Chapter 7 - Walking the DOM

DOM tree refers to the HTML page where all the nodes are objects. There can be 3 main types of nodes in the DOM tree :

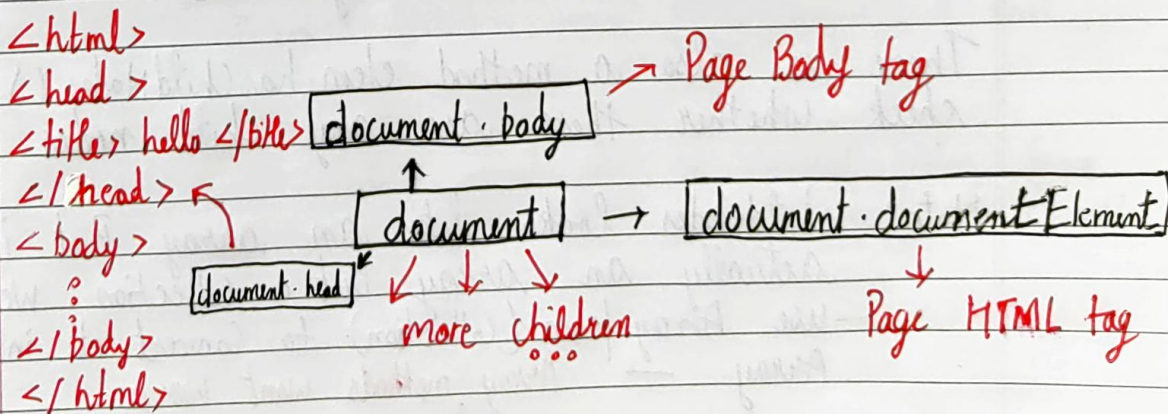1> text nodes
2> element nodes
3> comment nodes

In an HTML page, <html> is at the root and <head> and <body> are its children, etc.

A text node is always a leaf of the tree

Auto Correction
If an erroneous HTML is encountered by the browser, it tends to correct it. for example, if we put something after the body, it is automatically moved inside the body. Another example is <table> tag which must contain <tbody>

Walking the DOM

```
<html>
<head>
<title> hello </title>  document.body  ──→ Page Body tag
</head>
<body>
    ?
</body>
</html>
```

document.head → document → document.documentElement

↓ ↓ ↓
more children ...

Page HTML tag

Note : document.body can sometimes be null if the javascript is written before the body tag.

Children of an element

Direct as well as deeply nested elements of an element are called its children

Child nodes → Elements that are direct children
for example head & body are children of < html >

Descendant nodes → All nested elements, children, their children and so on ...

first Child, last Child & child Nodes

element. first Child → first child element
element. last Child → last child element
element. child Nodes → All Child nodes

Following is always true :

elem. child Nodes [0] === elem. first Child
elem. childNodes [elem. child Nodes. length − 1] === elem. last Child

There is also a method elem. hasChild Nodes () to check whether there are any child nodes.

Note: child Nodes looks like an array. But its not actually an array but a collection. We Can use Array. from (collection) to Convert it into an Array. → Array methods wont work

Notes on DoM collections
→ They are read-only
→ They are live. elem.childNodes variable (reference) will automatically update if childNodes of elem is changed.
→ They are iterable using for...of loop

Siblings and the parent
Siblings are nodes that are children of the same parent.

→ for example: <head> and <body> are siblings. Siblings have same parent. In the above example its html

→ <body> is said to be the "next" or "right" sibling of <head>, <head> is said to be the "previous" ore "left" sibling of <body>

→ The next sibling is in nextSibling property, and the previous one in previousSibling.
The parent is available as parentNode.

alert (document. documentElement. parentNode); //document
alert (document. documentElement. parentElement); // null

Element only Navigation
Sometimes, we dont want text or Comment nodes. Some links only take Element nodes into account. For example

document. previousElement Sibling → Previous sibling which is an Element

document · next Element Sibling → next sibling (Element)

document · first Element Child → first Element child

document · last Element Child → last Element child.

Table links

Certain DOM elements may provide additional properties specific to their type for convenience. Table element supports the following properties :

table · rows → Collection of tr elements
table · Caption → reference to < Caption>
table · thead → reference to < thead>
table · tfoot → reference to <tfoot>
table · tBodies → Collection of <tbody> elements
tbody · rows → Collection of <tr> inside

tr · cells → Collection of td and th
tr · Section Row Index → Index of tr inside enclosing element
tr · row Index → Row number starting from 0

td · cellIndex → no of cells inside enclosing <tr>

Quick Quiz : Print typeof document and typeof window in the Console & See what it prints

Searching the DoM

DoM navigation properties are helpful when the elements are close to each other. If they are not close to each other, we have some more methods to search the DOM

→ document.getElementById
This method is used to get the element with a given "id" attribute

```
let span = document.getElementById('span')
span.style.color = "red"
```

→ document.querySelectorAll
Returns all elements inside an element matching the given CSS selector

→ document.querySelector
Returns the first element for the given CSS Selector.
A efficient version of elem.querySelectorAll(css)[0]

→ document.getElementsByTagName
Returns elements with the given tag name

→ document.getElementsByClassName
Returns elements that have the given CSS class

    ↳ Dont forget the "s" letter

→ document.getElementsByName
Searches elements by the name attribute.

matches, closest & contains methods

There are three important methods to search the DOM

1> elem.matches (css) → To check if element matches the given CSS selector

2> elem.closest (css) → To look for the nearest ancestor that matches the given CSS- selector. The elem itself is also checked

3> elemA.contains (elemB) → Returns true if elemB is inside elemA (a descendant of elemA) or when elemA == elemB