

DOM

Document Object Model

Agenda

- Data-forløbet
- The DOM
- DOM-Manipulation
- Øvelser og Opbygning af UI

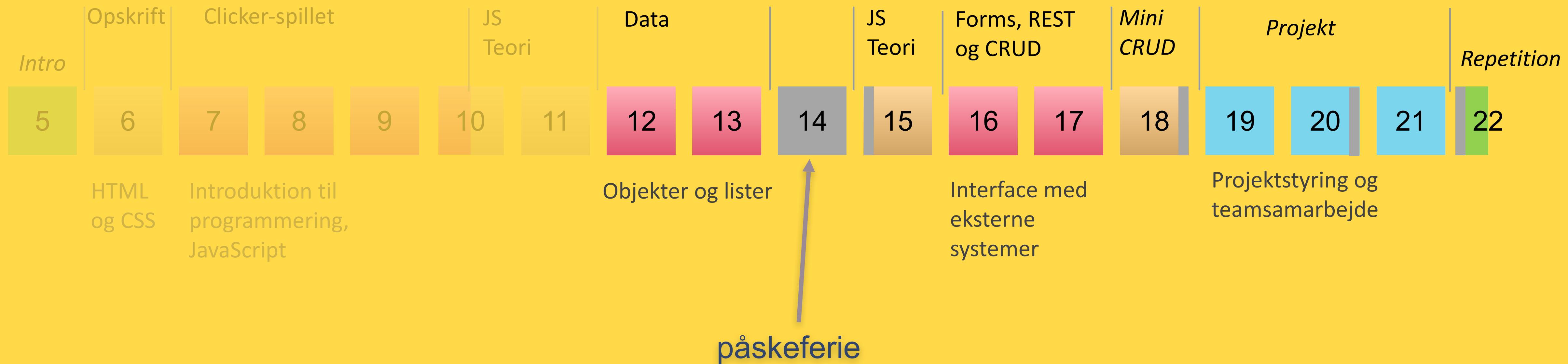
The image shows a screenshot of a code editor window. The menu bar at the top includes 'View', 'Selection', 'Find', 'Debugger', 'Packages', 'Window', 'Help', and 'Native'. The title bar says 'main.js'. The code editor displays the following JavaScript code:

```
1 "use strict";
2
3 fetch("http://headlesscms.cederdorff.com/wp-json/wp/v2/posts?_embed")
4   .then(function(response) {
5     return response.json();
6   })
7   .then(function(json) {
8     appendPosts(json);
9   });
10
11 function appendPosts(posts) {
12   for (let post of posts) {
13     console.log(post);
14     document.querySelector("#grid-posts").innerHTML += `
15       <article>
16         <h3>${post.title.rendered}</h3>
17         <p>Email: <a href="mailto:${post.acf.email}">${post.acf.email}</a></p>
18         <p>Phone: ${post.acf.phone}</p>
19       </article>
20     `;
21   }
22 }
```

The code uses the Fetch API to retrieve JSON data from a WordPress REST API endpoint. It then iterates over the posts and appends HTML to a div with the id 'grid-posts'.

1. SEMESTERS STRUKTUR

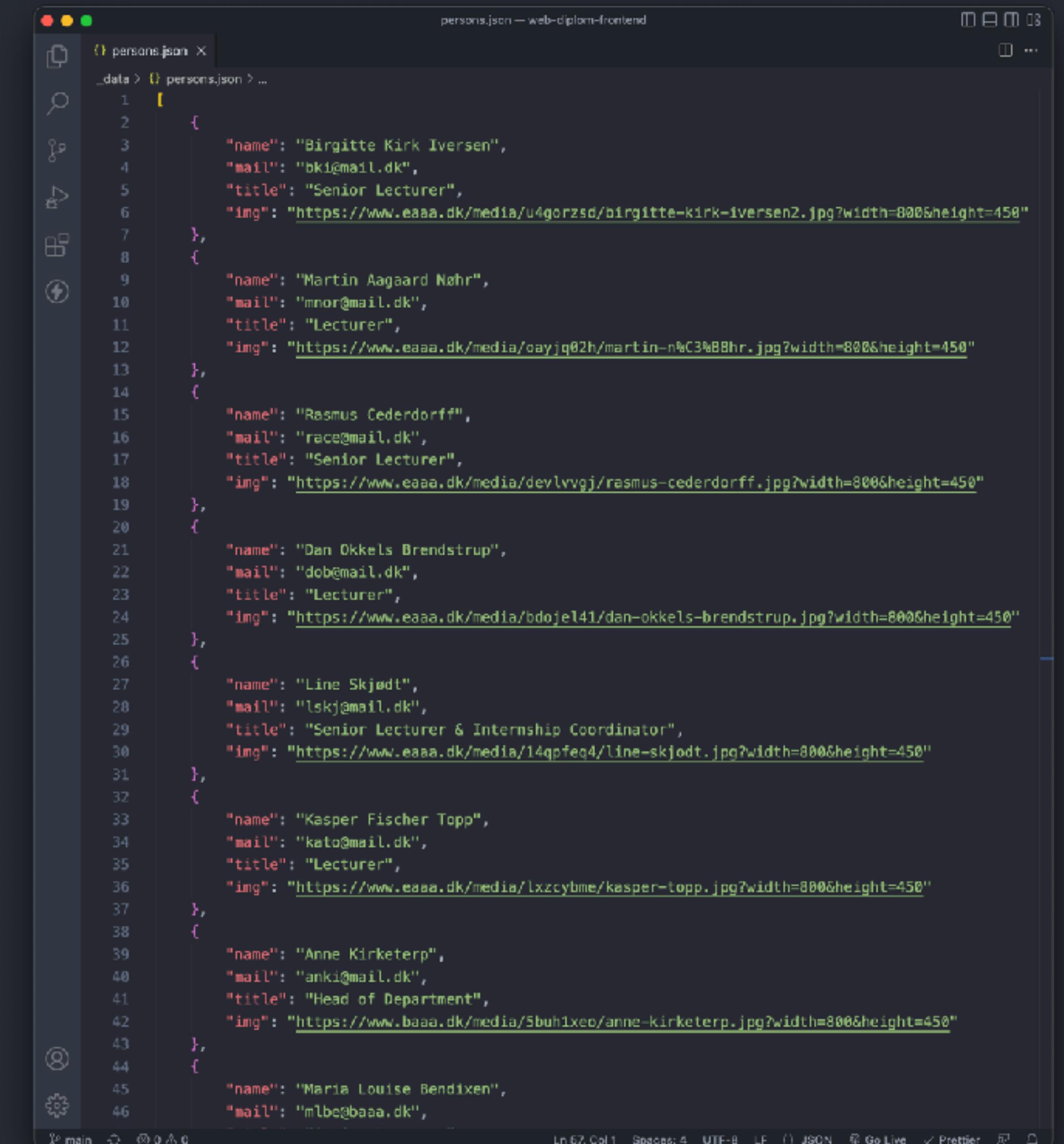
Semesteret består af 16 undervisningsuger (inkl. en masse helligdage)



Eksamnen ligger 12.-14. juni

Data

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



The screenshot shows a code editor window titled "persons.json — web-diplom-frontend". The file contains a JSON array of objects representing lecturers. Each object has properties: name, mail, title, and img. The img URLs are relative paths to images on a server.

```
_data > 0 persons.json > ...
1 [
2   {
3     "name": "Birgitte Kirk Iversen",
4     "mail": "bki@mail.dk",
5     "title": "Senior Lecturer",
6     "img": "https://www.eaaa.dk/media/u4gorzsd/birgitte-kirk-iversen2.jpg?width=800&height=450"
7   },
8   {
9     "name": "Martin Aagaard Nørh",
10    "mail": "mnor@mail.dk",
11    "title": "Lecturer",
12    "img": "https://www.eaaa.dk/media/oayjq02h/martin-n%C3%88hr.jpg?width=800&height=450"
13  },
14  {
15    "name": "Rasmus Cederdorff",
16    "mail": "race@mail.dk",
17    "title": "Senior Lecturer",
18    "img": "https://www.eaaa.dk/media/devlvvgj/rasmus-cederdorff.jpg?width=800&height=450"
19  },
20  {
21    "name": "Dan Okkels Brendstrup",
22    "mail": "dob@mail.dk",
23    "title": "Lecturer",
24    "img": "https://www.eaaa.dk/media/bdoje141/dan-okkels-brendstrup.jpg?width=800&height=450"
25  },
26  {
27    "name": "Line Skjødt",
28    "mail": "tskj@mail.dk",
29    "title": "Senior Lecturer & Internship Coordinator",
30    "img": "https://www.eaaa.dk/media/14qpfeq4/line-skjodt.jpg?width=800&height=450"
31  },
32  {
33    "name": "Kasper Fischer Topp",
34    "mail": "kato@mail.dk",
35    "title": "Lecturer",
36    "img": "https://www.eaaa.dk/media/lxzcybme/kasper-topp.jpg?width=800&height=450"
37  },
38  {
39    "name": "Anne Kirketerp",
40    "mail": "anki@mail.dk",
41    "title": "Head of Department",
42    "img": "https://www.baaa.dk/media/5buh1xeo/anne-kirketerp.jpg?width=800&height=450"
43  },
44  {
45    "name": "Maria Louise Bendixen",
46    "mail": "mlbe@baaa.dk",
47  }
```

CRUD App

Lars Bogetoft
Head of Education
lrb@kea.dk

Peter Lind
Senior Lecturer
pet@kea.dk

Magdalena "Lenka" Otap
Lecturer
mago@kea.dk

Rasmus Cederdorff
Senior Lecturer
race@kea.dk

Create a new User

Magdalena "Lenka" Otap
Lecturer
mago@kea.dk
<https://share.cederdorff.com/images/mago.jpg>

CREATE USER

Update User

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

Frontend (client)

CRUD App

Lars Bogetoft
Head of Education
larb@kea.dk
UPDATE **DELETE**

Peter Lind
Senior Lecturer
petl@kea.dk
UPDATE **DELETE**

Magdalena "Lenka" Otap
Lecturer
mago@kea.dk
UPDATE **DELETE**

Rasmus Cederdorff
Senior Lecturer
race@kea.dk
UPDATE **DELETE**

Create a new User

Update User

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

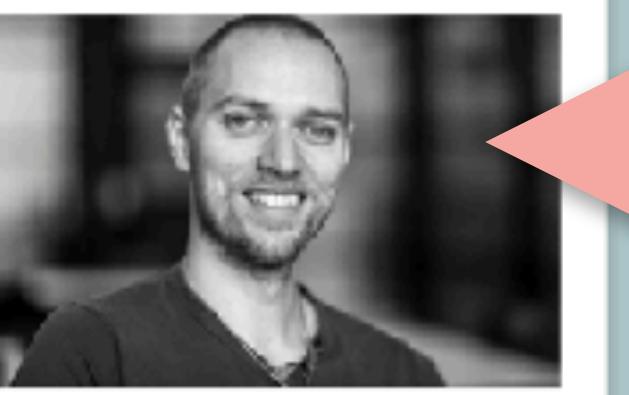
Backend (Server)

```
Raw Parsed
```

```
{  
  "-NNBOTuFaCLHSY-IzmCt": {  
    "image": "https://kea.dk/slir/w360-clxl/images/user-profile/chefer/larb.jpg",  
    "mail": "larb@kea.dk",  
    "name": "Lars Bogetoft",  
    "title": "Head of Education"  
  },  
  "-NNBQFn28wOOLJTxvma": {  
    "image": "https://share.cederdorff.com/images/petl.jpg",  
    "mail": "petl@kea.dk",  
    "name": "Peter Lind",  
    "title": "Senior Lecturer"  
  },  
  "-NNBQSO5gFb-4xB7liug": {  
    "image": "https://share.cederdorff.com/images/mago.jpg",  
    "mail": "mago@kea.dk",  
    "name": "Magdalena \"Lenka\" Otap",  
    "title": "Lecturer"  
  },  
  "fTs84KRoYw5pRZEWcq2Z": {  
    "image": "https://share.cederdorff.com/images/race.jpg",  
    "mail": "race@kea.dk",  
    "name": "Rasmus Cederdorff",  
    "title": "Senior Lecturer"  
  }  
}
```

<https://race-dat-v1-default-rtdb.firebaseio.com/users.json>

Fetch Persons

 <p>Birgitte Kirk Iversen Senior Lecturer hki@mail.dk</p>	 <p>Martin Aagaard Nøhr Lecturer mnor@mail.dk</p>	 <p>Rasmus Cederdorff Senior Lecturer mcn@mail.dk</p>
 <p>Dan Okkels Brendstrup Lecturer dob@mail.dk</p>	 <p>Line Skjødt Senior Lecturer & Internship Coordinator tskj@mail.dk</p>	 <p>Kasper Fischer Topp Lecturer kato@mail.dk</p>
 <p>Anne Kirketerp Head of Department anki@mail.dk</p>	 <p>Maria Louise Bendixen Senior Lecturer mlbe@baaa.dk</p>	 <p>Marlene Ahlgreen Jensen Senior Lecturer maj@baaa.dk</p>

```
persons.json — web-diplom-frontend
```

_data > persons.json > ...

```
1  [
2  {
3    "name": "Birgitte Kirk Iversen",
4    "mail": "bki@mail.dk",
5    "title": "Senior Lecturer",
6    "img": "https://www.eaaa.dk/media/u4gorzsd/birgitte-kirk-iversen2.jpg?width=800&height=450"
7  },
8  {
9    "name": "Martin Aagaard Nøhr",
10   "mail": "mnor@mail.dk",
11   "title": "Lecturer",
12   "img": "https://www.eaaa.dk/media/cayjq02h/martin-n%C3%88hr.jpg?width=800&height=450"
13 },
14 {
15   "name": "Rasmus Cederdorff",
16   "mail": "race@mail.dk",
17   "title": "Senior Lecturer",
18   "img": "https://www.eaaa.dk/media/devlvvgj/rasmus-cederdorff.jpg?width=800&height=450"
19 },
20 {
21   "name": "Dan Okkels Brendstrup",
22   "mail": "dob@mail.dk",
23   "title": "Lecturer",
24   "img": "https://www.eaaa.dk/media/bdoje141/dan-okkels-brendstrup.jpg?width=800&height=450"
25 },
26 {
27   "name": "Line Skjødt",
28   "mail": "tskj@mail.dk",
29   "title": "Senior Lecturer & Internship Coordinator",
30   "img": "https://www.eaaa.dk/media/14qpfeq4/line-skjødt.jpg?width=800&height=450"
31 },
32 {
33   "name": "Kasper Fischer Topp",
34   "mail": "kato@mail.dk",
35   "title": "Lecturer",
36   "img": "https://www.eaaa.dk/media/lxzcybme/kasper-topp.jpg?width=800&height=450"
37 },
38 {
39   "name": "Anne Kirketerp",
40   "mail": "anki@mail.dk",
41   "title": "Head of Department",
42   "img": "https://www.baaa.dk/media/5buh1xeo/anne-kirketerp.jpg?width=800&height=450"
43 },
44 {
45   "name": "Maria Louise Bendixen",
46   "mail": "mlbe@baaa.dk",
```

Ln 62, Col 1 Spaces: 4 UTF-8 LF () JSON Go Live Prettier

Fetch Persons

127.0.0.1:5503/fetch-persons-grid/index.html

Apps KEA DAT DAT.js IT Arkitektur EAAA BOOKS

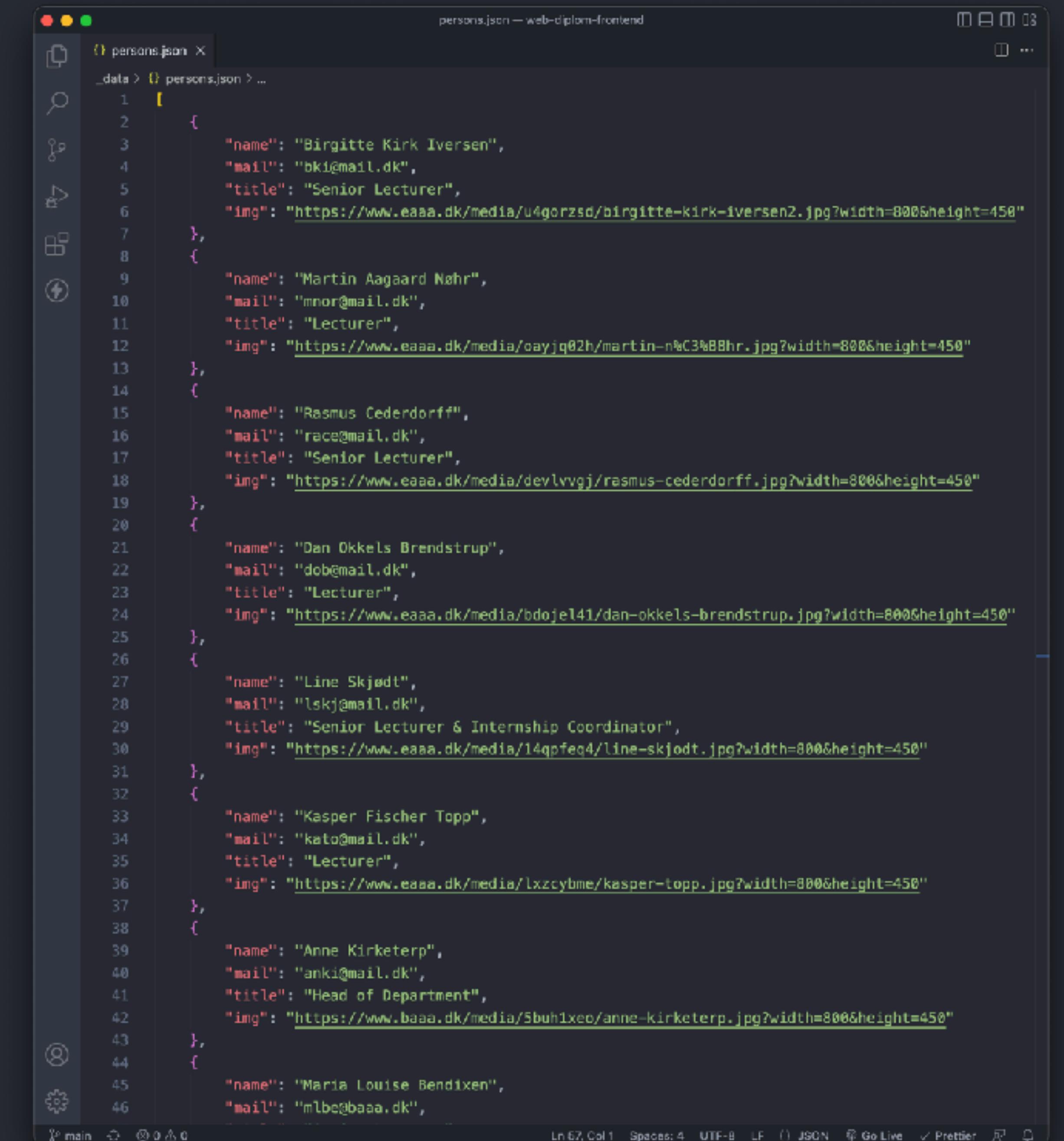
Fetch Persons

 <p>Birgitte Kirk Iversen Senior Lecturer hki@mail.dk</p>	 <p>Martin Aagaard Nøhr Lecturer mnr@mail.dk</p>	 <p>Rasmus Cederdorff Senior Lecturer rnc@mail.dk</p>
 <p>Dan Okkels Brendstrup Lecturer dob@mail.dk</p>	 <p>Line Skjødt Senior Lecturer & Internship Coordinator lskj@mail.dk</p>	 <p>Kasper Fischer Topp Lecturer keto@mail.dk</p>
 <p>Anne Kirketerp Head of Department anki@mail.dk</p>	 <p>Maria Louise Bendixen Senior Lecturer mlbe@basa.dk</p>	 <p>Marlene Ahlgreen Jensen Senior Lecturer maj@basa.dk</p>

Forståelse for objekter og arrays

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

- Denne rette visualisering
- Generere HTML
- DOM-manipulation

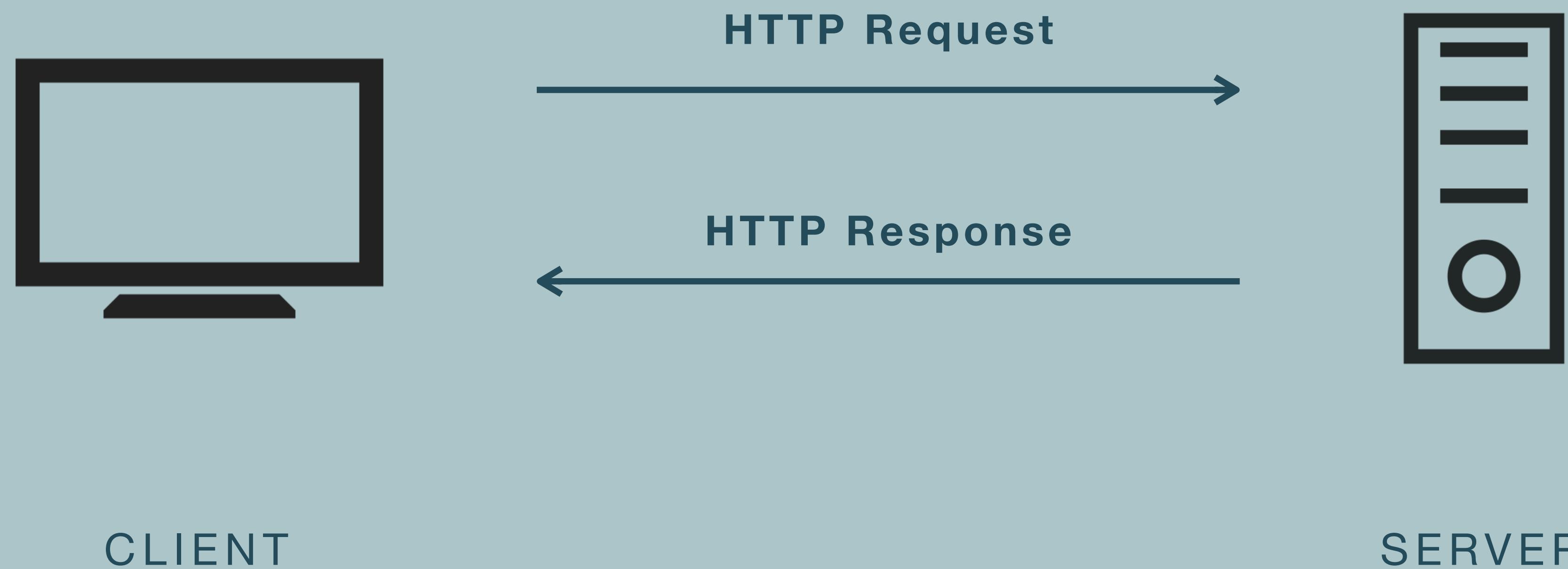


```
persons.json — web-diplom-frontend
persons.json
_ln 62, Col 1  Spaces: 4  UTF-8  LF  ()  JSON  Go Live  Prettier  ⌂  ⌂

1 [           _data > 0 persons.json > ...
2 {             1 [
3   "name": "Birgitte Kirk Iversen",       2 {
4   "mail": "bki@mail.dk",                 3   "name": "Martin Aagaard Nørh",
5   "title": "Senior Lecturer",           4   "mail": "mnor@mail.dk",
6   "img": "https://www.eaaa.dk/media/u4gorzsd/birgitte-kirk-iversen2.jpg?width=800&height=450" 5   "title": "Lecturer",
7 },                                         6   "img": "https://www.eaaa.dk/media/cayjq02h/martin-n%C3%88hr.jpg?width=800&height=450"
8 {                                         7 },
9   "name": "Rasmus Cederdorff",          8 },
10  "mail": "racer@mail.dk",              9   "name": "Dan Okkels Brendstrup",
11  "title": "Senior Lecturer",          10  "mail": "dob@mail.dk",
12  "img": "https://www.eaaa.dk/media/devlvvgj/rasmus-cederdorff.jpg?width=800&height=450" 11  "title": "Lecturer",
13 },                                         12  "img": "https://www.eaaa.dk/media/bdoje141/dan-okkels-brendstrup.jpg?width=800&height=450"
14 {                                         13 },
15   "name": "Line Skjødt",              14 },
16   "mail": "tskj@mail.dk",              15   "name": "Kasper Fischer Topp",
17   "title": "Senior Lecturer & Internship Coordinator", 16   "mail": "kato@mail.dk",
18   "img": "https://www.eaaa.dk/media/14qpfeq4/line-skjodt.jpg?width=800&height=450" 17   "title": "Lecturer",
19 },                                         18   "img": "https://www.eaaa.dk/media/lxzcybme/kasper-topp.jpg?width=800&height=450"
20 {                                         19 },
21   "name": "Anne Kirketerp",           20 },
22   "mail": "anki@mail.dk",             21   "name": "Maria Louise Bendixen",
23   "title": "Head of Department",      22   "mail": "mlbe@baaa.dk",
24   "img": "https://www.baaa.dk/media/5buhixeo/anne-kirketerp.jpg?width=800&height=450" 23 }
```

Client-Server Model

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



Client-Server Model

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

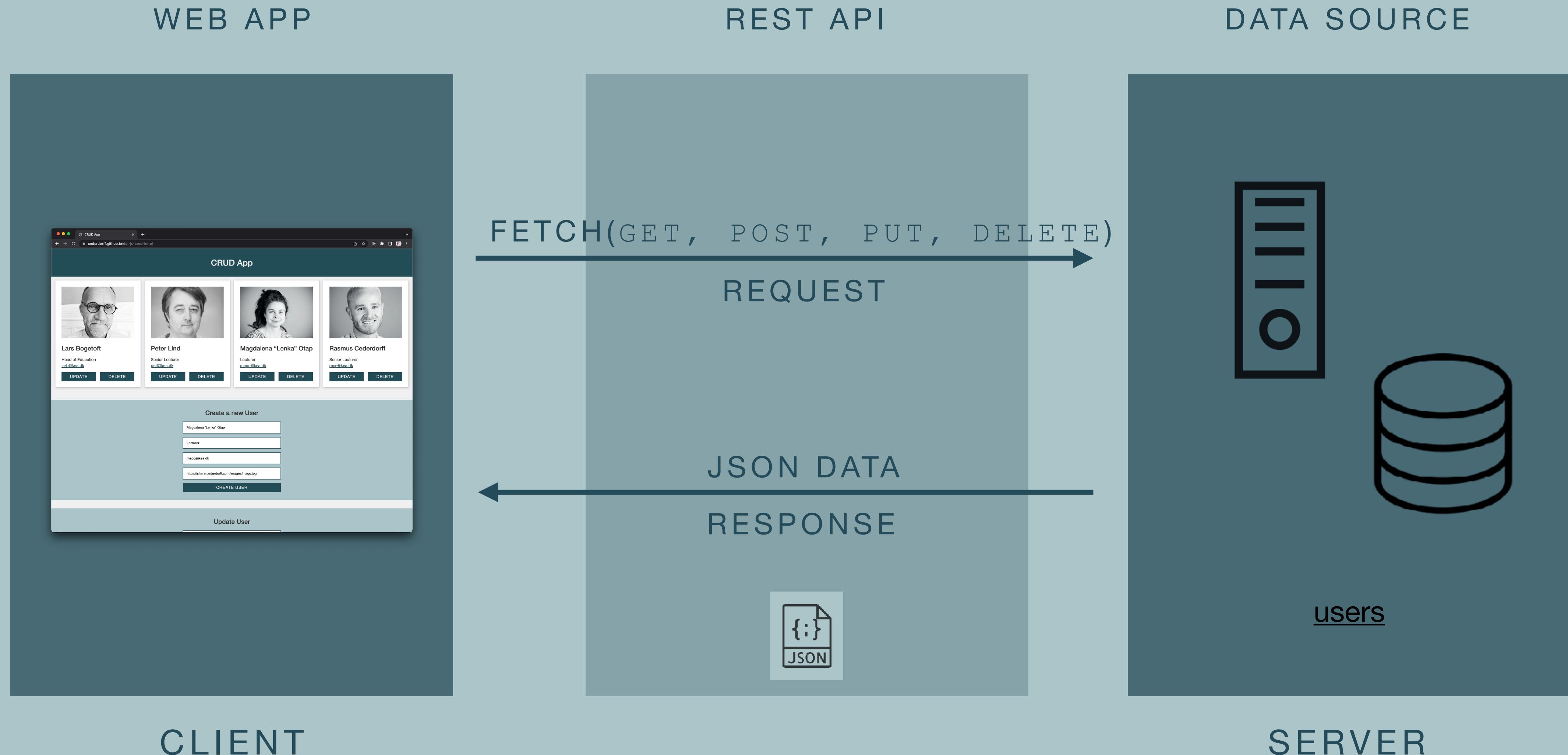


Hyper Text Transfer Protocol

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

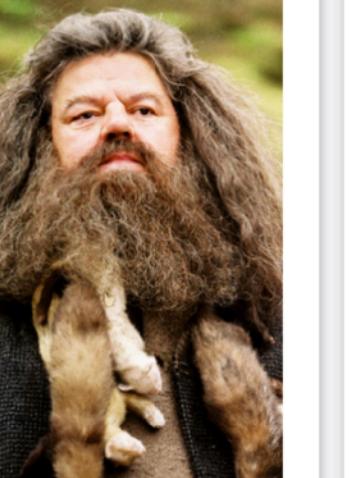


Web Development



The Potter App

Harry Potter Characters

				
Harry Potter Gryffindor	Hermione Granger Gryffindor	Ron Weasley Gryffindor	Draco Malfoy Slytherin	Minerva McGonagall Gryffindor
				

Harry Potter Wizarding Universe Characters

	Harry Potter	Gryffindor
	Hermione Granger	Gryffindor
	Ron Weasley	Gryffindor
	Draco Malfoy	Slytherin
	Minerva McGonagall	Gryffindor
	Cedric Diggory	Hufflepuff
	Cho Chang	Ravenclaw
	Severus Snape	Slytherin
	Rubeus Hagrid	Gryffindor
	Neville Longbottom	Gryffindor
	Luna Lovegood	Ravenclaw

Hvad sker der? Tjek Network-tabben!

The screenshot shows a web browser window titled "Harry Potter Characters" displaying a grid of eight Harry Potter characters: Harry Potter, Hermione Granger, Ron Weasley, Draco Malfoy, Professor McGonagall, Cedric Diggory, Luna Lovegood, and Professor Snape. Below each character is their name and house. To the right of the browser window is the developer tools Network tab, which is active. The Network tab shows a timeline at the top and a detailed list of requests below. The requests list includes 29 requests, totaling 7.5 kB transferred and 2.3 MB resources. The requests are mostly for image files (e.g., harry.jpg, hermione.jpg) and are categorized by initiator, size, and time.

Name	Status	Type	Initiator	Size	Time	Waterfall
potter-app/	304	document	Other	150 B	21...	
app.css	200	stylesheet	(index)	(me...)	0 ms	
app.js	200	script	(index)	(me...)	0 ms	
harry.jpg	304	image	(index)	294 B	10...	
potter.json	200	fetch	app.js:15	(dis...)	0 ms	
hermione.jpg	304	image	Other	294 B	15...	
ron.jpg	304	image	Other	294 B	15...	
draco.jpg	304	image	Other	294 B	15...	
snape.jpg	304	image	Other	294 B	28...	
mcgonagall.jpg	304	image	Other	294 B	15...	
cedric.png	304	image	Other	295 B	15...	
hagrid.png	304	image	Other	295 B	12...	
cho.jpg	304	image	Other	294 B	15...	
neville.jpg	304	image	Other	294 B	12...	
luna.jpg	304	image	Other	294 B	12...	
ginny.jpg	304	image	Other	294 B	12...	
sirius.JPG	304	image	Other	294 B	12...	
lupin.jpg	304	image	Other	294 B	11...	
arthur.jpg	304	image	Other	295 B	10...	
bellatrix.jpg	304	image	Other	294 B	11...	
voldemort.jpg	304	image	Other	294 B	10...	
slughorn.JPG	304	image	Other	294 B	10...	
kingsley.jpg	304	image	Other	295 B	10...	
umbridge.jpg	304	image	Other	295 B	10...	
lucius.jpg	304	image	Other	294 B	10...	
crabbe.jpg	304	image	Other	294 B	10...	
goyle.jpg	304	image	Other	294 B	11...	
norris.JPG	304	image	Other	294 B	99...	
filch.jpg	304	image	Other	294 B	10...	

<https://cederdorff.github.io/dat-js/05-data/potter-app/>

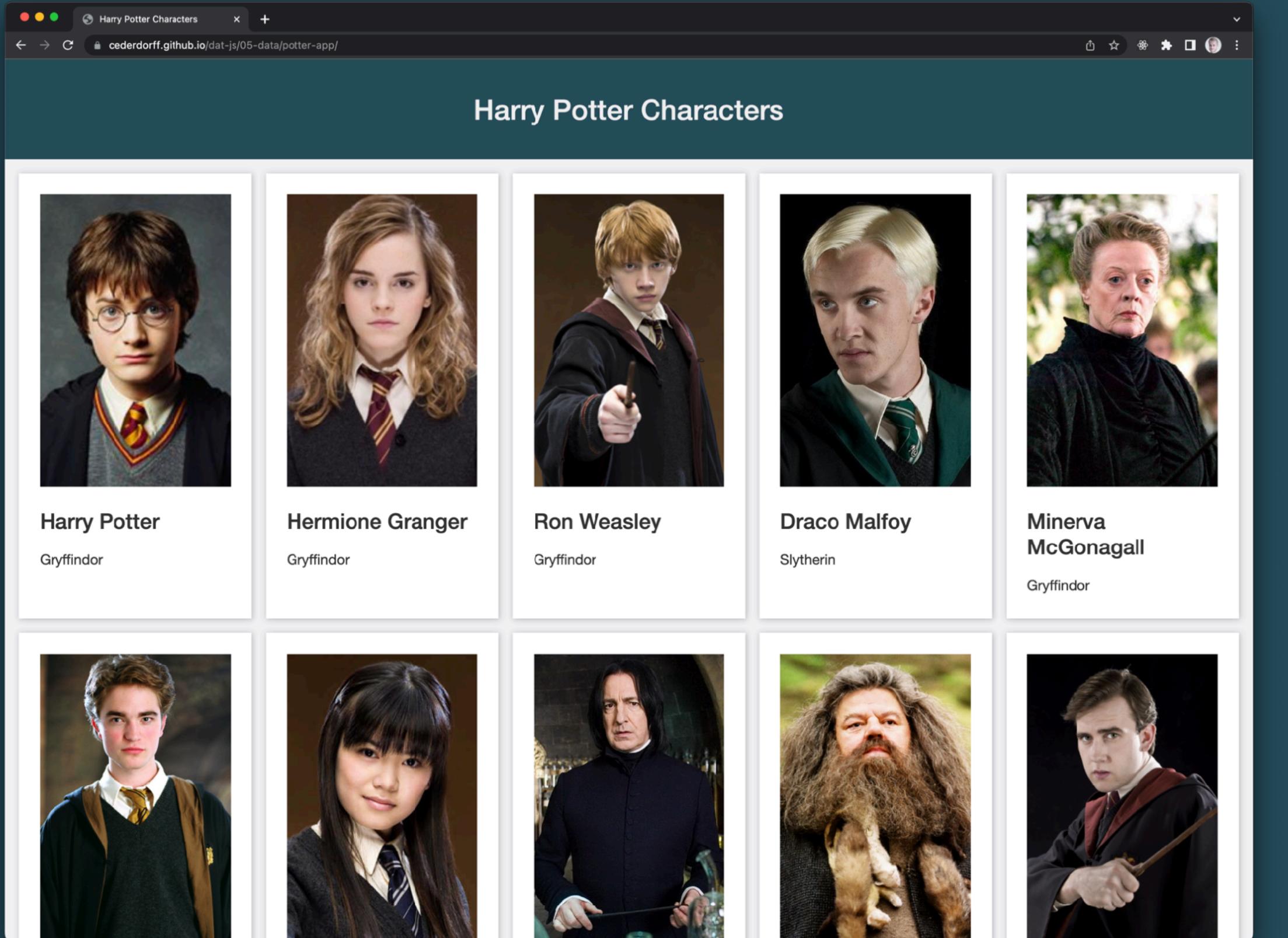
1. Brug Network-tabben til at undersøge:

1. <https://cederdorff.github.io/dat-js/05-data/peters-potter-app/>
 2. <https://cederdorff.github.io/dat-js/05-data/potter-app/>
2. Åben websitet i Chrome (eller en anden browser).
3. Åben Developer Tool (se: [How to open the dev tool in your browser](#)) og gå til Network/Netværk.
4. Genindlæs siden imens du står i Network-tabben og undersøg hvilke ressourcer, der bliver hentet.
5. Hvilke(n) type ressourcer er der tale om?
6. Overvej hvordan ressourcerne hentes.
7. Kan du se hvilken data, der hentes?

Network-tabben



Frontend (client)



<https://cederdorff.github.io/dat-js/05-data/potter-app/>

JSON Data Source (Server)

The screenshot shows a browser window displaying a JSON object representing the Harry Potter characters. The JSON is structured as follows:

```
name: "Harry Potter",  
species: "human",  
gender: "male",  
house: "Gryffindor",  
dateOfBirth: "31-07-1980",  
yearOfBirth: 1980,  
ancestry: "half-blood",  
eyeColour: "green",  
hairColour: "black",  
wand: {  
    wood: "holly",  
    core: "phoenix feather",  
    length: 11  
,  
    patronus: "stag",  
    hogwartsStudent: true,  
    hogwartsStaff: false,  
    actor: "Daniel Radcliffe",  
    alive: true,  
    image: "http://hp-api.herokuapp.com/images/harry.jpg"  
,  
name: "Hermione Granger",  
species: "human",  
gender: "female",  
house: "Gryffindor",  
dateOfBirth: "19-09-1979",  
yearOfBirth: 1979,  
ancestry: "muggleborn",  
eyeColour: "brown",  
hairColour: "brown",  
wand: {  
    wood: "vine",  
    core: "dragon heartstring",  
    length: "  
,  
    patronus: "otter",  
    hogwartsStudent: true,  
    hogwartsStaff: false,  
    actor: "Emma Watson",  
    alive: true,  
    image: "http://hp-api.herokuapp.com/images/hermione.jpeg"  
,  
name: "Ron Weasley",  
species: "human",  
gender: "male",  
house: "Gryffindor",  
dateOfBirth: "01-03-1980",  
yearOfBirth: 1980,  
ancestry: "pure-blood",  
eyeColour: "blue",  
hairColour: "red",  
wand: {
```

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

Hvad sker der uden JavaScript?

The screenshot shows a web application titled "Harry Potter Characters" displaying a grid of four characters: Harry Potter, Hermione Granger, Ron Weasley, and Draco Malfoy. Below each character is their name and house. The network tab in the developer tools shows 29 requests, with most being image files (e.g., harry.jpg, hermione.jpg) and one being the index file (potter-app/). The total transferred data is 7.5 KB.

The screenshot shows the same "Harry Potter Characters" application, but the images of the characters have been removed. The network tab shows the same 29 requests as the first screenshot, but the transferred data is significantly larger at 14.0 kB, indicating the cost of loading all those images.

How to Disable JavaScript

Med JavaScript

Uden JavaScript

```
...<!DOCTYPE html> == $0
<html lang="en">
  ><head> ...
  ><body>
    ><header> ...
    ><main>
      <section id="characters" class="grid-container"></section> grid
      </main>
    ><dialog id="dialog-character"> ...
    ><footer> ...
      <script src="app.js"></script>
    </body>
</html>
```

DOM

```
<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">  
  <link rel="shortcut icon" href="https://cederdorff.github.io/img/favicon.webp" type="image/x-icon">  
  <title>Harry Potter Characters</title>  
</head>  
  
<body>  
  <header>  
    <h1>Harry Potter Characters</h1>  
  </header>  
  <main>  
    <section id="characters" class="grid-container">  
      <article>  
          
        <h2>Harry Potter</h2>  
        <p>Gryffindor</p>  
      </article>  
  
      <article>  
          
        <h2>Hermione Granger</h2>  
        <p>Gryffindor</p>  
      </article>  
  
      <article>  
          
        <h2>Ron Weasley</h2>  
        <p>Gryffindor</p>  
      </article>  
  
      <article>  
          
        <h2>Draco Malfoy</h2>  
        <p>Slytherin</p>  
      </article>  
    </section>  
  </main>  
</body>
```

Source Code (HTML)

```
<!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">  
  <link rel="shortcut icon" href="https://cederdorff.github.io/img/favicon.webp" type="image/x-icon">  
  <title>Harry Potter Characters</title>  
</head>  
  
<body>  
  <header>  
    <h1>Harry Potter Characters</h1>  
  </header>  
  <main>  
    <section id="characters" class="grid-container"></section>  
  </main>  
  
  <dialog id="dialog-character">...</dialog>  
  <footer>  
    <p>Potter App by RACE</p>  
  </footer>  
  
<script src="app.js"></script>  
</body>
```

The DOM

“

The Document Object Model (DOM) is a programming interface for web documents.

It represents the page so that programs can change the document structure, style, and content.

”

https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction

Source Code (HTML)

The browser fetches the source code (HTML).

When the source code is loaded the browser creates The HTML DOM.

```
<!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">
  <link rel="shortcut icon" href="https://cederdorff.github.io/img/favicon.ico">
  <title>Harry Potter Characters</title>
</head>

<body>
  <header>
    <h1>Harry Potter Characters</h1>
  </header>
  <main>
    <section id="characters" class="grid-container"></section>
  </main>

  <dialog id="dialog-character">...
  </dialog>
  <footer>
    <p>Potter App by RACE</p>
  </footer>

  <script src="app.js"></script>
</body>
```

DOM

```
<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">
  <link rel="shortcut icon" href="https://cederdorff.github.io/img/favicon.webp" type="image/x-icon">
  <title>Harry Potter Characters</title>
</head>

<body>
  <header>
    <h1>Harry Potter Characters</h1>
  </header>
  <main>
    <section id="characters" class="grid-container">
      <article>
        
        <h2>Harry Potter</h2>
        <p>Gryffindor</p>
      </article>

      <article>
        
        <h2>Hermione Granger</h2>
        <p>Gryffindor</p>
      </article>

      <article>
        
        <h2>Ron Weasley</h2>
        <p>Gryffindor</p>
      </article>

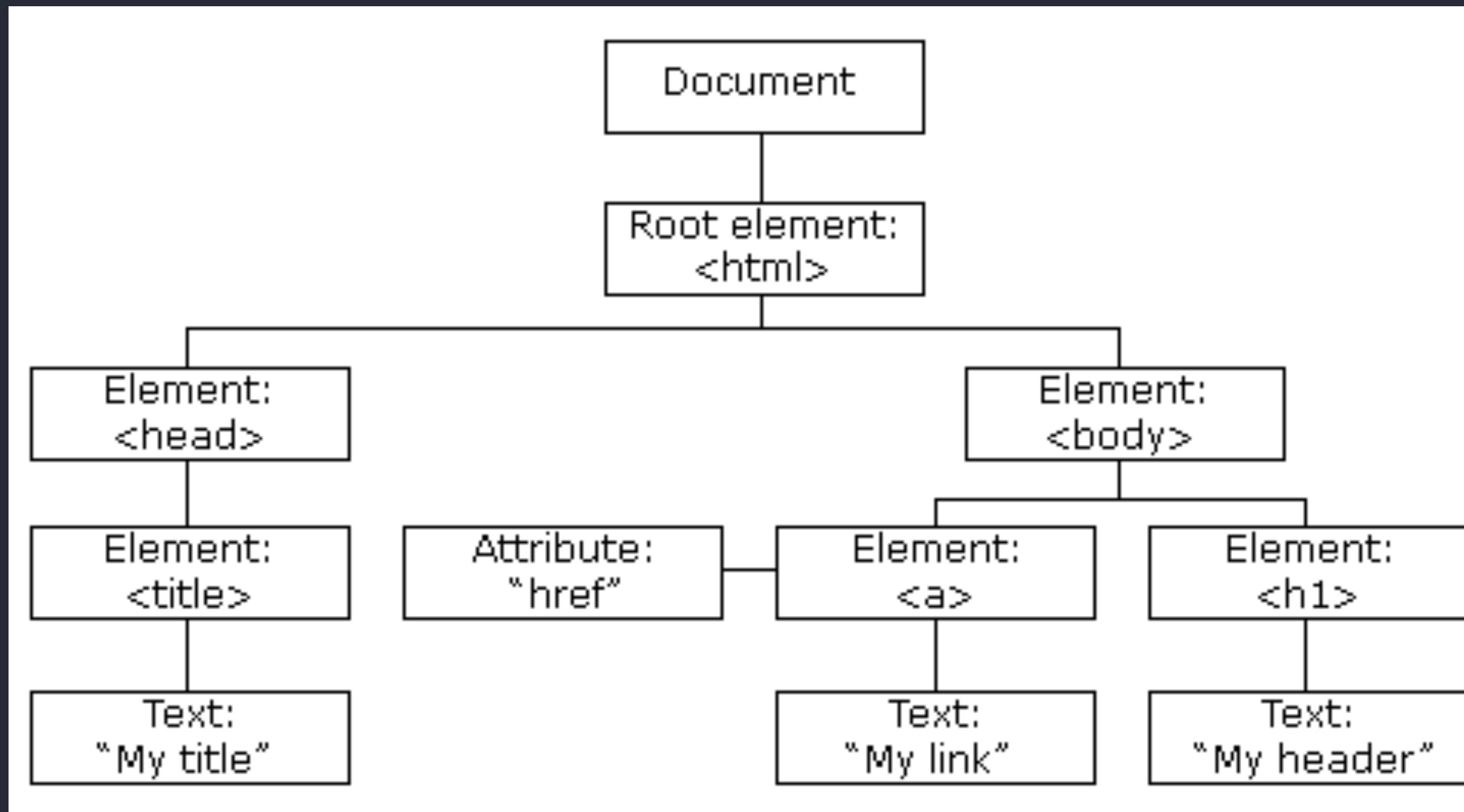
      <article>
        
        <h2>Draco Malfoy</h2>
        <p>Slytherin</p>
      </article>

      <article>
        
```

The DOM is represent the updated version of the (page) content, which can be updated and manipulated by JavaScript.

The HTML DOM (Document Object Model)

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

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title>Page Title</title>
5      <link rel="stylesheet" href="styles.css" />
6    </head>
7    <body>
8      <h1>This is a Heading</h1>
9      <p>This is a paragraph.</p>
10     <button onclick="tryMe()">Try me</button>
11     <script src="app.js"></script>
12   </body>
13 </html>
14
```

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.

```
1  function tryMe() {
2    document.body.style.backgroundColor = "red";
3    document.body.style.color = "white";
4  }
5
```

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Page Title</title>
5          <link rel="stylesheet" href="styles.css" />
6      </head>
7      <body>
8          <h1>This is a Heading</h1>
9          <p>This is a paragraph.</p>
10         <button onclick="tryMe()">Try me</button>
11         <script src="app.js"></script>
12     </body>
13 </html>
```

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.

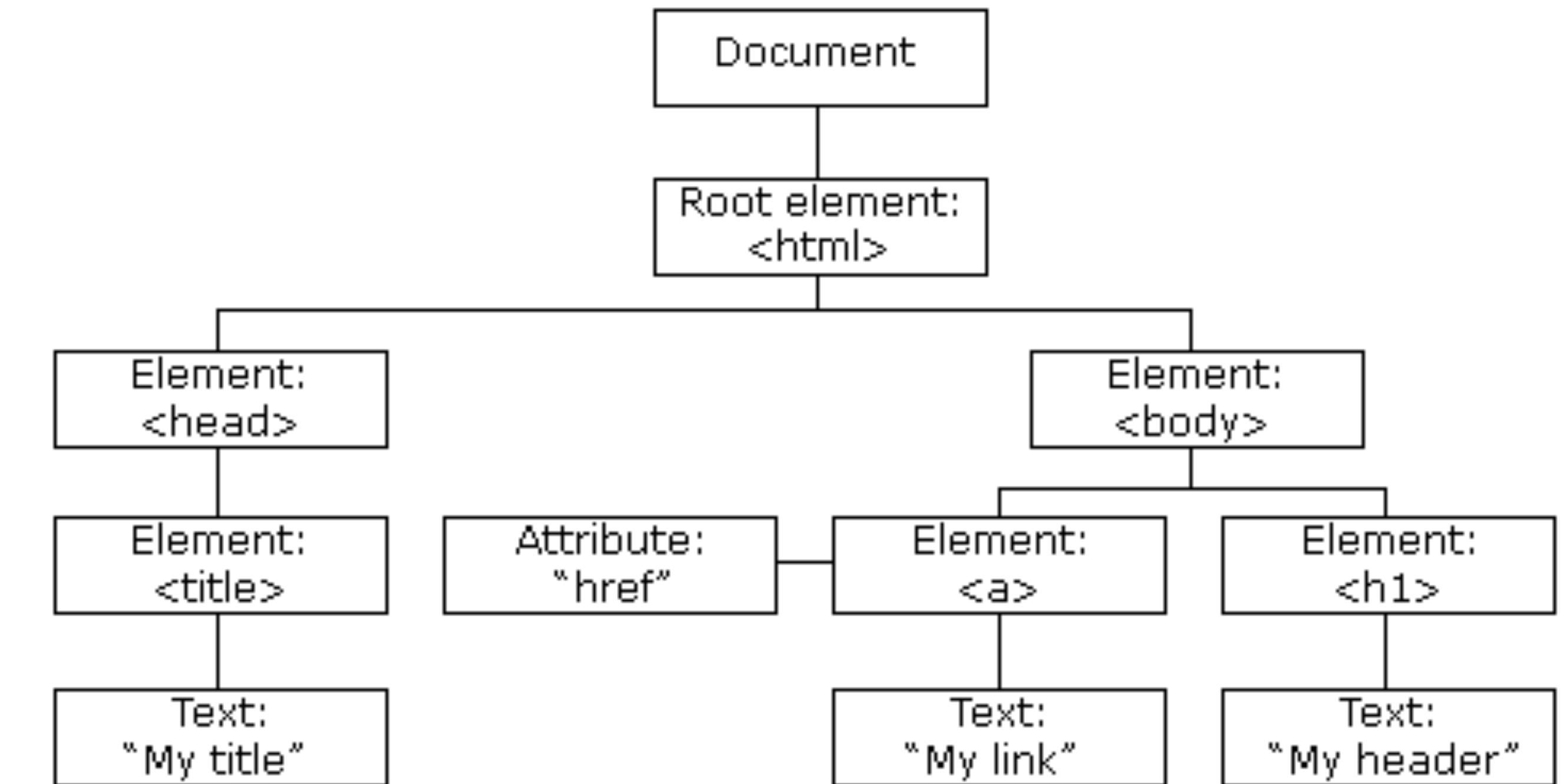
```
1  function tryMe() {
2      document.body.style.backgroundColor = "red";
3      document.body.style.color = "white";
4  }
5
```

... do DOM-manipulation.

... build and develop anything 

JavaScript HTML DOM

```
index.html *  
1  <!DOCTYPE html>  
2  <html>  
3  | <head>  
4  | | <title>My title</title>  
5  | </head>  
6  |  
7  <body>  
8  | | <h1>My header</h1>  
9  | | <a href="https://cederdorff.com">My link</a>  
10 | </body>  
11 |  
12 </html>
```



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

JavaScript HTML DOM

Document Object Model

```
index.html ×  
1  <!DOCTYPE html>  
2  <html>  
3  |   <head>  
4  |   |   <title>My title</title>  
5  |   </head>  
6  
7  <body>  
8  |   <h1>My header</h1>  
9  |   <a href="https://cederdorff.com">My link</a>  
10 |</body>  
11 </html>  
12
```

The HTML document as an object

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

JavaScript can:

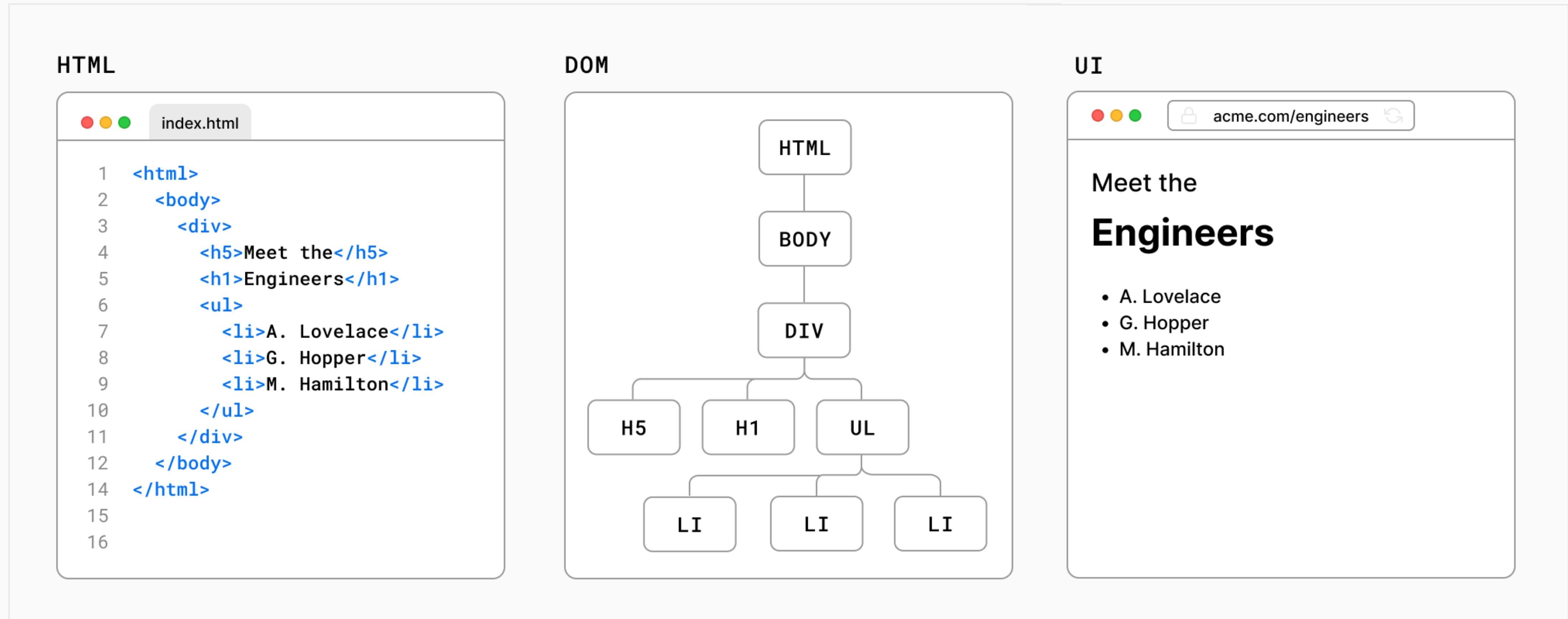
- ... change all the HTML elements in the page*
- ... change all the HTML attributes in the page*
- ... change all the CSS styles in the page*
- ... remove existing HTML elements and attributes*
- ... add new HTML elements and attributes*
- ... react to all existing HTML events in the page*
- ... create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

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

<https://javascript.info/dom-nodes>

<https://javascript.info/dom-navigation>

Example?



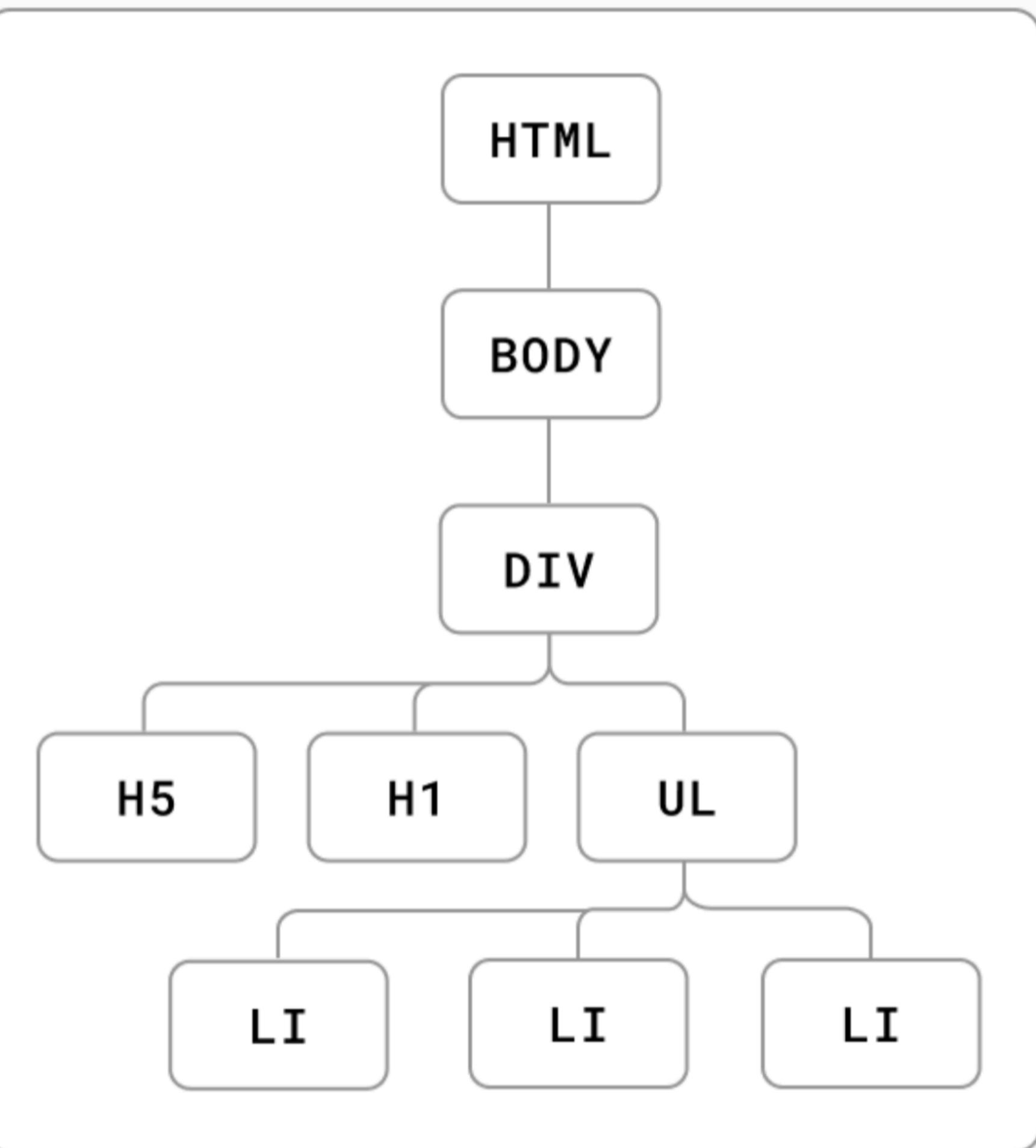
When a user visits a web page, the server returns an HTML file to the browser that may look like this:

HTML

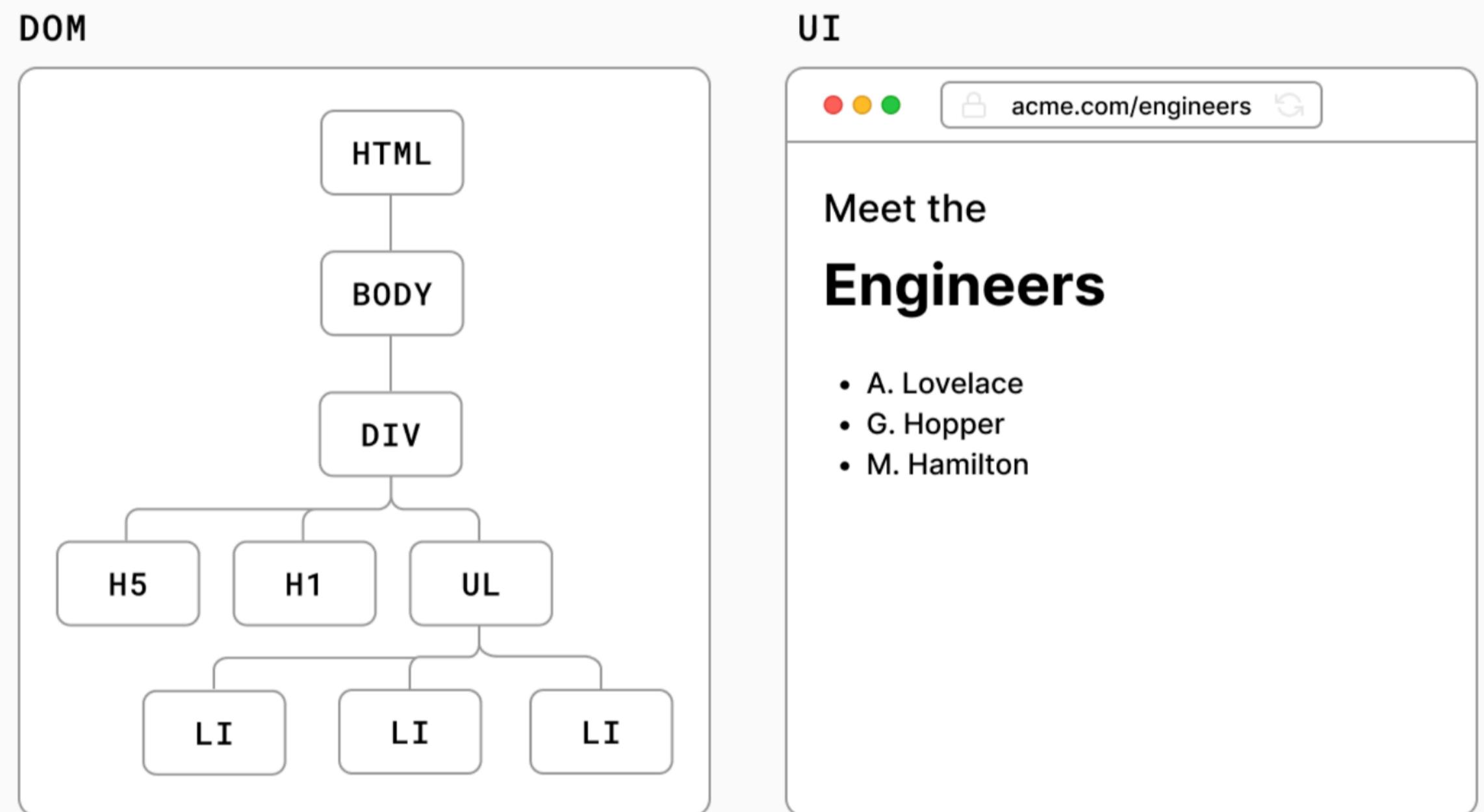
```
index.html  
1 <html>  
2   <body>  
3     <div>  
4       <h5>Meet the</h5>  
5       <h1>Engineers</h1>  
6       <ul>  
7         <li>A. Lovelace</li>  
8         <li>G. Hopper</li>  
9         <li>M. Hamilton</li>  
10        </ul>  
11      </div>  
12    </body>  
14  </html>  
15  
16
```

The browser then reads the HTML and constructs the Document Object Model (DOM).

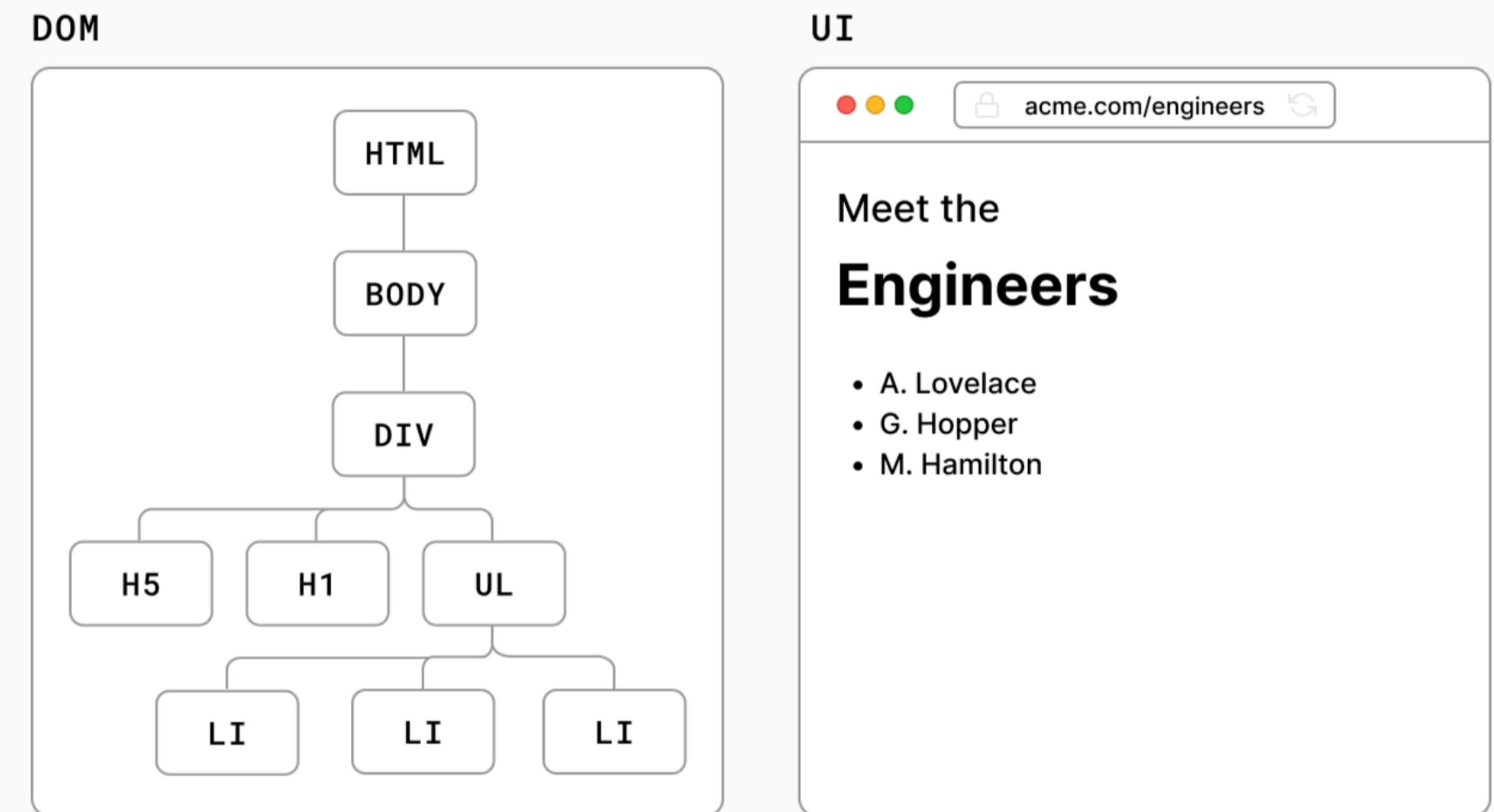
DOM



The DOM is an object representation of the HTML elements. It acts as a bridge between your code and the user interface, and has a tree-like structure with parent and child relationships.



You can use DOM methods and a programming language, such as JavaScript, to listen to user events and manipulate the DOM by selecting, adding, updating, and deleting specific elements in the user interface. DOM manipulation allows you to not only target specific elements, but also change their style and content.



Updating the UI with JavaScript and DOM Methods

```
// Create a new H1 element
const header = document.createElement("h1");

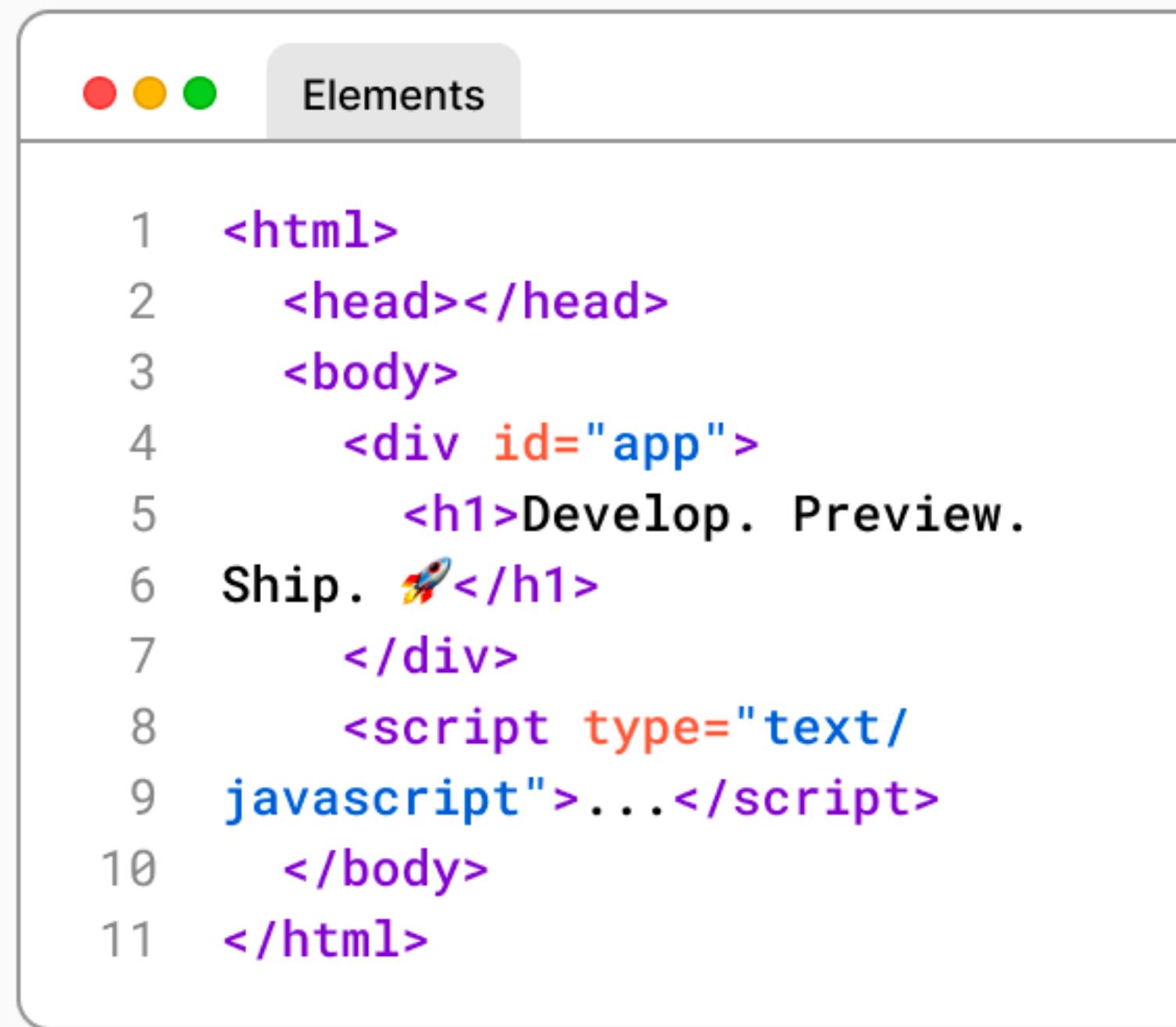
// Add content to H1 element
header.textContent = "Develop. Preview. Ship. 🚀";

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

Let's try
it out!

DOM VS Source Code

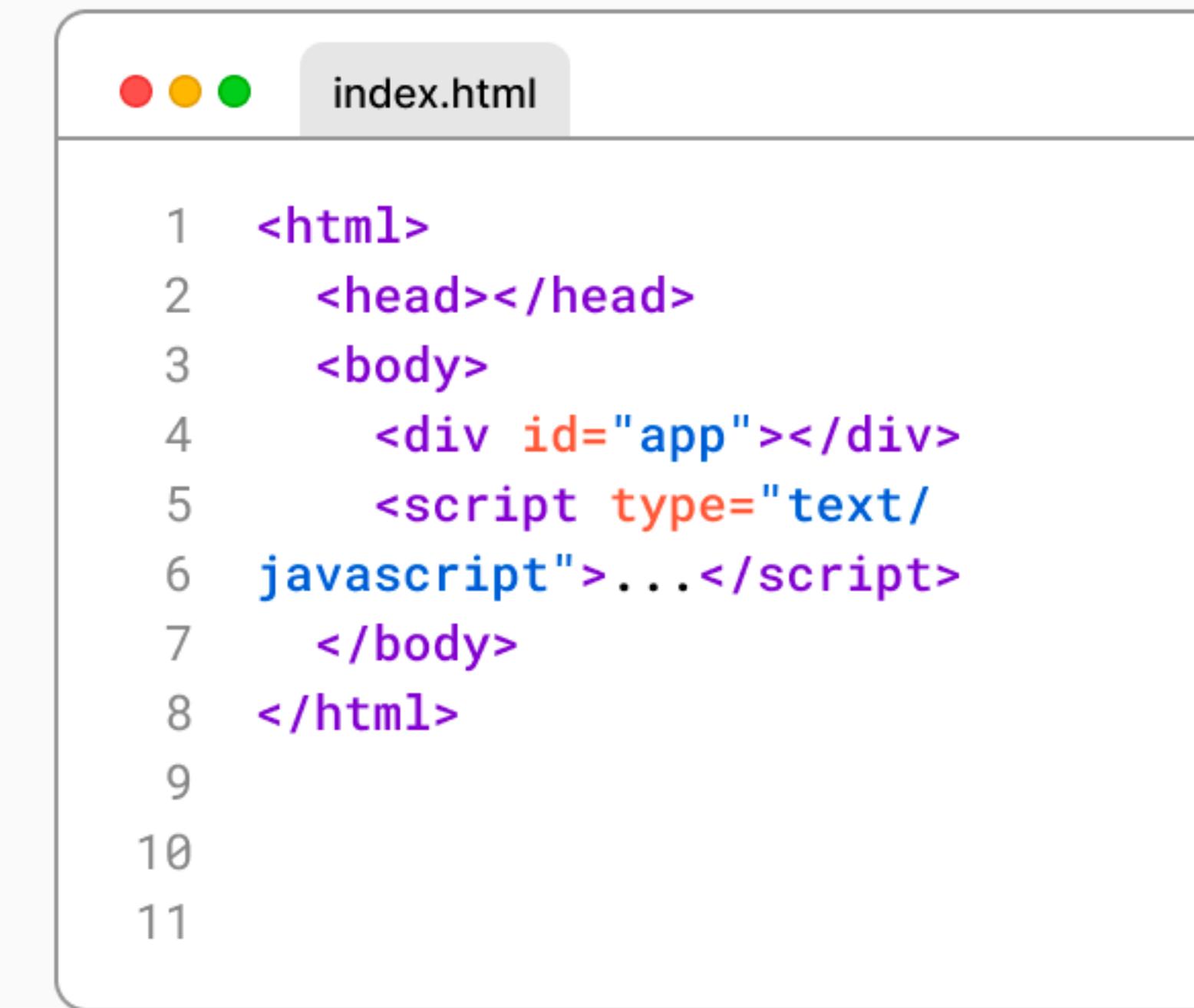
DOM



The screenshot shows the DOM tab of a browser's developer tools. It displays a hierarchical tree of elements, starting with the <html> root node. The <body> node contains a <div id="app"> element, which in turn contains an <h1> element with the text "Develop. Preview. Ship. 🚀". A script tag is also present within the body.

```
1 <html>
2   <head></head>
3   <body>
4     <div id="app">
5       <h1>Develop. Preview.
6       Ship. 🚀</h1>
7       </div>
8       <script type="text/
9 javascript">...</script>
10      </body>
11    </html>
```

SOURCE CODE (HTML)



The screenshot shows the Source Code tab of a browser's developer tools, displaying the raw HTML source code of the file index.html. The code is identical to the DOM structure shown above, with minor differences in line numbers and styling.

```
1 <html>
2   <head></head>
3   <body>
4     <div id="app"></div>
5     <script type="text/
6 javascript">...</script>
7   </body>
8 </html>
9
10
11
```

DOM Manipulation

“

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

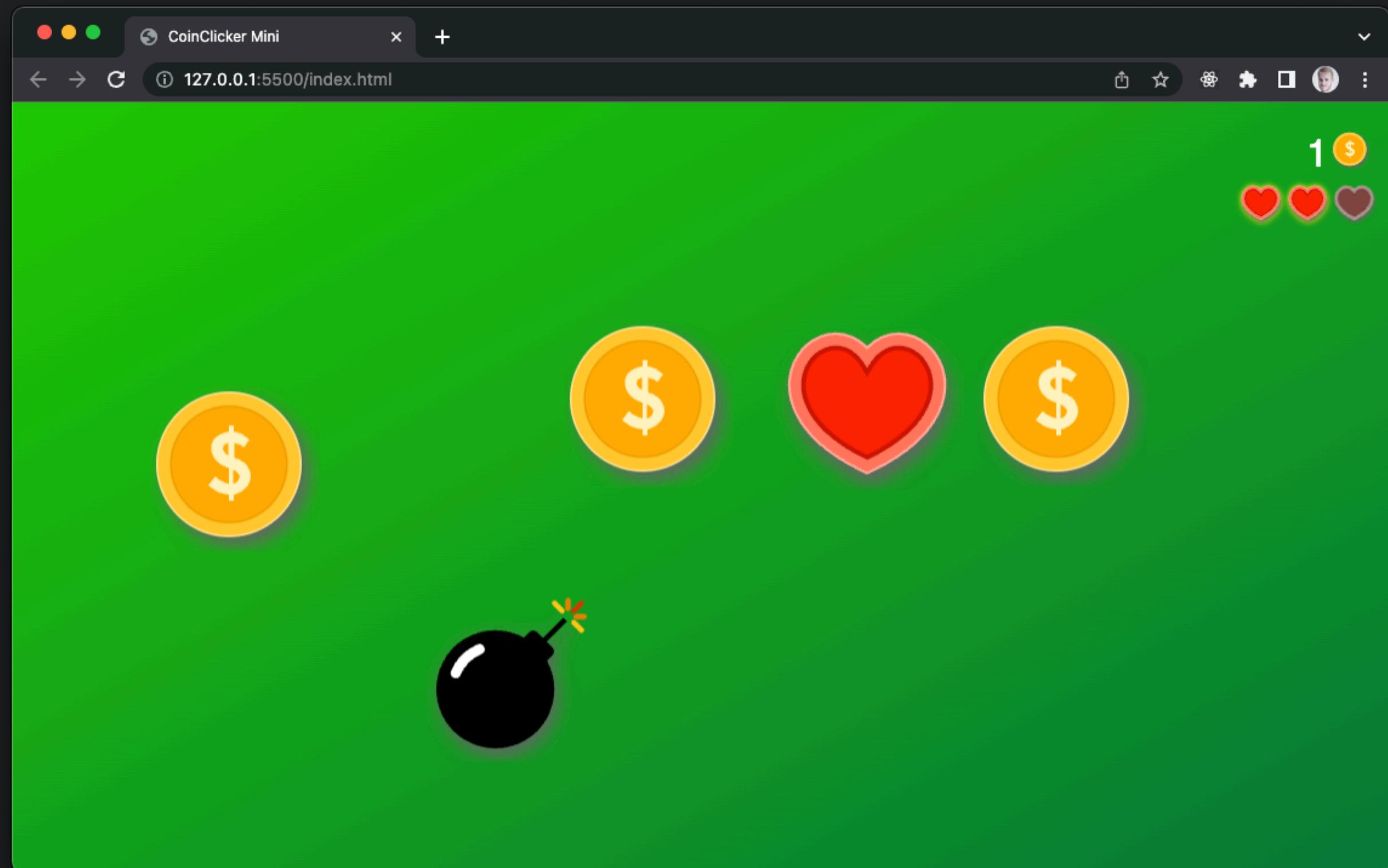
”

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

Methods for DOM Manipulation

Common Methods

I Clicker-spillet?



DOM Manipulation

```
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>
```



DOM Manipulation

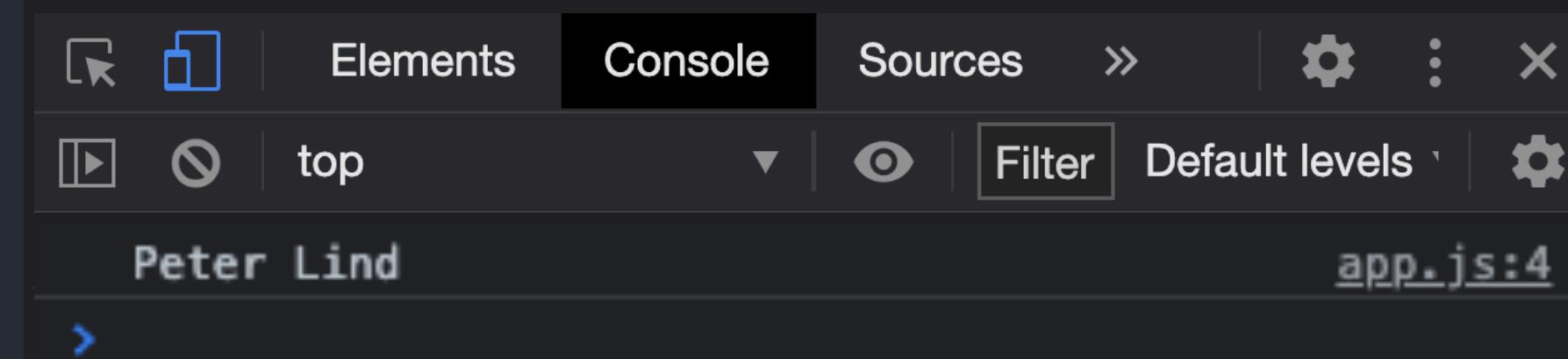
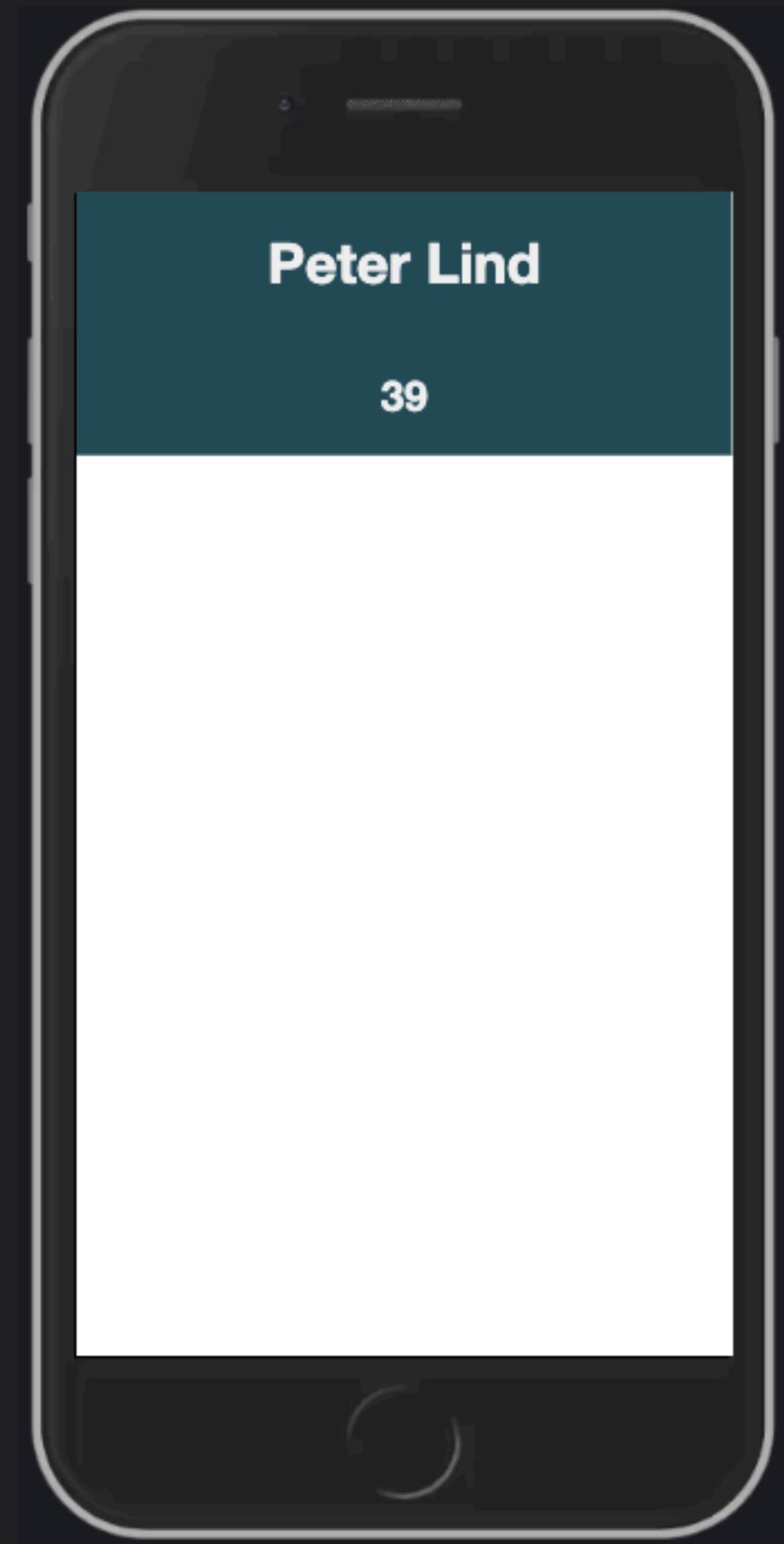
```
let fullName = "Peter Lind";
console.log(fullName);
document.querySelector("#fullName_container").textContent = fullName;
```

.textContent

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

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;
```



DOM Manipulation

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

let message = `${fullName} is ${age} years old.`;
console.log(message);
document.querySelector("#message_container").textContent = message;
```

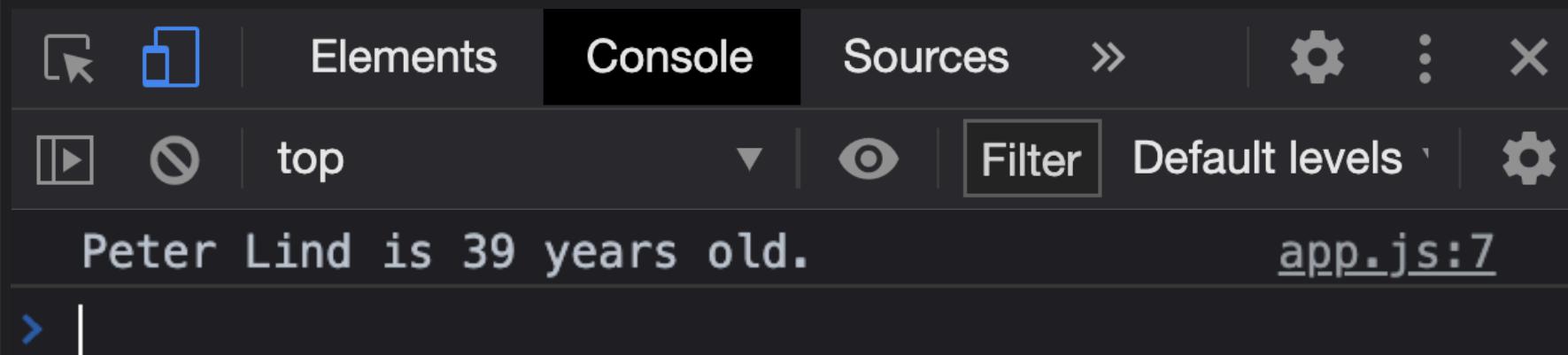
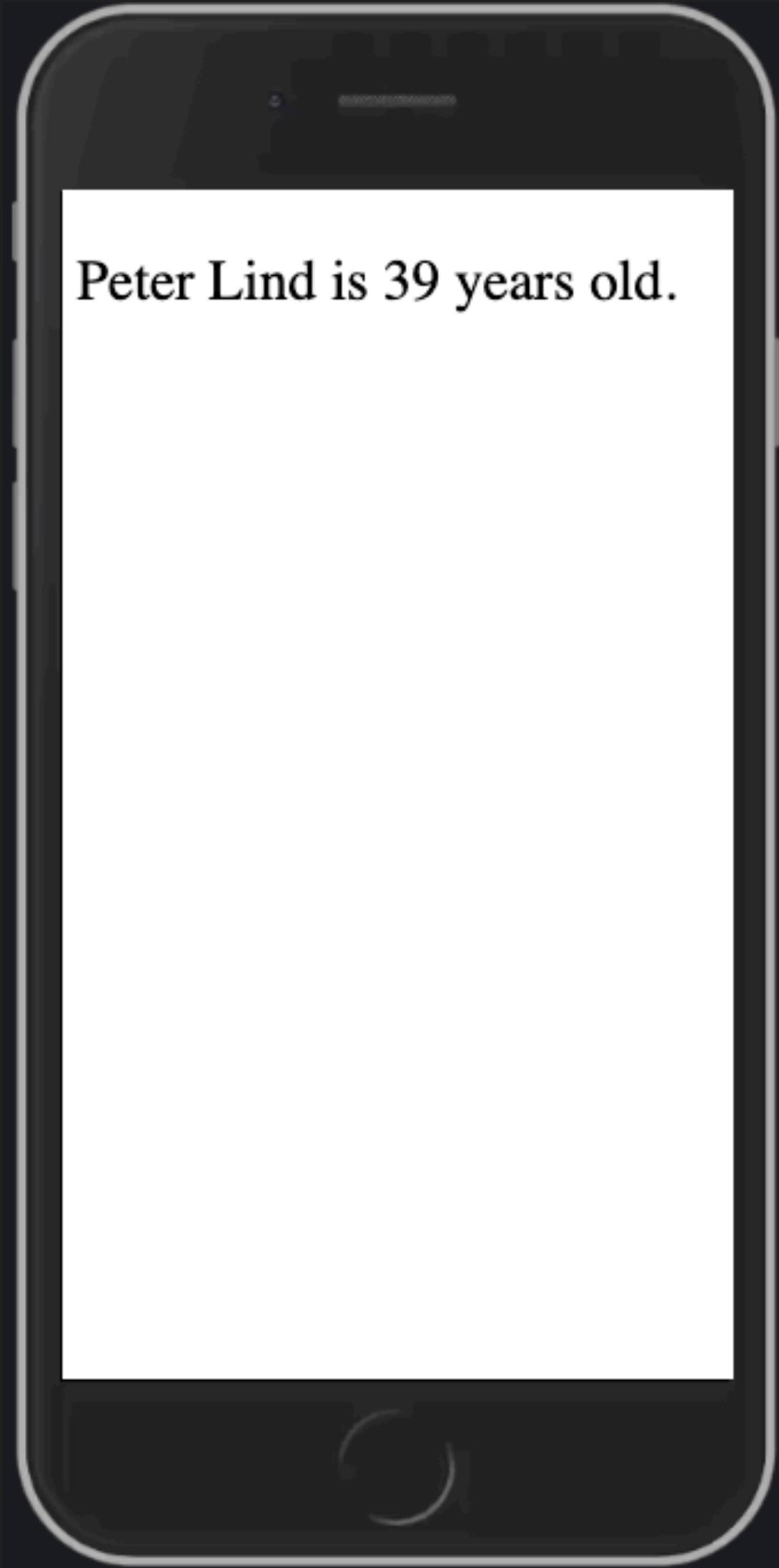
```
<body>
  <main>
    <p id="message_container"></p>
  </main>
  <script src="app.js"></script>
</body>
```



DOM Manipulation

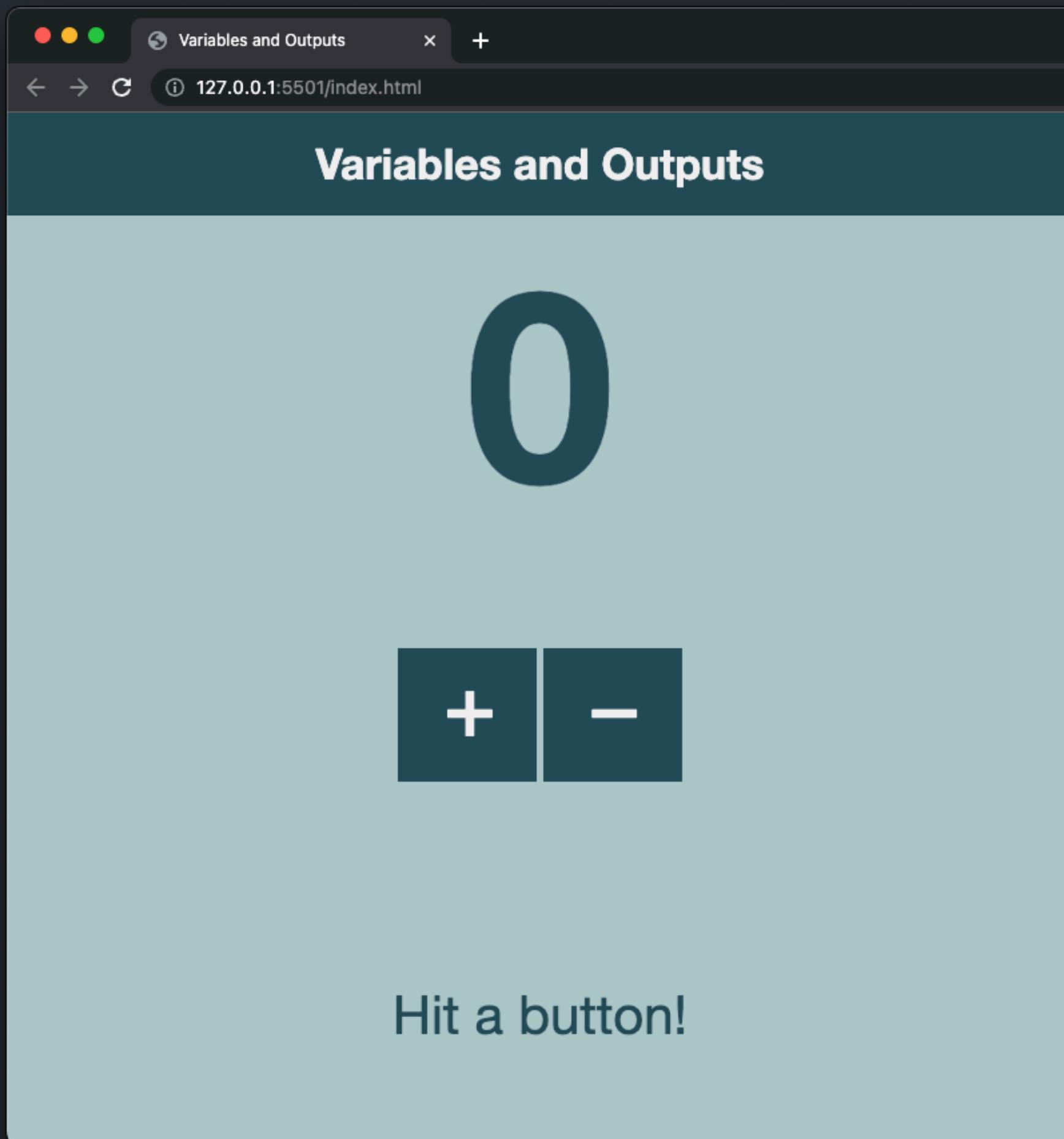
```
let fullName = "Peter Lind";
let age = 39;

let message = `${fullName} is ${age} years old.`;
console.log(message);
document.querySelector("#message_container").textContent = message;
```

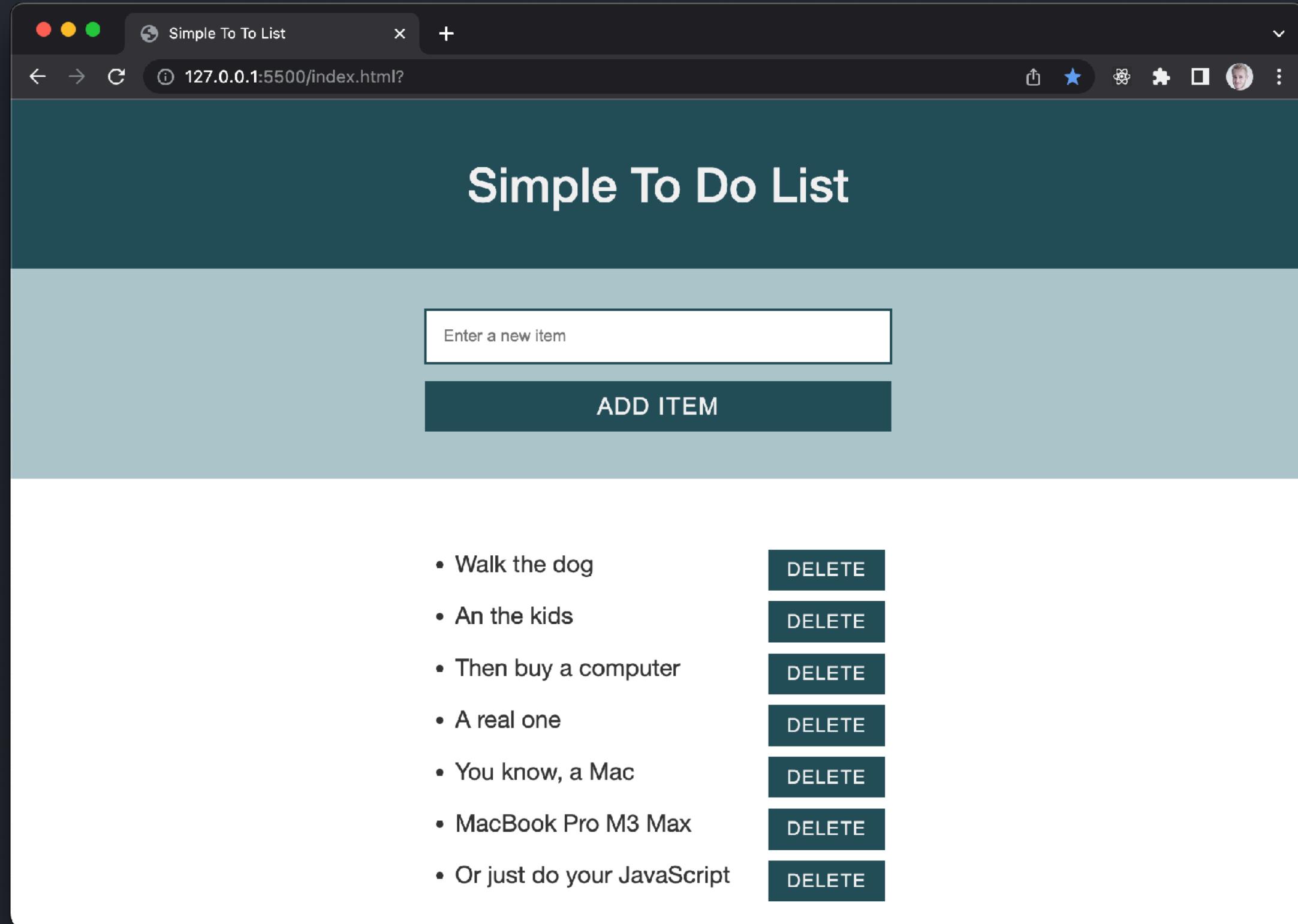


Udskrive til DOM'en

```
let variableName = 0;  
variableName++; //lægger 1 til den nuværende værdi  
variableName--; //trækker 1 fra den nuværende værdi  
variableName += x; //lægger værdien x til den nuværende værdi  
variableName -= x; //trækker værdien x fra den nuværende værdi  
  
document.querySelector("#mitElement").textContent = variableName;
```



Hit a button!



DOM Manipulation

Simple To Do List

[Link til øvelse](#)

Struktur - Del 1

```
"use strict";

window.addEventListener("load", initApp);

const list = document.querySelector("#to-do-list");

function initApp() {
    document.querySelector("#btn-add-item").addEventListener("click", addToDo);
}

function addToDo() {
    // create li
    // create span
    // create button (delete)

    // add textContent to span
    // add textContent to button

    //append span to li
    //append button to li

    //append li to list

    // add click event to button
}

function removeToDo() {
    // remove this.parentNode
}
```

DOM Manipulation Simple To Do List

[Link til øvelse](#)

document.createElement()

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

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

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

.createElement

Creates an HTML element specified by tagName.

.appendChild

Appends an element as last child of an element.

insertAdjacentHTML(position, html)

```
// 1

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

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

// 2

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

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

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

Backtick string

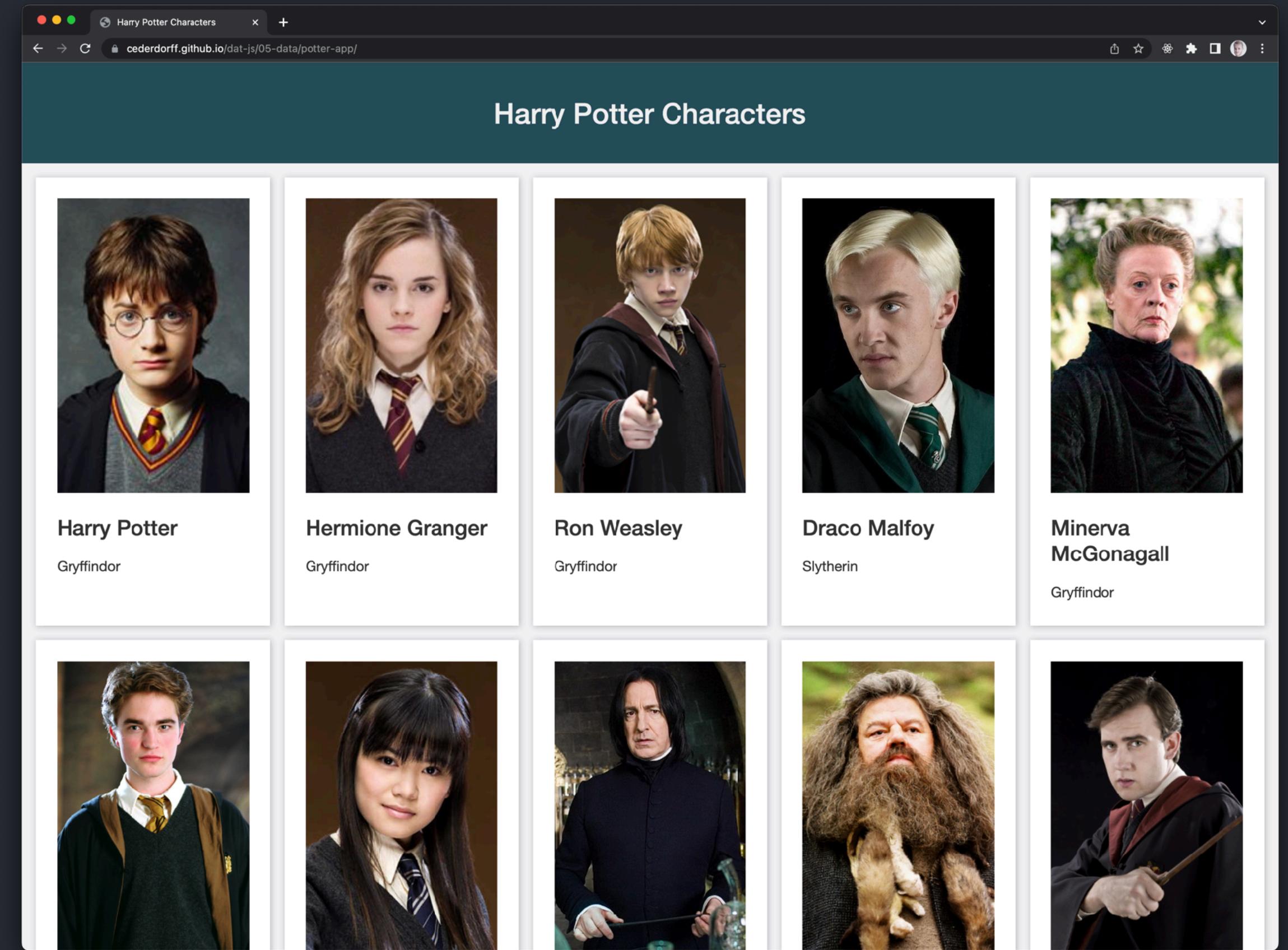
Create multiline strings (templates) with embedded expressions and tags.

.insertAdjacentHTML

Inserts HTML into a specified position.

DOM Manipulation Potter App

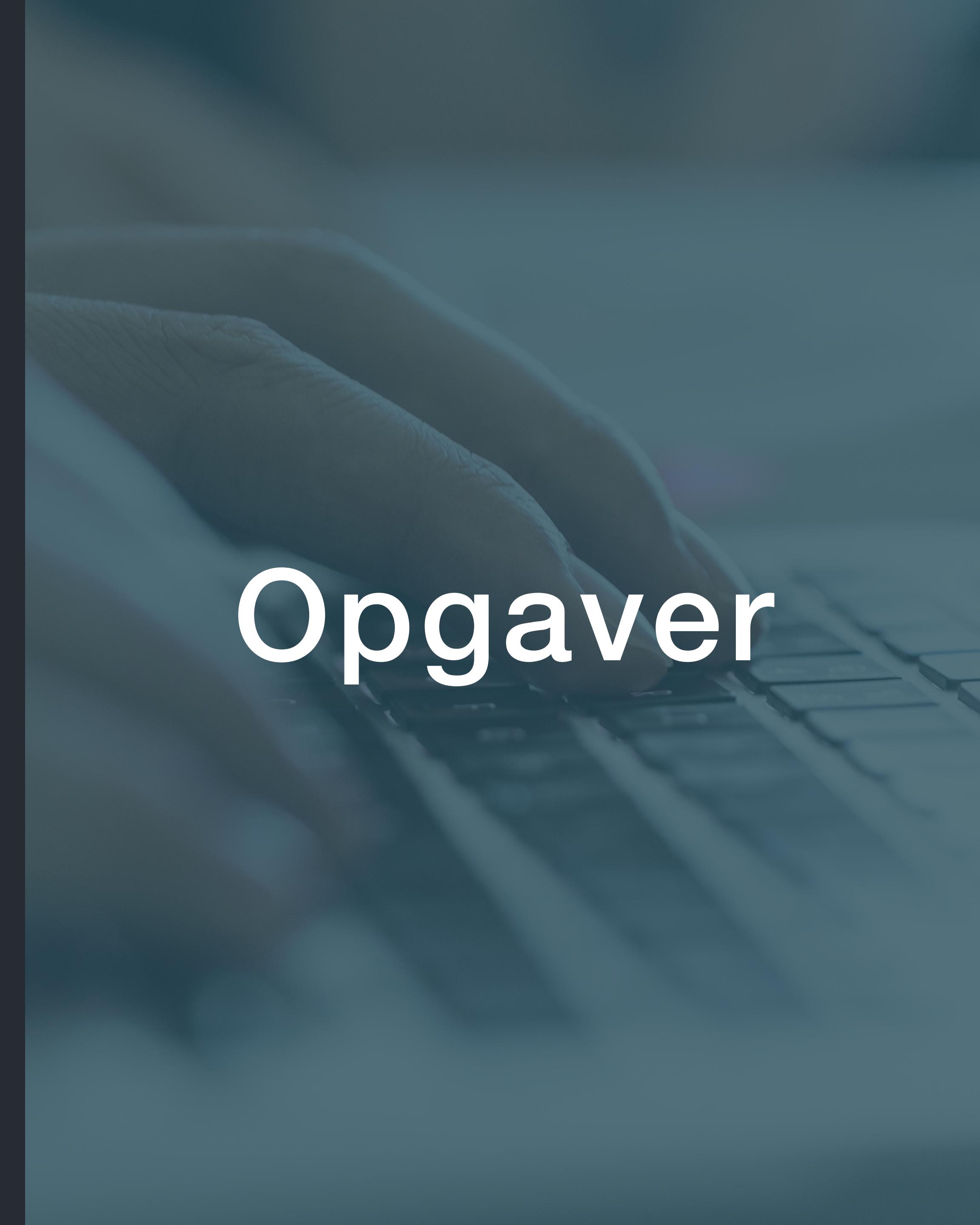
[Link til øvelse](#)



The DOM & DOM Manipulation

- Manipulating documents: [https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Client-side web APIs/Manipulating documents](https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Client-side_web_APIs/Manipulating_documents)
- Introduction to the DOM: [https://developer.mozilla.org/en-US/docs/Web/API/Document Object Model/Introduction](https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction)
- Using the Document Object Model: [https://developer.mozilla.org/en-US/docs/Web/API/Document object model/Using the Document Object Model](https://developer.mozilla.org/en-US/docs/Web/API/Document_object_model/Using_the_Document_Object_Model)
- Modifying the document: <https://javascript.info/modifying-document>

- Simple To Do List
- DOM Manipulation - Potter App
- Ekstra:
 - Active learning: Basic DOM manipulation
 - Active learning: A dynamic shopping list



Opgaver



Code
Every
Day