the Master Course

{CODENATION}

JS & DOM

Introduction

{CODENATION}

{CODENATION}

Learning Objectives

To explore the HTML & DOM structure

To be able to apply changes to the DOM by responding to user interaction

{CN}®

DOM

The DOM

... Document Object Model

What is the DOM

... a **representation of a webpage** that JS can use.
Changes **that JS makes** to the DOM **alter the webpage**.

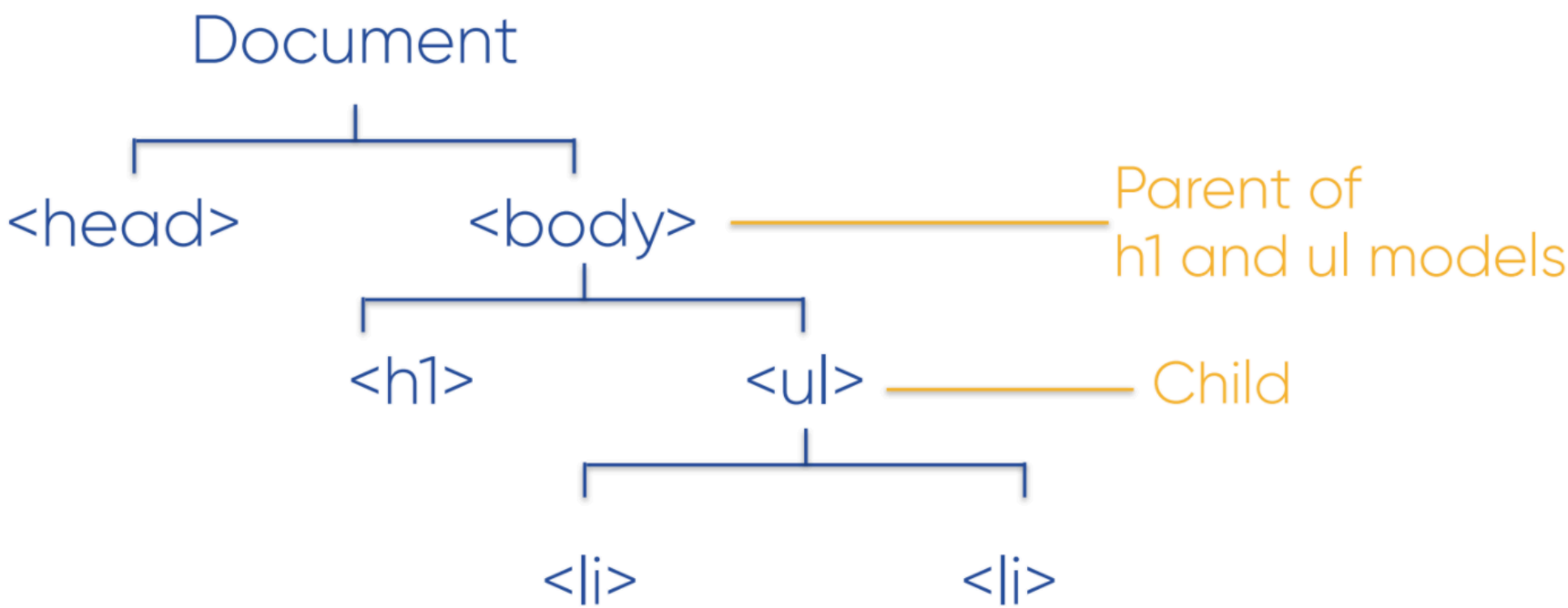
DOM represents a webpage as a tree like structure....

DOM

```
<!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>
```

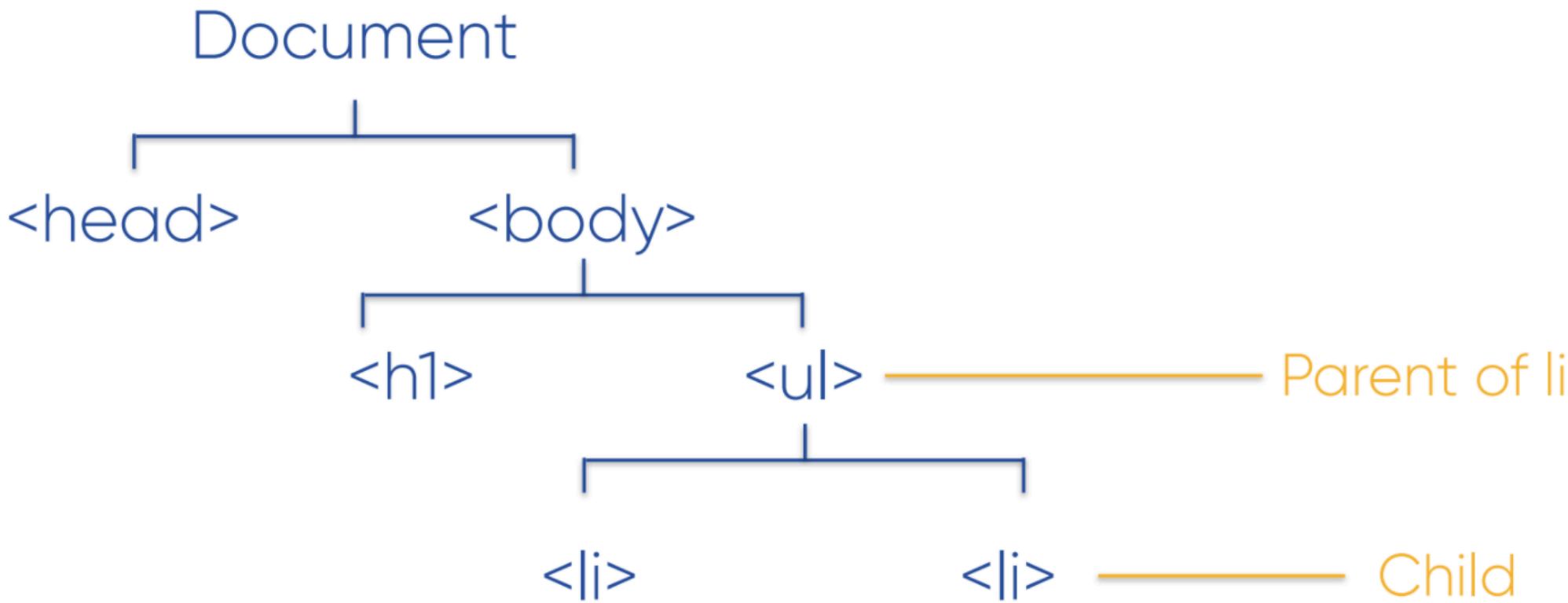
{CODENATION}

DOM



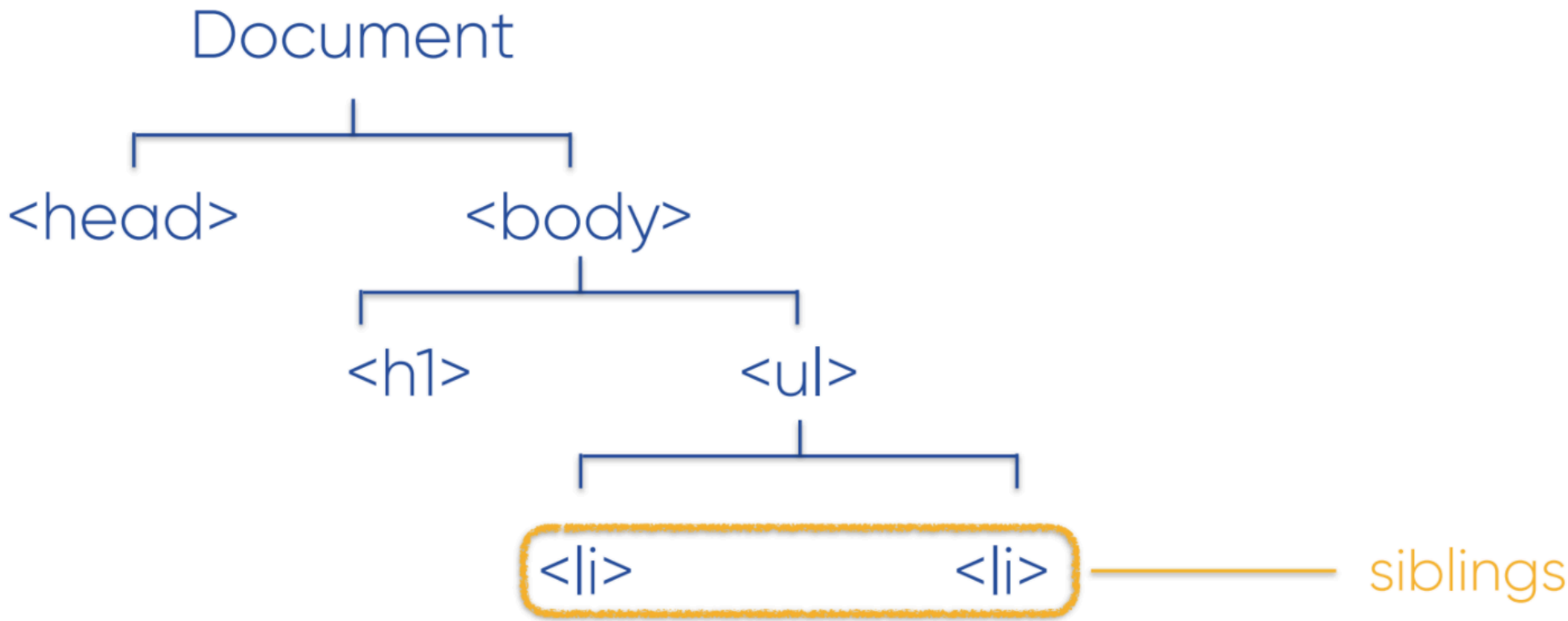
{CODENATION}

DOM



{CODENATION}

DOM



{CODENATION}

{CN}®

DOM

What can JS do with DOM?

... lots! **Selecting Elements**, **Reading/ Changing Elements**, **Responding to User Events**

Activity: Alert

Open Chrome, inspect and click on Console.
Type the below and 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).

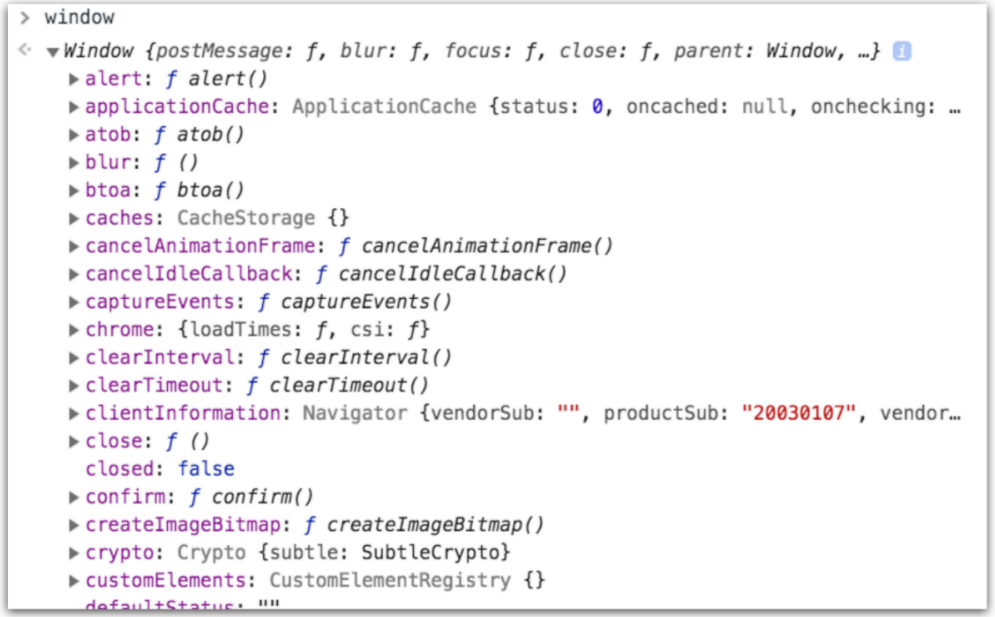
```
> location.href
<- "chrome-search://local-ntp/local-ntp.html"
```

```
> 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**".



Want to clear the console?

Ctrl + L

{CODENATION}

Activity: Alert Command

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

does the same thing as:

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

Want to clear the console?

Ctrl + L

{CN}®

DOM

Document Object

... document is a **global object** representing the **HTML and content of the page**

{CN}®

DOM

Document Object

... with JS you can **select and control elements** of the currently loaded webpage.

e.g Change the heading to a different color

{CN}®

DOM

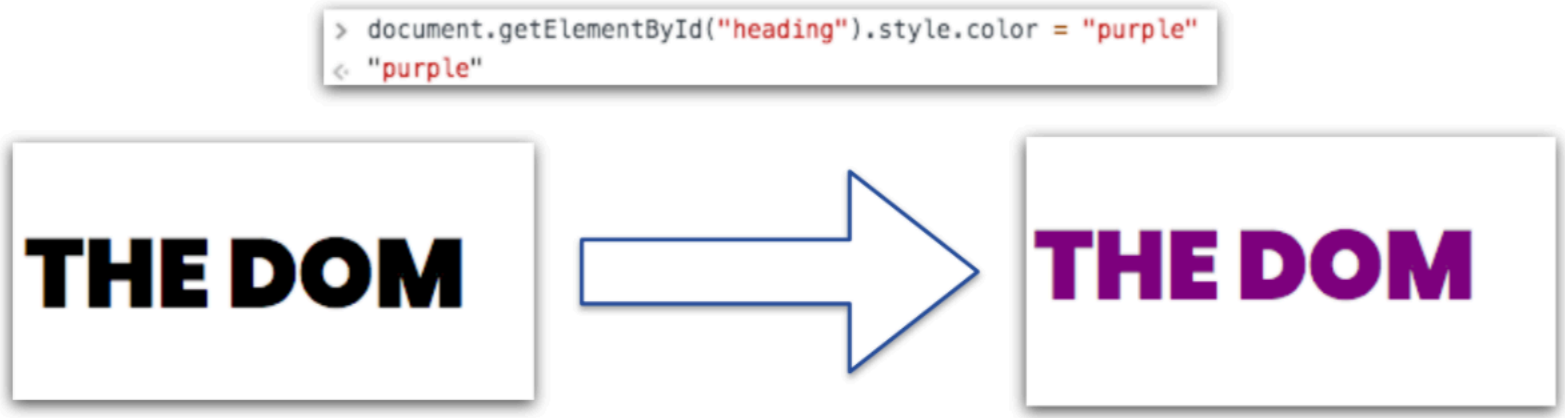
Lets look at...

getElementById

getElementById


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

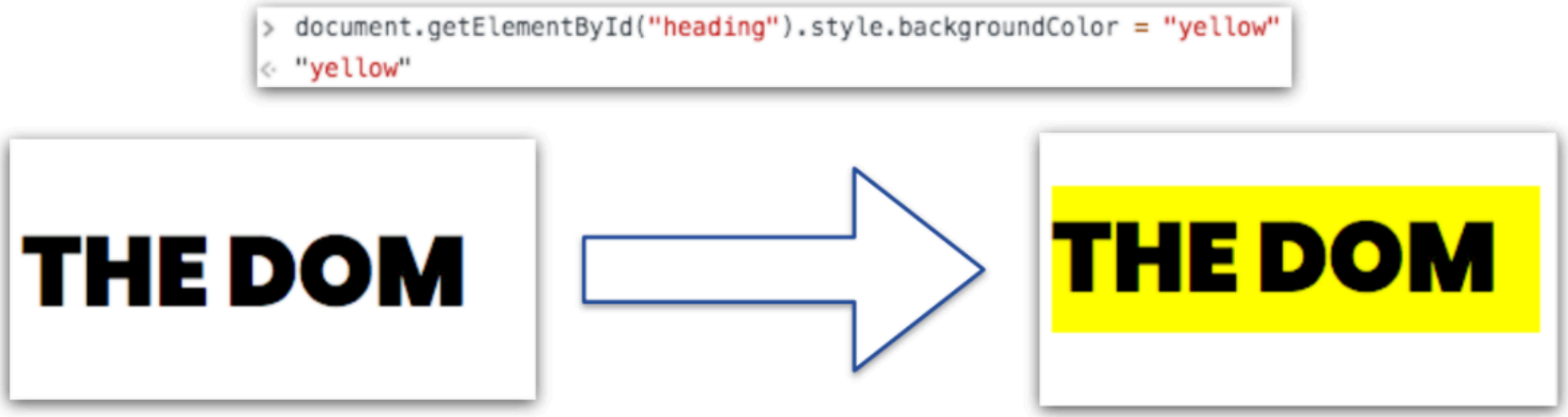
Type this into the console.
Change to different colors, then to black!



getElementById


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

Type this into the console.
Change to different colors, then to white!



What's the point?

... it allows **interactivity with the HTML**. For example, clicking, hovering, scrolling, submitting

Activity: Creating an App.js

Inside your folder, there should be an **app.js** file. In the HTML add below the heading this script tag.

```
<!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>
```

Why at the bottom?

{CN}®

DOM

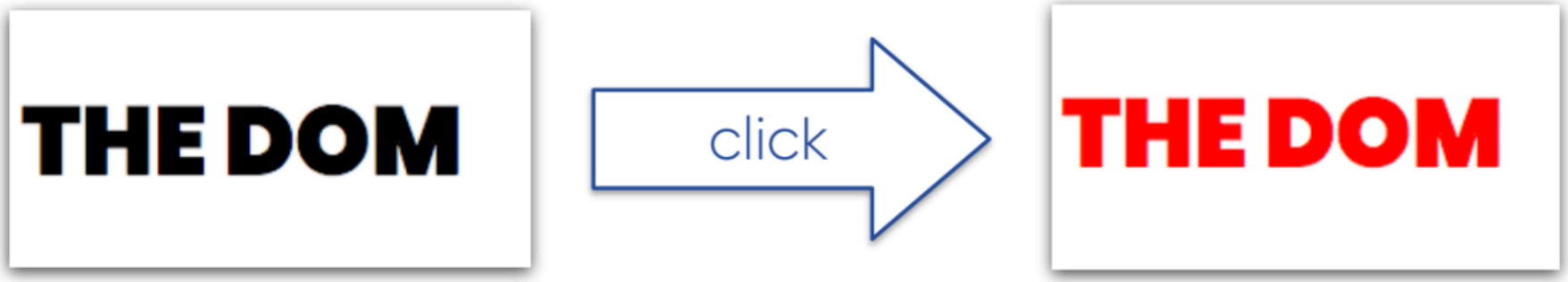
Lets look at...

Add Event Listener

Folder 1

Activity: Add inside your app.js


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

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

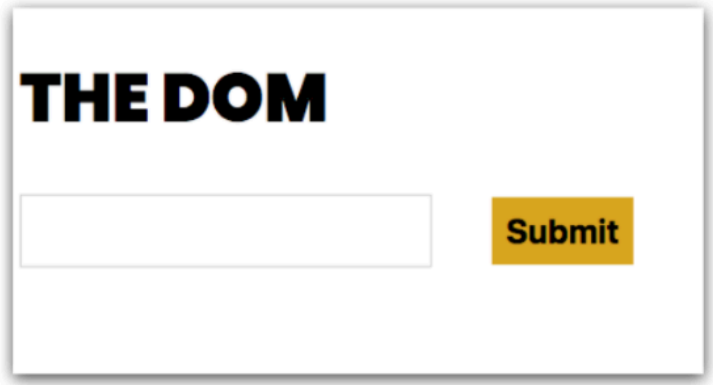


Lets look at...

Select Element by ID

DOM

```
<!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>
```



{CODENATION}

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**

DOM

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

Activity

Make the button **listen to mouse clicks** using the correct method.

Then, get the **value from the input value** and use that to **change the color of the heading.**

DOM

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


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

Lets look at...

Select Element by a particular type.

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");
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];
}
```

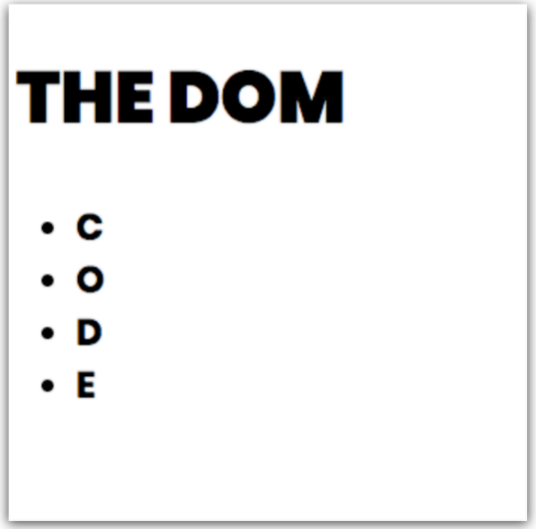
access all, one at a time

Activity:

DOM

In your HTML, 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>
```



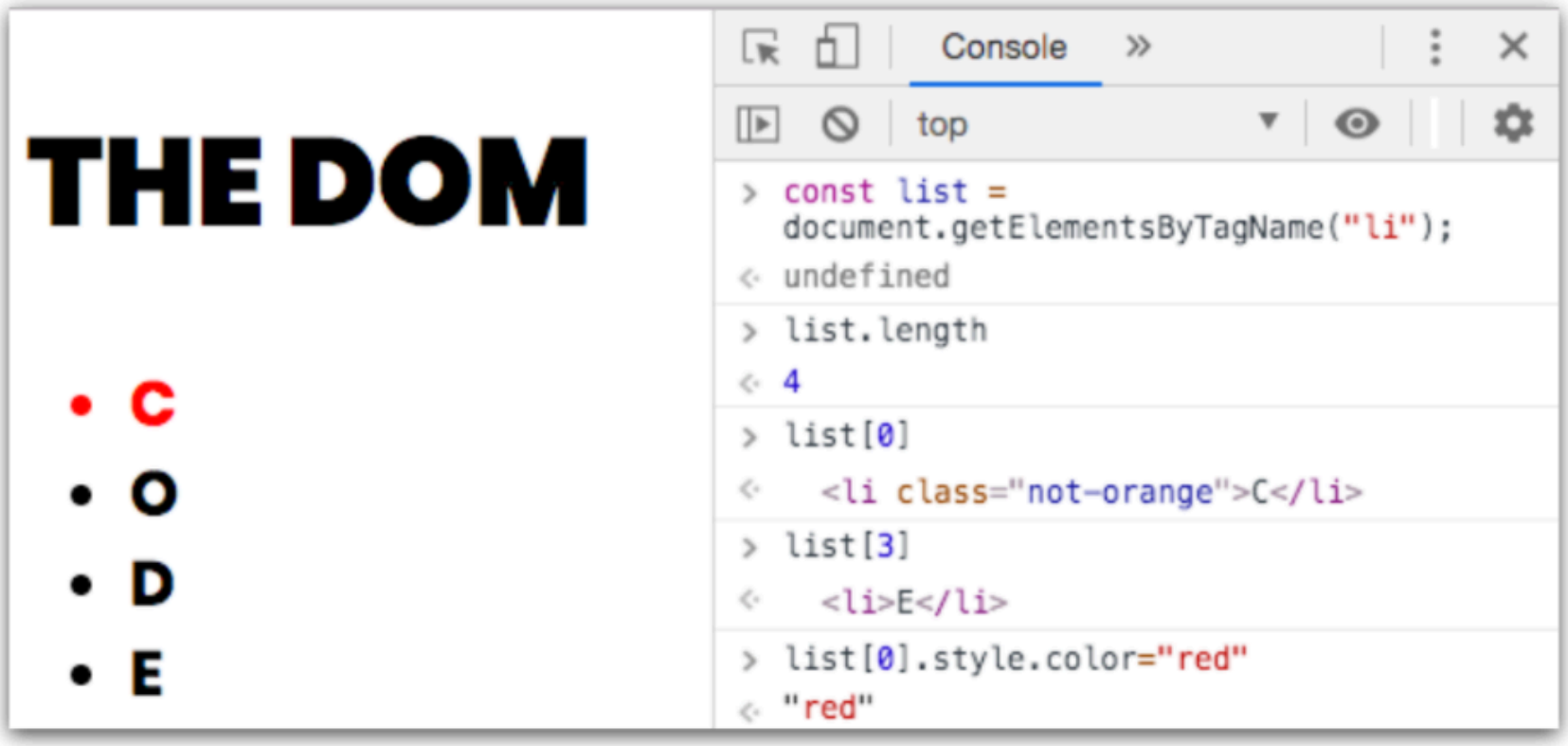
{CODENATION}

Activity: 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: 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 check it is doing it's job!

Solution(2): using 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

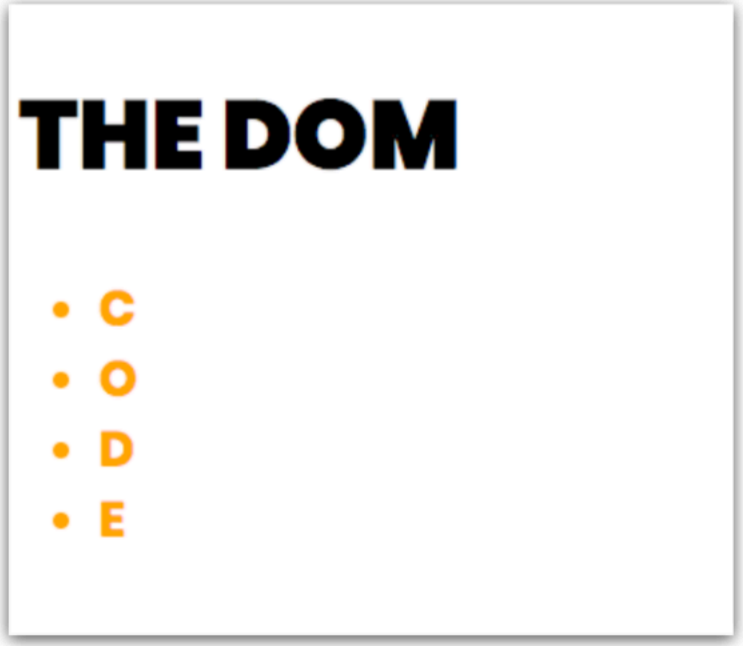
Use a for loop, change the text color of **each** element in the list to arrange. We should have already set the variable as below:

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

Solution(3): using app.js

Use a for loop, change the text color 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

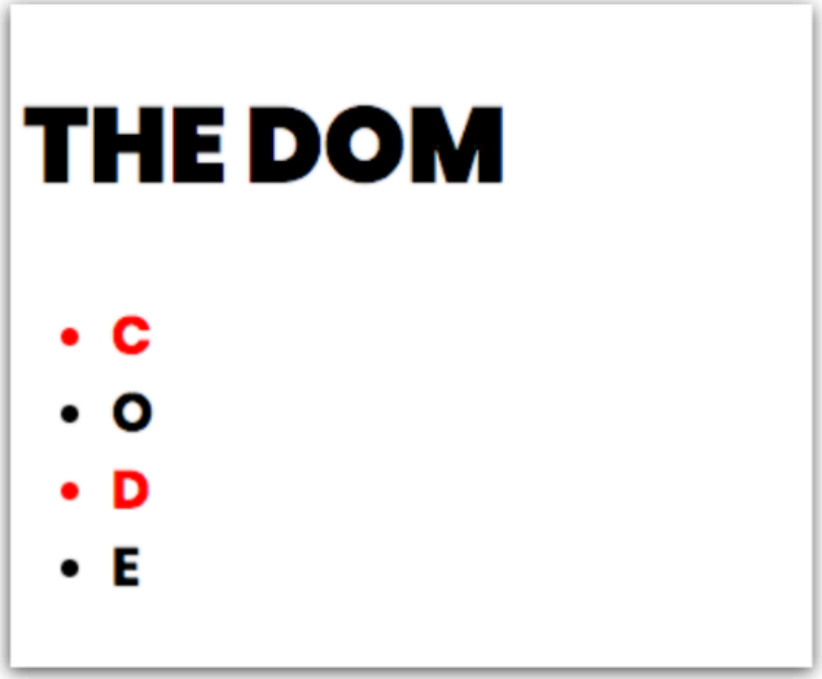
By using:

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

Repeat the same steps to change these to “red”

Solution(4): using app.js

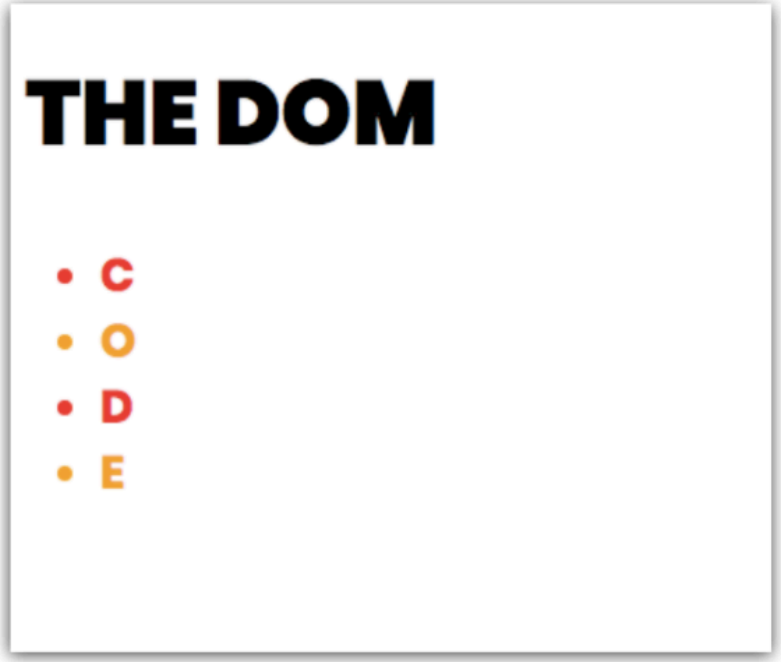
```
const list = document.getElementsByTagName("li");
const notOrange = document.getElementsByClassName("not-orange");

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";
}
```

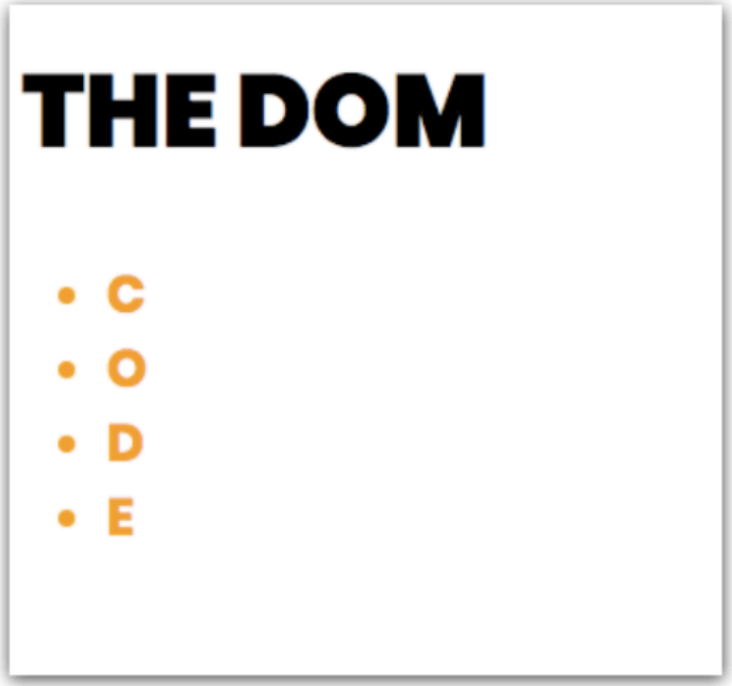
*This will change all li to orange, and then quickly change the “C” and “D” to red. It’s too quick you can’t see it happening.



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.



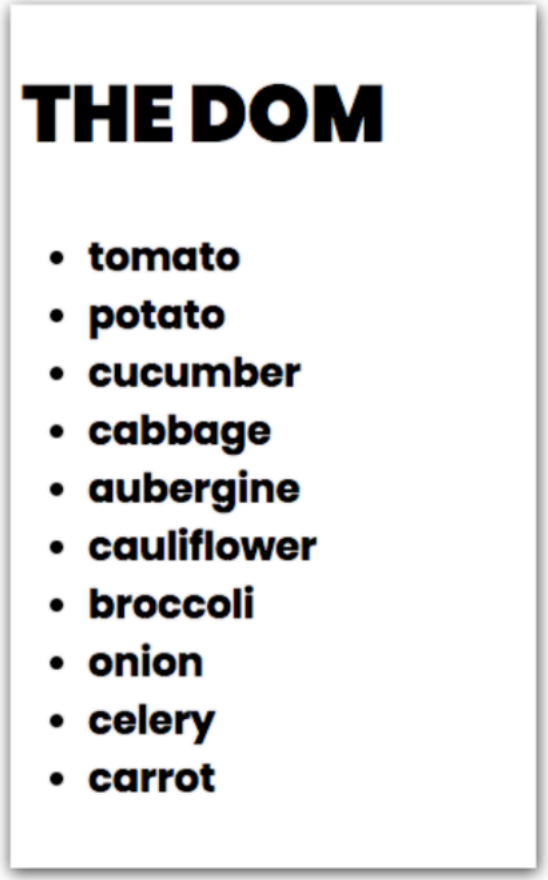
{CN}®

DOM

Lets look at...

querySelector & querySelectorAll

```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");
```

Lets take this in:

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

These code lines are **functionally identical.**

Activity(2): querySelectorAll

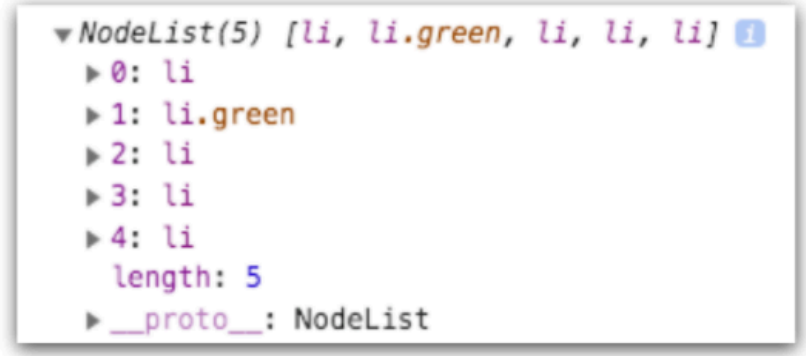
Add this code below to app.js, think about what each line does...

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

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

Activity(2): querySelectorAll

Add this 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 loop to display all even items to have text color “light green”.

Solution(3): querySelectorAll

Use a loop to display all even items to have text color “light green”.

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

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

Lets look at...

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?

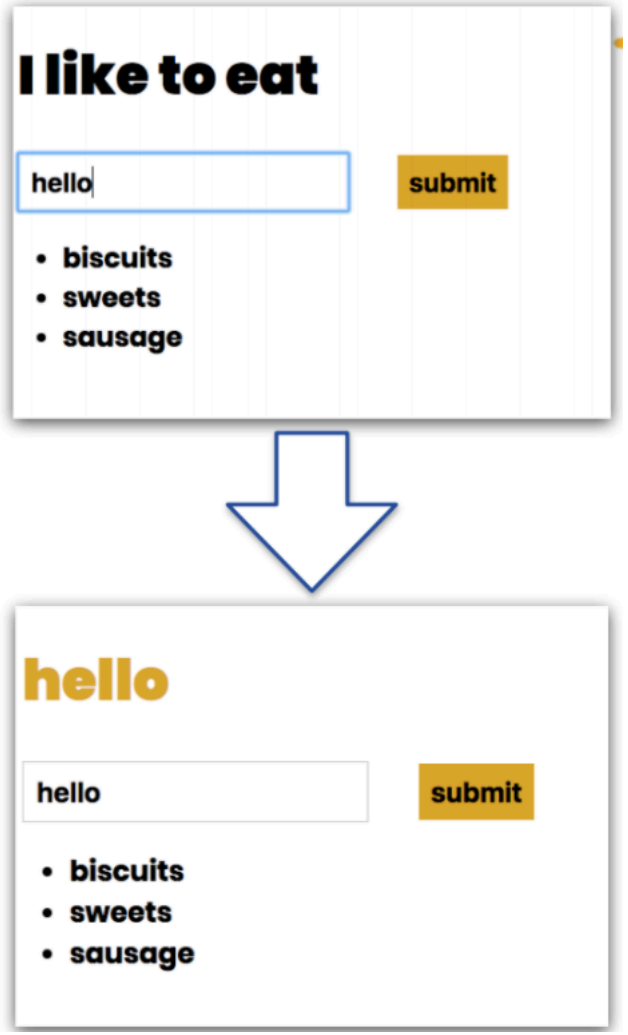
```js
const placeholder = document.getElementById("placeholder");
const input = document.getElementById("input");
const submit = document.getElementById("submit");
const list = document.getElementById("list");
```

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>`;
})
```

Activity (4): use of innerHTML

Add innerHTML into your app.js

```
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

{CODENATION}

{CN}®

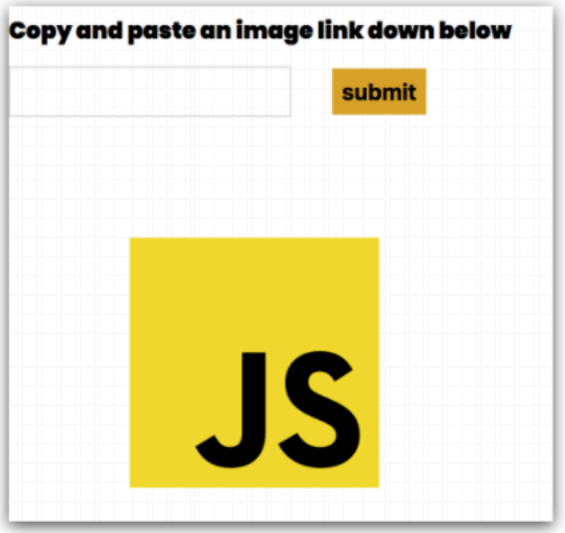
DOM

Lets look at...

... 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">
```

Folder 6

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"
```

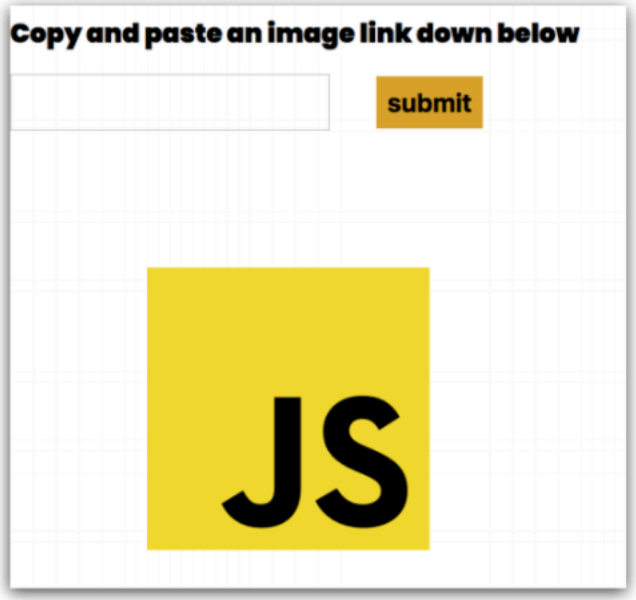
in app.js

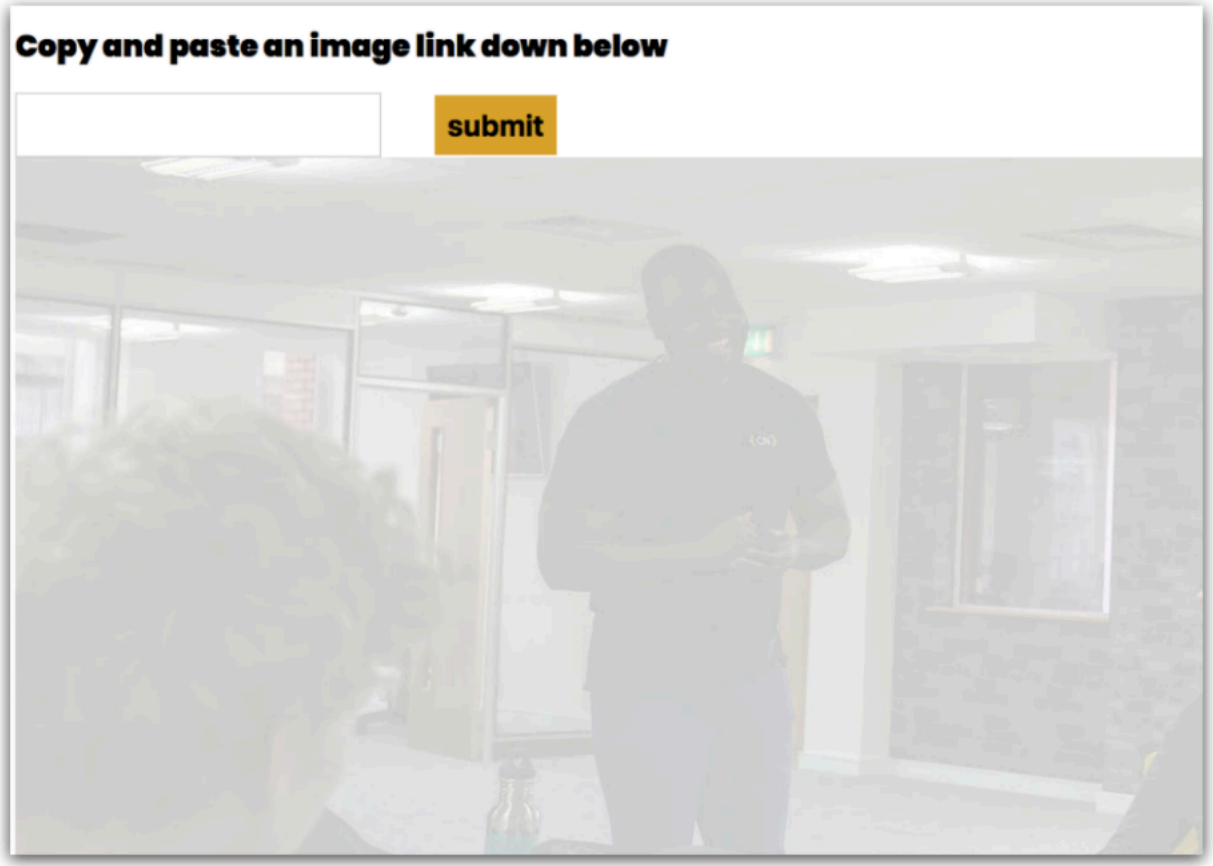
```
const image = document.getElementById("image");
```

in browser

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

DOM





{CODENATION}

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

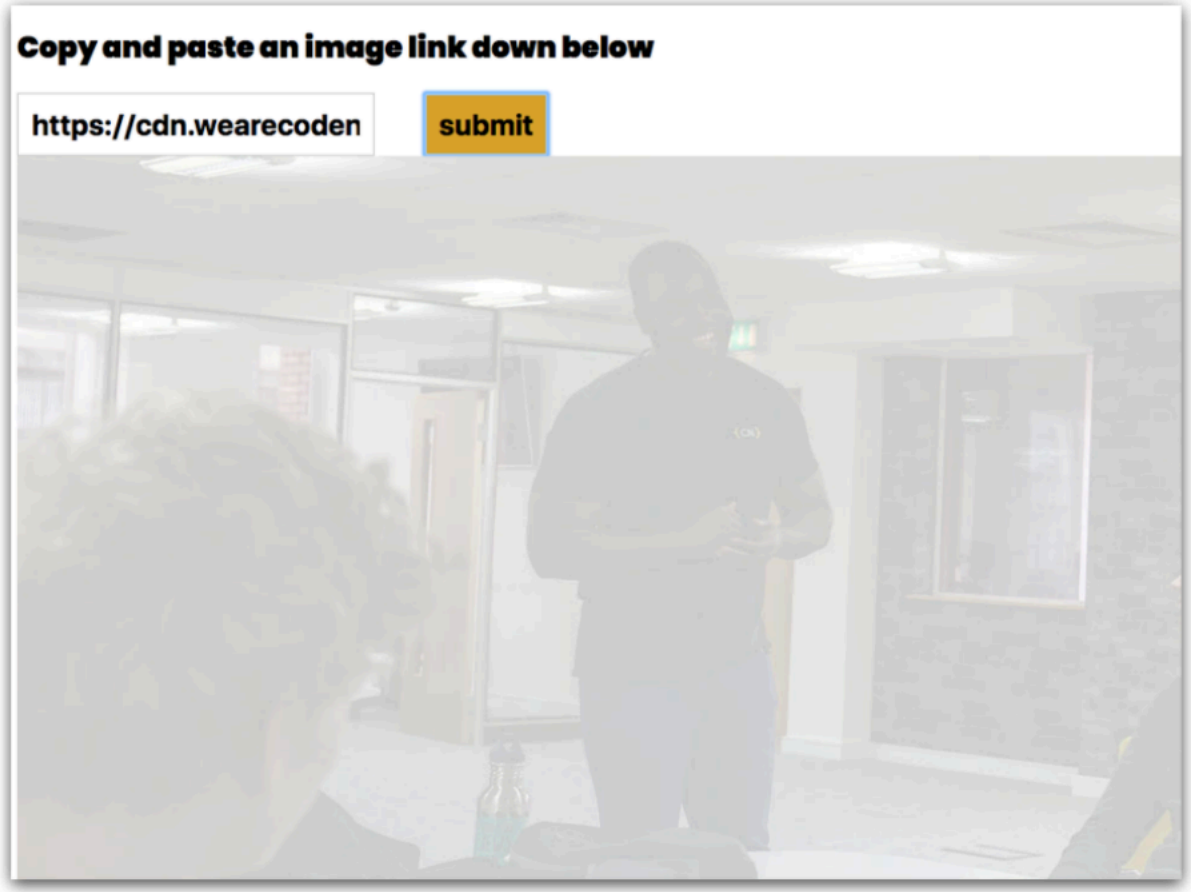
In app.js, check that you have created all variables you need, as below:

```
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;
})
```

DOM



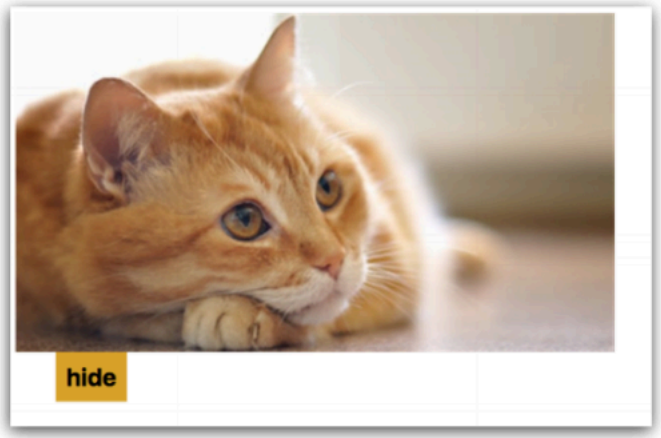
{CODENATION}

Lets look at...

... styling elements!

DOM

```
<!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>
```



{CODENATION}

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);
```

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";
    }
})
```

{CODENATION}

Learning Objectives

To explore the HTML & DOM structure

To be able to apply changes to the DOM by responding to user interaction