Code Nation

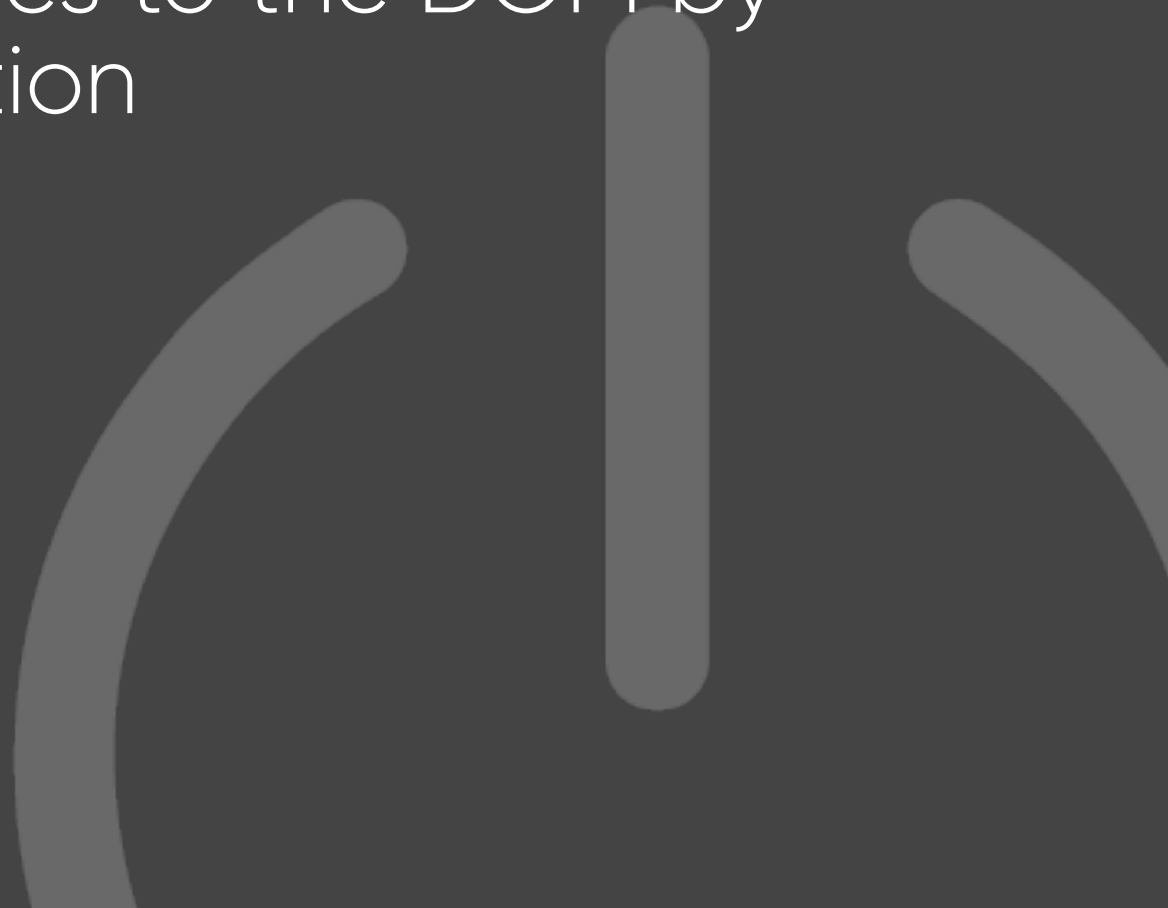
JS and the DOM

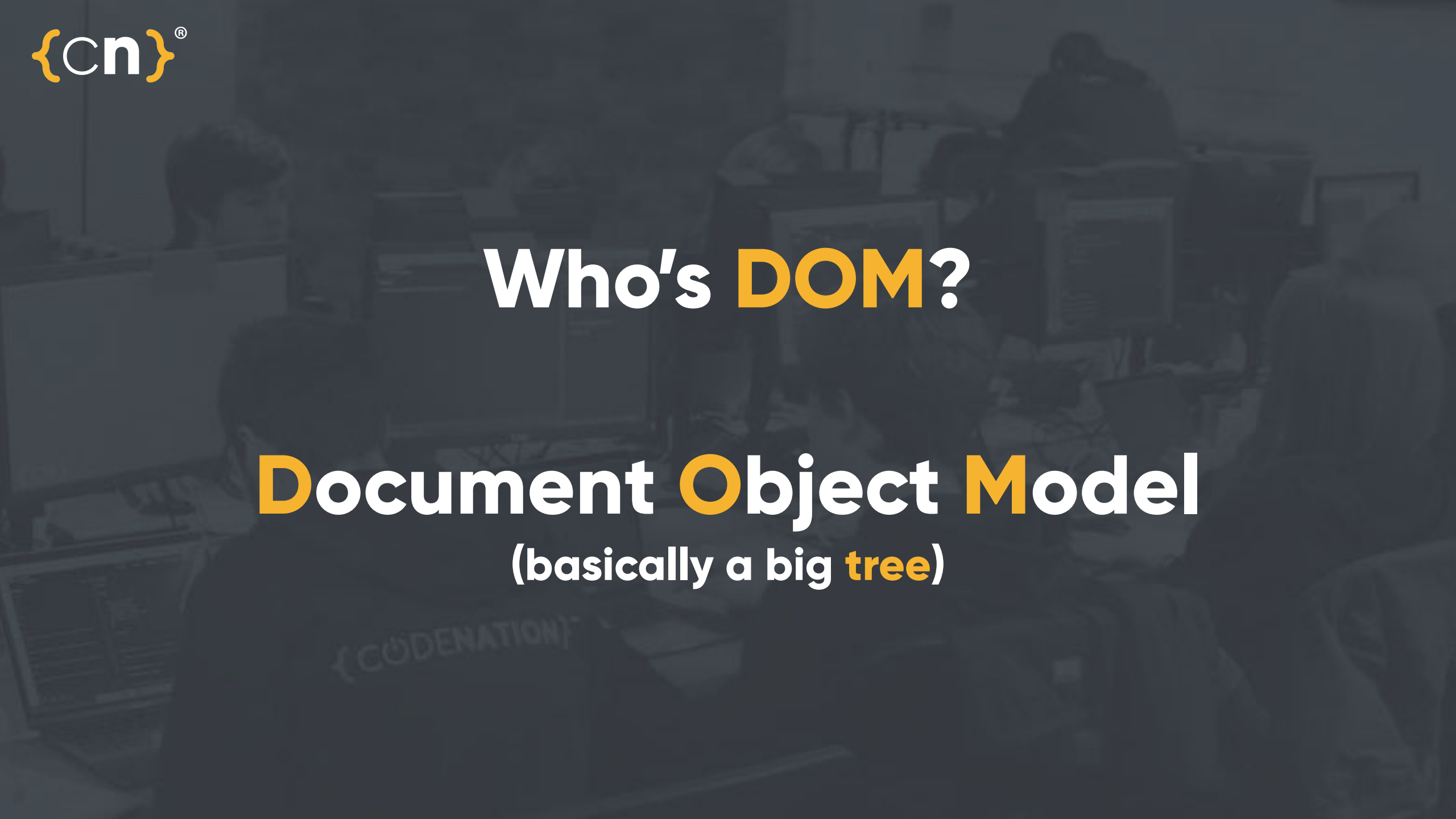
Part 1: tasks 1-7

{codenation}®

Learning Objectives

- Understand the html and DOM structure
- To be able to apply changes to the DOM by responding to user interaction




{cn}®
Who's DOM?
Document Object Model
(basically a big tree)

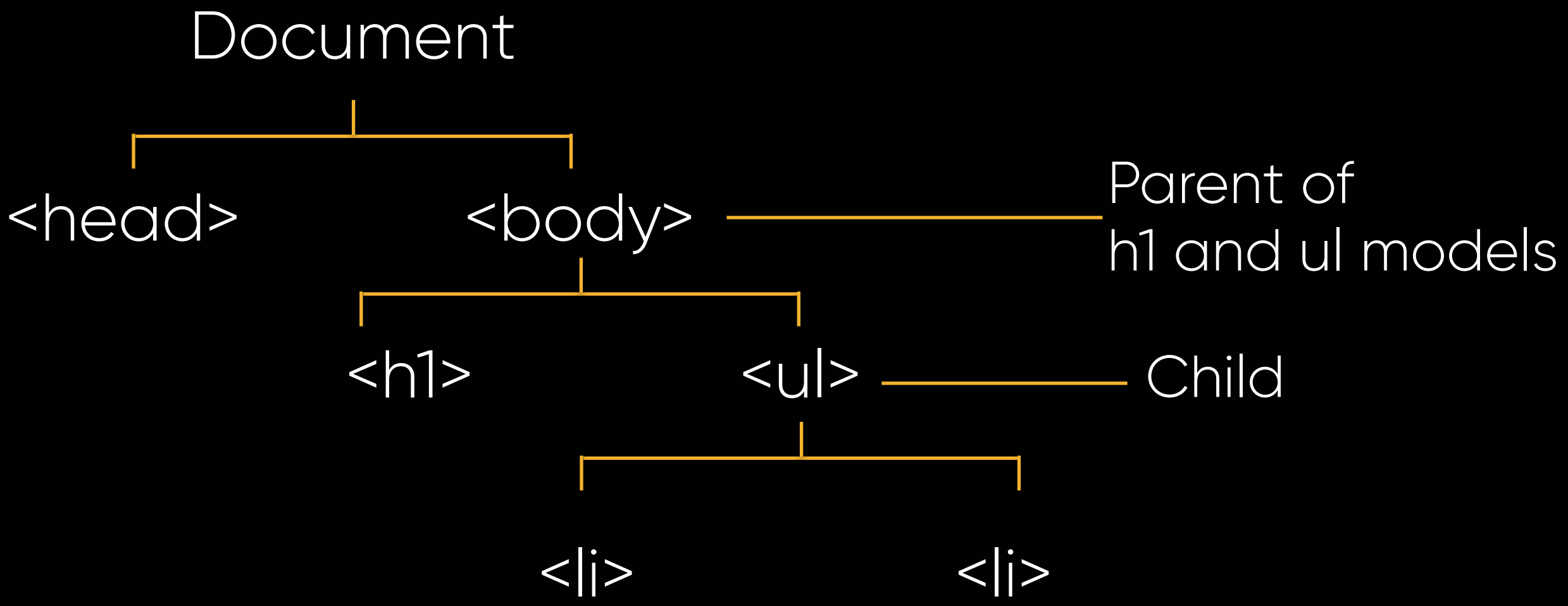
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Javascript and the DOM</title>
</head>
<body>
    <h1> Javascript and the DOM</h1>
    <p>Take control of the web page
        <a href="#">link to a page</a>
    </p>
    <ul>
        <li>First</li>
        <li>Second</li>
        <li>Third</li>
    </ul>
</body>
</html>
```

DOM represents a webpage as a **tree-like structure** (like a family tree)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Javascript and the DOM</title>
</head>
<body>
    <h1> Javascript and the DOM</h1>
    <p>Take control of the web page
        <a href="#">link to a page</a>
    </p>
    <ul>
        <li>First</li>
        <li>Second</li>
        <li>Third</li>
    </ul>
</body>
</html>
```

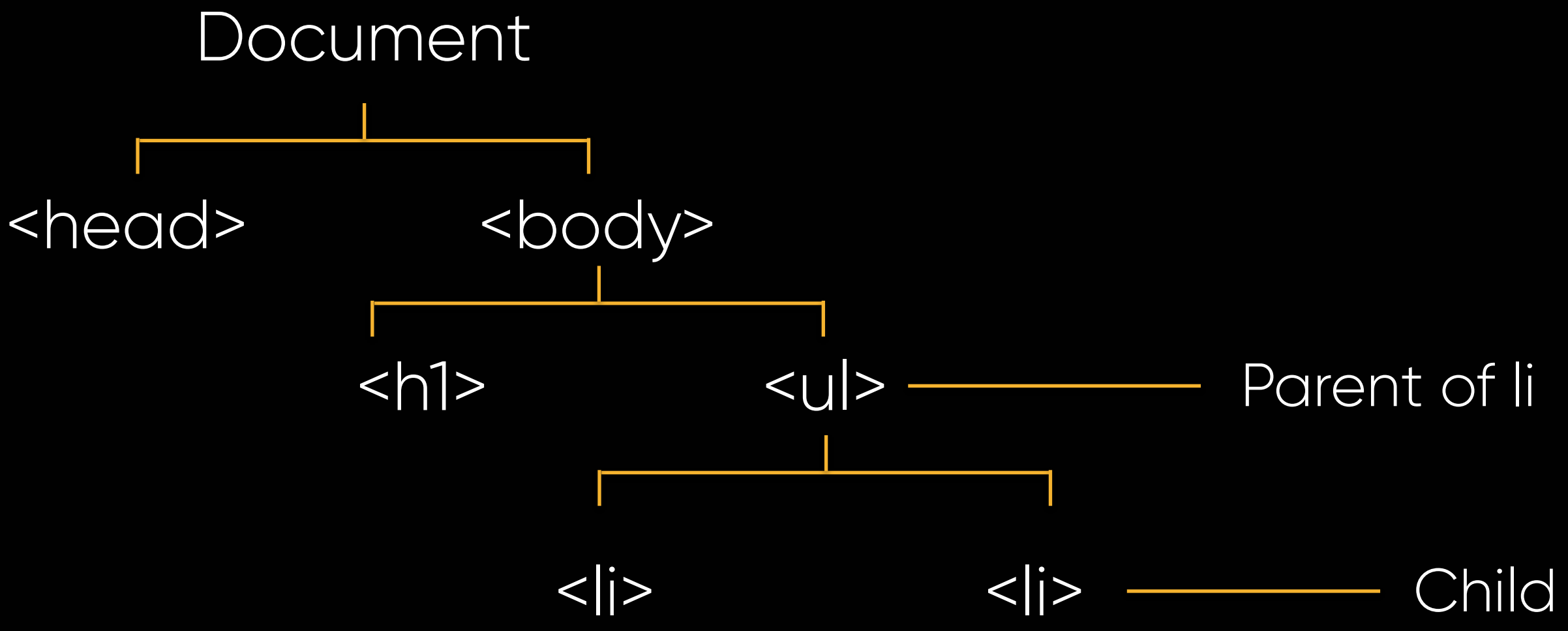
The nested elements within the <html> are what make the tree.

DOM

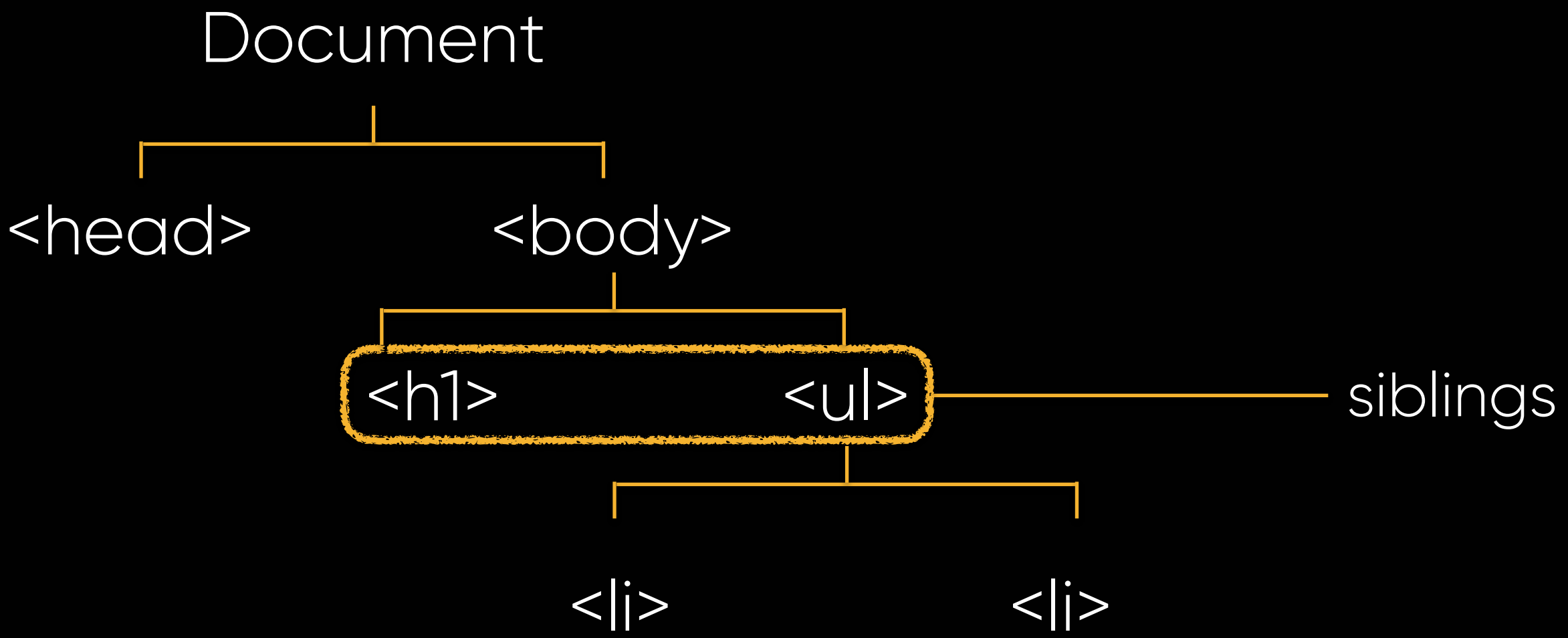

Document
<head>
<body>
Parent of
h1 and ul models
<h1>

Child

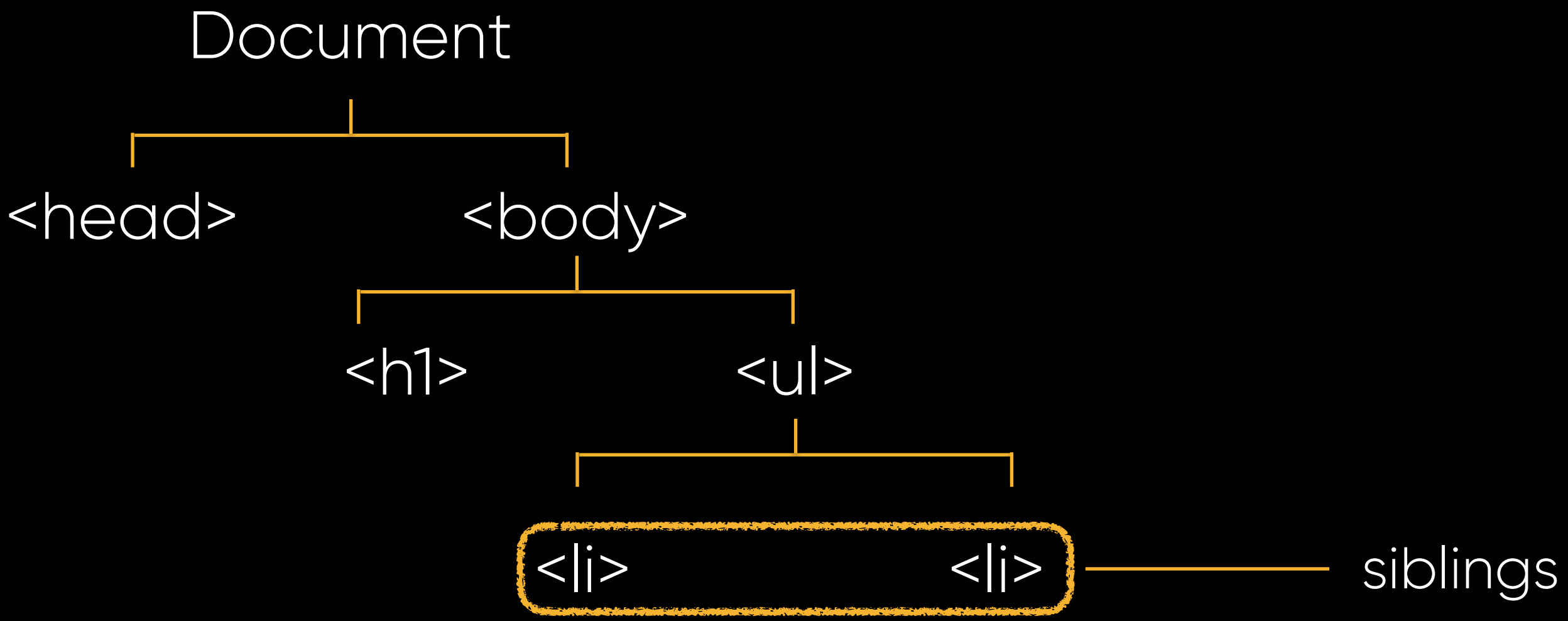
DOM



DOM



DOM



Basic task JS can do with the DOM

- Select an element
- Read or change element
- Respond to user events

Future Learn

Categories Courses Using FutureLearn

{cn}®

Register

Already joined? Sign in.

Register using Facebook

Or

First name *

Last name or initial *

Age *

Email address *

hello

Please check your email address hello to make sure it's correct.

Email address *

hello

Please check your email address hello to make sure it's correct.

Support


{cn}®
The Inspector and the Console.
worlds worst movie title

Activity: alert

- Open Chrome, inspect, and click on Console. (You may want to use the first file)
- Type the below then hit enter:

```
alert("Message me");
```

This is a browser function that shows a message window to the user.

Activity: location

`location.href`

- tells you the location of your file (locally or link to site)

```
> location.href
<- "chrome-search://local-ntp/local-ntp.html"
```

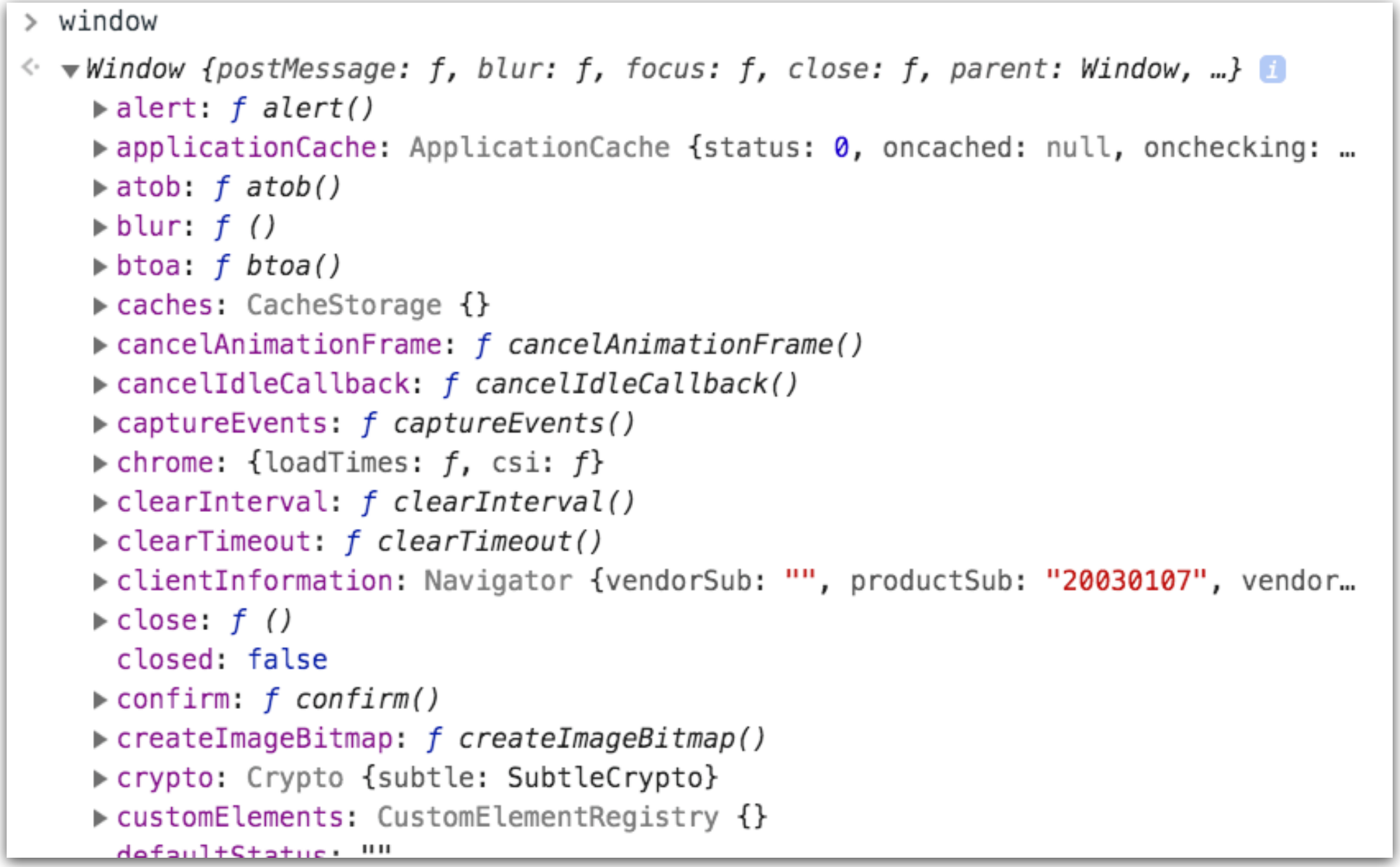
```
> location.href
<- "file:///Users/le/Desktop/mastercoding-4-JS-DOM-1-JS-DOM-files/index.html"
```

Activity: window

`window`

- shows you all the properties it contains, these are called “Window Object”

```
> window
<· ▶Window {postMessage: ƒ, blur: ƒ, focus: ƒ, close: ƒ, parent: Window, …}
```


> window
Window {postMessage: f, blur: f, focus: f, close: f, parent: Window, …}
alert: f alert()
applicationCache: ApplicationCache {status: 0, oncached: null, onchecking: …
atob: f atob()
blur: f ()
btoa: f btoa()
caches: CacheStorage {}
cancelAnimationFrame: f cancelAnimationFrame()
cancelIdleCallback: f cancelIdleCallback()
captureEvents: f captureEvents()
chrome: {loadTimes: f, csi: f}
clearInterval: f clearInterval()
clearTimeout: f clearTimeout()
clientInformation: Navigator {vendorSub: "", productSub: "20030107", vendor…
close: f ()
closed: false
confirm: f confirm()
createImageBitmap: f createImageBitmap()
crypto: Crypto {subtle: SubtleCrypto}
customElements: CustomElementRegistry {}

Activity: alert command

```
window.alert("Message me");
```

does the same thing as:

```
alert("Message me");
```


{cn}®
The global
document object.

The document global object

- Document is a global object representing the entire page

The document global object

- Document is a global object representing the entire page
- Elements on a page are contained within the document object

The document global object

- Document is a global object representing the entire page
- Elements on a page are contained within the document object
- By using the DOM we can access all of these elements, and do some fun stuff.

The document global object

- Document is a global object representing the entire page
- Elements on a page are contained within the document object
- By using the DOM we can access all of these elements, and do some fun stuff
- Like.. manipulating elements, accessing their data, or listening for global events by using the document object

- **document.element is similar in principal to object.property**


document.getElementId()

index.html

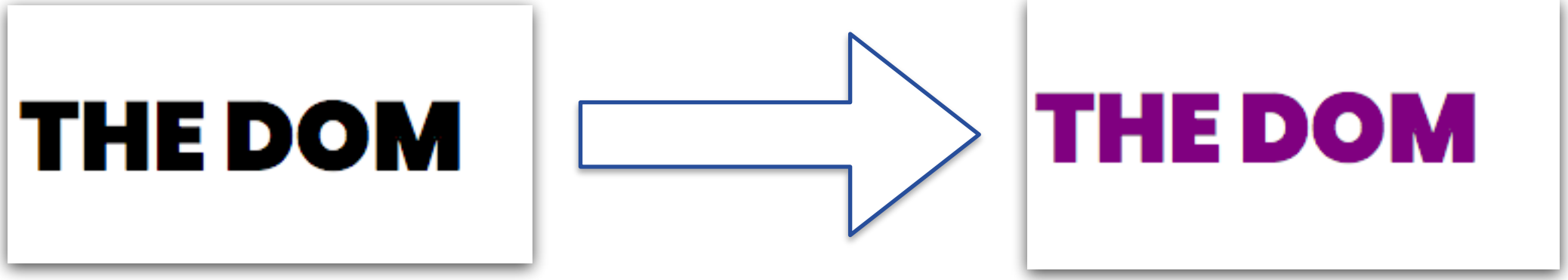
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="../main.css">
    <title>AddEventListener</title>
</head>
<body>
    <h1 id="heading">THE DOM</h1>
</body>
</html>
```

getElementById

```
document.getElementById("heading").style.color = "purple"
```

- Type this in the console.
- Change to different colours, then to black

```
> document.getElementById("heading").style.color = "purple"
<· "purple"
```

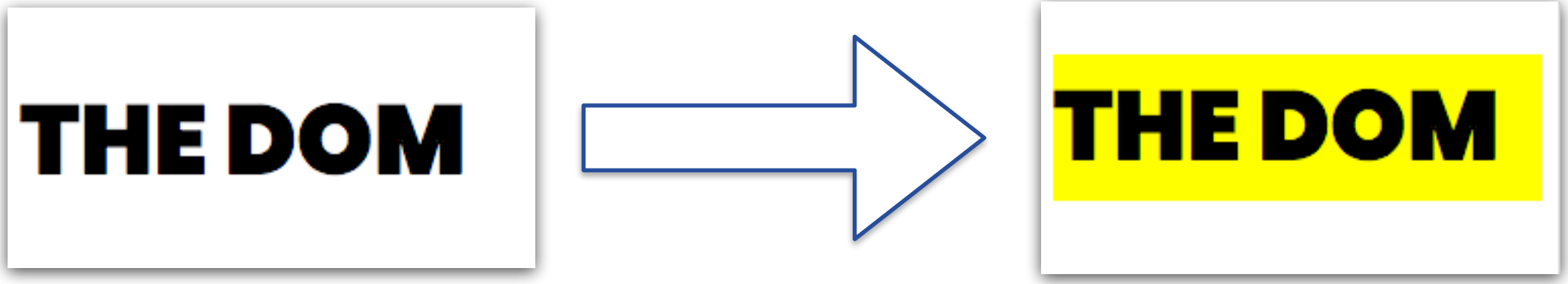


getElementById

```
document.getElementById("heading").style.backgroundColor = "yellow"
```

- Type this in the console.
- Change to different colours, then to white

```
> document.getElementById("heading").style.backgroundColor = "yellow"
<· "yellow"
```



What's the point to use these commands?

- Click the headline to change headline colour
- Actions, e.g. clicking, hovering over a button, scrolling or submitting a form

1 - Add Event Listener

Folder 1

Activity: creating an app.js

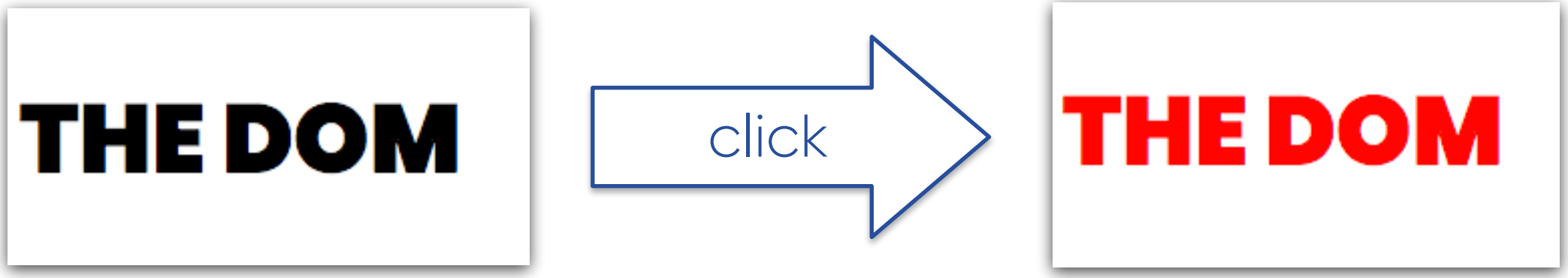
- Inside your folder, create a new file called app.js
- In the html, add below your heading inside the body:

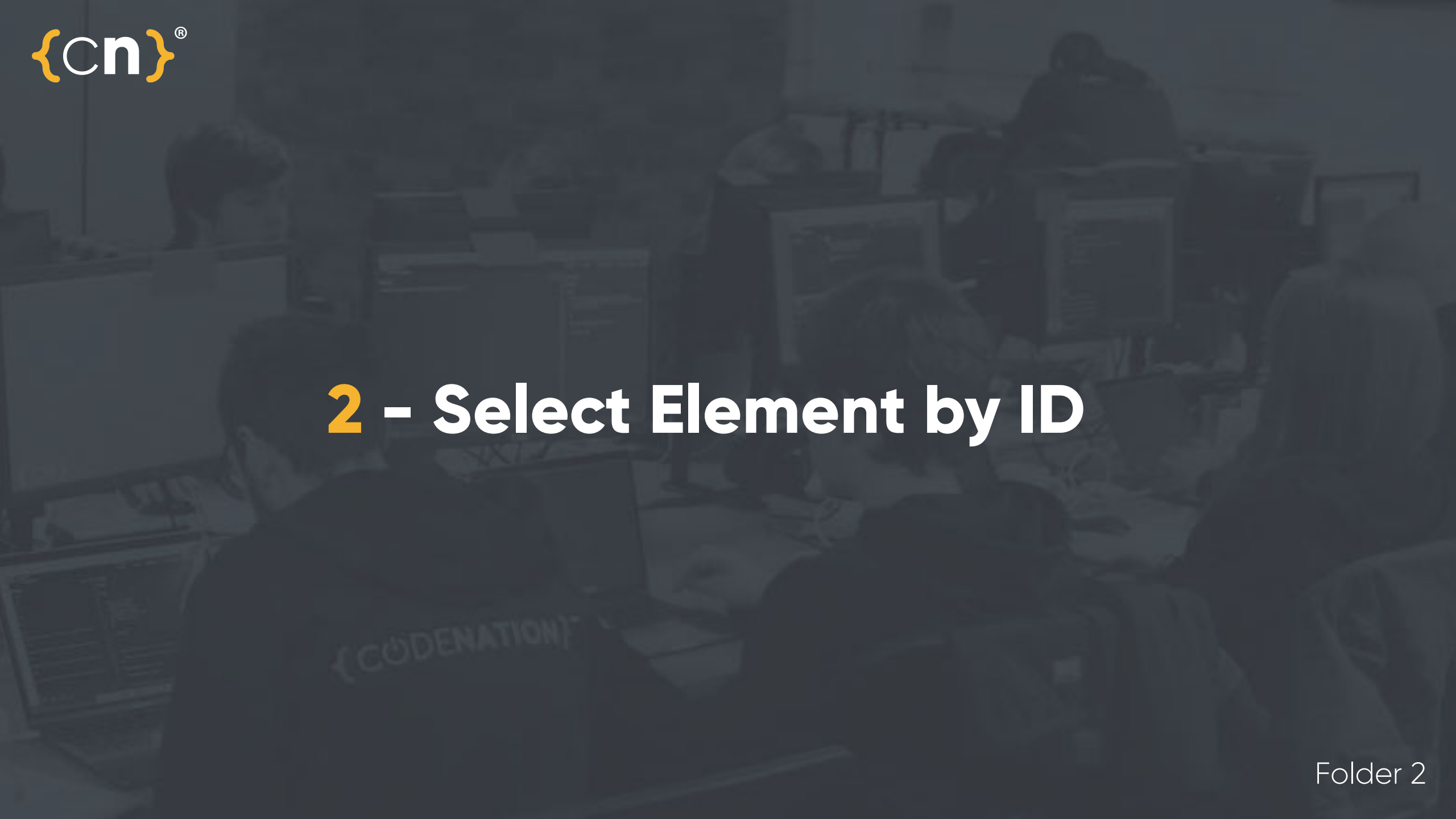
```
<script src="app.js"></script>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="../main.css">
    <title>AddEventListener</title>
</head>
<body>
    <h1 id="heading">THE DOM</h1>
    <script src="app.js"></script>
</body>
</html>
```

Activity: Add inside your app.js

```
const myHeading = document.getElementById("heading");

myHeading.addEventListener("click",()=>{
    myHeading.style.color = "red";
});
```

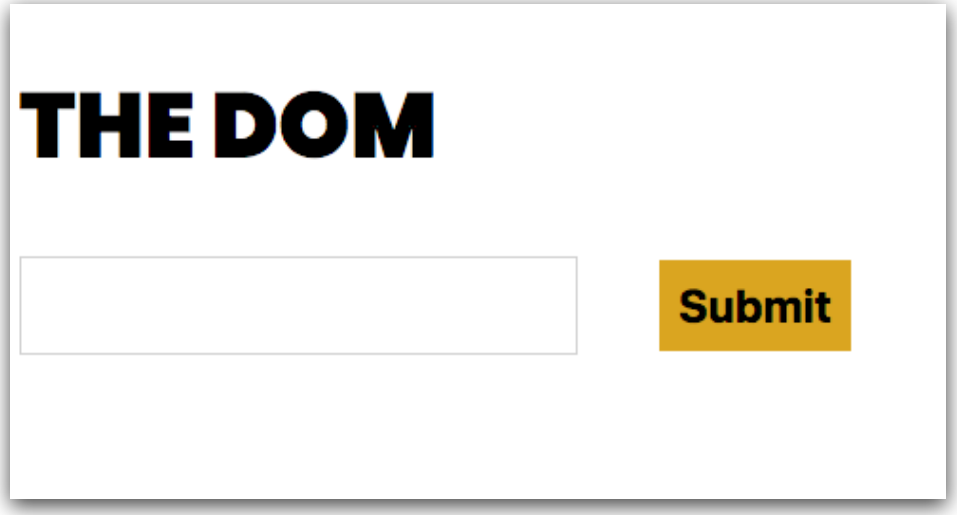




2 - Select Element by ID

Folder 2

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="../main.css">
    <title>Document</title>
</head>
<body>
    <h1 id="heading">THE DOM</h1>
    <input id="input" type="text">
    <button id="button">Submit</button>
    <script src="app.js"></script>
</body>
</html>
```



Activity: setting a variable

- Inside the app.js, store the value of the heading in a const.

```
const heading = document.getElementById("heading");
```

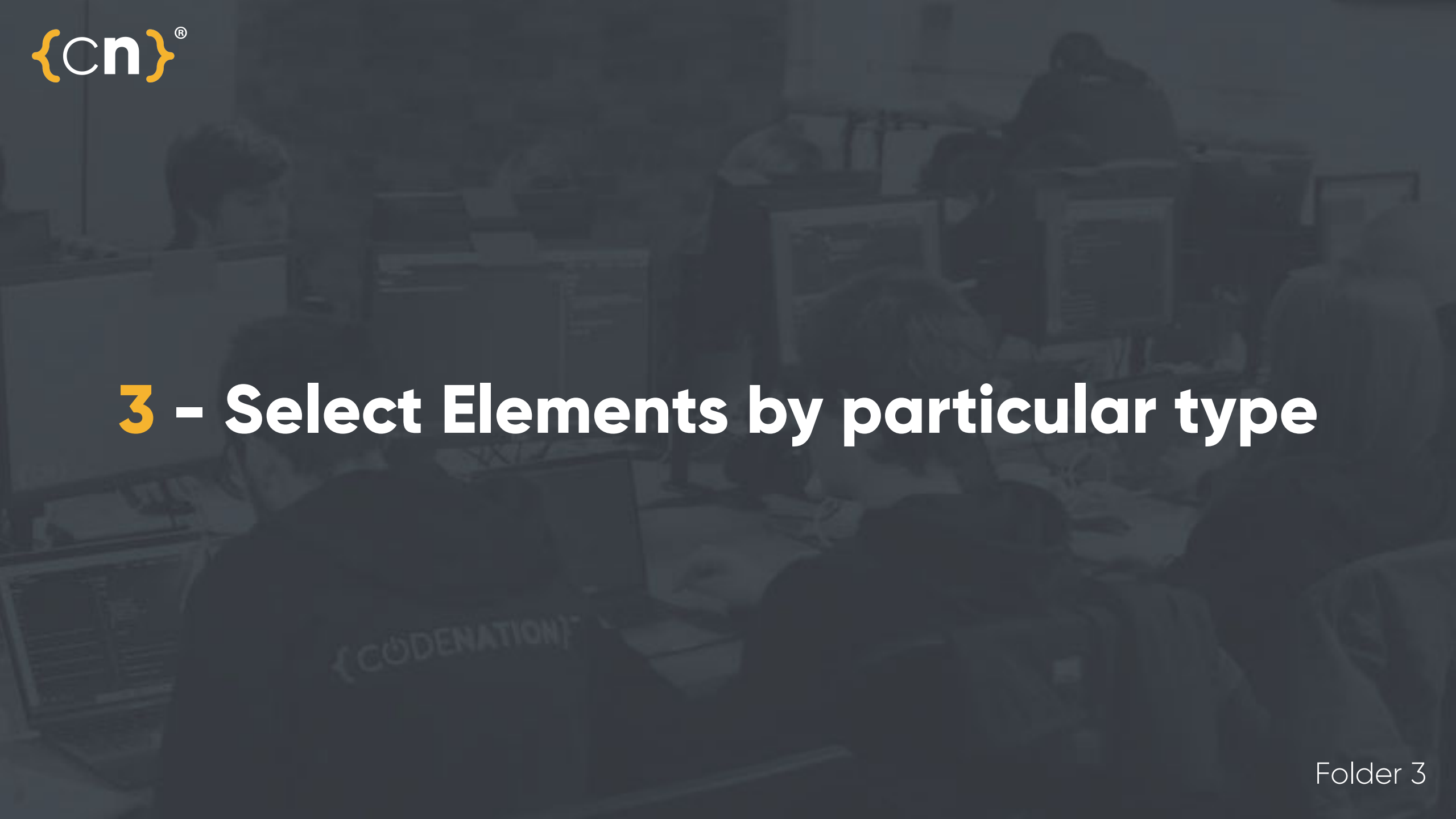
- Do the same for input and button

```
const heading = document.getElementById("heading");
const input = document.getElementById("input");
const button = document.getElementById("button");
```

Activity

- Make the button listen for mouse clicks using the correct method
- Then, get the value from the input value and use that to change the colour of the heading.

```
const heading = document.getElementById("heading");
const input = document.getElementById("input");
const button = document.getElementById(“button");


button.addEventListener("click",()=>{
    heading.style.color = input.value;
});
```



3 - Select Elements by particular type

Folder 3

3 - Select Elements by particular type

document.getElementsByTagName()

Example: getElementsByTagName

index.html

```
<ul id="drinks">
    <li>Tea</li>
    <li>Coffee</li>
    <li>Hot Chocolate</li>
</ul>
```

app.js

```
const drinks = document.getElementById("drinks");
```

Example: getElementsByTagName

index.html

```
<p>
    Lorem ipsum, dolor sit amet consectetur adipisicing elit.
    Atque, aspernatur?
</p>

<p>Lorem ipsum dolor sit amet consectetur adipisicing elit.
    Laborum odio eaque rerum corrupti ipsam debitis accusantium
    fugiat error minus. Sed?
</p>
```

app.js

```
const paragraphs = document.getElementsByTagName("p");
```

Example: getElementsByTagName

index.html

```
<p>
    Lorem ipsum, dolor sit amet consectetur adipisicing elit.
    Atque, aspernatur?
</p>

<p>Lorem ipsum dolor sit amet consectetur adipisicing elit.
    Laborum odio eaque rerum corrupti ipsam debitis accusantium
    fugiat error minus. Sed?
</p>
```

app.js

```
const paragraphs = document.getElementsByTagName("p");
let firstPara = paragraphs[0];
```

First element on the page

Example: getElementsByTagName

index.html

```
<p>
    Lorem ipsum, dolor sit amet consectetur adipisicing elit.
    Atque, aspernatur?
</p>

<p>Lorem ipsum dolor sit amet consectetur adipisicing elit.
    Laborum odio eaque rerum corrupti ipsam debitis accusantium
    fugiat error minus. Sed?
</p>
```

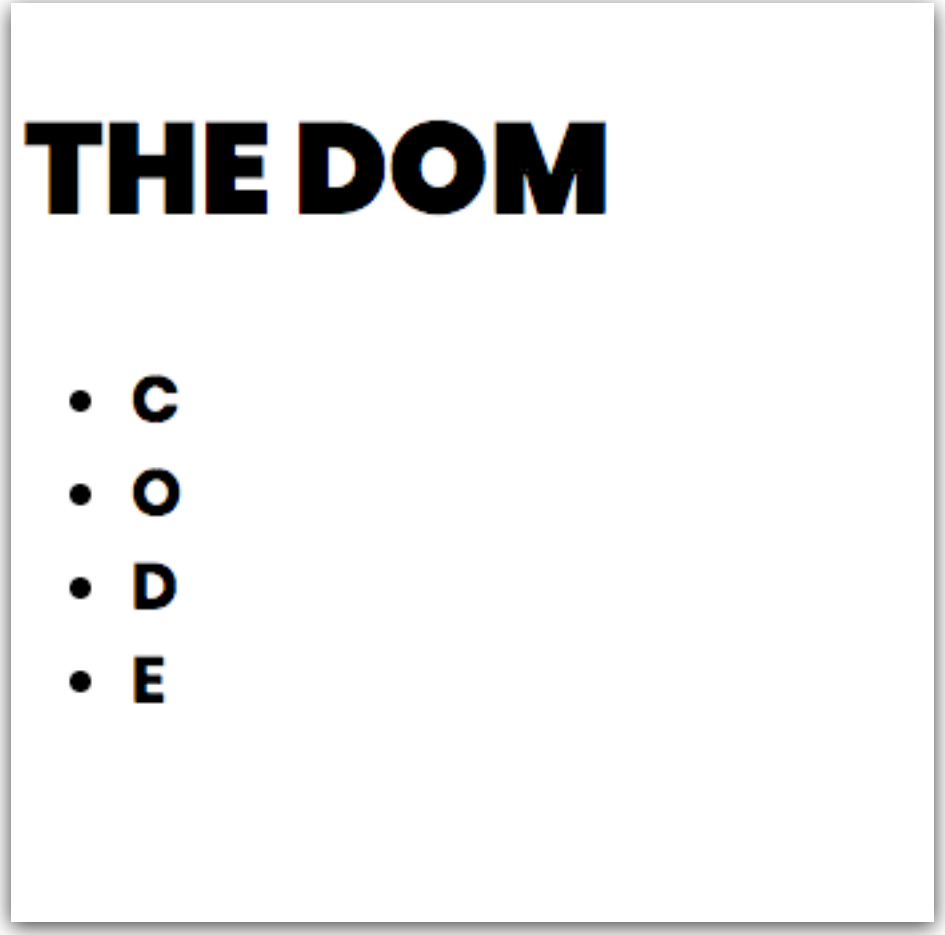
app.js

```
const paragraphs = document.getElementsByTagName("p");
for (let i = 0;  i < paragraphs.length; i++){
    paragraphs[i];
}
```

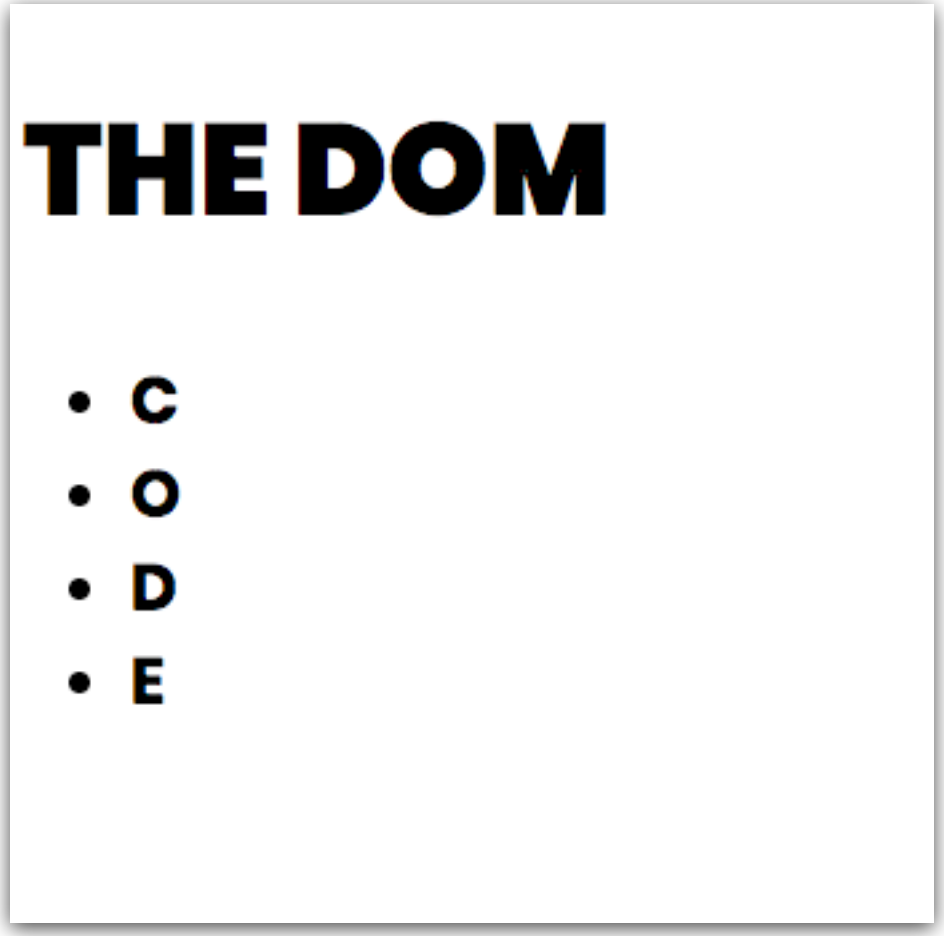
access all, one at a time

Activity(1): looking at html

In your html, you have the heading “THE DOM”, and four items on the list.



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="../main.css">
    <title>Selecting elements of particular type</title>
</head>
<body>
    <h1 id="myHeading">THE DOM</h1>
    <ul>
        <li class="not-orange">C</li>
        <li>O</li>
        <li class="not-orange">D</li>
        <li>E</li>
    </ul>

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

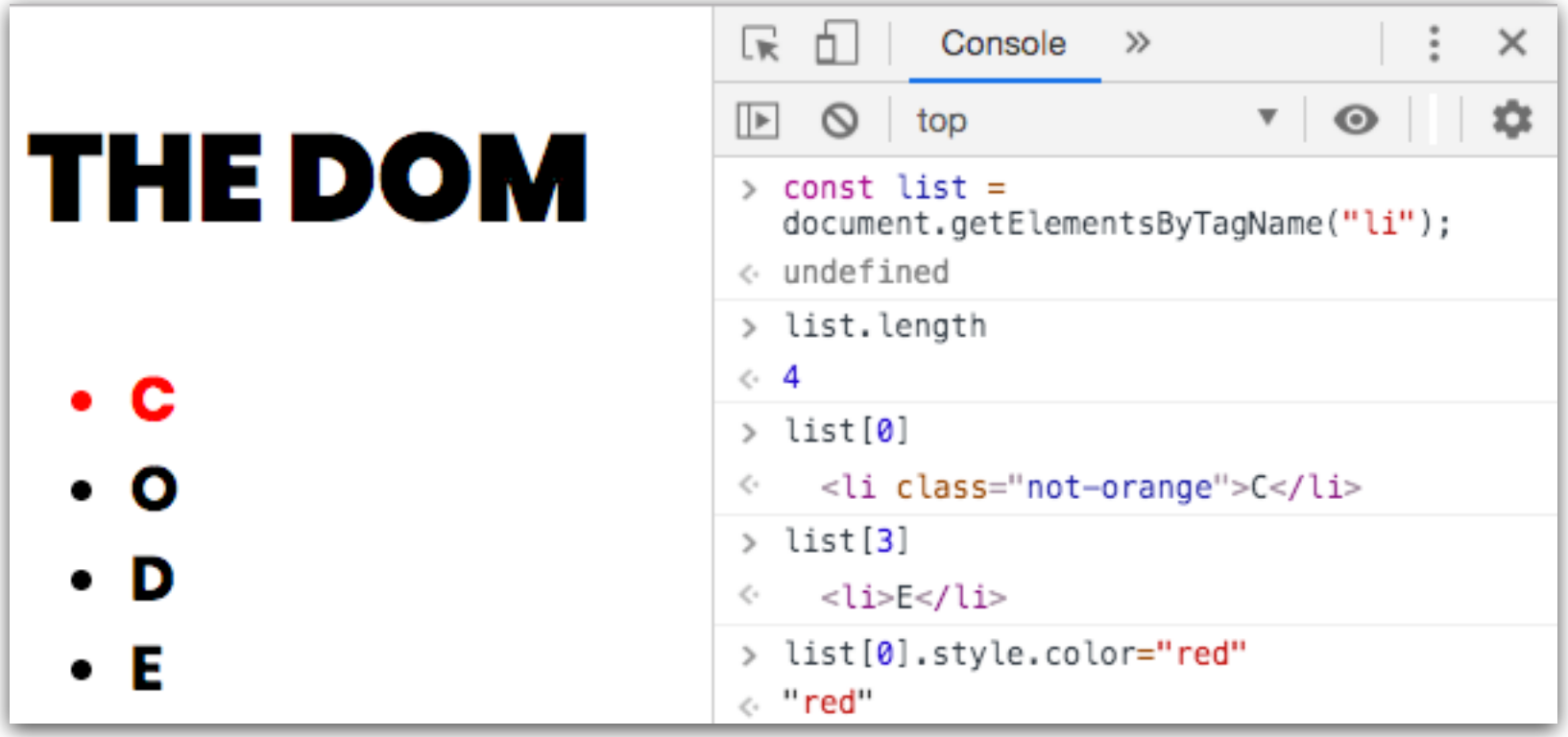


Activity(1): Using the console

Try each of the below in your console:

```
const list = document.getElementsByTagName("li");

list.length;
list[0];
list[3];
list[0].style.color="red";
```

Activity(1): Using the console



Activity(2): using app.js

Target the list elements and store in a constant, add to your app.js

You may want to tell the console to log it so we can see if it's doing the job.

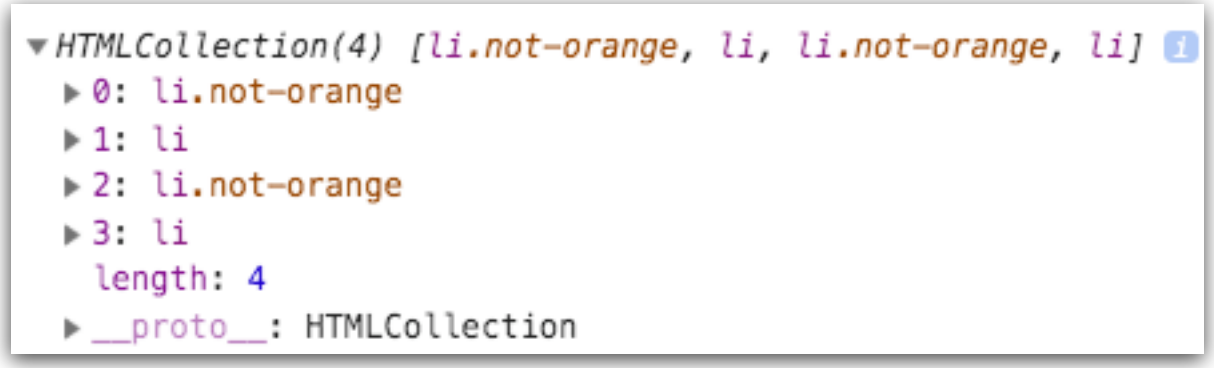
Solution(2): using app.js

Target the list elements and store in a constant, add to your app.js

```
const list = document.getElementsByTagName("li");
```

You may want to tell the console to log it so we can see if it's doing the job.

```
console.log(list);
console.log(list.length); //4
```



Activity(3): using app.js

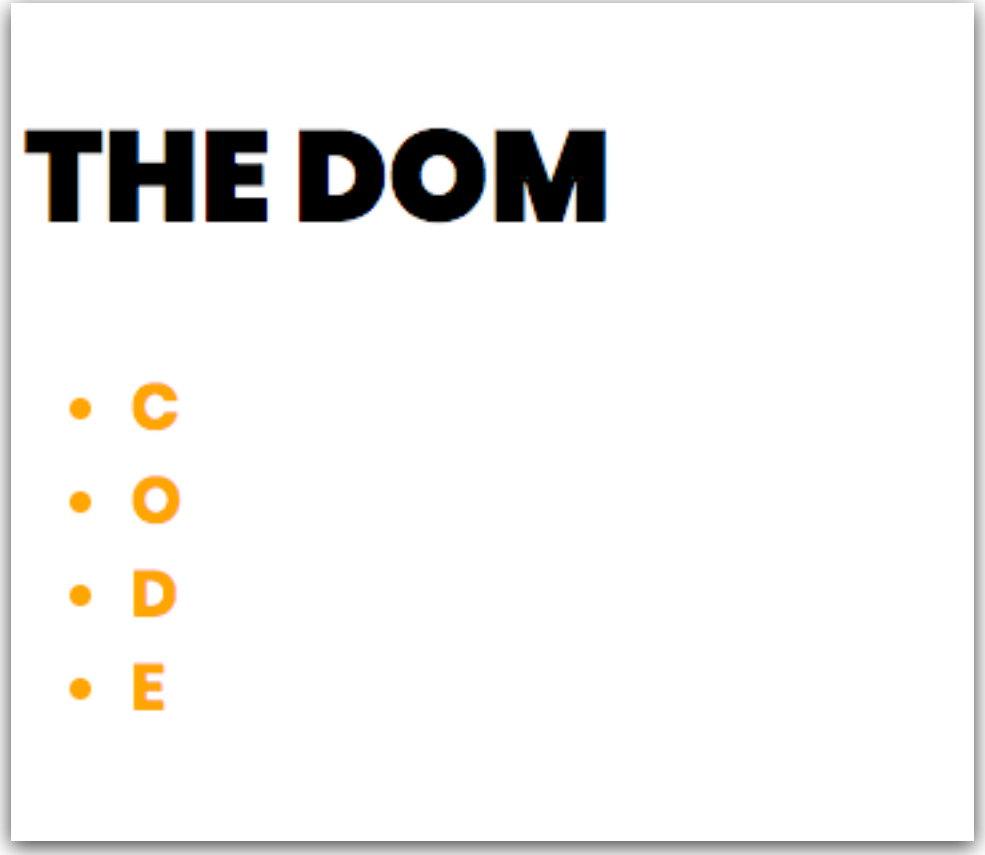
Use a for loop, change the text colour of **each** element in the list to orange. We should have already set the variable as below:

```
const list = document.getElementsByTagName("li");
```

Solution(3): using app.js

Use a for loop, change the text colour of each element in the list to orange.

```
for (let i = 0; i< list.length; i++){
    list[i].style.color = "orange";
}
```



Activity(4): using app.js

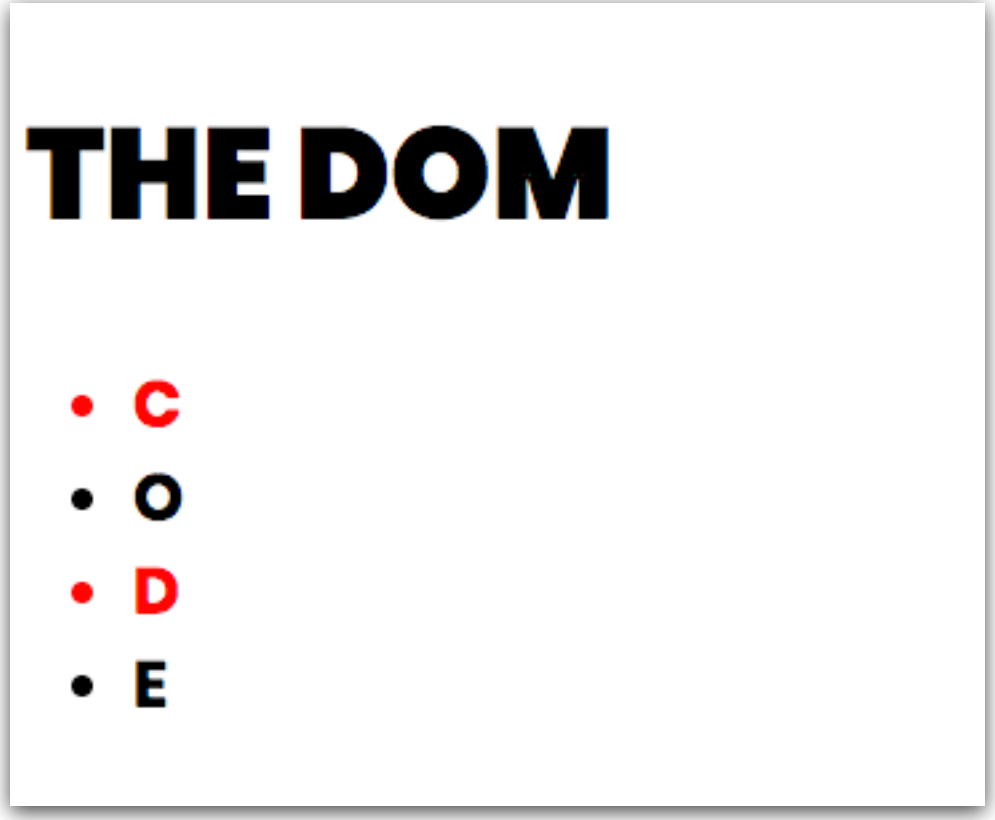
By using:

```
document.getElementsByClassName("not-orange");
```

Repeat the same steps to change these to “red”

Solution(4): using app.js

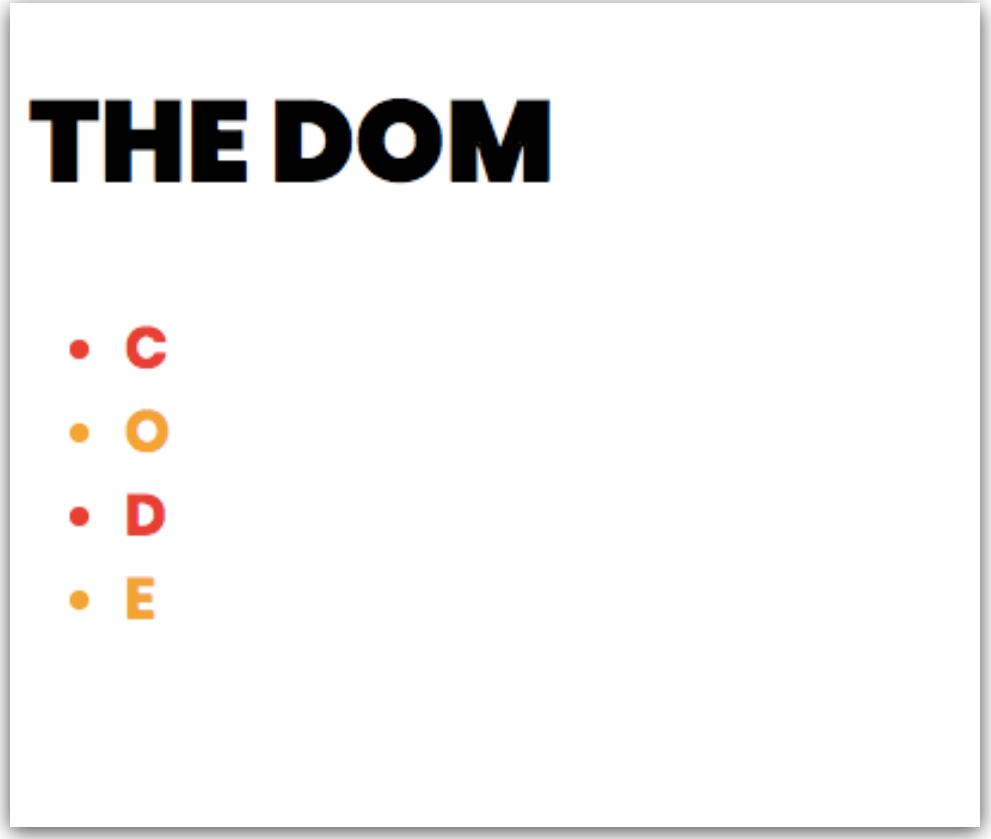
```
const list = document.getElementsByTagName("li");
const notOrange = document.getElementsByClassName("not-orange");

for (let i = 0; i< notOrange.length; i++){
    notOrange[i].style.color = "red";
}
```



Food for thought: ordering(1)

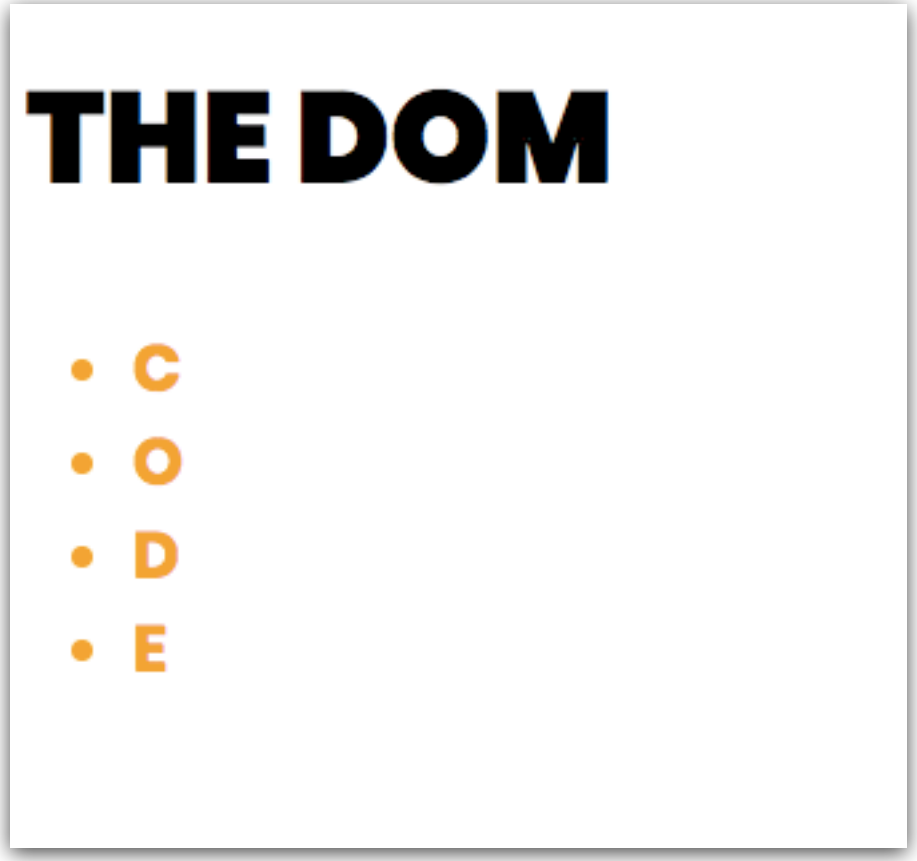
```
const list = document.getElementsByTagName(“li");
const notOrange = document.getElementsByClassName("not-orange");

for (let i = 0; i< list.length; i++){
    list[i].style.color = "orange";
}

for (let i = 0; i< notOrange.length; i++){
    notOrange[i].style.color = "red";
}
```

*This will change all li to orange, and then quickly change the “C” and “D” to red. It’s too quick you can’t see it happening.



Food for thought: ordering(2)

```
const list = document.getElementsByTagName(“li");
const notOrange = document.getElementsByClassName("not-orange");

for (let i = 0; i< notOrange.length; i++){
    notOrange[i].style.color = "red";
}

for (let i = 0; i< list.length; i++){
    list[i].style.color = "orange";
}
```



*This will change the “C” and “D” to red, and then change every li to orange, it’s too quick you can’t see it happening.

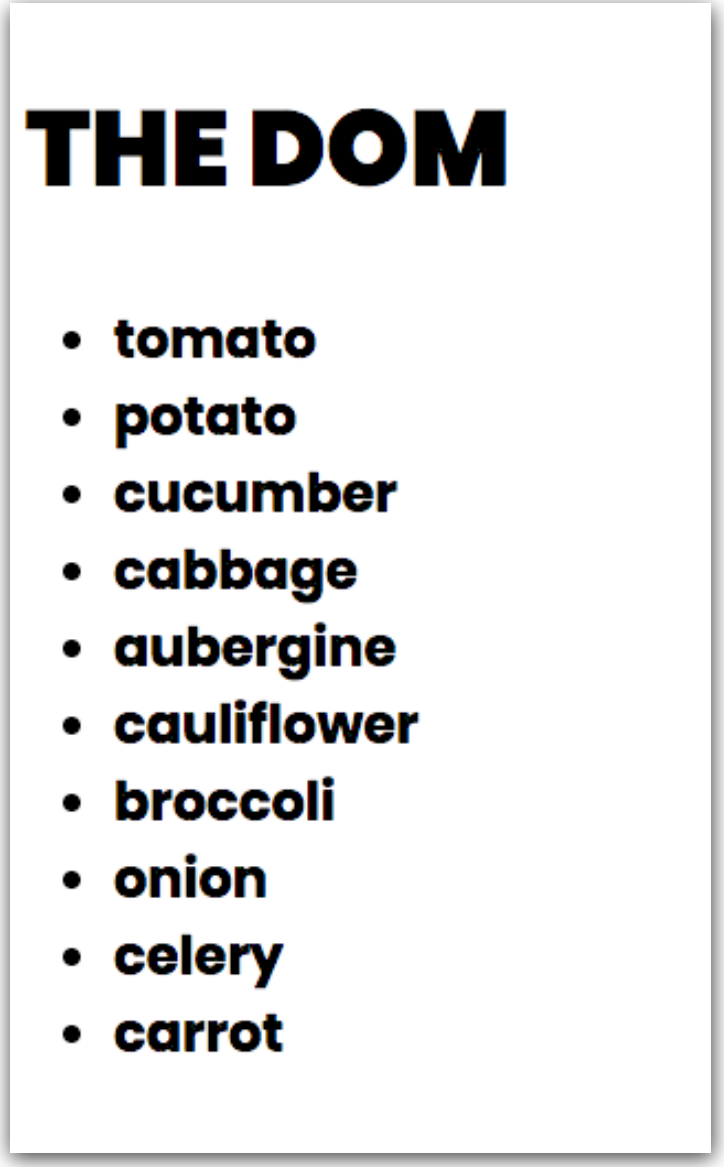
4 - querySelector & querySelectAll

Folder 4

4 - querySelector & querySelectAll

querySelector is super flexible, and versatile - accepts id, tag name, class...

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="../main.css">
    <title>querySelector and querySelectorAll</title>
</head>
<body>
    <h1 id='heading'>THE DOM</h1>
    <ul class='list-parent'>
        <li>tomato</li>
        <li>potato</li>
        <li>cucumber</li>
        <li class='green'>cabbage</li>
        <li class='green'>aubergine</li>
        <li>cauliflower</li>
        <li class='green'>broccoli</li>
        <li>onion</li>
        <li class='green'>celery</li>
        <li>carrot</li>
    </ul>
   <script src="app.js"></script>
</body>
</html>
```



Activity(1): querySelector/SelectorAll

Open index.html in Chrome and try each of these in the console. Remember to set these to a variable name.

```
document.querySelectorAll("li");
document.querySelector("li");

document.querySelector("#heading");

document.querySelector(".list-parent");
document.querySelectorAll(".green");
```

Let's take this in:

```
const myElement = document.getElementById("myId");
const myElement = document.querySelector("#myId");
```

Let's take this in:

```
const myElement = document.getElementById("myId");
const myElement = document.querySelector("#myId");
```

These code are functionally identical.

Activity(2): querySelectorAll

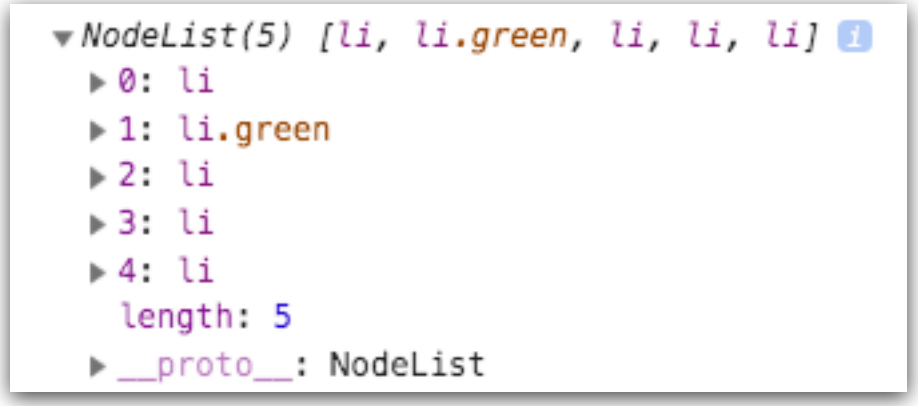
Add the code below to app.js, think about what each line does.

```
const listItems = document.querySelectorAll("li:nth-child(even)");

console.log(listItems);
console.log(listItems.length);
```

Activity(2): querySelectorAll

Add the code below to app.js, think about what each line does.

```
const listItems = document.querySelectorAll("li:nth-child(even)");

console.log(listItems);
console.log(listItems.length); //5
```



Activity(3): querySelectorAll

Use a for loop to display all even items to have text colour “lightgreen”.

Solution(3): querySelectorAll

Use a for loop to display all even items to have text colour “lightgreen”.

```
const listItems = document.querySelectorAll("li:nth-child(even)");

for(let i=0; i < listItems.length; i++){
    listItems[i].style.color = "lightgreen";
}
```

Challenge

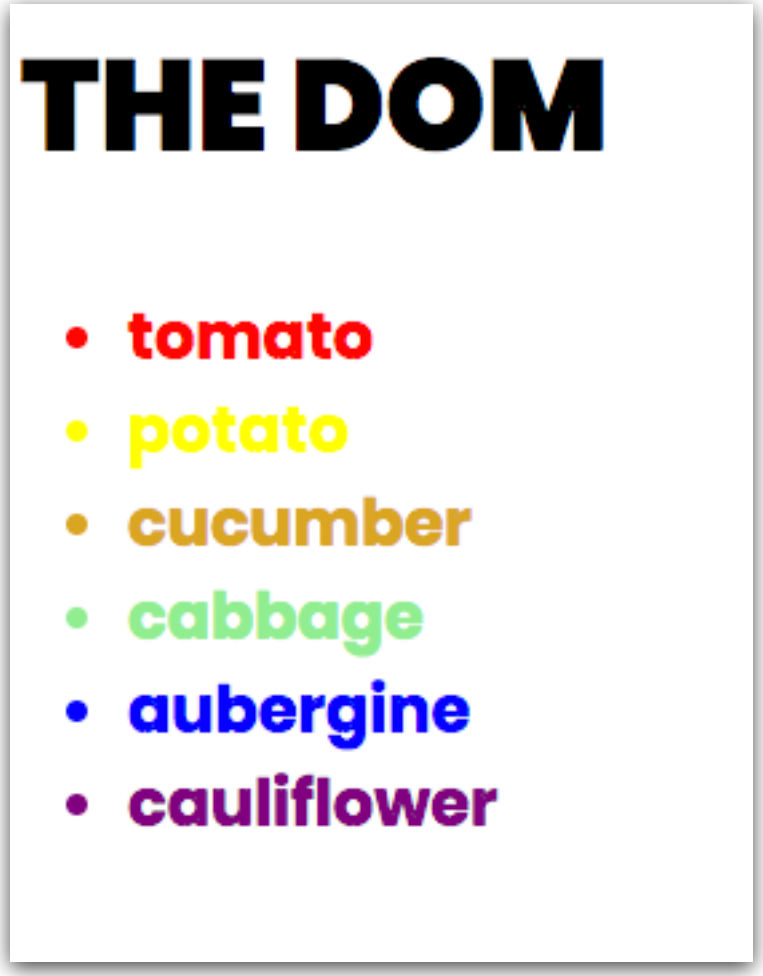
- Given: challenge.html and challenge.js

- Cycle over the list items and apply colours from the array called colours.

```
const colours = ['red', 'yellow', 'goldenrod', 'lightgreen', 'blue', 'purple'];
```

Solution: Challenge

```
const colours = ['red', 'yellow', 'goldenrod', 'lightgreen', 'blue', 'purple'];
const listItems = document.querySelectorAll("li");

for(let  i = 0; i < listItems.length; i++) {
    listItems[i].style.color  =  colours[i];
}
```



5 - textContent and innerHTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="../main.css">
    <title>textContent & innerHTML</title>
</head>
<body>
    <h1 id="placeholder">I like to eat</h1>
    <input id="input" type="text">
    <button id="submit">submit</button>

    <ul id="list">
        <li>biscuits</li>
        <li>sweets</li>
        <li>sausage</li>
    </ul>

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

Activity(1): Storing elements in const

Add these to app.js, can you find where these are in html?

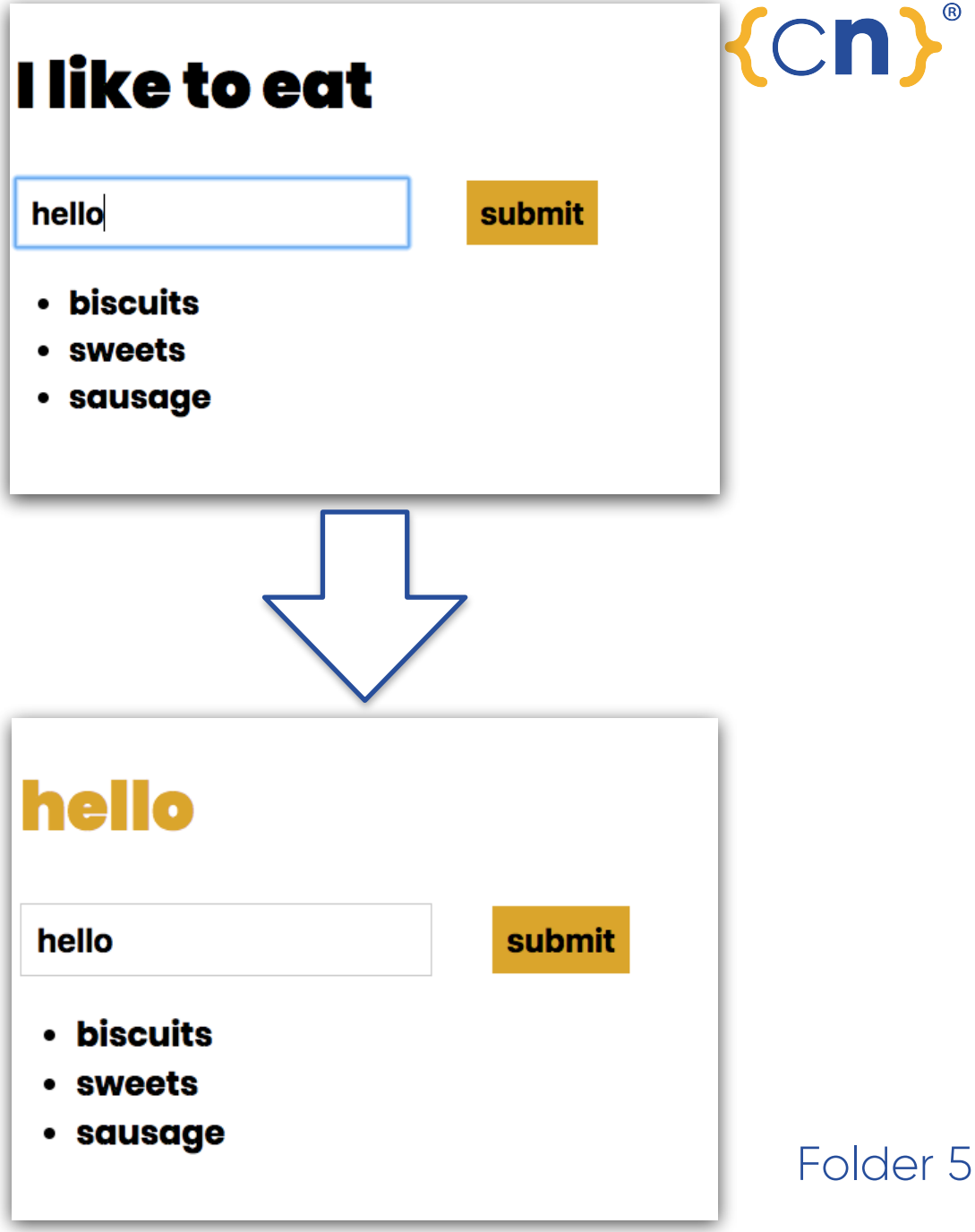
```
const placeholder = document.getElementById("placeholder");
const input = document.getElementById("input");
const submit = document.getElementById("submit");
const list = document.getElementById("list");
```

Activity(2): apply textContent

We want the text input to be placed in the heading, we can attach “addEventListener” to the button.

```
submit.addEventListener("click", () =>{
    placeholder.style.color ="goldenrod";
    placeholder.textContent = input.value;
})
```

```
const placeholder = document.getElementById("placeholder");
const input = document.getElementById("input");
const submit = document.getElementById("submit");
const list = document.getElementById("list");

submit.addEventListener("click", () =>{
    placeholder.style.color ="goldenrod";
    placeholder.textContent = input.value;
})
```



Activity(3): use of ``

Comment out previous and do this instead, notice the difference.

```
submit.addEventListener("click", () =>{
    placeholder.style.color ="goldenrod";
    placeholder.textContent = `<li>${input.value}</li>`;
})
```

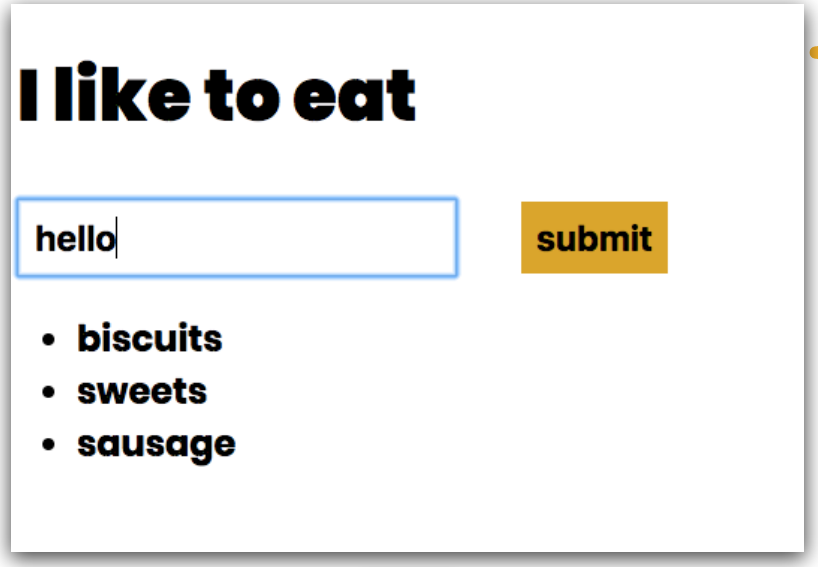
Activity(4): use of innerHTML

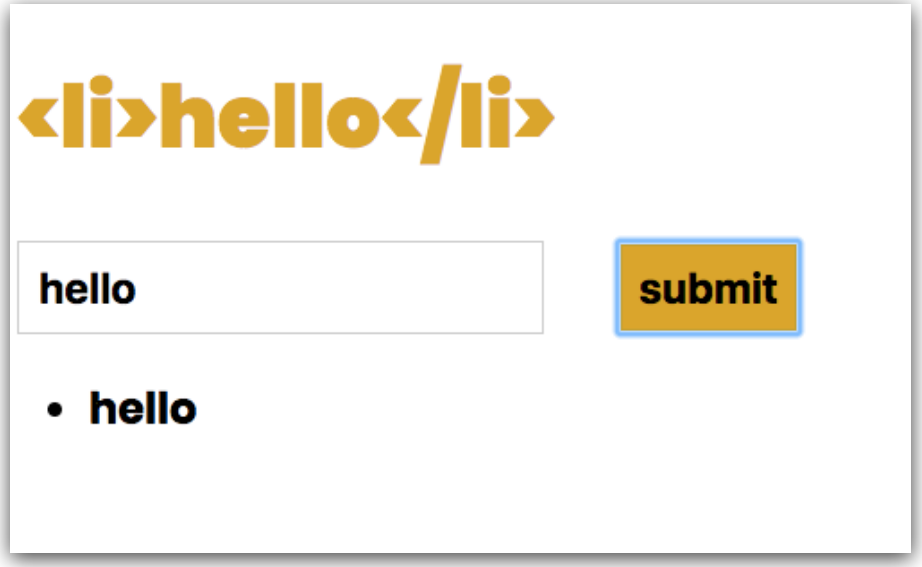
Add innerHTML into your app.js

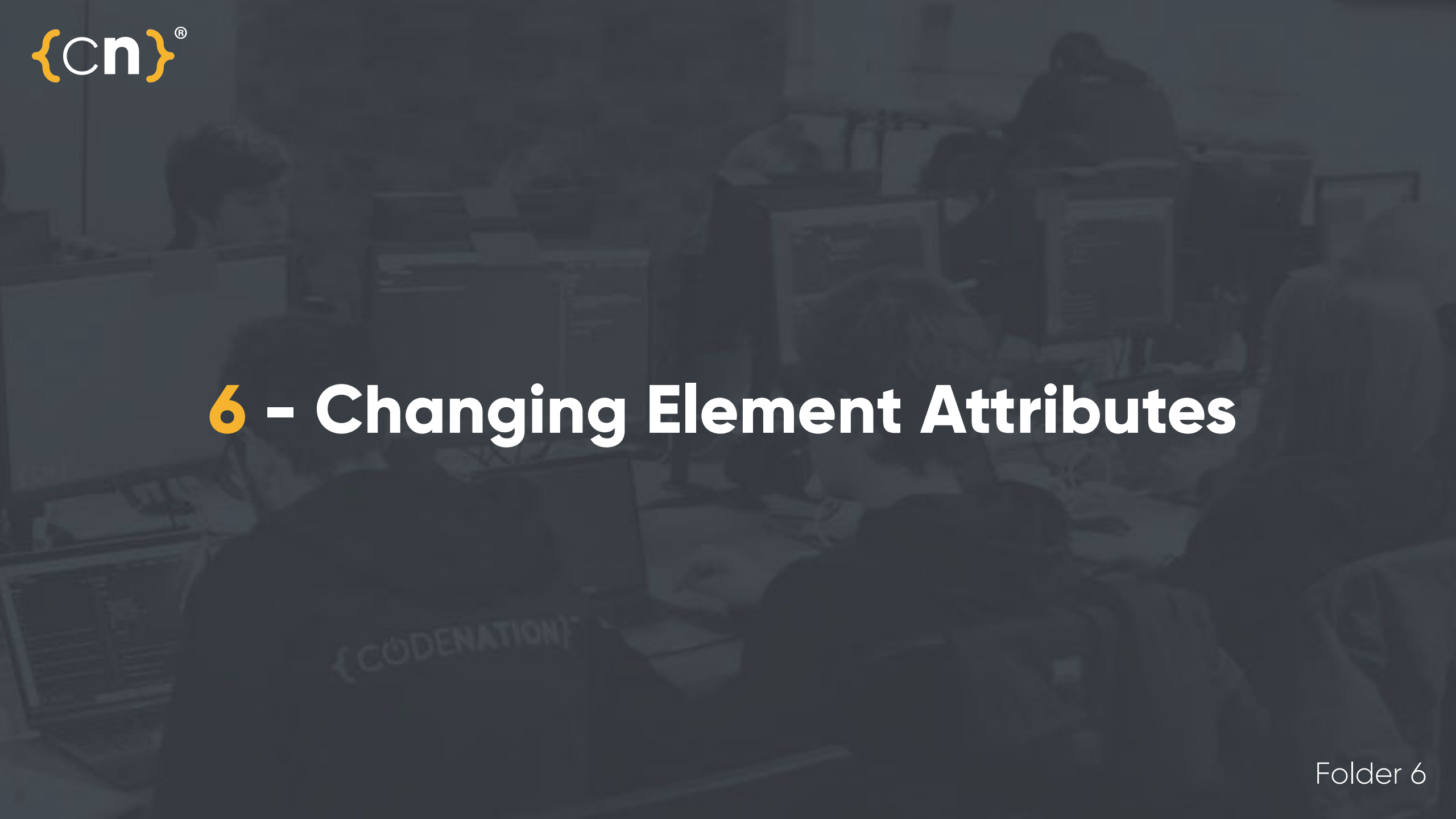
```
submit.addEventListener("click", () =>{
    placeholder.style.color ="goldenrod";
    placeholder.textContent = `<li>${input.value}</li>`;

    list.innerHTML = `<li>${input.value}</li>`;
})
```

```
const placeholder = document.getElementById("placeholder");
const input = document.getElementById("input");
const submit = document.getElementById("submit");
const list = document.getElementById("list");

submit.addEventListener('click', () => {
    placeholder.style.color = 'goldenrod';
    placeholder.textContent = `<li>${input.value}</li>`;

    list.innerHTML = `<li>${input.value}</li>`
})
```

This replaced **all** list content in html



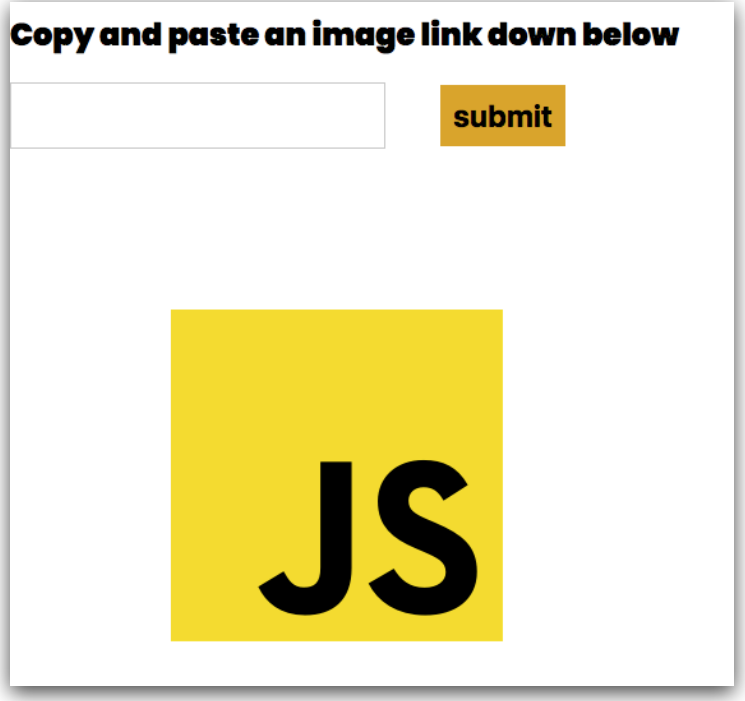




6 - Changing Element Attributes

Folder 6

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="../main.css">
    <title>Changing Element Attributes</title>
</head>
<body>
    <h2>Copy and paste an image link down below</h2>
    <input id="input" type="text">
    <button id="submit">submit</button>
    <img id="image" src="https://pluralsight.imgix.net/paths/path-icons/
javascript-36f5949a45.png">

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



Activity(1): store image element

Create a variable called “image”, assign this to the ID image.

console.log the variable “image”.

Solution(1): store image element

Create a variable called “image”, assign this to the ID image.

console.log the variable “image”.

```
const image = document.getElementById("image");
console.log(image);
```

```
<img id="image" src="https://pluralsight.imgix.net/paths/path-icons/javascript-36f5949a45.png">
```

Activity(2): using the console

Before continuing adding more code to app.js, we want to test a few things using the console in the browser first.

Open index.html, open console and type:

```
image.src = "ADD LINK TO IMAGE HERE"
```

In my case:

```
image.src = "https://cdn.wearecodenation.com/app/uploads/20190220150002/new-class-32.jpg"
```

in app.js

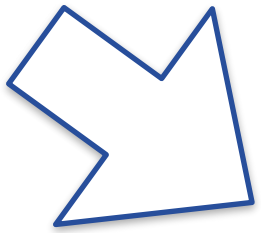
```
const image = document.getElementById("image");
```

in browser

```
image.src = "https://cdn.wearecodenation.com/app/uploads/20190220150002/new-class-32.jpg"
```

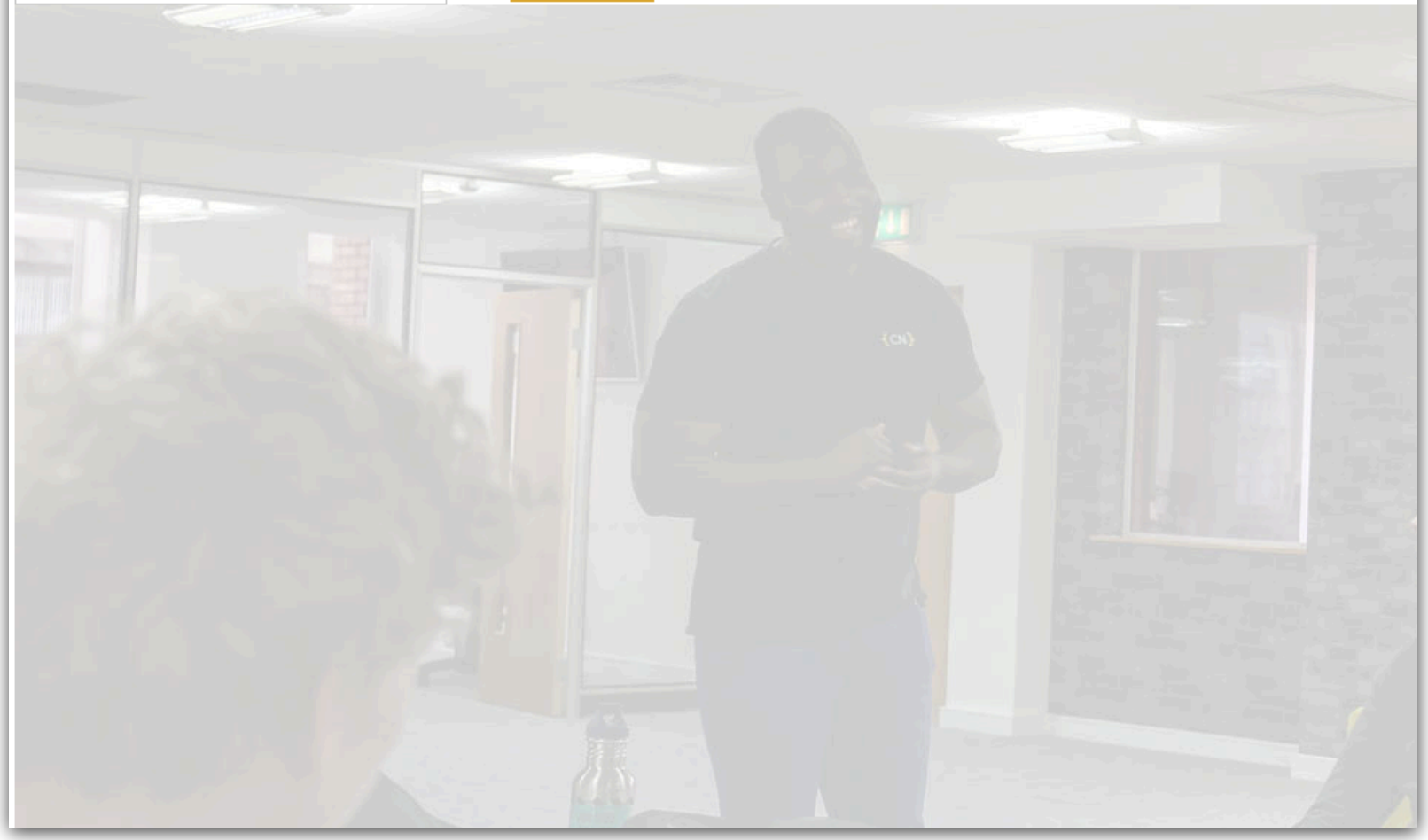
Copy and paste an image link down below

submit



Copy and paste an image link down below

submit



Activity(3): applying in app.js

In app.js, check that you have created all variables you need, as below:

```
const input = document.getElementById("input");
const button = document.getElementById("submit");
```

Add a function that changes the source of the image when the “submit” button is clicked.

Solution(3): applying in app.js

In app.js, check that you have created all variables you need, as below:

```
const input = document.getElementById("input");
const button = document.getElementById("submit");
```

Add a function that changes the source of the image when the “submit” button is clicked.

```
button.addEventListener("click", ()=>{
    image.src = input.value;
})
```

Copy and paste an image link down below

0002/new-class-32.jpg

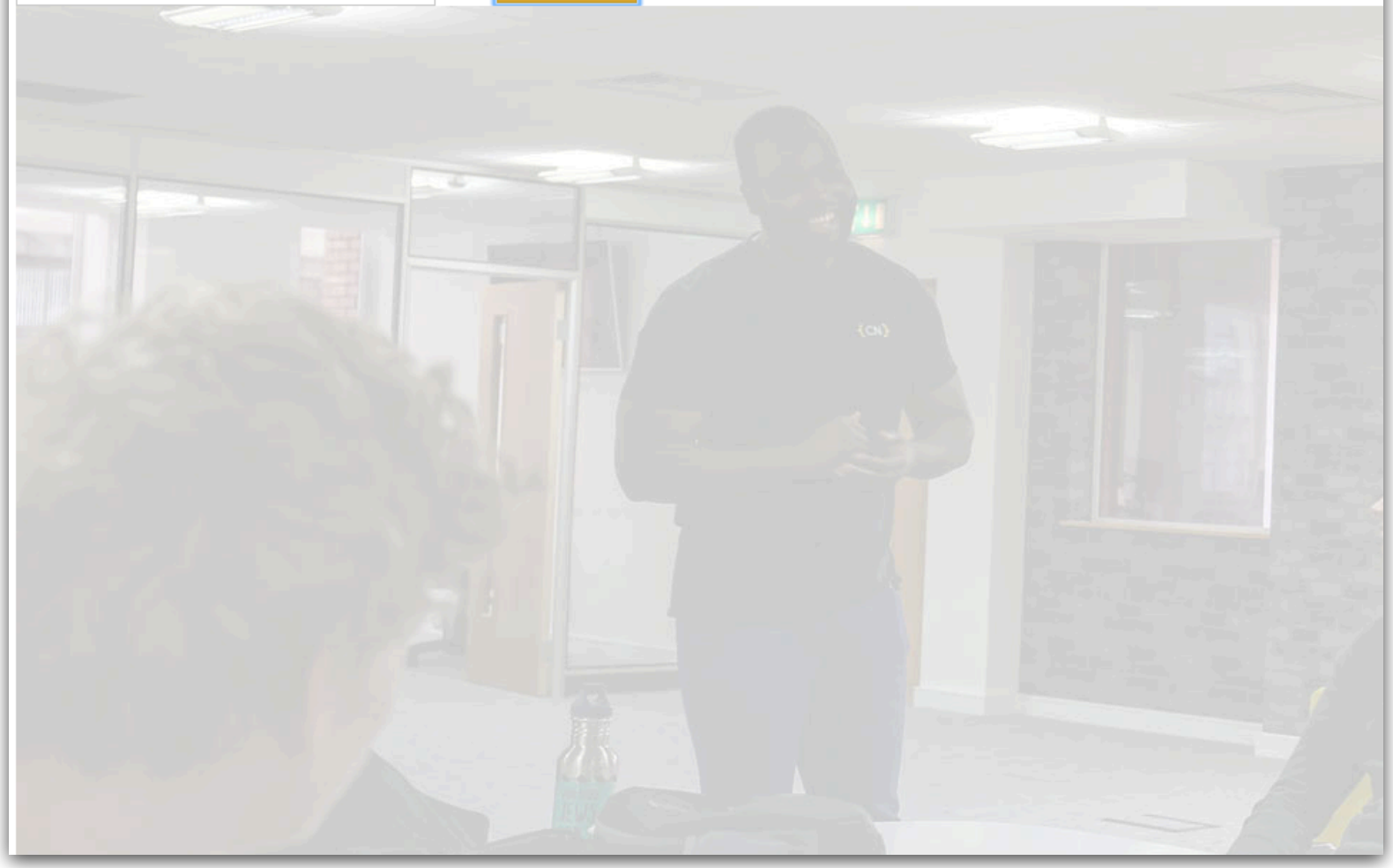
submit

JS

Copy and paste an image link down below

https://cdn.wearecoden

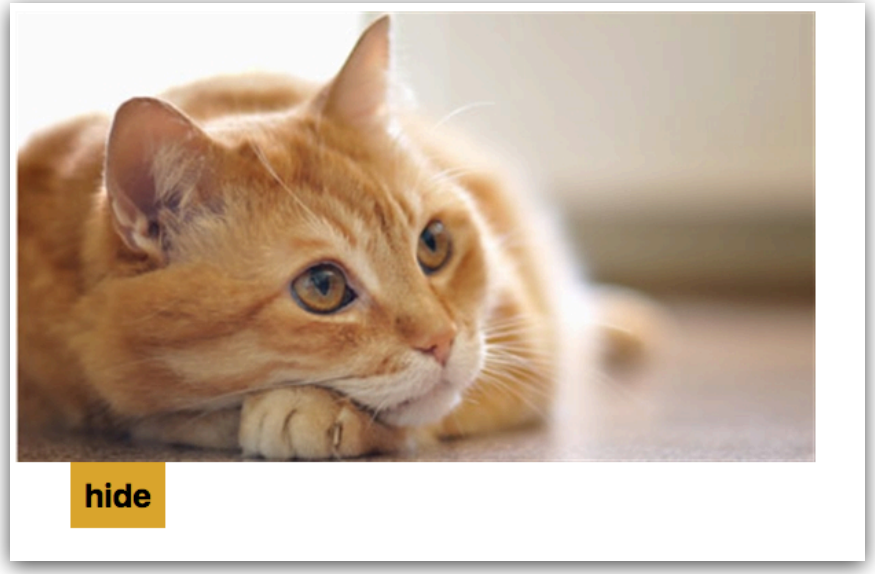
submit



7 - Styling Elements

Folder 7

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="../main.css">
    <title>Styling Elements with JS</title>
</head>
<body>
    <img id="cat" src="https://www.cats.org.uk/uploads/images/
featurebox_sidebar_kids/grief-and-loss.jpg">
    <button id="submit">hide</button>

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



Activity(1): viewing the image style

First up we want to check out the image style by giving the “img” a const.

```
const image = document.getElementById("cat");
console.log(image.style);
```

Or in the console, type:

```
image.style
```

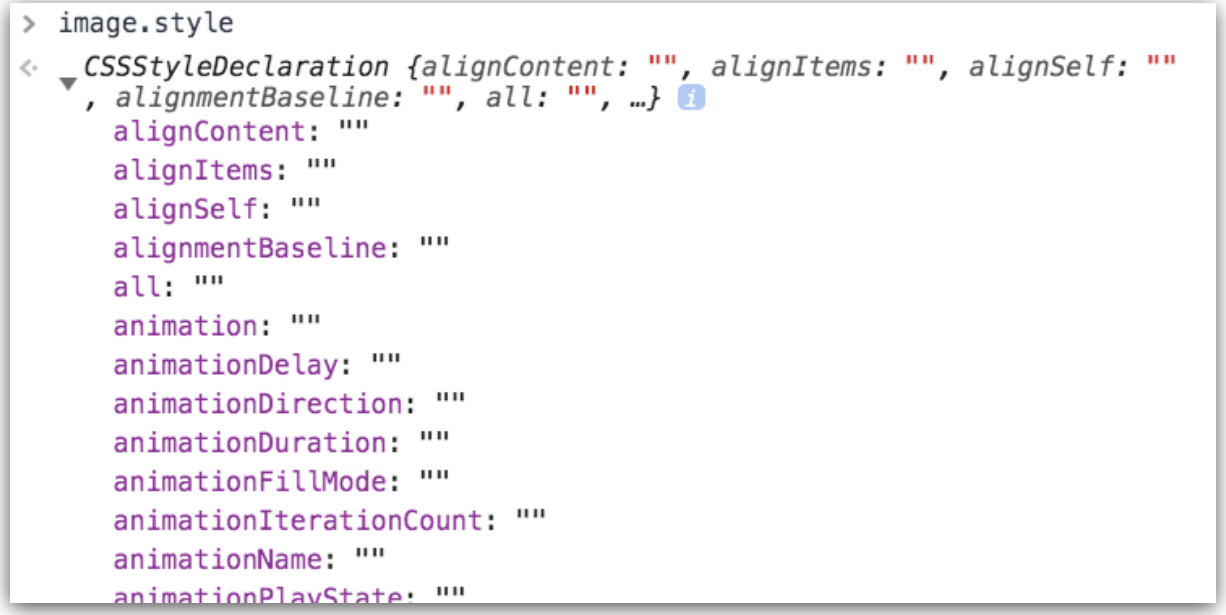
Activity(1): viewing the image style

First up we want to check out the image style by giving the “img” a const.

```
const image = document.getElementById("cat");
console.log(image.style);
```

Or in the console, type:

```
image.style
```



Activity(2): show and hide image

We want to hide the image when the button is clicked, and show when it is clicked again, so we need to set another const and then a function, add code inside the function:

```
const button= document.getElementById("submit");

button.addEventListener("click", () => {
 //your code here
})
```

Solution(2): show and hide image

We want to hide the image when the button is clicked, and show when it is clicked again, so we need to set another const and then a function, add code inside the function:

```
const button= document.getElementById("submit");

button.addEventListener("click", () => {
    if(image.style.display == "none") {
        image.style.display = "block";
        button.textContent = "hide";
    } else {
        image.style.display = "none";
        button.textContent = "show";
    }
})
```

***The style is added to the html directly (inspect...), remember inline styling is the priority?**

Learning Objectives

- Understand the html and DOM structure
- To be able to apply changes to the DOM by responding to user interaction