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

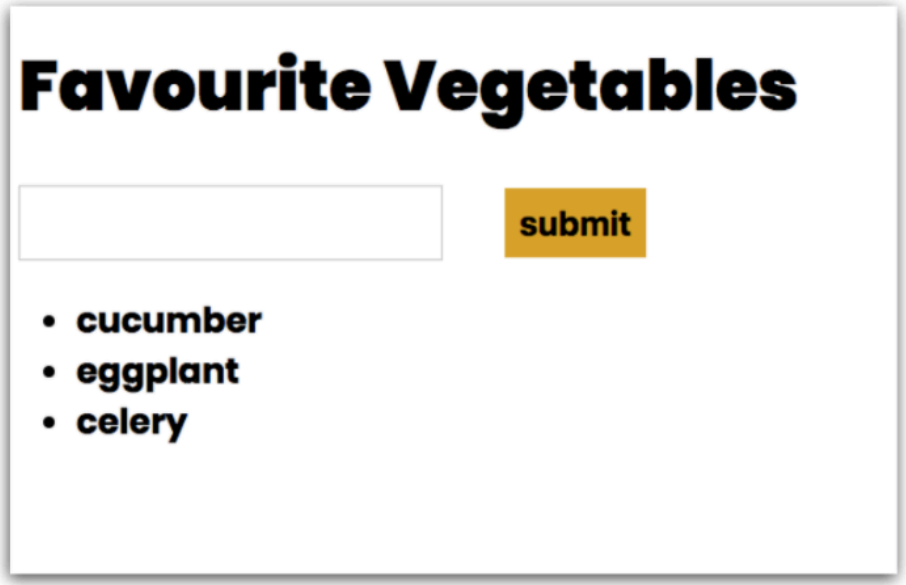
Lets look at...

Creating a **New** Element

Folder 8

DOM

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="../main.css">
    <title>Creating New Elements with JS</title>
</head>
<body>
    <h1>Favourite Vegetables</h1>
    <input id="input" type="text">
    <button id="submit">submit</button>

    <ul id="list">
        <li>cucumber</li>
        <li>eggplant</li>
        <li>celery</li>
    </ul>

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



Folder 8

{CODENATION}

Activity(1): Create Variables

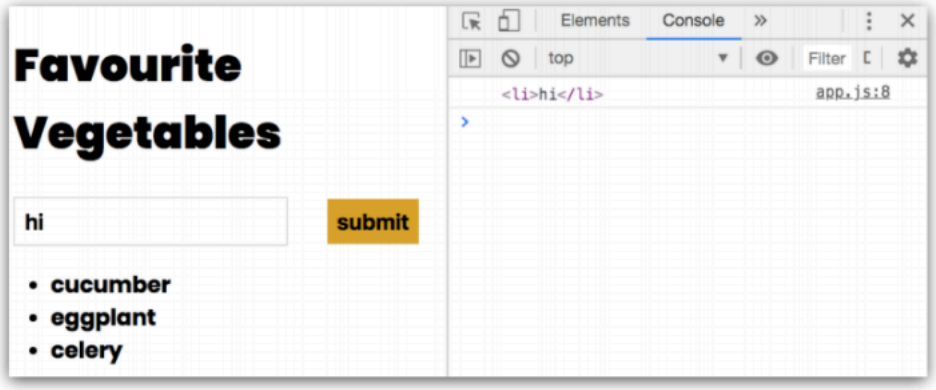
First, set up two variables, for the **input** and **submit**.

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

Activity(2): Create a List Item

Now create a “list” item when the person presses the submit button

```
button.addEventListener("click", ()=> {
    let listItem = document.createElement("li");
    listItem.textContent = input.value;
    //console.log(listItem);
})
```



*It’s stored in the listItem for the time being.

Activity(3): Show updated list

First, create a new variable for the list:

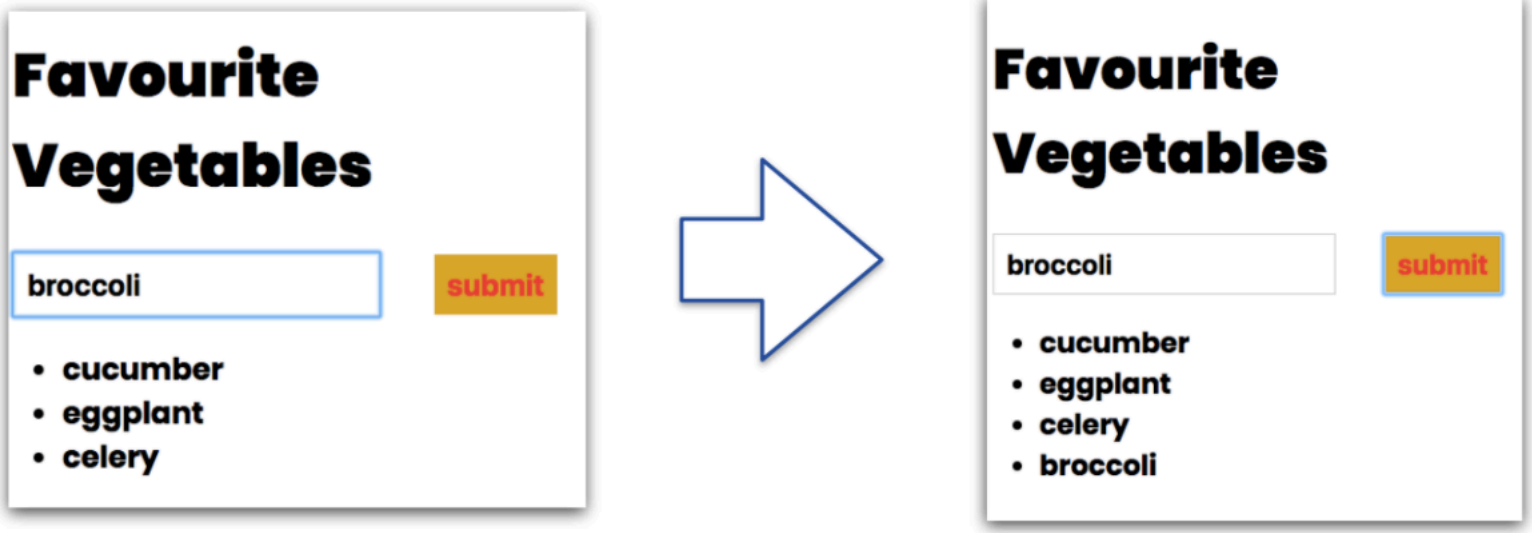
```
let list = document.getElementsByTagName("ul")[0];
```

Then add the following inside the function:

```
list.appendChild(listItem);
```

DOM

```
const input = document.getElementById("input");
const button = document.getElementById("submit");
let list = document.getElementsByTagName("ul")[0];


button.addEventListener("click", ()=> {
    let listItem = document.createElement("li");
    listItem.textContent = input.value;
    list.appendChild(listItem);
    //console.log(listItem);
})
```

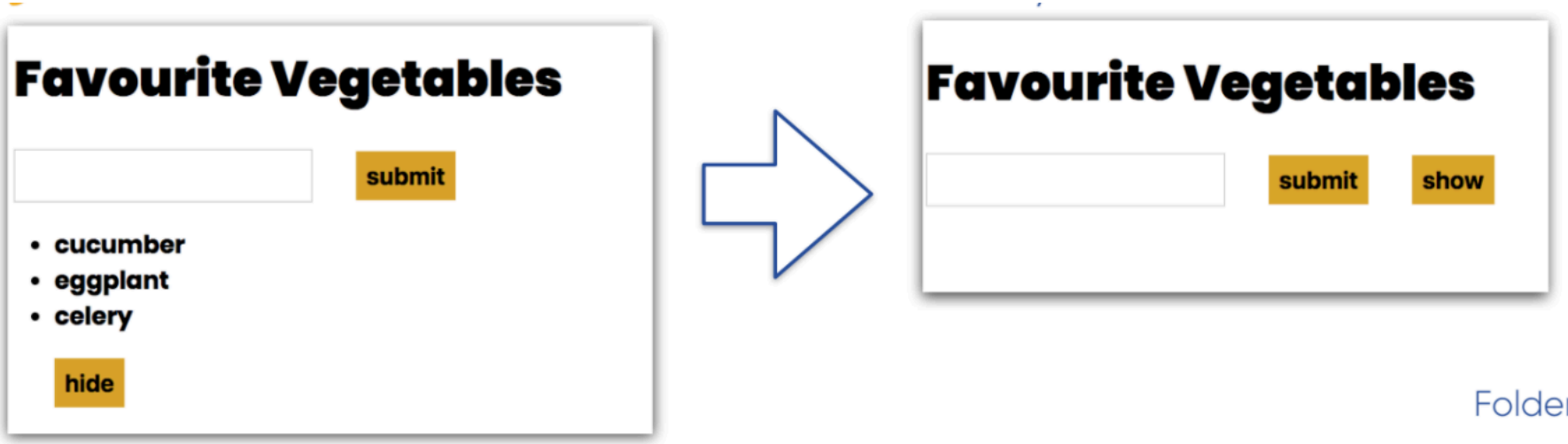


{CODENATION}

Challenge

Clear the input files when the user presses the "submit" button.

Add a feature where the user can show/hide the list



Folder 8

Solution (1 of 2)

Clear the input files when the user presses the "submit" button.

```
input.value ="";
```

Add a feature where the user can show/hide the list

Solution (2 of 2)

Add a feature where the user can show/hide the list

DOM

Add a **new button** in HTML

```
<button id='showhide-btn'>hide</button>
```

In app.js add a **new const**

```
const showhidebtn=document.getElementbyId("showhide-btn");
```

Add a **function**

```
showhidebtn.addEventListener("click", () => {
    let list = document.getElementsByTagName("ul")[0];
    if(list.style.display == "none") {
        list.style.display = "block";
        showhidebtn.textContent = "hide";
    } else {
        list.style.display = "none";
        showhidebtn.textContent = "show";
    }
})
```

{CODENATION}

DOM

```
const input = document.getElementById('input');
const button = document.getElementById('submit');
const showhidebtn = document.getElementById('showhide-btn');

button.addEventListener('click', () => {
    let listItem = document.createElement('li');
    let list = document.getElementsByTagName('ul')[0];
    listItem.textContent = input.value;
    list.appendChild(listItem);
    input.value = '';
})

showhidebtn.addEventListener("click", () => {
    let list = document.getElementsByTagName('ul')[0];
    if(list.style.display == 'none') {
        list.style.display = 'block';
        showhidebtn.textContent = 'hide';
    } else {
        list.style.display = 'none';
        showhidebtn.textContent = 'show';
    }
})
```

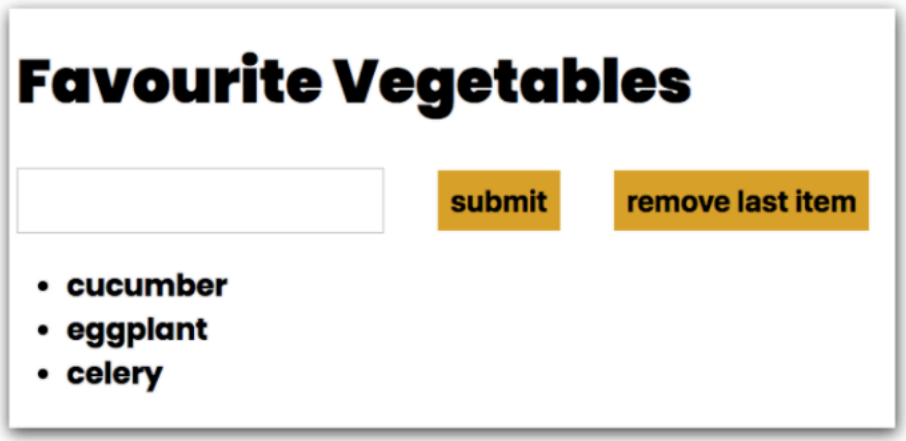
{CN}®

DOM

Lets look at...

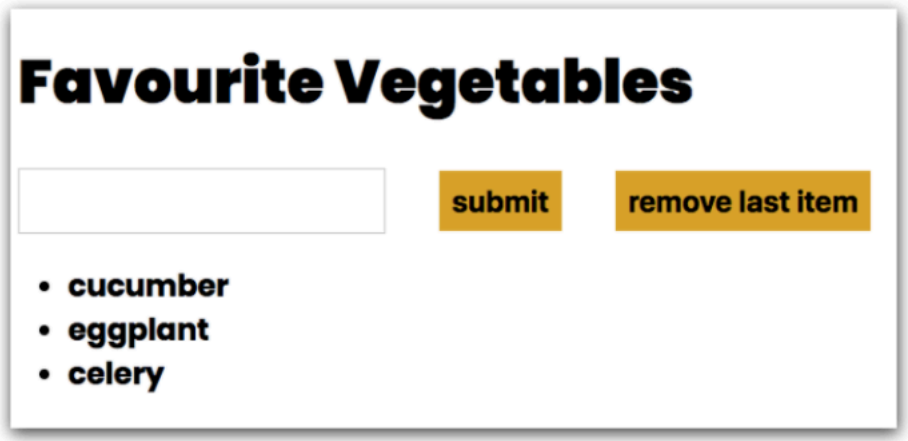
Removing Elements

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="../main.css">
    <title>Removing Elements</title>
</head>
<body>
    <h1>Favourite Vegetables</h1>
    <input id="input" type="text">
    <button id="submit">submit</button>
    <button id="remove">remove last item</button>

    <ul id="list">
        <li>cucumber</li>
        <li>eggplant</li>
        <li>celery</li>
    </ul>

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



DOM

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

/*from previous ex*/
button.addEventListener('click', () => {
    let listItem = document.createElement('li');

    let list = document.getElementsByTagName('ul')[0];
    listItem.textContent = input.value;

    list.appendChild(listItem);

    input.value = '';
})
```



Folder 9

{CODENATION}

Activity: Remove last item

Think about the steps you need:

> Set a **new const** for the remove button.
> Create a **new function** to remove the last item when the button is clicked using the **last child** method:

```
node.removeChild(childElement);
```

Node in this case represents the list of items <li>, so the last child would be:

```
li:last-child
```

Solution: Remove last item

Set a new const for the remove button:

```
const removeBtn = document.getElementById('remove');
```

Create a new function to remove last item when the button is clicked using the last child method:

```
removeBtn.addEventListener('click', () => {

    let lastItem = document.querySelector('li:last-child');
    let list = document.getElementsByTagName('ul')[0];

    list.removeChild(lastItem);
})
```

{CN}®

DOM

Lets look at...

setTimeout()

Times

... are calculated in **milliseconds**. 1,000ms = 1s

Activity: setTimeout()

```
window.setTimeout((something)=> {
    console.log(something);
}, 5000, "Greetings Everyone");
```

What is an event?

{CN}®

DOM

click

dblclick

mousedown

mouseup

mousemove

mouseout

touchstart

touchmove

touchend

keydown

keyup

keypress

More on MDN Event Reference - https://developer.mozilla.org/en-US/docs/Web/Events

{CN}®

DOM

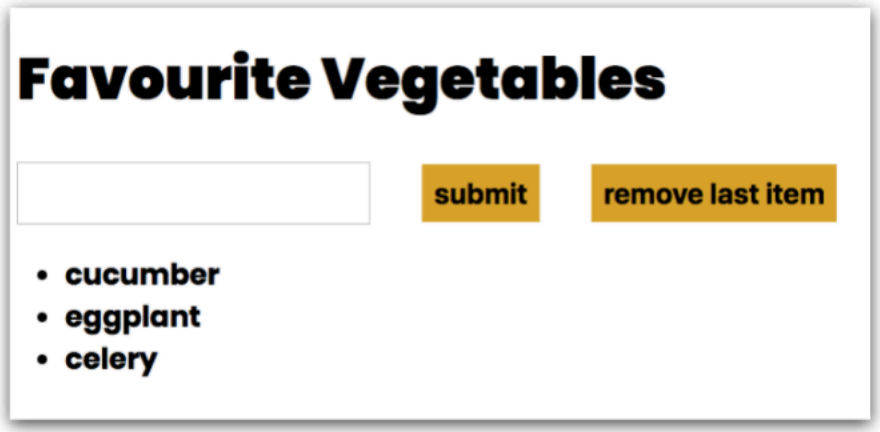
Lets look at...

addEventListener,
mouseover, mouseout

Folder 11

DOM

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="../main.css">
    <title>Mouseover Mouseout</title>
</head>
<body>
    <h1>Favourite Vegetables</h1>

    <input id="input" type="text">
    <button id="submit">submit</button>
    <button id="remove">remove last item</button>

    <ul id="list">
        <li>cucumber</li>
        <li>eggplant</li>
        <li>celery</li>
    </ul>

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

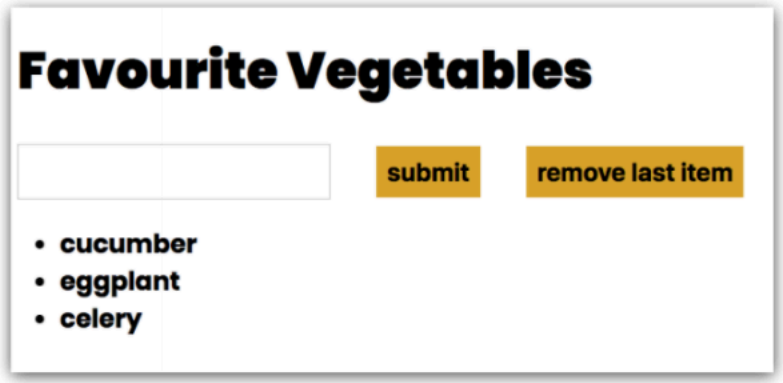


Folder 11

{CODENATION}

DOM

```
const input = document.getElementById('input');
const button = document.getElementById('submit');
const removeBtn = document.getElementById('remove');

button.addEventListener('click', () => {
    let listItem = document.createElement('li');

    let list = document.getElementsByTagName('ul')[0];
    listItem.textContent = input.value;

    list.appendChild(listItem);

    input.value = '';
})

removeBtn.addEventListener('click', () => {
    let listItem = document.querySelector('li:last-child');
    let list = document.getElementsByTagName('ul')[0];

    list.removeChild(listItem);
})
```



Folder 11

{CODENATION}

Activity: add listItem

DOM

Add these to your code:

A new Const

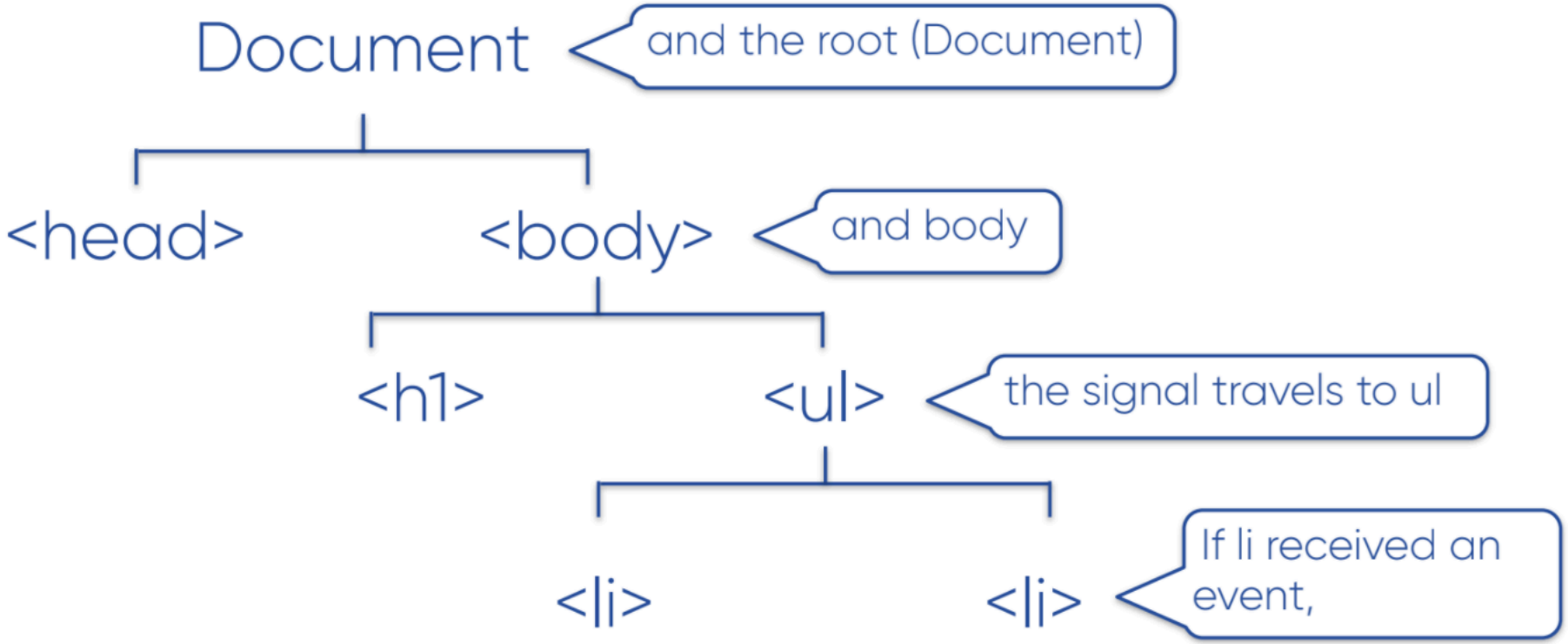
```
const listItems = document.getElementsByTagName("li");
```

Add a for loop:

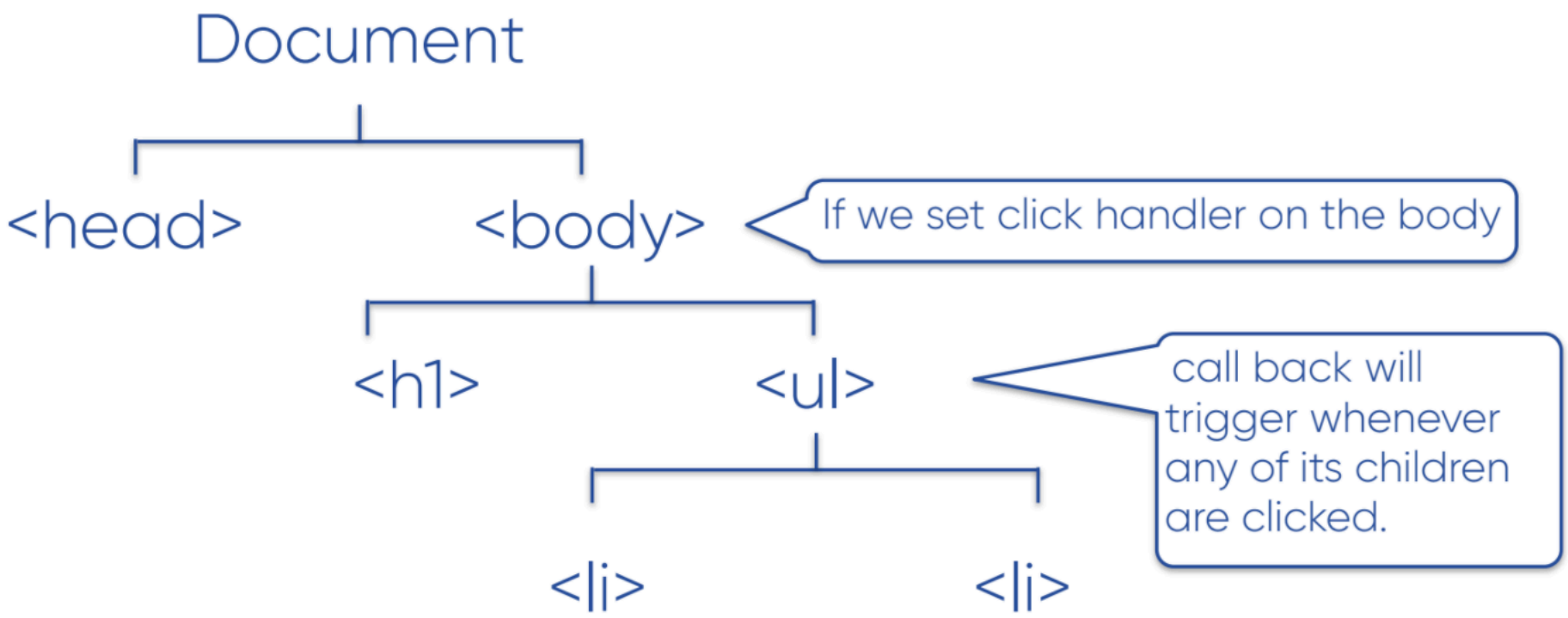
```
for (let listItem of listItems) {
    listItem.addEventListener("mouseover", () => {
        listItem.textContent = listItem.textContent.toUpperCase();
    });

    listItem.addEventListener("mouseout", () => {
        listItem.textContent = listItem.textContent.toLowerCase();
    });
}
```

Event Bubbling

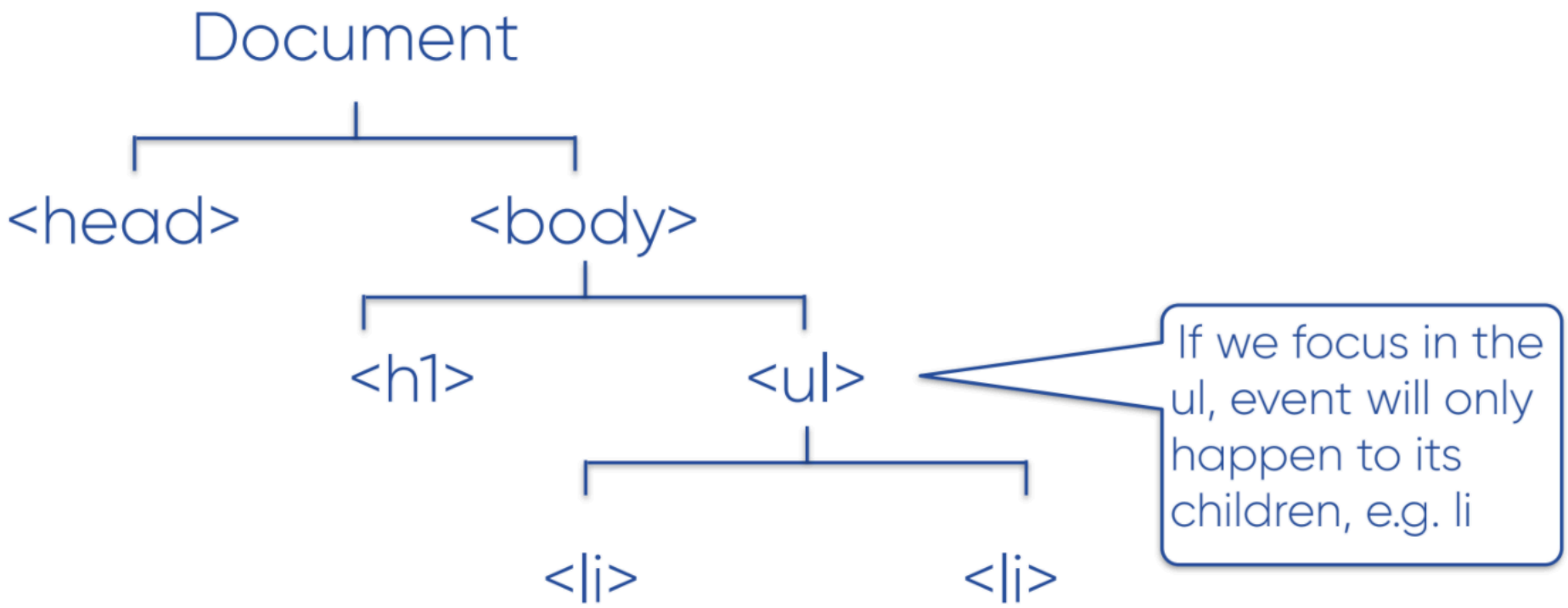
Li receiving a signal



Setting click handler on the body



Setting click handler on the ul



{CN}®

DOM

Lets look at...

Event Object

Folder 12

Activity: checking event in console

Inside app.js, we want to check which object we are clicking:

```
document.addEventListener("click", (event) => {
    console.log(event);
    console.log(event.target);
})
```

Activity: add listItem

Consider our code from earlier. How could we write it **without a loop?**

```
for (let listItem of listItems) {
    listItem.addEventListener("mouseover", () => {
        listItem.textContent = listItem.textContent.toUpperCase();
    });

    listItem.addEventListener("mouseout", () => {
        listItem.textContent = listItem.textContent.toLowerCase();
    });
}
```

Solution: add listItem

Implement into **addEventListener!**

```
list.addEventListener("mouseover", (event) => {
    event.target.textContent = event.target.textContent.toUpperCase();
});

list.addEventListener("mouseout", (event) => {
    event.target.textContent = event.target.textContent.toLowerCase();
});
```

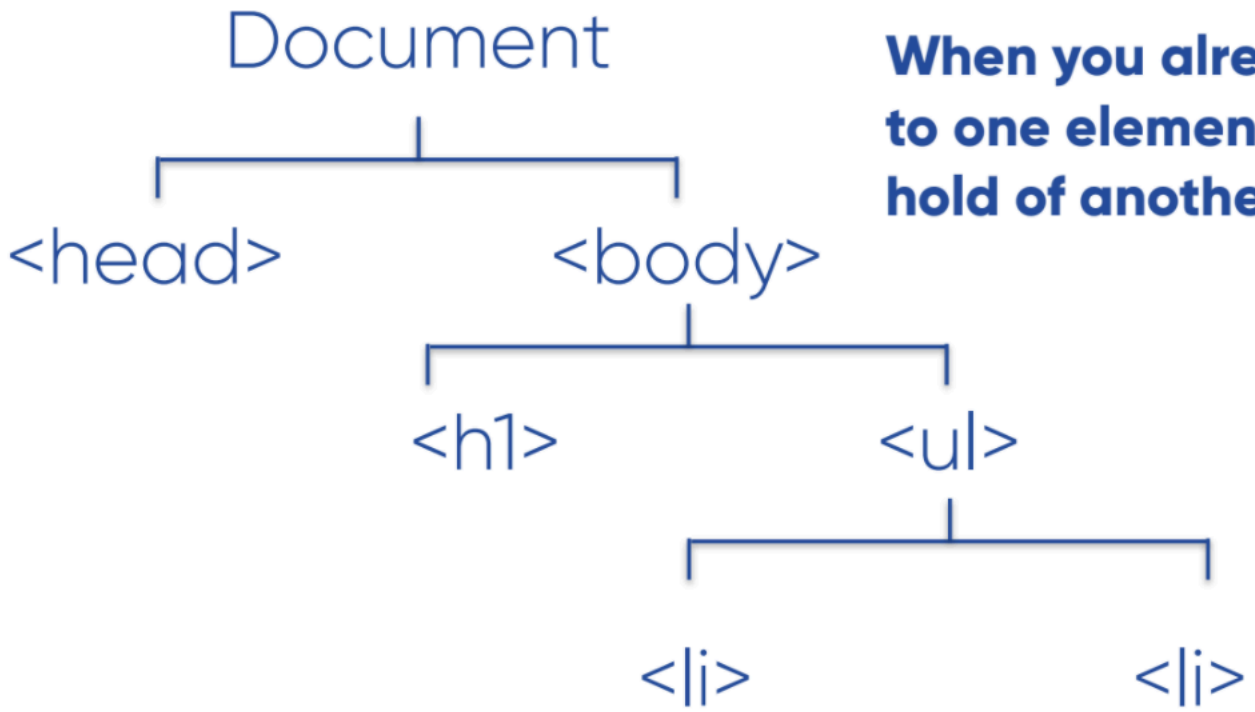
{CN}®

DOM

Lets look at...

Event Object

DOM Traversal



When you already have a reference to one element and you need to get hold of another element nearby.

*It is a way to move from one part of the DOM to another and select an element based on its relationship to another element

Example

```
let paragraph = document.getElementById("myParagraph");
let parent = paragraph.parentNode;
parent.removeChild(paragraph);
```

Activity

using parent node

```
list.addEventListener("click", (event) => {
    const li = event.target;
    const ul = li.parentNode;
    ul.removeChild(li);
});
```

This variable sets up the “co ordinates” of what you click on

this variable sets up UL as the parent of LI which will allow us to remove whichever LI element we click on

```
const input = document.getElementById('input');
const button = document.getElementById('submit');
const removeBtn = document.getElementById('remove');
const listItem = document.getElementsByTagName('li');
const list = document.getElementById('list');

button.addEventListener('click', () => {
    let listItem = document.createElement('li');
    let list = document.getElementsByTagName('ul')[0];

    listItem.textContent = input.value;
    list.appendChild(listItem);
    input.value = '';
})

list.addEventListener('click', (event) => {
    const li = event.target;
    const ul = li.parentNode;
    ul.removeChild(li);
});
```

So the li will be removed when clicked on.

{CODENATION}

Learning Objectives

To explore the HTML & DOM structure

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

More Info

https://developer.mozilla.org/en-US/docs/Web/Events

https://caniuse.com/