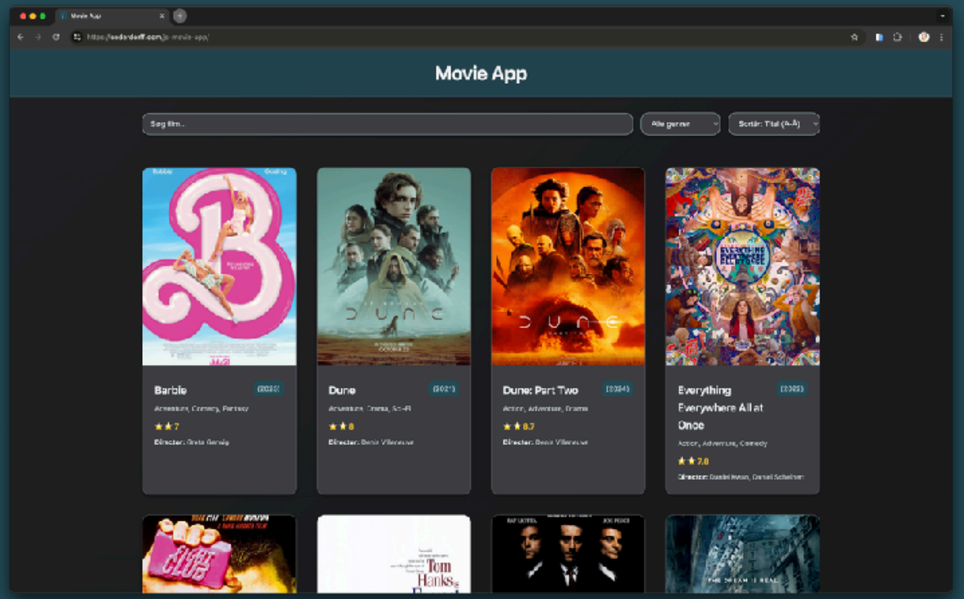
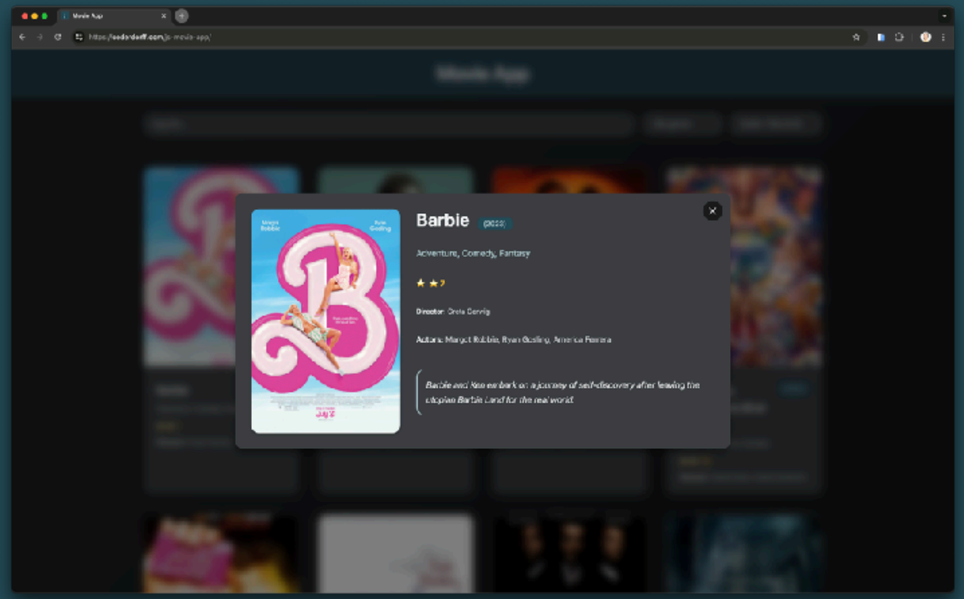
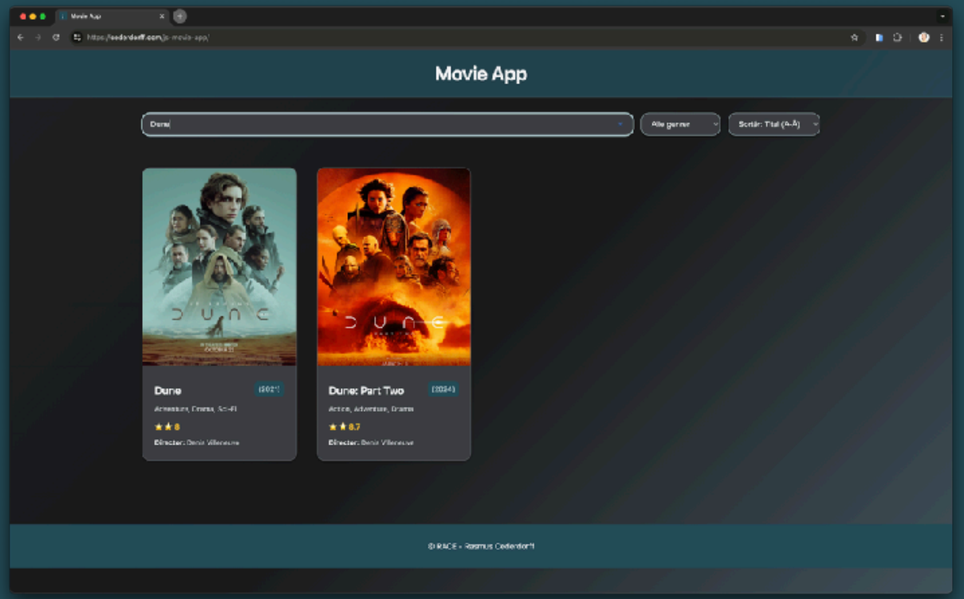
ERHVERVSAKADEMI AARHUS
BUSINESS ACADEMY AARHUS

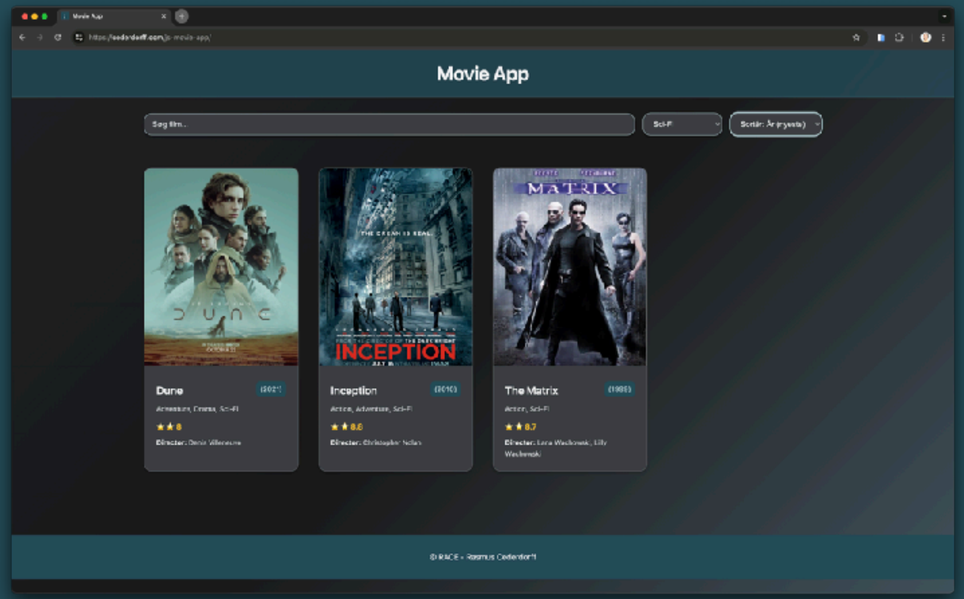
1. Semester - JavaScript

Client-Server, Variabler, Datatyper, Template String, Objects, Arrays, Funktioner, DOM-Manipulation, CSS Grid & Fetch







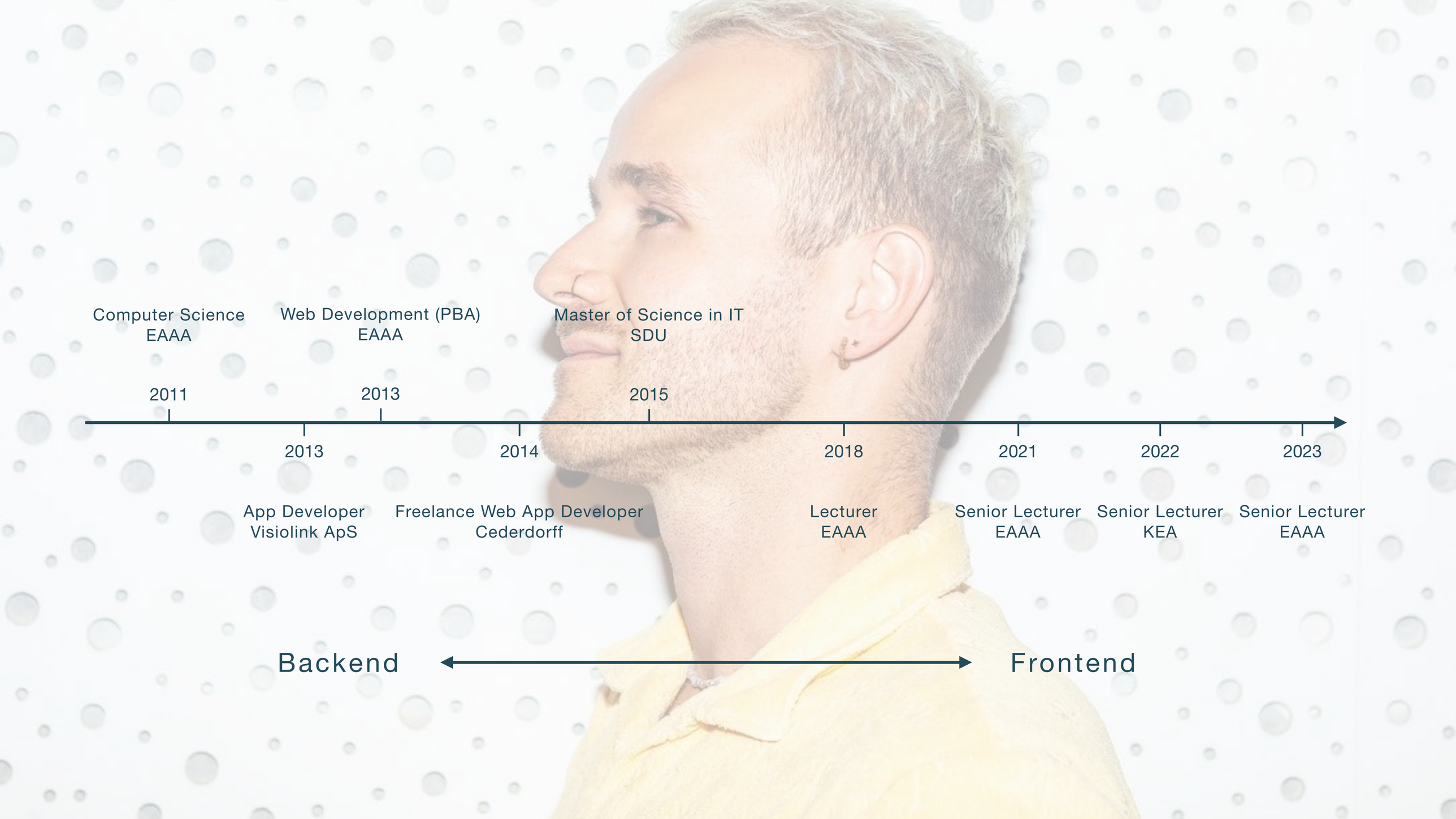


https://cederdorff.com/js-movie-app/

I’m Rasmus Cederdorff (RACE)

Senior Lecturer
Freelance Web App Developer

- Programming with an eye for UI and UX.
- Web Development, JavaScript, React, BaaS & Node.js
- “I speak JavaScript”.
- Websites, Webshops, Web Apps, Mobile Apps, Server App and BaaS.


Computer Science
EAAA
2011
Web Development (PBA)
EAAA
2013
Master of Science in IT
SDU
2015
2013
App Developer
Visiolink ApS
2014
Freelance Web App Developer
Cederdorff
2018
Lecturer
EAAA
2021
Senior Lecturer
EAAA
2022
Senior Lecturer
KEA
2023
Senior Lecturer
EAAA
Backend
Frontend

I’m Rasmus Cederdorff

Holstebro

Alicia & Ida

From Holstebro

I’m into sports

Love (🍎) gadgets, to take pictures & interior design projects

What’s with my arm?

“Learning to code requires a HUGE investment of time and energy.”

https://medium.com/martinsoft/learn-javascript-c1cca9db9015

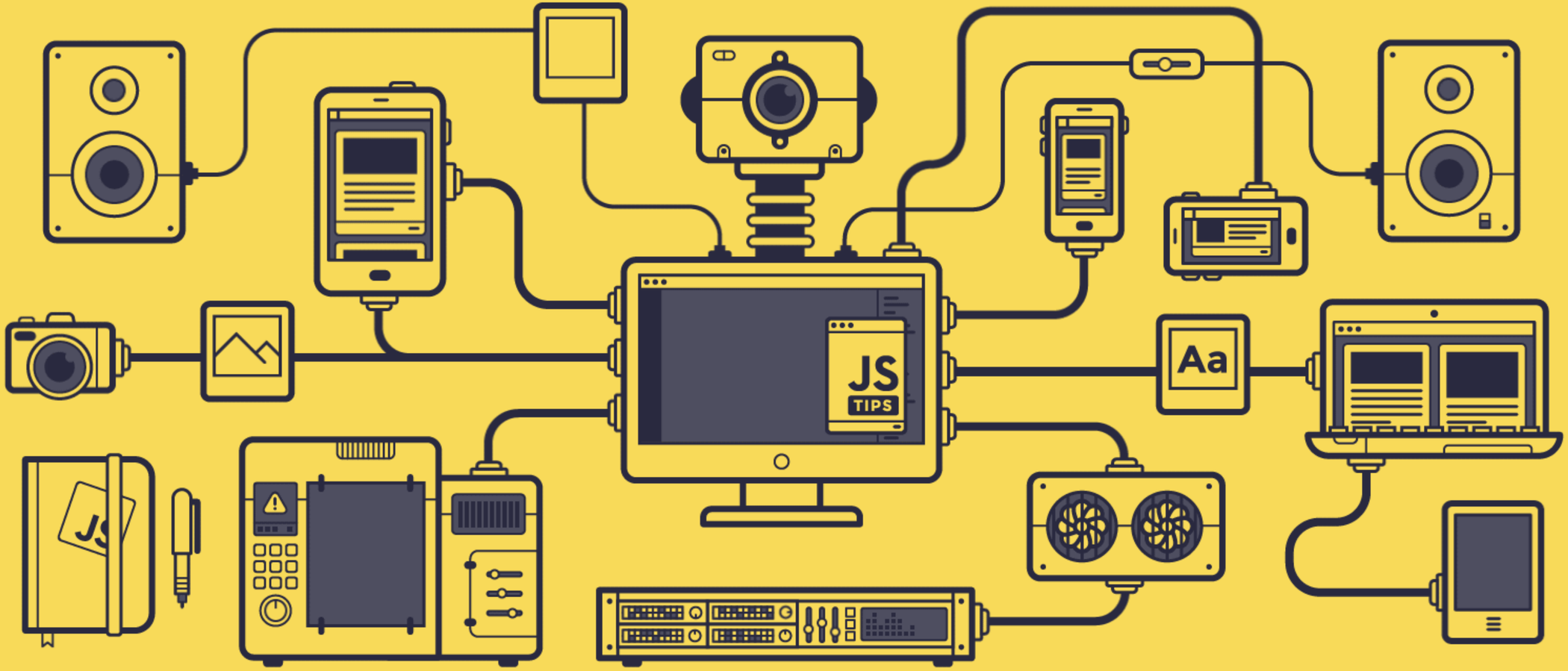
“The cool thing about JavaScript is that you can create stuff that will put a smile on your face, as a developer.

You can create stuff and it will visually look like something, and it'll do stuff, and it makes you feel good, **it makes you fall in love with programming.**”

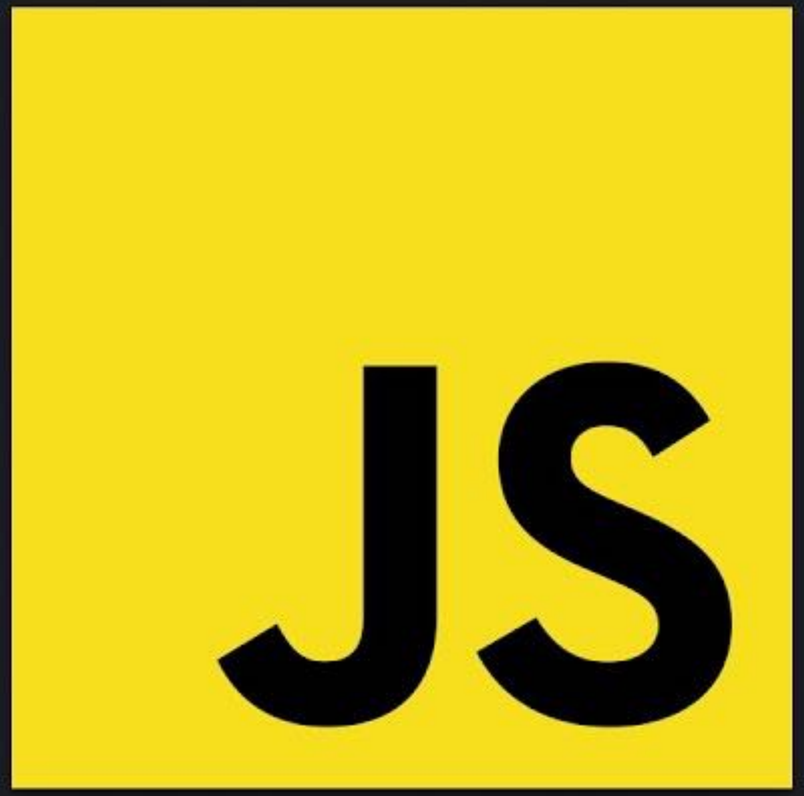
Lex Fridman

Computer Scientist

https://www.youtube.com/watch?v=GLhyjVZp0cw&t=252s


JS
TIPS
Aa

100 SECONDS OF



https://www.youtube.com/watch?v=DHjqpvDnNGE

JS

101

https://www.youtube.com/watch?v=lklFF4maKMU

index.html

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
    <link rel="stylesheet" href="styles.css" />
  </head>
  <body>
    <h1>This is a Heading</h1>
    <p>This is a paragraph.</p>
    <button onclick="tryMe()">Try me</button>
    <script src="app.js"></script>
  </body>
</html>
```

app.js

```js
function tryMe() {
  document.body.style.backgroundColor = "red";
  document.body.style.color = "white";
}
```

What is JavaScript?

.. is the world's most popular programming language.

... is the programming language of the Web.

... is easy to learn.

... can change content of a webpage (HTML content).

... can change styling of HTML.

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

index.html

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
    <link rel="stylesheet" href="styles.css" />
  </head>
  <body>
    <h1>This is a Heading</h1>
    <p>This is a paragraph.</p>
    <button onclick="tryMe()">Try me</button>
    <script src="app.js"></script>
  </body>
</html>
```

app.js

```js
function tryMe() {
  document.body.style.backgroundColor = "red";
  document.body.style.color = "white";
}
```

With JavaScript we are able to

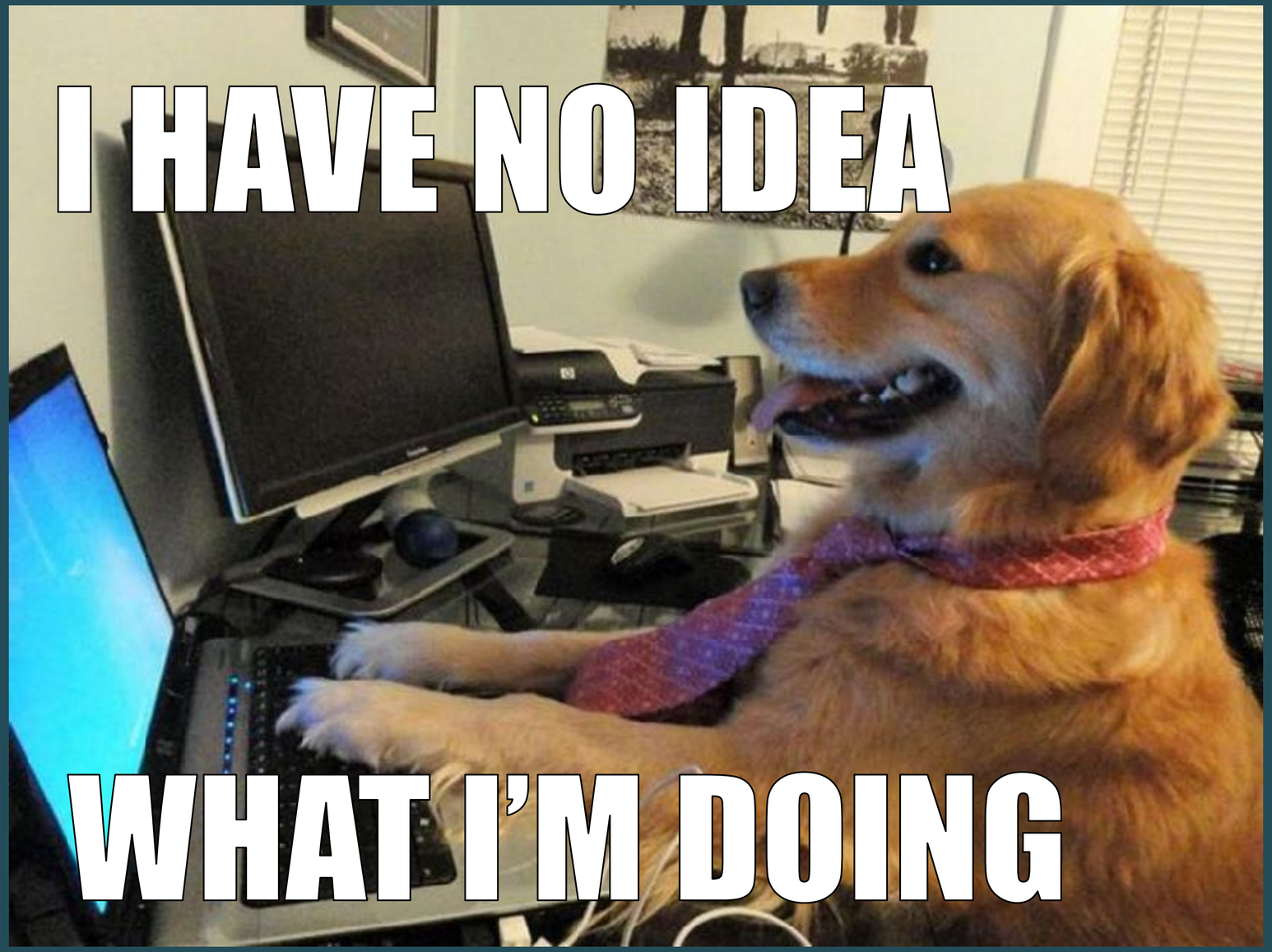
... build dynamic web pages and web apps.

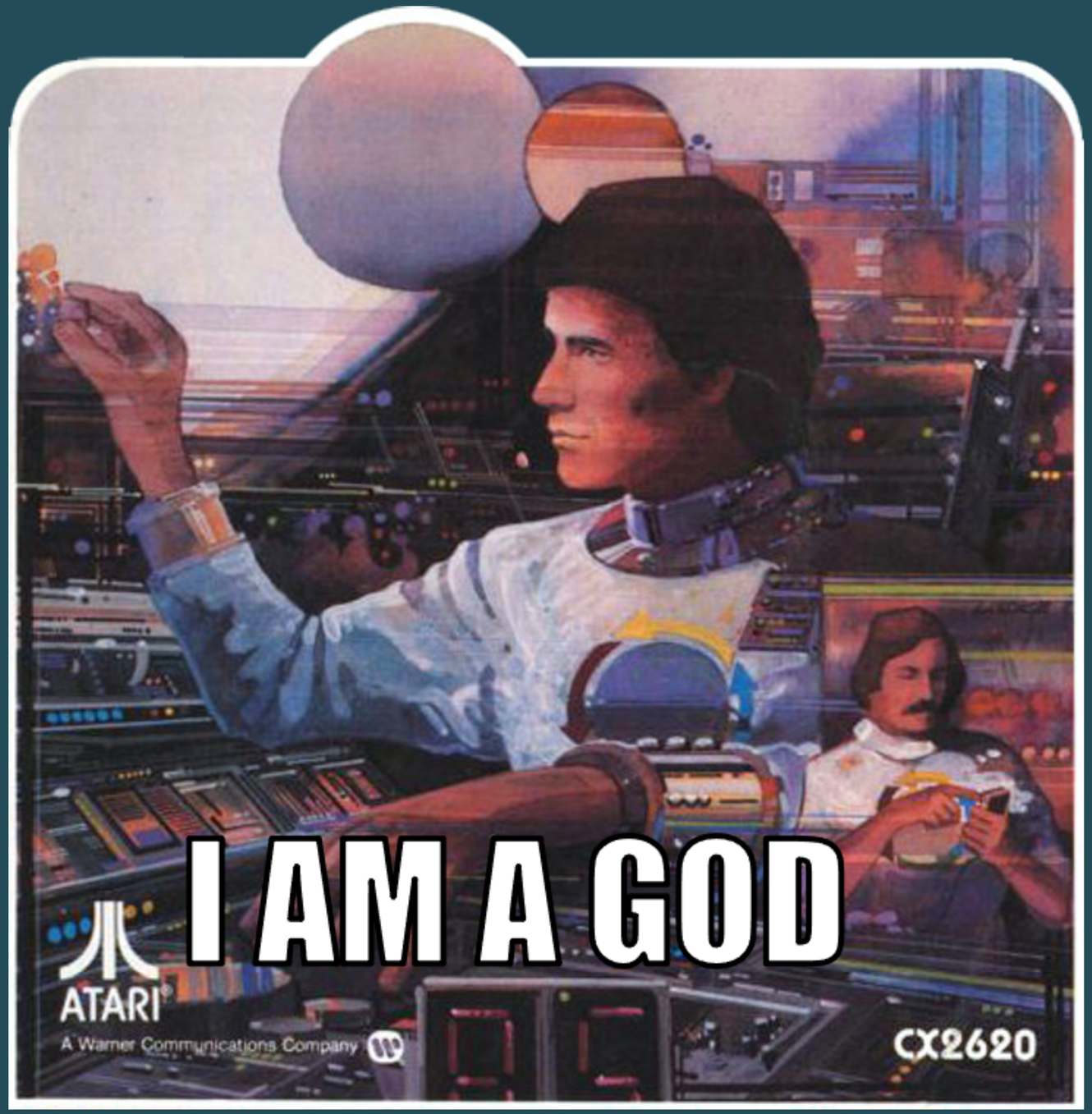
... fetch content/ data from a backend (web service, data source, etc.) through an API.

... do DOM-manipulation.

... build and develop anything 🤷

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

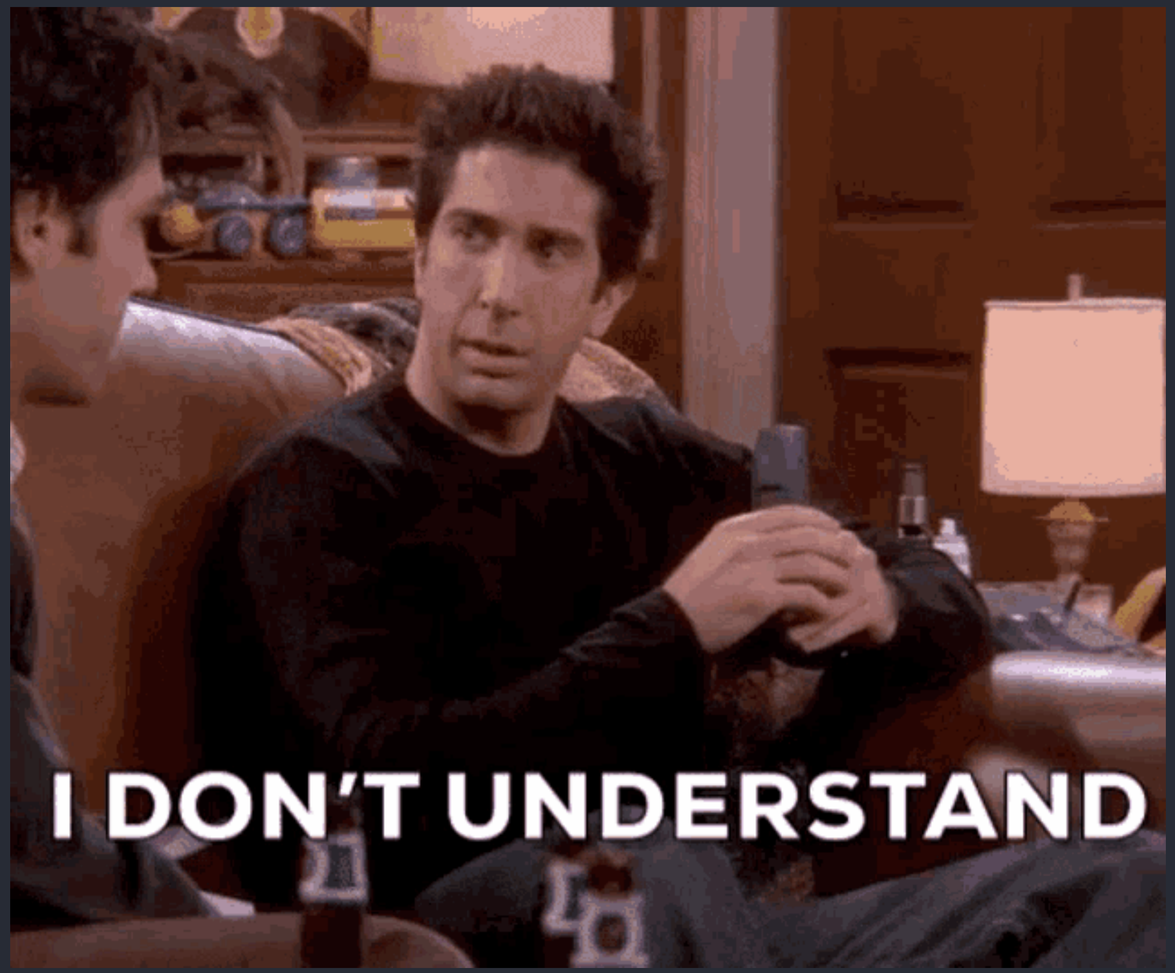

I HAVE NO IDEA
WHAT I'M DOING


I AM A GOD
ATARI
A Warner Communications Company
CX2620


JUST DO IT.

Please,

Do not try to understand everything!



Accept the gaps

— understanding grows with practice

Chef on TV

Real Chef

IxD - Interaction Design

Se det som en smagsprøve på IxD

Client Server

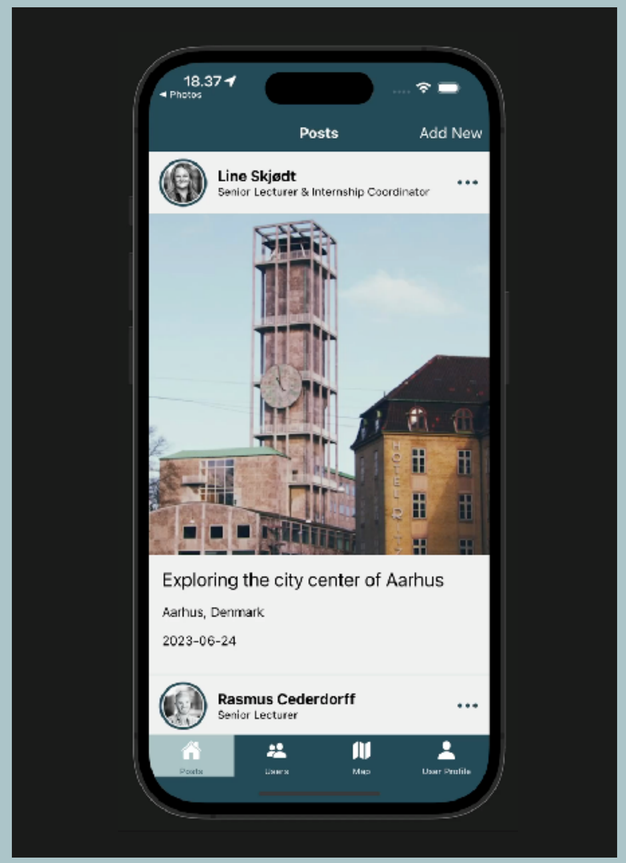
The Basic Architecture of the Web

Client Server Architecture

APP

REST API

DATA SOURCE



FETCH(GET, POST, PUT, DELETE)

REQUEST

JSON DATA

RESPONSE

JSON

posts

users

CLIENT

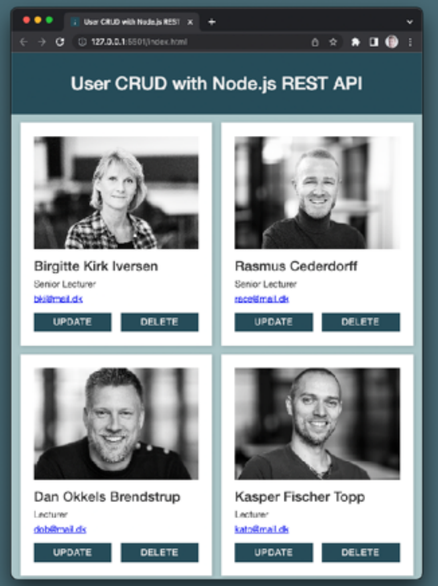
SERVER

Web Development

WEB APP

REST API

DATA SOURCE



cederdorff.com/dat-js-crud-intro

FETCH(GET, POST, PUT, DELETE)

REQUEST

JSON DATA

RESPONSE

posts

users

CLIENT

SERVER

Webudvikling

FRONTEND

REST API

BACKEND

FETCH(GET, POST, PUT, DELETE)

REQUEST

JSON DATA

RESPONSE

Express

Client-Server Model

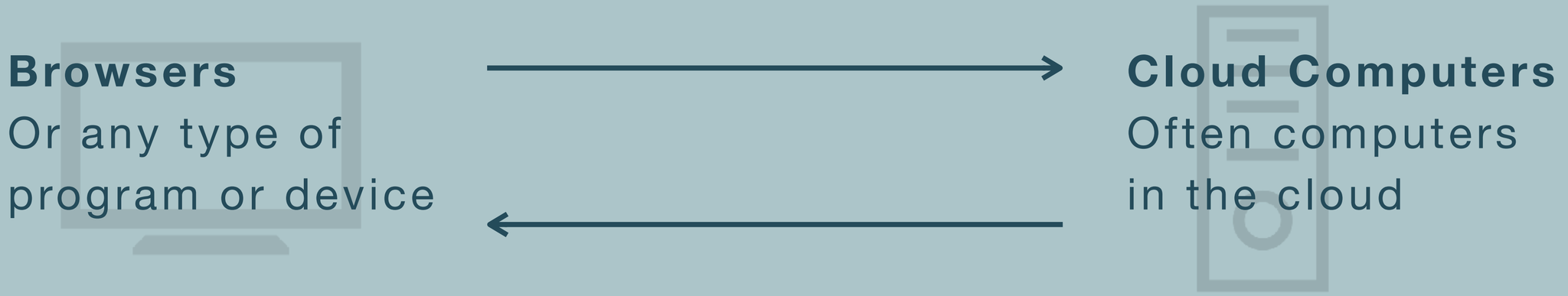
Communication between web **clients** and web **servers**.

CLIENT

SERVER

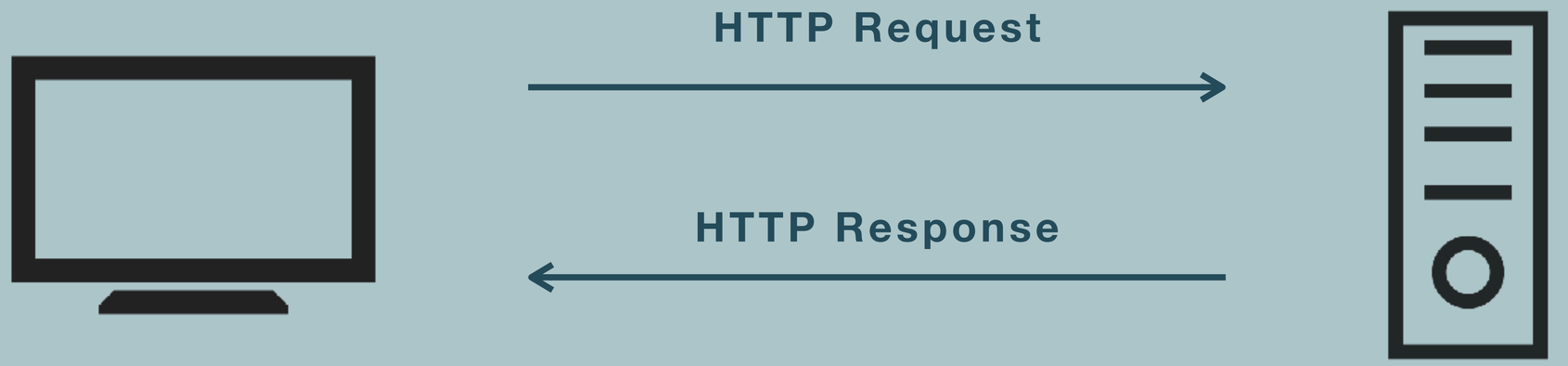
Client-Server Model

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



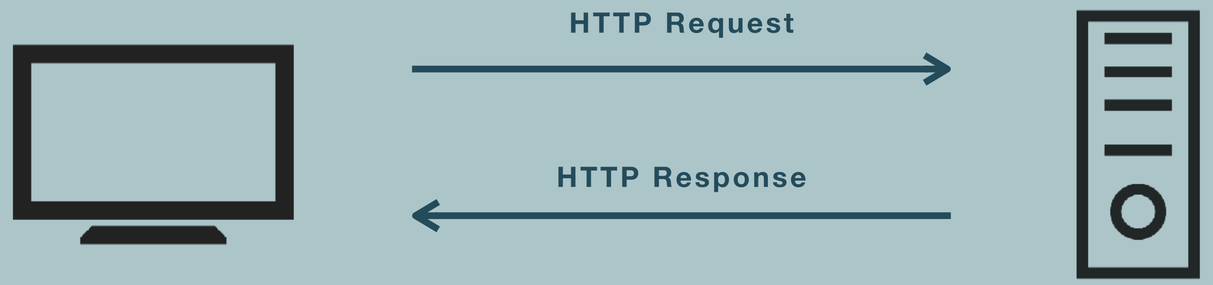
Client-Server Model

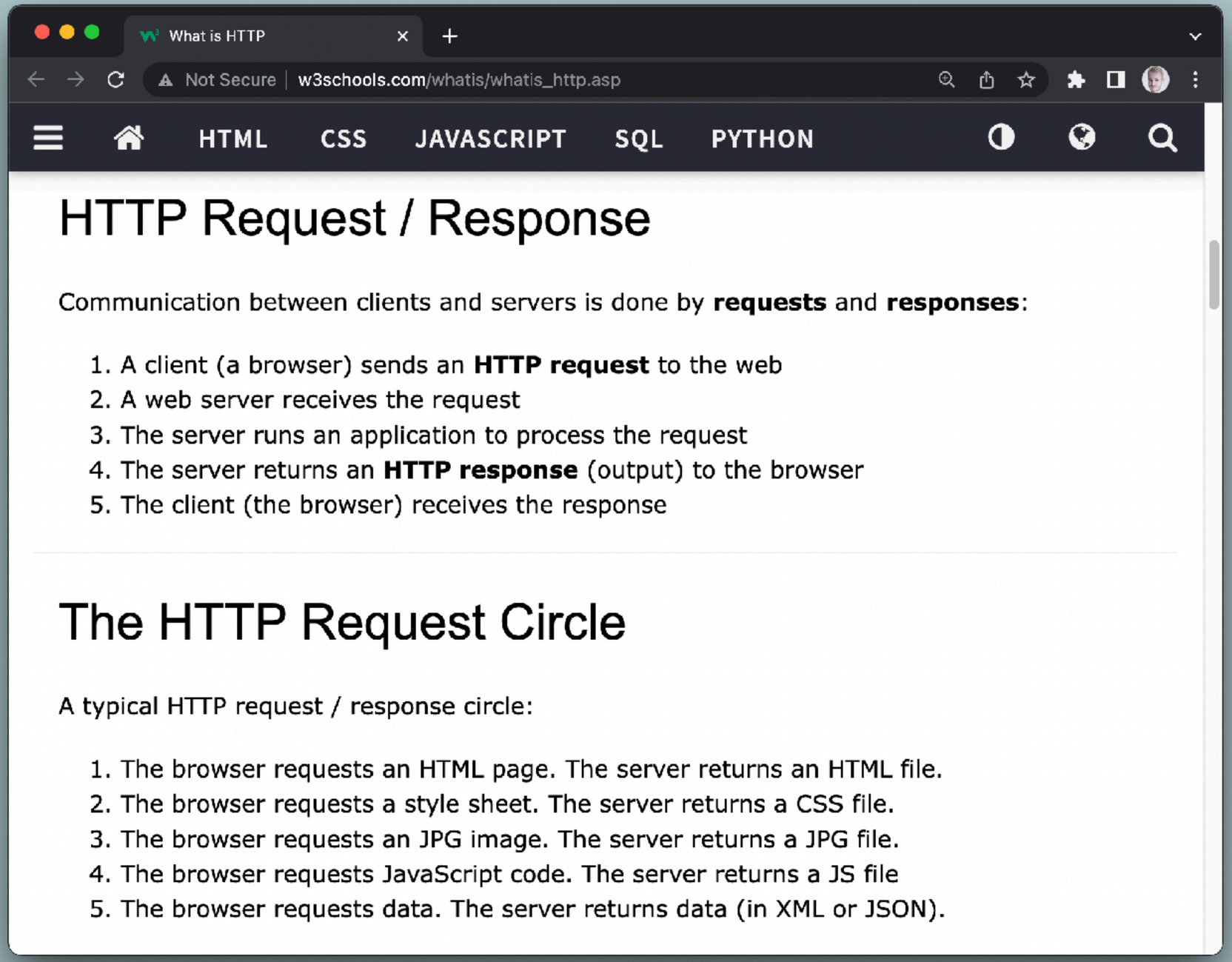
Communication between web **clients** and web **servers**.



Hyper Text Transfer Protocol

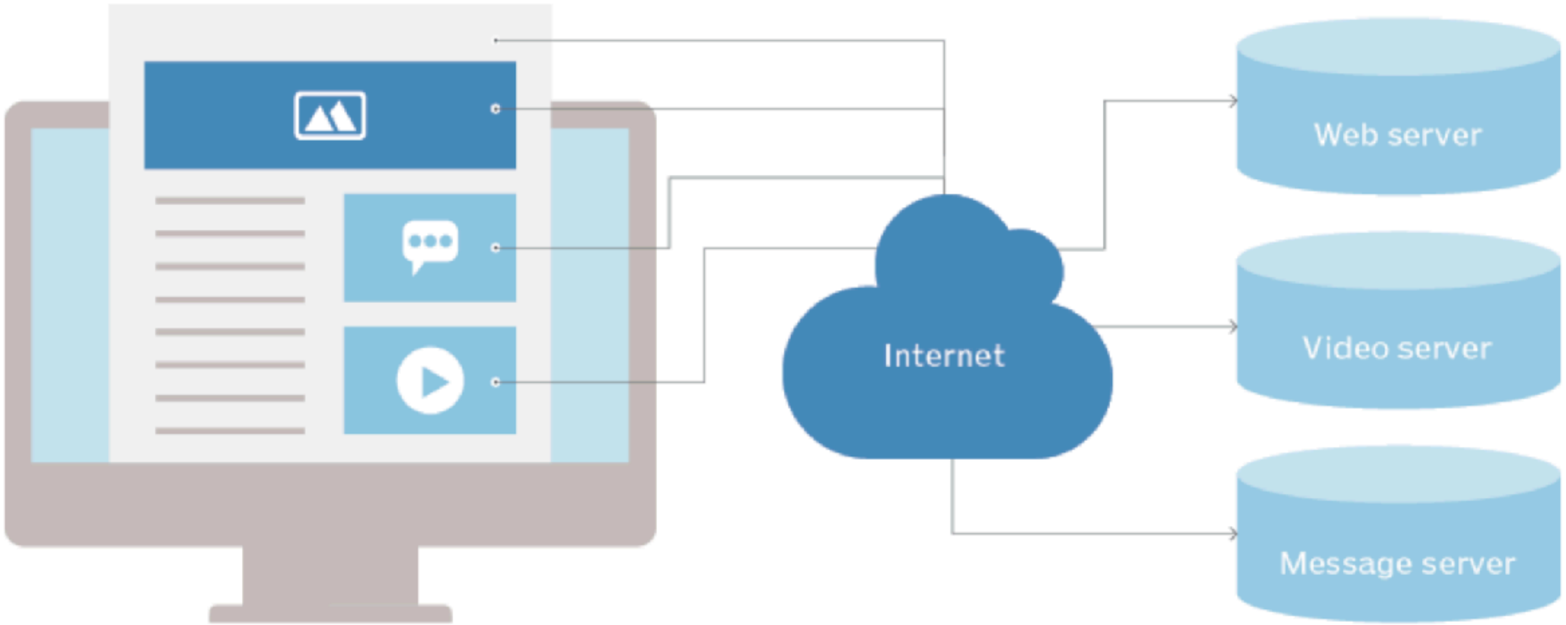
- A protocol and standard for fetching data, HTML and other resources (text, images, videos, scripts, JSON).
- The foundation of the web.





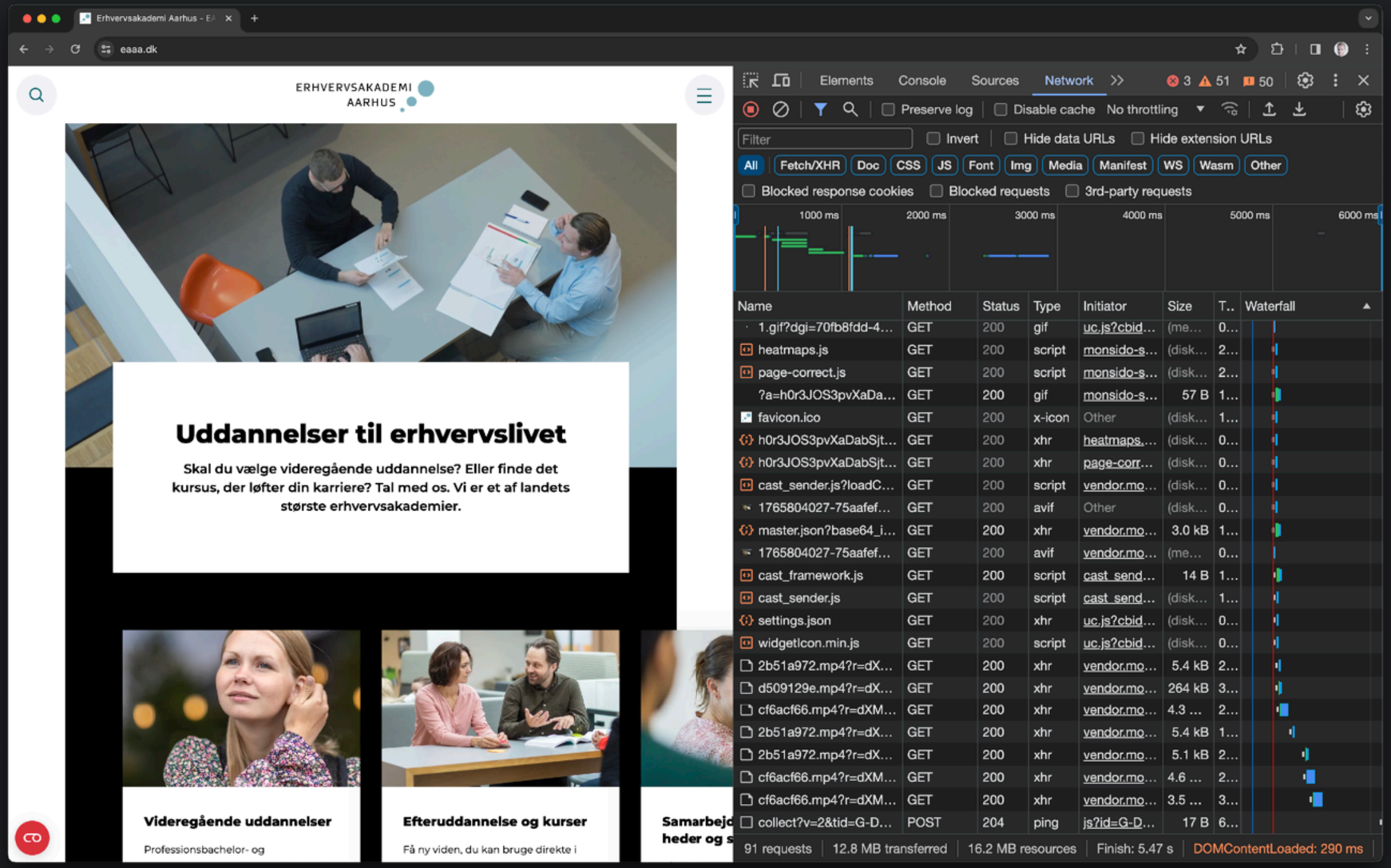
https://www.w3schools.com/whatis/whatis_http.asp

How HTTP Works

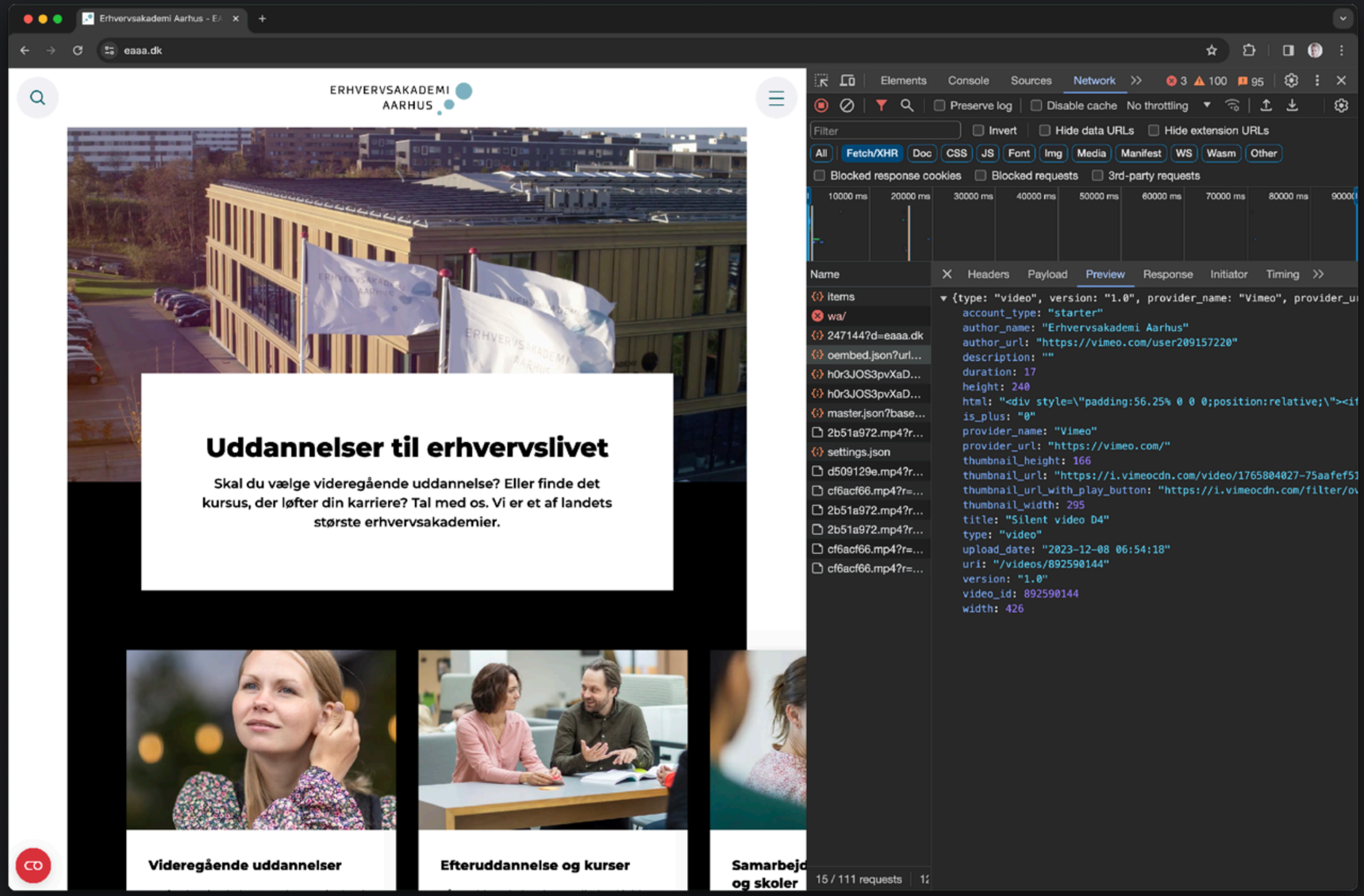


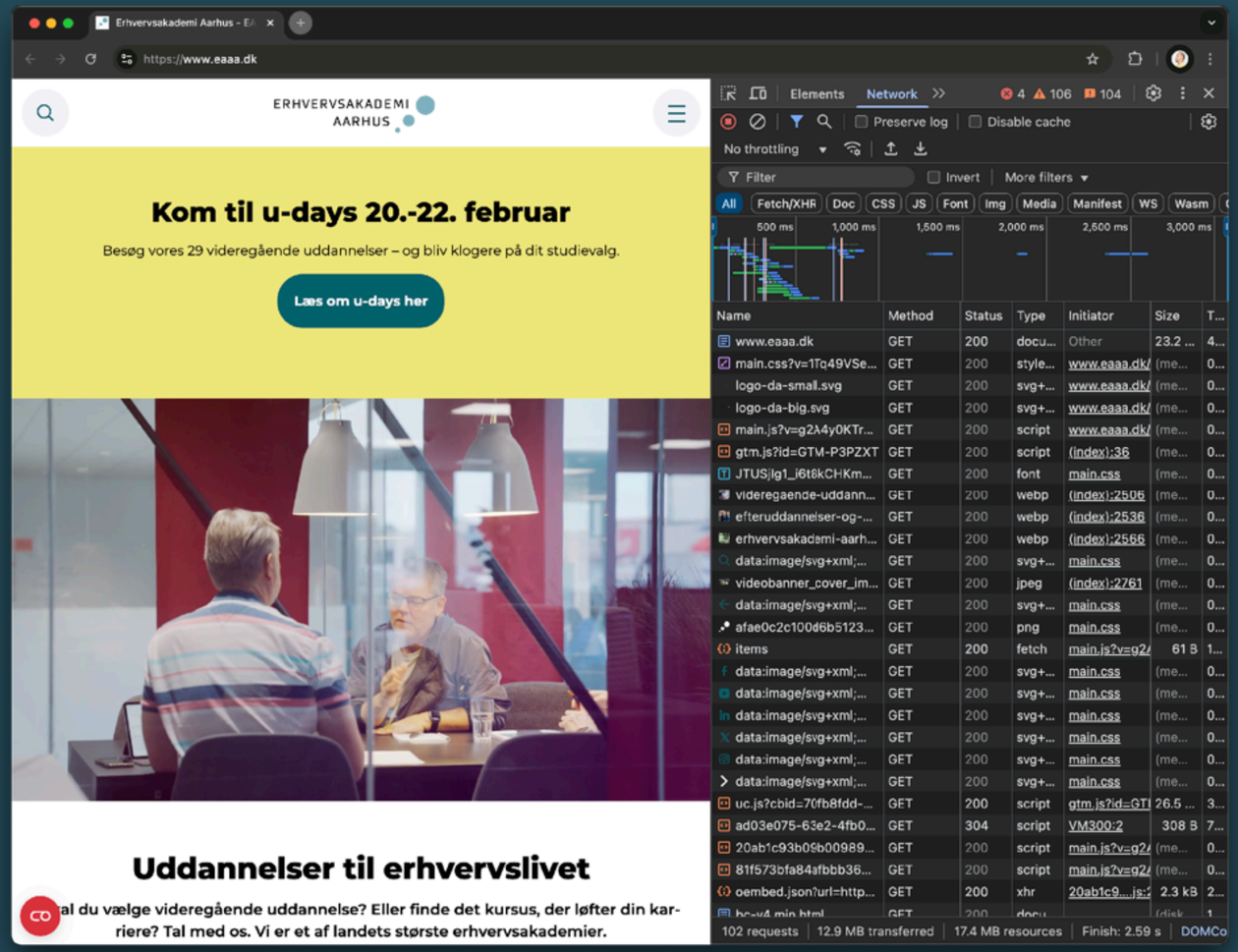
https://www.techtarget.com/whatis/definition/HTTP-Hypertext-Transfer-Protocol

Network Tab


Erhvervsakademi Aarhus - EA
eaaa.dk
ERHVERVSAKADEMI AARHUS
Uddannelser til erhvervslivet
Skal du vælge videregående uddannelse? Eller finde det kursus, der løfter din karriere? Tal med os. Vi er et af landets største erhvervsakademier.
Videregående uddannelser
Professionsbachelor- og
Efteruddannelse og kurser
Få ny viden, du kan bruge direkte i
Elements
Console
Sources
Network
3 51 50
Preserve log
Disable cache
No throttling
Filter
Invert
Hide data URLs
Hide extension URLs
All
Fetch/XHR
Doc
CSS
JS
Font
Img
Media
Manifest
WS
Wasm
Other
Blocked response cookies
Blocked requests
3rd-party requests
1000 ms
2000 ms
3000 ms
4000 ms
5000 ms
6000 ms
Name
Method
Status
Type
Initiator
Size
T..
Waterfall
1.gif?dgi=70fb8fdd-4... GET 200 gif uc.js?cbid... (me... 0...
heatmaps.js GET 200 script monsido-s... (disk... 2...
page-correct.js GET 200 script monsido-s... (disk... 2...
?a=h0r3JOS3pvXaDa... GET 200 gif monsido-s... 57 B 1...
favicon.ico GET 200 x-icon Other (disk... 1...
h0r3JOS3pvXaDabSjt... GET 200 xhr heatmaps,... (disk... 0...
h0r3JOS3pvXaDabSjt... GET 200 xhr page-corr... (disk... 0...
cast_sender.js?loadC... GET 200 script vendor.mo... (disk... 0...
1765804027-75aafef... GET 200 avif Other (disk... 0...
master.json?base64_i... GET 200 xhr vendor.mo... 3.0 kB 1...
1765804027-75aafef... GET 200 avif vendor.mo... (me... 0...
cast_framework.js GET 200 script cast_send... 14 B 1...
cast_sender.js GET 200 script cast_send... (disk... 1...
settings.json GET 200 xhr uc.js?cbid... (disk... 0...
widgetIcon.min.js GET 200 script uc.js?cbid... (disk... 0...
2b51a972.mp4?r=dX... GET 200 xhr vendor.mo... 5.4 kB 2...
d509129e.mp4?r=dX... GET 200 xhr vendor.mo... 264 kB 3...
cf6acf66.mp4?r=dXM... GET 200 xhr vendor.mo... 4.3 ... 2...
2b51a972.mp4?r=dX... GET 200 xhr vendor.mo... 5.4 kB 1...
2b51a972.mp4?r=dX... GET 200 xhr vendor.mo... 5.1 kB 2...
cf6acf66.mp4?r=dXM... GET 200 xhr vendor.mo... 4.6 ... 2...
cf6acf66.mp4?r=dXM... GET 200 xhr vendor.mo... 3.5 ... 3...
collect?v=2&tid=G-D... POST 204 ping js?id=G-D... 17 B 6...
91 requests
12.8 MB transferred
16.2 MB resources
Finish: 5.47 s
DOMContentLoaded: 290 ms

Network Tab


Erhvervsakademi Aarhus - EA
eaaa.dk
ERHVERVSAKADEMI AARHUS
Uddannelser til erhvervslivet
Skal du vælge videregående uddannelse? Eller finde det kursus, der løfter din karriere? Tal med os. Vi er et af landets største erhvervsakademier.
Videregående uddannelser
Efteruddannelse og kurser
Elements
Console
Sources
Network
Preserve log
Disable cache
No throttling
Filter
Invert
Hide data URLs
Hide extension URLs
All
Fetch/XHR
Doc
CSS
JS
Font
Img
Media
Manifest
WS
Wasm
Other
Blocked response cookies
Blocked requests
3rd-party requests
10000 ms
20000 ms
30000 ms
40000 ms
50000 ms
60000 ms
70000 ms
80000 ms
Name
Headers
Payload
Preview
Response
Initiator
Timing
items
wa/
247144?d=eaaa.dk
oembed.json?url...
h0r3JOS3pvXaD...
h0r3JOS3pvXaD...
master.json?base...
2b51a972.mp4?r...
settings.json
d509129e.mp4?r...
cf6acf66.mp4?r=...
2b51a972.mp4?r...
2b51a972.mp4?r...
cf6acf66.mp4?r=...
cf6acf66.mp4?r=...
{type: "video", version: "1.0", provider_name: "Vimeo", provider_u
account_type: "starter"
author_name: "Erhvervsakademi Aarhus"
author_url: "https://vimeo.com/user20915722O"
description: ""
duration: 17
height: 240
html: "<div style=\"padding:56.25% 0 0 0;position:relative;\"><i
is_plus: "0"
provider_name: "Vimeo"
provider_url: "https://vimeo.com/"
thumbnail_height: 166
thumbnail_url: "https://i.vimeocdn.com/video/1765804027-75aafef51
thumbnail_url_with_play_button: "https://i.vimeocdn.com/filter/ov
thumbnail_width: 295
title: "Silent video D4"
type: "video"
upload_date: "2023-12-08 06:54:18"
uri: "/videos/892590144"
version: "1.0"
video_id: 892590144
width: 426
15 / 111 requests



Network Tab

Investigating Network Activity in the Browser

Variabler

`const` & `let`

Variables

... are used to store data (values, objects, collections) in the memory

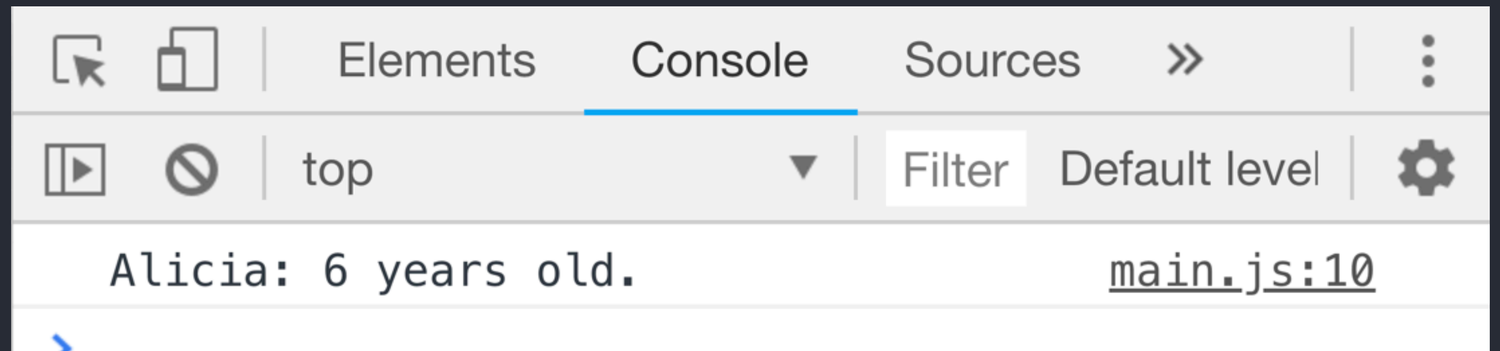
Variables

... a simple way to store, get and set data in your code.

Variables

Store data in the memory

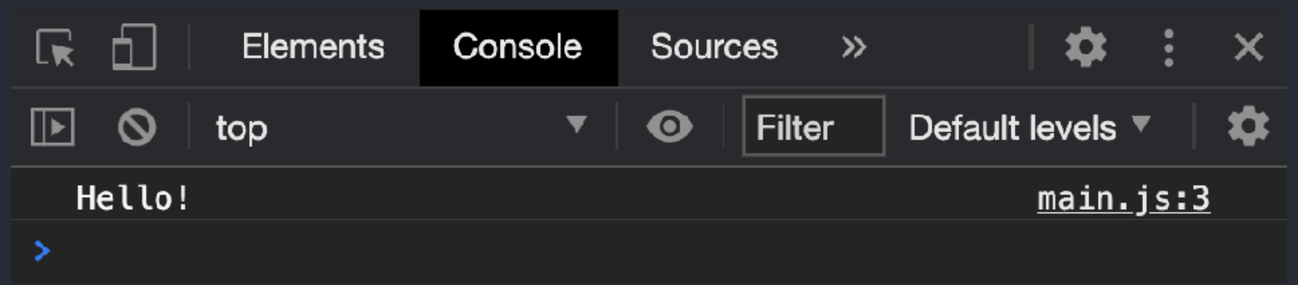
```
let name = "Alicia";
let age = 6;

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



Variables

Store data in the memory

```
let message = "Hello!";

console.log(message);
```



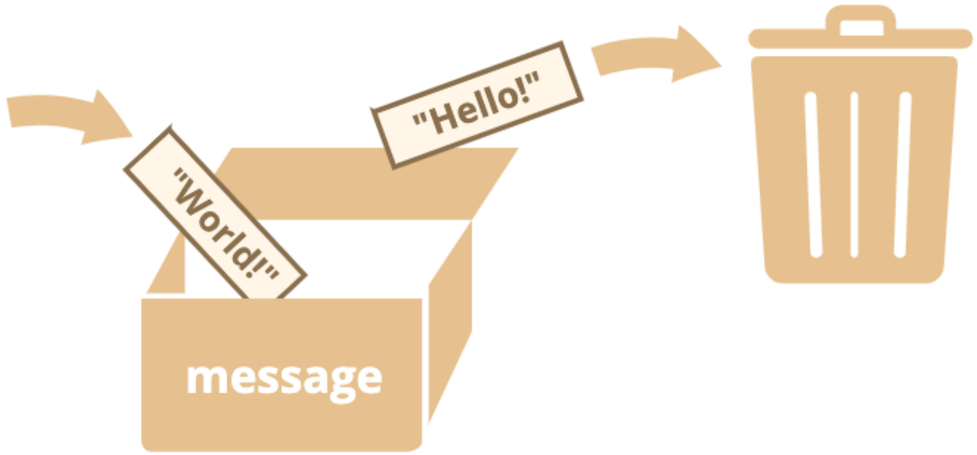
Variables

Store data in the memory

```
let message = "Hello!";
console.log(message);

message = "World!";
console.log(message);
```

Variable

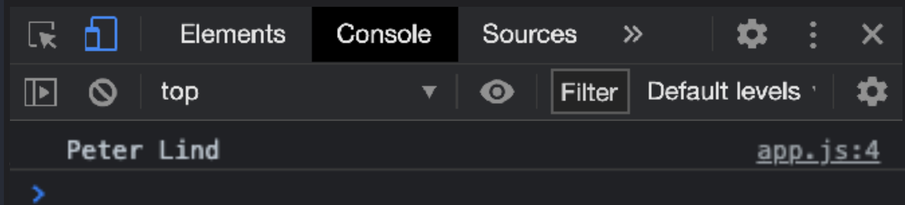
A variable is a “named storage” and stored in the memory of the browser.



We can change the value of the variables as many times as we want.

Variables & UI

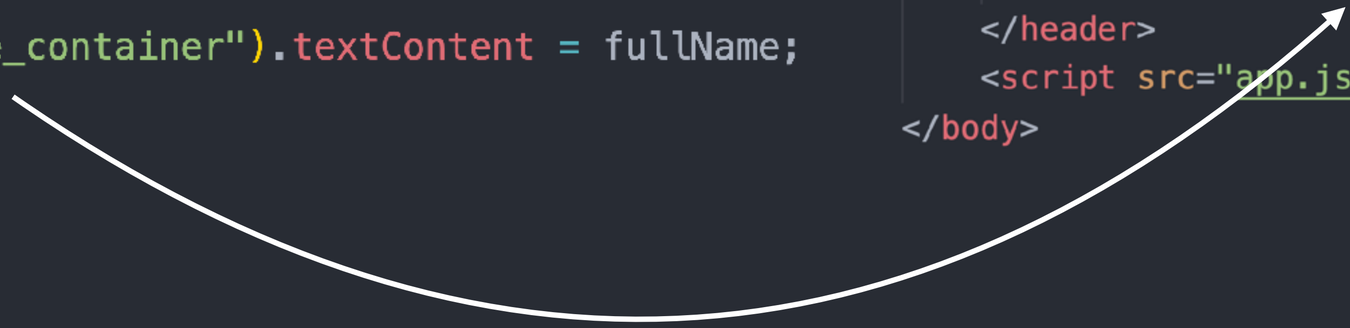
```
let fullName = "Peter Lind";
console.log(fullName);
document.querySelector("#fullName_container").textContent = fullName;
```



Variables & UI

```
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>
```



Variables & UI

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

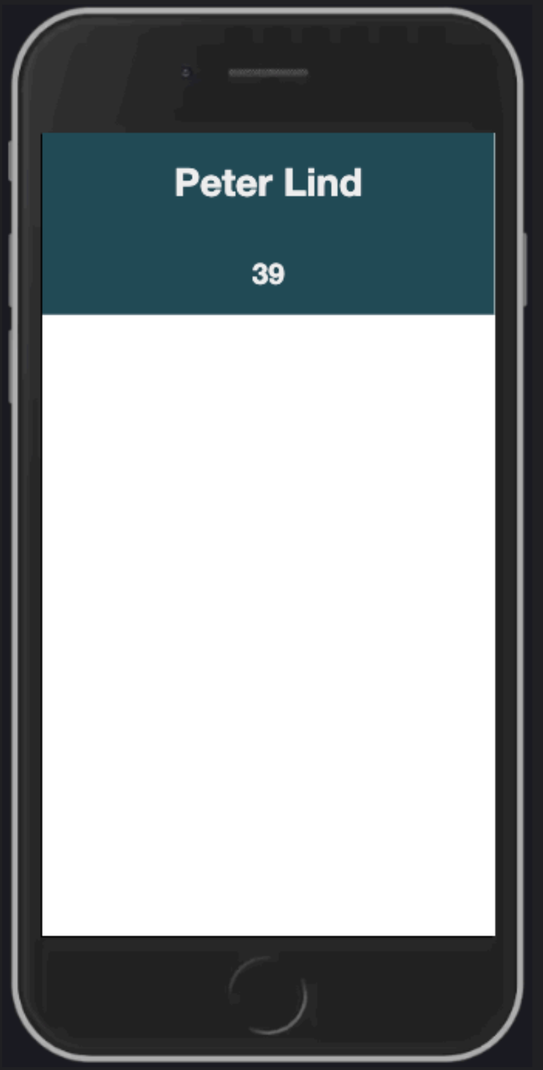
.textContent

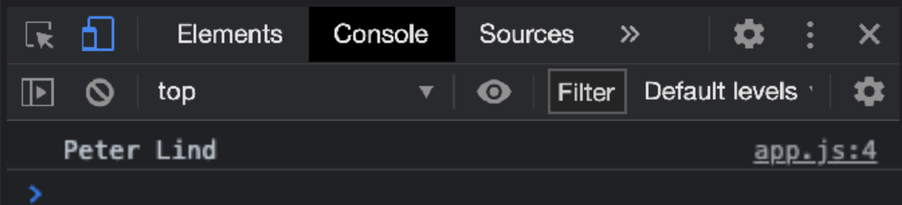
Get or set the text content of a given (HTML) element

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

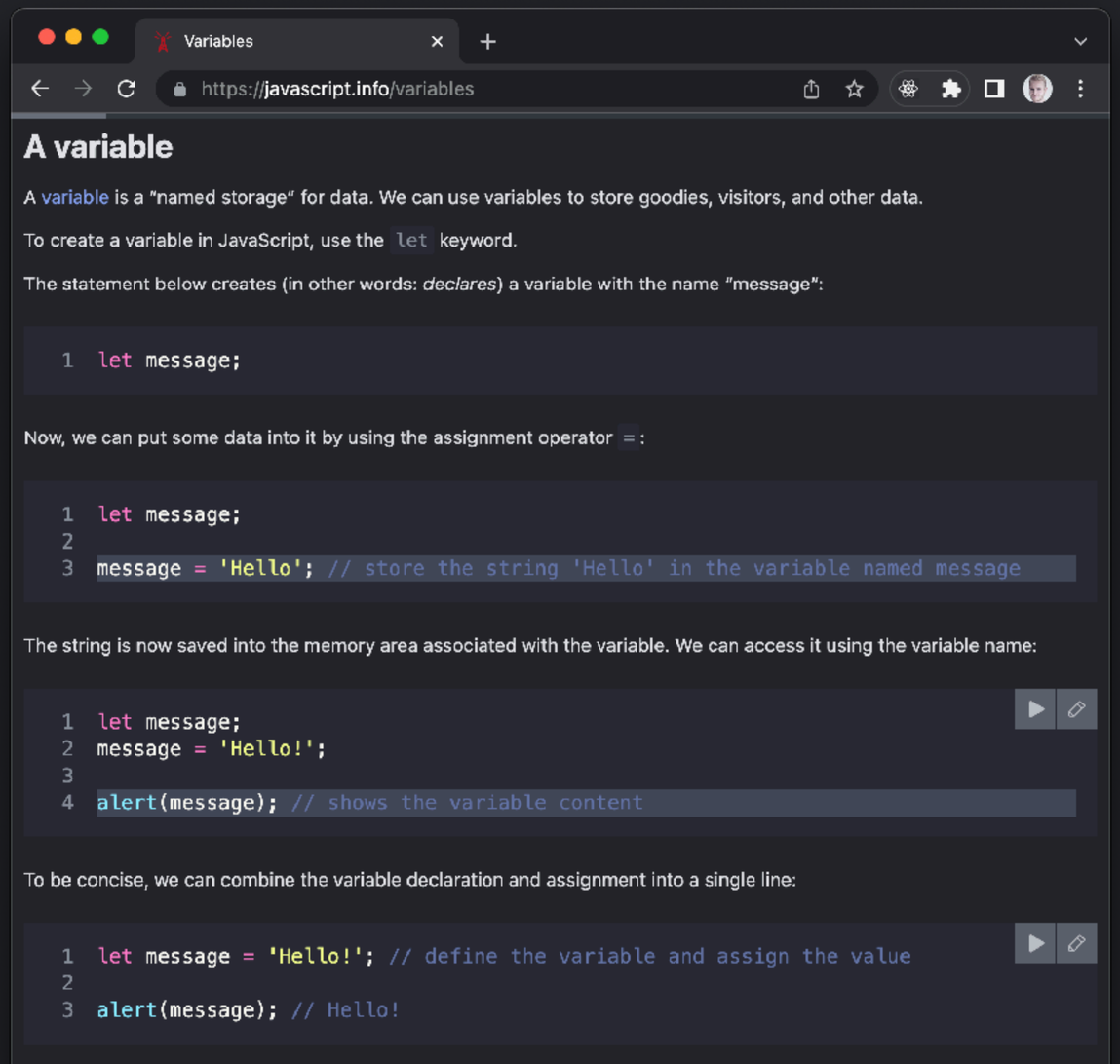
Variables & UI

```
let fullName = "Peter Lind";
let age = 39;
console.log(fullName, age);
document.querySelector("#fullName_container").textContent = fullName;
document.querySelector("#age_container").textContent = age;
```





JavaScript.info/Variables



Read, read,
read,
The docs 🤷

var **vs** let

THE DIFFERENCE IS THE SCOPING

VAR IS FUNCTION-WIDE OR GLOBAL SCOPE

LET IS BLOCK SCOPED

VAR TOLERATES REDECLARATION

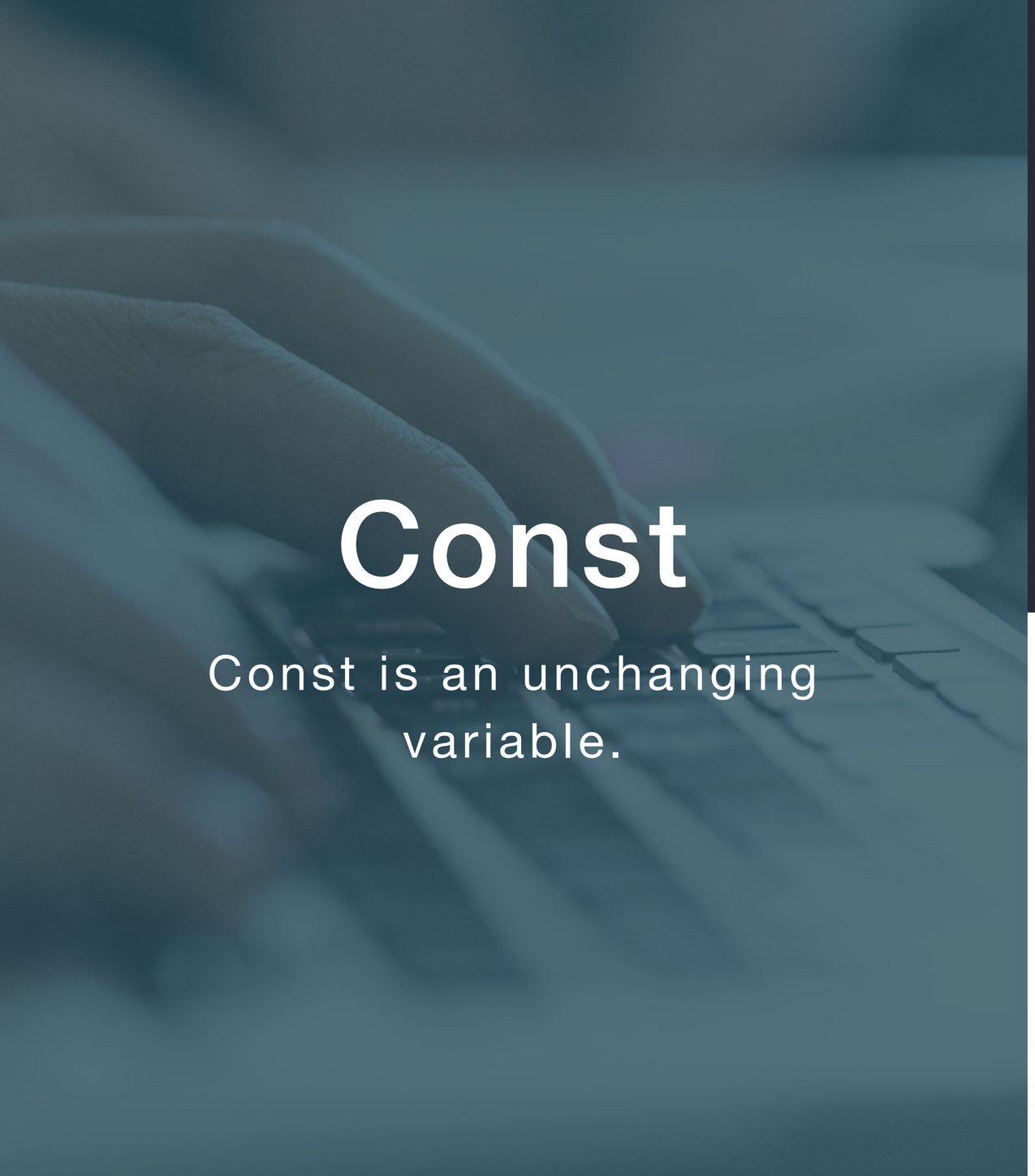
https://javascript.info/variables
https://javascript.info/var

```
// Example 1
// “var” has no block scope
if (true) {
  var test1 = true; // use "var" instead of "let"
}
console.log(test1); // true, the variable lives after if

// Example 2
if (true) {
  let test2 = true; // use "let"
}
console.log(test2); // Error: test is not defined

// Example 3
for (var i = 0; i < 10; i++) {
  // ...
}
console.log(i); // 10, "i" is visible after loop, it's a global variable
```

var-vs-let

```
// “var” tolerates redeclarations
var user1 = "Pete";
var user1 = "John"; // this "var" does nothing (already declared)
// ...it doesn't trigger an error
console.log(user1); // John

let user2;
let user2; // SyntaxError: 'user' has already been declared
```

var-vs-let



Const

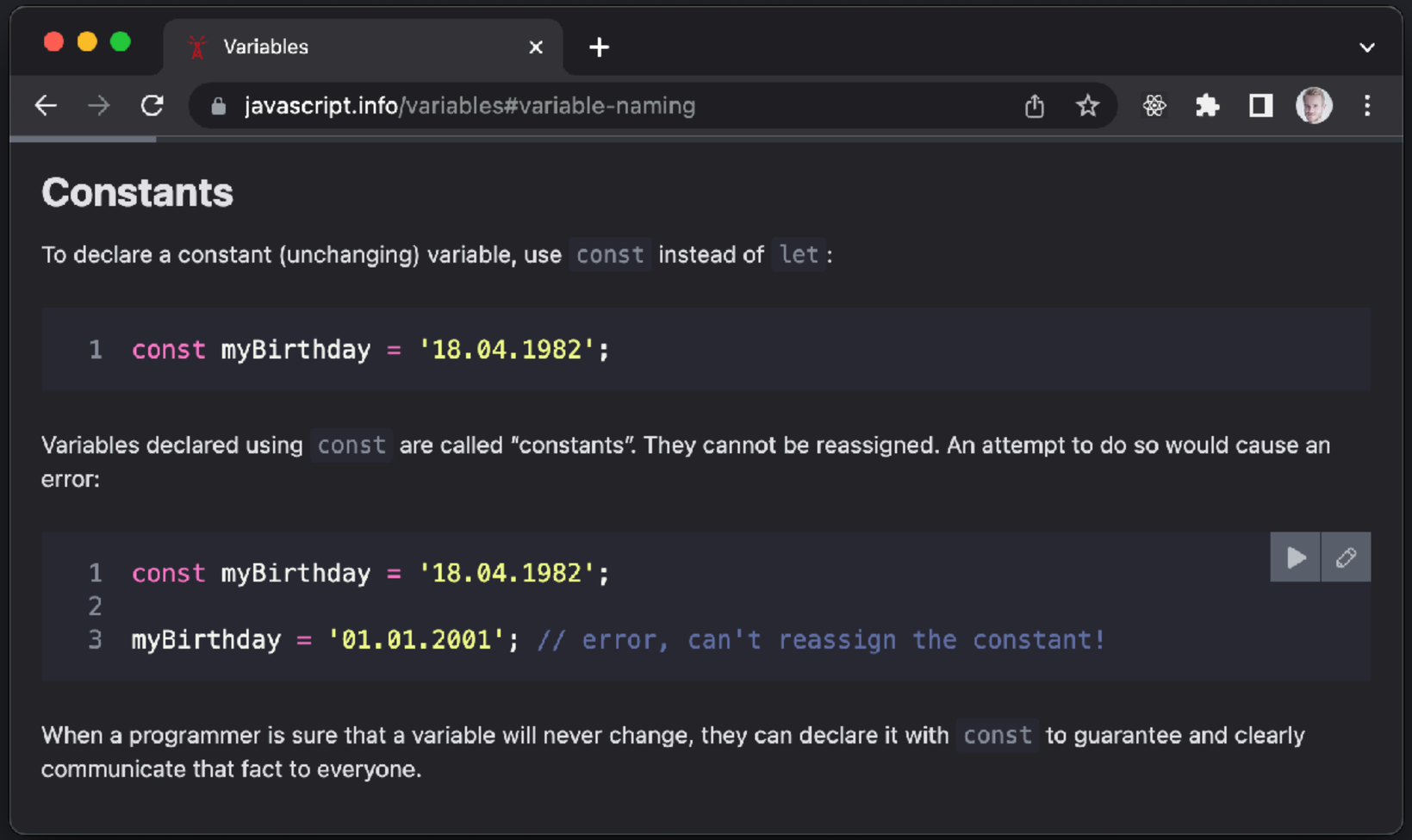
Const is an unchanging variable.

```
const myBirthday = "12-03-1990";
myBirthday = "12-03-1989";
// Uncaught TypeError: can't reassign the constant!
```

`const` cannot be reassigned.

If you try to, an `error` will be thrown.

And the doc says...



Const can’t be reassigned

```
const myBirthday = "12-03-1990";
myBirthday = "12-03-1989"; // Uncaught TypeError: can't reassign the constant!

const person = {
    name: "Kasper",
    mail: "kato@eaaa.dk",
    age: 32
};

person.age = 33; // no error

person = {
    name: "Rasmus",
    mail: "race@eaaa.dk",
    age: 31
}; // Uncaught TypeError: can't reassign the constant!
```

Use `let` & `const` instead of `var`

https://javascript.info/variables
https://javascript.info/var

But which one? 🤷

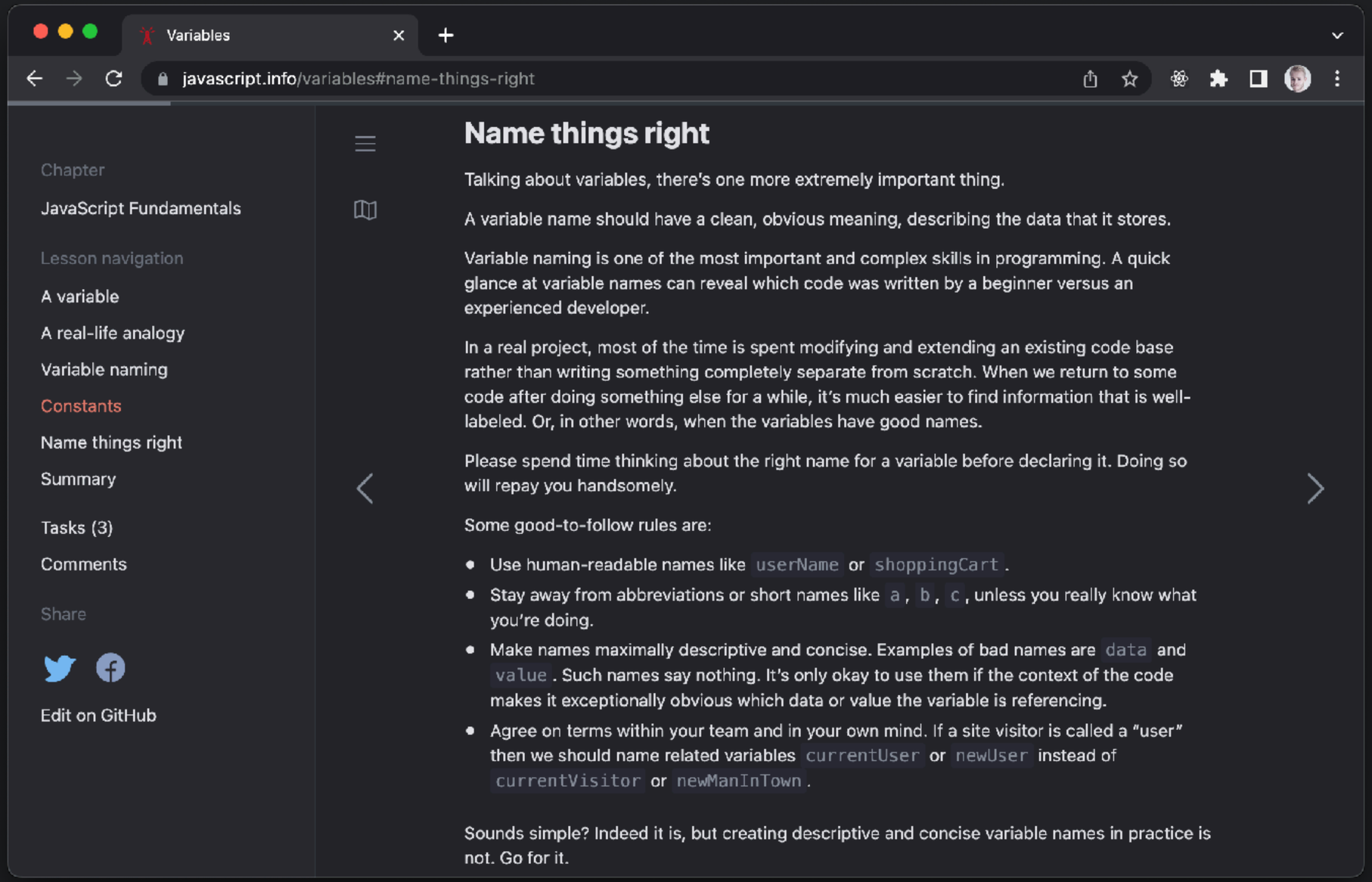
https://javascript.info/variables
https://javascript.info/var

Start with `const`

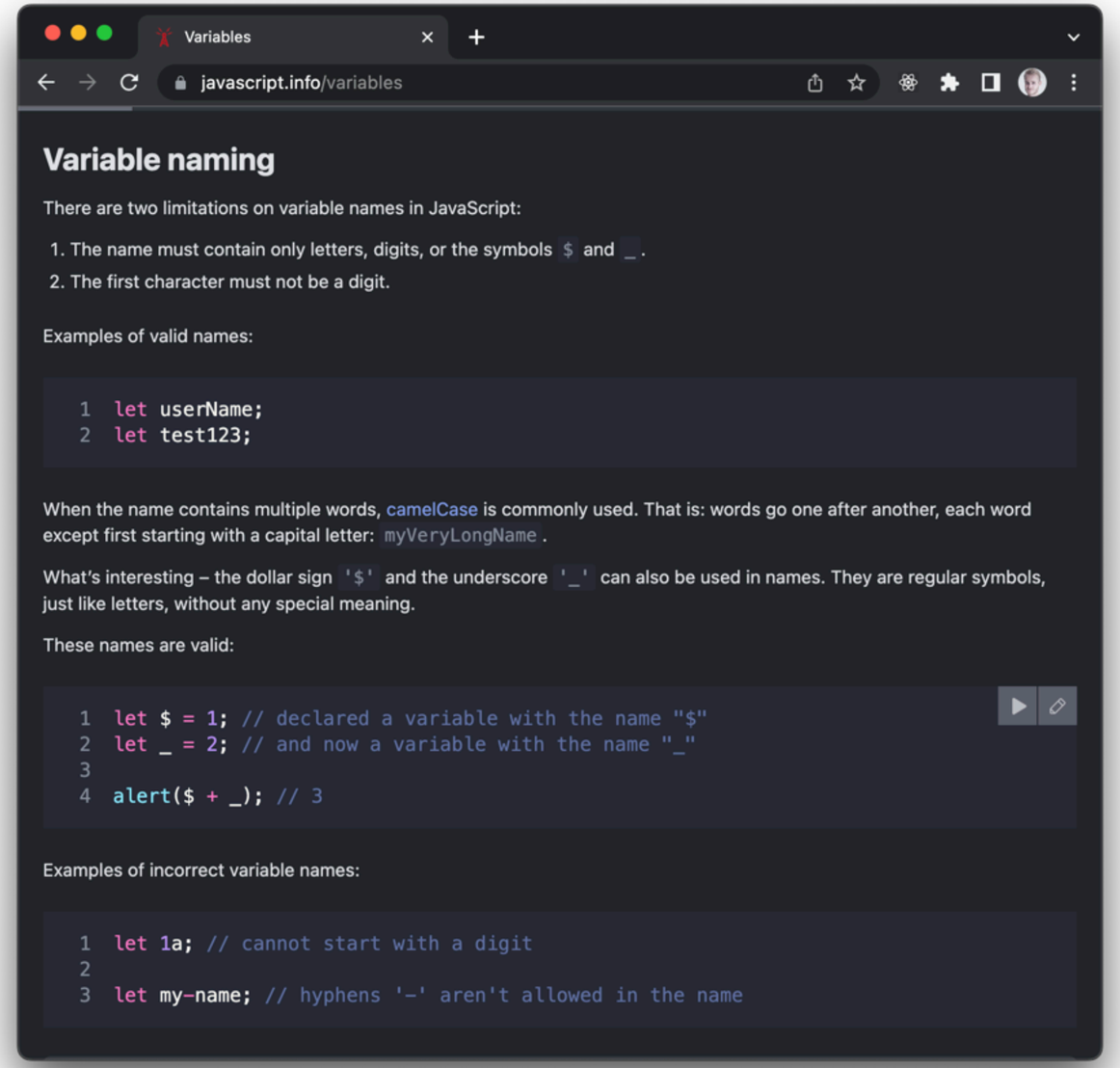
If needed, change to `let`

Never use `var`!

https://javascript.info/variables
https://javascript.info/var



https://javascript.info/variables#name-things-right



https://javascript.info/variables#variable-naming

Name things right (or wrong?)

1. Create a variable with the name of our planet. How would you name such a variable?
2. Create a variable to store the name of a current visitor to a website. How would you name that variable?

https://javascript.info/variables

Global Variables

Variables outside a function (and scopes) are global variables

JavaScript.info/function-basics#local-variables

Scopes:

- Local Variable
- Global Variable

Local variables

A variable declared inside a function is only visible inside that function.

For example:

```
1  function showMessage() {
2    let message = "Hello, I'm JavaScript!"; // local variable
3
4    alert( message );
5  }
6
7  showMessage(); // Hello, I'm JavaScript!
8
9  alert( message ); // <-- Error! The variable is local to the function
```

Outer variables

A function can access an outer variable as well, for example:

```
1  let userName = 'John';
2
3  function showMessage() {
4    let message = 'Hello, ' + userName;
5    alert(message);
6  }
7
8  showMessage(); // Hello, John
```

Global variables

Variables declared outside of any function, such as the outer `userName` in the code above, are called *global*.

Global variables are visible from any function (unless shadowed by locals).

It's a good practice to minimize the use of global variables. Modern code has few or no globals. Most variables reside in their functions. Sometimes though, they can be useful to store project-level data.

https://javascript.info/function-basics#outer-variables

Datatyper

Application Programming Interface

Data Types

In JavaScript there are two main data types:

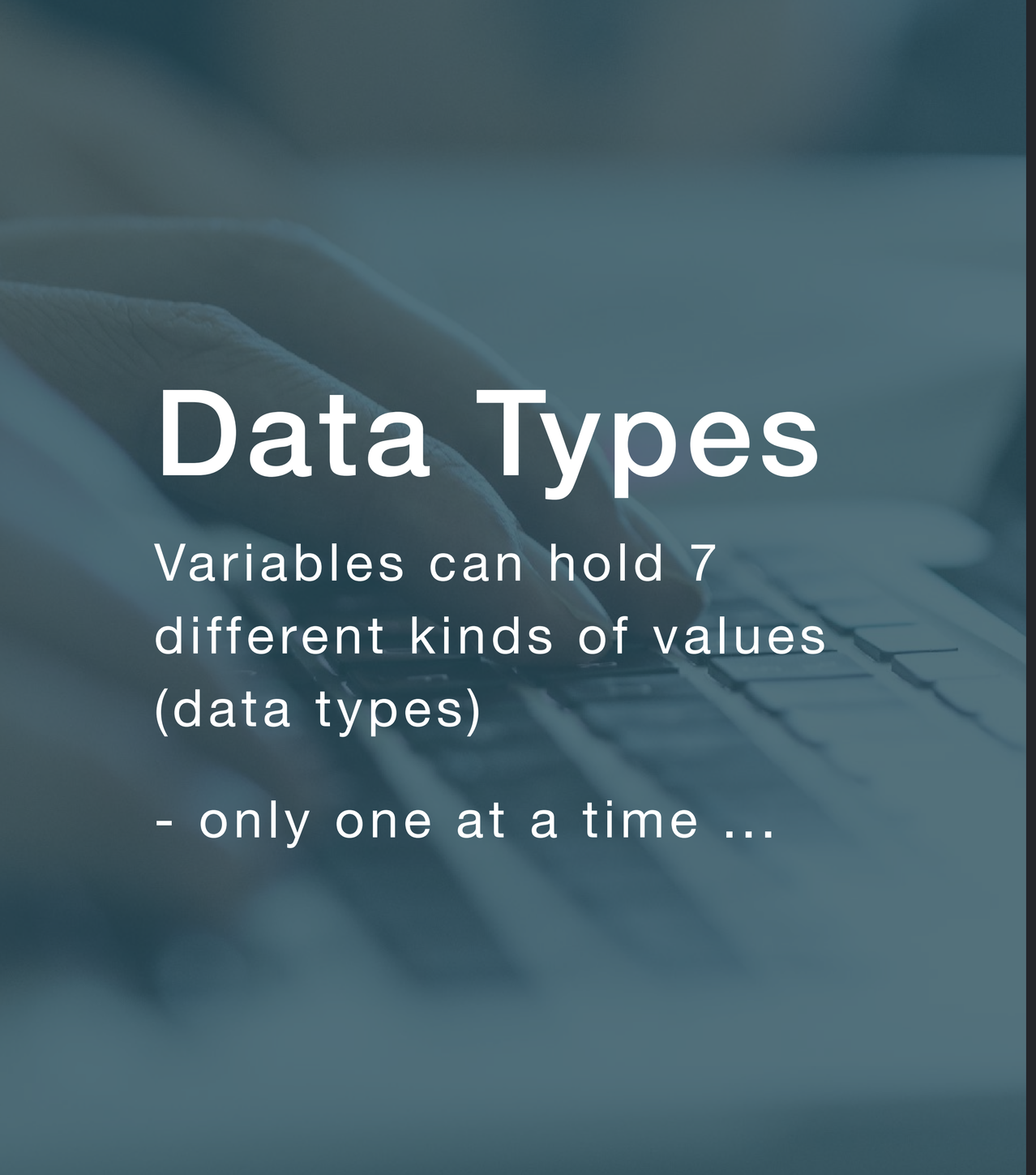
- **Primitive values** like strings, numbers and booleans.
- **Objects** with properties.

```
let str = "Hello";
let str2 = 'Single quotes are ok too';
let phrase = `can embed another ${str}`;
```

```
let n = 123;
n = 12.345;
```

```
let user = new Object(); // "object cons
let user = {};  // "object literal" synt
```

In JavaScript, a value always has a certain type like a string, number, boolean, object, array, etc.



Data Types

Variables can hold 7 different kinds of values (data types)

- only one at a time ...

Boolean

Number

String

Object

Null

Undefined

Symbol

Datatypes - when to use which one?

Boolean	Values that are true or false – conditions
Number	Numbers! Points, counting, **math-operations**, measuring …
String	Text! Input-fields – HTML and CSS-attributes … URLs (Phone-number)
Object	Everything (Arrays, Functions) Combined/collected data (of the other types)
Null	No datatype yet – no value!
Undefined	Something that hasn't defined before – don't set things to undefined!
Symbol	Something complicated

Datatyper

1. Prøv disse og test med `console.log`
2. Hvordan ser vi værdien?
3. Hvordan ser vi typen?
4. Hvordan definerer du, hvilken type, du vil gemme i en variabel?

```
const bool = true;
const num = 41;
const str = "Peter";
const obj = {
  cats: 2,
  cars: 1
};
const nothing = null;
let undf;
const symbol = Symbol("symbol");
```

https://javascript.info/variables

Strings

In JavaScript textual data is stored as strings.

Strings are defined with either ‘single quotes’, “double quotes” or `backticks`.

```
let single = 'single-quoted';

let double = "double-quoted";

let backticks = `backticks`;
```

```
let fullName = "Peter Lind";
let age = 39;

let message = fullName + " is " + age + " years old.";
console.log(message); // Peter Lind is 39 years old.
```

Concatenation

We can combine two or more variables in one string (variable).

```
let fullName = "Peter Lind";
let age = 39;

// single
console.log(fullName + ' is ' + age + ' years old.');
// double
console.log(fullName + " is " + age + " years old.");
// backticks
console.log(`${fullName} is ${age} years old.`);
```

Concatenation

Strings are defined with either ‘single quotes’, “double quotes” or `backticks`.

String quotes

1. Test the code to the right.
2. What is the output of the script to the right?

```
"use strict";

let fullName = "Alicia Keys";

console.log(`hello ${1}`); // ?

console.log(`hello ${"name"}`); // ?

console.log(`hello ${fullName}`); // ?
```

Data Type Conversion

In JavaScript, we never specify the type of a variable (String, Number, Boolean, Object, etc.).

We set the value and use the value as the type we expect it to be.

```
const firstName = "Rasmus";
const lastName = "Cederdorff";

let age = 32;

let isSeniorLecture = true;

const obj = {
    cats: 0,
    cars: 1
};

const kids = ["Alicia", "Ida"];
```

It happens when we do:

- Concatenation (+)
- Comparisons (==, !=, <, >, <=, >=)
- Calculations (-, *, /, %)

Automatic conversion

So often JavaScript has to do automatic conversion.

Everything in HTML is strings, so when writing or reading numbers, they have to be converted.

Most of the time this happens automatically.

“Semi-automatic” conversion

`String(`*`value`*`)` converts a value to a string, usually by calling *value*`.toString()`

`Number(`*`value`*`)` converts a value to a number, usually by calling *value*`.valueOf()`

`Boolean(`*`value`*`)` converts a value to a boolean, by unknown magic means!

Note:
Boolean(0) is false, but
Boolean("0") is true

It is always possible to convert something into a string –
but it might not be possible to convert everything into a number!

Numeric conversion

1. How do you convert to Number?

```
let string = "1234";

console.log(string);

// number??
```

String conversion

1. How do you convert to String?

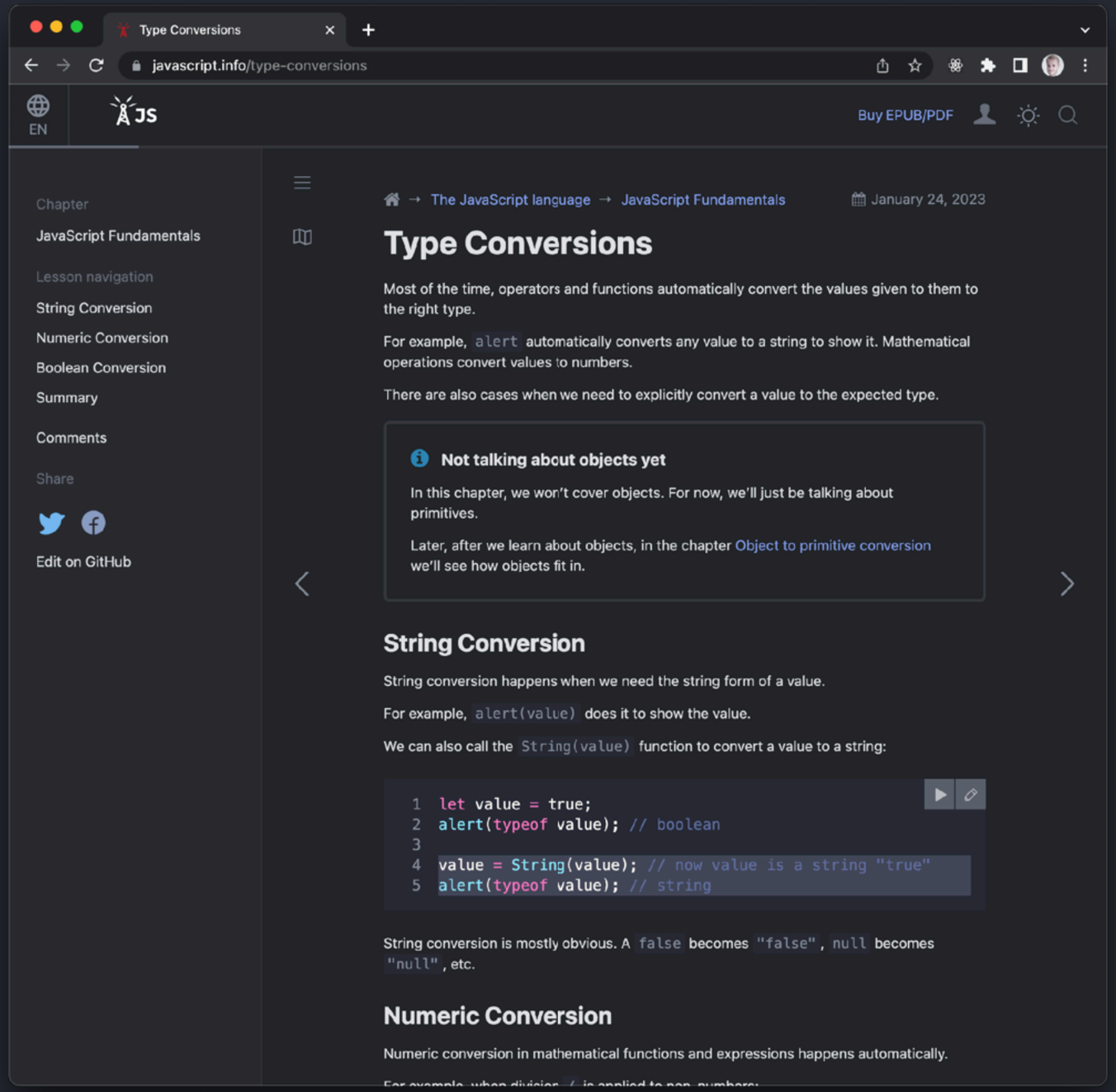
```
let number = 1234;

console.log(number);

// string??
```

Numeric conversion 2

1. Try yourself!
2. What is the output in the console?

```
console.log(Number("   123   ")); // ?
console.log(Number("123z")); // ?
console.log(Number(true)); // ?
console.log(Number(false)); // ?
```

https://javascript.info/strict-mode



Read, read, read, The docs 🤷

Manual conversion

You can ommit the prefix `Number.` if you want

`Number.parseInt(`*`string`*`)`

Converts a string **to** a **number**, using the specified numerical system (MDN)

`Number.prototype.toString()`

Converts a number **to** a **string**, in the specified numerical system (MDN)

There's also: parseFloat, toExponential, toFixed, toPrecision for other kinds of conversion

```
const n1 = 5;
const n2 = 6;

const s1 = "4";
const s2 = "9";

console.log(n1 + n2);
console.log(s1 + s2);

console.log(n1 - n2);
console.log(s1 - s2);

console.log(n1 + s2);
console.log(s1 + n2);

console.log(n1 - s2);
console.log(s1 - n2);
```

Concatenation looks like addition

1. What will be the result of these calculations/ concatenations?

2. Test them in the console!

Automatic data type conversion with concatenation

String + string is a **concatenation**

Number + number is a **calculation**

But if the types are different, one gets converted to the other!

The challenge is to figure out which one gets converted to what ...

```
"" + 1 + 0;
"" - 1 + 0;
true + false;
6 / "3";
"2" * "3";
4 + 5 + "px";
"$" + 4 + 5;
"4" - 2;
"4px" - 2;
"   -9   " + 5;
"   -9   " - 5;
null + 1;
undefined + 1;
" \t \n" - 2;
```

Type Conversion

1. Test these expressions.
2. First: Try to guess the result, then check your guess in the console!
3. Explain to each other why your guess was correct/ wrong.

https://javascript.info/type-conversions

Template String

A template string (template literal) in JavaScript is a string enclosed in backticks (`) that allows embedding variables and expressions using ${ ... } and supports multi-line text.

`template string`

“Template literals are literals delimited with backtick (`) characters, allowing for multi-line strings, for string interpolation with embedded expressions, and for special constructs called tagged templates.”

```
let name = "Alicia";
let age = 6;

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

console.log(`${name} is ${age} years old.`);
```

```
Alicia is 6 years old.          main.js:10
Alicia is 6 years old.          main.js:12
>
```

`template string`

A template string (template literal) in JavaScript is a string enclosed in backticks (`) that allows embedding variables and expressions using ${ ... } and supports multi-line text.

```
const navn = "Rasmus";
const alder = 34;

const tekst = `Hej, jeg hedder ${navn} og jeg er ${alder} år gammel.`;

console.log(tekst);
// Output: Hej, jeg hedder Rasmus og jeg er 34 år gammel.
```

```
const navn = "Rasmus";
const alder = 34;

const tekst = `Hej, jeg hedder ${navn} og jeg er ${alder} år gammel.`;
const tekstUdenTemplate = "Hej, jeg hedder " + navn + " og jeg er " + alder + " år gammel.";

console.log(tekst);
// Output: Hej, jeg hedder Rasmus og jeg er 34 år gammel.
```

`template string`

Backtick String / Template Literals

- Extended functionality
- Simplifies concatenating strings
- Embed values and expression into a string with `${...}`
- Simplifies the syntax and the reading
- Let us create more readable HTML templates

```
let name = "Alicia";
console.log(`Hello, ${name}`);
```

```
Hello, Alicia                                    main.js:8
```

`template string`

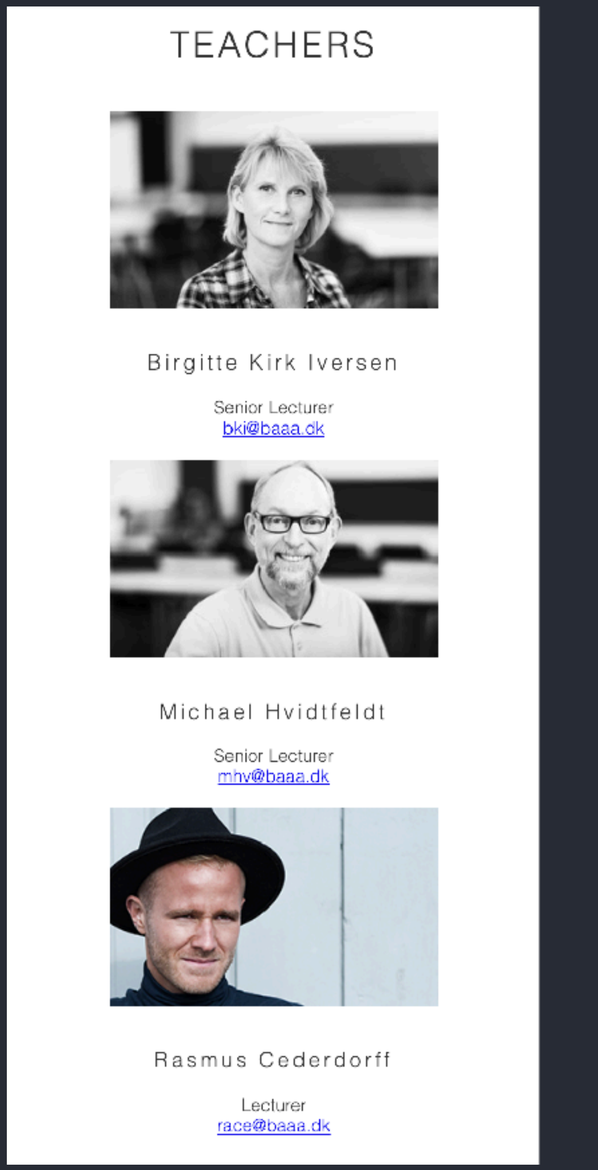
```
let name = "Alicia";
let age = 6;

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

console.log(`${name} is ${age} years old.`);
```

```
Alicia is 6 years old.          main.js:10
Alicia is 6 years old.          main.js:12
>
```

`template string`

REGULAR STRING EXPRESSION

```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML +=
      "<article>" +
      "<img src='" + teacher.img + "'>" +
      "<h3>" + teacher.name + "</h3>" +
      teacher.position + "<br>" +
      "<a href='mailto:" + teacher.mail + "'>" + teacher.mail + "</a>" +
      "</article>";
  }
}
```



`template string`

...EMBED VARIABLES AND EXPRESSIONS IN A STRING

```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML +=
      "<article>" +
      "<img src='" + teacher.img + "'>" +
      "<h3>" + teacher.name + "</h3>" +
      teacher.position + "<br>" +
      "<a href='mailto:" + teacher.mail + "'>" + teacher.mail + "</a>" +
      "</article>";
  }
}
```

```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML += `
      <article>
        <img src='${teacher.img}'>
        <h3>${teacher.name}</h3>
        ${teacher.position}<br>
        <a href='mailto:${teacher.mail}'>${teacher.mail}</a>
      </article>`;
  }
}
```

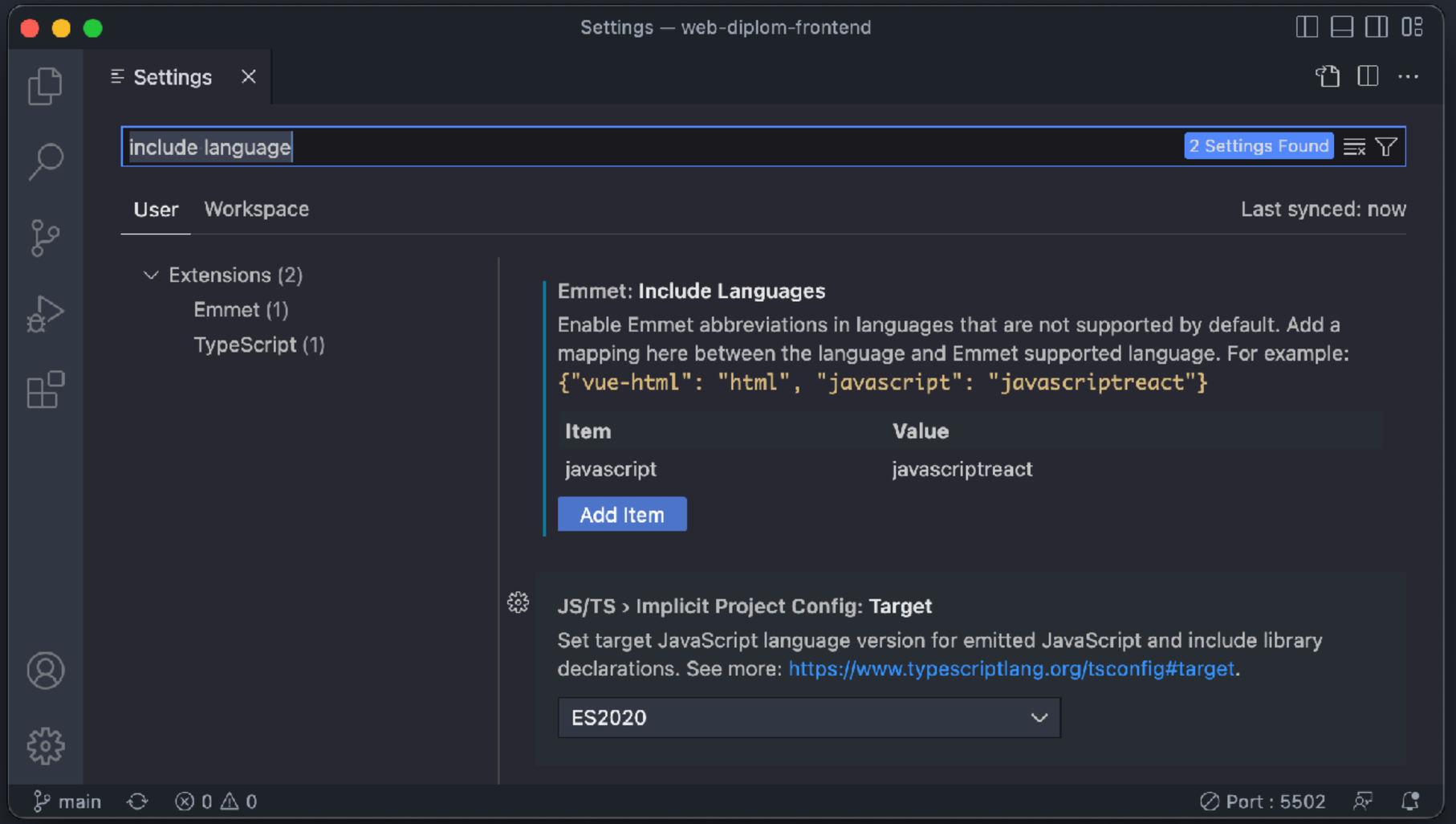
`VS Code ES6 String HTML`

https://marketplace.visualstudio.com/items?itemName=hjb2012.vscode-es6-string-html

```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML += `
      <article>
        <img src='${teacher.img}'>
        <h3>${teacher.name}</h3>
        ${teacher.position}<br>
        <a href='mailto:${teacher.mail}'>${teacher.mail}</a>
      </article>`;
  }
}
```

```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML +=/*html*/ `
    <article>
      <img src='${teacher.img}'>
      <h3>${teacher.name}</h3>
      ${teacher.position}<br>
      <a href='mailto:${teacher.mail}'>${teacher.mail}</a>
    </article>`;
  }
}
```

Add language support in template string



Objects

A set of named values: `key:value`

Objects

A set of named values

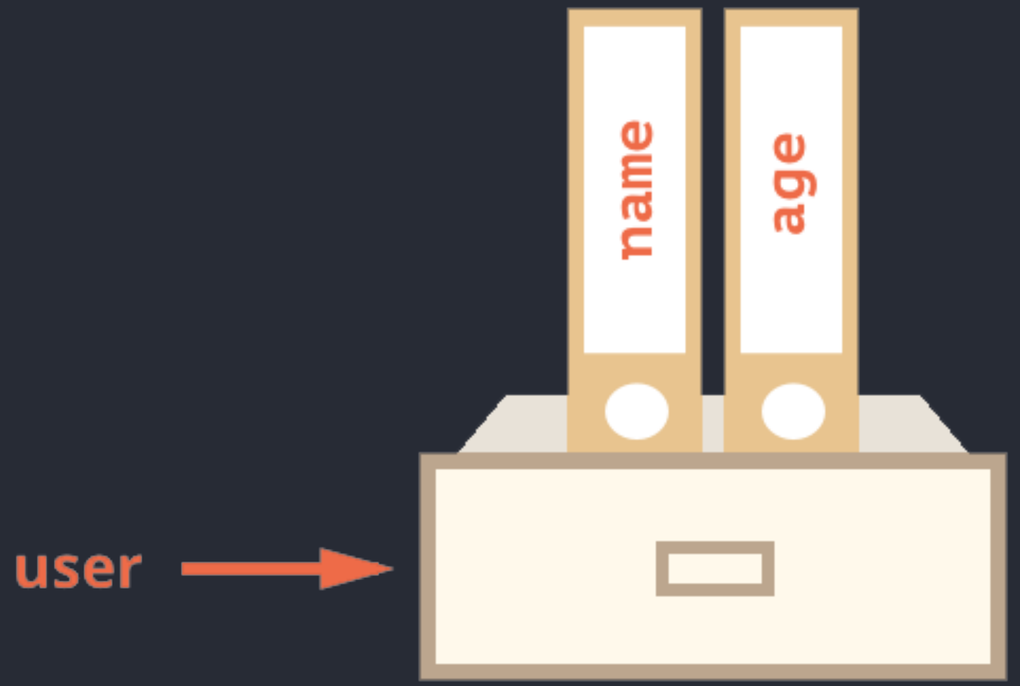
Objects are used to store keyed collections of various data

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

Objects

A set of named values

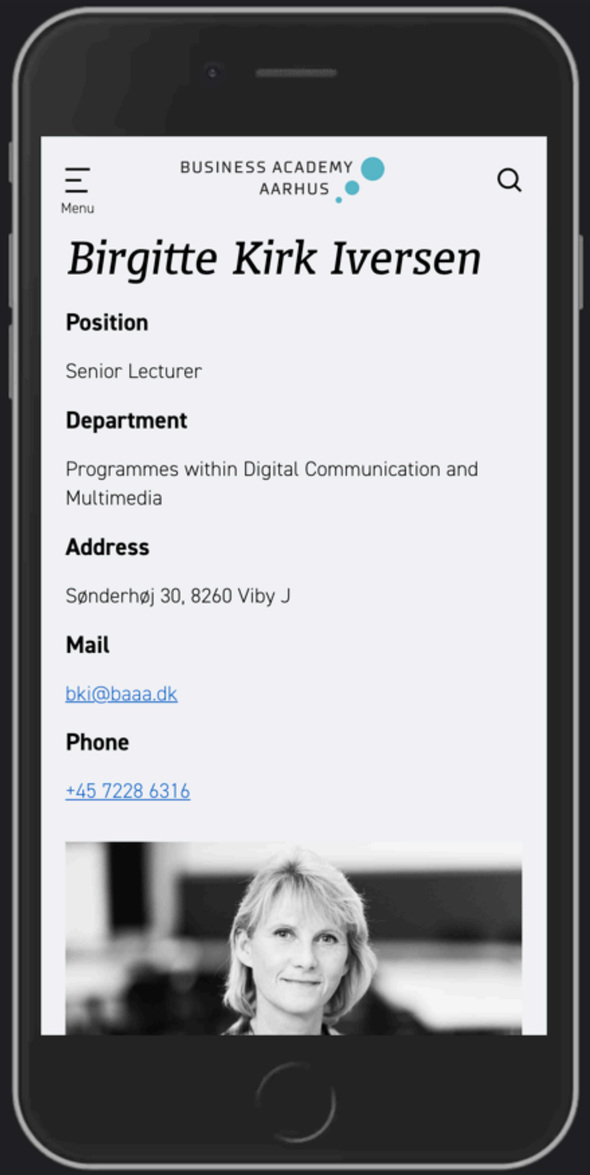
```
let user = {
  name: 'Alicia',
  age: 6
};

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




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

Objects

A set of named values



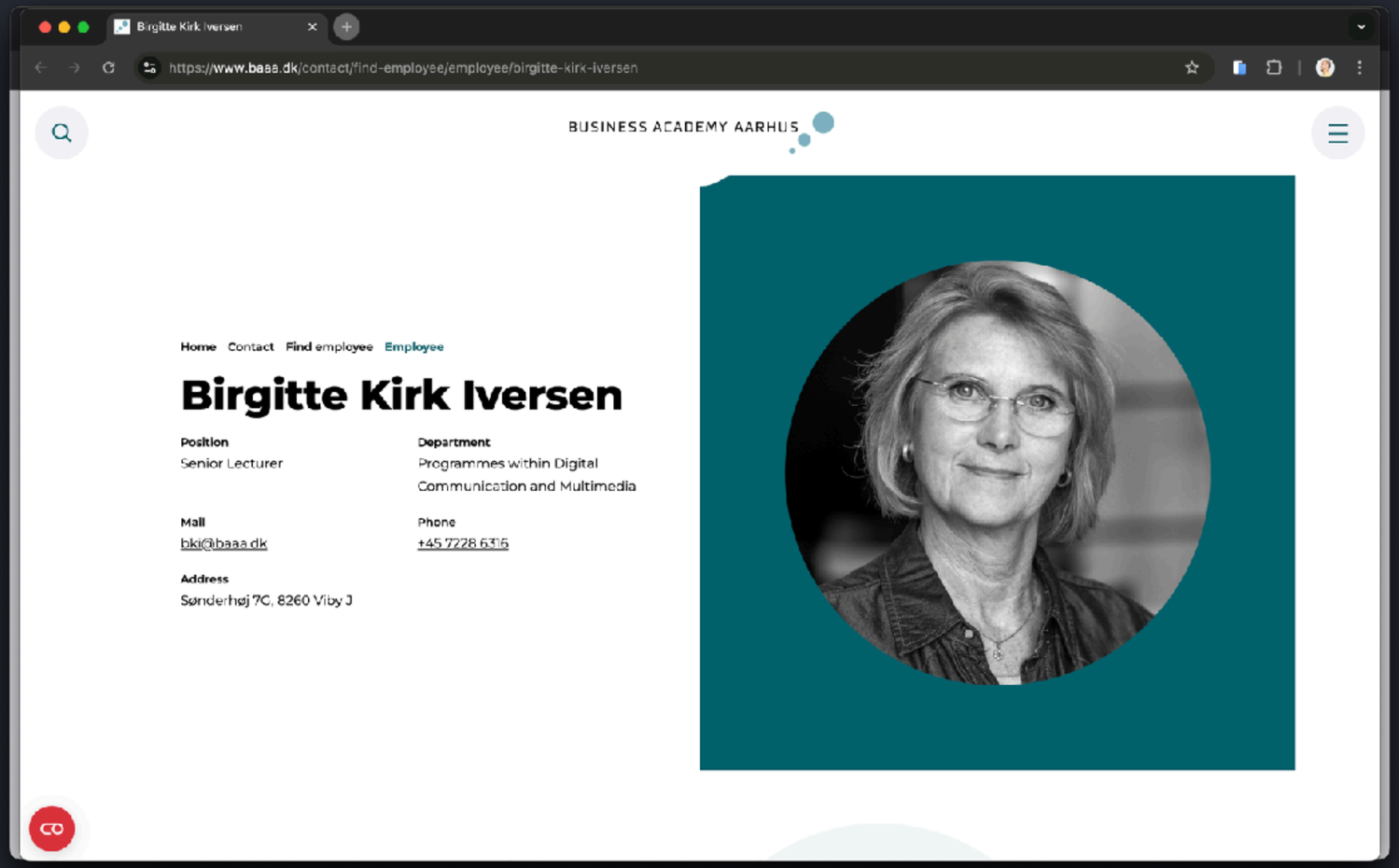
key

value

```
let person = {
    name: "Birgitte Kirk Iversen",
    position: "Senior Lecturer",
    department: "Programmes within Digital Communication and Multimedia",
    address: "Sønderhøj 30, 8260 Viby J",
    mail: "bki@baaa.dk",
    phone: "+45 7228 6316",
    image: "https://www.baaa.dk/media/u4gorzsd/birgitte-kirk-iversen2.jpg"
};
```

https://www.baaa.dk/contact/find-employee/employee/birgitte-kirk-iversen

Objects



https://www.baaa.dk/contact/find-employee/employee/birgitte-kirk-iversen

Objects

A set of named values

key

value

```
const mrBackend = {
  name: "Kasper Fischer Topp",
  mail: "kato@eaaa.dk",
  phone: "72286328",
  position: "Lecturer",
  favTechnologies: ["PHP", "SQL"]
};
```

netflix.com

NETFLIX

Hvem ser?

Personen der rent faktisk betaler for profilen

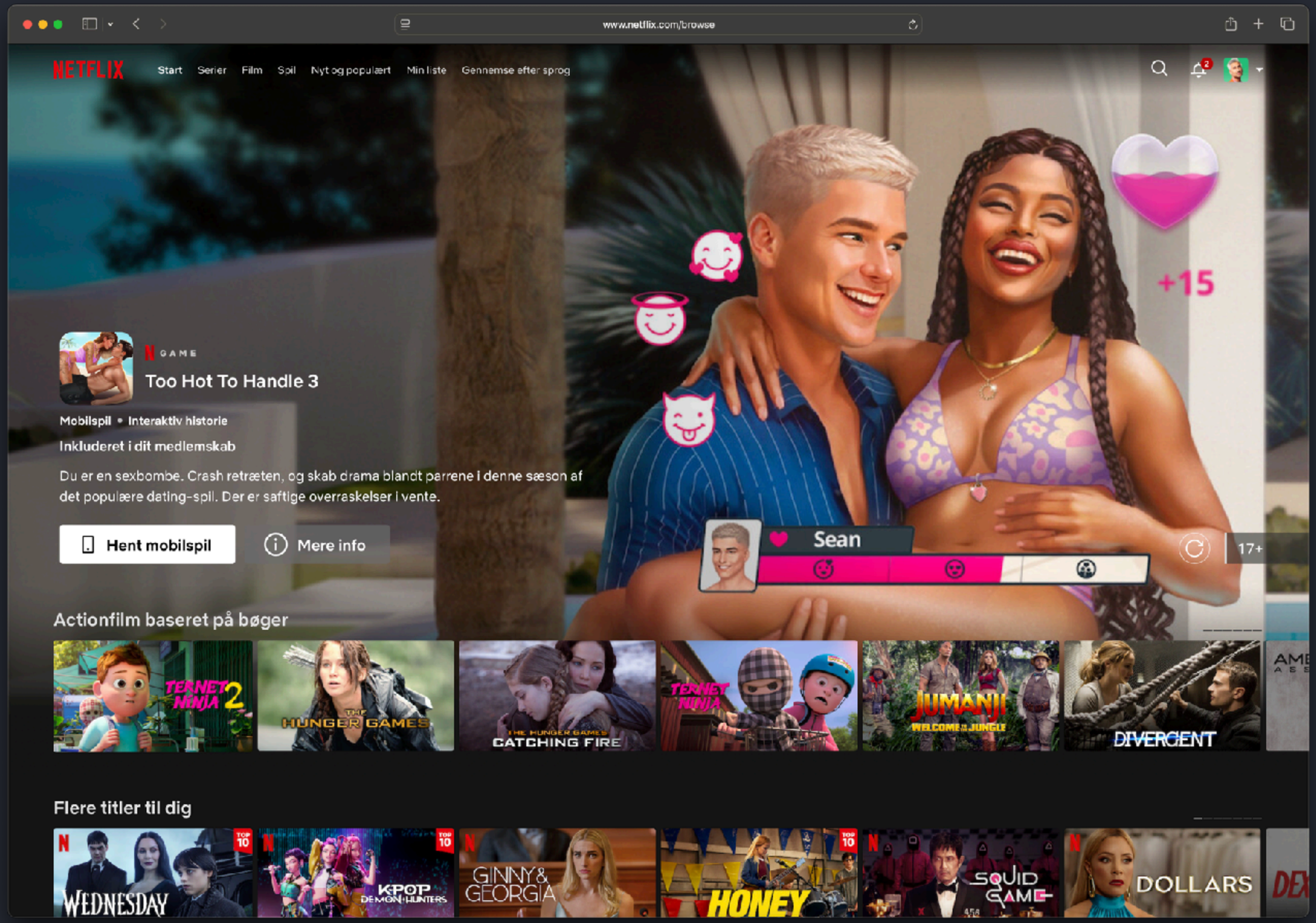
Nasser 1

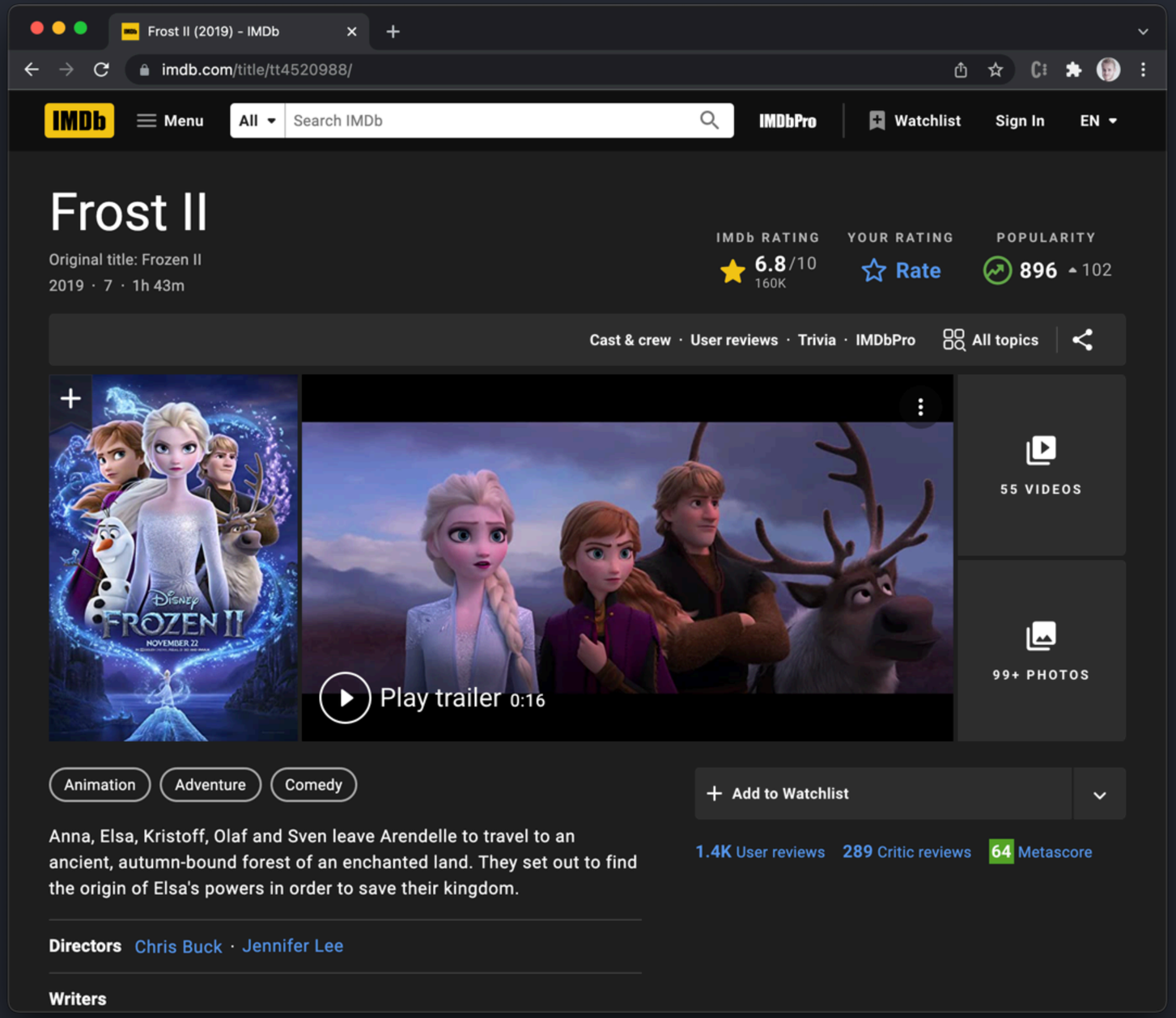
Nasser 2

Nasser 3

Nasser 4 Khader

Administrer profiler


www.netflix.com/browse
NETFLIX
Start
Serier
Film
Spil
Nyt og populært
Min liste
Gennemse efter sprog
GAME
Too Hot To Handle 3
Mobilspil • Interaktiv historie
Inkluderet i dit medlemskab
Du er en sexbombe. Crash retræten, og skab drama blandt parrene i denne sæson af det populære dating-spil. Der er saftige overraskelser i vente.
Hent mobilspil
Mere info
+15
Sean
17+
Actionfilm baseret på bøger
TERNET NINJA 2
THE HUNGER GAMES
THE HUNGER GAMES CATCHING FIRE
TERNET NINJA
JUMANJI WELCOME TO THE JUNGLE
DIVERGENT
Flere titler til dig
WEDNESDAY
KPOP DEMON HUNTERS
GINNY & GEORGIA
HONEY
SQUID GAME
DOLLARS




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

Define yourself as an object

with the following properties

`name, age, mail, phone, city, address`

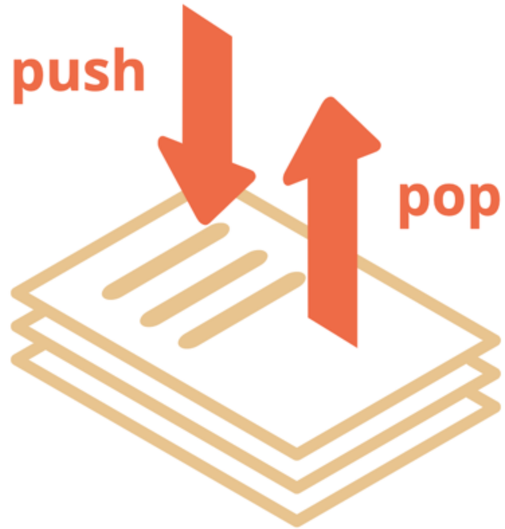
Arrays

Lists - collection of data

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.

Arrays

Collections

Ordered collection of values or objects

```
// Only strings (text)
const movieTitles = ["The Matrix", "Inception"];

// Only numbers
const movieYears = [1999, 2010, 2014];

// Only booleans
const erGode = [true, true, false];

// Blandet indhold (fungerer også!)
const blandedListe = ["The Matrix", 1999, true];

console.log("Film navne:", filmNavne);
console.log("Film år:", filmÅr);
console.log("Er gode:", erGode);
console.log("Blandet:", blandedListe);
```

```
const allMovies = [
  {
    id: 1,
    title: "The Matrix",
    year: 1999,
    rating: 8.7,
    genre: ["Action", "Sci-Fi"]
  },
  {
    id: 2,
    title: "Inception",
    year: 2010,
    rating: 8.8,
    genre: ["Action", "Thriller"]
  }
];

console.log("Complete movie database:", allMovies);
console.log("Number of movies:", allMovies.length);
```

First element

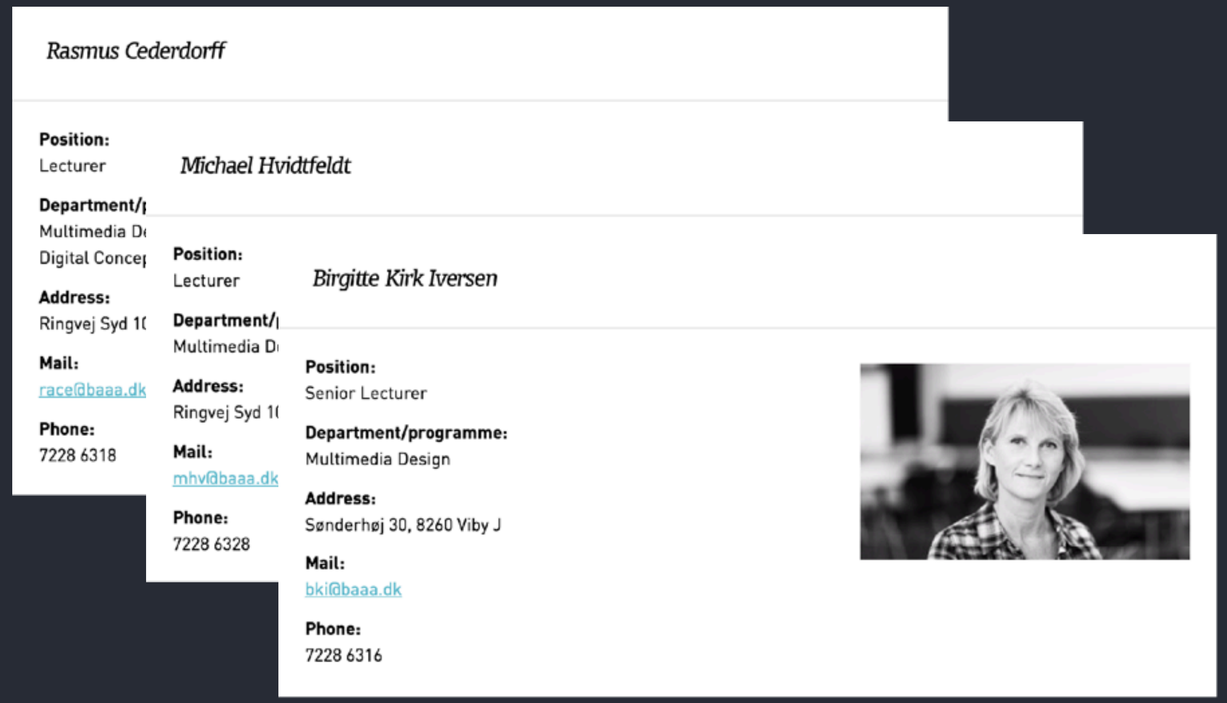
Second element

```
let todaysLecturers = [
  {
    name: "Kasper Fischer Topp",
    mail: "kato@eaaa.dk",
    phone: "72286328",
    position: "Lecturer",
    favTechnologies: ["PHP", "SQL"],
    nickname: "Mr. Backend"
  },
  {
    name: "Rasmus Cederdorff",
    mail: "race@eaaa.dk",
    phone: "72286318",
    position: "Lecturer",
    favTechnologies: ["JavaScript"],
    nickname: "Mr. Frontend"
  }
];
```

First element

Second element

Arrays




```
main.js:21
▼(3) [{…}, {…}, {…}] i
  ▶0: {name: "Birgitte Kirk Iversen", mail: "bki@baaa…
  ▶1: {name: "Michael Hvidtfeldt", mail: "mhv@baaa.dk…
  ▶2: {name: "Rasmus Cederdorff", mail: "race@baaa.dk…
    length: 3
  ▶__proto__: Array(0)
main.js:22
▶{name: "Michael Hvidtfeldt", mail: "mhv@baaa.dk"}
3
main.js:23
```

```
let teachers = [{
    name: "Birgitte Kirk Iversen",
    mail: "bki@baaa.dk"
  },
  {
    name: "Michael Hvidtfeldt",
    mail: "mhv@baaa.dk"
  },
  {
    name: "Rasmus Cederdorff",
    mail: "race@baaa.dk"
  }
];

console.log(teachers);
console.log(teachers[1]);
console.log(teachers.length);
```

Teachers

http://127.0.0.1:5501/array-teachers/index.html

Teachers

Birgitte Kirk Iversen

Senior Lecturer
bki@baaa.dk

Maria Louise Bendixen

Senior Lecturer
mlbe@baaa.dk

Kim Elkjær Marcher-Jepsen

Lecturer
kije@baaa.dk

Rasmus Cederdorff

Lecturer
race@baaa.dk

Console

top

Filter

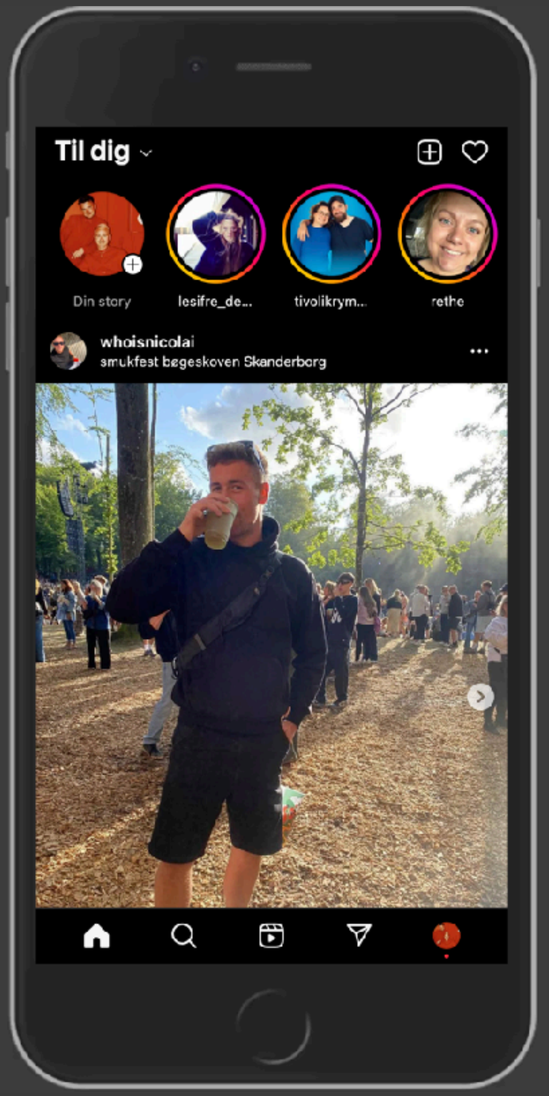
Default levels

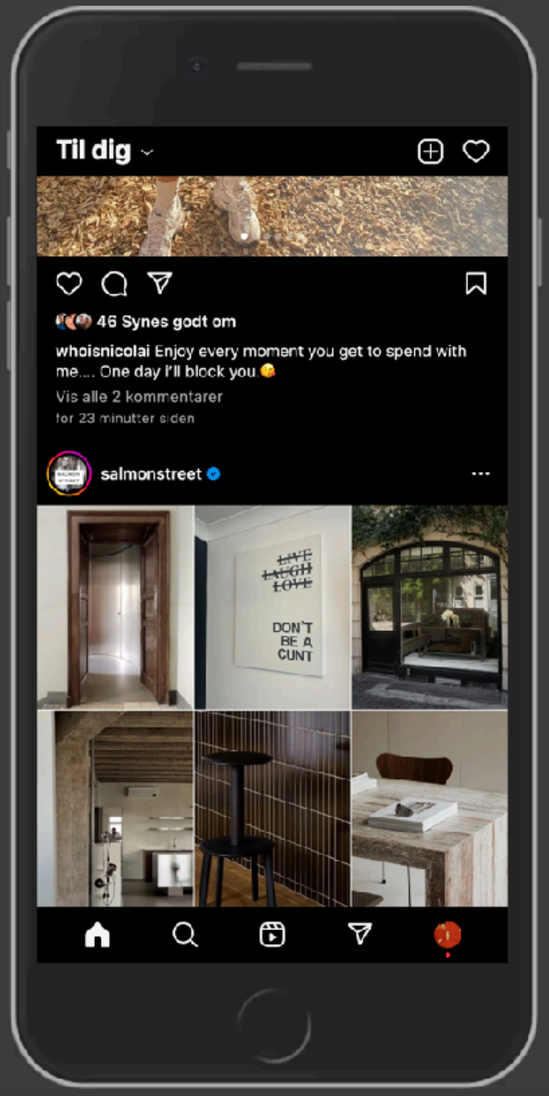
1 Issue: 1

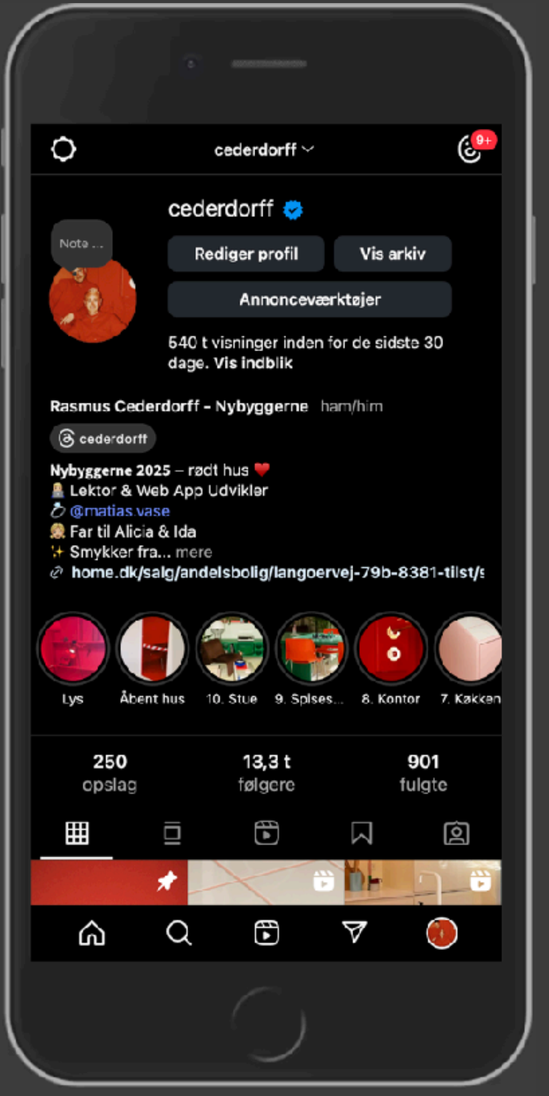
main.js:46

```
(4) [{…}, {…}, {…}, {…}]
  0:
    address: "Sønderhøj 30, 8260 Viby J"
    department: "Multimedia Design"
    img: "https://www.eaaa.dk/media/u4gorzs
    initials: "bki"
    mail: "bki@baaa.dk"
    name: "Birgitte Kirk Iversen"
    phone: "72286316"
    position: "Senior Lecturer"
    [[Prototype]]: Object
  1: {name: 'Maria Louise Bendixen', initia
  2: {name: 'Kim Elkjær Marcher-Jepsen', in
  3: {name: 'Rasmus Cederdorff', initials:
  length: 4
  [[Prototype]]: Array(0)
```

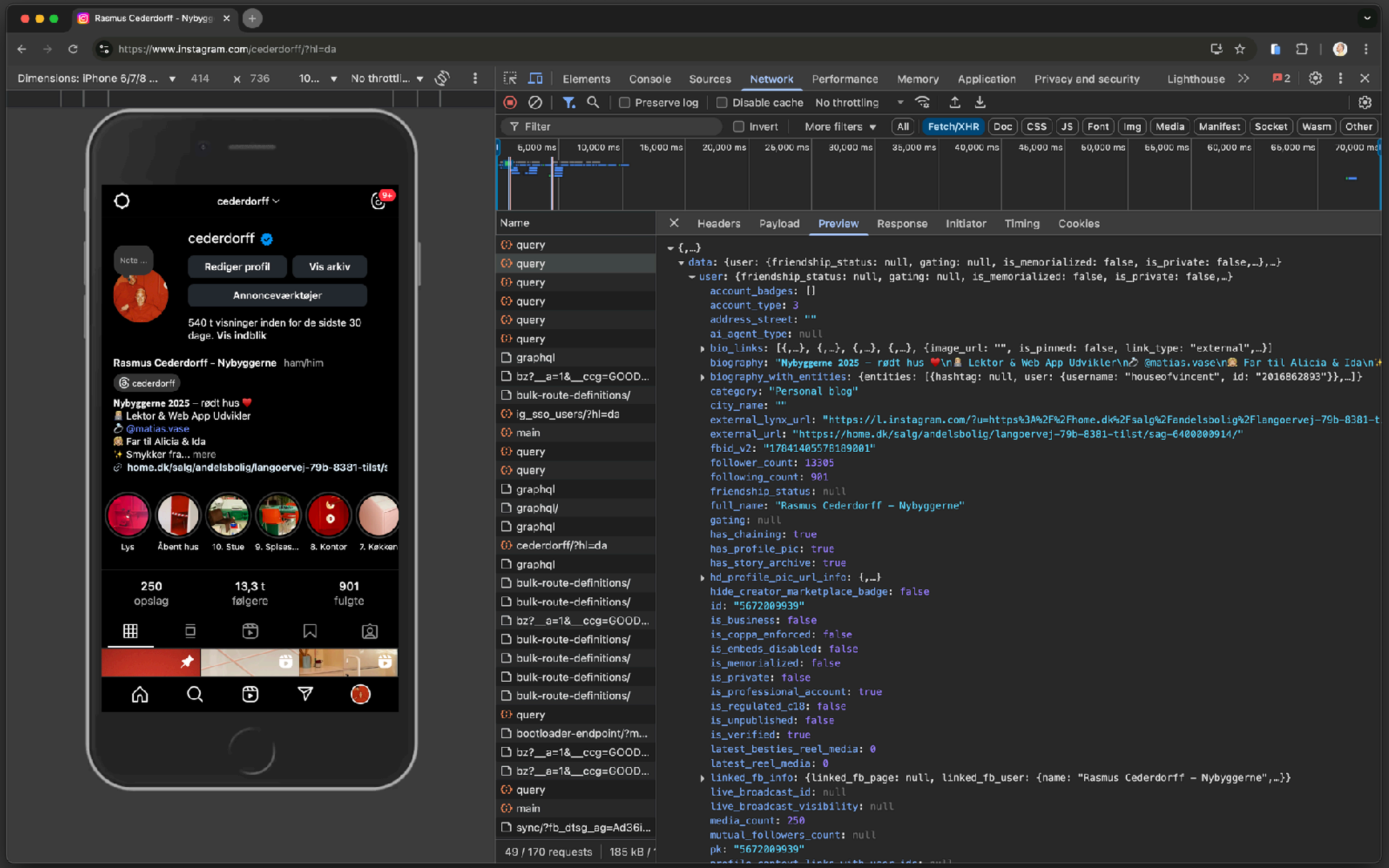
array-teachers

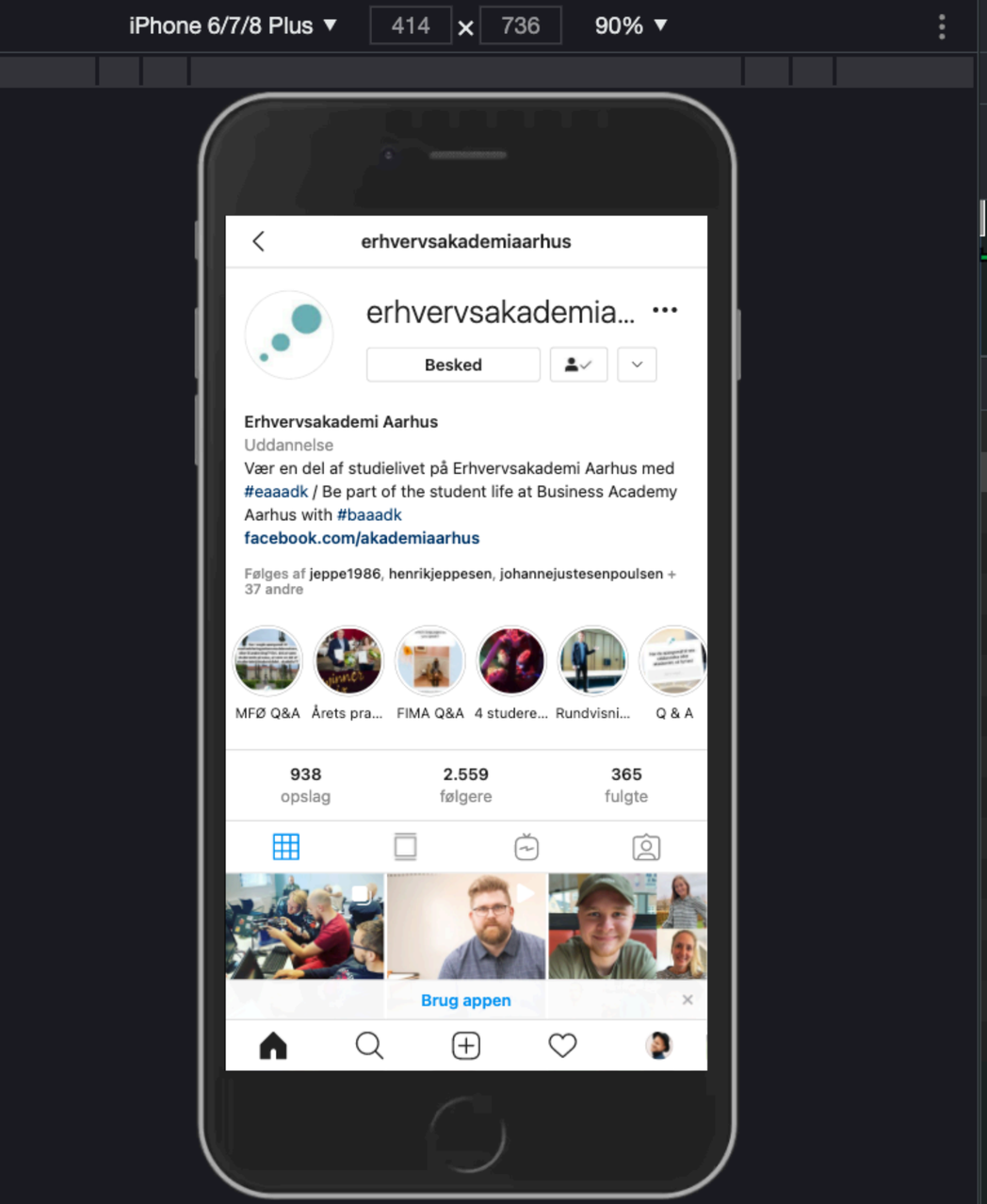

Til dig
Din story
lesifre_de...
tivolikrym...
rethe
whoisnicolai
smukfest begeskoven Skanderborg

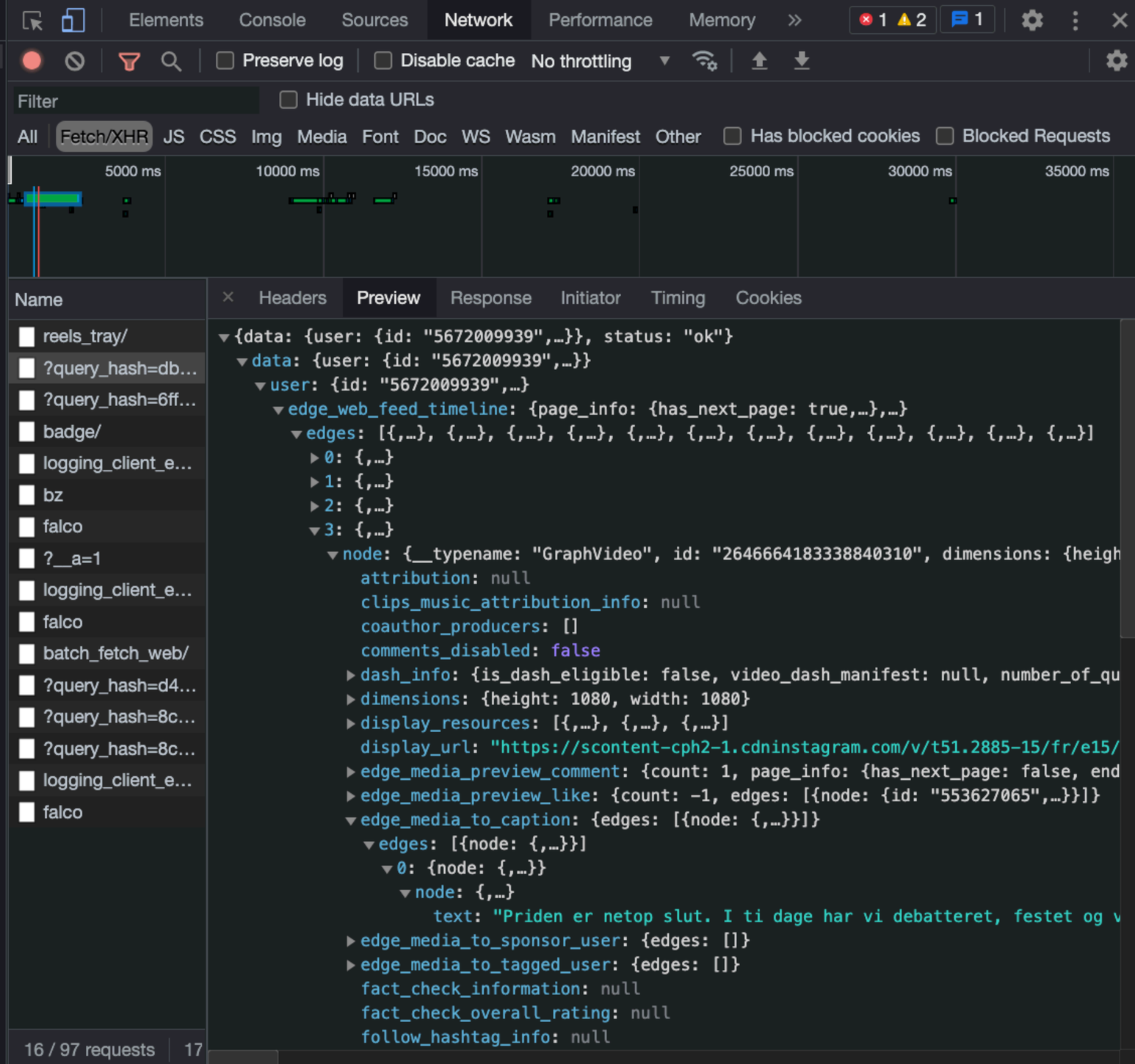

Til dig
46 Synes godt om
whoisnicolai Enjoy every moment you get to spend with me.... One day I'll block you
Vis alle 2 kommentarer
for 23 minutter siden
salmonstreet
DON'T BE A CUNT

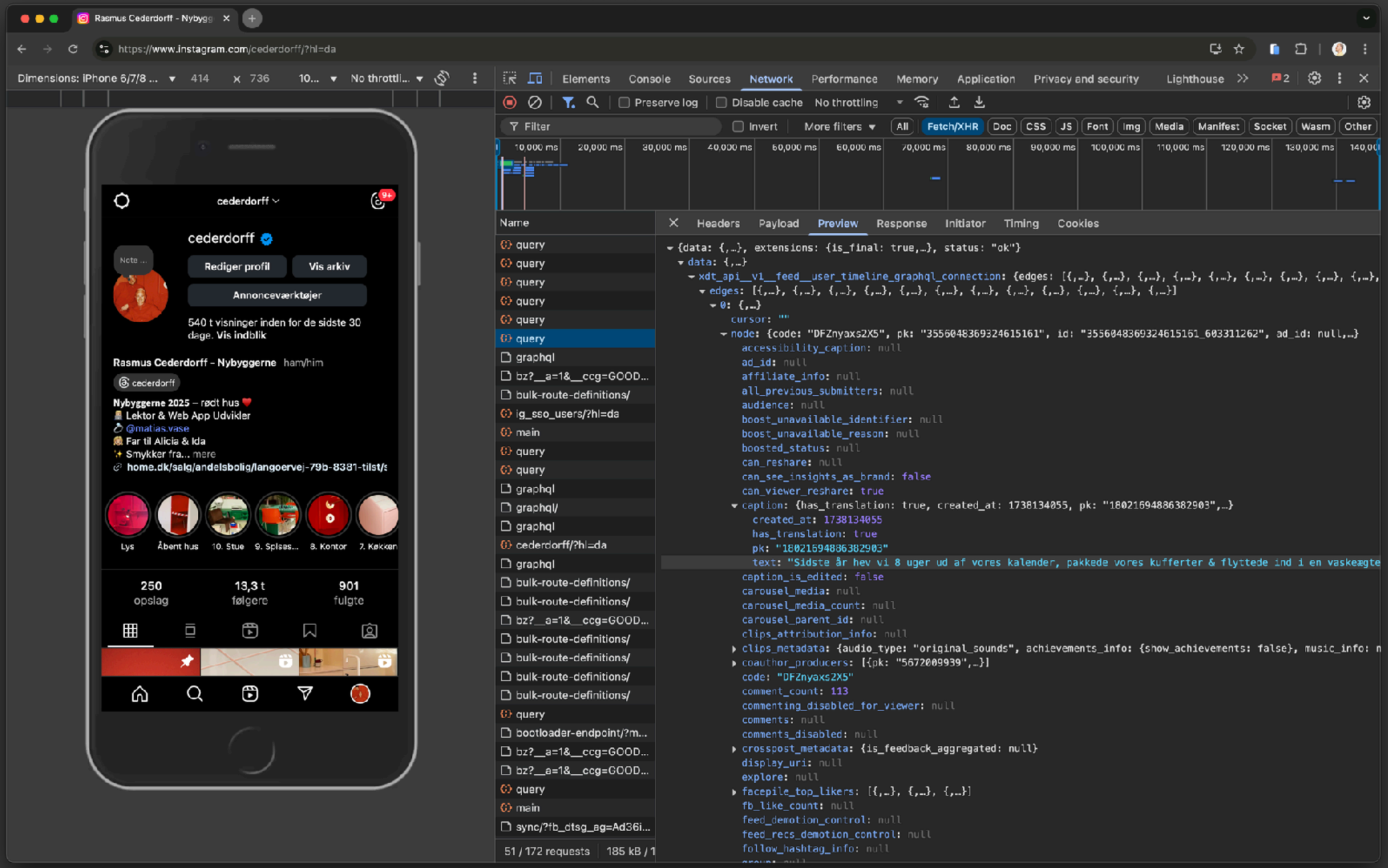

cederdorff
cederdorff
Rediger profil
Vis arkiv
Annonceværktøjer
640 t visninger inden for de sidste 30 dage. Vis indblik
Rasmus Cederdorff - Nybyggerne he/him
cederdorff
Nybyggerne 2025 – rødt hus
Lektor & Web App Udvikler
@matias.vase
Far til Alicia & Ida
Smykker fra... mere
home.dk/salg/andelsbolig/langoervej-79b-8381-tilst/
Lys
Åbent hus
10. Stue
9. Spises...
8. Kontor
7. Køkken
250 opslag
13,3 t følgere
901 fulgte

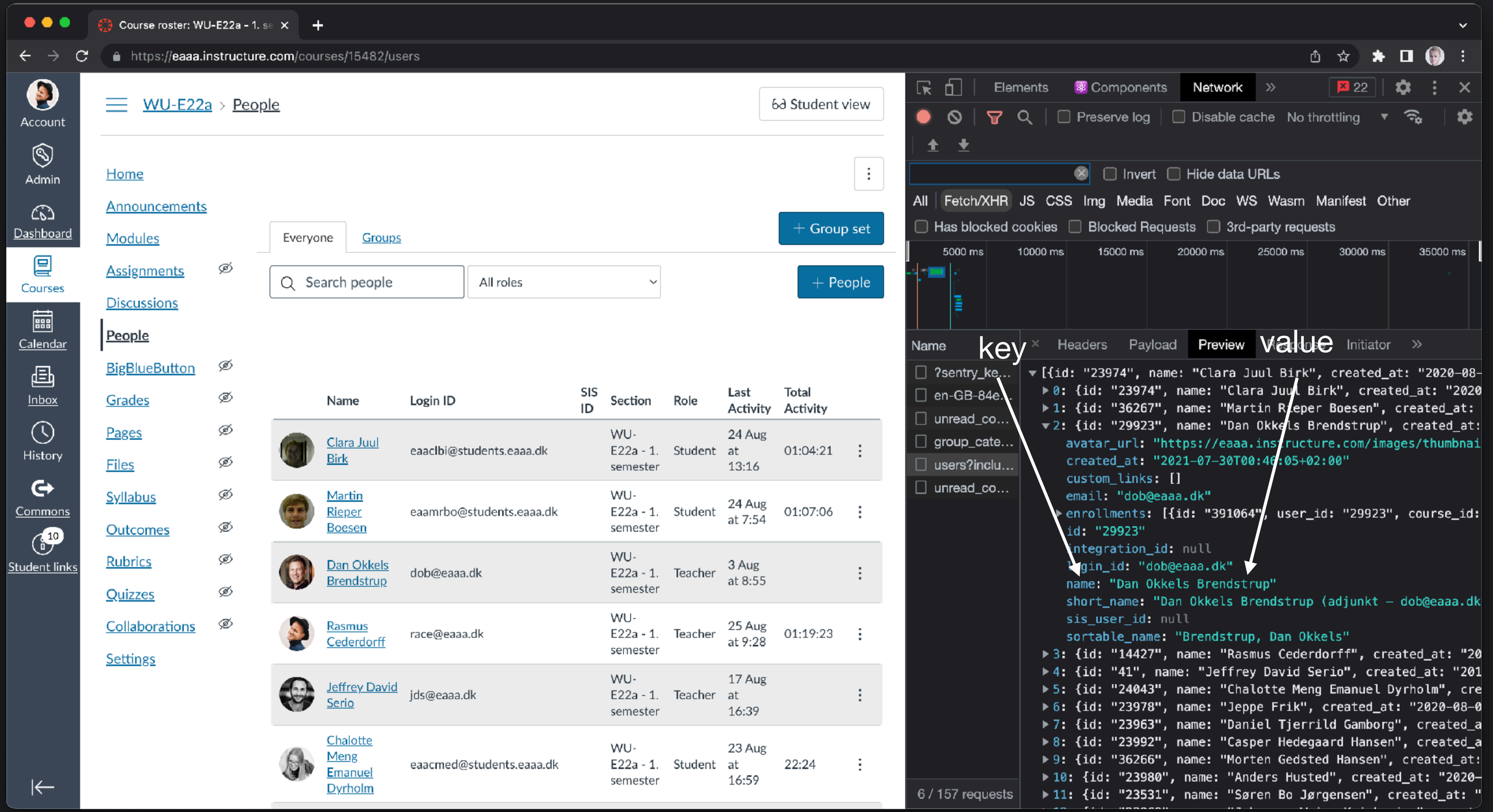

cederdorff
Lys
Åbent hus
10. Stue
9. Spises...
8. Kontor
7. Køkken
250 opslag
13,3 t følgere
901 fulgte


Rasmus Cederdorff - Nybygg
https://www.instagram.com/cederdorff/?hl=da
Dimensions: IPhone 6/7/8 ... 414 x 736 10... No throttli...
cederdorff
cederdorff
Rediger profil
Vis arkiv
Annonceværktøjer
540 t visninger inden for de sidste 30 dage. Vis indblik
Rasmus Cederdorff - Nybyggerne he/him
cederdorff
Nybyggerne 2025 – rødt hus
Lektor & Web App Udvikler
@matias.vase
Far til Alicia & Ida
Smykker fra... mere
home.dk/salg/andelsbolig/langoervej-79b-8381-tilst/s
Lys
Åbent hus
10. Stue
9. Spisas...
8. Kontor
7. Køkken
250 opslag
13,3 t følgere
901 fulgte
Elements
Console
Sources
Network
Performance
Memory
Application
Privacy and security
Lighthouse
Preserve log
Disable cache
No throttling
Filter
Invert
More filters
All
Fetch/XHR
Doc
CSS
JS
Font
Img
Media
Manifest
Socket
Wasm
Other
Name
Headers
Payload
Preview
Response
Initiator
Timing
Cookies
query
graphql
bz?__a=1&__ccg=GOOD...
bulk-route-definitions/
ig_sso_users/?hl=da
main
graphql/
cederdorff/?hl=da
bootloader-endpoint/?m...
sync/?fb_dtsg_ag=Ad36i...
49 / 170 requests
185 kB /
{,...}
data: {user: {friendship_status: null, gating: null, is_memorialized: false, is_private: false,...},...}
user: {friendship_status: null, gating: null, is_memorialized: false, is_private: false,...}
account_badges: []
account_type: 3
address_street: ""
ai_agent_type: null
bio_links: [{,...}, {,...}, {,...}, {,...}, {image_url: "", is_pinned: false, link_type: "external",...}]
biography: "Nybyggerne 2025 – rødt hus ❤️\n👨 Lektor & Web App Udvikler\n🎣 @matias.vase\n👧 Far til Alicia & Ida\n✨
biography_with_entities: {entities: [{hashtag: null, user: {username: "houseofvincent", id: "2016862893"}},...]}
category: "Personal blog"
city_name: ""
external_lynx_url: "https://l.instagram.com/?u=https%3A%2F%2Fhome.dk%2Fsalg%2Fandelsbolig%2Flangoervej-79b-8381-t
external_url: "https://home.dk/salg/andelsbolig/langoervej-79b-8381-tilst/sag-6400009914/"
fbid_v2: "17841405578189001"
follower_count: 13305
following_count: 901
friendship_status: null
full_name: "Rasmus Cederdorff – Nybyggerne"
gating: null
has_chaining: true
has_profile_pic: true
has_story_archive: true
hd_profile_pic_url_info: {,...}
hide_creator_marketplace_badge: false
id: "5672009939"
is_business: false
is_coppa_enforced: false
is_embeds_disabled: false
is_memorialized: false
is_private: false
is_professional_account: true
is_regulated_c18: false
is_unpublished: false
is_verified: true
latest_besties_reel_media: 0
latest_reel_media: 0
linked_fb_info: {linked_fb_page: null, linked_fb_user: {name: "Rasmus Cederdorff – Nybyggerne",...}}
live_broadcast_id: null
live_broadcast_visibility: null
media_count: 250
mutual_followers_count: null
pk: "5672009939"


iPhone 6/7/8 Plus
414
736
90%
erhvervsakademiaarhus
erhvervsakademia...
Besked
Erhvervsakademi Aarhus
Uddannelse
Vær en del af studielivet på Erhvervsakademi Aarhus med #eaaadk / Be part of the student life at Business Academy Aarhus with #baaadk
facebook.com/akademiaarhus
Følges af jeppe1986, henrikjeppesen, johannejustesenpoulsen + 37 andre
MFØ Q&A
Årets pra...
FIMA Q&A
4 studere...
Rundvisni...
Q & A
938
opslag
2.559
følgere
365
fulgte
Brug appen

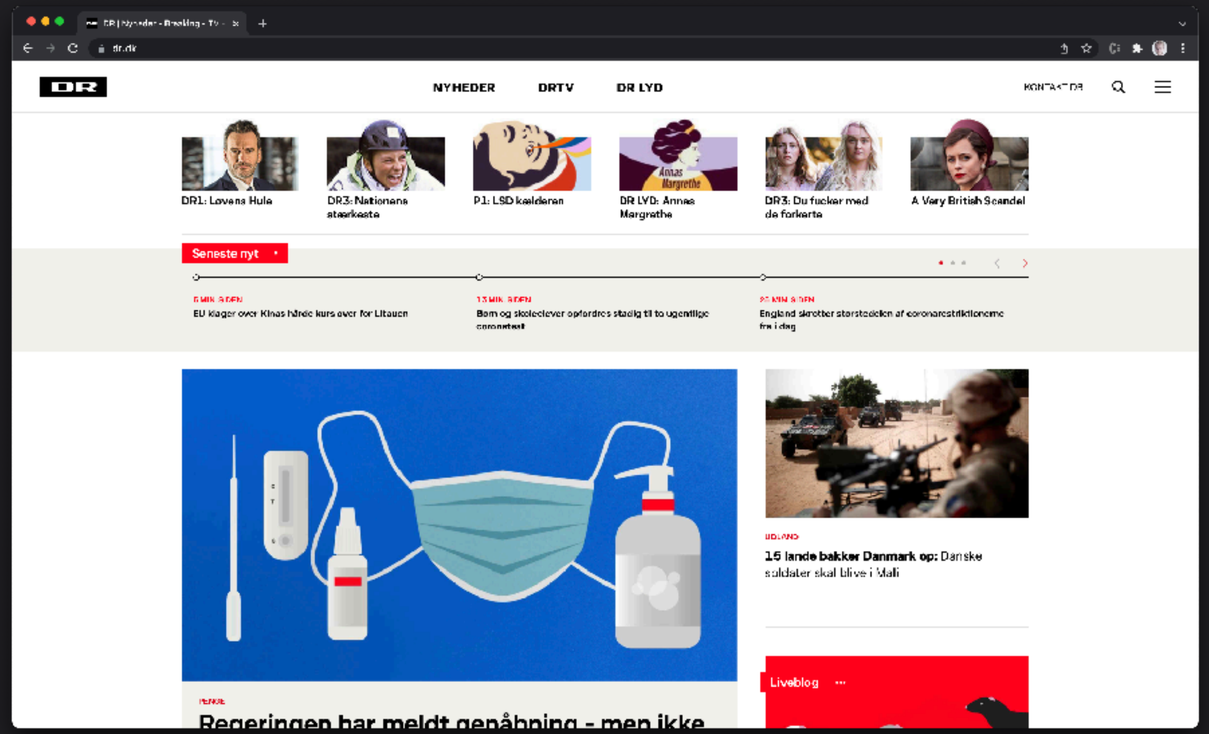

Elements
Console
Sources
Network
Performance
Memory
Preserve log
Disable cache
No throttling
Filter
Hide data URLs
All
Fetch/XHR
JS
CSS
Img
Media
Font
Doc
WS
Wasm
Manifest
Other
Has blocked cookies
Blocked Requests
5000 ms
10000 ms
15000 ms
20000 ms
25000 ms
30000 ms
35000 ms
Name
reels_tray/
?query_hash=db...
?query_hash=6ff...
badge/
logging_client_e...
bz
falco
?__a=1
logging_client_e...
falco
batch_fetch_web/
?query_hash=d4...
?query_hash=8c...
?query_hash=8c...
logging_client_e...
falco
16 / 97 requests
17
Headers
Preview
Response
Initiator
Timing
Cookies
{data: {user: {id: "5672009939",…}}, status: "ok"}
data: {user: {id: "5672009939",…}}
user: {id: "5672009939",…}
edge_web_feed_timeline: {page_info: {has_next_page: true,…},…}
edges: [{,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}]
0: {,…}
1: {,…}
2: {,…}
3: {,…}
node: {__typename: "GraphVideo", id: "2646664183338840310", dimensions: {heigh
attribution: null
clips_music_attribution_info: null
coauthor_producers: []
comments_disabled: false
dash_info: {is_dash_eligible: false, video_dash_manifest: null, number_of_qu
dimensions: {height: 1080, width: 1080}
display_resources: [{,…}, {,…}, {,…}]
display_url: "https://scontent-cph2-1.cdninstagram.com/v/t51.2885-15/fr/e15/
edge_media_preview_comment: {count: 1, page_info: {has_next_page: false, end
edge_media_preview_like: {count: -1, edges: [{node: {id: "553627065",…}}]}
edge_media_to_caption: {edges: [{node: {,…}}]}
edges: [{node: {,…}}]
0: {node: {,…}}
node: {,…}
text: "Priden er netop slut. I ti dage har vi debatteret, festet og v
edge_media_to_sponsor_user: {edges: []}
edge_media_to_tagged_user: {edges: []}
fact_check_information: null
fact_check_overall_rating: null
follow_hashtag_info: null

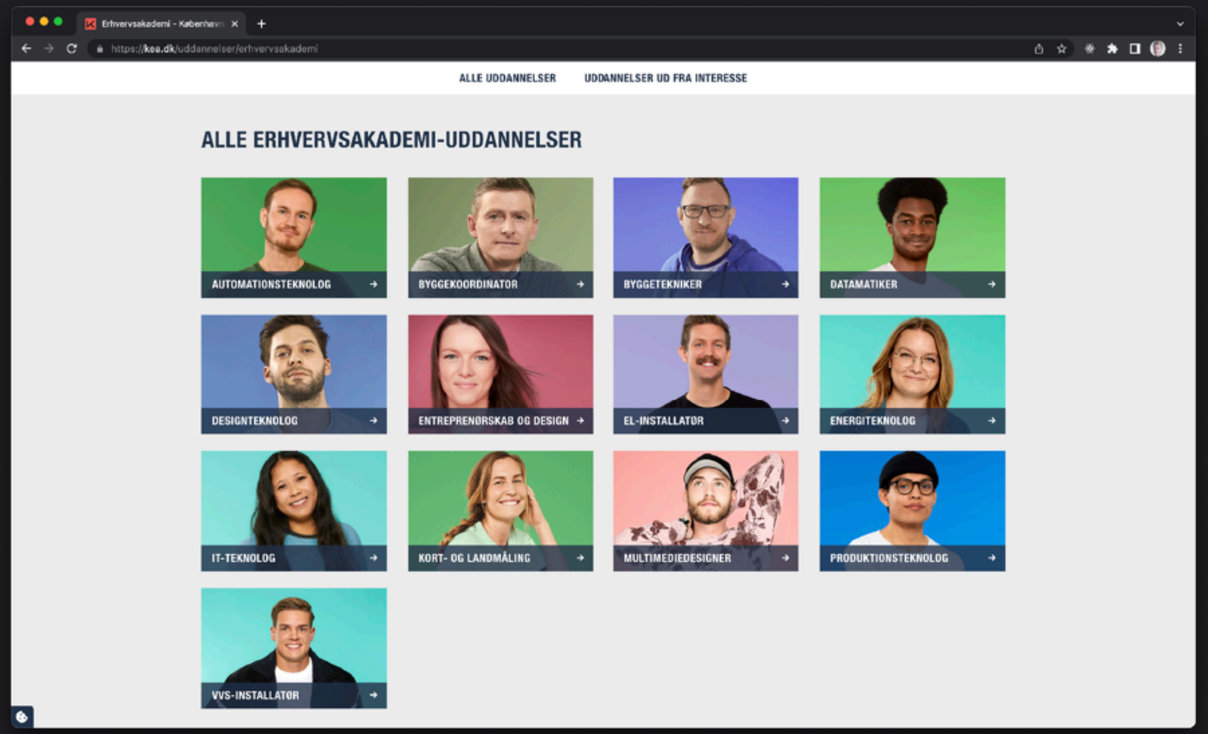

Rasmus Cederdorff - Nybygg
https://www.instagram.com/cederdorff/?hl=da
Dimensions: IPhone 6/7/8 ... 414 x 736 10... No throttli...
Elements Console Sources Network Performance Memory Application Privacy and security Lighthouse
Preserve log Disable cache No throttling
Filter Invert More filters All Fetch/XHR Doc CSS JS Font Img Media Manifest Socket Wasm Other
10,000 ms 20,000 ms 30,000 ms 40,000 ms 50,000 ms 60,000 ms 70,000 ms 80,000 ms 90,000 ms 100,000 ms 110,000 ms 120,000 ms 130,000 ms 140,000
cederdorff
cederdorff
Rediger profil
Vis arkiv
Annonceværktøjer
540 t visninger inden for de sidste 30 dage. Vis indblik
Rasmus Cederdorff - Nybyggerne ham/him
cederdorff
Nybyggerne 2025 – rødt hus
Lektor & Web App Udvikler
@matias.vase
Far til Alicia & Ida
Smykker fra... mere
home.dk/salg/andelsbolig/langoervej-79b-8381-tilst/s
Lys
Åbent hus
10. Stue
9. Spisas...
8. Kontor
7. Køkken
250 opslag
13,3 t følgere
901 fulgte
Name
Headers Payload Preview Response Initiator Timing Cookies
query
graphql
bz?__a=1&__ccg=GOOD...
bulk-route-definitions/
ig_sso_users/?hl=da
main
cederdorff/?hl=da
bootloader-endpoint/?m...
sync/?fb_dtsg_ag=Ad36i...
51 / 172 requests
185 kB / 1
{data: {,…}, extensions: {is_final: true,…}, status: "ok"}
data: {,…}
xdt_api__v1__feed__user_timeline_graphql_connection: {edges: [{,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…},
edges: [{,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}]
0: {,…}
cursor: ""
node: {code: "DFZnyaxs2X5", pk: "3556048369324615161", id: "3556048369324615161_603311262", ad_id: null,…}
accessibility_caption: null
ad_id: null
affiliate_info: null
all_previous_submitters: null
audience: null
boost_unavailable_identifier: null
boost_unavailable_reason: null
boosted_status: null
can_reshare: null
can_see_insights_as_brand: false
can_viewer_reshare: true
caption: {has_translation: true, created_at: 1738134055, pk: "18021694886382903",…}
created_at: 1738134055
has_translation: true
pk: "18021694886382903"
text: "Sidste år hev vi 8 uger ud af vores kalender, pakkede vores kufferter & flyttede ind i en vaskeægte
caption_is_edited: false
carousel_media: null
carousel_media_count: null
carousel_parent_id: null
clips_attribution_info: null
clips_metadata: {audio_type: "original_sounds", achievements_info: {show_achievements: false}, music_info: n
coauthor_producers: [{pk: "5672009939",…}]
code: "DFZnyaxs2X5"
comment_count: 113
commenting_disabled_for_viewer: null
comments: null
comments_disabled: null
crosspost_metadata: {is_feedback_aggregated: null}
display_uri: null
explore: null
facepile_top_likers: [{,…}, {,…}, {,…}]
fb_like_count: null
feed_demotion_control: null
feed_recs_demotion_control: null
follow_hashtag_info: null



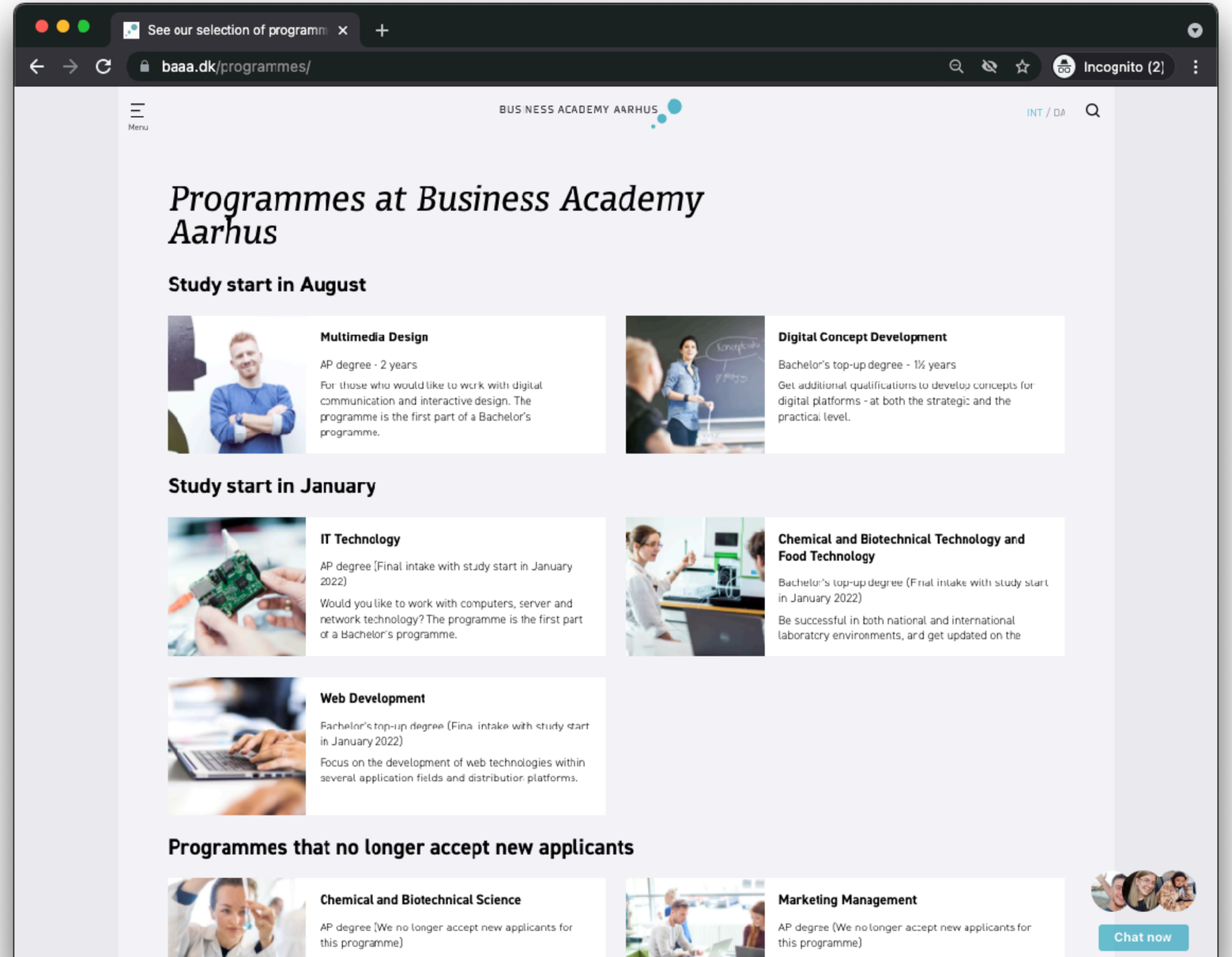
eaaa.instructure.com/api/v1/courses/15482/users?include_inactive=true&include%5B%5D=avatar_url&include%5B%5D=email&include%5B%5D=observed_users&per_page=50

Objects? Arrays?





Objects with properties in arrays



It’s all objects & arrays!

Data Types & Data Structures

Objects & Arrays

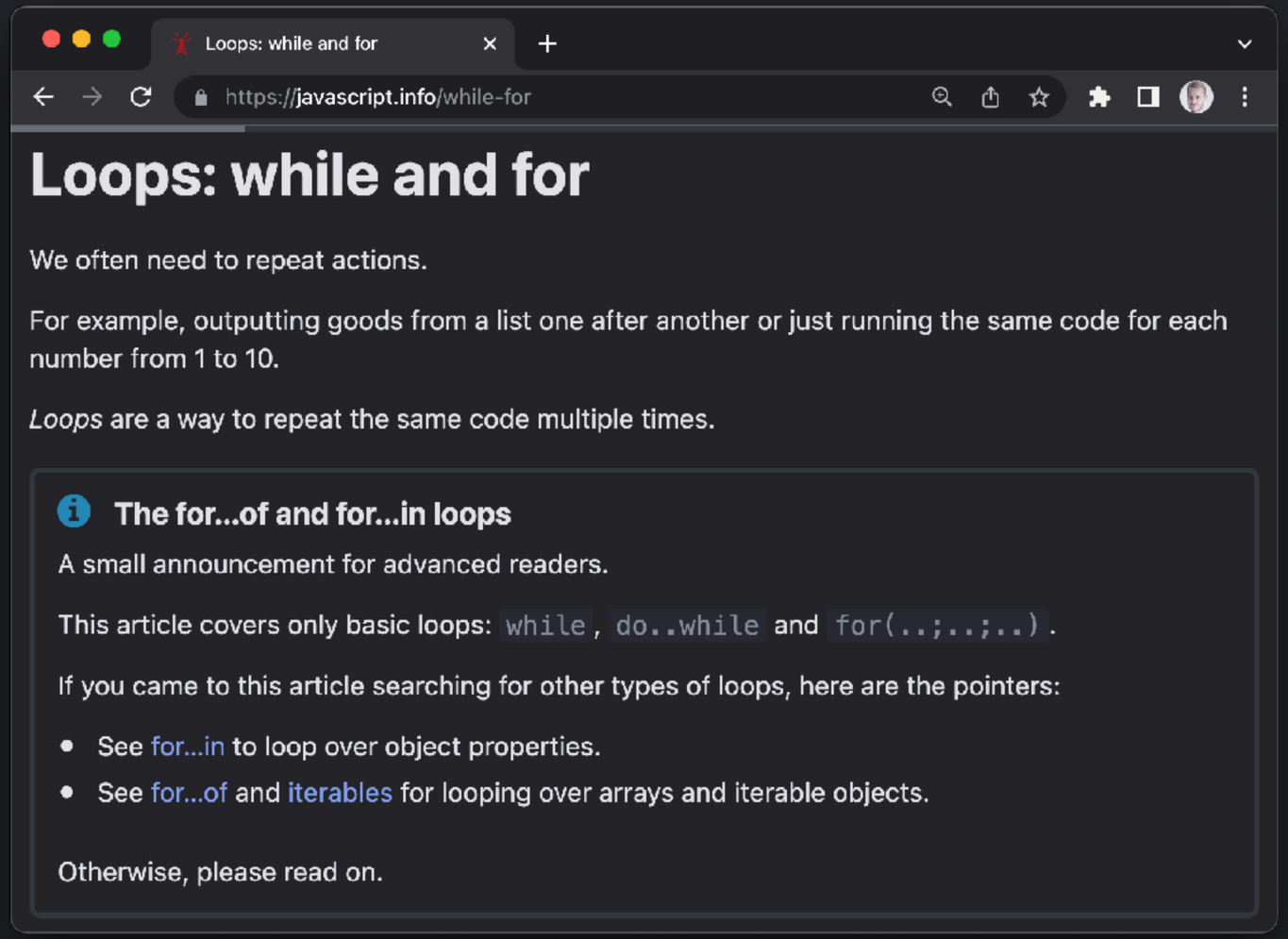
Arrays

Loops

```
for (let teacher of teachers) {
  console.log(teacher);
}
```

```
▶ {name: "Birgitte Kirk Iversen", mail: "bki@baaa.dk"}    main.js:20
▶ {name: "Michael Hvidtfeldt", mail: "mhv@baaa.dk"}       main.js:20
▶ {name: "Rasmus Cederdorff", mail: "race@baaa.dk"}       main.js:20
```

Loops



https://javascript.info/while-for

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

Loops

```
for (const familyMember of familyMembers) {
  console.log(familyMember);
}

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

https://www.w3schools.com/js/js_loop_for.asp
https://javascript.info/array#loops
https://javascript.info/while-for

JavaScript.info/array#loops

One of the oldest ways to cycle array items is the `for` loop over indexes:

```
let arr = ["Apple", "Orange", "Pear"];

for (let i = 0; i < arr.length; i++) {
  alert( arr[i] );
}
```

But for arrays there is another form of loop, `for..of`:

```
let fruits = ["Apple", "Orange", "Plum"];

// iterates over array elements
for (let fruit of fruits) {
  alert( fruit );
}
```

```
const allMovies = [
  {
    id: 1,
    titel: "The Matrix",
    år: 1999,
    rating: 8.7,
    genre: ["Action", "Sci-Fi"],
    instruktører: ["Lana Wachowski", "Lilly Wachowski"]
  },
  {
    id: 2,
    titel: "Inception",
    år: 2010,
    rating: 8.8,
    genre: ["Action", "Thriller", "Sci-Fi"],
    instruktører: ["Christopher Nolan"]
  },
  {
    id: 3,
    titel: "The Dark Knight",
    år: 2008,
    rating: 9.0,
    genre: ["Action", "Crime", "Drama"],
    instruktører: ["Christopher Nolan"]
  }
];
```

```
// Loop through all movies
for (const movie of expandedMovieDatabase) {
  console.log(`🎬 ${movie.title} (${movie.year})`);
  console.log(`⭐ Rating: ${movie.rating}`);
  console.log(`🎭 Genre: ${movie.genre[0]}`);
  console.log("---");
}
```

ARRAYS

LOOPS

```
for (let teacher of teachers) {
  console.log(teacher.mail);
}
```

```
bki@baaa.dk                                    main.js:20
mhv@baaa.dk                                    main.js:20
race@baaa.dk                                   main.js:20
```

LOOPS

... LOOP THROUGH AN ARRAY AND ADD A CONDITION

```
for (let teacher of teachers) {
  if (teacher.name === "Rasmus Cederdorff") {
    console.log(teacher);
  }
}
```

```
▶ {name: "Rasmus Cederdorff", mail: "race@baaa.dk"}        main.js:21
```

Functions

Write reusable code

Functions

A block of code to perform a specific task.

A way to make reusable code by storing tasks we can use again and again.

Best practice: write reusable code

```
function log(message) {
  console.log(message);
}

log("Hi Frontenders!");
```

https://javascript.info/function-basics

Functions

3 different types

```
function log(message) {
    console.log(message);
}
```

FUNCTION DECLARATION

```
const log = function (message) {
    console.log(message);
};
```

FUNCTION EXPRESSION

```
const log = (message) => {
    console.log(message);
};
```

ARROW FUNCTION

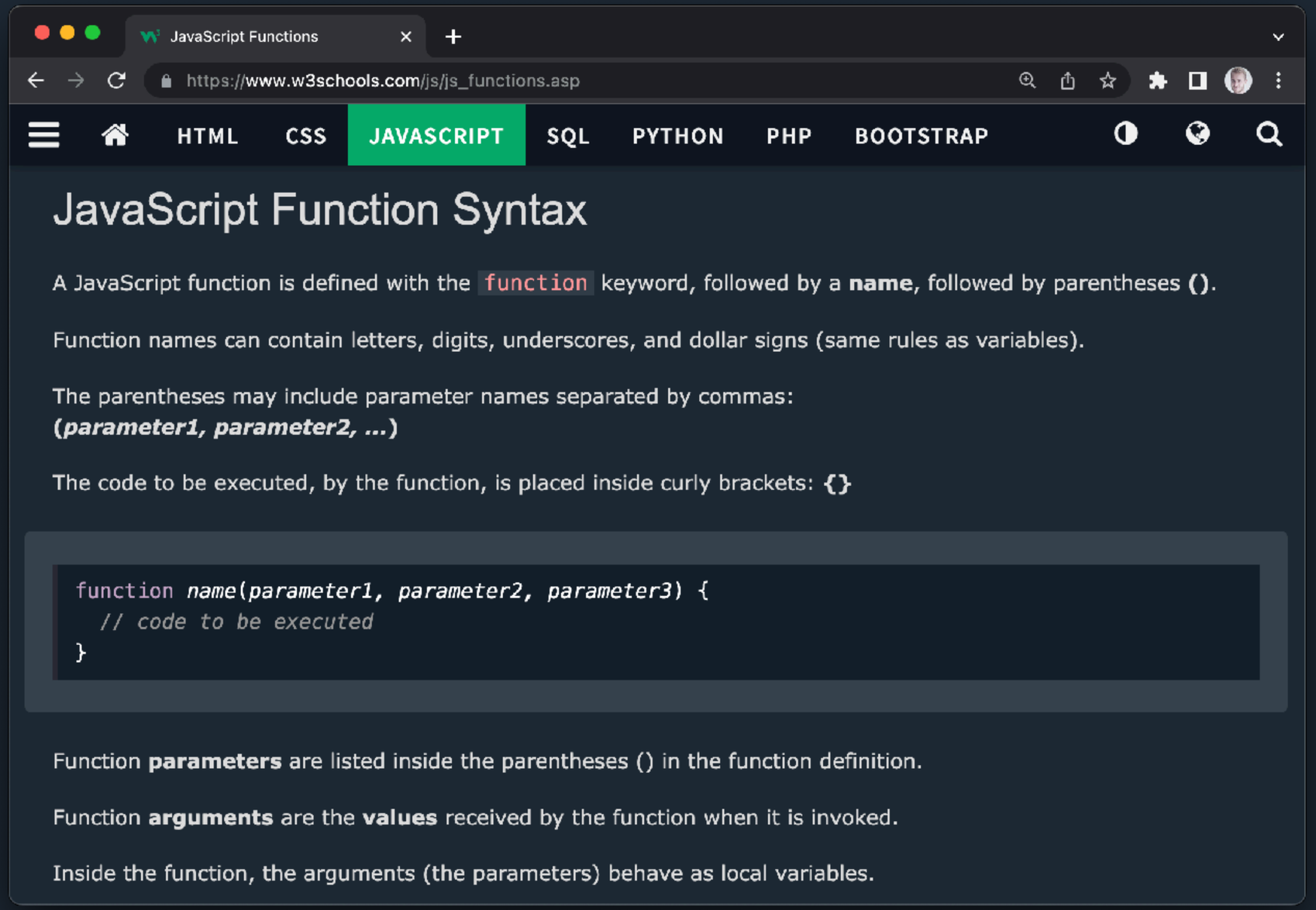
Functions

Function declaration

```
console.log("Hi Frontenders!");
console.log("Good job!");
console.log("I'm testing something!");
console.log("Hola");
```

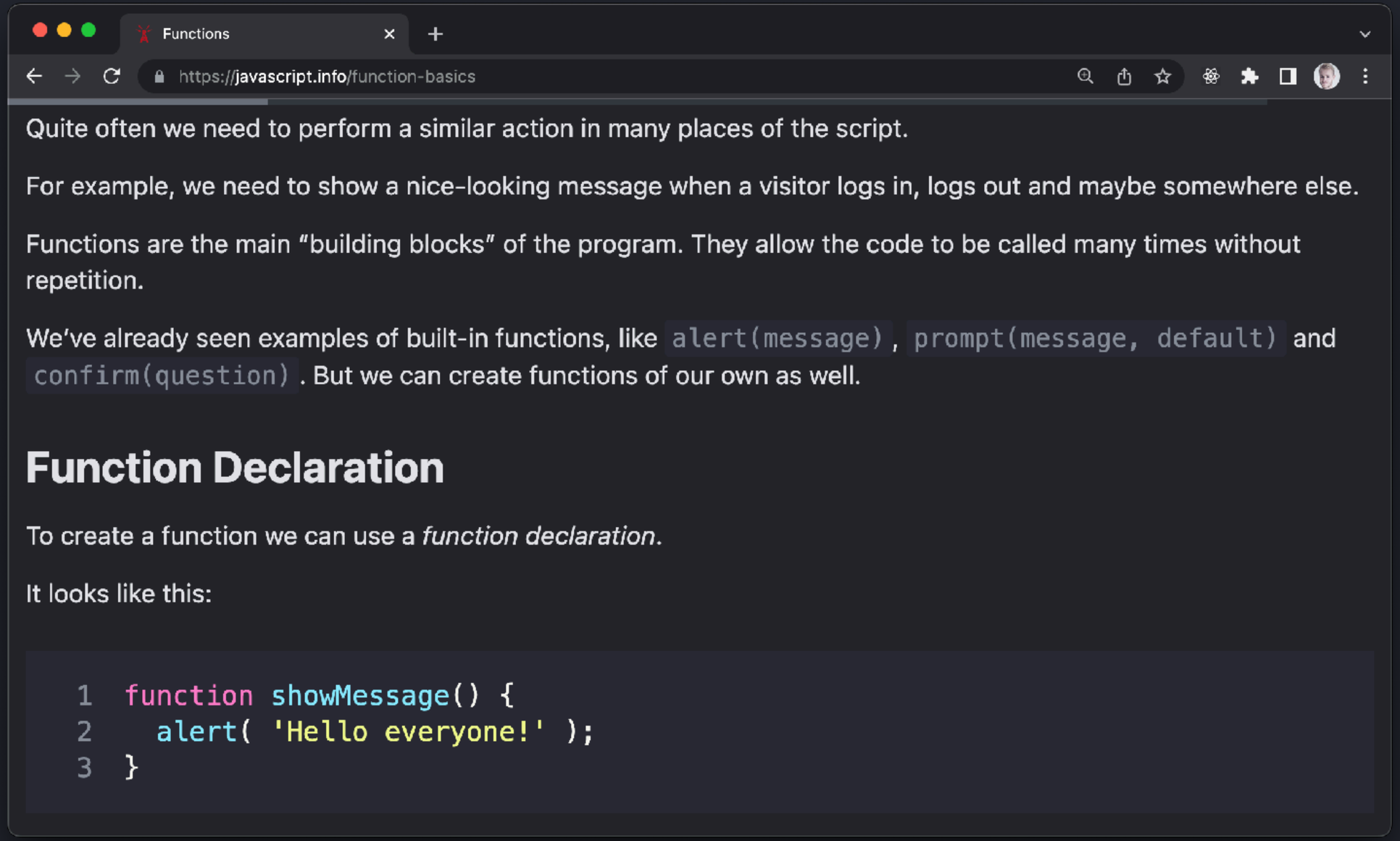
```
function log(message) {
  console.log(message);
}

log("Hi Frontenders!");
log("Good job!");
log("I'm testing something!");
log("Hola");
```

https://javascript.info/function-basics



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

JavaScript.info/Function-Basics



Functions

Function declaration

The name of the function

```
function showMessage() {
    alert("Hello everyone!");
}

showMessage();
```

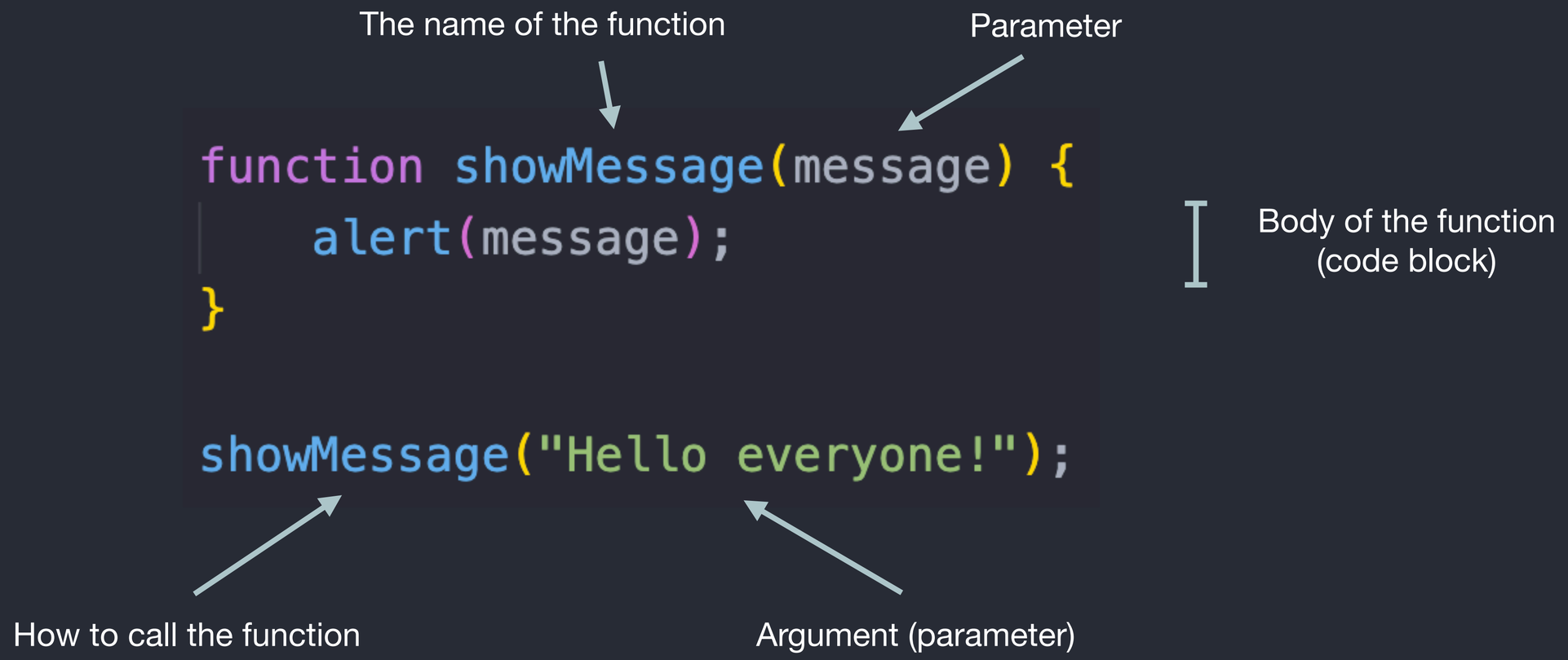
Body of the function
(code block)

How to call the function

https://javascript.info/function-basics

Functions

Function declaration



https://javascript.info/function-basics

Functions

Function declaration

Argument passed to the function

```
function showMessage(message) {
    alert(message);
}

showMessage("Hello everyone!");
showMessage("How are you?");
showMessage("Good, you?");
```

The function can be called as many times as you want
And with different argument values

"When a value is passed as a function parameter, it's also called an argument.

In other words, to put these terms straight:

- *A parameter is the variable listed inside the parentheses in the function declaration (it's a declaration time term).*
- *An argument is the value that is passed to the function when it is called (it's a call time term)."*

https://javascript.info/function-basics#parameters

JavaScript.info/function-basics#parameters

We can pass arbitrary data to functions using parameters.

In the example below, the function has two parameters: `from` and `text`.

```
1  function showMessage(from, text) { // parameters: from, text
2    alert(from + ': ' + text);
3  }
4
5  showMessage('Ann', 'Hello!'); // Ann: Hello! (*)
6  showMessage('Ann', "What's up?"); // Ann: What's up? (**)
```

Functions

Arrays & Loops

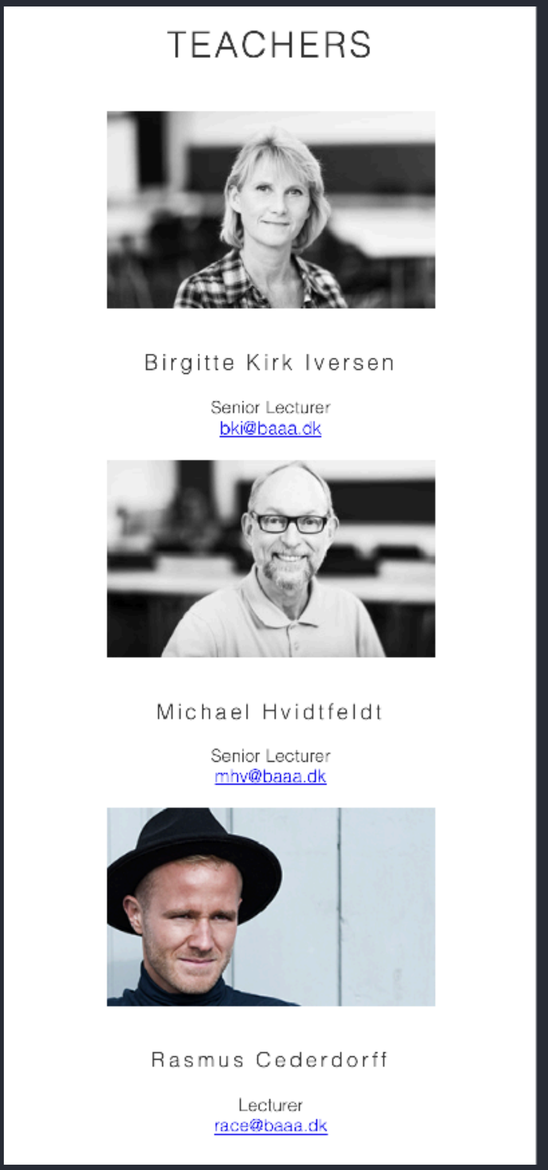
The name of the function

Parameters

```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML +=
      "<article>" +
      "<img src='" + teacher.img + "'>" +
      "<h3>" + teacher.name + "</h3>" +
      teacher.position + "<br>" +
      "<a href='mailto:" + teacher.mail + "'>" + teacher.mail + "</a>" +
      "</article>";
  }
}

appendTeachers(teachers);
```

Body of the function (code block)

How to call the function



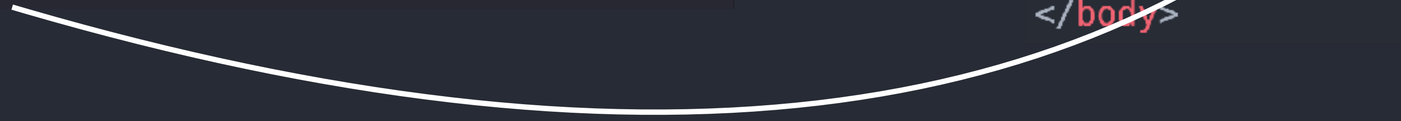
DOM Manipulation

Change the content of the website with JavaScript

DOM Manipulation

```
// declaring a variable with a value
let message = "Hi Frontenders!"

//accessing the variable and logging it to the console
console.log(message);

// appending the variable (the string) to the DOM element #content
document.querySelector("#content").innerHTML = message;
```

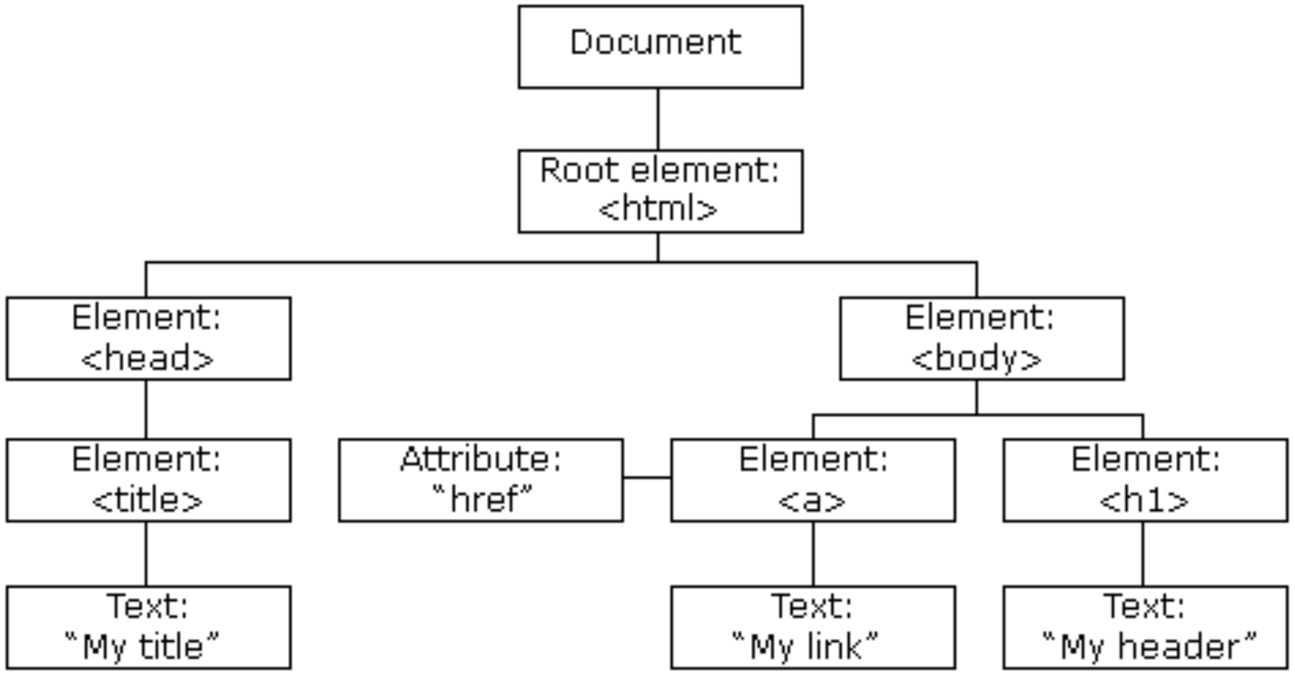
```
<body>
  <header>
    <h1>PROJECT TEMPLATE</h1>
  </header>
  <section id="content"></section>

  <!-- main js file -->
  <script src="js/main.js"></script>
</body>
```



JavaScript HTML DOM

index.html

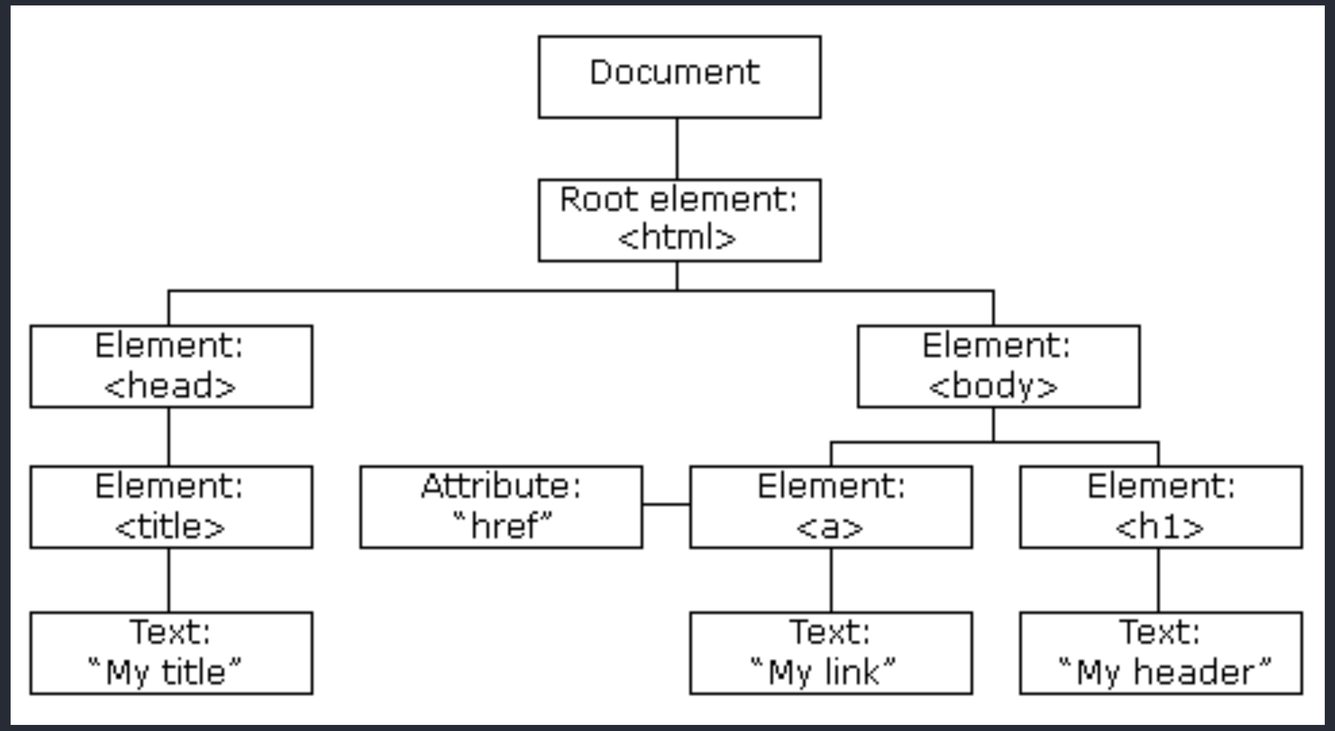
```html
<!DOCTYPE html>
<html>
  <head>
    <title>My title</title>
  </head>

  <body>
    <h1>My header</h1>
    <a href="https://cederdorff.com">My link</a>
  </body>
</html>
```



https://www.w3schools.com/js/js_htmldom.asp
https://javascript.info/dom-nodes
https://javascript.info/dom-navigation

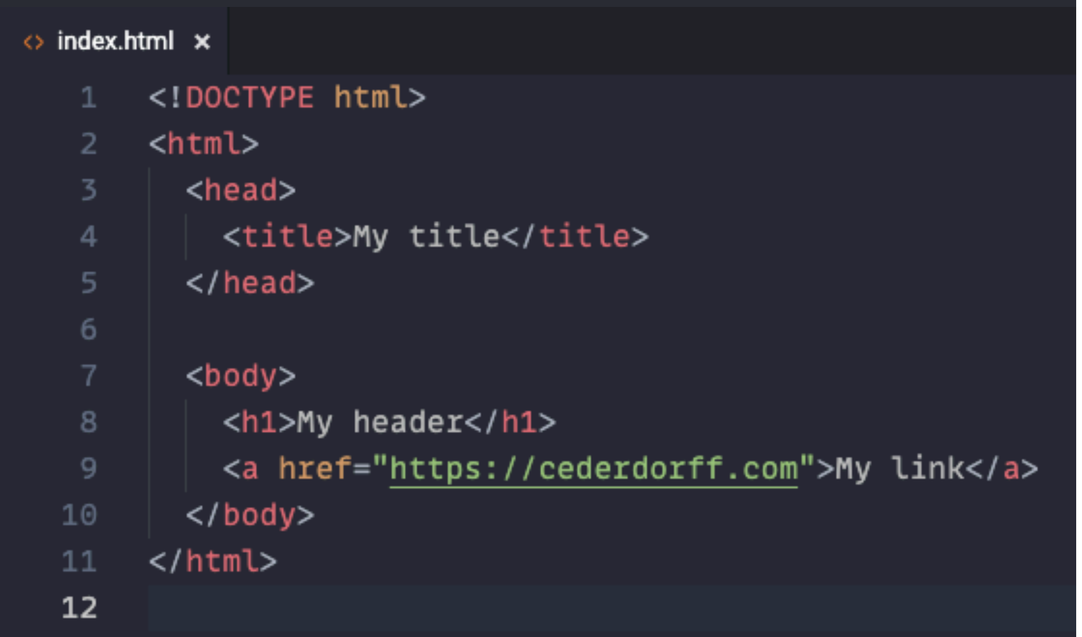
The HTML DOM (Document Object Model)

A model as a tree of Objects



- Object Model for HTML:
  - HTML elements as objects
  - Properties for all HTML elements
  - Methods for all HTML elements
  - Events for all HTML elements

https://www.w3schools.com/whatis/whatis_htmldom.asp



JavaScript HTML DOM

Document Object Model

The HTML document as an object

Gives us the power to create dynamic HTML and manipulate with the HTML (the DOM).

JavaScript can:

… change all the HTML elements in the page
… change all the HTML attributes in the page
… change all the CSS styles in the page
… remove existing HTML elements and attributes
… add new HTML elements and attributes
… react to all existing HTML events in the page
… create new HTML events in the page

https://www.w3schools.com/js/js_htmldom.asp
https://javascript.info/dom-nodes
https://javascript.info/dom-navigation

Searching the DOM: `getElement*` & `querySelector*`

```
<section id="elem">
    <article id="elem-content">Element</article>
</section>

<script>
    // get the element
    const element = document.getElementById('elem');
    // make its background red
    element.style.background = 'red';
    // get the elementContent
    const elementContent = document.querySelector('#elem-content');
    // change inner HTML
    elementContent.innerHTML = "<h2>Hi Web Developers!</h2>"
</script>
```

https://javascript.info/searching-elements-dom

Searching the DOM: `getElementsBy*`

```
<section id="elem">
    <article class="elem-content">Element</article>
    <article class="elem-content">Element</article>
    <article class="elem-content">Element</article>
</section>

<script>
    // get all elements matching the selector - returns an array
    const elements = document.getElementsByClassName('elem-content');
    // loop through all elements
    for (const element of elements) {
        element.innerHTML = "<h2>Hi Web Developers!</h2>";
    }
</script>
```

https://javascript.info/searching-elements-dom

Searching the DOM: `querySelectorAll`

```
<section id="elem">
    <article class="elem-content">Element</article>
    <article class="elem-content">Element</article>
    <article class="elem-content">Element</article>
</section>

<script>
    // get all elements matching the selector - returns an array
    const elements = document.querySelectorAll('.elem-content');
    // loop through all elements
    for (const element of elements) {
        element.innerHTML = "<h2>Hi Web Developers!</h2>";
    }
</script>
```

https://javascript.info/searching-elements-dom

Insert HTML Element: `insertAdjacentHTML`

```
const movieList = document.querySelector("#movie-list"); // Find container til film

// Byg HTML struktur dynamisk - template literal med ${} til at indsætte data
const movieHTML = /*html*/ `
  <article class="movie-card" tabindex="0">
    <img src="${movie.image}"
         alt="Poster of ${movie.title}"
         class="movie-poster" />
    <div class="movie-info">
      <h3>${movie.title} <span class="movie-year">(${movie.year})</span></h3>
      <p class="movie-genre">${movie.genre.join(", ")}</p>
      <p class="movie-rating">⭐ ${movie.rating}</p>
      <p class="movie-director"><strong>Director:</strong> ${movie.director}</p>
    </div>
  </article>
`;

// Tilføj movie card til DOM (HTML) - insertAdjacentHTML sætter HTML ind uden at overskrive
movieList.insertAdjacentHTML("beforeend", movieHTML);
```

https://developer.mozilla.org/en-US/docs/Web/API/Element/insertAdjacentHTML

JS HTML DOM

`getElement*` or `querySelector*`?

CSS Grid

CSS Grid is a two-dimensional layout system in CSS that allows you to arrange content in rows and columns with precision and flexibility.

CSS Layouts

Introduction to CSS layout

Normal flow

The display property

Flexbox

Grid

Floats

Positioning

Table layout

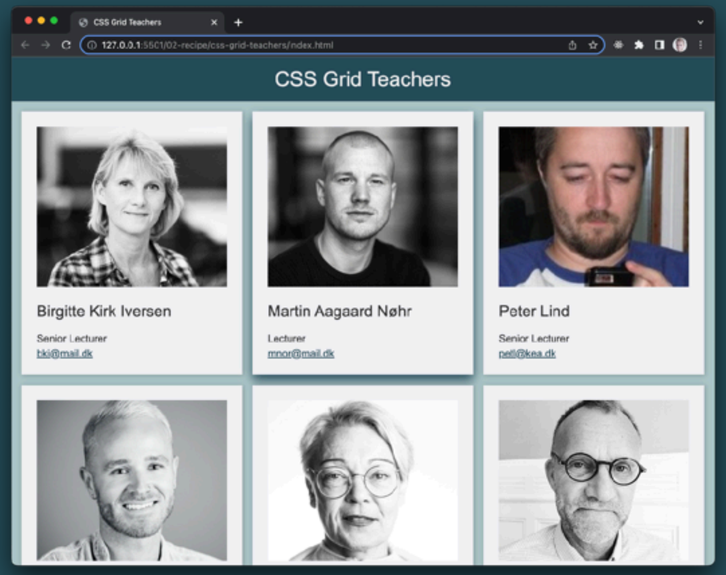
Multiple-column layout

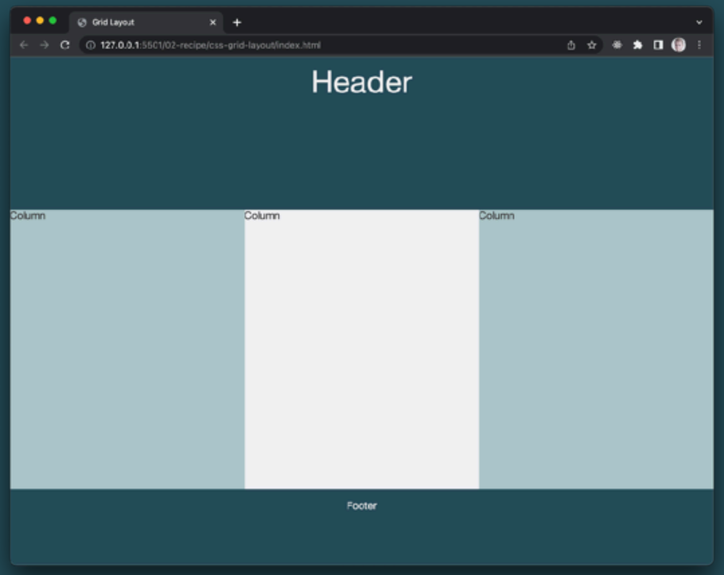
Grid & Flexbox

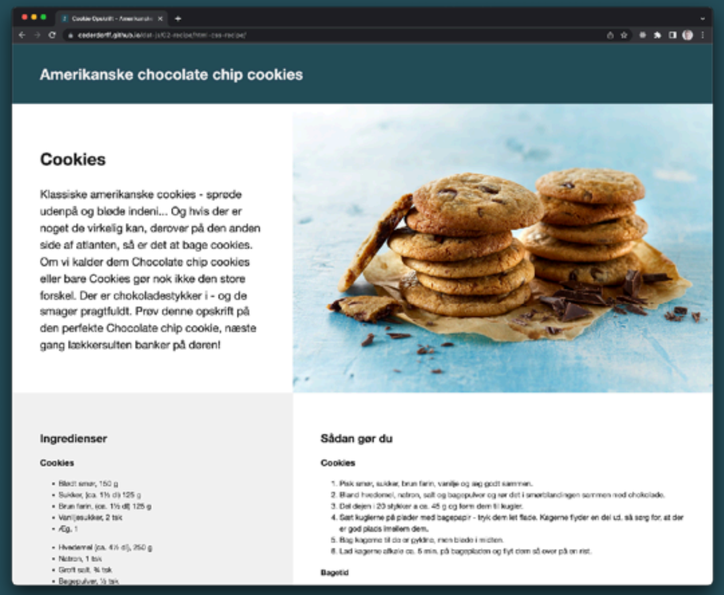
... two CSS layout models that can be used to create layouts with just a couple of CSS rules.

... responsive and flexible layouts.

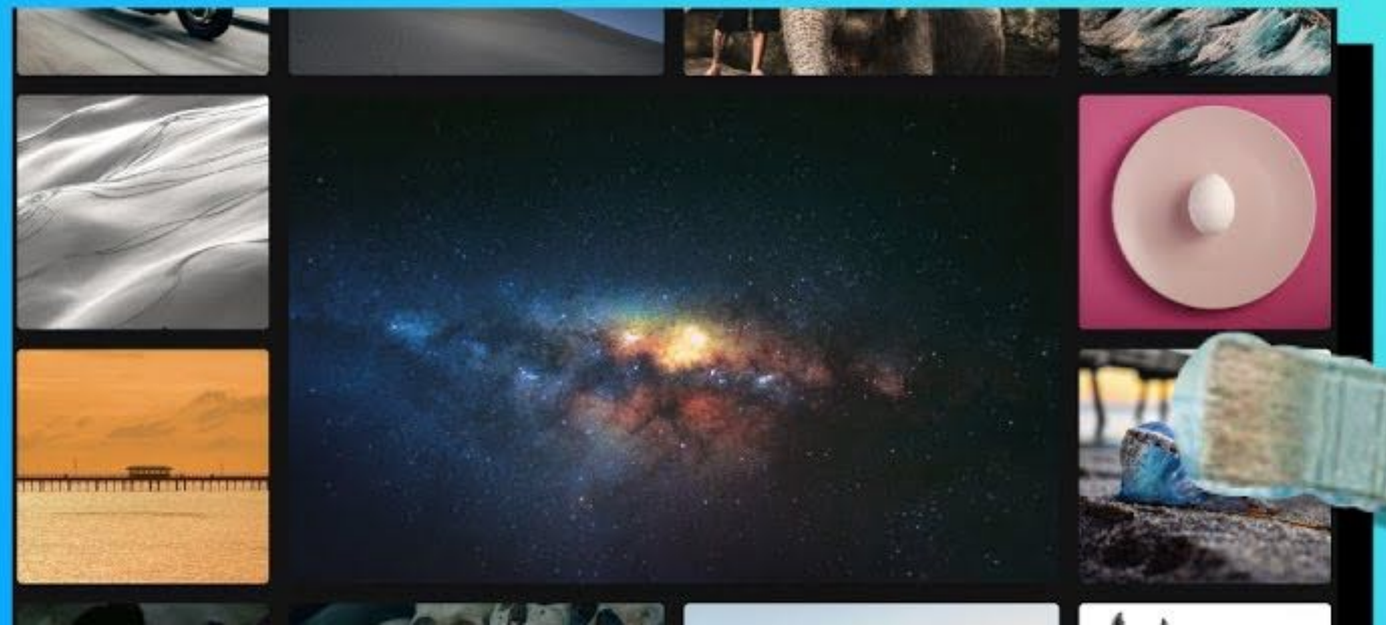
CSS Grids


CSS Grid Teachers
127.0.0.1:55C1/02-recipe/css-grid-teachers/index.html
CSS Grid Teachers
Birgitte Kirk Iversen
Senior Lecturer
bki@mail.dk
Martin Aagaard Nøhr
Lecturer
mnor@mail.dk
Peter Lind
Senior Lecturer
peli@kea.dk

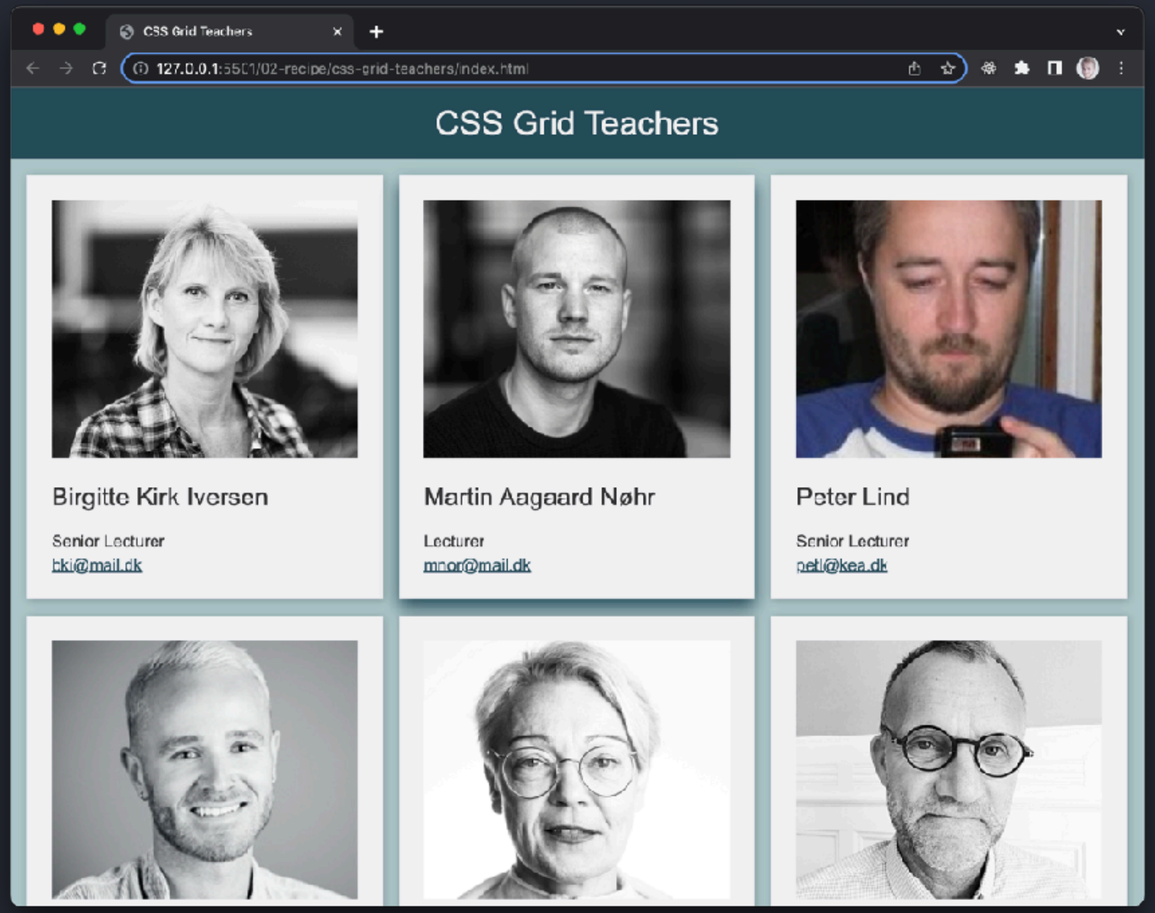

Grid Layout
127.0.0.1:55C1/02-recipe/css-grid-layout/index.html
Header
Column
Column
Column
Footer


Amerikanske chocolate chip cookies
Cookies
Klassiske amerikanske cookies - sprøde udenpå og bløde indeni... Og hvis der er noget de virkelig kan, derover på den anden side af atlanten, så er det at bage cookies. Om vi kalder dem Chocolate chip cookies eller bare Cookies gør nok ikke den store forskel. Der er chokoladestykker i - og de smager pragtfuldt. Prøv denne opskrift på den perfekte Chocolate chip cookie, næste gang lækkersulten banker på døren!
Ingredienser
Cookies
Sådan gør du
Cookies
Bagetid

The Joy of GRID



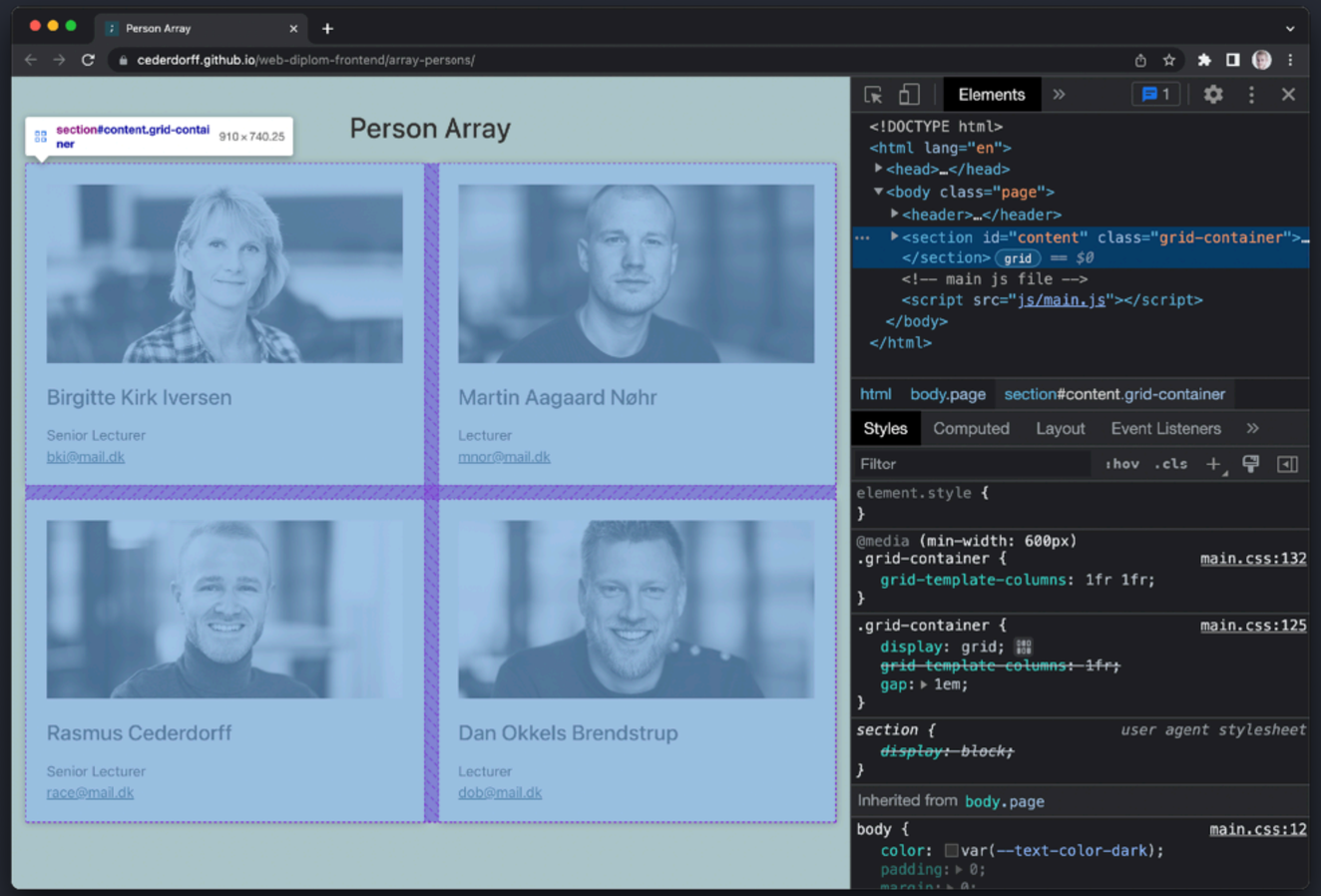
CSS Grid




```
<section class="grid-container">…
</section>
```

```
/* ---------- grid container styling ---------- */
.grid-container {
    display: grid;
    grid-template-columns: 1fr;
    gap: 1em;
    padding: 1em;
}

@media (min-width: 600px) {
    .grid-container {
        grid-template-columns: 1fr 1fr;
    }
}

@media (min-width: 992px) {
    .grid-container {
        grid-template-columns: 1fr 1fr 1fr;
    }
}
```

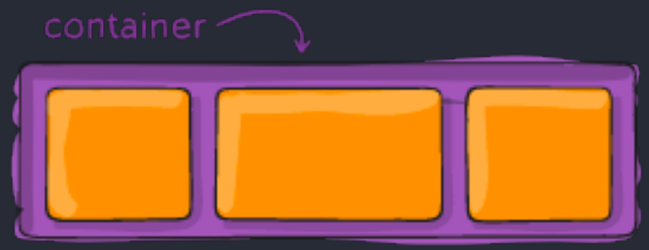
CSS Grid



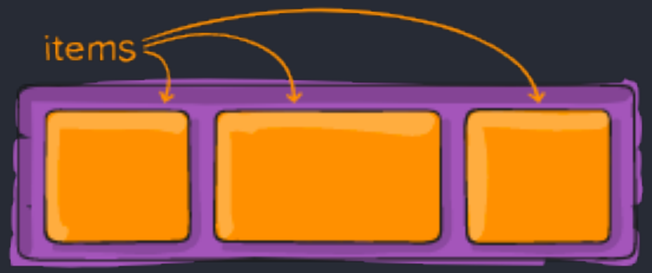

```html
<section id="content" class="grid-container"></section>
```

```css
/* ---------- grid container styling ---------- */
.grid-container {
    display: grid;
    grid-template-columns: 1fr;
    gap: 1em;
}

@media (min-width: 600px) {
    .grid-container {
        grid-template-columns: 1fr 1fr;
    }
}

@media (min-width: 992px) {
    .grid-container {
        grid-template-columns: 1fr 1fr 1fr;
    }
}
```

CSS Grid

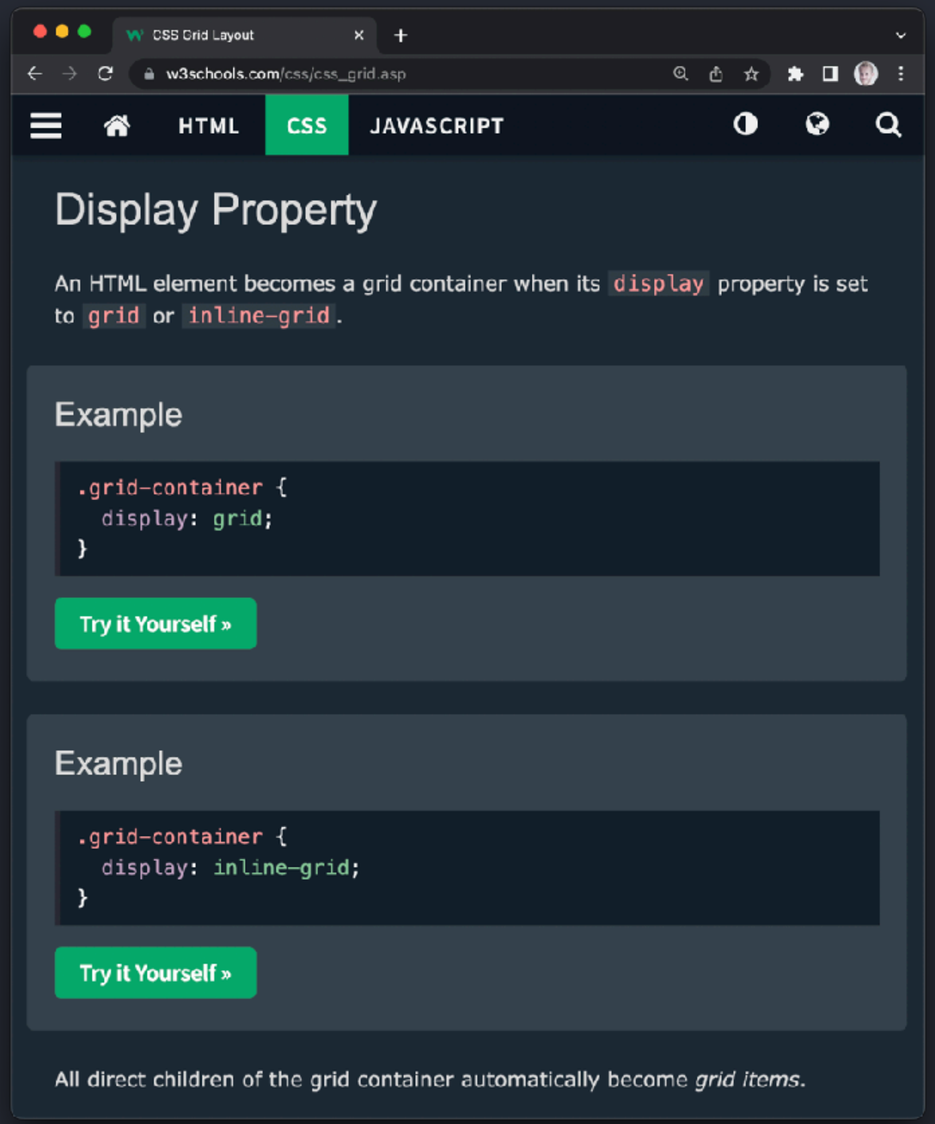



```css
/* ---------- grid container styling ---------- */
.grid-container {
    display: grid;
    grid-template-columns: 1fr;
    gap: 1em;
    padding: 1em;
}

@media (min-width: 600px) {
    .grid-container {
        grid-template-columns: 1fr 1fr;
    }
}

@media (min-width: 992px) {
    .grid-container {
        grid-template-columns: 1fr 1fr 1fr;
    }
}
```




```css
.grid-item {
    box-shadow: 1px 1px 8px rgba(0, 0, 0, 0.25);
    padding: 1.5em;
    background-color: #f1f1f4;
    transition: 0.5s;
    animation: fadeIn 0.5s;
}

.grid-item:hover {
    box-shadow: 0 8px 16px 0 rgb(38, 76, 89);
}

.grid-item img {
    width: 100%;
    height: 250px;
    object-fit: cover;
}

.grid-item p {
    margin: 0.3em 0;
```

Grid Container



Example

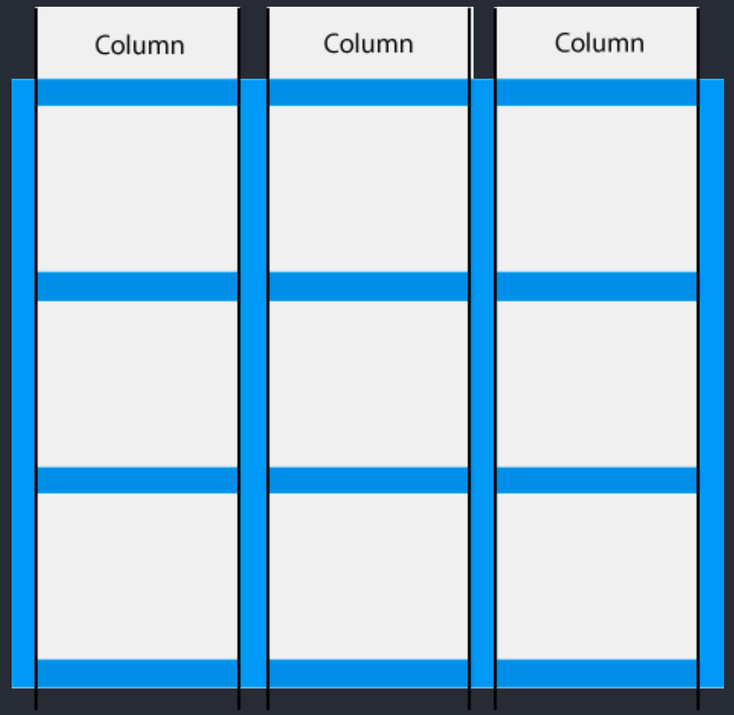
```
<div class="grid-container">
  <div class="grid-item">1</div>
  <div class="grid-item">2</div>
  <div class="grid-item">3</div>
  <div class="grid-item">4</div>
  <div class="grid-item">5</div>
  <div class="grid-item">6</div>
  <div class="grid-item">7</div>
  <div class="grid-item">8</div>
  <div class="grid-item">9</div>
</div>
```

1	2	3
4	5	6
7	8	9

https://www.w3schools.com/css/css_grid.asp

Grid Columns

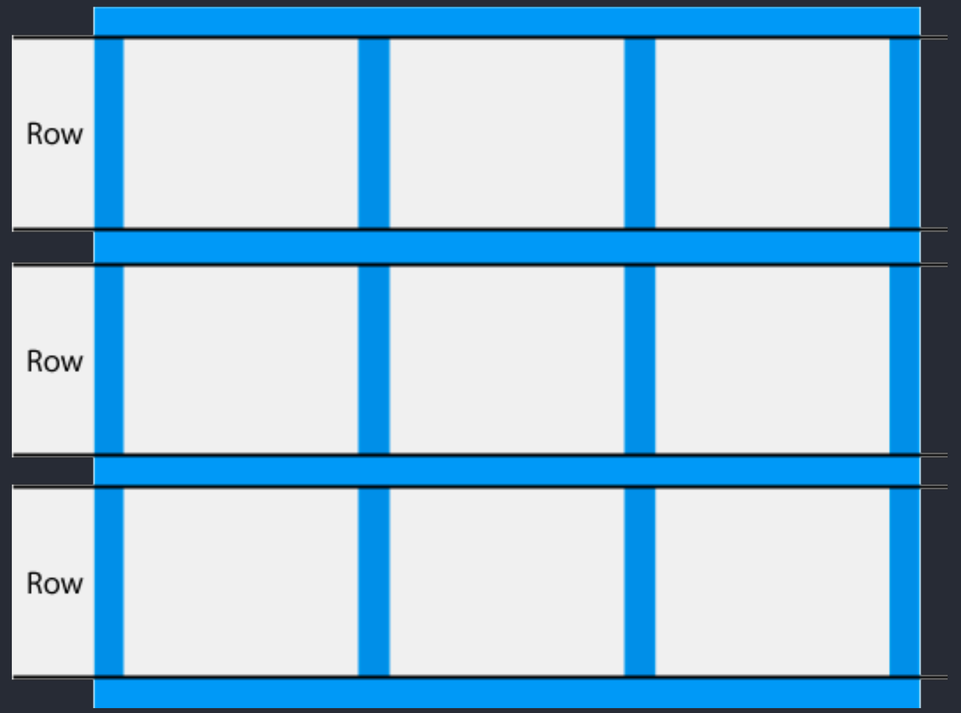
Vertical Lines




```
.grid-container {
  display: grid;
  grid-template-columns: auto auto auto auto;
}
```

Grid Rows

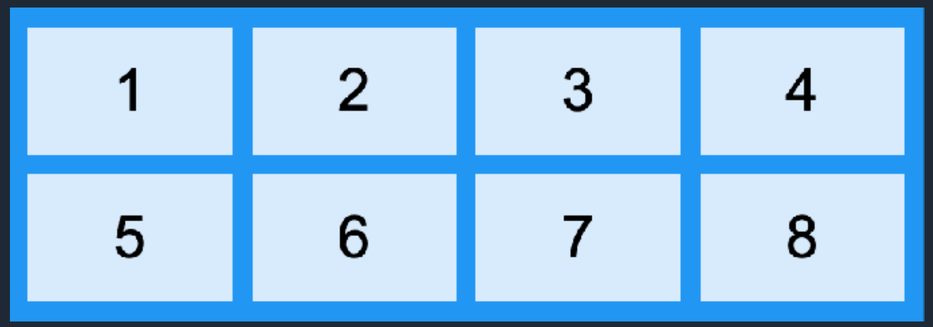
Horisontal Lines




```
.grid-container {
  display: grid;
  grid-template-rows: 100px 300px;
}
```

https://www.w3schools.com/css/css_grid.asp

grid-template-columns



The grid-template-columns Property

The `grid-template-columns` property defines the number of columns in your grid layout, and it can define the width of each column.

Make a grid with 4 columns:

```
.grid-container {
  display: grid;
  grid-template-columns: auto auto auto auto;
}
```

Set a size for the 4 columns:

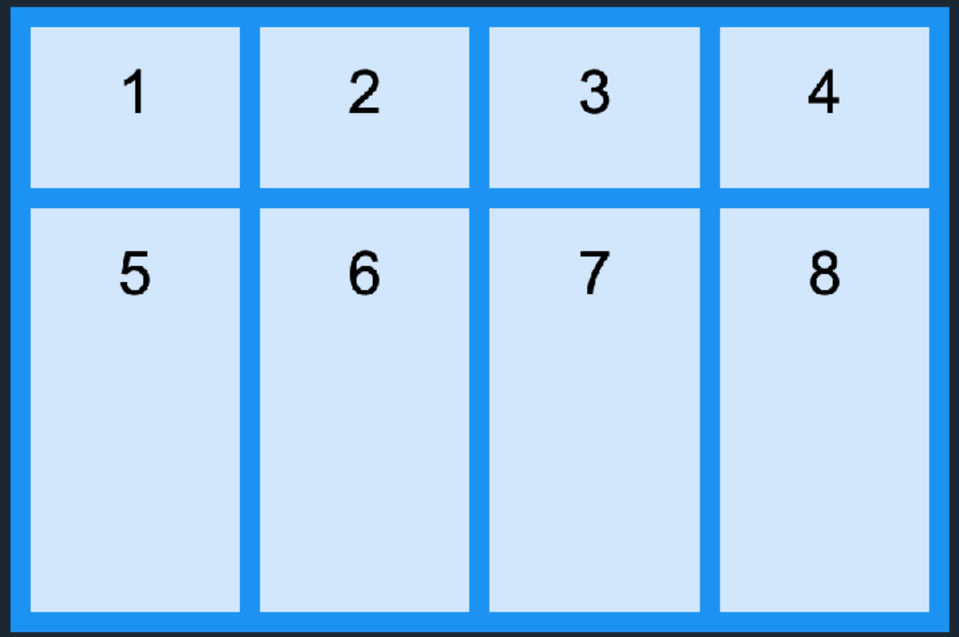
```
.grid-container {
  display: grid;
  grid-template-columns: 80px 200px auto 40px;
}
```

https://www.w3schools.com/css/css_grid.asp

grid-template-rows

The grid-template-rows Property

The `grid-template-rows` property defines the height of each row.



The value is a space-separated-list, where each value defines the height of the respective row:

```
.grid-container {
  display: grid;
  grid-template-rows: 80px 200px;
}
```

https://www.w3schools.com/css/css_grid.asp

CSS Grid Container

w3schools.com/css/css_grid_container.asp

HTML CSS JAVASCRIPT SQL

The justify-content Property

The `justify-content` property is used to align the whole grid inside the container.

Note: The grid's total width has to be less than the container's width for the `justify-content` property to have any effect.

Example

```
.grid-container {
  display: grid;
  justify-content: space-evenly;
}
```

Try it Yourself »

Example

```
.grid-container {
  display: grid;
  justify-content: space-around;
}
```

Try it Yourself »

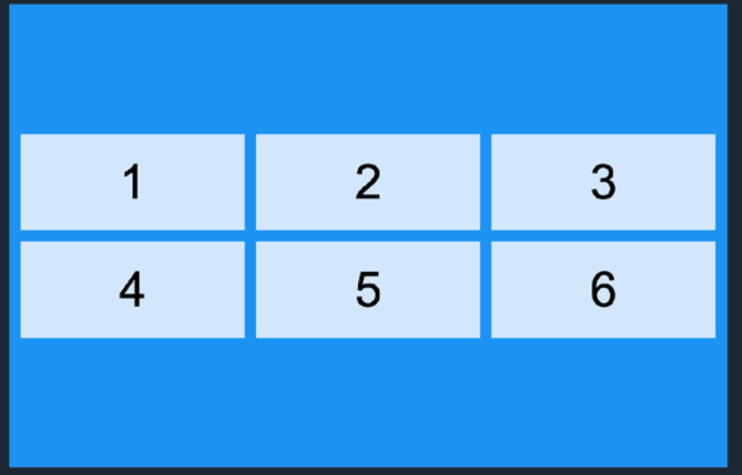
CSS Grid Container

w3schools.com/css/css_grid_container.asp

HTML CSS JAVASCRIPT SQL

The align-content Property

The `align-content` property is used to *vertically* align the whole grid inside the container.



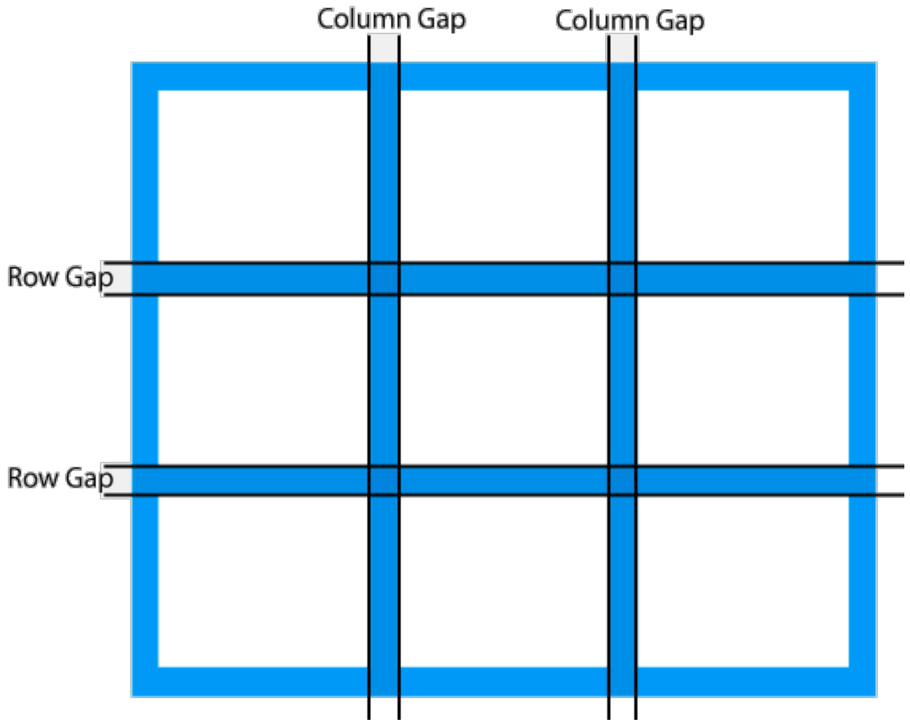
Note: The grid's total height has to be less than the container's height for the `align-content` property to have any effect.

Example

```
.grid-container {
  display: grid;
  height: 400px;
  align-content: center;
}
```

Try it Yourself »

Grid Gap



The `column-gap` property sets the gap between the columns:

```
.grid-container {
  display: grid;
  column-gap: 50px;
}
```

The `row-gap` property sets the gap between the rows:

```
.grid-container {
  display: grid;
  row-gap: 50px;
}
```

The `gap` property is a shorthand property for the `row-gap` and the `column-gap` properties:

```
.grid-container {
  display: grid;
  gap: 50px 100px;
}
```

The `gap` property can also be used to set both the row gap and the column gap in one value:

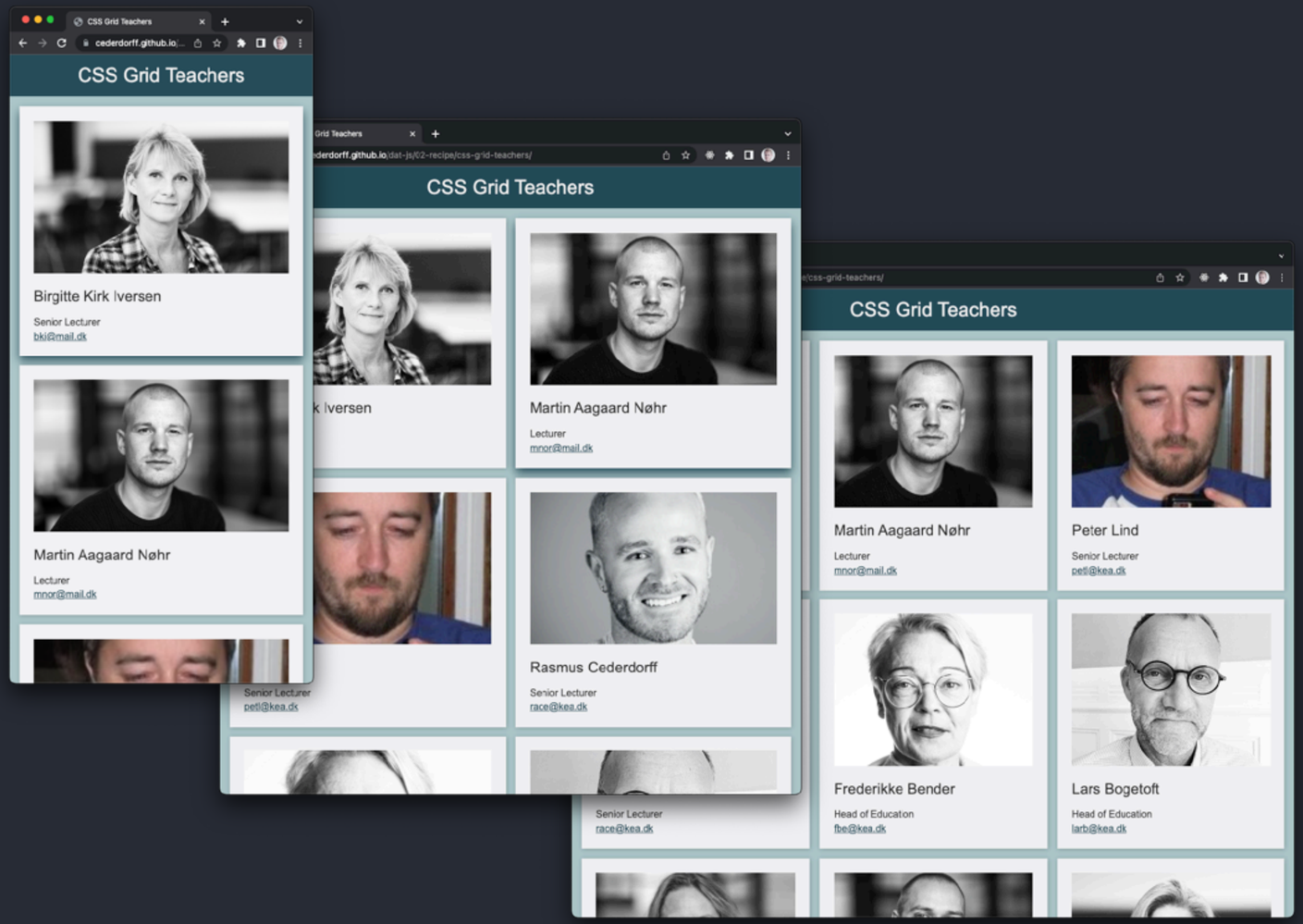
```
.grid-container {
  display: grid;
  gap: 50px;
}
```

https://www.w3schools.com/css/css_grid.asp

100 seconds of CSS

GRID LAYOUT

CSS Grid & media queries



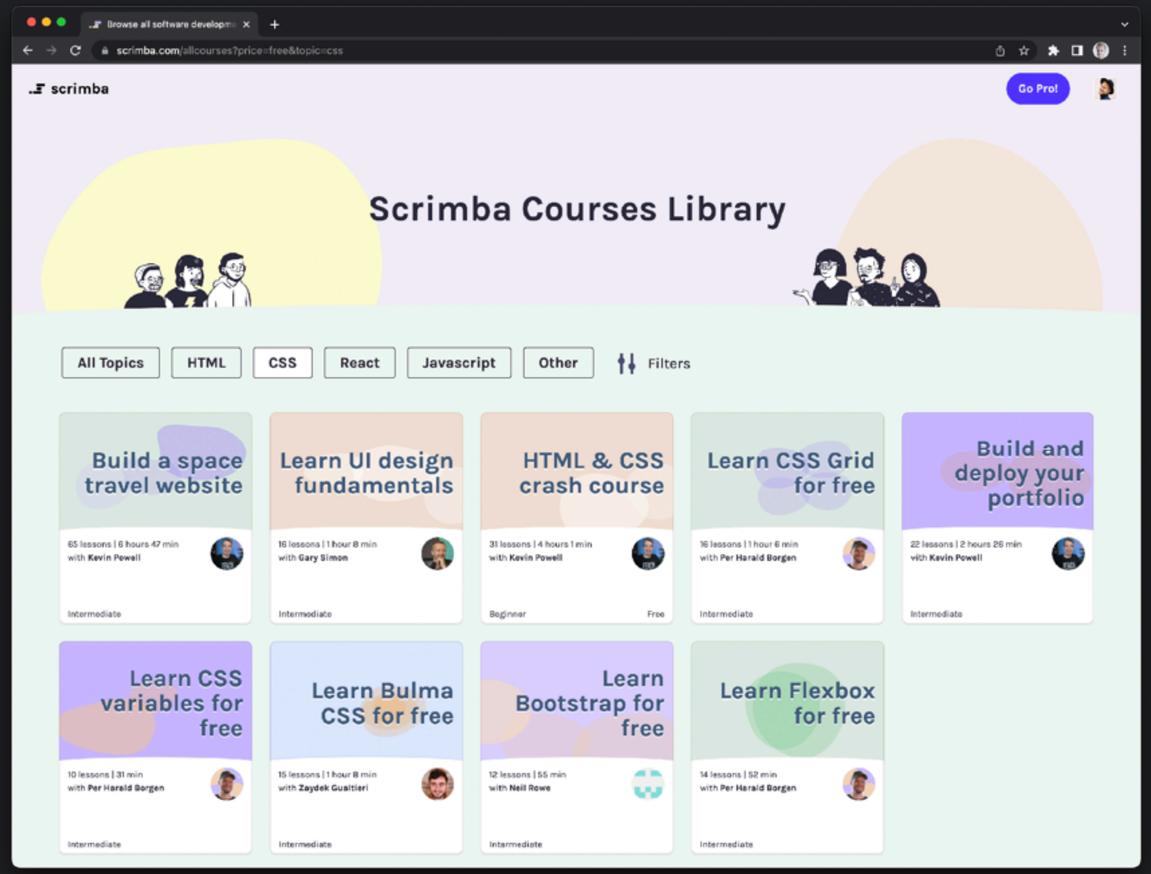

```css
/* ---------- grid container styling ---------- */
.grid-container {
    display: grid;
    grid-template-columns: 1fr;
    gap: 1em;
    padding: 1em;
}

@media (min-width: 600px) {
    .grid-container {
        grid-template-columns: 1fr 1fr;
    }
}

@media (min-width: 992px) {
    .grid-container {
        grid-template-columns: 1fr 1fr 1fr;
    }
}
```

css-grid-teachers

Scrimba Courses

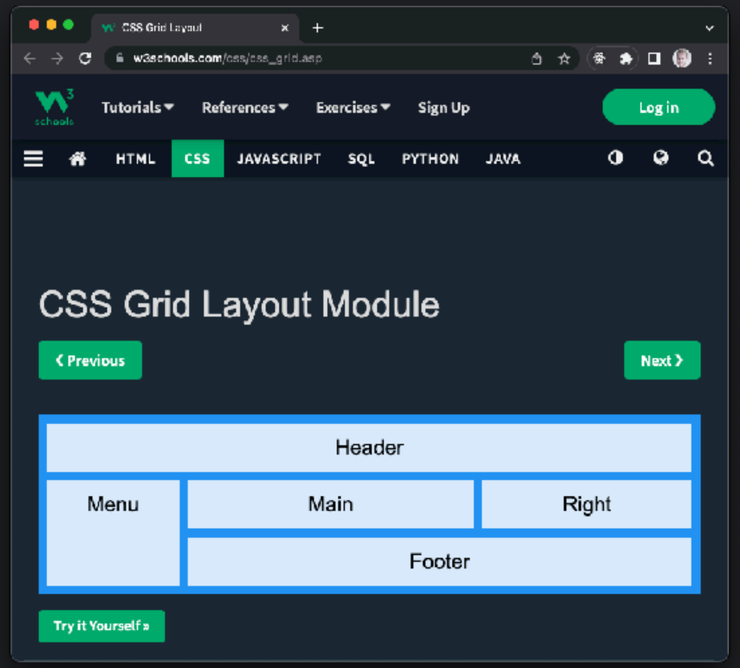
Learn CSS Grid for free: https://scrimba.com/learn/cssgrid

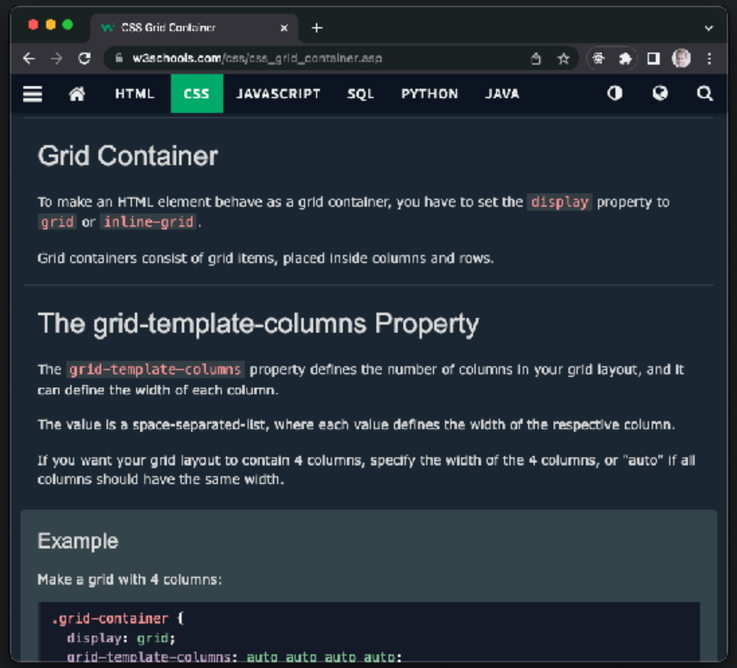
Learn Flexbox for free: https://scrimba.com/learn/flexbox

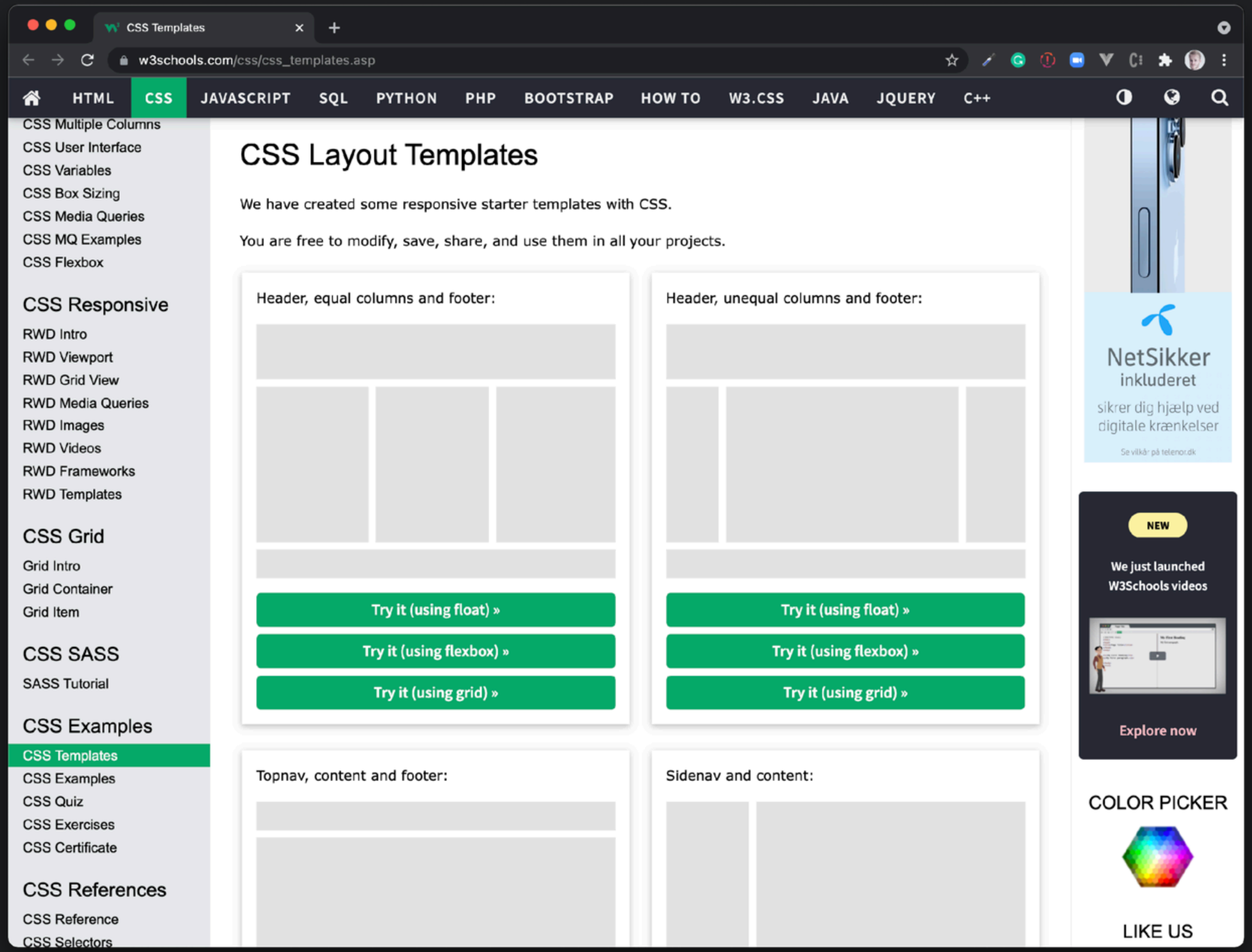


https://scrimba.com/allcourses

CSS Grid Layout Module







https://www.w3schools.com/css/css_templates.asp

Fetch

Fetch is a built-in JavaScript API (functionality) for requesting data from a server.
Fetch = get data from a server.

Client-Server Model

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

CLIENT

SERVER

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.

`getCharacters` fetches a list of characters from a JSON data source, parses the JSON to JS and returns the data.

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

fetch(...)

Hent og send data med JavaScript

- `fetch()` bruges til at lave HTTP requests – fx hente data fra en API eller sende data til en server.
- Giver adgang til eksterne data-kilder (JSON, REST API, osv.)

Kort fortalt (flow af kode):

`fetch()` → henter data → parser JSON → giver dig JavaScript-objekter, du kan bruge i din app.

```
// Fetch movies from JSON file – asynkron funktion d
async function getMovies() {
  // Hent data fra URL – await venter på svar før vi
  const response = await fetch("https://raw.githubus
  // Parser JSON til JS array og gem i global variab
  const data = await response.json();
  // Gør noget med data
  console.log(data);
}
```

fetch(...)

HTTP Requests i JavaScript

- `fetch()` returnerer et Promise – et løfte om data, der kommer senere.
- `await` fortæller JavaScript, at den skal vente på et svar, før den går videre i koden.
- Når du bruger `await`, skal funktionen være asynkron – det betyder, du skal skrive `async` foran funktionen.

Asynkront betyder: JavaScript kan lave andet, mens den venter på data.

```
// Fetch movies from JSON file – asynkron funktion d
async function getMovies() {
  // Hent data fra URL – await venter på svar før vi
  const response = await fetch("https://raw.githubus
  // Parser JSON til JS array og gem i global variab
  const data = await response.json();
  // Gør noget med data
  console.log(data);
}
```

fetch(...)

... get & post data from and to a data source
... can perform network requests to a server

```
let promise = fetch(url, [options])
```

- `url` – the URL to access.
- `options` – optional parameters: method, headers etc.

Without `options`, this is a simple GET request, downloading the contents of the `url`.

https://javascript.info/fetch

Fetch, fetch, fetch

WEB APP

API

DATA SOURCE

FETCH(...)

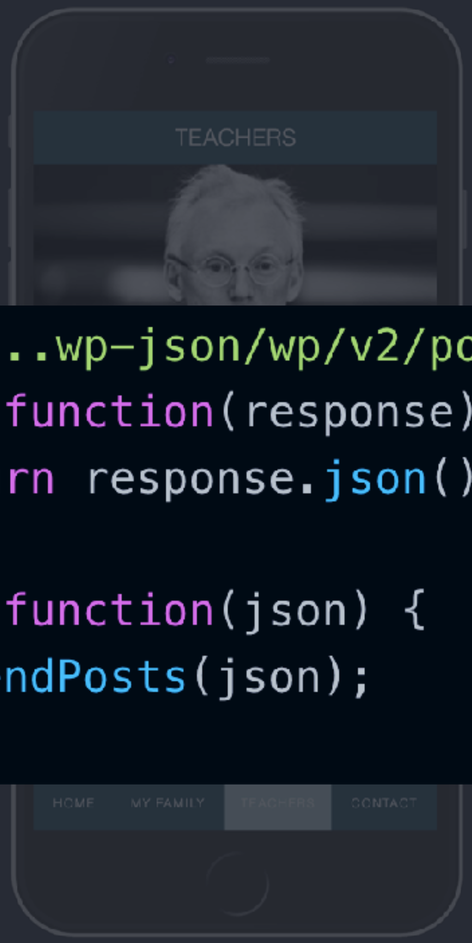
REQUEST

JSON DATA

RESPONSE

Fetch

WEB APP




```
fetch("...wp-json/wp/v2/posts")
  .then(function(response) {
    return response.json();
  })
  .then(function(json) {
    appendPosts(json);
  });
```

REQUEST (fetch)

RESPONSE (then)

DATA SOURCE

```js
/*
Fetches json data from the file persons.json
*/
fetch('json/persons.json')
  .then(function (response) {
    return response.json();
  })
  .then(function (jsonData) {
    console.log(jsonData);
    appendPersons(jsonData)
  });

/*
Appends json data to the DOM
*/
function appendPersons(persons) {
  let htmlTemplate = "";
  for (let person of persons) {
    htmlTemplate += /*html*/`
      <article>
        <img src="${person.img}">
        <h4>${person.name}</h4>
        <p>${person.age} years old</p>
        <p>Hair color: ${person.hairColor}</p>
        <p>Relation: ${person.relation}</p>
      </article>
    `;
  }
  document.querySelector("#persons").innerHTML = htmlTemplate;
}
```

```json
[
  {
    "name": "Peter Madsen",
    "age": 52,
    "hairColor": "blonde",
    "relation": "dad",
    "img": "img/dad.jpg"
  },
  {
    "name": "Ane Madsen",
    "age": 51,
    "hairColor": "brown",
    "relation": "mom",
    "img": "img/ane.jpg"
  },
  {
    "name": "Rasmus Madsen",
    "age": 28,
    "hairColor": "blonde",
    "relation": "brother",
    "img": "img/IMG_0526_kvadrat.jpg"
  },
  {
    "name": "Mie Madsen",
    "age": 25,
    "hairColor": "brown",
    "relation": "blonde",
    "img": "img/mie.jpg"
  },
  {
    "name": "Mads Madsen",
    "age": 18,
    "hairColor": "dark",
    "relation": "blonde",
    "img": "img/mads.jpg"
  },
  {
    "name": "Jens Madsen",
    "age": 14,
    "hairColor": "blonde",
    "relation": "uncle",
    "img": "img/jenspeter.jpg"
  }
]
```

fetch-family-members

```js
/*
Fetches json data from the file persons.json
*/
fetch('json/persons.json')
  .then(function (response) {
    return response.json();
  })
  .then(function (jsonData) {
    console.log(jsonData);
    appendPersons(jsonData)
  });

/*
Appends json data to the DOM
*/
function appendPersons(persons) {
  let htmlTemplate = "";
  for (let person of persons) {
    htmlTemplate += /*html*/`
      <article>
        <img src="${person.img}">
        <h4>${person.name}</h4>
        <p>${person.age} years old</p>
        <p>Hair color: ${person.hairColor}</p>
        <p>Relation: ${person.relation}</p>
      </article>
    `;
  }
  document.querySelector("#persons").innerHTML = htmlTemplate;
}
```

REQUEST (fetch)

RESPONSE (then)

```json
[
  {
    "name": "Peter Madsen",
    "age": 52,
    "hairColor": "blonde",
    "relation": "dad",
    "img": "img/dad.jpg"
  },
  {
    "name": "Ane Madsen",
    "age": 51,
    "hairColor": "brown",
    "relation": "mom",
    "img": "img/ane.jpg"
  },
  {
    "name": "Rasmus Madsen",
    "age": 28,
    "hairColor": "blonde",
    "relation": "brother",
    "img": "img/IMG_0526_kvadrat.jpg"
  },
  {
    "name": "Mie Madsen",
    "age": 25,
    "hairColor": "brown",
    "relation": "blonde",
    "img": "img/mie.jpg"
  },
  {
    "name": "Mads Madsen",
    "age": 18,
    "hairColor": "dark",
    "relation": "blonde",
    "img": "img/mads.jpg"
  },
  {
    "name": "Jens Madsen",
    "age": 14,
    "hairColor": "blonde",
    "relation": "uncle",
    "img": "img/jenspeter.jpg"
  }
]
```

fetch-family-members

```js
/*
Fetches json data from the file persons.json
*/
fetch('json/persons.json')
  .then(function (response) {
    return response.json();
  })
  .then(function (jsonData) {
    console.log(jsonData);
    appendPersons(jsonData)
  });

/*
Appends json data to the DOM
*/
function appendPersons(persons) {
  let htmlTemplate = "";
  for (let person of persons) {
    htmlTemplate += /*html*/`
      <article>
        <img src="${person.img}">
        <h4>${person.name}</h4>
        <p>${person.age} years old</p>
        <p>Hair color: ${person.hairColor}</p>
        <p>Relation: ${person.relation}</p>
      </article>
    `;
  }
  document.querySelector("#persons").innerHTML = htmlTemplate;
}
```

```json
},
{
  "name": "Rasmus Madsen",
  "age": 28,
  "hairColor": "blonde",
  "relation": "brother",
  "img": "img/IMG_0526_kvadrat.jpg"
},
{
  "name": "Mie Madsen",
  "age": 25,
  "hairColor": "brown",
  "relation": "blonde",
  "img": "img/mie.jpg"
},
{
  "name": "Mads Madsen",
  "age": 18,
  "hairColor": "dark",
  "relation": "blonde",
  "img": "img/mads.jpg"
},
{
```

fetch-family-members

request (fetch)

response (JSON)

app.js — web-diplom-frontend

app.js

fetch-persons-grid > app.js > ...

```
let persons = []; // global variable

async function getPerson() {
    const response = await fetch(
        "https://raw.githubusercontent.com/cederdorff/web-diplom-frontend/main/_data/persons.json"
    ); // fetch request - fetch data from a given url
    persons = await response.json(); // setting global variable with fetched data
    displayPersons(persons); // calling displayPersons with persons as parameter
}

function displayPersons(listOfPersons) {
    let html = ""; // variable to store html
    //loop through all persons and create an article with content for each
    for (const person of listOfPersons) {
        html += /*html*/ `
            <article>
                <img src="${person.img}">
                <h2>${person.name}</h2>
                <p>${person.title}</p>
                <a href="mailto:${person.mail}">${person.mail}</a>
            </article>
        `; // generate and save html for every person in html variable
    }
    // set grid container content with person <article> elements
    // saved in html
    document.querySelector("#content").innerHTML = html;
}

getPerson(); // execute get persons to start the fun
```

main ⊗ 0 ⚠ 0 Ln 1, Col 1 Spaces: 4 UTF-8 LF {} JavaScript Go Live ✓ Prettier

https://raw.githubusercontent.c

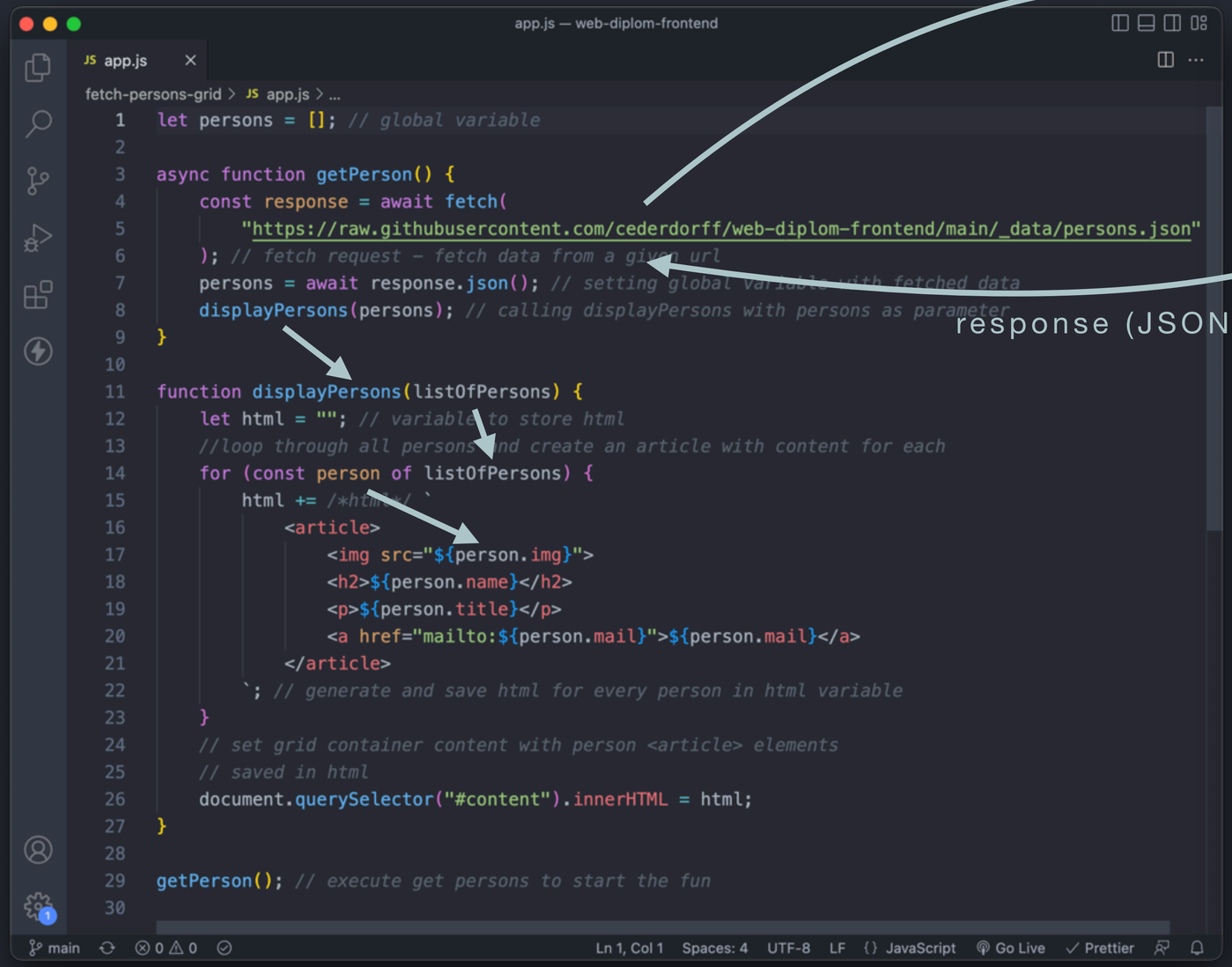
raw.githubusercontent.com/cederdorff/web-diplom-frontend/mai...

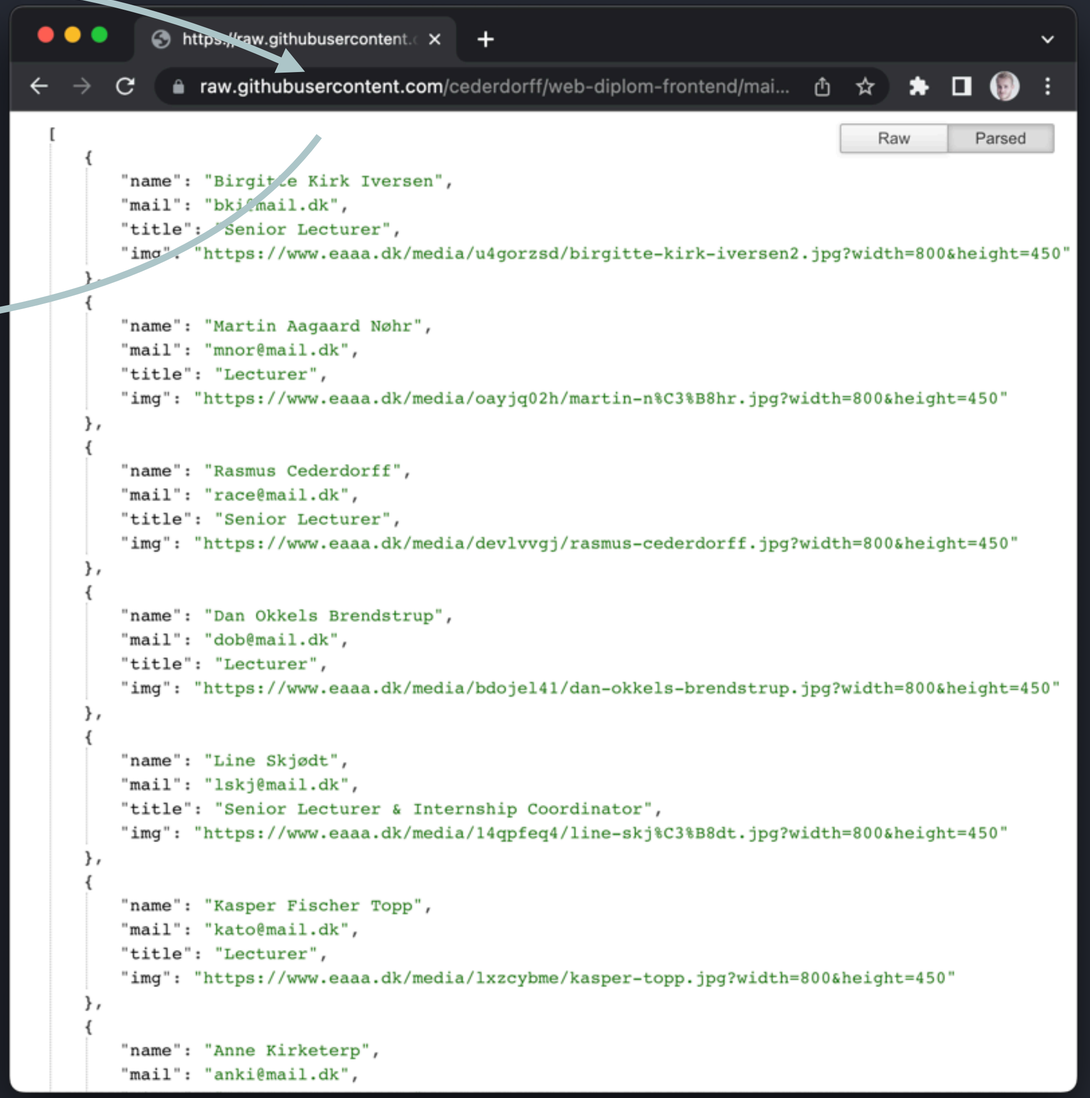
Raw Parsed


```
[
    {
        "name": "Birgitte Kirk Iversen",
        "mail": "bkj@mail.dk",
        "title": "Senior Lecturer",
        "img": "https://www.eaaa.dk/media/u4gorzsd/birgitte-kirk-iversen2.jpg?width=800&height=450"
    },
    {
        "name": "Martin Aagaard Nøhr",
        "mail": "mnor@mail.dk",
        "title": "Lecturer",
        "img": "https://www.eaaa.dk/media/oayjq02h/martin-n%C3%B8hr.jpg?width=800&height=450"
    },
    {
        "name": "Rasmus Cederdorff",
        "mail": "race@mail.dk",
        "title": "Senior Lecturer",
        "img": "https://www.eaaa.dk/media/devlvvgj/rasmus-cederdorff.jpg?width=800&height=450"
    },
    {
        "name": "Dan Okkels Brendstrup",
        "mail": "dob@mail.dk",
        "title": "Lecturer",
        "img": "https://www.eaaa.dk/media/bdojel41/dan-okkels-brendstrup.jpg?width=800&height=450"
    },
    {
        "name": "Line Skjødt",
        "mail": "lskj@mail.dk",
        "title": "Senior Lecturer & Internship Coordinator",
        "img": "https://www.eaaa.dk/media/14qpfeq4/line-skj%C3%B8dt.jpg?width=800&height=450"
    },
    {
        "name": "Kasper Fischer Topp",
        "mail": "kato@mail.dk",
        "title": "Lecturer",
        "img": "https://www.eaaa.dk/media/lxzcybme/kasper-topp.jpg?width=800&height=450"
    },
    {
        "name": "Anne Kirketerp",
        "mail": "anki@mail.dk",
```

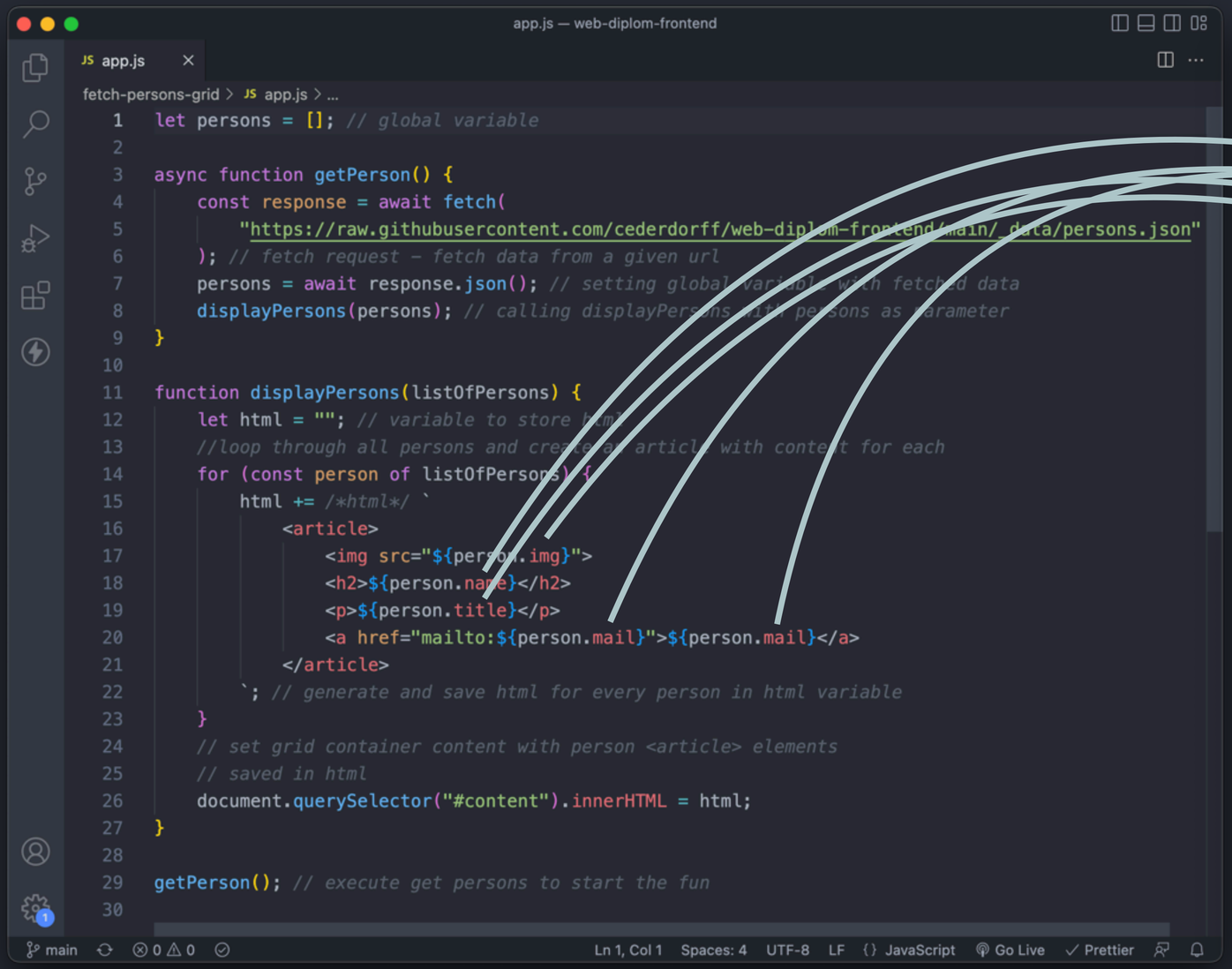
request (fetch)

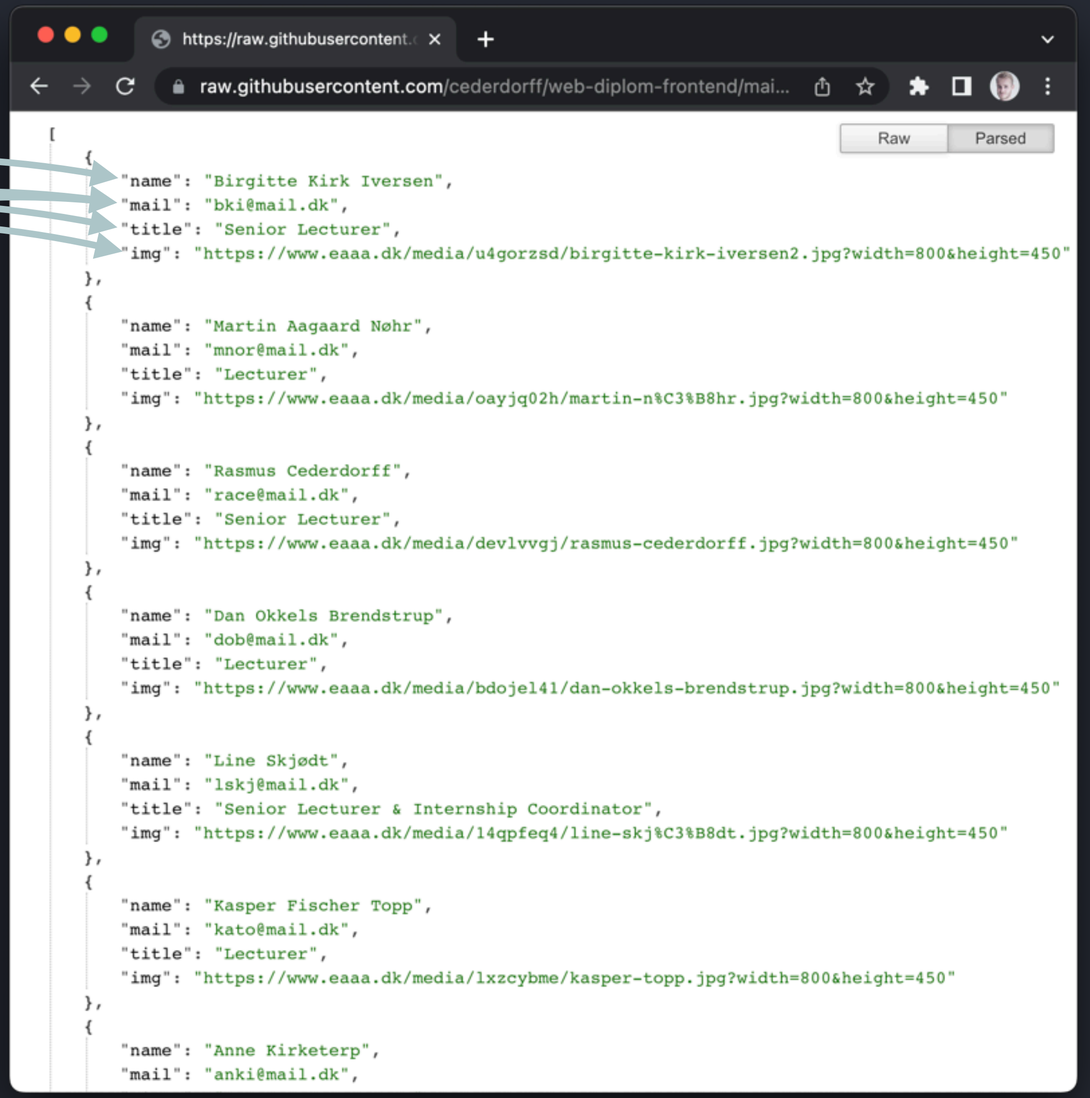
response (JSON)



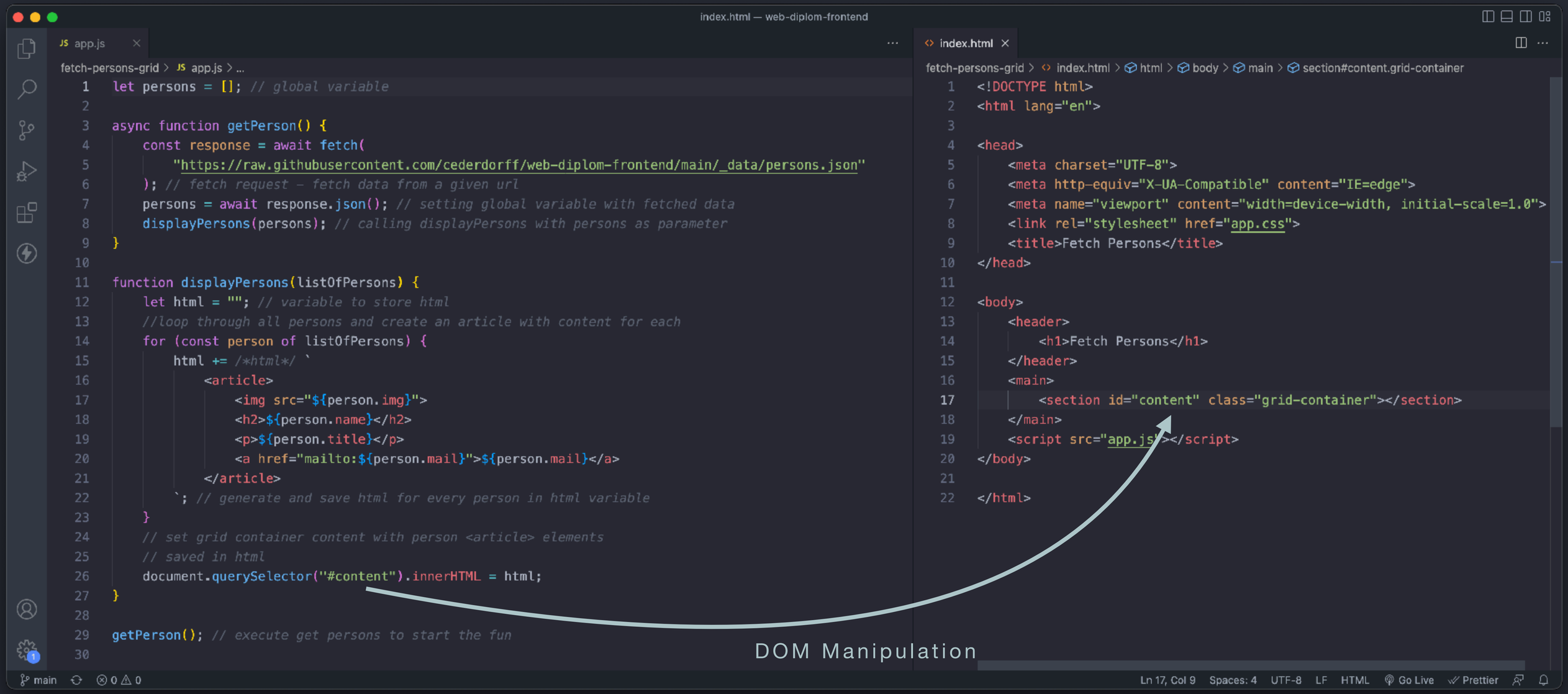


fetch-persons-grid





fetch-persons-grid


index.html — web-diplom-frontend
app.js
fetch-persons-grid > app.js > ...
let persons = []; // global variable
async function getPerson() {
const response = await fetch(
"https://raw.githubusercontent.com/cederdorff/web-diplom-frontend/main/_data/persons.json"
); // fetch request - fetch data from a given url
persons = await response.json(); // setting global variable with fetched data
displayPersons(persons); // calling displayPersons with persons as parameter
}
function displayPersons(listOfPersons) {
let html = ""; // variable to store html
//loop through all persons and create an article with content for each
for (const person of listOfPersons) {
html += /*html*/ `
<article>
<img src="${person.img}">
<h2>${person.name}</h2>
${person.title}
${person.mail}
</article>
`; // generate and save html for every person in html variable
}
// set grid container content with person <article> elements
// saved in html
document.querySelector("#content").innerHTML = html;
}
getPerson(); // execute get persons to start the fun
index.html
fetch-persons-grid > index.html > html > body > main > section#content.grid-container
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link rel="stylesheet" href="app.css">
<title>Fetch Persons</title>
</head>
<body>
<header>
<h1>Fetch Persons</h1>
</header>
<main>
<section id="content" class="grid-container"></section>
</main>
<script src="app.js"></script>
</body>
</html>
DOM Manipulation
main
Ln 17, Col 9
Spaces: 4
UTF-8
LF
HTML
Go Live
Prettier

Fetch

... get & post data from and to a data source

```
// Simple javascipt 😄

//Synchronous fetch using async/await.

// Usual way
✓ const jsonData = fetch('URL')
                    .then(response ⇒ response.json())
                    .then(json ⇒ console.log(json));

// Using await
✓ const jsonData = await fetch('URL').then(res ⇒ res.json())


// Shorter syntax 😍
✓ const jsonData = await (await fetch('URL')).json();
```

https://www.instagram.com/p/B0nxQjXj9Zi/

Async JS

JavaScript reads and runs the script from top to bottom.

JavaScript functions are executed in the sequence they are called. Not in the sequence they are defined.

... by default JavaScript is synchronous.

With callbacks, we can make JS Asynchronous

```
setTimeout(() => {
    console.log("Hey, I'm async!");
}, 3000);

btn.addEventListener('click', () => {
    alert("Hey, you clicked me!");
});
```

```
// fetch with callbacks
fetch("https://cederdorff.github.io/web-frontend/canva
    .then(function (response) {
        return response.json();
    })
    .then(function (data) {
        console.log(data);
    });
```

Fetch is Asynchronous

And it’s about making HTTP requests in JavaScript.
... and a way to get & post data from and to a data source.

```
// fetch with callbacks
fetch("https://cederdorff.github.io/web-frontend/canvas-users/data.json")
    .then(function (response) {
        return response.json();
    })
    .then(function (data) {
        console.log(data);
    });
```

Fetch: callback (`then`) vs `async/await`

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

will you be async to my await?

async & await

- Use `await` to tell JS to wait for a fetch call to finish and to wait for JSON to parse.
- When using `await` you must tell JS that inside of the function goes some asynchronous code by wrapping it in an `async` function.

```
async function getPosts() {
    const url = "https://raw.githubuserconten
    const response = await fetch(url);
    const data = await response.json();
    setPosts(data);
}
getPosts();
```

JSON

Java**S**cript **O**bject **N**otation

JSON

Java**S**cript **O**bject **N**otation

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

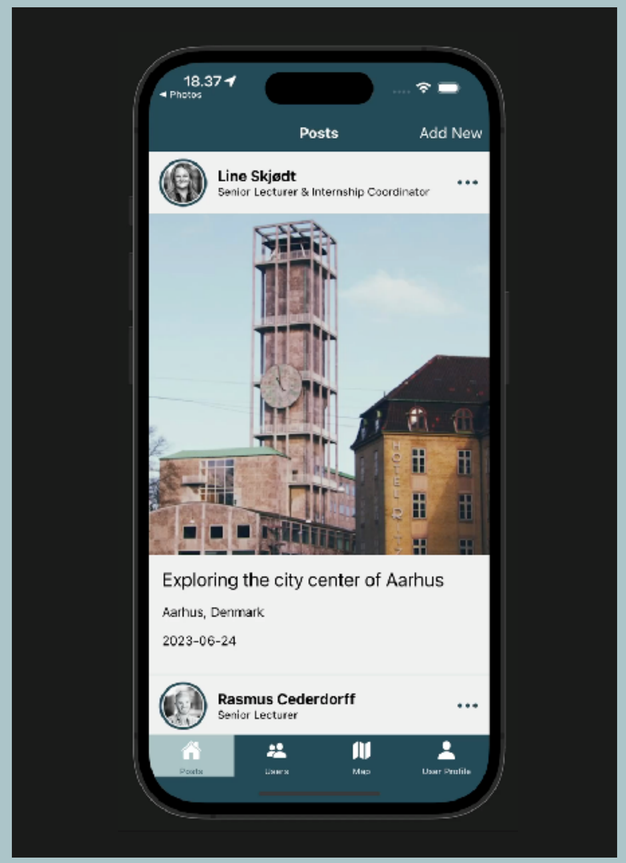
https://www.instagram.com/p/CVqbCzgsZUF/

JSON (& API) is the glue

APP

API

DATA SOURCE

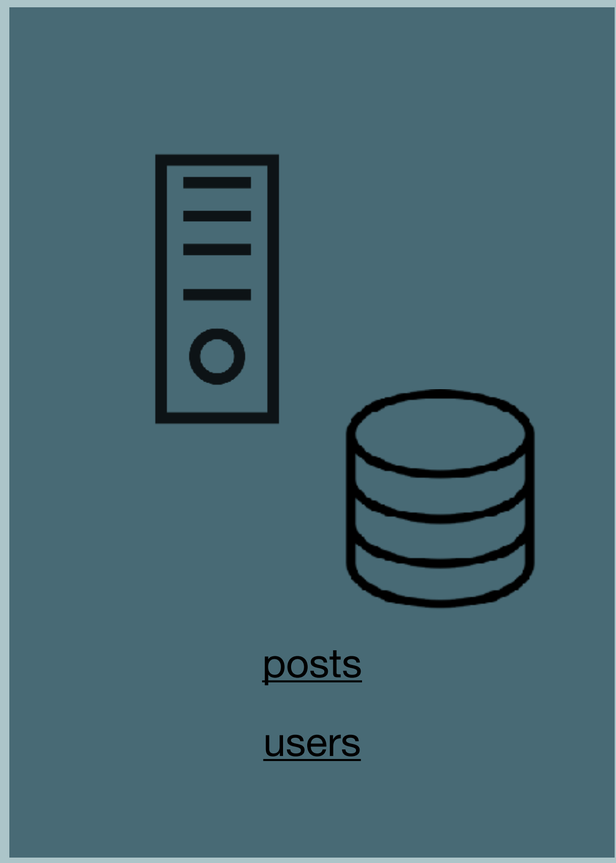


FETCH(GET, POST, PUT, DELETE)

REQUEST

JSON DATA

RESPONSE



CLIENT

SERVER

JSON

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

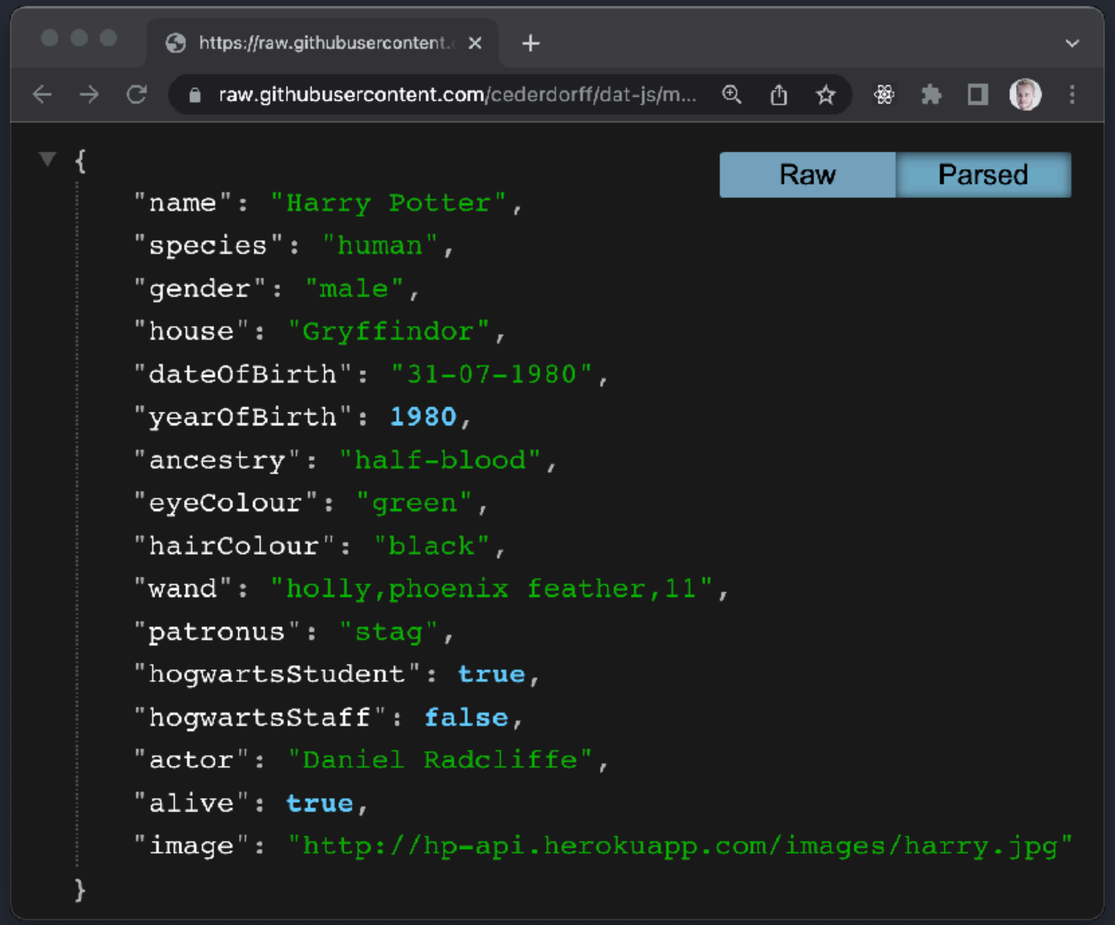
```
{
  "name": "Alicia",
  "age": 6
}
```

JSON OBJECT

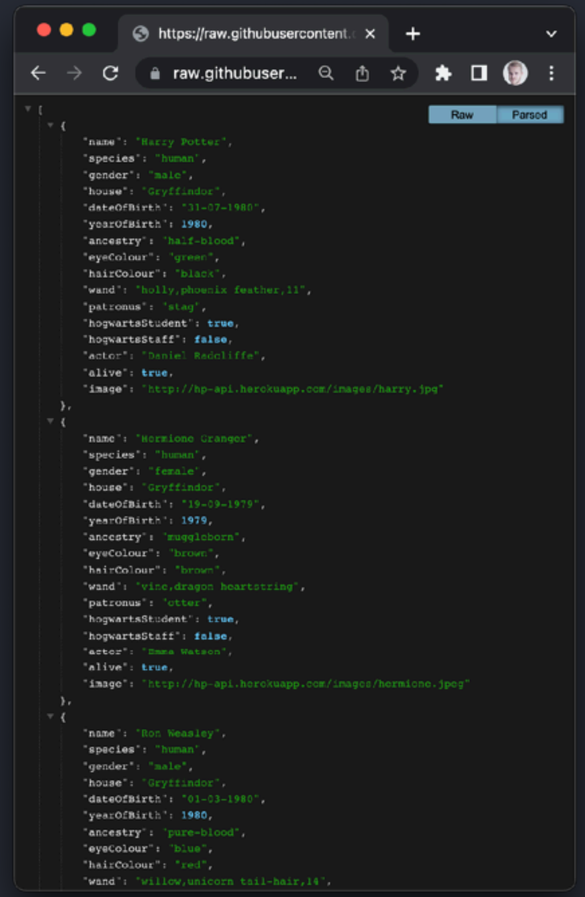
```
[{
  "name": "Alicia",
  "age": 6
}, {
  "name": "Peter",
  "age": 22
}]
```

LIST OF JSON OBJECTS

JSON

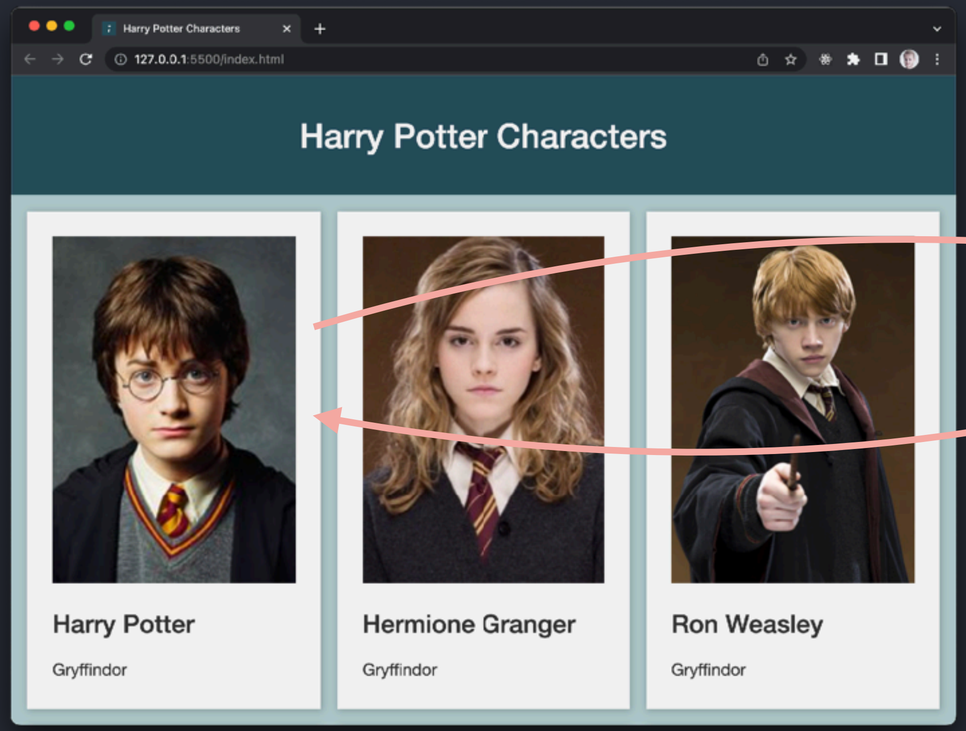


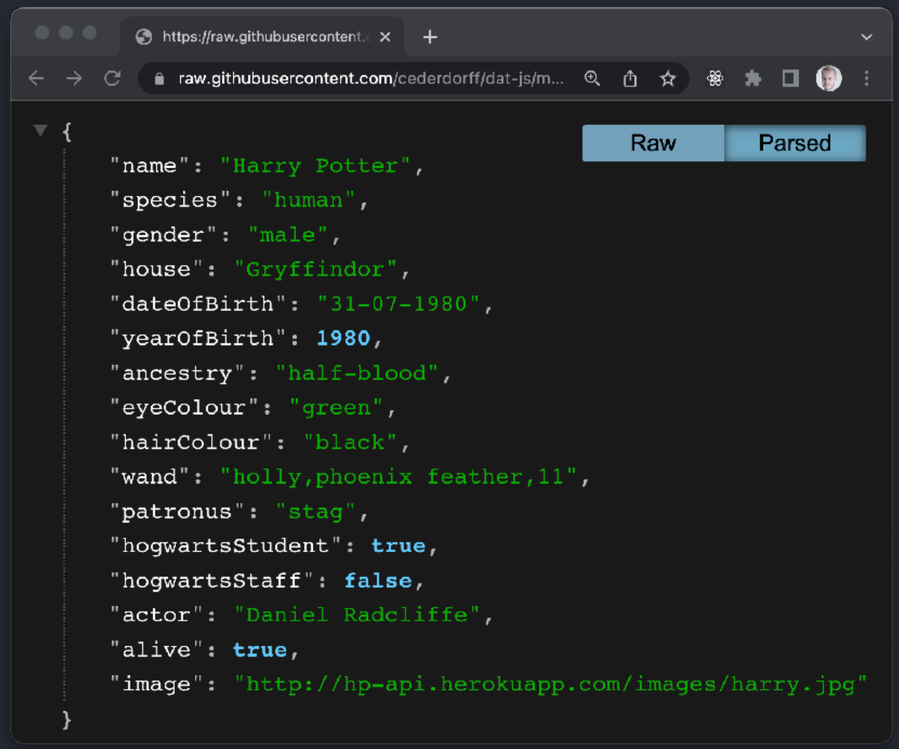
JSON OBJECT



LIST OF JSON OBJECTS

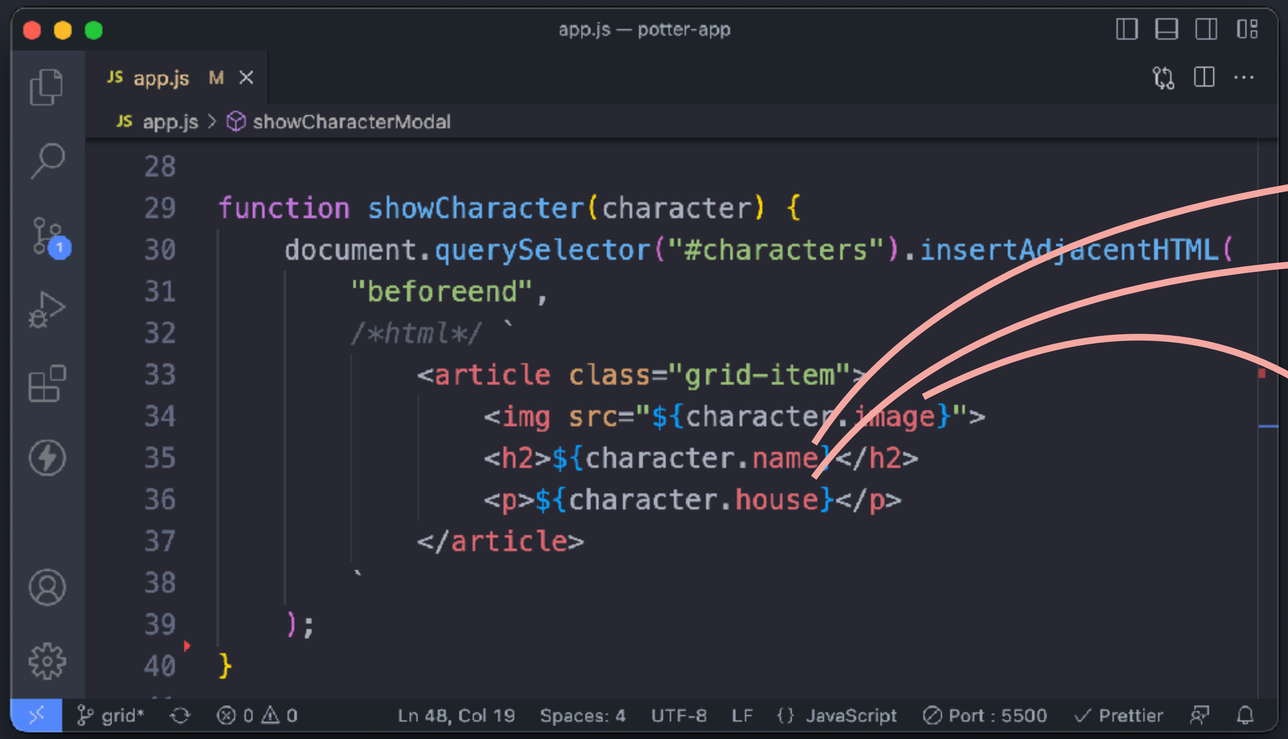
JSON Object

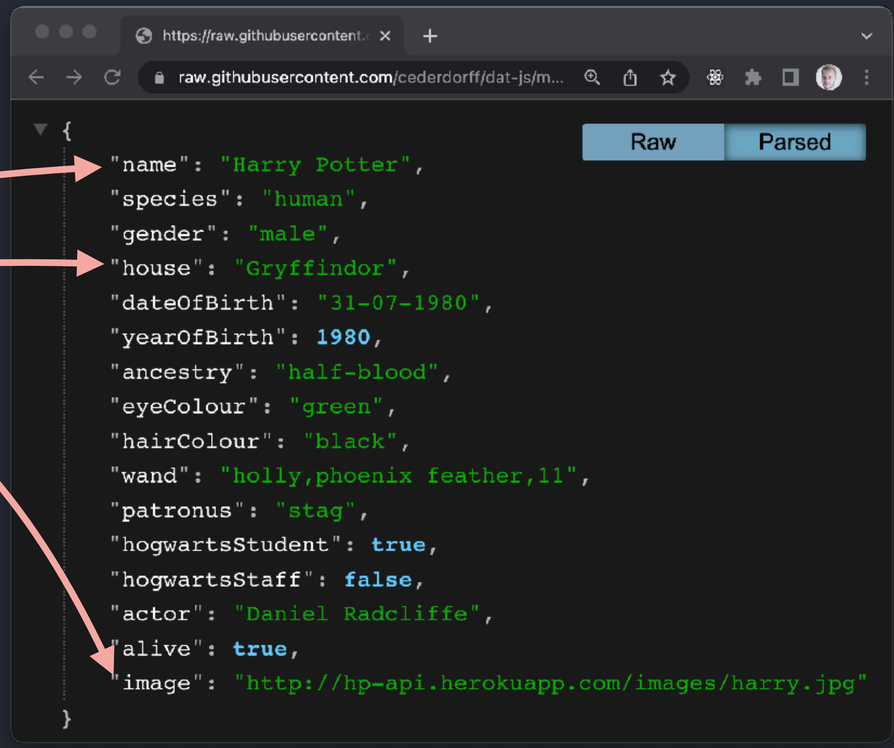




https://raw.githubusercontent.com/cederdorff/dat-js/main/data/harry.json

JSON Object





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

JAVASCRIPT OBJECT NOTATION

- Collection of key-value pair: “`key”: “value”`
- List of values, collections or objects
- Lightweight data-interchange format
- Syntax / text format for storing and exchanging data over the web
- Human and machine readable **text**: small, fast and simple
- Language independent
- Can be parsed directly to JavaScript Object
- JavaScript Objects can be converted directly to JSON
- The glue between programs (interface between frontend and backend)


```
[
    {
        "id": "1",
        "firstname": "Kasper",
        "lastname": "Topp",
        "age": "34",
        "haircolor": "Dark Blonde",
        "countryName": "Denmark",
        "gender": "Male",
        "lookingFor": "Female"
    },
    {
        "id": "2",
        "firstname": "Nicklas",
        "lastname": "Andersen",
        "age": "22",
        "haircolor": "Brown",
        "countryName": "Denmark",
        "gender": "Male",
        "lookingFor": "Female"
    },
    {
        "id": "3",
        "firstname": " Sarah",
        "lastname": "Dybvad ",
        "age": "34",
        "haircolor": "Blonde",
        "countryName": "Denmark",
        "gender": "Female",
        "lookingFor": "Male"
    },
    {
        "id": "4",
        "firstname": "Alex"
```

JSON METHODS

```
const user = {
  name: "John",
  age: 30,
  gender: "male",
  lookingFor: "female"
};

// === JSON.stringify === //
const jsonUser = JSON.stringify(user);
console.log(jsonUser); // {"name":"John","age":30,"gender":"male","lookingFor":"female"}

// === JSON.parse === //
const jsonString = '{"name":"John","age":30,"gender":"male","lookingFor":"female"}';
const userObject = JSON.parse(jsonString);
console.log(userObject); // logging userObject
```

json-parse-stringify

CRUD App

127.0.0.1:5500/index.html

EAAA Canvas KEA BOOKS Cccc & Templates Summer School -...

CRUD App

Rasmus Cederdorff

Senior Lecturer
race@dev.dk

UPDATE DELETE

Lars Bogetoft

Head of Education
larb@eaaa.dk

UPDATE DELETE

Edith Terte

Lecturer
edan@kea.dk

UPDATE DELETE

Frederikke Bender

Head of Education
fbe@kea.dk

UPDATE DELETE

Murat Kilic

Senior Lecturer
mki@eaaa.dk

UPDATE DELETE

Anne Andersen

Head of Education
anki@mail.dk

UPDATE DELETE

Elements Console Sources Network Performance Memory Application

Preserve log Disable cache No throttling

Filter Invert Hide data URLs Hide extension URLs

All Doc JS Fetch/XHR CSS Font Img Media Manifest WS Wasm Other Blocked response cookies

Blocked requests 3rd-party requests

10 ms 20 ms 30 ms 40 ms 50 ms 60 ms 70 ms 80 ms 90 ms 100 ms 110

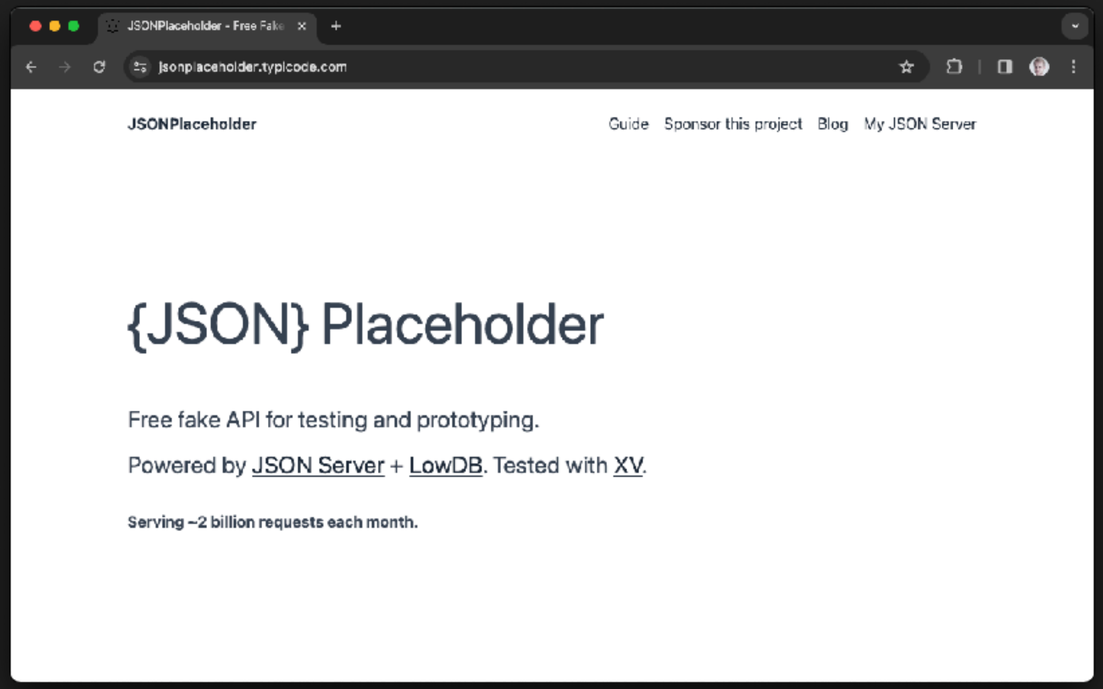
Name

users

Headers Preview Response Initiator Timing


```
[
    {
        "id": 2,
        "name": "Rasmus Cederdorff",
        "mail": "race@dev.dk",
        "title": "Senior Lecturer",
        "image": "https://share.cederdorff.com/images/race.jpg"
    },
    {
        "id": 3,
        "name": "Lars Bogetoft",
        "mail": "larb@eaaa.dk",
        "title": "Head of Education",
        "image": "https://kea.dk/slir/w200-c1x1/images/user-profile/chefer/larb.jpg"
    },
    {
        "id": 4,
        "name": "Edith Terte",
        "mail": "edar@kea.dk",
        "title": "Lecturer",
        "image": "https://kea.dk/slir/w180-c1x1/images/user-profile/synced/EDAN.jpg"
    },
    {
        "id": 5,
        "name": "Frederikke Bender",
        "mail": "fbe@kea.dk",
        "title": "Head of Education",
        "image": "https://kea.dk/slir/w200-c1x1/images/user-profile/chefer/fbe.jpg"
    },
    {
        "id": 6,
        "name": "Murat Kilic",
        "mail": "mki@eaaa.dk",
        "title": "Senior Lecturer",
        "image": "https://www.eaaa.dk/media/llyavasj/murat-kilic.jpg?width=800&heigh
    },
    {
        "id": 7,
        "name": "Anne Andersen",
        "mail": "anki@mail.dk",
        "title": "Head of Education",
        "image": "https://www.eaaa.dk/media/5tuh1xeo/anne-kirketerp.jpg?width=800&he
    }
]
```


1 / 11 requests



https://jsonplaceholder.typicode.com/

Test endpoints

- Install JSON Formatter
- Test the following endpoints:
  - https://jsonplaceholder.typicode.com/posts/
  - https://jsonplaceholder.typicode.com/posts/5
  - https://jsonplaceholder.typicode.com/users/
  - https://jsonplaceholder.typicode.com/users/1
  - https://jsonplaceholder.typicode.com/todos
  - https://jsonplaceholder.typicode.com/todos/5

```
{
  "name": "Rasmus Cederdorff",
  "birthday": "1990-03-12",
  "title": "Experienced JavaScript Developer",
  "experienceYears": 10,
  "education": {
    "degree": "Master of Science in IT - Web Communication",
    "specialization": "Web Architecture"
  },
  "skills": ["JavaScript", "React", "Node.js", "UI/UX Design"],
  "currentPosition": "Senior Lecturer at EAAA",
  "teachingSubjects": ["Web Development", "JavaScript", "React", "BaaS & Node.js"],
  "lovesJavaScript": true
}
```

Define yourself with JSON

- https://race.notion.site/Define-yourself-with-JSON-391944b6b1a041bfa341a0d46f5c0e4a?pvs=4

Array Methods

Array methods are built-in JavaScript functions that make it easier to work with lists of data.

Array Methods

Array methods are built-in JavaScript functions that make it easier to work with lists of data.

They can be used to:

- Loop through all items (`.forEach`, `.map`)
- Filter items based on conditions (`.filter`)
- Search or find specific items (`.find`, `.some`, `.every`)
- Sort or reorder data (`.sort`, `.reverse`)

https://javascript.info/array-methods

Array Methods to know

- `.push(…)`
- `.map(…)`
- `.filter(…)`
- `.find(…)`
- `.sort(…)`

Computer science student

Senior developer, 10+ years experience

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

Array methods

https://javascript.info/array-methods#filter

filter

The `find` method looks for a single (first) element that makes the function return `true`.

If there may be many, we can use arr.filter(fn).

The syntax is similar to `find`, but `filter` returns an array of all matching elements:

```
let results = arr.filter(function(item, index, array) {
  // if true item is pushed to results and the iteration continues
  // returns empty array if nothing found
});
```

For instance:

```
let users = [
  {id: 1, name: "John"},
  {id: 2, name: "Pete"},
  {id: 3, name: "Mary"}
];

// returns array of the first two users
let someUsers = users.filter(item => item.id < 3);

alert(someUsers.length); // 2
```

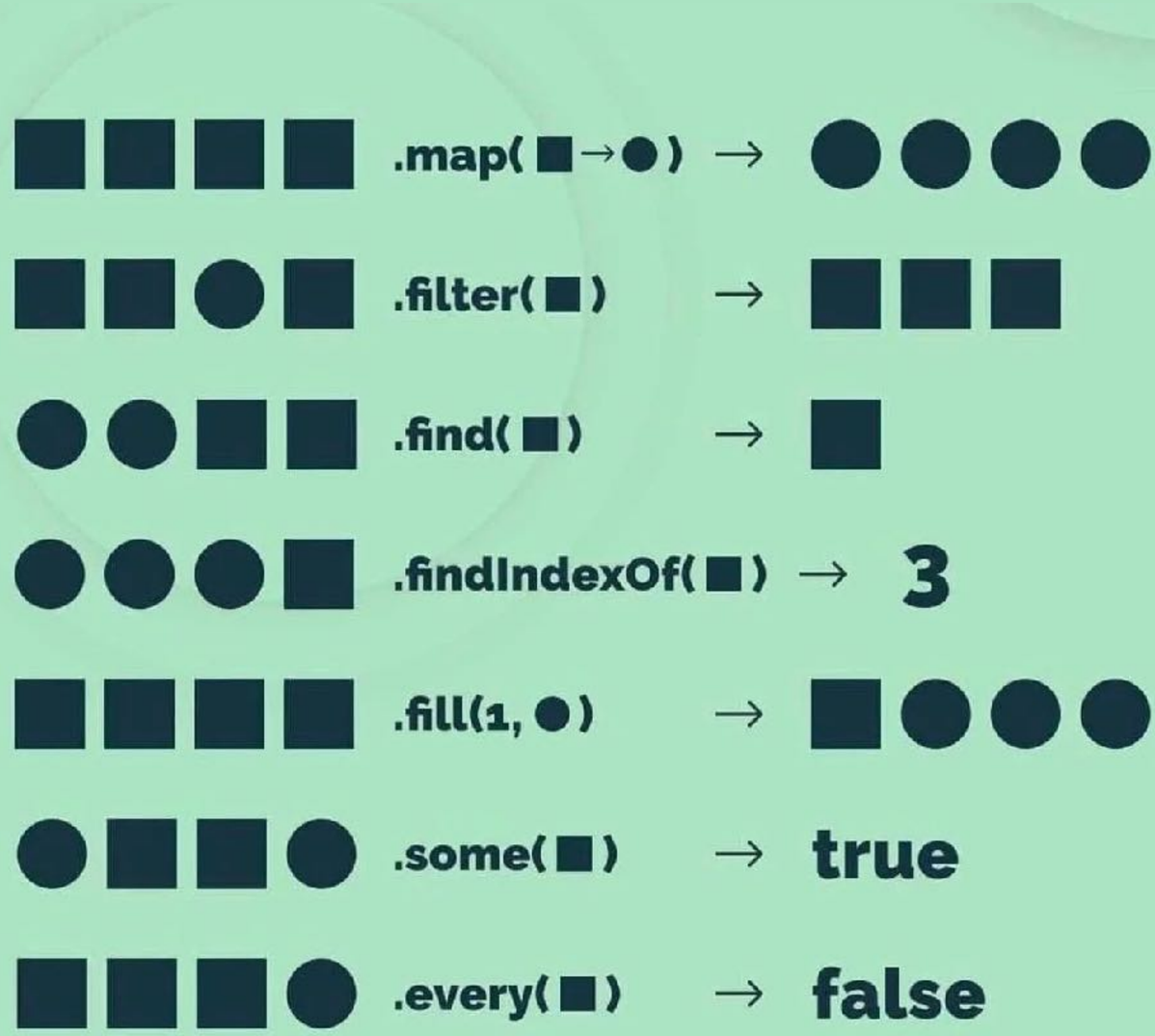
Transform an array

Let's move on to methods that transform and reorder an array.

map

The arr.map method is one of the most useful and often used.

It calls the function for each element of the array and returns the array of results.

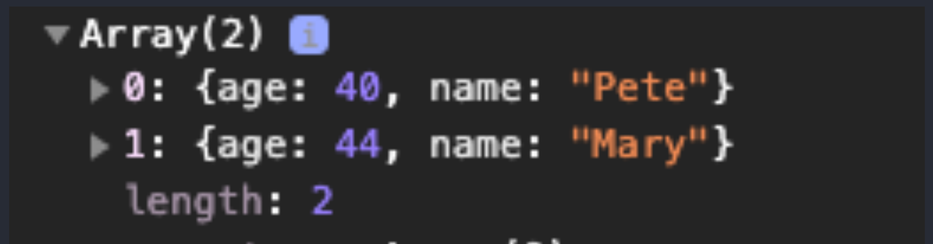


https://javascript.info/array-methods
https://medium.com/@mandeepkaur1/a-list-of-javascript-array-methods-145d09dd19a0

Arrays

.filter()

```
let users = [
  { age: 35, name: "John" },
  { age: 40, name: "Pete" },
  { age: 44, name: "Mary" }
];

// returns array of with users older than 39
let someUsers = users.filter(item => item.age > 39);

console.log(someUsers);
```



Arrays

.filter() to search

```
const persons = [
    {
        name: "Birgitte Kirk Iversen",
        mail: "bki@mail.dk",
        title: "Senior Lecturer",
        img: "https://www.eaaa.dk/media/u4gorzsd/birgitte-kirk-iversen2.jpg?width=300&height=450"
    },
    {
        name: "Martin Aagaard Nøhr",
        mail: "mnor@mail.dk",
        title: "Lecturer",
        img: "https://www.eaaa.dk/media/oeyjq02h/martin-n%C3%B8hr.jpg?width=900&height=450"
    },
    {
        name: "Rasmus Cederdorff",
        mail: "race@mail.dk",
        title: "Senior Lecturer",
        img: "https://www.eaaa.dk/media/devlvvgj/rasmus-cederdorff.jpg?width=800&height=450"
    },
    {
        name: "Dan Okkels Brendstrup",
        mail: "dobg@mail.dk",
        title: "Lecturer",
        img: "https://www.eaaa.dk/media/bdojel41/dan-okkels-brendstrup.jpg?width=800&height=450"
    },
    {
        name: "Anne Kirketerp",
        mail: "anki@mail.dk",
        title: "Head of Department",
        img: "https://www.eaaa.dk/media/5bunlxeo/anne-kirketerp.jpg?width=800&height=450"
    }
];
```

```
function search(event) {
    const searchValue = event.target.value.toLowerCase(); // input text to lower case
    const results = persons.filter(function (person) {
        return person.name.toLowerCase().includes(searchValue);
    });
    displayPersons(results); // call displayPersons with the filtered results
}
```

Filter condition:
`person.name`
must include
`searchValue`

```
// with arrow function
function search(event) {
    const searchValue = event.target.value.toLowerCase(); // input text to lower case
    const results = persons.filter(person => person.name.toLowerCase().includes(searchValue));
    displayPersons(results); // call displayPersons with the filtered results
}
```

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/filter

Arrays

.sort()

JavaScript Demo: Array.sort()

```
1  const months = ['March', 'Jan', 'Feb', 'Dec'];
2  months.sort();
3  console.log(months);
4  // expected output: Array ["Dec", "Feb", "Jan", "March"]
5
6  const array1 = [1, 30, 4, 21, 100000];
7  array1.sort();
8  console.log(array1);
9  // expected output: Array [1, 100000, 21, 30, 4]
10
```

“The `sort()` method sorts the elements of an array in place and returns the reference to the same array, now sorted.”

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/sort

Arrays w/ objects

.sort()

```
const persons = [
    {
        name: "Birgitte Kirk Iversen",
        mail: "bki@mail.dk",
        title: "Senior Lecturer",
        img: "https://www.eaaa.dk/media/u4gorzsd/birgitte-kirk-iversen2.jpg?width=800&height=450"
    },
    {
        name: "Martin Aagaard Nøhr",
        mail: "mnor@mail.dk",
        title: "Lecturer",
        img: "https://www.eaaa.dk/media/oayjq02h/martin-n%C3%B8hr.jpg?width=800&height=450"
    },
    {
        name: "Rasmus Cederdorff",
        mail: "race@mail.dk",
        title: "Senior Lecturer",
        img: "https://www.eaaa.dk/media/devlvvgj/rasmus-cederdorff.jpg?width=800&height=450"
    },
    {
        name: "Dan Okkels Brendstrup",
        mail: "dob@mail.dk",
        title: "Lecturer",
        img: "https://www.eaaa.dk/media/bdojel41/dan-okkels-brendstrup.jpg?width=800&height=450"
    },
    {
        name: "Anne Kirketerp",
        mail: "anki@mail.dk",
        title: "Head of Department",
        img: "https://www.baaa.dk/media/5buh1xeo/anne-kirketerp.jpg?width=800&height=450"
    }
];
```

```
// Use localeCompare for strings
persons.sort(function (person1, person2) {
    person1.name.localeCompare(person2.name);
});

// with arrow function
persons.sort((person1, person2) => person1.name.localeCompare(person2.name));
```

https://javascript.info/array-methods#sort-fn

Code Every Day