parent element

Traversing element

1. What is a Node?

In the DOM (Document Object Model), everything is a node:
The entire document → a document node
Elements like or → element nodes
Text inside an element → text nodes
Comments → comment nodes
Note : So, a node is a generic term that can be any type of object inside the DOM tree.

An element is a specific type of node that represents an HTML tag, like
 ...
 ...
So:

All elements are nodes
But not all nodes are elements (because text and comments are also nodes).

ParentNode vs ParentElement

When you want to move upward in the DOM tree:

parentNode

→ Returns the parent node (can be an element, document, or document fragment).

parentElement

→ Returns the parent only if it is an element node.
→ If the parent is not an element (like document), it returns null.

```
<div id="main">
        test
        <p class="note">This is a note!</p>
    </div>


       let cnode = document.getElementsByClassName('note');
       let pnode = cnode[0].parentNode;
       pnode.style.backgroundColor = "red";
        let cnodeq = document.querySelector('.note');
        let cnodeqparent = cnodeq.parentNode;
```

1. To get the parent node of a specified node in the DOM tree, you use the parentNode property:
let parent = node.parentNode;

The parentNode is read-only
The Document and DocumentFragment nodes do not have a parent. Therefore, the parentNode will always be null.

If you create a new node but haven’t attached it to the DOM tree, the parentNode of that node will also be null

sibling

Types of Sibling Properties in JavaScript :

Visual Tree (DOM)

```
<ul id="list">
  ├── #text (newline/space)
  ├── <li id="first">Apple</li>
  ├── #text (newline/space)
  ├── <li id="second">Banana</li>   <-- our focus
  ├── #text (newline/space)
  ├── <li id="third">Cherry</li>
  ├── #text (newline/space)
</ul>
```

That's why nextSibling / previousSibling often return #text (whitespace).
To safely get elements, prefer nextElementSibling / previousElementSibling.

1.nextSibling
Returns the next node (could be an element, text node, or comment).
2.previousSibling
Returns the previous node (could also be element, text node, or comment).
3.nextElementSibling
Returns the next sibling element (skips text and comments).
4.previousElementSibling
Returns the previous sibling element.

The nextElementSibling returns the next sibling of an element or null if the element is the last one in the list.
The previousElementSibling returns the previous sibling of an element or null if the element is the first one in the list.
To get all siblings of an element, you can use a helper function that utilizes the nextElementSibling property.

Children

Children

What are "children" in JavaScript DOM?

Child nodes = All nodes inside an element (including text nodes, comments, and element nodes).
Child elements = Only element nodes inside a parent (ignores text and comments).
 Properties / methods to use:
children → returns HTMLCollection (only element nodes).
childNodes → returns NodeList (includes text nodes, comments, etc).
firstElementChild → first child element.
lastElementChild → last child element.
hasChildNodes() → check if any child nodes exist.

```
<div id="container">
        <p>Paragraph 1</p>
        <p>Paragraph 2</p>
        <span>Span inside container</span>
    </div>

        let parent = document.getElementById('container');
        let childrens = parent.children;
        console.log(childrens);
        console.log(parent.firstChild);
        console.log(parent.firstElementChild);
        console.log(parent.lastChild);
        console.log(parent.lastElementChild);
```

The firstChild and lastChild return the first and last child of a node, which can be any node type including text node, comment node, and element node.
The firstElementChild and lastElementChild return the first and last child Element node.
The childNodes returns a live NodeList of all child nodes of any node type of a specified node. The children return all child Element nodes of a specified node.