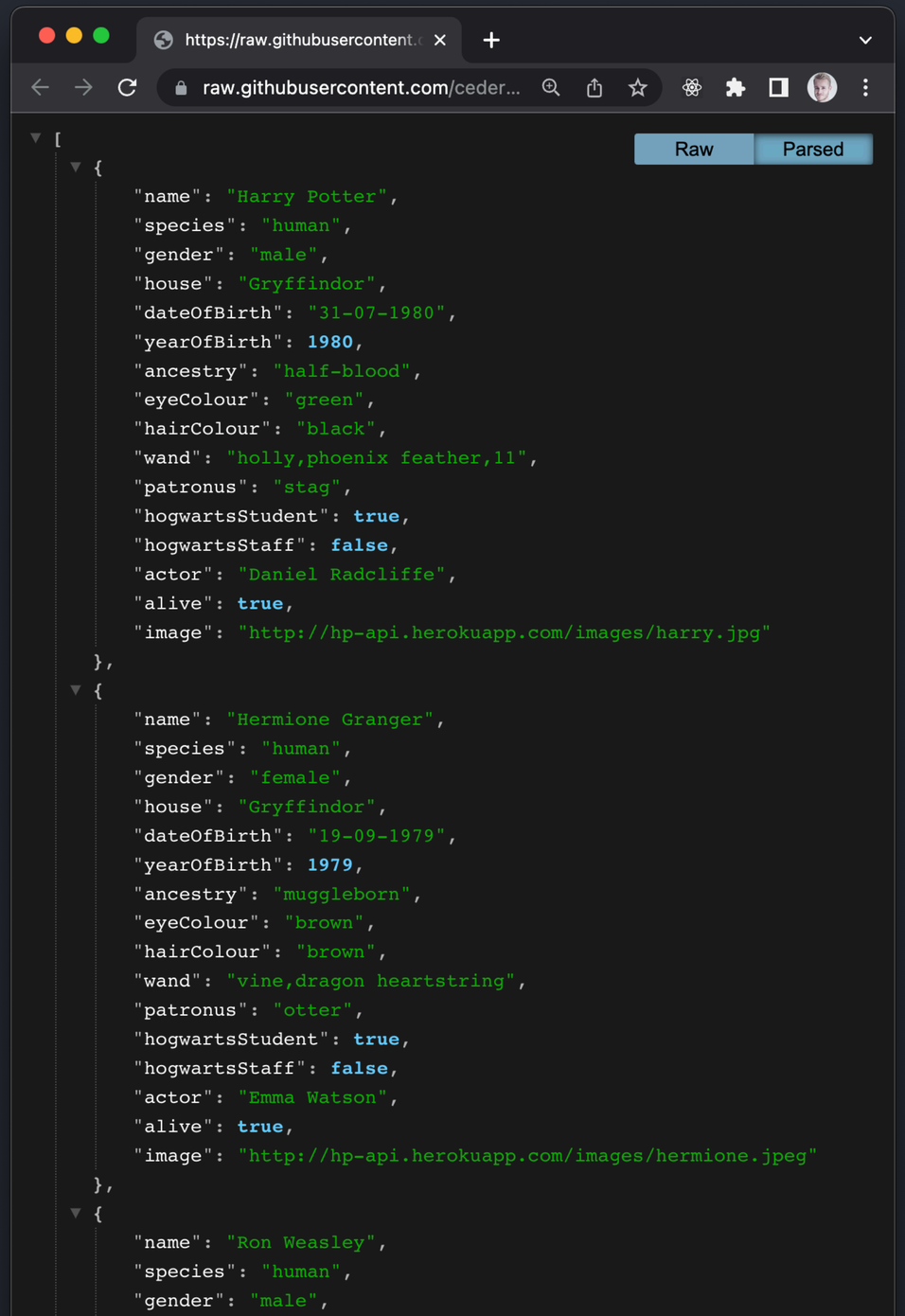
Data

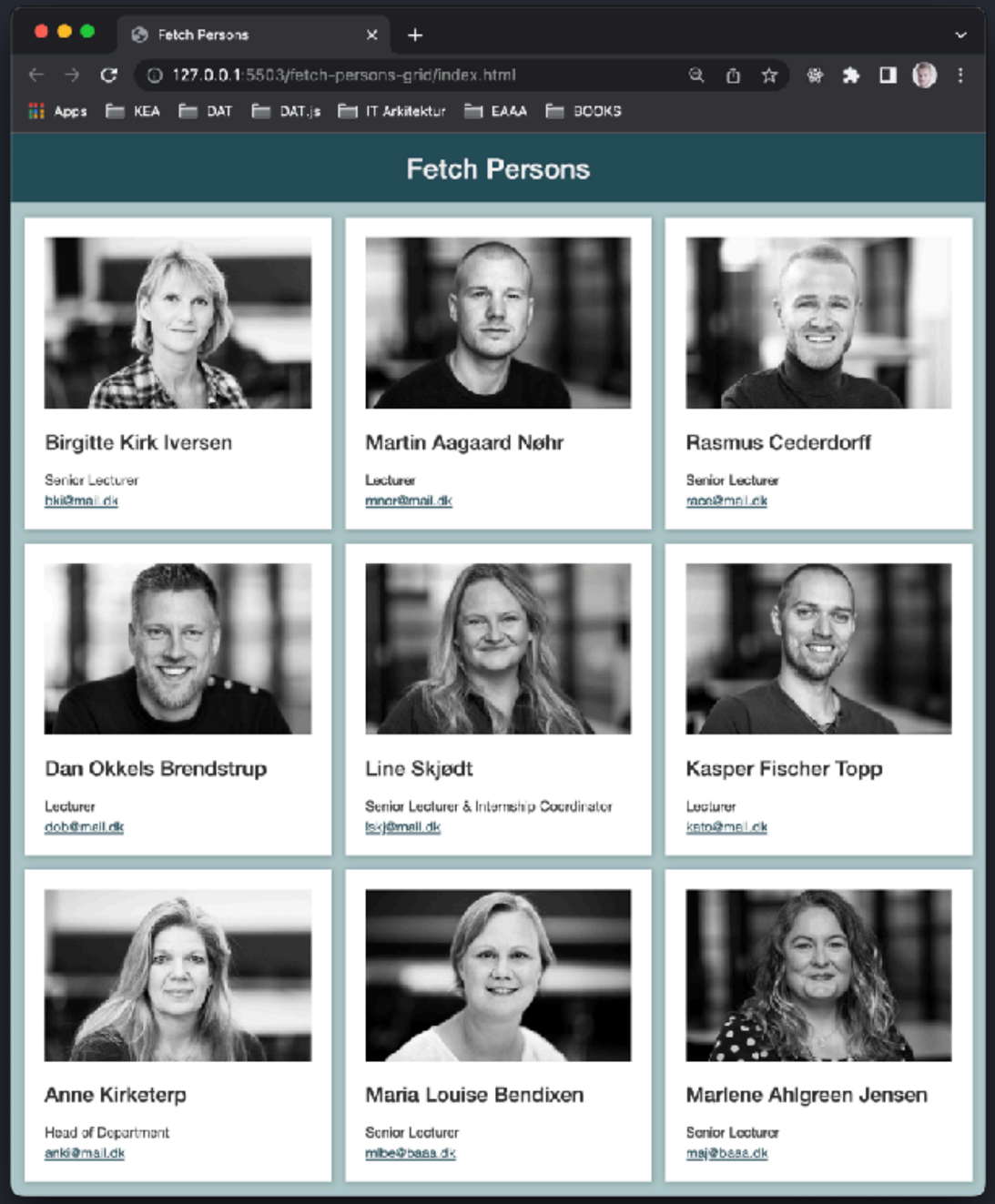
JavaScript, lister, objekter, JSON, fetch, DOM-manipulation og meget mere…



Hvordan viser vi denne JSON-data, så det giver mening for brugerne?

- Denne rette visualisering
- Generere HTML
- DOM-manipulation

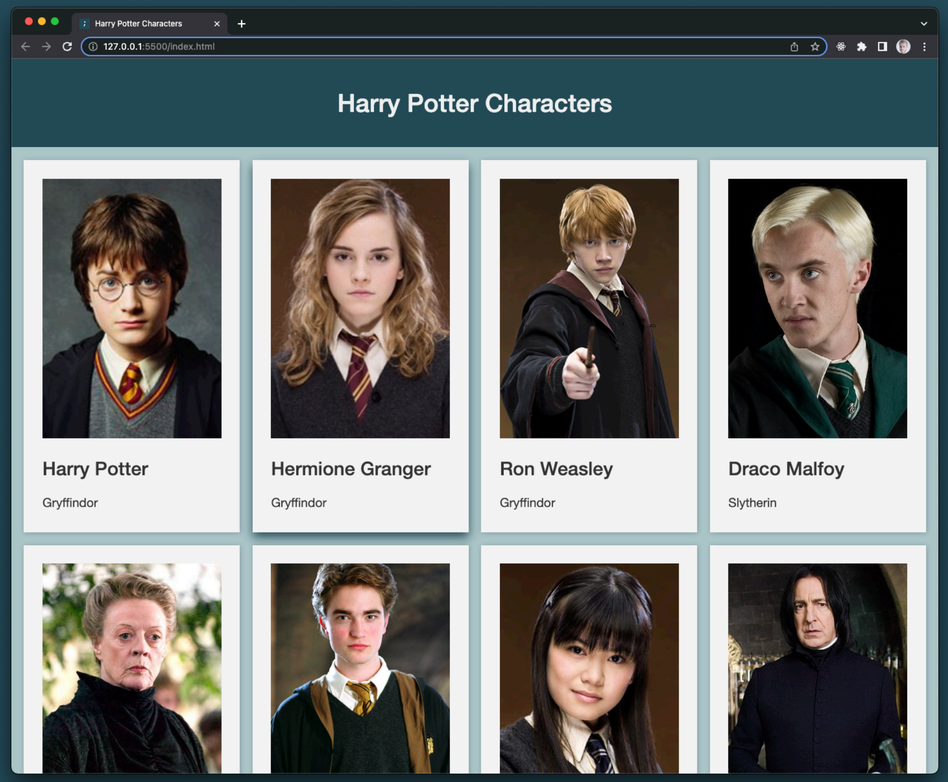




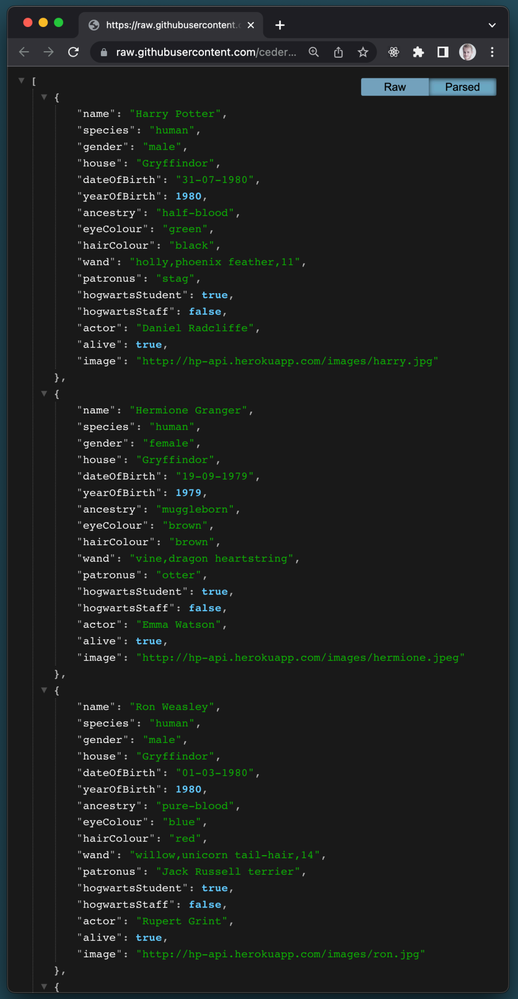
Forståelse for objekter og arrays


Programming knowledge
Backend & Frontend

Frontend (client)



JSON Data Source (Server)



https://cederdorff.github.io/potter-app/

https://raw.githubusercontent.com/cederdorff/dat-js/main/data/potter.json

Objects

A set of named values

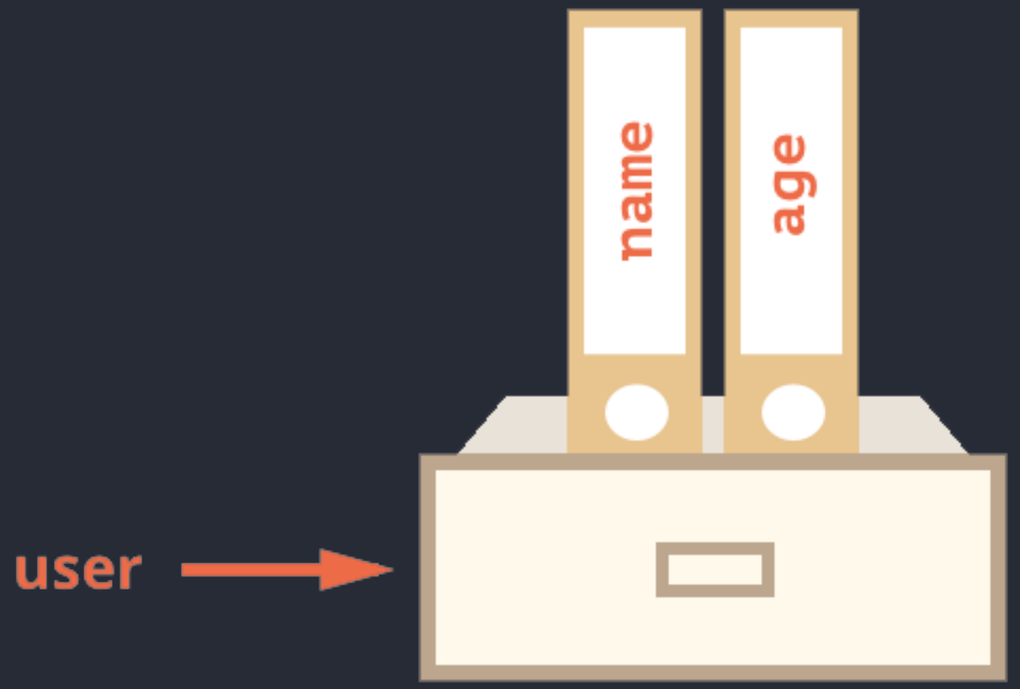
Objects are used to store keyed collections of various data

Containers for named values called properties. A property is a “`key: value`” pair

Objects

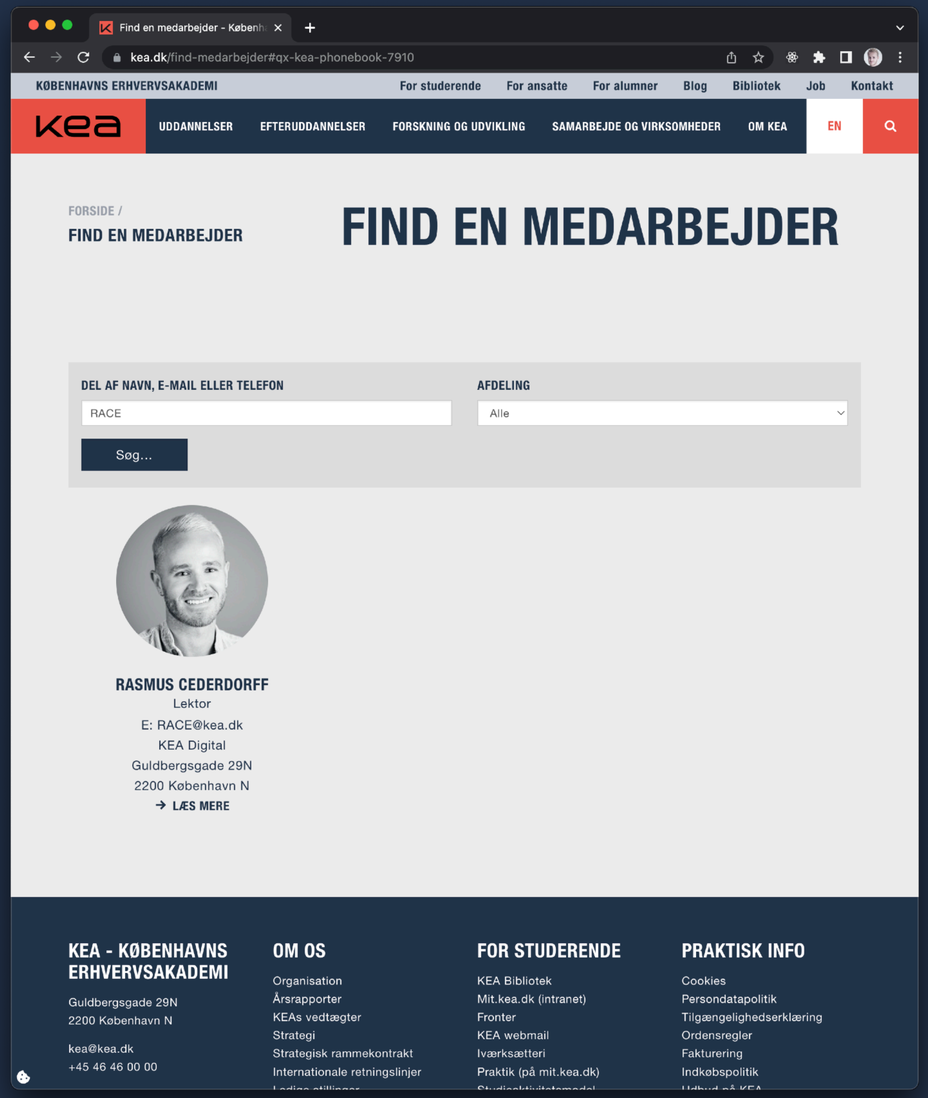
A set of named values

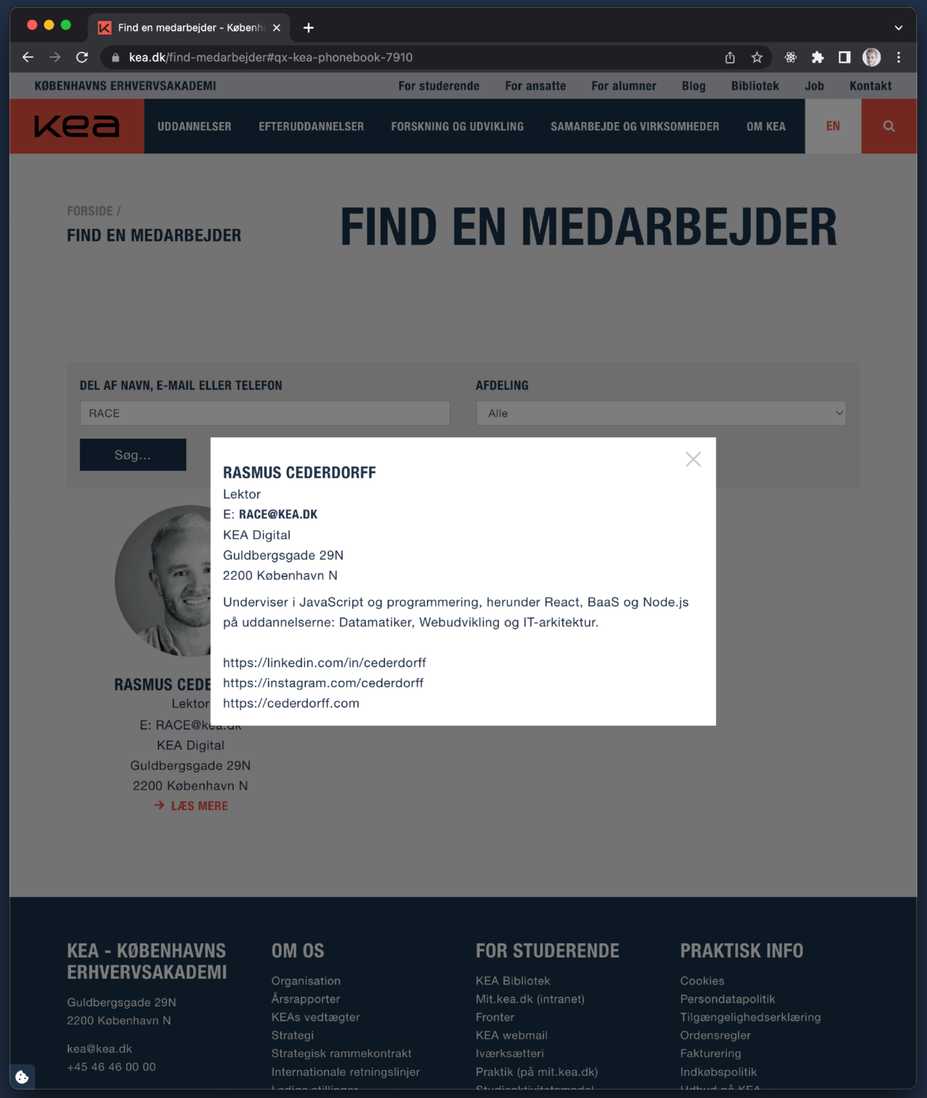
```
let user = {
  name: 'Alicia',
  age: 6
};

console.log(user.name +
  " is " + user.age +
  " years old.");
```



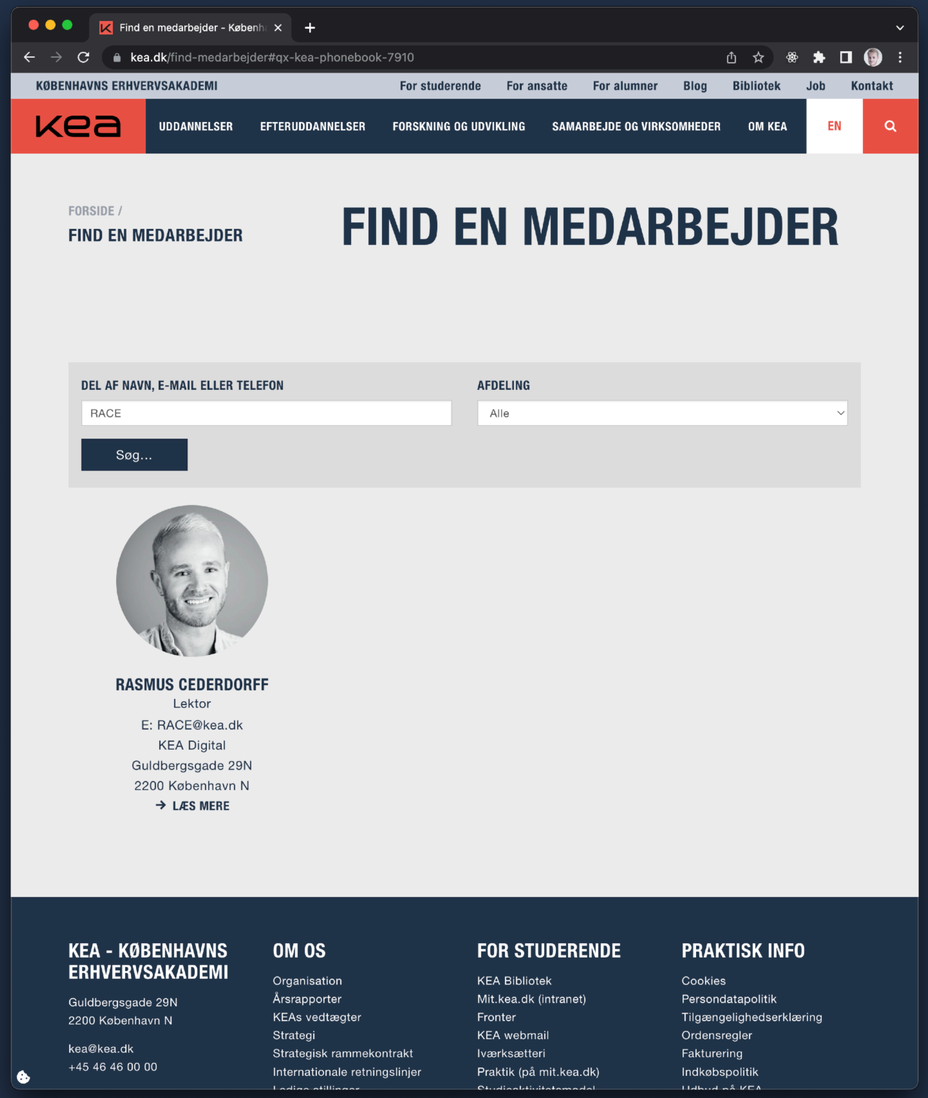

```
Alicia is 6 years old.          main.js:11
>
```

Hvad ser du?


Find en medarbejder - Københ
kea.dk/find-medarbejder#qx-kea-phonebook-7910
KØBENHAVNS ERHVERVSAKADEMI
For studerende
For ansatte
For alumner
Blog
Bibliotek
Job
Kontakt
kea
UDDANNELSER
EFTERUDDANNELSER
FORSKNING OG UDVIKLING
SAMARBEJDE OG VIRKSOMHEDER
OM KEA
EN
FORSIDE /
FIND EN MEDARBEJDER
FIND EN MEDARBEJDER
DEL AF NAVN, E-MAIL ELLER TELEFON
RACE
AFDELING
Alle
Søg...
RASMUS CEDERDORFF
Lektor
E: RACE@kea.dk
KEA Digital
Guldbergsgade 29N
2200 København N
LÆS MERE
KEA - KØBENHAVNS ERHVERVSAKADEMI
Guldbergsgade 29N
2200 København N
kea@kea.dk
+45 46 46 00 00
OM OS
Organisation
Årsrapporter
KEAs vedtægter
Strategi
Strategisk rammekontrakt
Internationale retningslinjer
FOR STUDERENDE
KEA Bibliotek
Mit.kea.dk (intranet)
Fronter
KEA webmail
Iværksætteri
Praktik (på mit.kea.dk)
PRAKTISK INFO
Cookies
Persondatapolitik
Tilgængelighedserklæring
Ordensregler
Fakturering
Indkøbspolitik


Find en medarbejder - Københ
kea.dk/find-medarbejder#qx-kea-phonebook-7910
KØBENHAVNS ERHVERVSAKADEMI
For studerende
For ansatte
For alumner
Blog
Bibliotek
Job
Kontakt
kea
UDDANNELSER
EFTERUDDANNELSER
FORSKNING OG UDVIKLING
SAMARBEJDE OG VIRKSOMHEDER
OM KEA
EN
FORSIDE /
FIND EN MEDARBEJDER
FIND EN MEDARBEJDER
DEL AF NAVN, E-MAIL ELLER TELEFON
RACE
AFDELING
Alle
Søg...
RASMUS CEDERDORFF
Lektor
E: RACE@KEA.DK
KEA Digital
Guldbergsgade 29N
2200 København N
Underviser i JavaScript og programmering, herunder React, BaaS og Node.js på uddannelserne: Datamatiker, Webudvikling og IT-arkitektur.
https://linkedin.com/in/cederdorff
https://instagram.com/cederdorff
https://cederdorff.com
RASMUS CEDE
Lektor
E: RACE@kea.dk
KEA Digital
Guldbergsgade 29N
2200 København N
LÆS MERE
KEA - KØBENHAVNS ERHVERVSAKADEMI
Guldbergsgade 29N
2200 København N
kea@kea.dk
+45 46 46 00 00
OM OS
Organisation
Årsrapporter
KEAs vedtægter
Strategi
Strategisk rammekontrakt
Internationale retningslinjer
FOR STUDERENDE
KEA Bibliotek
Mit.kea.dk (intranet)
Fronter
KEA webmail
Iværksætteri
Praktik (på mit.kea.dk)
PRAKTISK INFO
Cookies
Persondatapolitik
Tilgængelighedserklæring
Ordensregler
Fakturering
Indkøbspolitik

Hvad ser du?




```
const race = {
    name: "Rasmus Cederdorff",
    position: "Lektor",
    mail: "race@kea.dk",
    department: "KEA Digital",
    address: "Guldbergsgade 29N, 2200 København N",
    image: "https://kea.dk/slir/w180-c1x1/images/us
};
```

netflix.com

NETFLIX

Hvem ser?

Personen der rent faktisk betaler for profilen

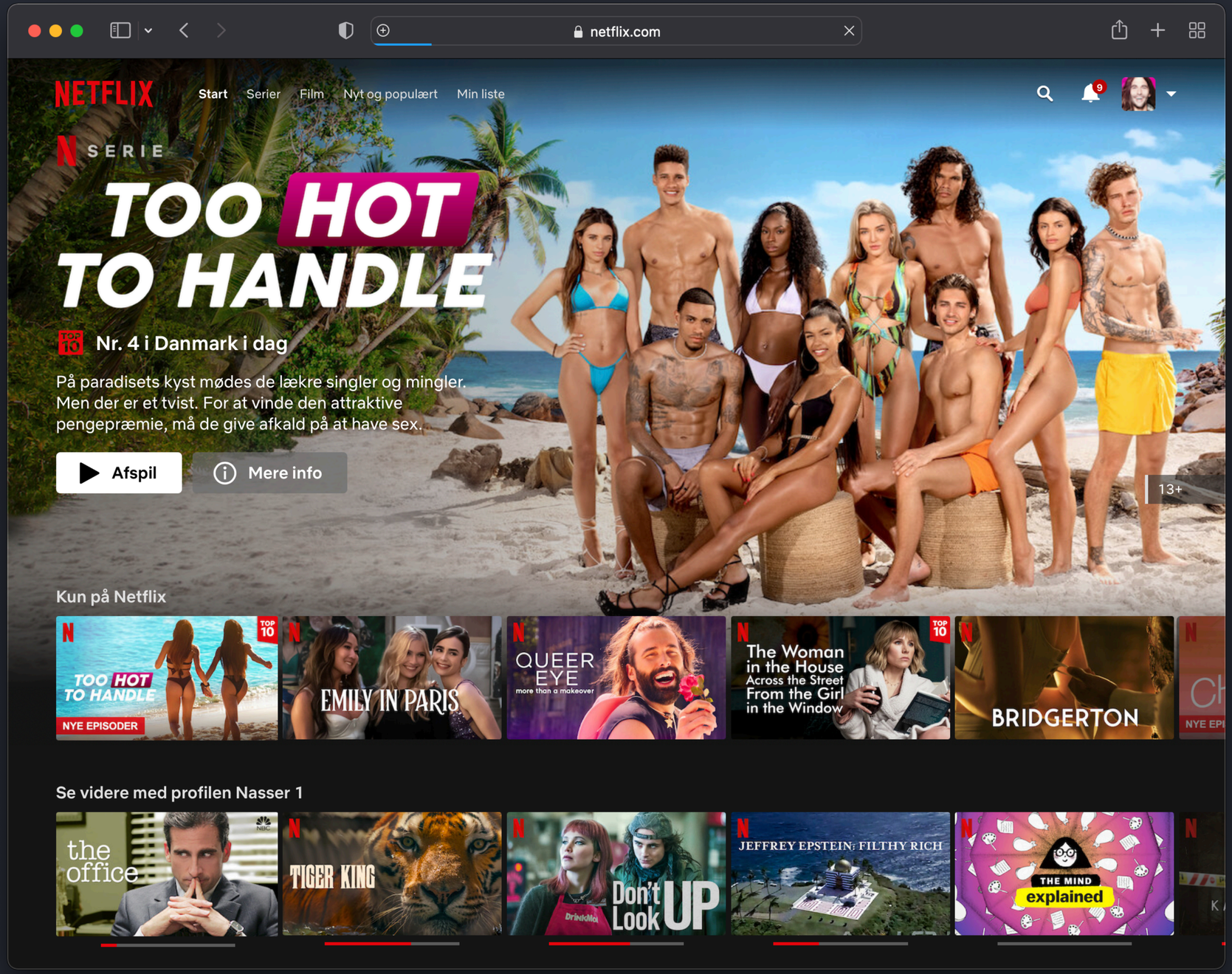
Nasser 1

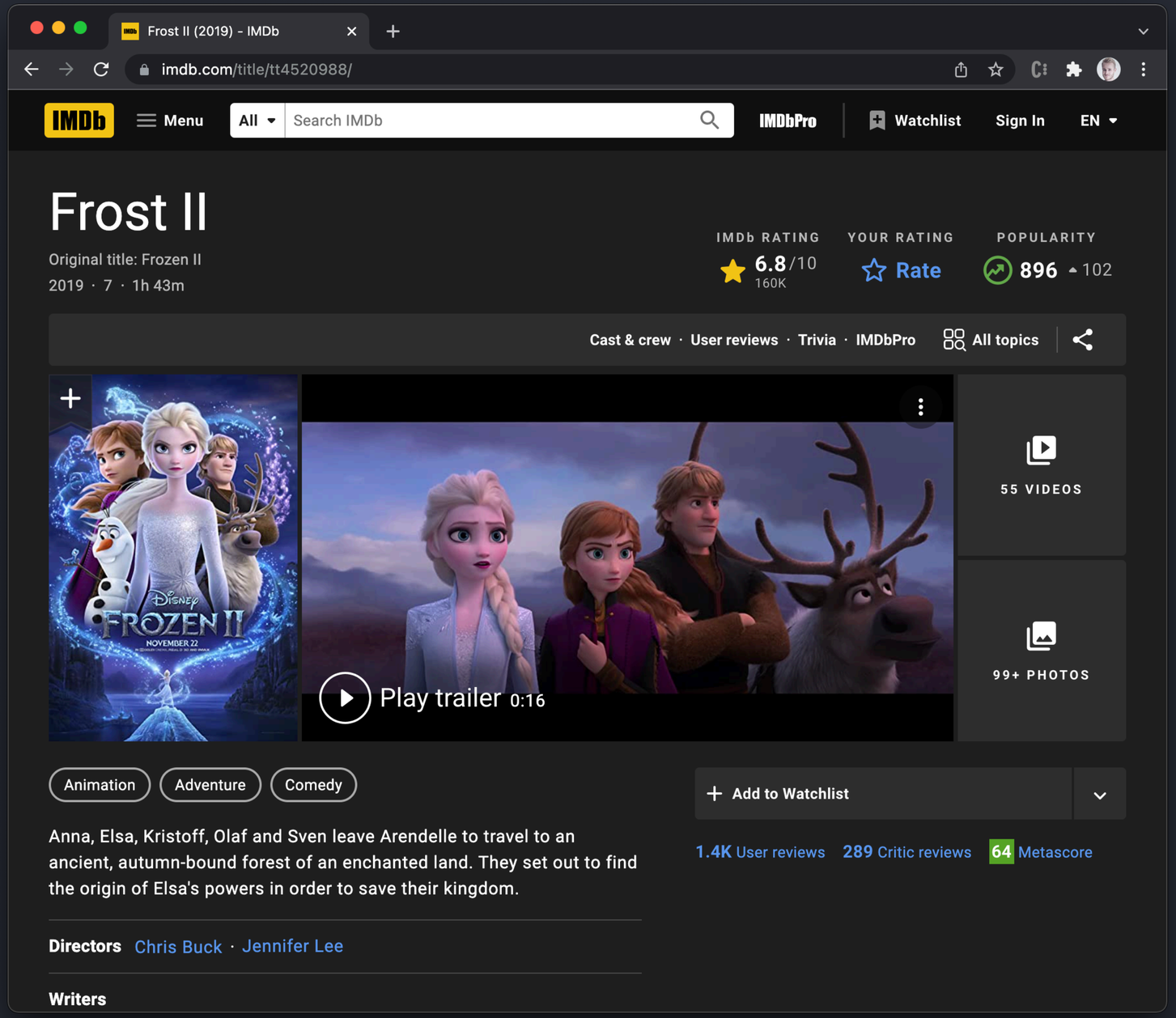
Nasser 2

Nasser 3

Nasser 4 Khader

Administrer profiler


netflix.com
NETFLIX
Start
Serier
Film
Nyt og populært
Min liste
SERIE
TOO HOT TO HANDLE
Nr. 4 i Danmark i dag
På paradisets kyst mødes de lækre singler og mingler. Men der er et tvist. For at vinde den attraktive pengepræmie, må de give afkald på at have sex.
Afspil
Mere info
13+
Kun på Netflix
NYE EPISODER
EMILY IN PARIS
QUEER EYE
more than a makeover
The Woman in the House Across the Street From the Girl in the Window
BRIDGERTON
Se videre med profilen Nasser 1
the office
TIGER KING
Don't Look UP
JEFFREY EPSTEIN: FILTHY RICH
THE MIND
explained

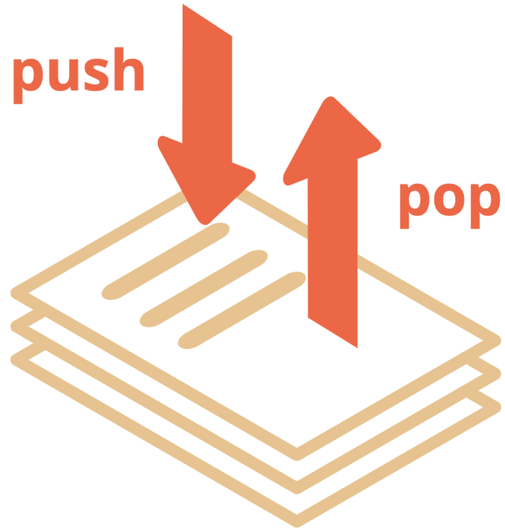



```
let movie = {
  title: "Frozen 2",
  description: "Elsa the Snow Queen has a
  trailer: "https://www.youtube.com/embed
  length: "1h 43m",
  year: "2019"
}
```

Arrays

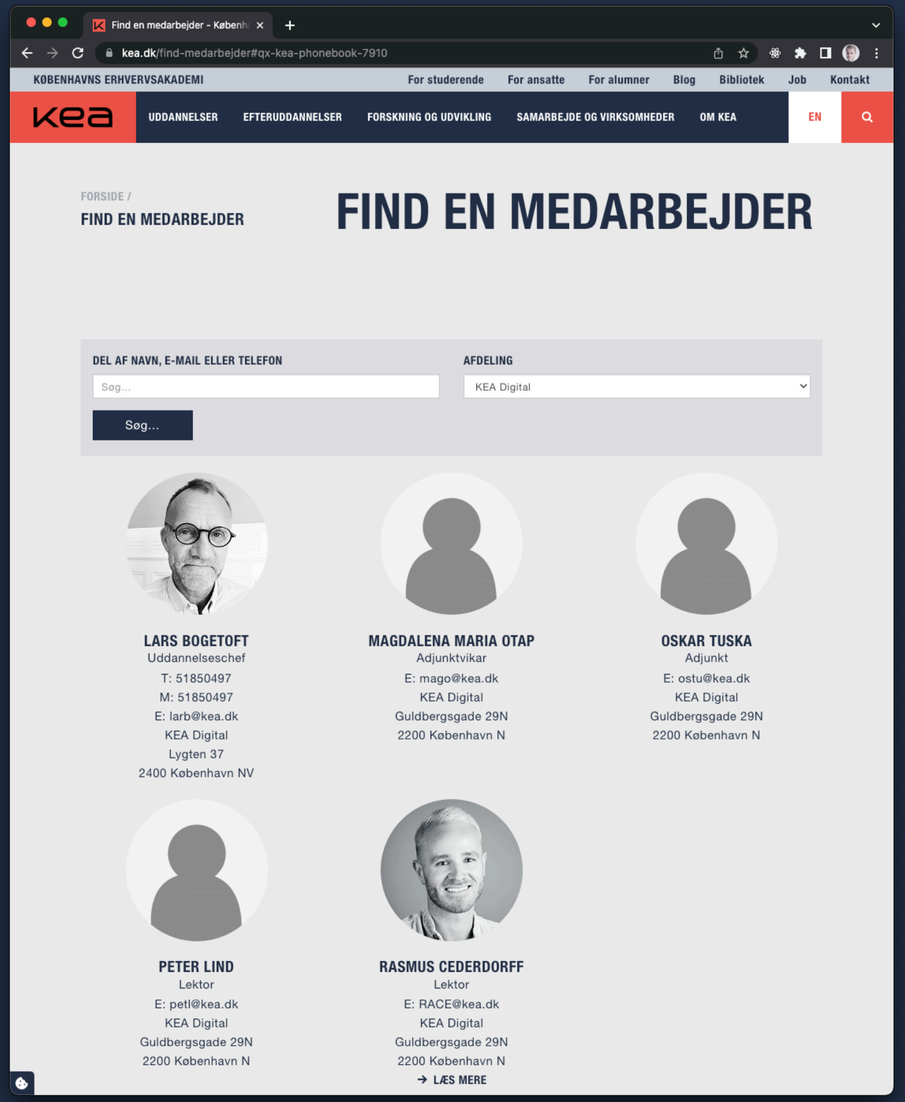
Collections

Ordered collection of values or objects

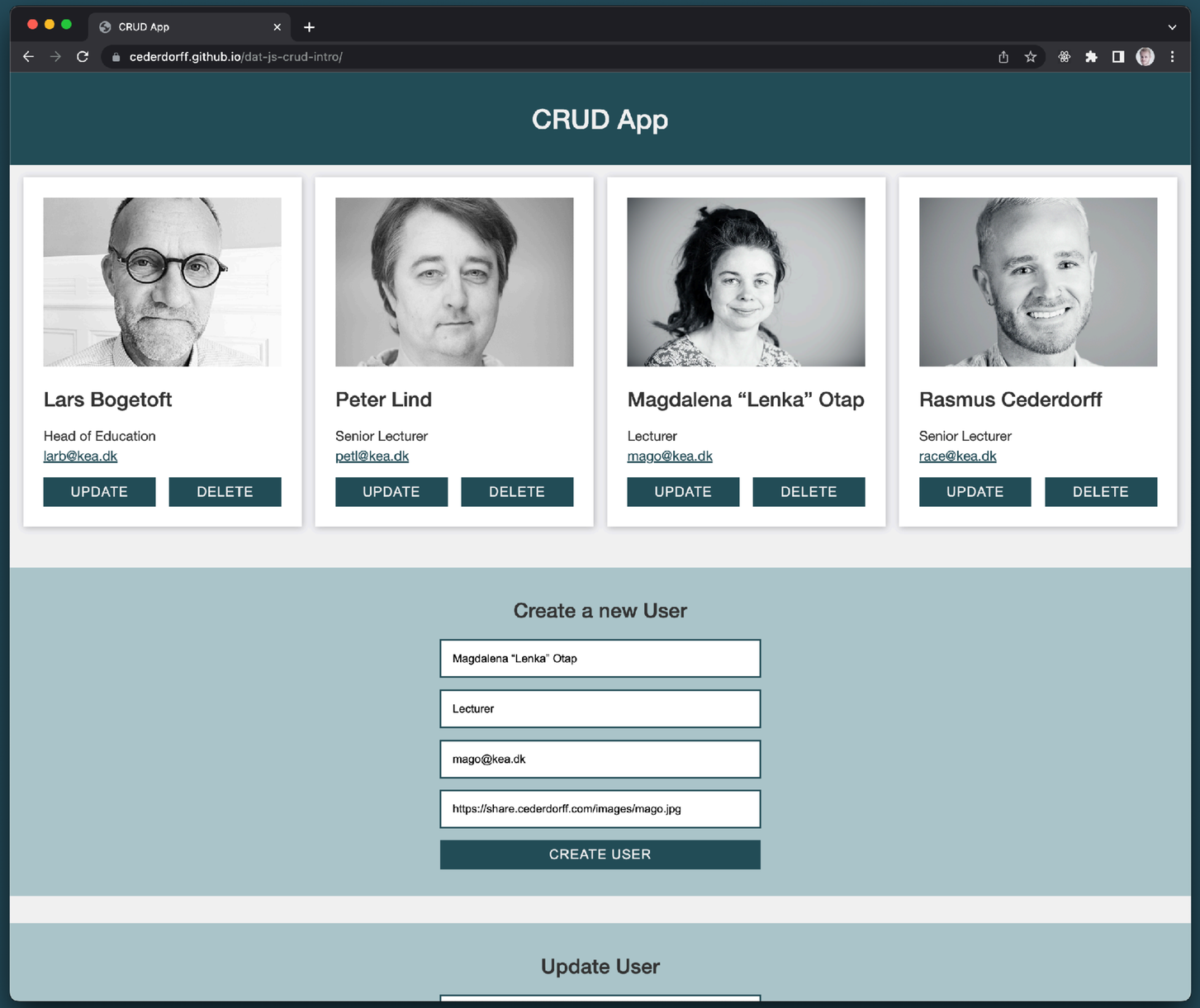


An array is a way to hold more than one value at a time we have a 1st, a 2nd, a 3rd, a 4th element and so on.

Hvad ser du?




```
const lecturers = [
    {
        name: "Peter Lind",
        position: "Lektor",
        mail: "petl@kea.dk",
        department: "KEA Digital",
        address: "Guldbergsgade 29N, 2200 København
        image: "https://share.cederdorff.com/images/
    },
    {
        name: "Rasmus Cederdorff",
        position: "Lektor",
        mail: "race@kea.dk",
        department: "KEA Digital",
        address: "Guldbergsgade 29N, 2200 København
        image: "https://kea.dk/slir/w180-c1x1/images
    }
];
```

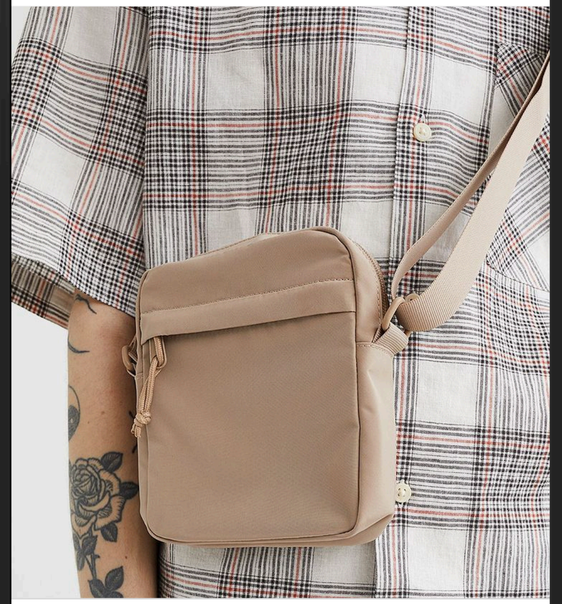


https://cederdorff.github.io/dat-js-crud-intro/

Instagram

Your story | fhergram | vladaafa... | kristybra... | gabriel

hm_man



Instagram

Liked by **praaajyot** and **others**

hm_man Shades, style and summer vibes, they all fit inside. #HMMan... more

View all 5 comments

44 MINUTES AGO

programmerhumor_io

I Am Devloper
@iamdevloper

Elon Musk: I'm putting people on Mars!
Developers: Fantastic, more timezones to support

cederdorff

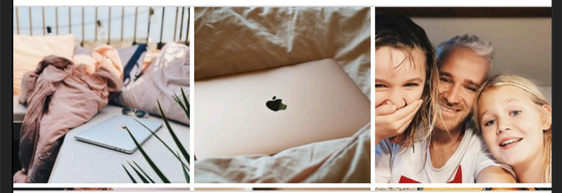
cederdorff

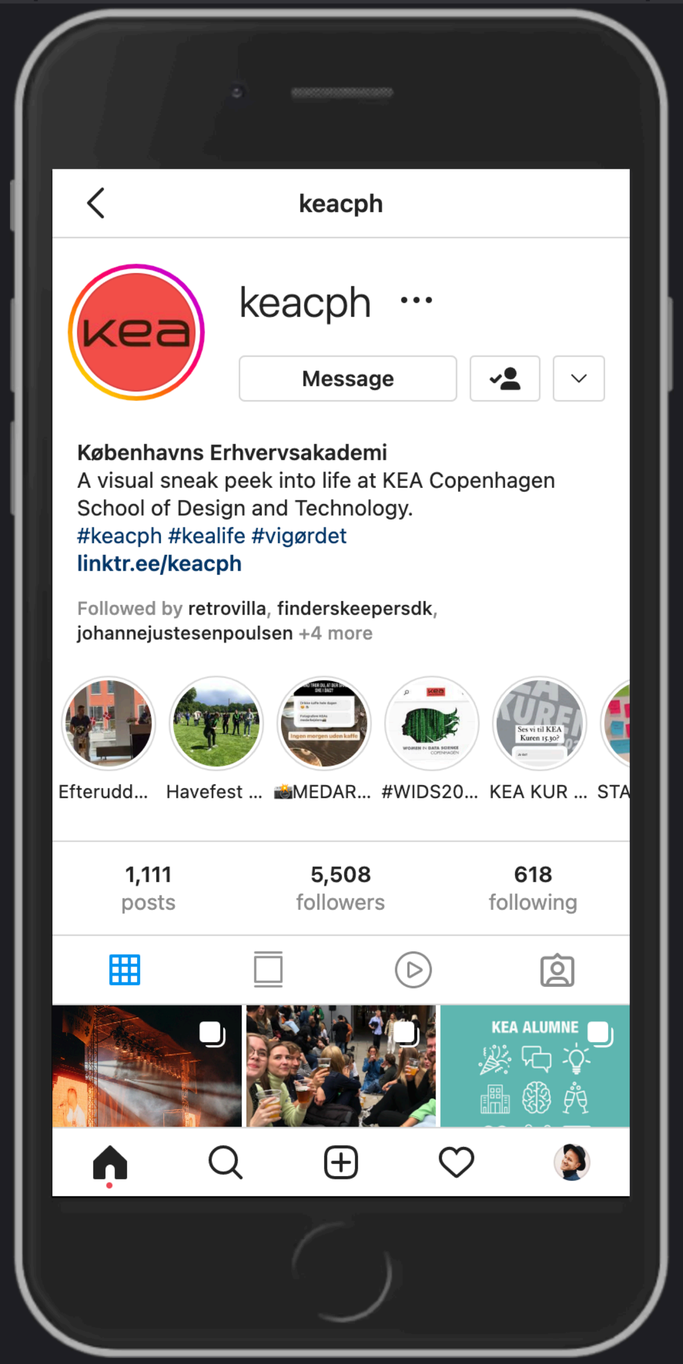
Edit Profile

Rasmus Cederdorff
Senior Lecturer & Web App Developer
Alicia & Ida
Aarhus, Denmark
cederdorff.com

AMSTER... | LONDON ... | Karla & Pi... | @villacph | IDA

181	5,441	635
posts	followers	following





Dimensions: iPhone ... 375 x 667 100%

Elements Components Console Performance insights Recorder Sources Network

Preserve log Disable cache No throttling

Invert Hide data URLs All Fetch/XHR JS CSS Img Media Font Doc WS Wasm Manifest Other

Has blocked cookies Blocked Requests 3rd-party requests

2000 ms 4000 ms 6000 ms 8000 ms 10000 ms 12000 ms 14000 ms

Name

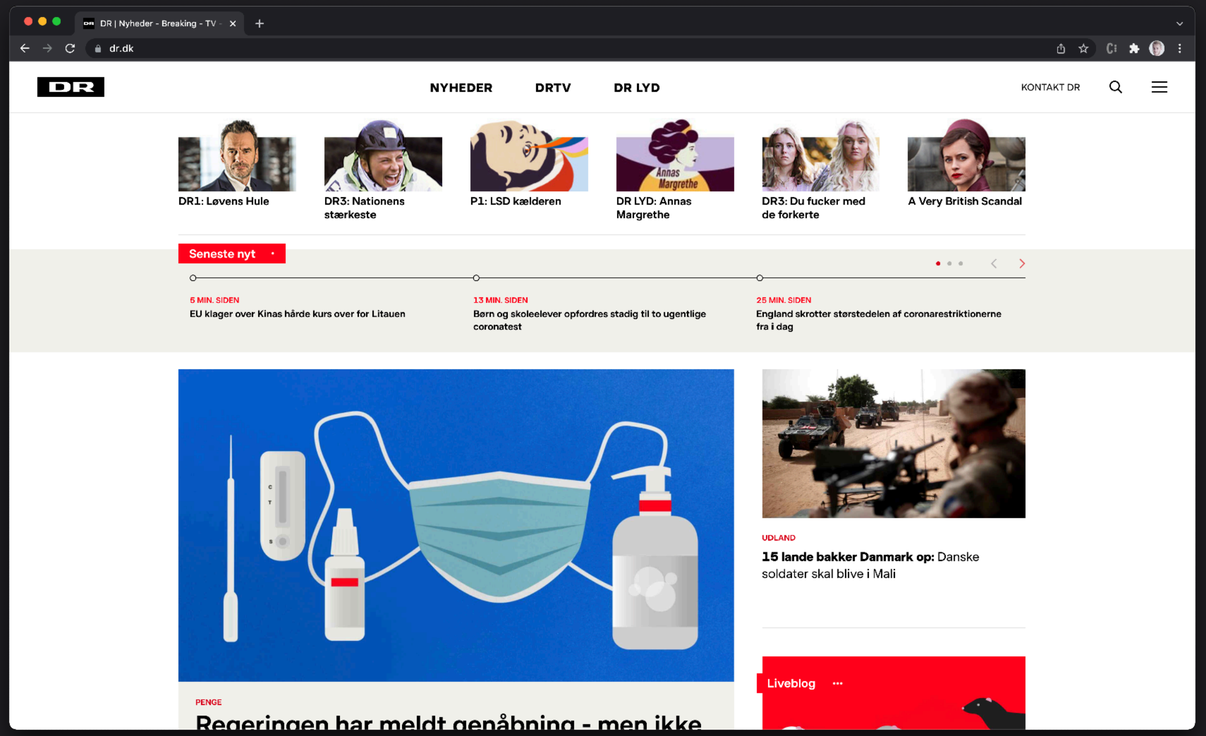
?modules=PolarisBD...
?content_type=PROF...
?username=keacph
reels_tray/
timeline/
ig_sso_users/
logging_client_events
bz?__d=dis
falco
bz?__a=1&__ccg=G...
batch_fetch_web/
get_encrypted_crede...
highlights_tray/
?target_user_id=140...
badge/
story/
bulk-route-definitions/
bulk-route-definitions/
?query_hash=69cba4...
bz?__a=1&__ccg=G...
bz?__a=1&__ccg=G...
bulk-route-definitions/
bz?__a=1&__ccg=G...
logging_client_events
bz?__d=dis

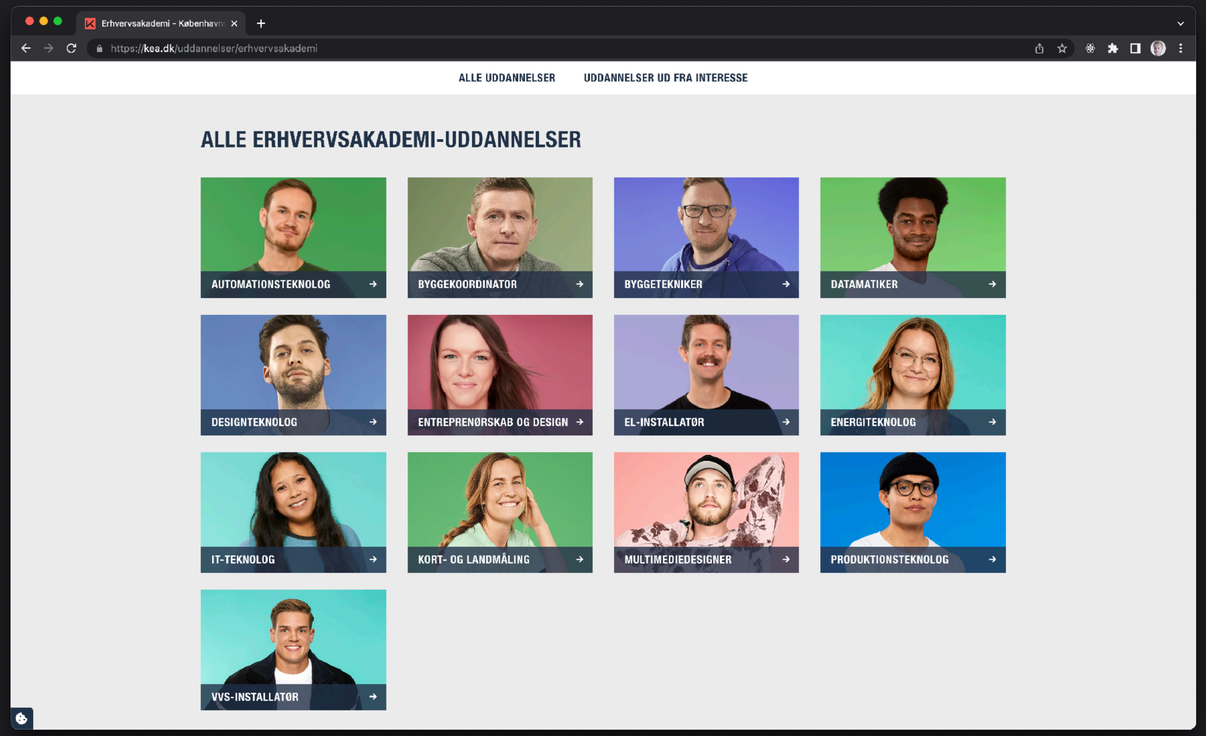
26 / 96 requests 257 kB

Headers Payload Preview Response Initiator Timing Cookies

{data: {user: {,…}}, status: "ok"}
data: {user: {,…}}
user: {,…}
bio_links: []
biography: "A visual sneak peek into life at KEA Copenhagen School of Design and Technology. \n#keacph
biography_with_entities: {,…}
blocked_by_viewer: false
business_address_json: null
business_category_name: "Home Services"
business_contact_method: "CALL"
business_email: null
business_phone_number: null
category_enum: "LOCAL"
category_name: null
connected_fb_page: null
country_block: false
edge_felix_video_timeline: {count: 116, page_info: {has_next_page: true,…},…}
edge_follow: {count: 618}
edge_followed_by: {count: 5508}
edge_media_collections: {count: 0, page_info: {has_next_page: false, end_cursor: null}, edges: []}
edge_mutual_followed_by: {count: 7, edges: [{node: {username: "retrovilla"}}, {node: {username: "finde
edge_owner_to_timeline_media: {count: 1111, page_info: {has_next_page: true,…},…}
edge_saved_media: {count: 0, page_info: {has_next_page: false, end_cursor: null}, edges: []}
external_url: "https://linktr.ee/keacph"
external_url_linkshimmed: "https://l.instagram.com/?u=https%3A%2F%2Flinktr.ee%2Fkeacph&e=ATNF7APjyvCLD
fbid: "17841400899781985"
followed_by_viewer: true
follows_viewer: true
full_name: "Københavns Erhvervsakademi"
guardian_id: null
has_ar_effects: false
has_blocked_viewer: false
has_channel: false
has_clips: false

Objects? Arrays?





It’s all objects & arrays!

Data Types & Data Structures

Objects & Arrays

fetch(...)

HTTP Requests in JavaScript.

A way to get and post data from and to data sources.

```
// fetch with callbacks
fetch("https://cederdorff.github.io/web-frontend/ca
    .then(function (response) {
        return response.json();
    })
    .then(function (data) {
        console.log(data);
    });
//
//
// or with async/await
const response = await fetch("https://cederdorff.gi
const data = await response.json();
console.log(data);
```

fetch(...)

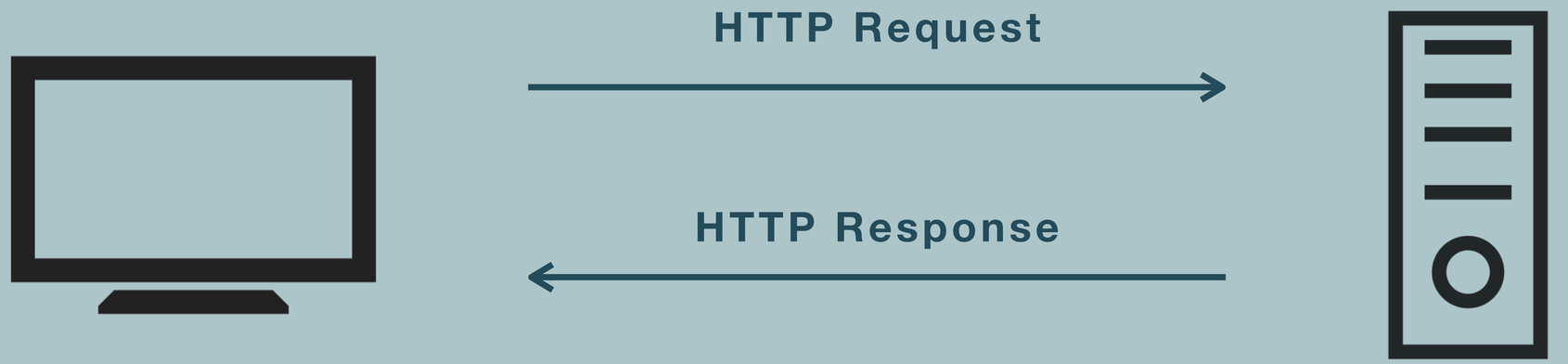
HTTP Requests in JavaScript.

A way to get and post data from and to data sources.

```
async function getCharacters() {
    const response = await fetch(url);
    const data = await response.json();
    console.log(data);
    return data;
}
```

Client-Server Model

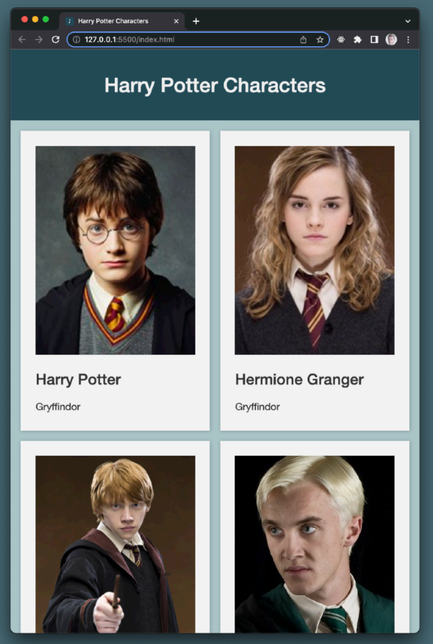
Communication between web **clients** and web **servers**.



Fetch, HTTP Request & Response

WEB APP

DATA SOURCE



FETCH

REQUEST (GET)

JSON DATA

RESPONSE

potters

pokemons

southpark

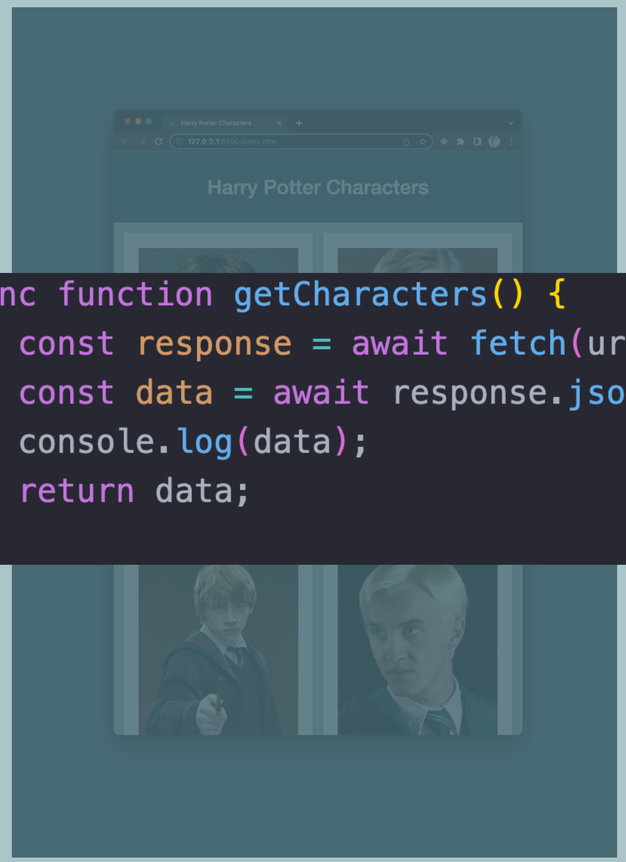
CLIENT

SERVER

Fetch, HTTP Request & Response

WEB APP

DATA SOURCE




```
async function getCharacters() {
    const response = await fetch(url);
    const data = await response.json();
    console.log(data);
    return data;
}
```

FETCH

REQUEST (GET)

JSON DATA

RESPONSE

potters

pokemons

southpark

CLIENT

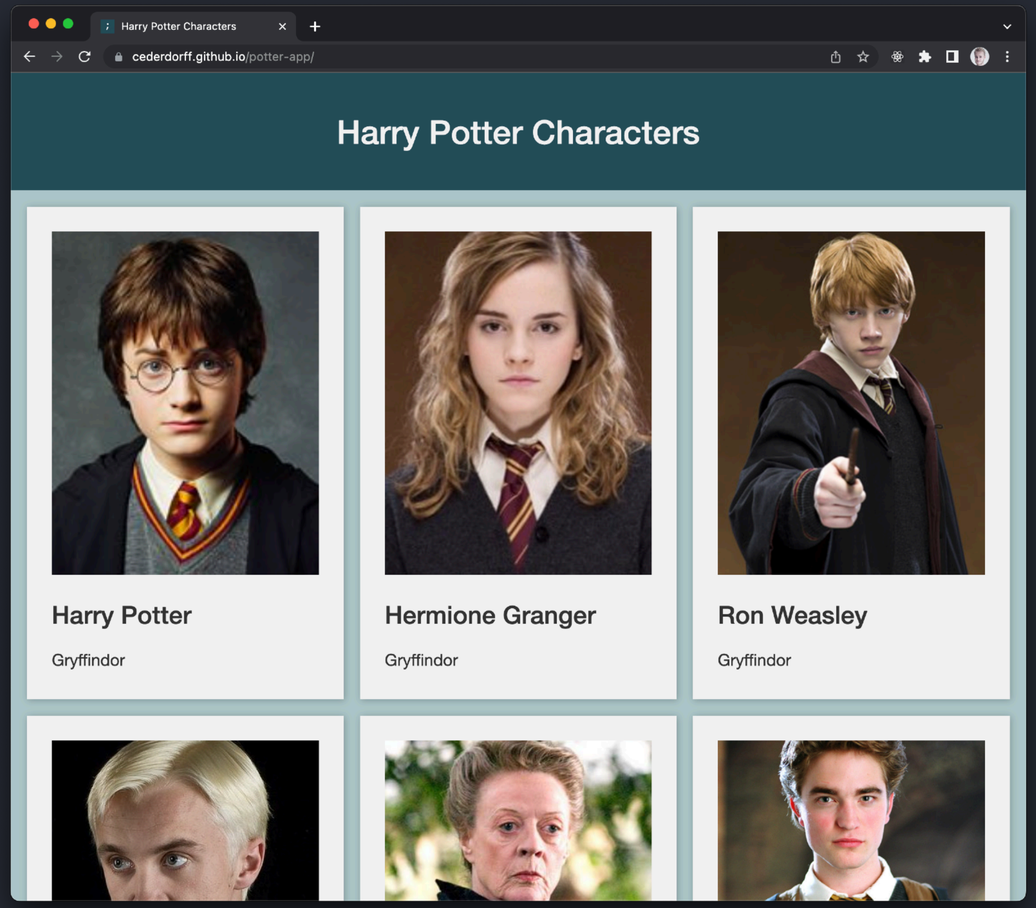
SERVER

fetch(...)

Og Potter App

Løsning: https://cederdorff.github.io/potter-app/

Kode: https://github.com/cederdorff/potter-app/tree/grid



fetch(...) og “call stack”

1) `initApp` kaldes på `load`.

2) `initApp` kalder `getCharacters` og gemmer hvad der returneres i `const`’en `characters`.

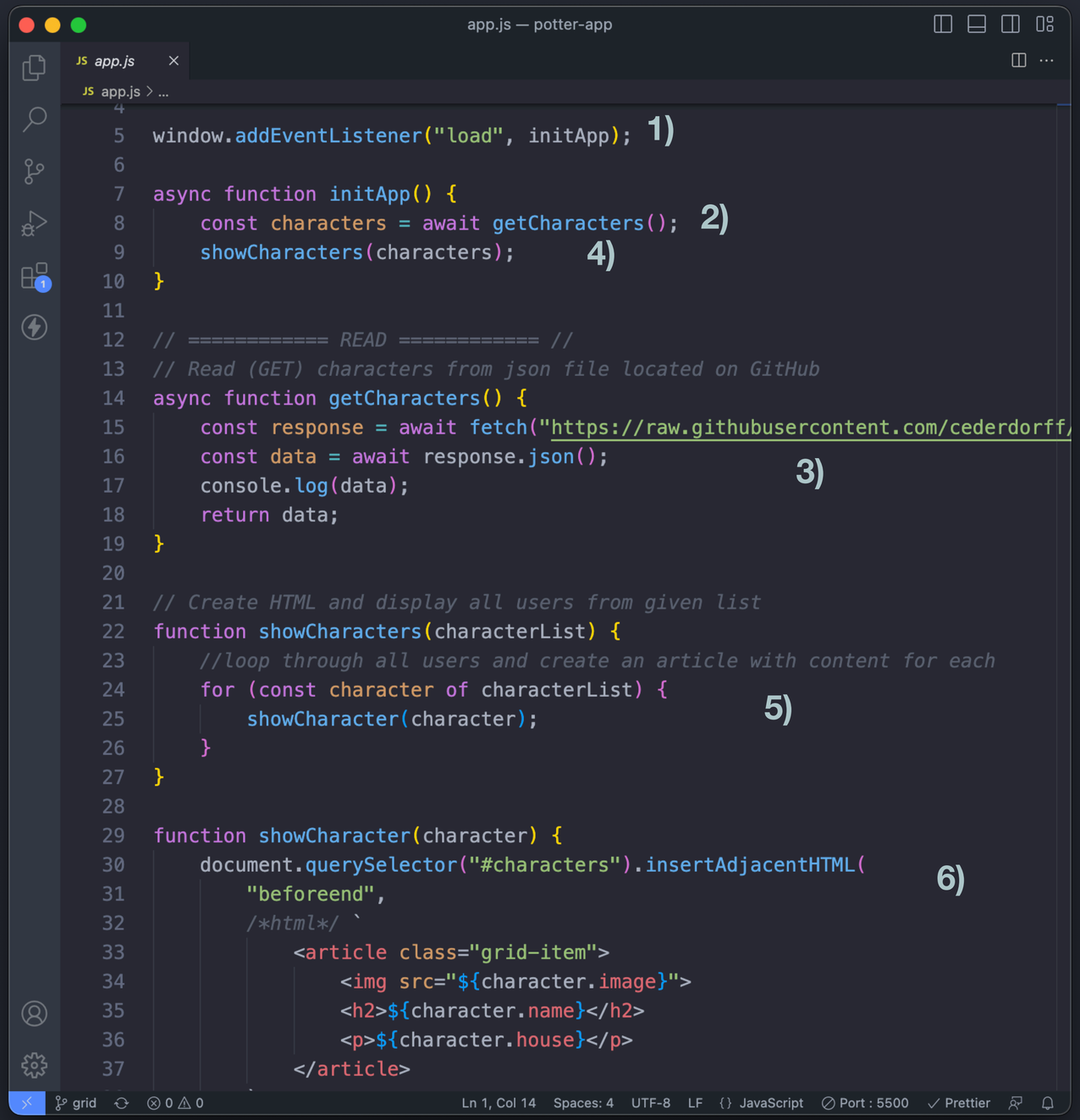
3) `getCharacters` er en `async` funktion, da der anvendes `await` inde i funktionen. Funktionen fetcher data fra en given url. Data parses fra JSON til JavaScript-array, der gemmes i `data`-variablen. `data` returneres, så det kan læses hvor funktionen kaldes.

4) `initApp` kalder nu `showCharacters` med `characters` som argument (parameter). `characters` holder den data, som returneres fra `getCharacters` i 3).

5) `showCharacters` har `characterList` som parameter. `characterList` holder den data som blev sendt med fra `initApp`, dvs at det er samme data som lå i `characters` i `initApp` og tidligere i `data` i `getCharacters`.

`showCharacters` loop’er igennem `characterList`. For hver `character` (objekt) i `characterList` (array) kaldes `showCharacter`.

6) `showCharacter` tager en `character` (et objekt) som parameter og generer HTML med værdier fra `character`-objektets properties. HTML’en tilføjes med insertAdjacentHTML.




```
5  window.addEventListener("load", initApp);

7  async function initApp() {
8      const characters = await getCharacters();
9      showCharacters(characters);
10 }

12 // ========== READ ========== //
13 // Read (GET) characters from json file located on GitHub
14 async function getCharacters() {
15     const response = await fetch("https://raw.githubusercontent.com/cederdorff/
16     const data = await response.json();
17     console.log(data);
18     return data;
19 }

21 // Create HTML and display all users from given list
22 function showCharacters(characterList) {
23     //loop through all users and create an article with content for each
24     for (const character of characterList) {
25         showCharacter(character);
26     }
27 }

29 function showCharacter(character) {
30     document.querySelector("#characters").insertAdjacentHTML(
31         "beforeend",
32         /*html*/ `
33             <article class="grid-item">
34                 <img src="${character.image}">
35                 <h2>${character.name}</h2>
36                 <p>${character.house}</p>
37             </article>
```

fetch(...) og “call stack”

7) `showCharacter` tilføjer click-event til senest tilføjede `article`. Derfor `article:last-child`. Dette matcher med, at der i 6) anvendes “beforeend” som position til at indsætte.

8) Click-eventet er `characterClicked`, der er en nested function i `showCharacter`. Det er en nested function, da vi har brug for at kunne læse `character` og sende det med som argument (parameter) til `showCharacterModal`. På denne måde svarer character altid den karakter, der bliver klikket på.

9) `showCharacterModal` tager `character` som parameter. `character` er netop den vi klikker på. `showCharacterModal` bruger værdierne i character objektet til at foretage DOM-manipulationen og opdatere de eksisterende HTML-elementer.

10) I `showCharacterModal` kaldes `generateDescription`, der er en hjælpe funktion.

11) `generateDescription` genererer en beskrivelse på baggrund af `character` objektets properties `hogwartsStaff`, `hogwartsStudent`, `alive` og `name`. Beskrivelsen returneres tilbage til `showCharacterModal` og vises.

12) Når alle informationer i dialog er sat med DOM-manipulation og `.textContent`, vises dialog’en med metoden `.showModal()`.


```
function showCharacter(character) {
    document.querySelector("#characters").insertAdjacentHTML(
        "beforeend",
        /*html*/ `
            <article class="grid-item">
                <img src="${character.image}">
                <h2>${character.name}</h2>
                <p>${character.house}</p>
            </article>
        `
    );

document.querySelector("#characters article:last-child").addEventListener("click", characterClicked);

    function characterClicked() {
        showCharacterModal(character);
    }
}

function showCharacterModal(character) {
    console.log(character);
    document.querySelector("#dialog-image").src = character.image;
    document.querySelector("#dialog-title").textContent = character.name;
    document.querySelector("#dialog-house").textContent = character.house;

    // description
    let description = generateDescription(character);
    document.querySelector("#dialog-character-description").textContent = description;

    document.querySelector("#dialog-gender").textContent = character.gender;
    document.querySelector("#dialog-birth-date").textContent = character.dateOfBirth;
    document.querySelector("#dialog-eye-color").textContent = character.eyeColour;
    document.querySelector("#dialog-hair-color").textContent = character.hairColour;
    document.querySelector("#dialog-ancestry").textContent = character.ancestry;
    document.querySelector("#dialog-species").textContent = character.species;

    document.querySelector("#dialog-name").textContent = character.name;
    document.querySelector("#dialog-actor-name").textContent = character.actor;

    // show dialog
    document.querySelector("#dialog-character").showModal();
}

function generateDescription(character) {
    let description = "";
    if (character.hogwartsStaff && character.alive) {
        description = `${character.name} is employed at Hogwarts.`;
    } else if (character.hogwartsStaff && !character.alive) {
        description = `${character.name} was employed at Hogwarts but is no longer alive.`;
```

JSON

... a syntax for storing and exchanging data over the web

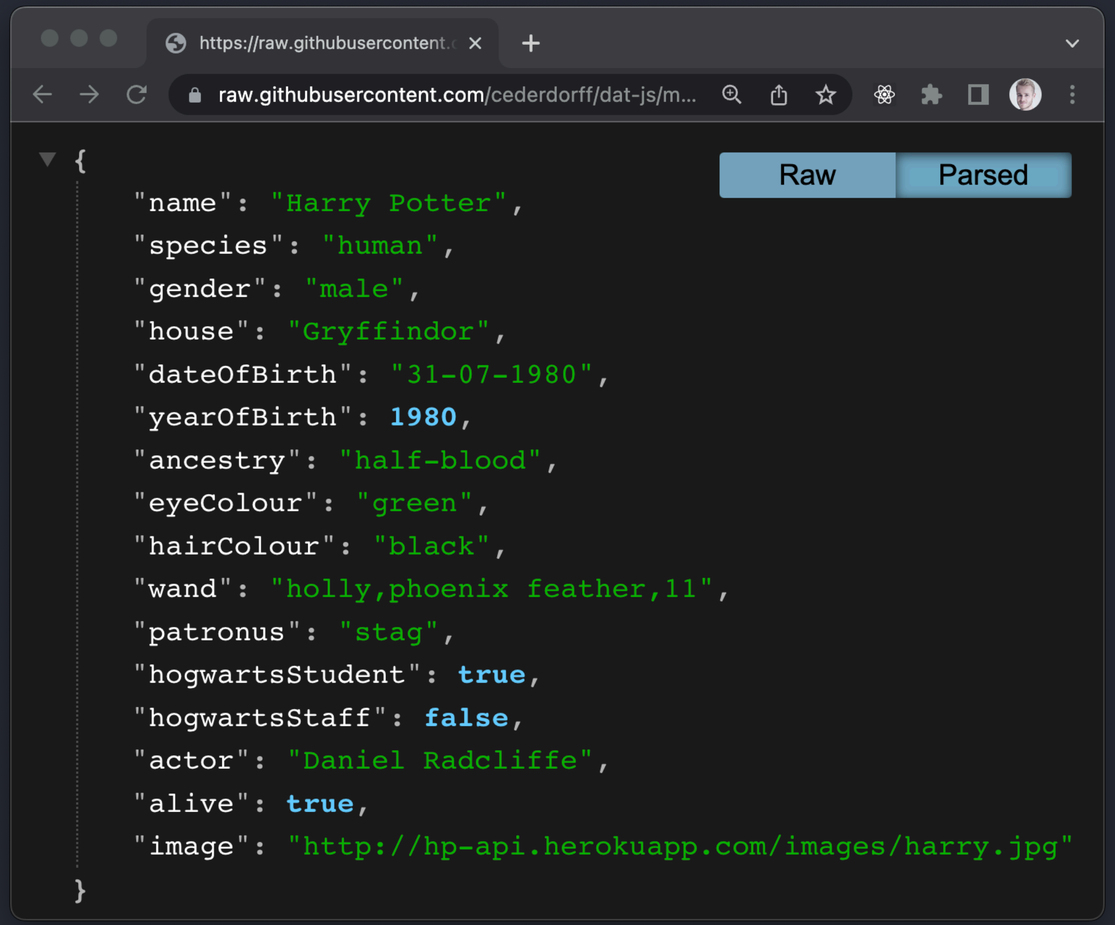
```
{
  "name": "Alicia",
  "age": 6
}
```

JSON OBJECT

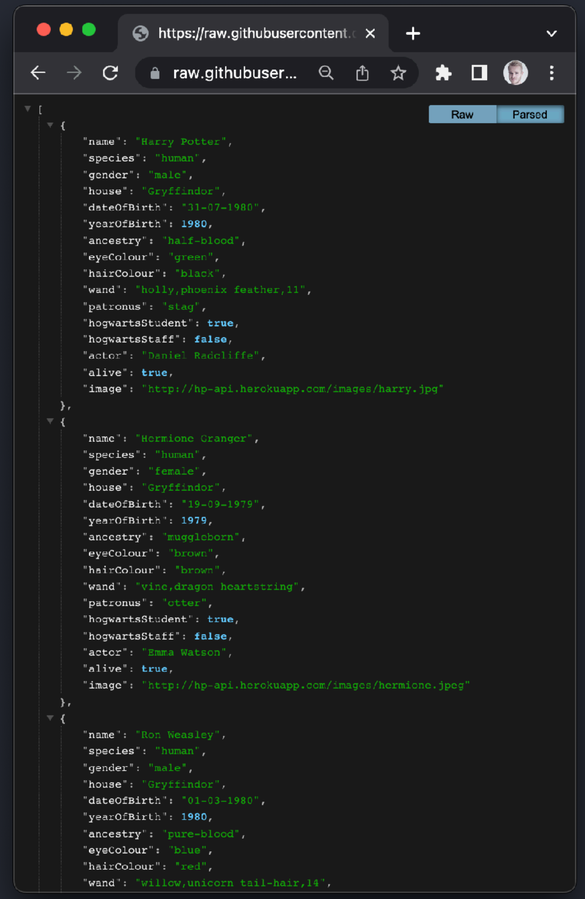
```
[{
  "name": "Alicia",
  "age": 6
}, {
  "name": "Peter",
  "age": 22
}]
```

LIST OF JSON OBJECTS

JSON



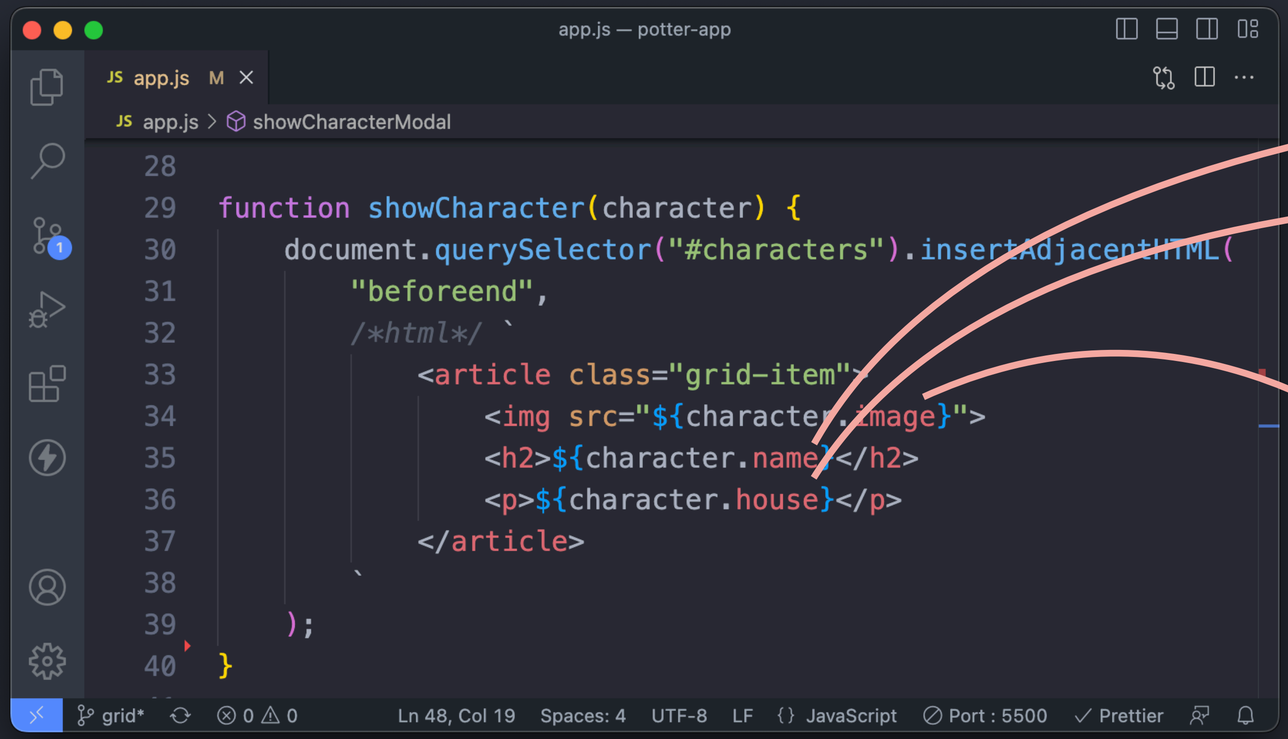
JSON OBJECT

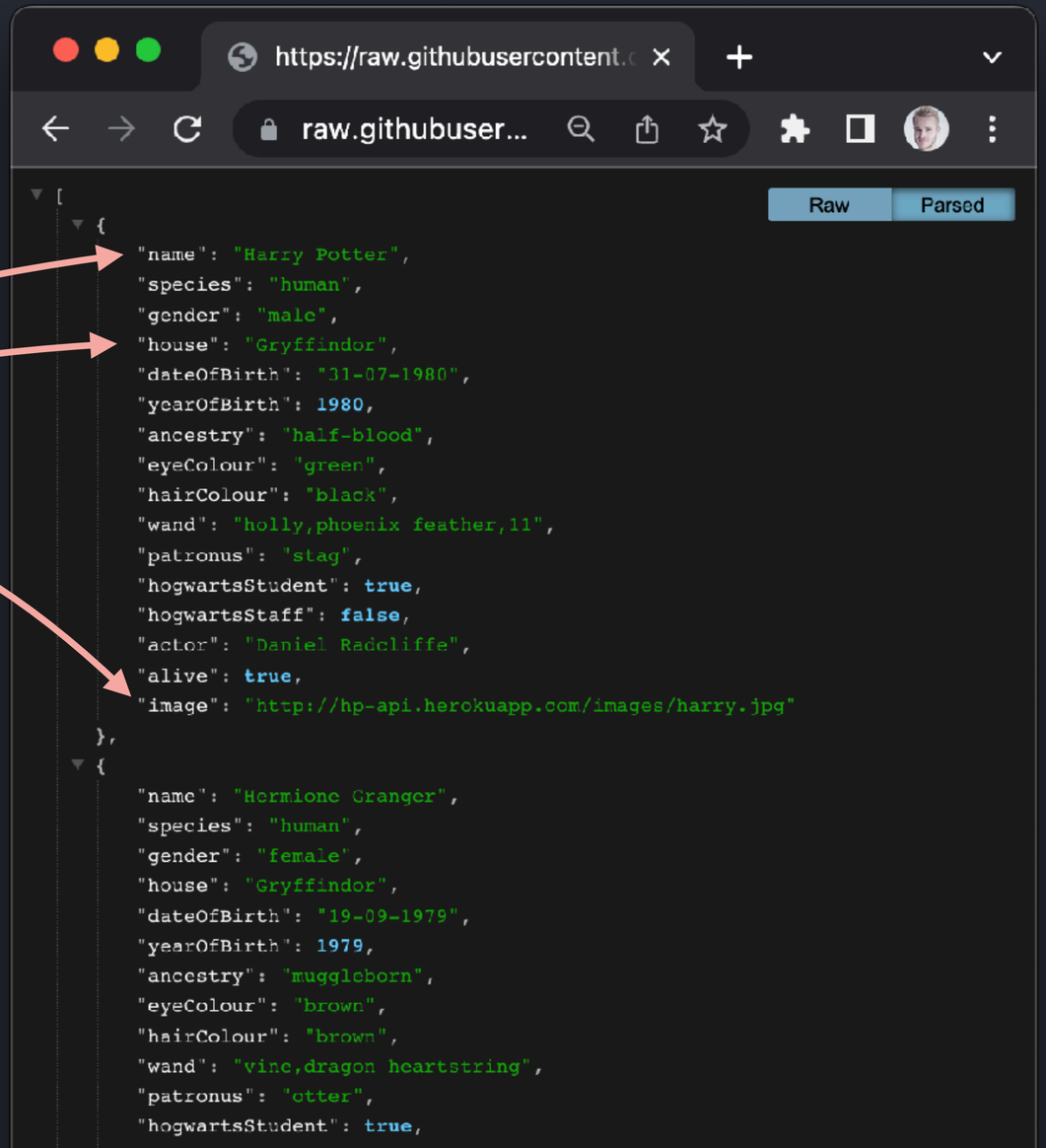


LIST OF JSON OBJECTS

https://www.instagram.com/p/CVqbCzgsZUF/

JSON


app.js — potter-app
app.js M
app.js > showCharacterModal
function showCharacter(character) {
document.querySelector("#characters").insertAdjacentHTML(
"beforeend",
/*html*/ `
<article class="grid-item">
<img src="${character.image}">
<h2>${character.name}</h2>
${character.house}
</article>
`
);
}
grid* 0 0 Ln 48, Col 19 Spaces: 4 UTF-8 LF JavaScript Port : 5500 Prettier

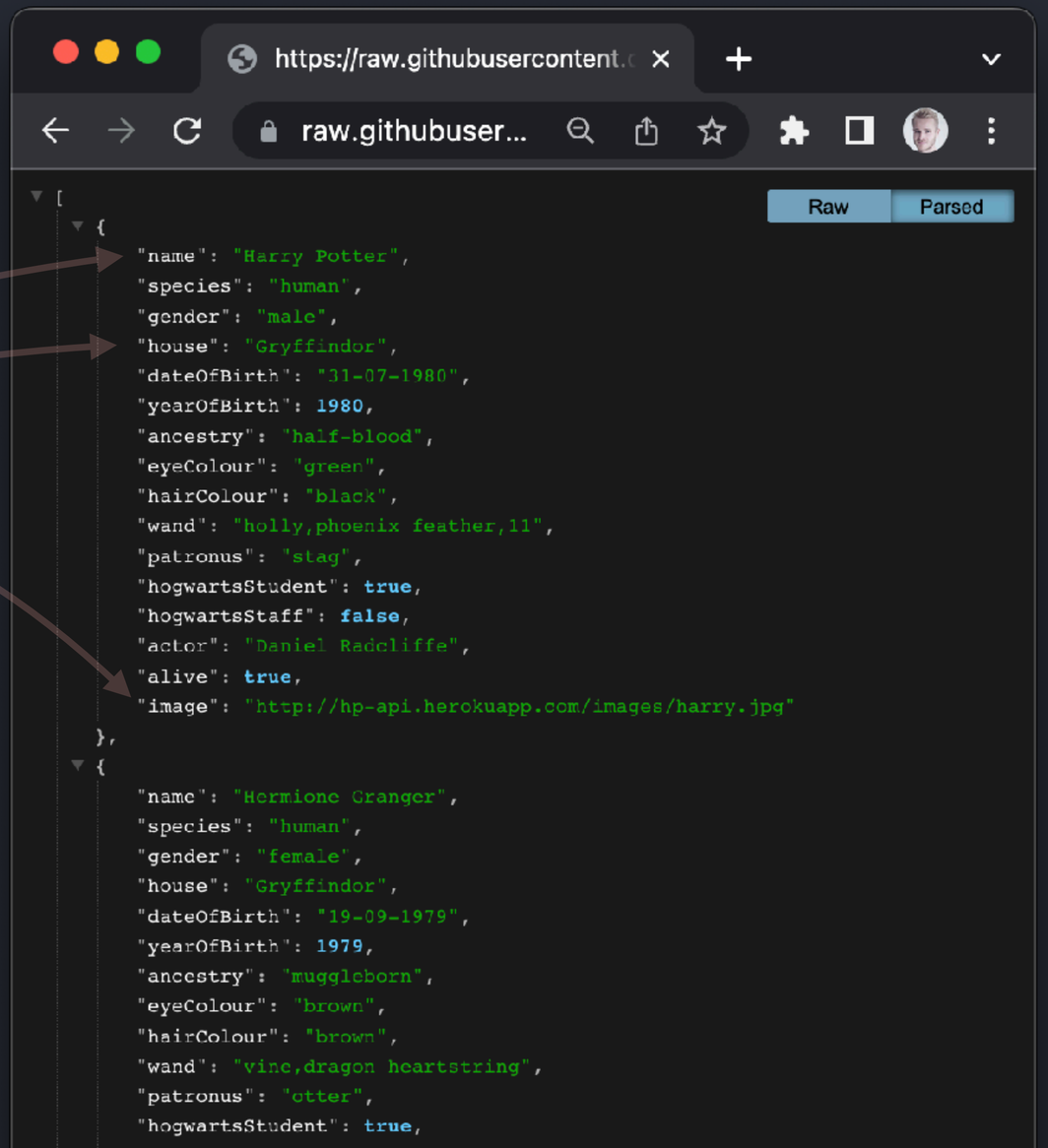

https://raw.githubusercontent.
raw.githubuser...
Raw
Parsed
[
{
"name": "Harry Potter",
"species": "human",
"gender": "male",
"house": "Gryffindor",
"dateOfBirth": "31-07-1980",
"yearOfBirth": 1980,
"ancestry": "half-blood",
"eyeColour": "green",
"hairColour": "black",
"wand": "holly,phoenix feather,11",
"patronus": "stag",
"hogwartsStudent": true,
"hogwartsStaff": false,
"actor": "Daniel Radcliffe",
"alive": true,
"image": "http://hp-api.herokuapp.com/images/harry.jpg"
},
{
"name": "Hermione Granger",
"species": "human",
"gender": "female",
"house": "Gryffindor",
"dateOfBirth": "19-09-1979",
"yearOfBirth": 1979,
"ancestry": "muggleborn",
"eyeColour": "brown",
"hairColour": "brown",
"wand": "vine,dragon heartstring",
"patronus": "otter",
"hogwartsStudent": true,

Review JSON-objekt

Det er vigtigt at vores properties matcher med det forventede! Det skal matche med den datamodel vi er bleve enige om: https://cederdorff.github.io/dat-js/05-data/

Kontroller:

- Values - er der værdier i properties? Se hvilke her.
- Navngivning af properties - ja også store og små bogstaver 🤦‍♀️ Se navngivning her.
- Er det den rigtige datatype? Brug dem vi er blevet enige om - se her. Det må fx ikke være et array, hvis vi er blevet enige om en kommasepareret tekststreng.
- Dobbelttjek properties og values matcher med: https://cederdorff.github.io/dat-js/05-data/
- Din billed url. Er det et billede? Kontroller i browser. Du skal højeklikke på billede og vælge “Copy Image Address” eller “Copy Image Link” - ikke bare “Copy Link”.
- Foretag tilpasninger af dit JSON-objekt (fra sidste uge), ikke noget med et array. Commit og push, så vi kan få de tilrettede objekter med.



Loops

Iterate over arrays or other iterable objects.

Ex loop through an array of objects.

```
// .forEach
characterList.forEach(showCharacter);

// for of loop
for (const character of characterList) {
    showCharacter(character);
}

// for loop
for (let index = 0; index < characterList.length; index++) {
    const character = characterList[index];
    showCharacter(character);
}
```

.forEach

an array method

```
const names = ["Peter", "Lenka", "Oskar", "Rasmus"];

names.forEach(showNames);

function showNames(name) {
    console.log(name);
}
```

You can use it on an array to do something for each element in the array.

`forEach` will call a function for each element in an array.

https://www.w3schools.com/jsref/jsref_foreach.asp

For of

Loop

```
const names = ["Peter", "Lenka", "Oskar", "Rasmus"];

for (const name of names) {
    console.log(name);
}
```

You can use it to loop over an array to do something for every element.

`for of` loops through the values of an iterable object.

https://www.w3schools.com/js/js_loop_forof.asp

For of loop

iterate over arrays or other iterable objects

https://scrimba.com/learn/introductiontojavascript/for-loops-cMMM8U9

https://scrimba.com/learn/introductiontojavascript/challenge-for-loops-cPkpJrcv

For

Loop

```
const names = ["Peter", "Lenka", "Oskar", "Rasmus"];

for (let index = 0; index < names.length; index++) {
    const name = names[index];
    console.log(name);
}
```

You can use it to loop over an array.

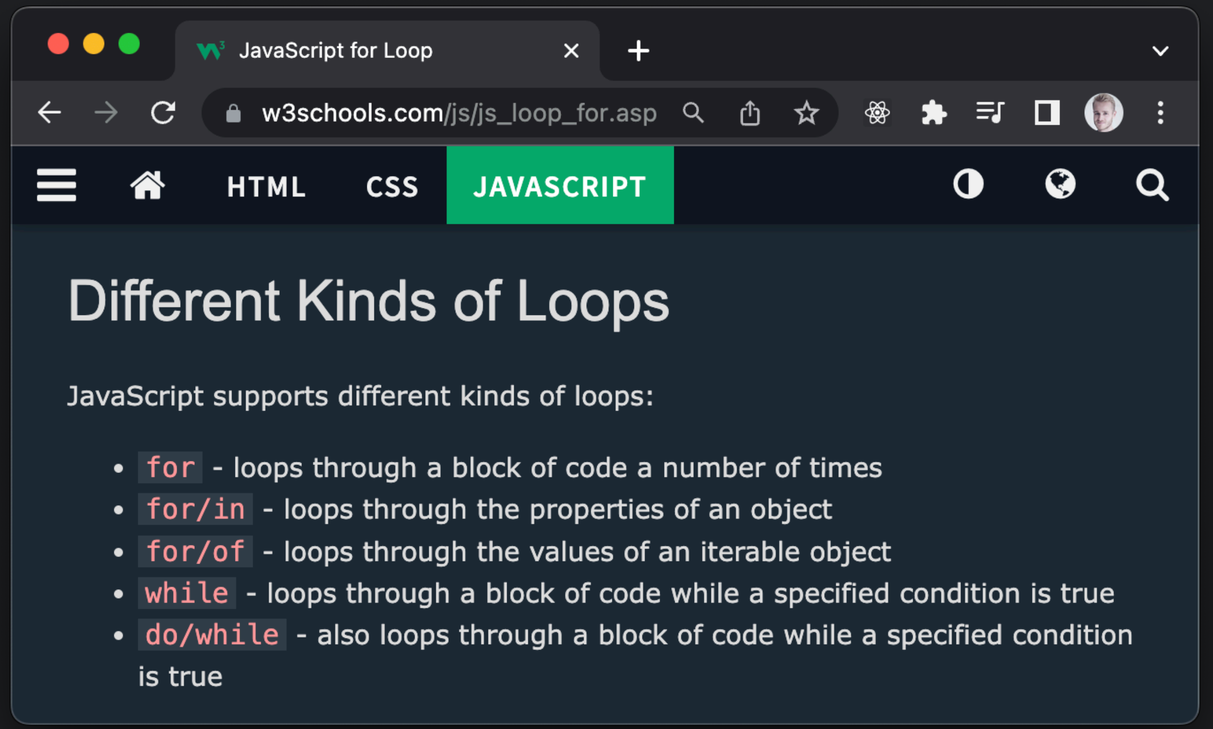
A `for loop` repeats until a specified condition is false.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration#for_statement
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/for
https://www.w3schools.com/js/js_loop_for.asp

Loops

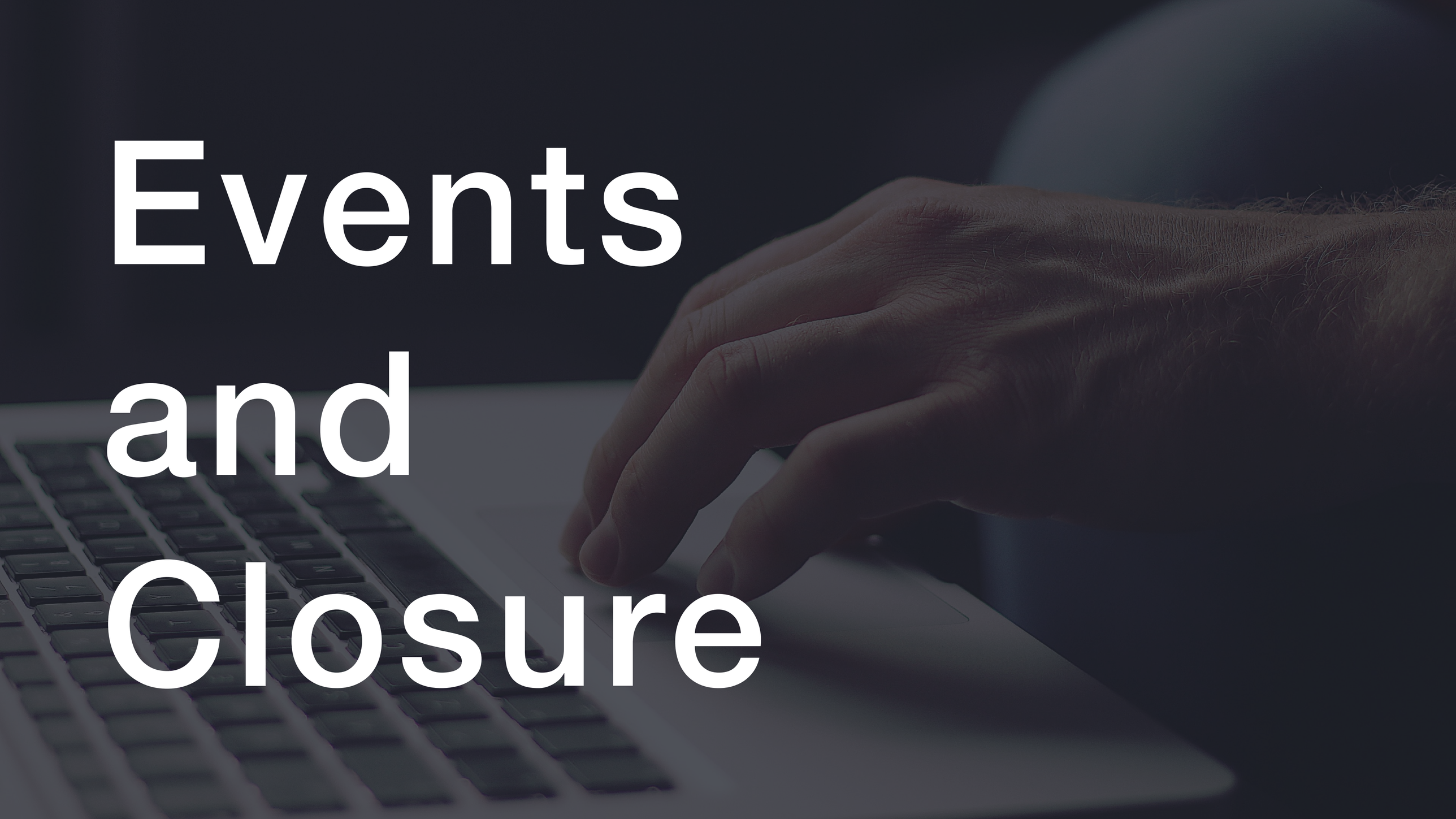
Even more options 🎉

But use `forEach` or `for of` for now.



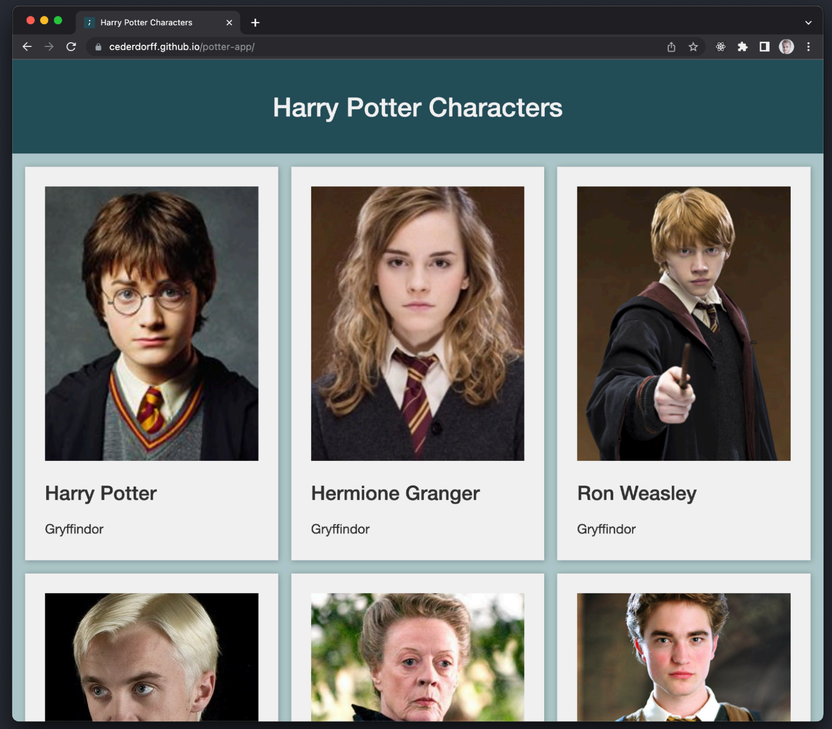
https://www.w3schools.com/js/js_loop_for.asp

Events and Closure



EventListener

The eventListener attaches an event handler to the specified element

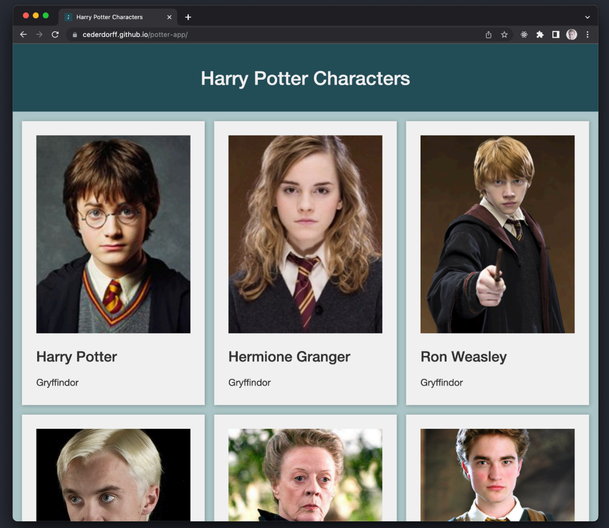



```
document.querySelector("#characters article:last-child")
.addEventListener("click", characterClicked);


function characterClicked() {
    // do something
}
```

https://www.w3schools.com/js/js_htmldom_events.asp
https://www.w3schools.com/js/js_htmldom_eventlistener.asp

EventListener

Vi tilføjer event for hver `article` som vi indsætter i DOM'en

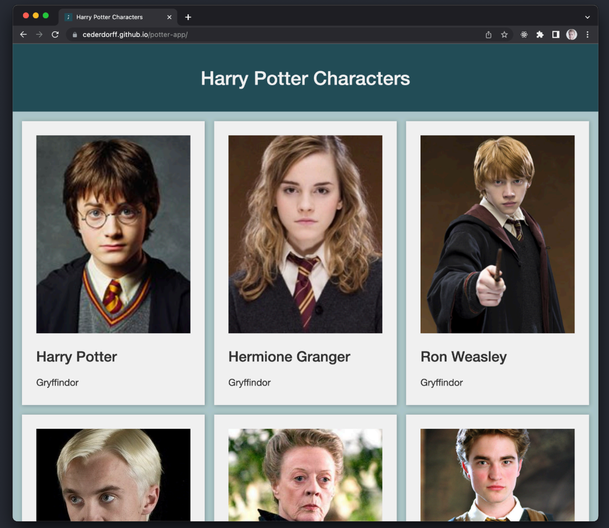



```
function showCharacter(character) {
    document.querySelector("#characters").insertAdjacentHTML(
        "beforeend",
        /*html*/ `
            <article class="grid-item">
                <img src="${character.image}">
                <h2>${character.name}</h2>
                <p>${character.house}</p>
            </article>
        `
    );

    document.querySelector("#characters article:last-child").addEventListener("click", characterClicked);
}
```

https://www.w3schools.com/js/js_htmldom_events.asp
https://www.w3schools.com/js/js_htmldom_eventlistener.asp

Nested functions & closure

Men vi skal kende `character`, for at kunne vise lige præcis den `character` vi klikker på.

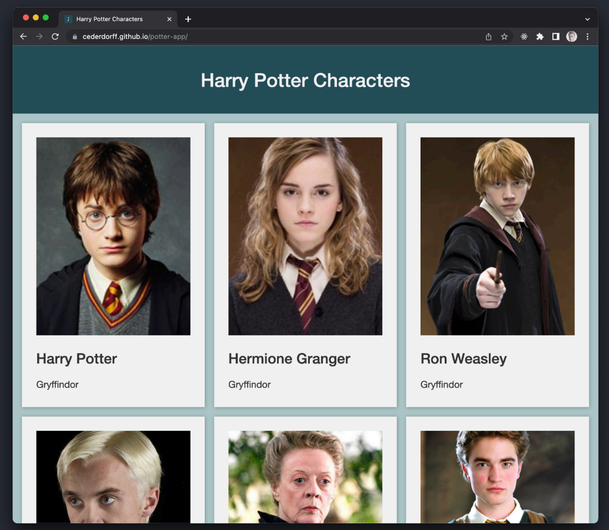



```
function showCharacter(character) {
    document.querySelector("#characters").insertAdjacentHTML(
        "beforeend",
        /*html*/ `
            <article class="grid-item">
                <img src="${character.image}">
                <h2>${character.name}</h2>
                <p>${character.house}</p>
            </article>
        `
    );

    document.querySelector("#characters article:last-child").addEventListener("click", characterClicked);

    function characterClicked() {
        showCharacterModal(character);
    }
}
```

https://www.w3schools.com/js/js_function_closures.asp

Nested functions & closure

Derfor har vi en funktion i en funktion. Så vil `characterClicked` (nested function) have adgang til `showCharacter`'s scope, og der ved kan vi læse `character`




```
function showCharacter(character) {
    document.querySelector("#characters").insertAdjacentHTML(
        "beforeend",
        /*html*/ `
            <article class="grid-item">
                <img src="${character.image}">
                <h2>${character.name}</h2>
                <p>${character.house}</p>
            </article>
        `
    );

    document.querySelector("#characters article:last-child").addEventListener("click", characterClicked);

    function characterClicked() {
        showCharacterModal(character);
    }
}
```

https://www.w3schools.com/js/js_function_closures.asp

Nested function

A function inside a function.

All functions have access to the global scope.

Nested functions have access to the scope "above" them

Closure

A closure is a function that have access to the parent scope - also after the parent function has closed / has been executed

```
function showCharacter(character) {
    document.querySelector("#characters").insertAdjacentHTML(
        "beforeend",
        /*html*/ `
            <article class="grid-item">
                <img src="${character.image}">
                <h2>${character.name}</h2>
                <p>${character.house}</p>
            </article>
        `
    );

    document.querySelector("#characters article:last-child").addE

    function characterClicked() {
        showCharacterModal(character);
    }
}
```

https://www.w3schools.com/js/js_function_closures.asp

DOM Manipulation

> When you use JavaScript to add, remove, and modify elements of a website

- document.querySelector(selector)
- document.querySelectorAll(name)
- document.createElement(name)
- parentNode.appendChild(node)
- element.append(node, node, …)
- element.insertAdjacentHTML(position, html)
- element.innerHTML
- element.textContent
- element.style.xxx
- element.setAttribute()
- element.getAttribute()
- element.addEventListener()
- element.classList
- window.content
- Window.onload
- window.scrollTo()

Methods for DOM Manipulation

Common Methods

```
// .createElement and .appendChild
const newTitle = document.createElement("h1");
newTitle.textContent = "Hi!";
document.querySelector("body").appendChild(newTitle);

// backticks and .insertAdjacentHTML
const secondTitle = /*html*/ `<h1>Hi, again!</h1>`;
document.querySelector("body").insertAdjacentHTML(secondTitle);

// .textContent - must have defined an h1 element in your HTML
const thirdTitle = document.querySelector("h1");
thirdTitle.textContent = "Hi, again, again!";
```

Vi har anvendt

... i hvertfald for nu

https://www.w3schools.com/jsref/prop_node_textcontent.asp

DOM Manipulation

```
let fullName = "Peter Lind";
console.log(fullName);
document.querySelector("#fullName_container").textContent = fullName;
```

.textContent

Get or set the text content of a given (HTML) element

https://www.w3schools.com/jsref/prop_node_textcontent.asp

.textContent

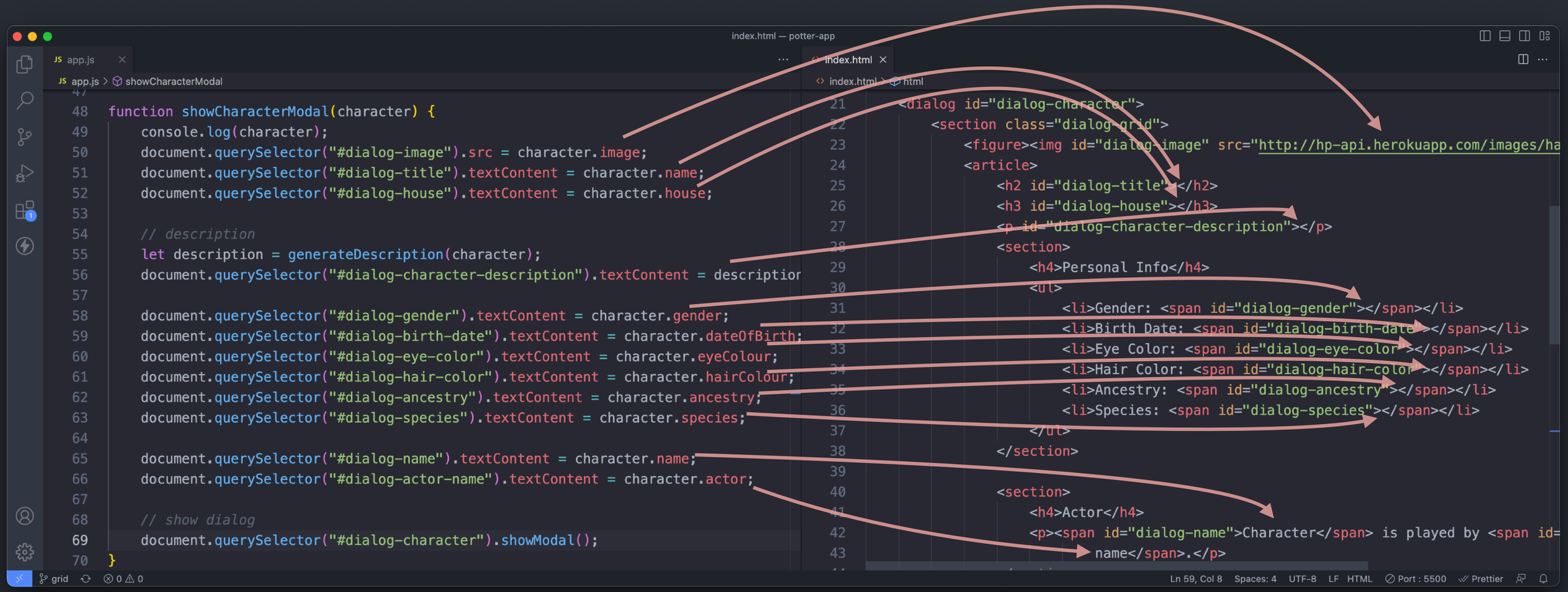
Get or set the text content of a given (HTML) element. The element muse be in the DOM.

```
let fullName = "Peter Lind";
console.log(fullName);
document.querySelector("#fullName_container").textContent = fullName;
```

```
<body>
    <header>
        <h1 id="fullName_container"></h1>
    </header>
    <script src="app.js"></script>
</body>
```

.textContent

Get or set the text content of a given (HTML) element. The element muse be in the DOM.



document.createElement()

```
// create a new h1 element
const newTitle = document.createElement("h1");

// and give it some content
newTitle.textContent = "Hi there and greetings!";

// add the newly created element into the DOM
document.querySelector("body").appendChild(newTitle);
```

.createElement

Creates an HTML element specified by tagName.

.appendChild

Appends an element as last child of an element.

https://developer.mozilla.org/en-US/docs/Web/API/Document/createElement
https://developer.mozilla.org/en-US/docs/Web/API/Node/appendChild

insertAdjacentHTML(position, html)

```
// 1

// create a new h1 with content using a backtick
const newTitle = /*html*/ `<h1>Hi there and greetings!</h1>`;

// add the newly created element into the DOM
document.querySelector("body").insertAdjacentHTML("beforeend", newTitle);

// 2

// variable holding username
const username = "RACE";

// create a new p with content using a backtick and username variable
const newParagraph = /*html*/ `<p>Welcome, ${username}!</p>`;

// add the newly created element into the DOM
document.querySelector("body").insertAdjacentHTML("beforeend", newParagraph);
```

Backtick string

Create multiline strings (templates) with embedded expressions and tags.

.insertAdjacentHTML

Inserts HTML into a specified position.

https://www.w3schools.com/js/js_string_templates.asp
https://www.w3schools.com/jsref/met_node_insertadjacenthtml.asp

showCharacter(character)

```
function showCharacter(character) {
    document.querySelector("#characters").insertAdjacentHTML(
        "beforeend",
        /*html*/ `
            <article class="grid-item">
                <img src="${character.image}">
                <h2>${character.name}</h2>
                <p>${character.house}</p>
            </article>
        `
    );
}
```

Backtick string

Create multiline strings (templates) with embedded expressions and tags.

.insertAdjacentHTML

Inserts HTML into a specified position.

https://www.w3schools.com/js/js_string_templates.asp
https://www.w3schools.com/jsref/met_node_insertadjacenthtml.asp

Hvorfor ikke .innerHTML?

```
function showCharacter(character) {
    document.querySelector("#characters").innerHTML = /*html*/ `
            <article class="grid-item">
                <img src="${character.image}">
                <h2>${character.name}</h2>
                <p>${character.house}</p>
            </article>
        `;
}
```

Backtick string

Create multiline strings (templates) with embedded expressions and tags.

.innerHTML

... sets or returns the HTML content (inner HTML) of an element.

“To insert the HTML into the document rather than replace the contents of an element, use the method insertAdjacentHTML().”

https://www.w3schools.com/jsref/prop_html_innerhtml.asp
https://developer.mozilla.org/en-US/docs/Web/API/Element/innerHTML

.innerHTML & events

```
function showCharacter(character) {
    document.querySelector("#characters").innerHTML = /*html*/ `
            <article class="grid-item">
                <img src="${character.image}">
                <h2>${character.name}</h2>
                <p>${character.house}</p>
            </article>
        `;

    document.querySelector("#characters article:last-child").addEventListener("click", characterClicked);

    function characterClicked() {
        showCharacterModal(character);
    }
}
```

Backtick string

Create multiline strings (templates) with embedded expressions and tags.

.innerHTML

“Please note that using innerHTML to append HTML elements [...] will result in the removal of any previously set event listeners. That is, after you append any HTML element that way you won't be able to listen to the previously set event listeners.”

https://www.w3schools.com/jsref/prop_html_innerhtml.asp
https://developer.mozilla.org/en-US/docs/Web/API/Element/innerHTML

Which on?

```
// backticks and .insertAdjacentHTML
const secondTitle = /*html*/ `<h1>Hi, again!</h1>`;
document.querySelector("body").insertAdjacentHTML(secondTitle);

// .textContent - must have defined an h1 element in your HTML
const thirdTitle = document.querySelector("h1");
thirdTitle.textContent = "Hi, again, again!";
```

.insertAdjacentHTML

Inserts HTML into a specified position.

.textContent

Get or set the text content of a given (HTML) element

https://www.w3schools.com/jsref/prop_node_textcontent.asp

Which on?

.insertAdjacentHTML

Inserts HTML into a specified position.

```
function showCharacter(character) {
    document.querySelector("#characters").insertAdjacentHTML(
        "beforeend",
        /*html*/ `
            <article class="grid-item">
                <img src="${character.image}">
                <h2>${character.name}</h2>
                <p>${character.house}</p>
            </article>
        `
    );
    //...
}
```

.textContent

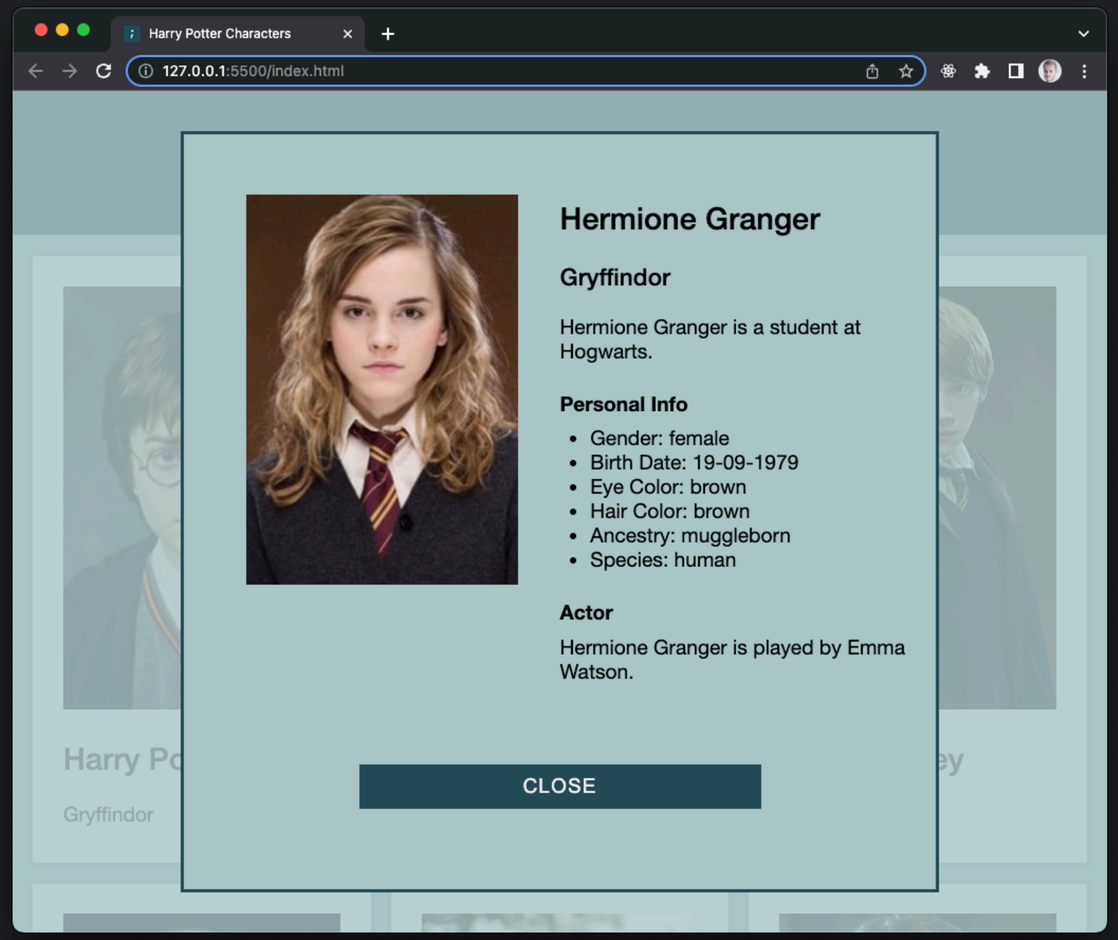
Get or set the text content of a given (HTML) element

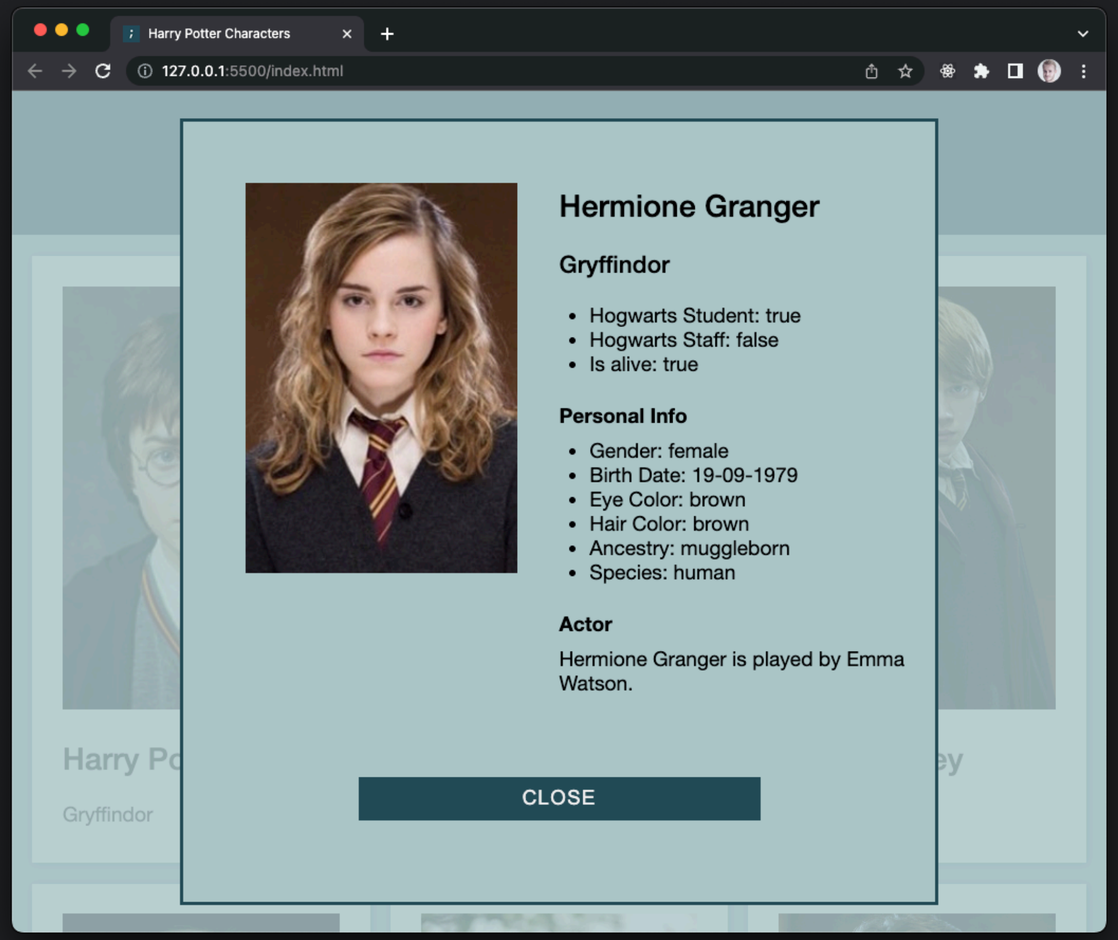
```
function showCharacterModal(character) {
    console.log(character);
    document.querySelector("#dialog-image").src = character.image;
    document.querySelector("#dialog-title").textContent = character.name;
    document.querySelector("#dialog-house").textContent = character.house;

    // description
    let description = generateDescription(character);
    document.querySelector("#dialog-character-description").textContent = description;

    document.querySelector("#dialog-gender").textContent = character.gender;
    document.querySelector("#dialog-birth-date").textContent = character.dateOfBirth;
    document.querySelector("#dialog-eye-color").textContent = character.eyeColour;
    document.querySelector("#dialog-hair-color").textContent = character.hairColour;
    document.querySelector("#dialog-ancestry").textContent = character.ancestry;
    document.querySelector("#dialog-species").textContent = character.species;

    document.querySelector("#dialog-name").textContent = character.name;
    document.querySelector("#dialog-actor-name").textContent = character.actor;

    // show dialog
    document.querySelector("#dialog-character").showModal();
}
```

Specialisering af output

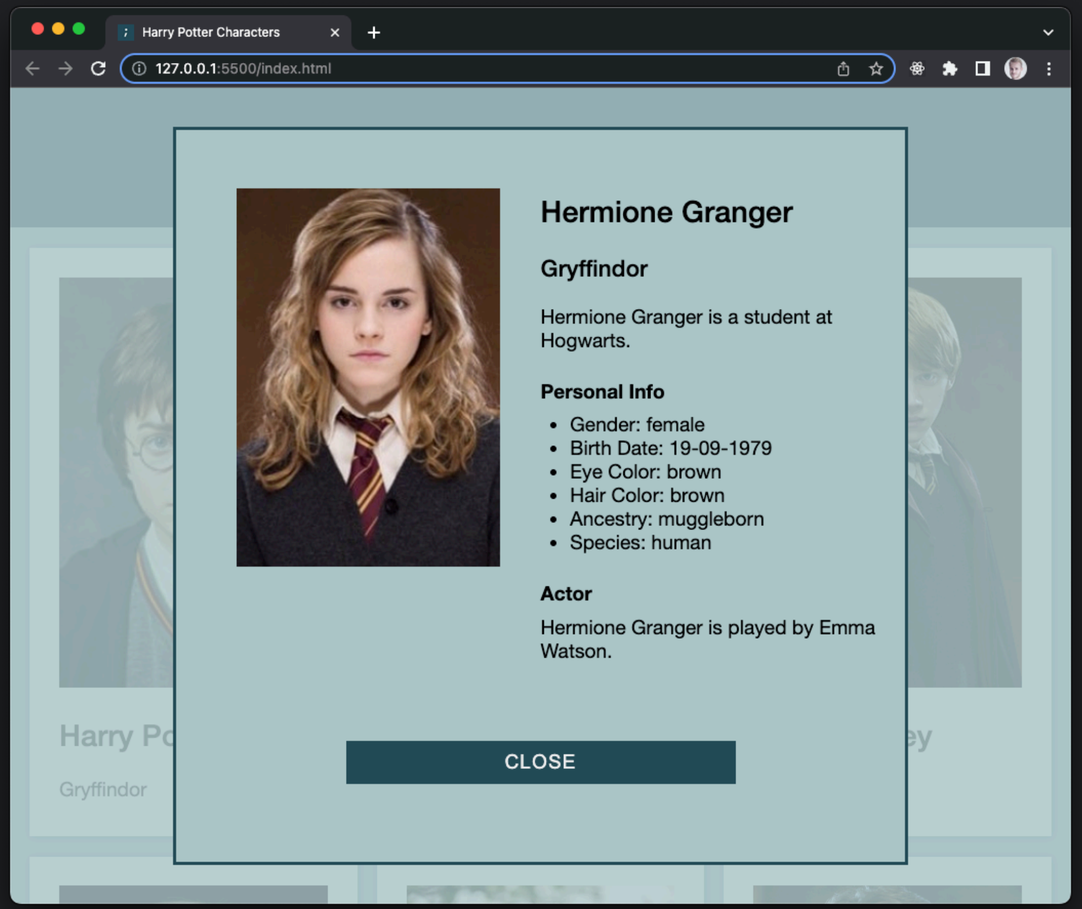
“Hmm, det handler vel om a vise det “rigtige”, og det der giver bedst mening for brugeren.”

Hvis du nu ikke kendte til programmering?





Forbedringer



- Er der noget, vi gerne vil vise på en særlig måde, illustrerer, tilpasse osv?

Specialisering af styling

Hvad nu, hvis du vil differentiere stylingen af dine grid-elementer eller dialog?


Code
Every
Day