Document object model (DOM)

·) Data representation of the objects that comprise the structure and content of a document on the web.

·) DOM: programming interface for web documents.

↓

represents the document as nodes and objects, allowing programming languages to interact with it.

HTML: markup language → depending on the different tags, represented accordingly on the page.

↓

These markup documents can be imagined in a tree structure, with each tag sort of represented by a node in the tree having its own value and children.

↓

This tree can be interacted with as a whole with special methods (interface) to get the content of nodes, to modify nodes, add nodes, delete nodes, etc.

·) DOM representation allows for the document to be manipulated. It's basically the object oriented representation of the document.

```
const paragraphs = document.querySelectorAll('p');
```

·) DOM & Javascript

→ without DOM, Javascript won't have any notion of web pages.

→ Document as a whole, the head, the tables within the document, table headers, text within table cells, and all other elements in a document are parts of the DOM for that document.

→ DOM is part of the web api, meant to be independent of any language, making the structural representation of the document available from a single, consistent api.

↓

Allowing other languages to also be used to interact with the web api & consequently with the DOM of the document, namely python.

→ The document object is used to interact with DOM in a script running on a browser, and along with window object are available by default.

→ example :

```
const heading = document.createElement("h1");
const heading_text = document.createTextNode("Big Head!");
heading.appendChild(heading_text);
document.body.appendChild(heading);
```

*) Fundamental Data Types

→ strictly speaking, not every node is an element.

i) Document : object is the root document itself.

ii) Node : every object located within a document is a node of some kind, can be an element node, or a text node or attribute node in an HTML document.

iii) Element: implements DOM Element interface, and the Node interface. further enhanced by HTML Dom Api's HTMLElement interface in a HTML document, along with other interfaces describing capabilities of other kinds.

iv) NodeList: return type for methods that return a list of nodes (majorly elements)

v) Attr: special reference that expores a special interface for attributes. rarely used.

vi) NamedNodeMap: collection of 'Attr' objects. Return type of

Element.attributes

```
const para = document.getElementsByTagName("p")[0];
```

→ Interfaces and objects

```
const table = document.getElementById("table");
const tableAttrs = table.attributes; // node/element interface
for (let i=0; i<tableAttrs.length; i++) {
    if (tableAttrs[i].nodeName.toLowerCase() == "border")
        table.border = "1";
            ↳ // HTMLTableElement interface
}
table.summary = "note: increased border";
```

→ Core Interfaces in the DOM

i) document.querySelector(selector)

ii) document.querySelectorAll(selector)

iii) document.createElement(name)

iv) parentNode.appendChild(node)

v) element.innerHTML()

vi) element.style.left

vii) element.setAttribute()

viii) element.getAttribute()

ix) element.addEventListener()

x) window.content

xi) GlobalEventHandlers → onload

xii) window.scroll(TOC)