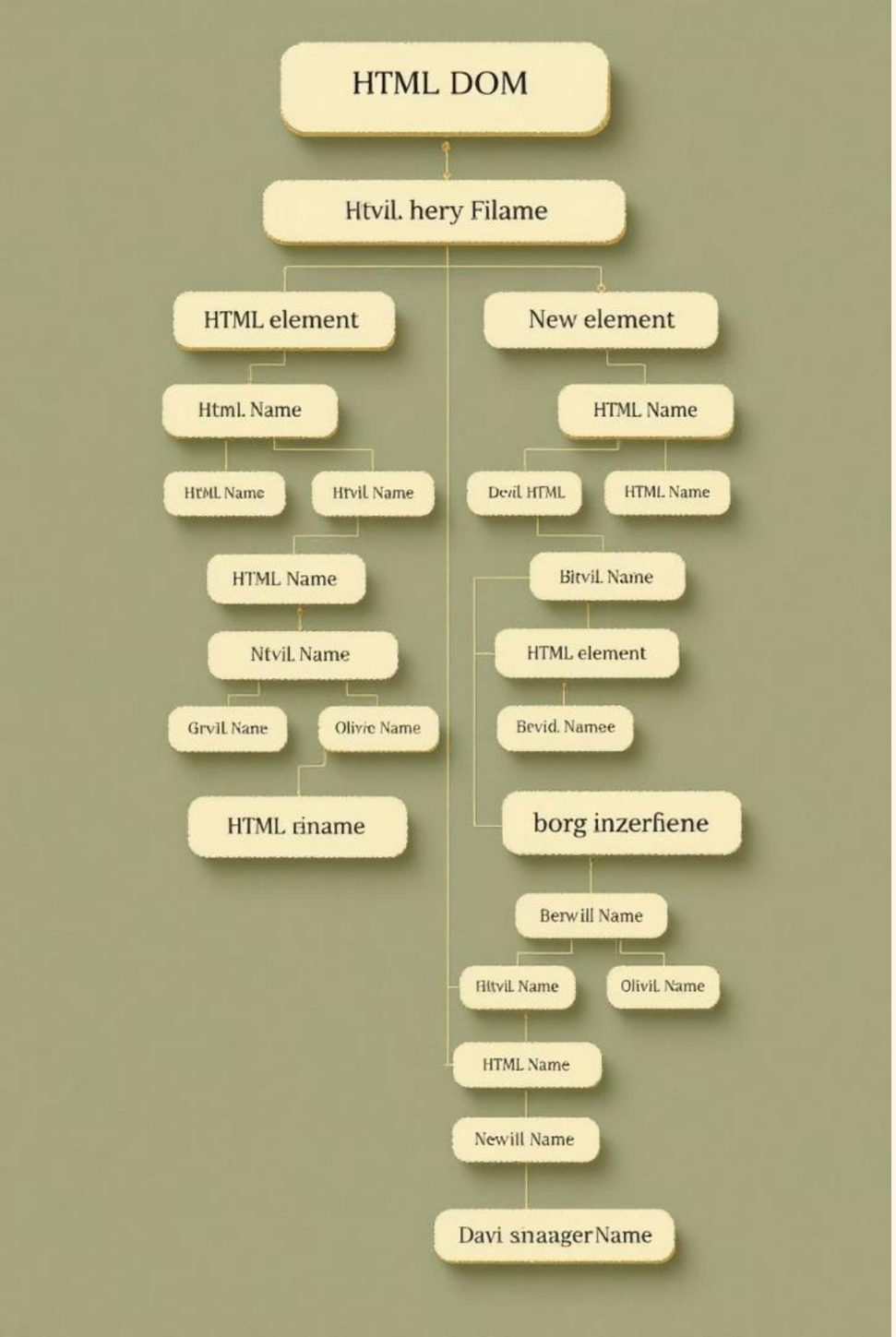
JavaScript and DOM Interaction

Manipulating HTML and CSS using JavaScript

by : Asst. Lect. Ali Al-khawaja



What is the DOM?

Programming Interface

DOM is a programming interface for web documents.

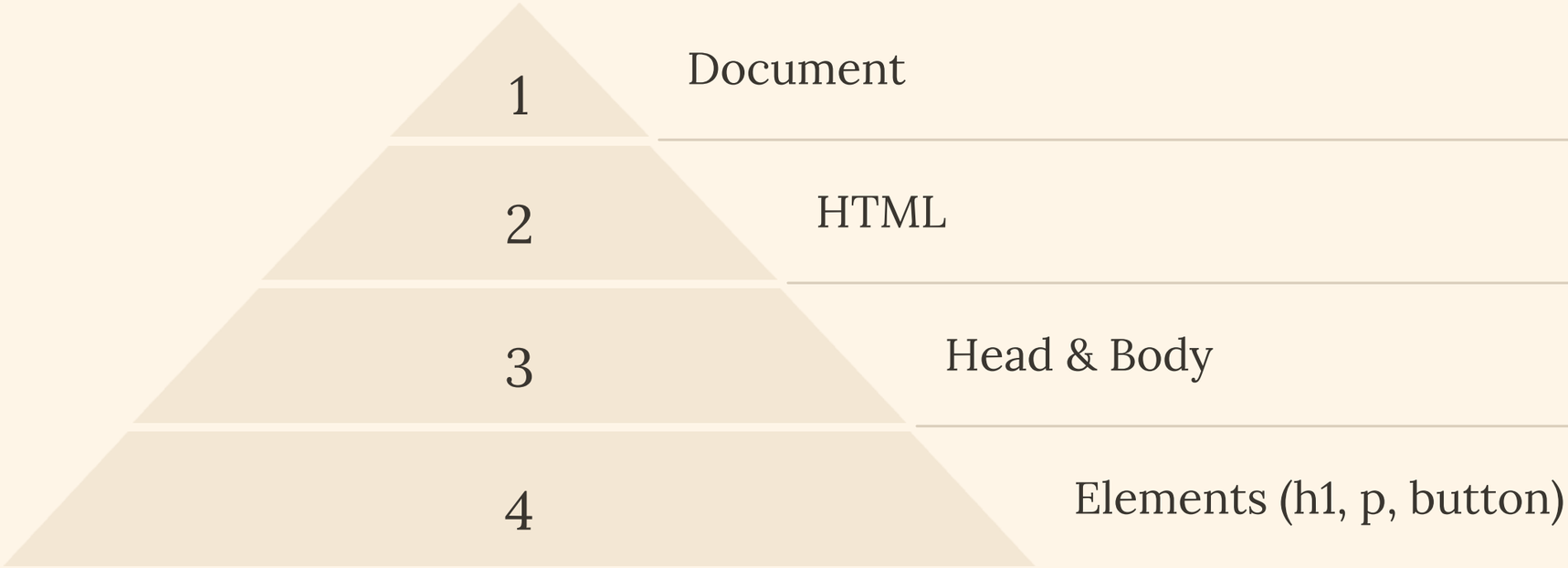
Tree Structure

It represents HTML documents as a tree where each element is a node.

Dynamic Interaction

Allows JavaScript to interact with HTML elements dynamically.

DOM Tree Structure



Each node represents an element, attribute, or text in the HTML document.

Selecting Elements by ID

Define HTML

Create HTML with unique ID attributes.

Use getElementById()

Select specific elements using their ID.

Access Content

Use properties like textContent to read element data.

```
let title = document.getElementById("title");
console.log(title.textContent); // Output: Hello, World!
```

Selecting Elements by Class & Tag

getElementsByClassName()

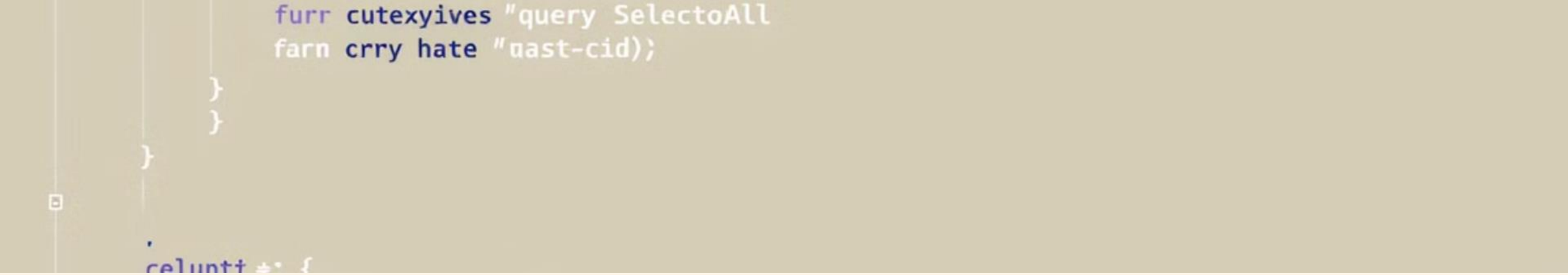
Returns a collection of elements with the specified class.

```
let paragraphs =
document.getElementsByClassName("info");
```

getElementsByTagName()

Returns a collection of elements with the specified tag.

```
let divs = document.getElementsByTagName("div");
```



Modern Methods – querySelector & querySelectorAll

1

querySelector()

Selects the first matching element using CSS selector syntax.

2

querySelectorAll()

Selects all matching elements using CSS selector syntax.

3

Powerful Selection

Can use complex CSS selectors for precise targeting.

```
let heading = document.querySelector("h1");
console.log(heading.textContent);

let allParagraphs = document.querySelectorAll("p");
console.log(allParagraphs.length);
```

Changing Element Content

1 **textContent**

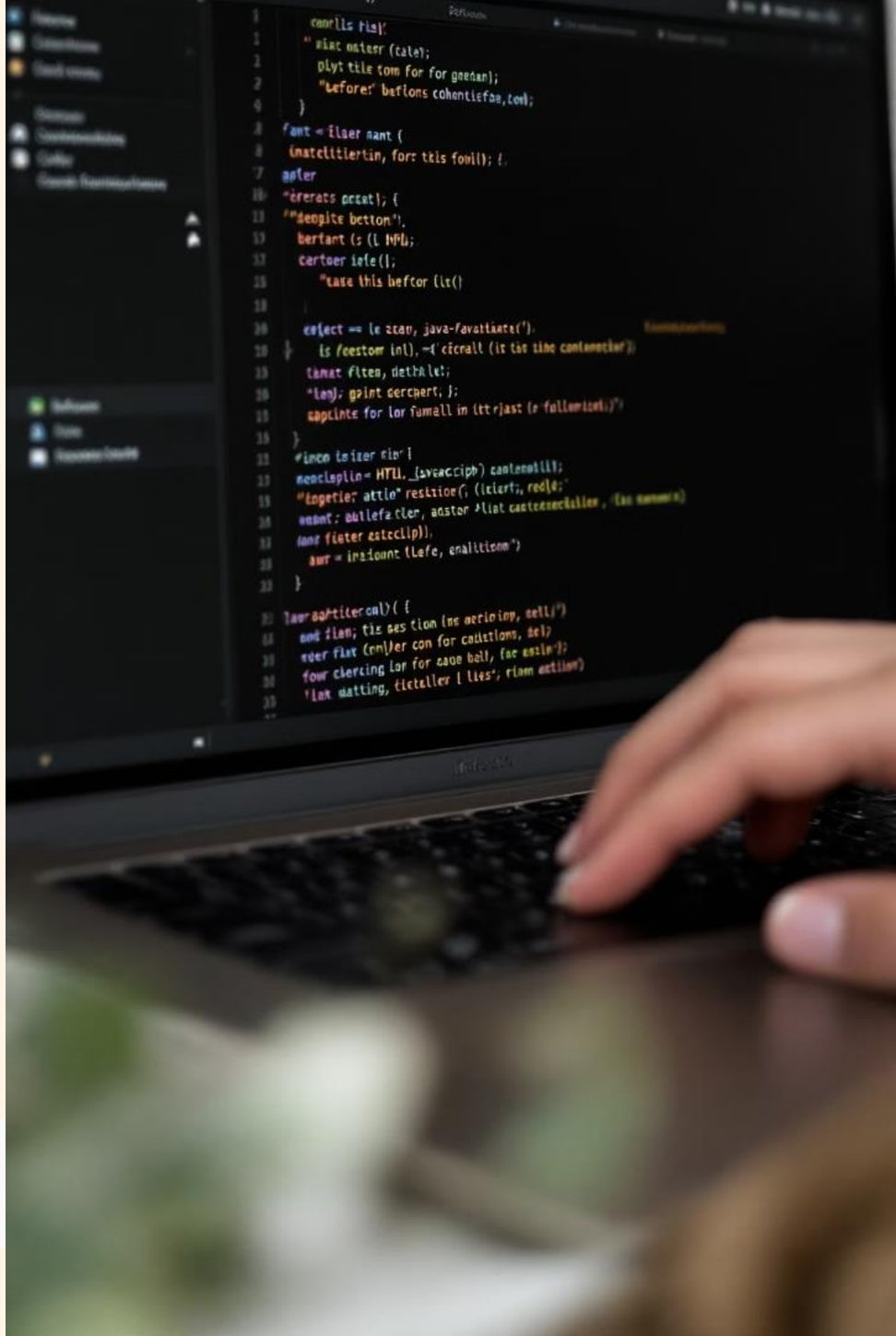
Changes text inside elements without parsing HTML.

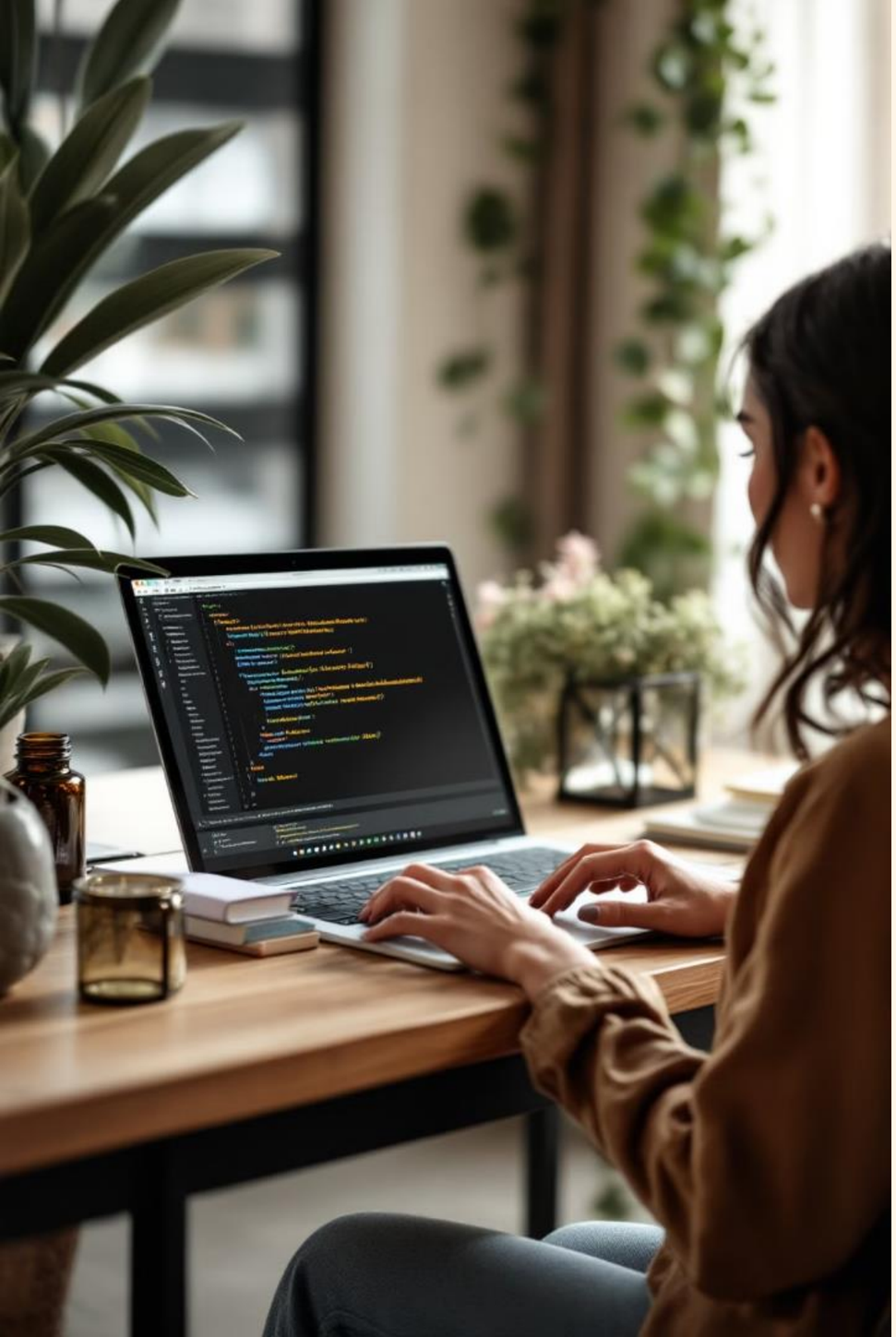
2 **innerHTML**

Allows inserting HTML content into elements.

```
let heading = document.querySelector("h1");
heading.textContent = "Welcome to JavaScript!";
```

```
document.querySelector("#content").innerHTML = "<strong>Updated Content</strong>";
```





Modifying Attributes

setAttribute()

Sets the value of an attribute on the specified element.

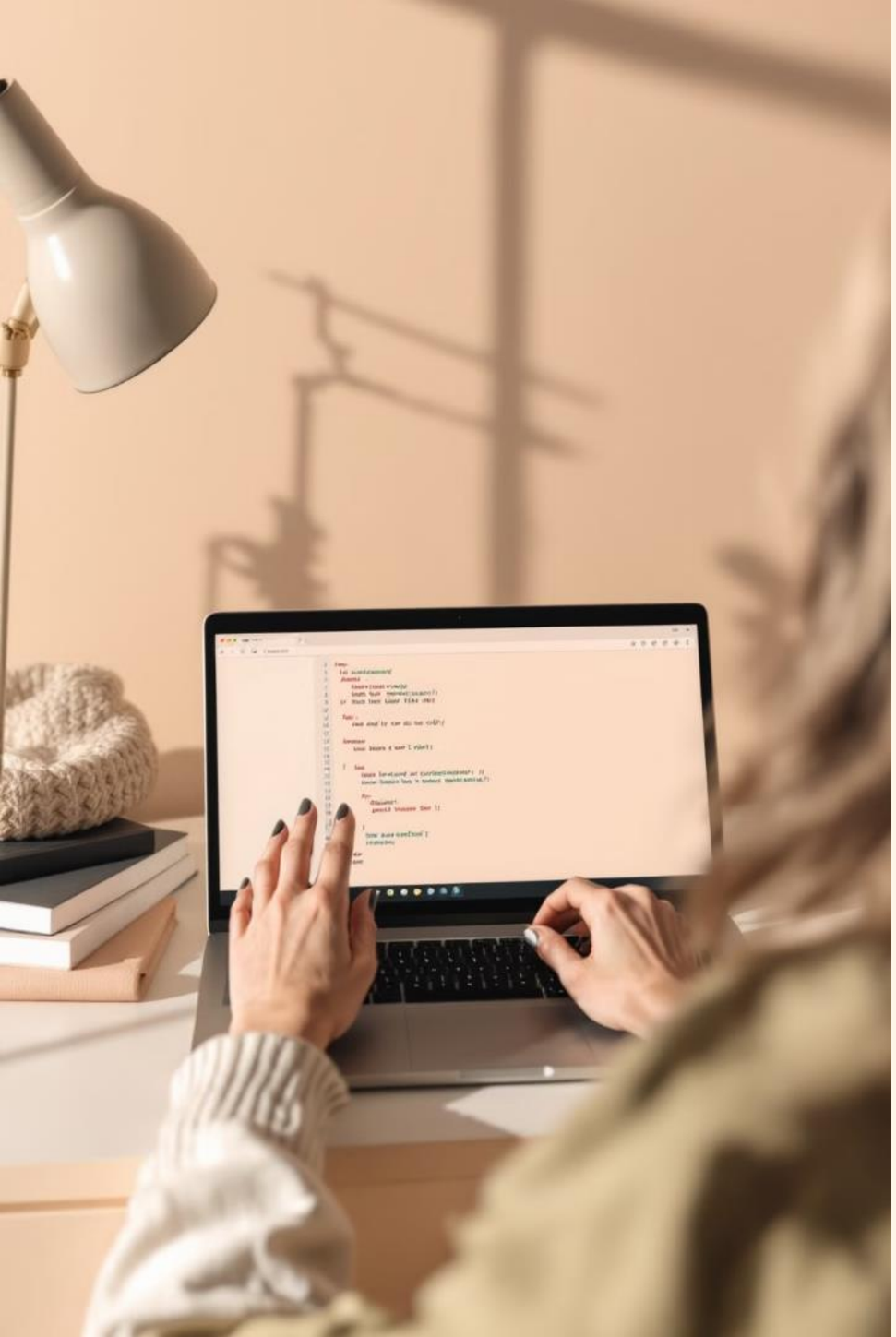
getAttribute()

Gets the value of an attribute on the specified element.

removeAttribute()

Removes an attribute from the specified element.

```
let link = document.querySelector("a");
link.setAttribute("href", "https://example.com");
console.log(link.getAttribute("href"));
```



Modifying CSS Styles

Use **style** property to change CSS dynamically.

```
let title = document.getElementById("title");
title.style.color = "blue";
title.style.fontSize = "24px";
```

Adding and Removing CSS Classes:

```
title.classList.add("highlight");
title.classList.remove("highlight");
```

What are Events?

Click

User clicks on an element.

Mouseover

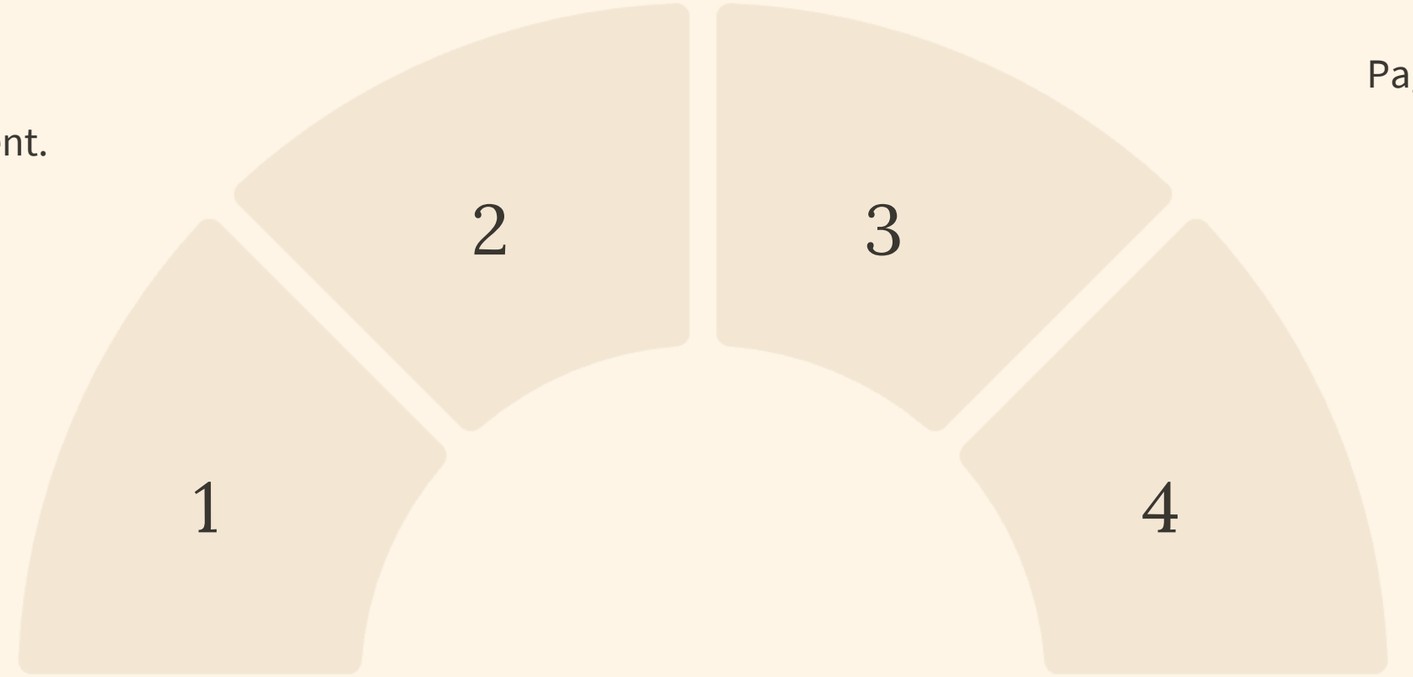
Cursor moves over an element.

Keydown

User presses a key on the keyboard.

Load

Page or resource finishes loading.



Handling **Click** Events

Select Element

Use querySelector to target the element.

Add Event Listener

Use addEventListener with "click" event type.

Define Callback

Create function to execute when event occurs.

```
document.querySelector("button").addEventListener("click", function() {
    alert("Button Clicked!");
});
```

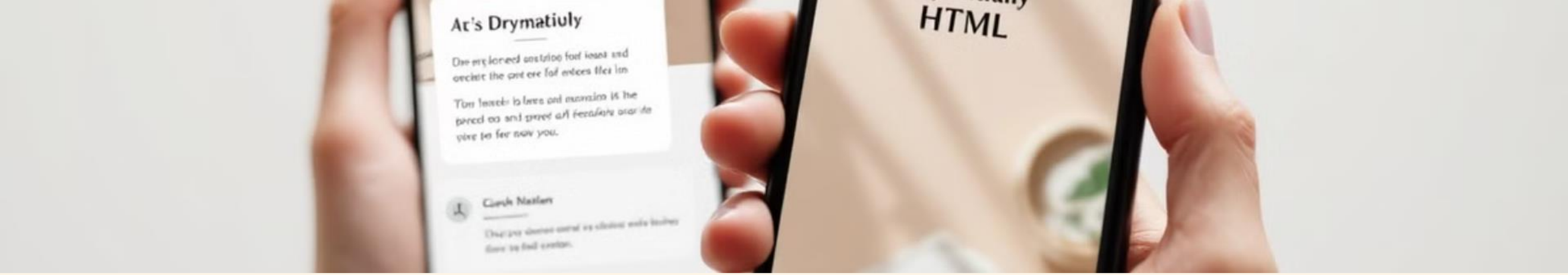
Event Listeners – **mouseover** and **keydown**

Mouseover Event

```
let box = document.getElementById("box");
box.addEventListener("mouseover", function() {
    box.style.backgroundColor = "lightblue";
});
```

Keydown Event

```
document.addEventListener("keydown", function(event) {
    console.log("Key pressed:", event.key);
});
```



Creating Elements

1

Create Element

Use document.createElement() to create a new DOM element.

2

Set Properties

Add content, attributes, and styles to the new element.

3

Append to DOM

Use appendChild() to add the element to the document.

```
let newParagraph = document.createElement("p");
newParagraph.textContent = "This is a new paragraph!";
document.body.appendChild(newParagraph);
```

Removing Elements

element.remove()

Modern method to remove an element directly.

parent.removeChild()

Traditional method requiring parent reference.

Element Cleanup

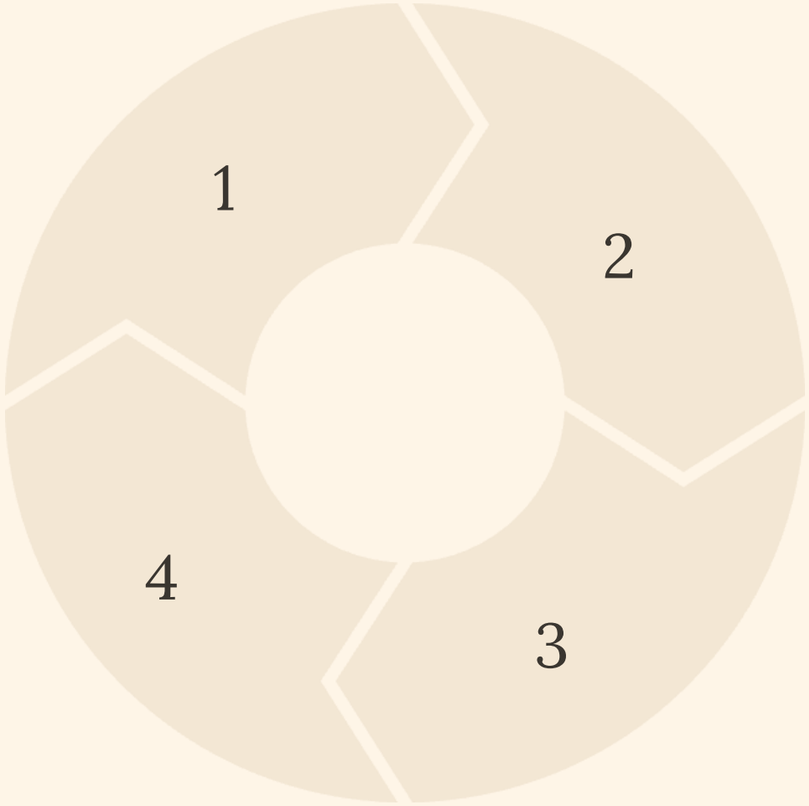
Remove event listeners before removing elements.

```
let element = document.getElementById("myDiv");
element.remove();
```

Exercise 1 – Change Text on Button Click

HTML Setup

Create paragraph and button elements.



JavaScript Function

Write changeText() function.

Connect Event

Add onclick attribute to button.

Test Interaction

Click button to see text change.

Exercise 2 – Toggle Dark Mode

1

HTML/CSS Setup

Create button and dark-mode class.

2

Add Event Listener

Listen for button clicks.

3

Toggle Class

Add/remove dark-mode class on body.

Summary

1

DOM Interaction

JavaScript can interact with HTML dynamically via the DOM.

2

Element Selection

Use selectors to access elements for manipulation.

3

Dynamic Modification

Change content, attributes, and styles on the fly.

4

Event Handling

Respond to user interactions with event listeners.