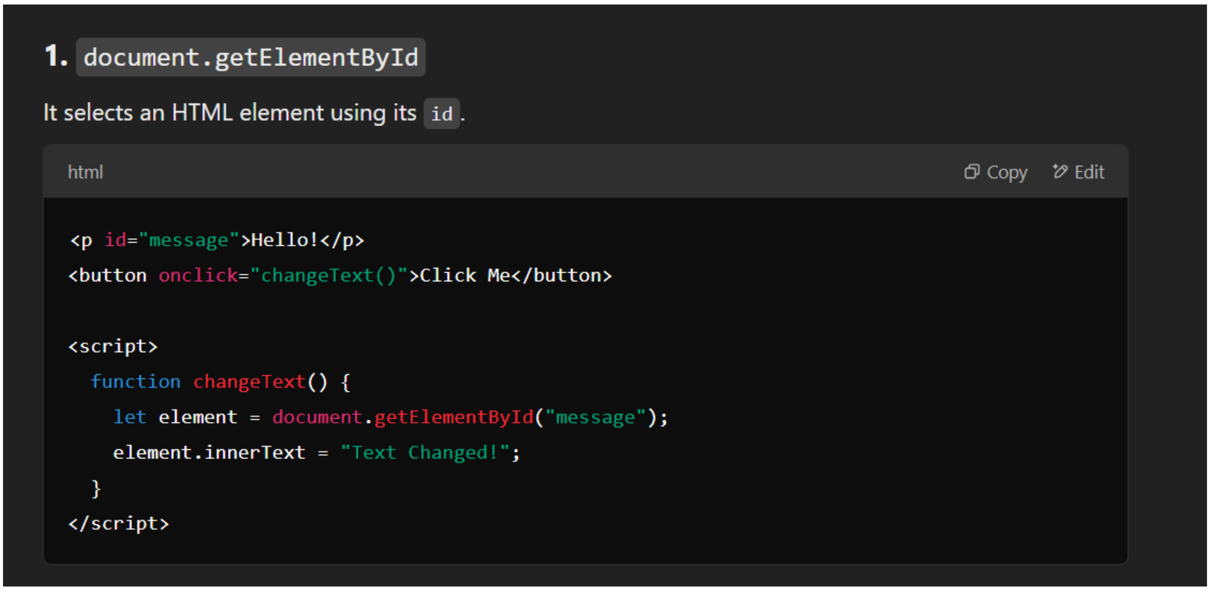
Day-1 week-1



1. `document.getElementById`

It selects an HTML element using its `id`.

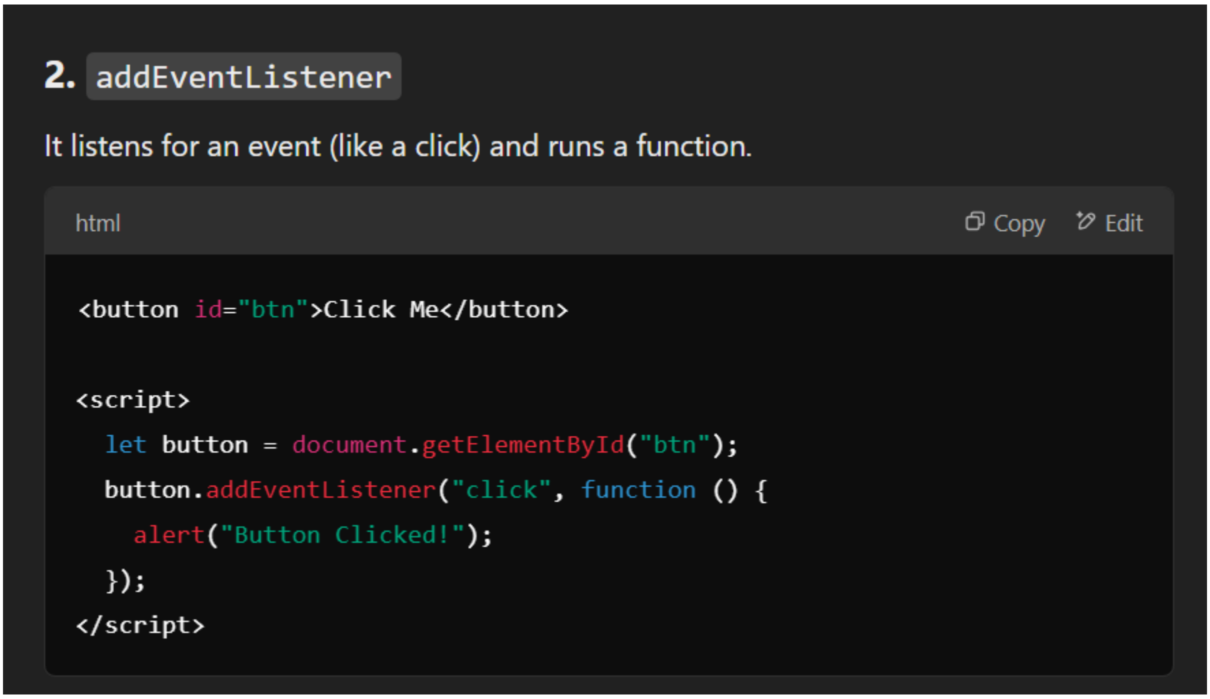
```html
<p id="message">Hello!</p>
<button onclick="changeText()">Click Me</button>

<script>
  function changeText() {
    let element = document.getElementById("message");
    element.innerText = "Text Changed!";
  }
</script>
```

s

document.getElementById-it is used to select html using id tag

What happens?

- The function changeText() gets the <p> element by its id and updates its text when the button is clicked.



2. `addEventListener`

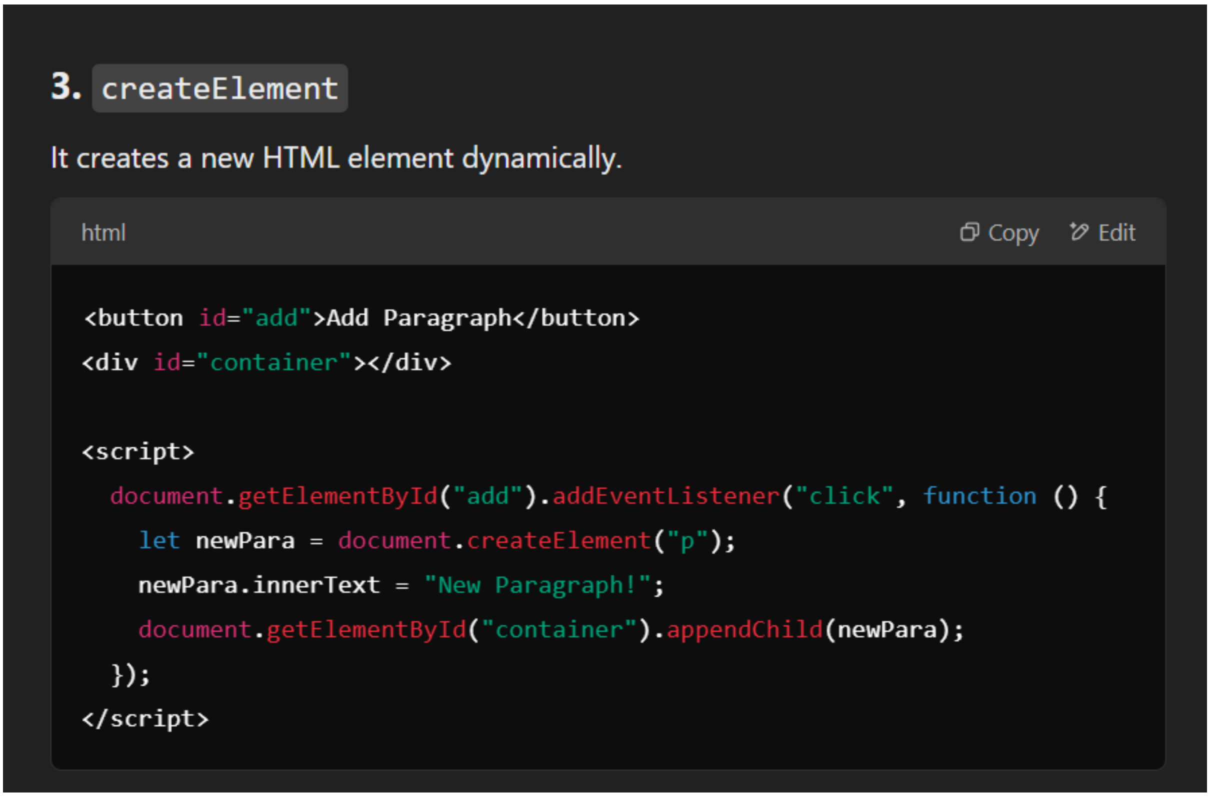
It listens for an event (like a click) and runs a function.

```html
<button id="btn">Click Me</button>

<script>
  let button = document.getElementById("btn");
  button.addEventListener("click", function () {
    alert("Button Clicked!");
  });
</script>
```

addEventListener -It performs and event

What happens?

- When the button is clicked, an alert pops up.



3. `createElement`

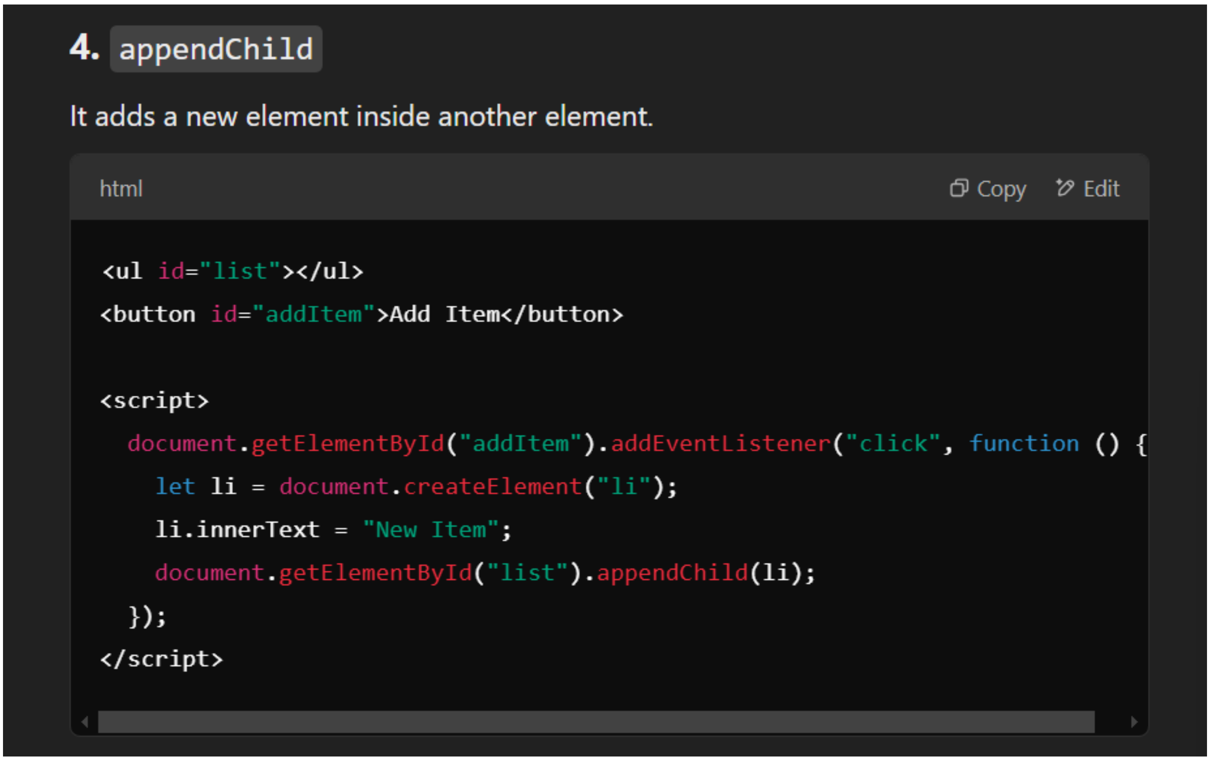
It creates a new HTML element dynamically.

```html
<button id="add">Add Paragraph</button>
<div id="container"></div>

<script>
  document.getElementById("add").addEventListener("click", function () {
    let newPara = document.createElement("p");
    newPara.innerText = "New Paragraph!";
    document.getElementById("container").appendChild(newPara);
  });
</script>
```

createElement -It create a new html element dynamically

What happens?

- Clicking the button adds a new <p> inside the <div>.



4. `appendChild`

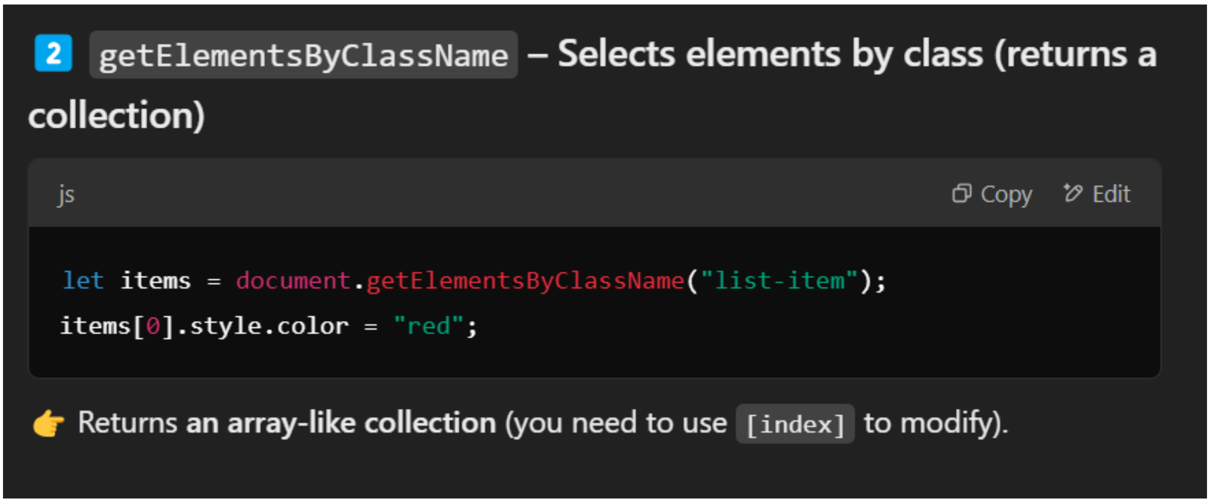
It adds a new element inside another element.

```html
<ul id="list"></ul>
<button id="addItem">Add Item</button>

<script>
  document.getElementById("addItem").addEventListener("click", function () {
    let li = document.createElement("li");
    li.innerText = "New Item";
    document.getElementById("list").appendChild(li);
  });
</script>
```

appendChild---It adds a new element inside another element

What happens?

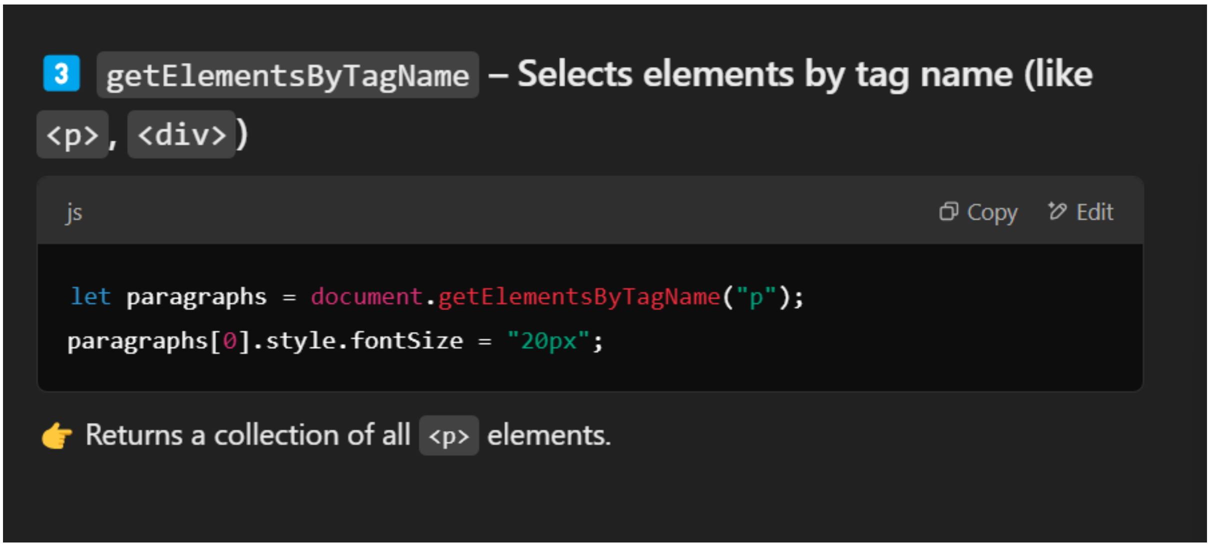
- Clicking the button adds a new list item to the <ul>.



2 `getElementsByClassName` – Selects elements by class (returns a collection)

```js
let items = document.getElementsByClassName("list-item");
items[0].style.color = "red";
```

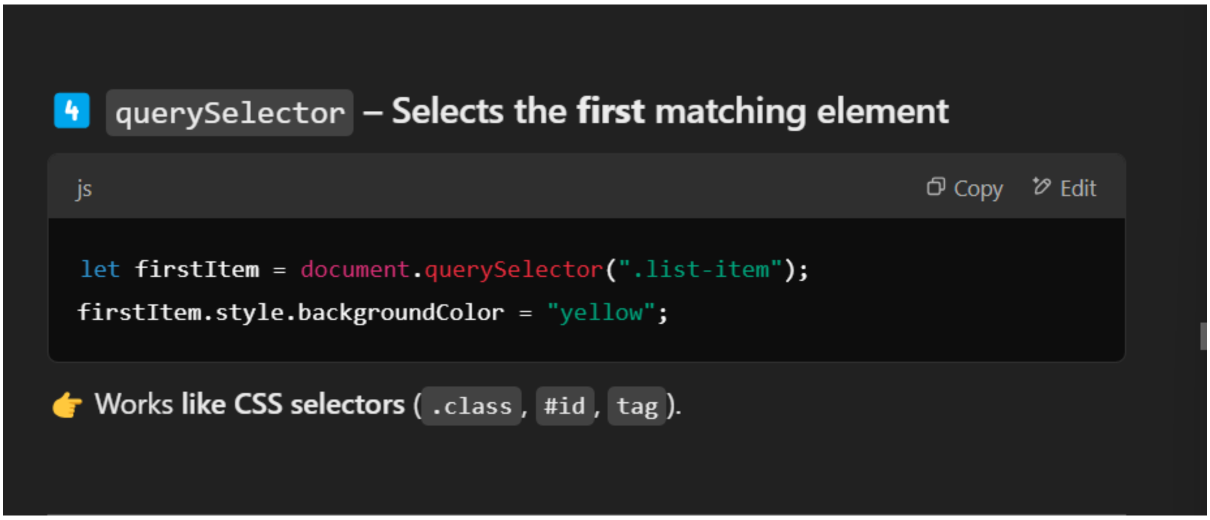
👉 **Returns an array-like collection** (you need to use `[index]` to modify).



3️⃣ `getElementsByTagName` – Selects elements by tag name (like `<p>`, `<div>`)

```js
let paragraphs = document.getElementsByTagName("p");
paragraphs[0].style.fontSize = "20px";
```

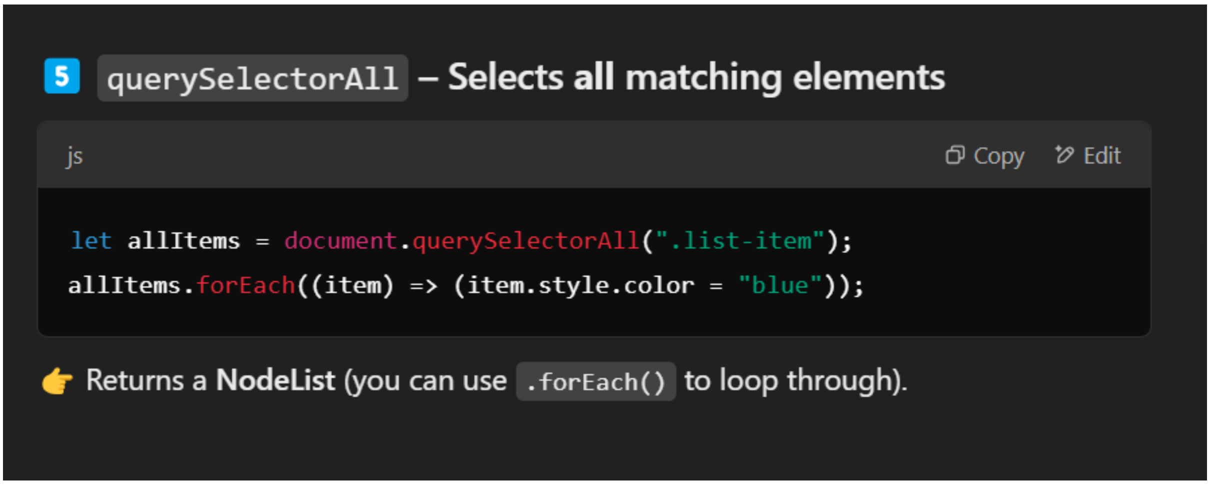
👉 Returns a collection of all `<p>` elements.



4️⃣ `querySelector` – Selects the first matching element

```js
let firstItem = document.querySelector(".list-item");
firstItem.style.backgroundColor = "yellow";
```

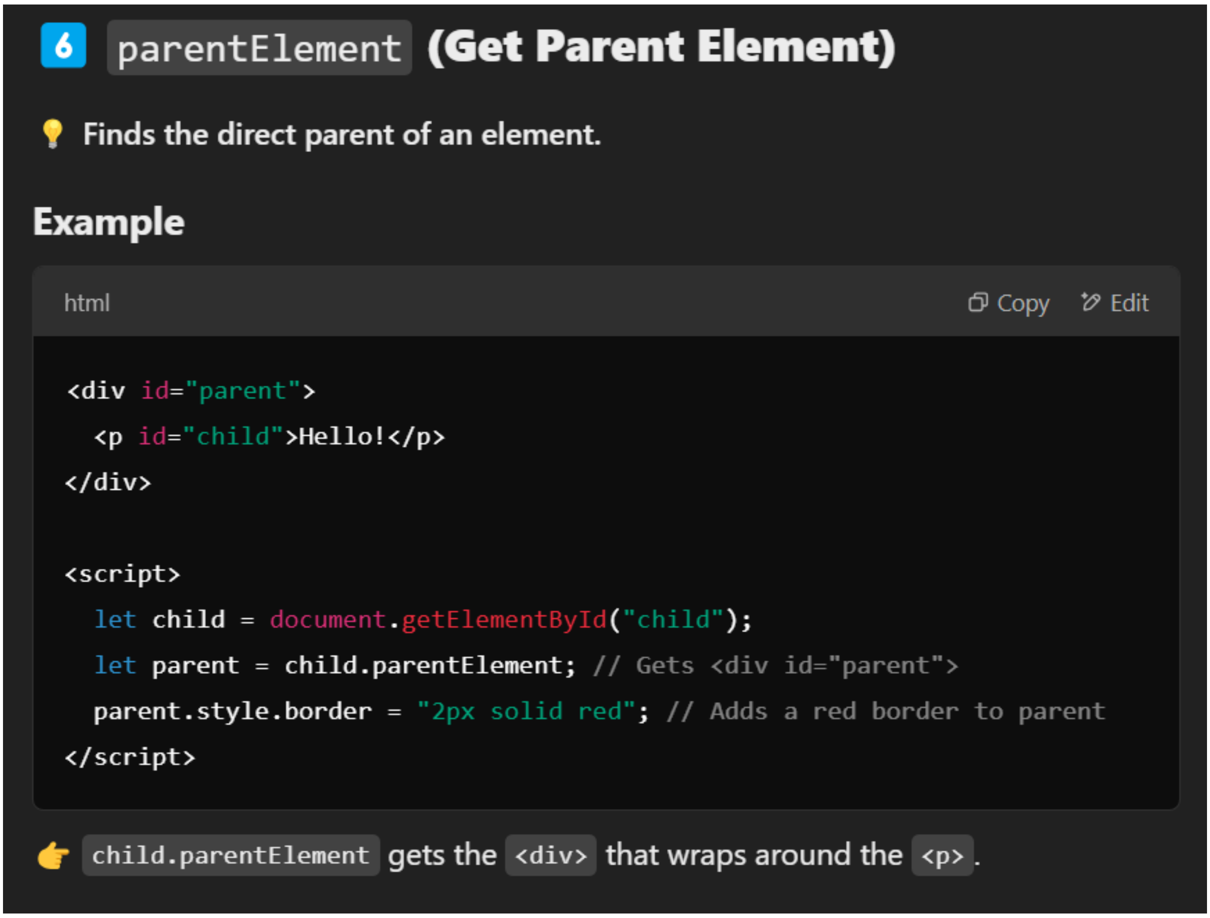
👉 **Works like CSS selectors** (`.class`, `#id`, `tag`).



5️⃣ `querySelectorAll` – Selects all matching elements

```js
let allItems = document.querySelectorAll(".list-item");
allItems.forEach((item) => (item.style.color = "blue"));
```

👉 Returns a **NodeList** (you can use `.forEach()` to loop through).



6️⃣ `parentElement` (Get Parent Element)

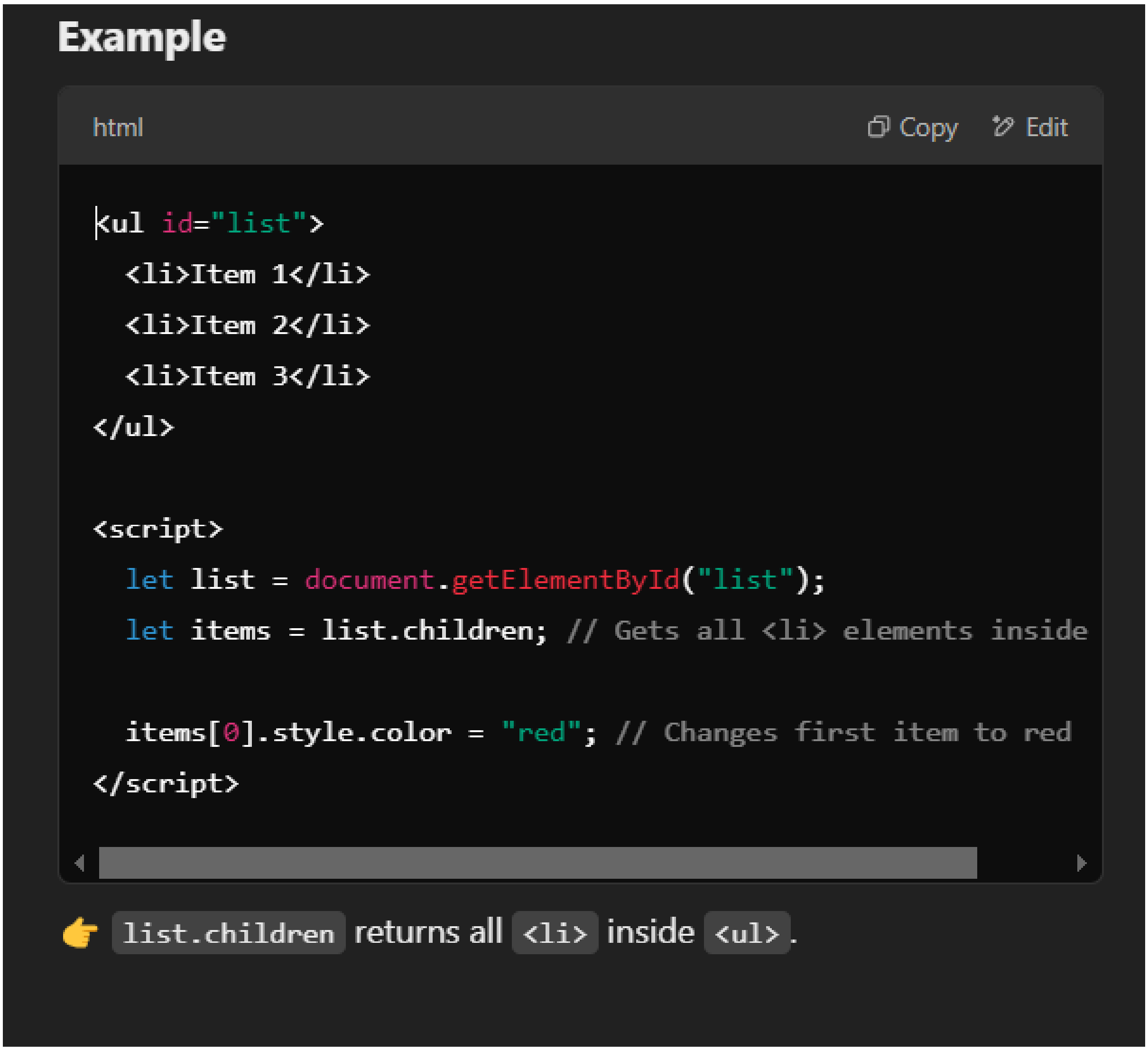
💡 Finds the direct parent of an element.

Example

```html
<div id="parent">
  <p id="child">Hello!</p>
</div>

<script>
  let child = document.getElementById("child");
  let parent = child.parentElement; // Gets <div id="parent">
  parent.style.border = "2px solid red"; // Adds a red border to parent
</script>
```

👉 `child.parentElement` gets the `<div>` that wraps around the `<p>`.

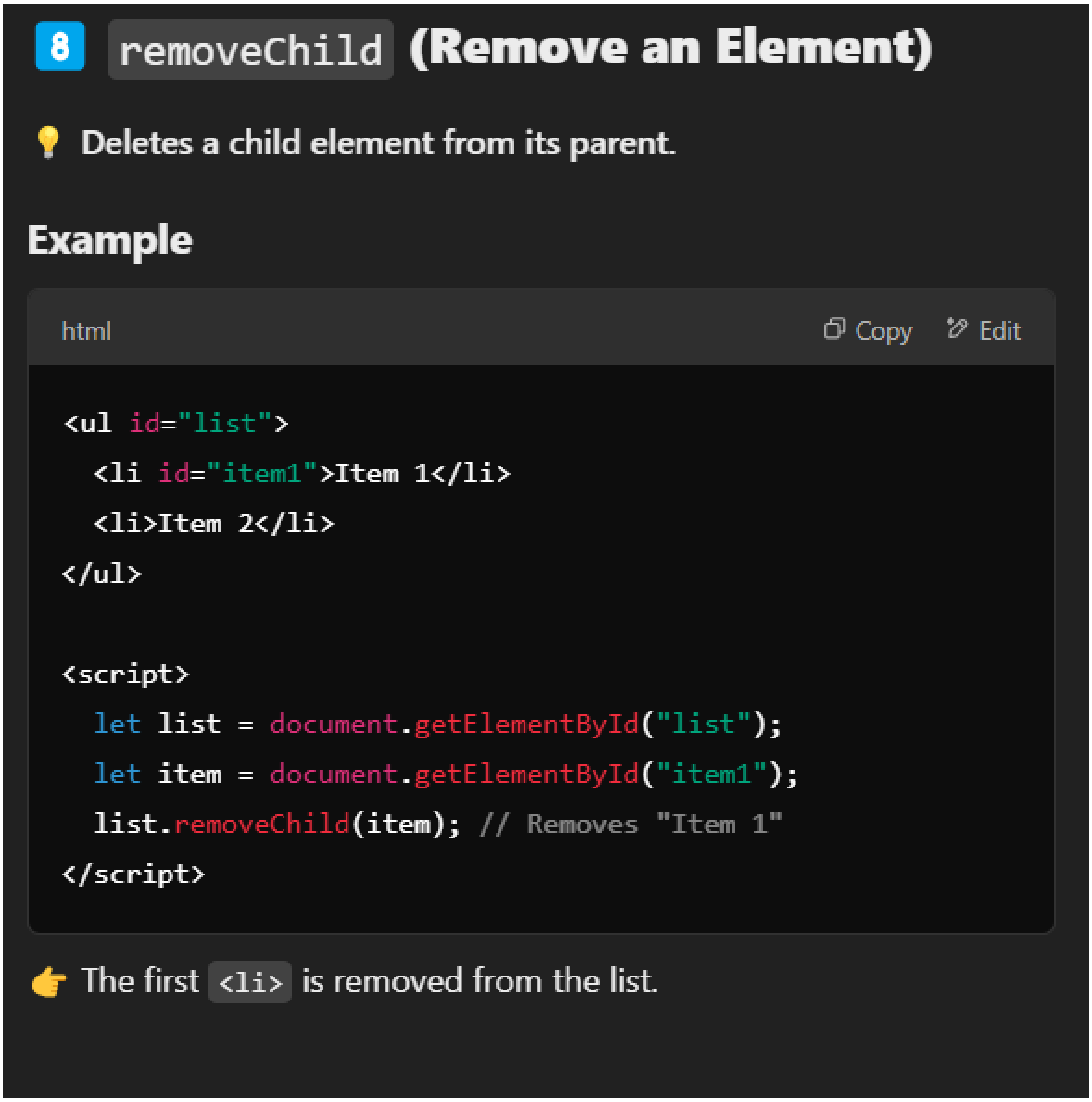
–



Example

```html
<ul id="list">
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ul>

<script>
  let list = document.getElementById("list");
  let items = list.children; // Gets all <li> elements inside

  items[0].style.color = "red"; // Changes first item to red
</script>
```

👉 `list.children` returns all `<li>` inside `<ul>`.



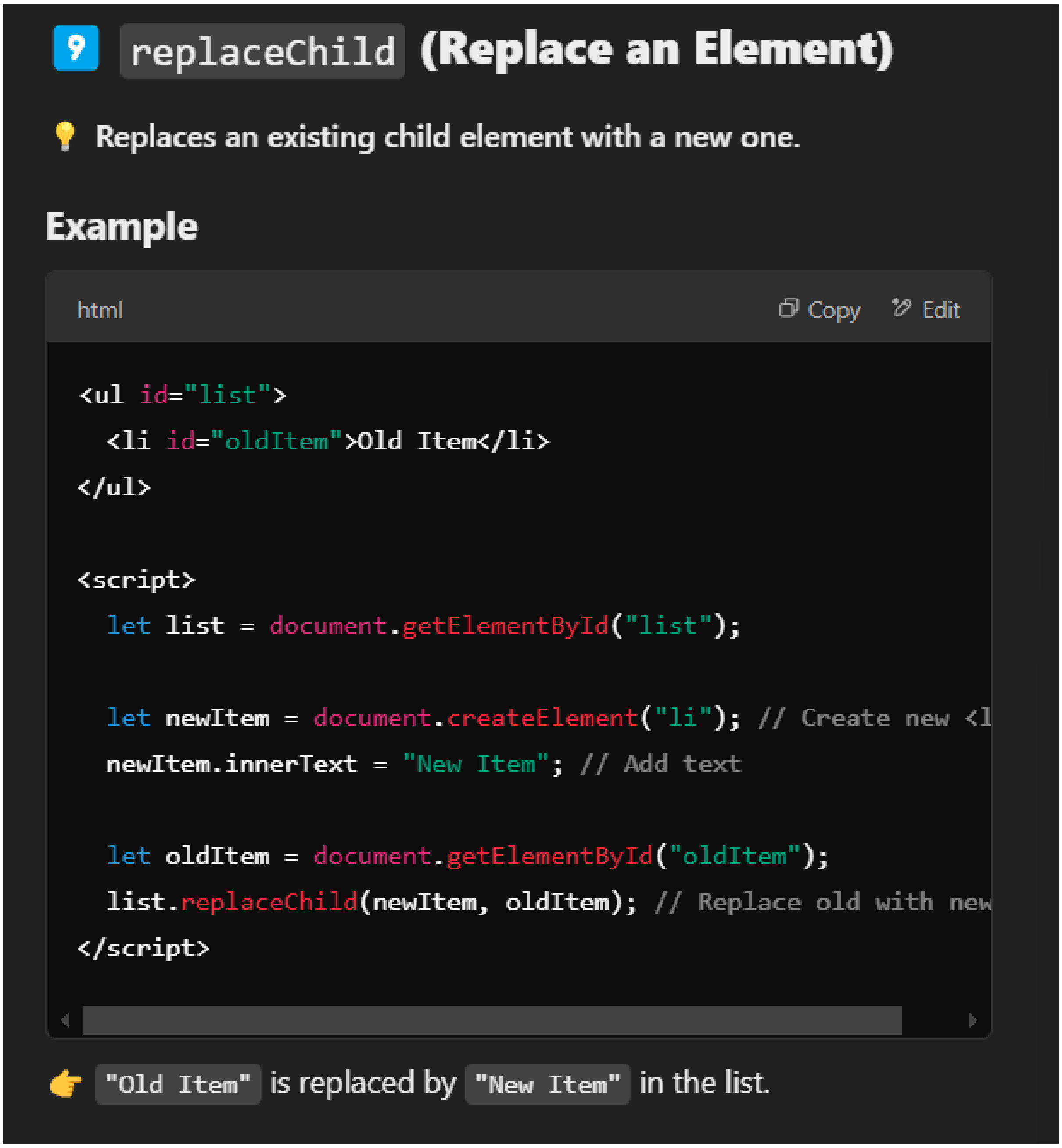
8️⃣ `removeChild` (Remove an Element)

💡 Deletes a child element from its parent.

Example

```html
<ul id="list">
  <li id="item1">Item 1</li>
  <li>Item 2</li>
</ul>

<script>
  let list = document.getElementById("list");
  let item = document.getElementById("item1");
  list.removeChild(item); // Removes "Item 1"
</script>
```

👉 The first `<li>` is removed from the list.



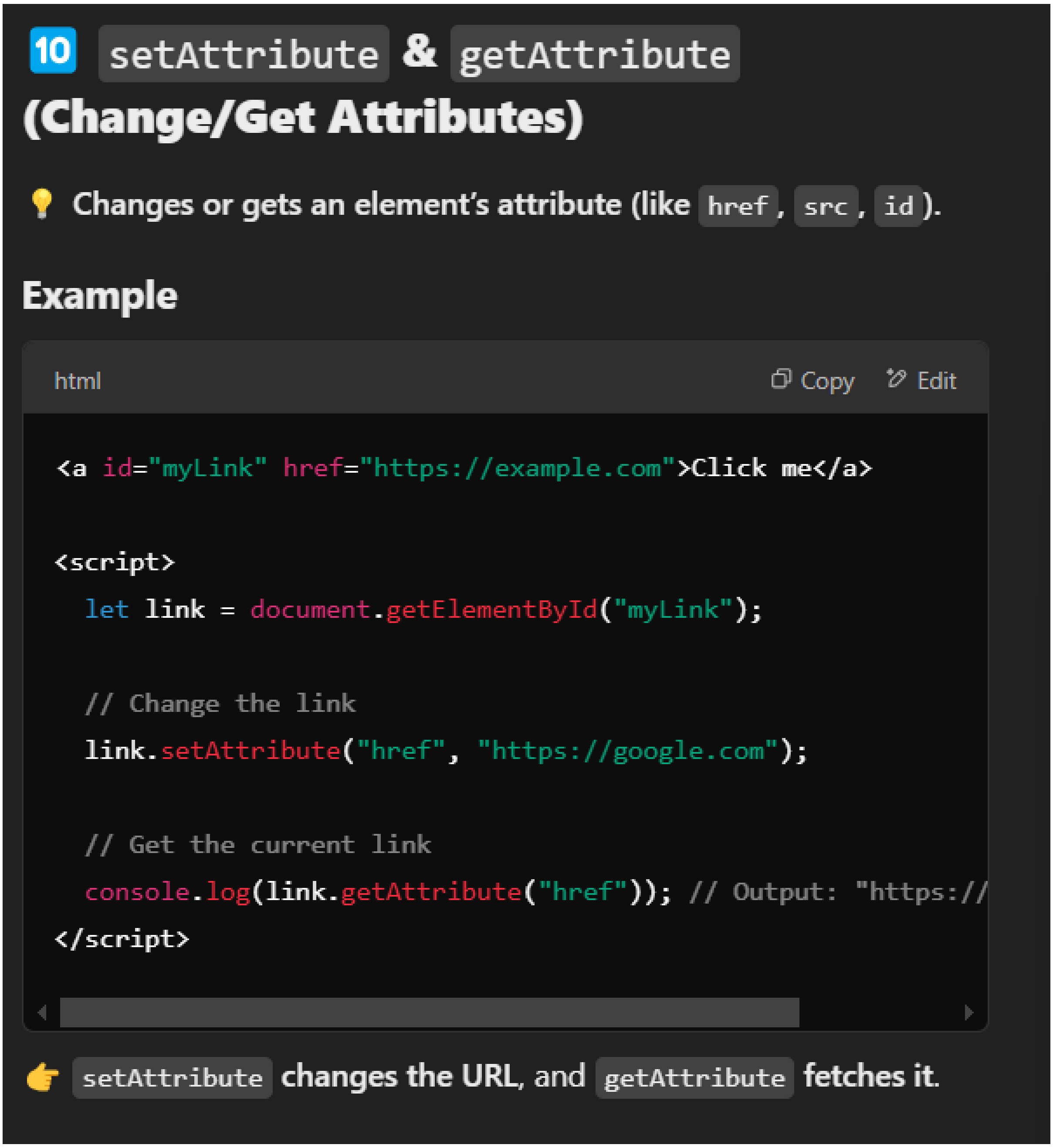
9️⃣ `replaceChild` (Replace an Element)

💡 Replaces an existing child element with a new one.

Example

```html
<ul id="list">
  <li id="oldItem">Old Item</li>
</ul>

<script>
  let list = document.getElementById("list");

  let newItem = document.createElement("li"); // Create new <l
  newItem.innerText = "New Item"; // Add text

  let oldItem = document.getElementById("oldItem");
  list.replaceChild(newItem, oldItem); // Replace old with new
</script>
```

👉 `"Old Item"` is replaced by `"New Item"` in the list.



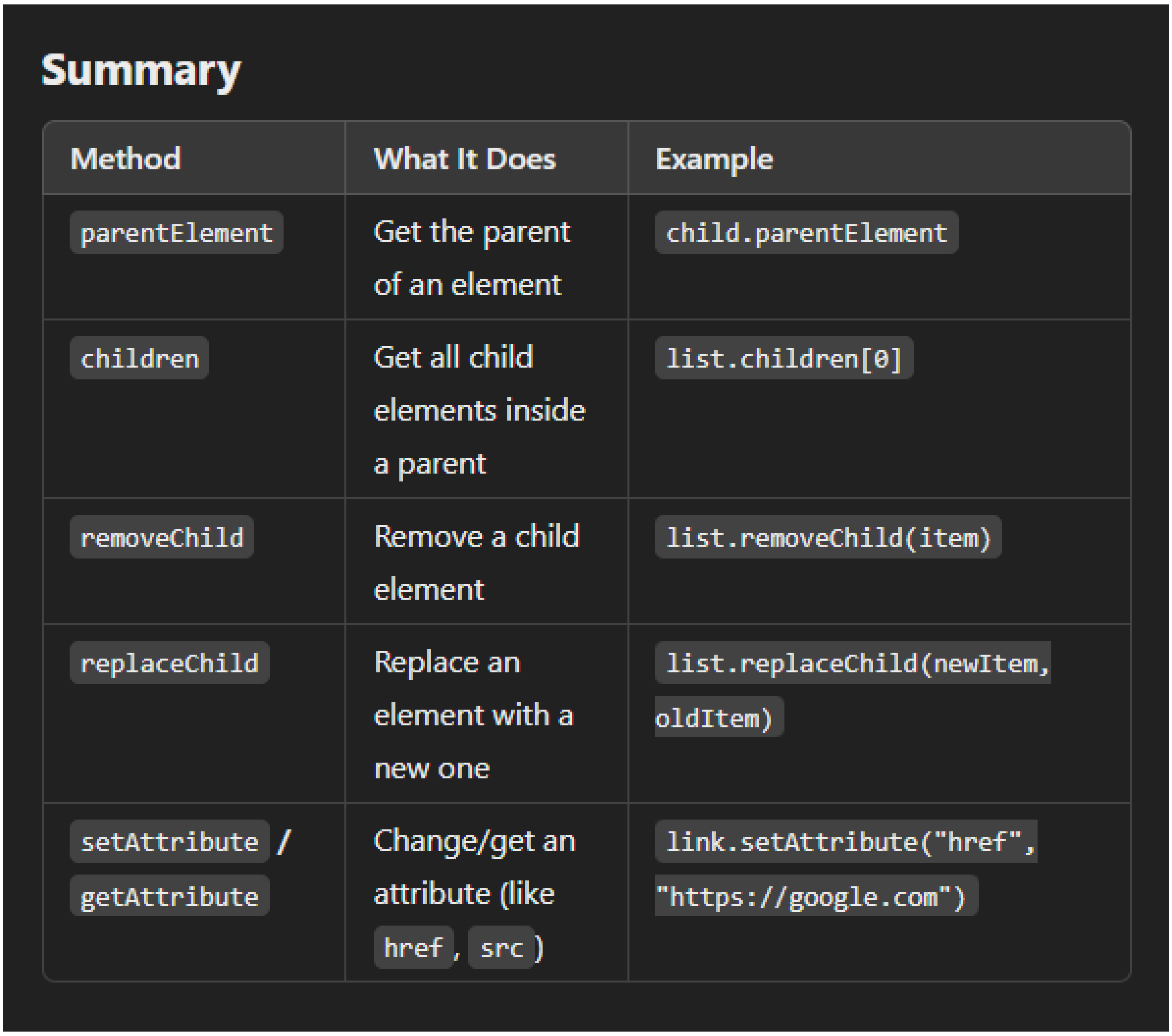
🔟 `setAttribute` & `getAttribute` (Change/Get Attributes)

💡 Changes or gets an element's attribute (like `href`, `src`, `id`).

Example

```html
<a id="myLink" href="https://example.com">Click me</a>

<script>
  let link = document.getElementById("myLink");

  // Change the link
  link.setAttribute("href", "https://google.com");

  // Get the current link
  console.log(link.getAttribute("href")); // Output: "https://
</script>
```

👉 `setAttribute` changes the URL, and `getAttribute` fetches it.



Summary

Method	What It Does	Example
`parentElement`	Get the parent of an element	`child.parentElement`
`children`	Get all child elements inside a parent	`list.children[0]`
`removeChild`	Remove a child element	`list.removeChild(item)`
`replaceChild`	Replace an element with a new one	`list.replaceChild(newItem, oldItem)`
`setAttribute` / `getAttribute`	Change/get an attribute (like `href`, `src`)	`link.setAttribute("href", "https://google.com")`