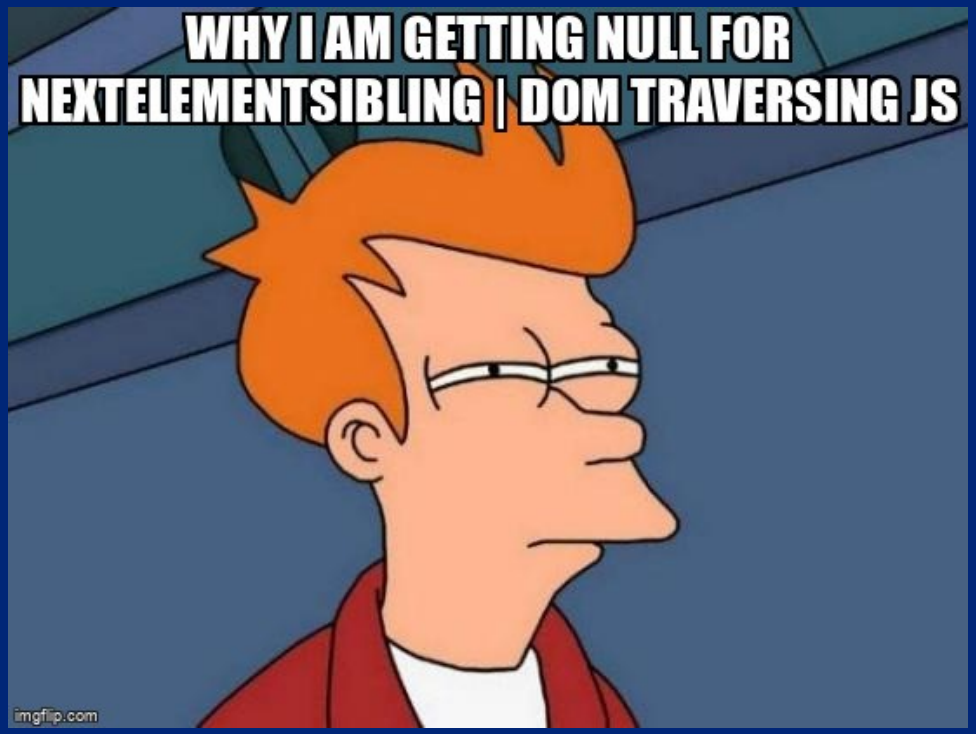
Università di Genova

JavaScript (3)



Marina Ribaudo, marina.ribaudo@unige.it

Document Object Model (DOM)

The **Document Object Model** (**DOM**) connects web pages to scripts or programming languages by representing the structure of a document—such as the HTML representing a web page—in memory. Usually it refers to JavaScript, even though modeling HTML, SVG, or XML documents as objects are not part of the core JavaScript language.

The DOM represents a document with a logical tree. Each branch of the tree ends in a node, and each node contains objects. DOM methods allow programmatic access to the tree. With them, you can change the document's structure, style, or content.

Nodes can also have event handlers attached to them. Once an event is triggered, the event handlers get executed.

To learn more about what the DOM is and how it represents documents, see our article Introduction to the DOM.

DOM interfaces

https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model

Document Object Model

- Per **accedere agli elementi** del documento è possibile **selezionarli** usando i metodi offerti dal DOM
  - let el1 = document.getElement**ById**(“id”);
  - let el2 = document.getElement**sByName**(“error”);
  - let el3 = document.getElement**sByTagName**(“span”);
  - let el4 = document.getElement**sByClassName**(“warning”);
  - let el5 = document.**querySelector**(“.warning”);

- Esiste anche document.**all**
  - let el6 = document.all

Document Object Model

- Il **tipo Node** definisce proprietà e metodi che permettono di attraversare l'albero del DOM
- Ogni nodo ha le seguenti **proprietà**
  - nodeType (intero da 1 a 12)
    - Document = 9
    - Element = 1
    - Attribute = 2
    - Text = 3
    - Comments = 8
  - nodeName (il nome del nodo)
  - nodeValue (il contenuto di un nodo)

Document Object Model

- Si può navigare l’albero del DOM usando
  - parentNode (nodo padre del nodo corrente)
  - childNodes (lista di nodi figli, nell'ordine in cui appaiono nel file HTML)
  - firstChild, lastChild, nextSibling, previousSibling

Document Object Model

- Molti elementi HTML hanno degli **attributi**
- Il DOM definisce i metodi per **accedere/modificare i valori** degli attributi
- **object.getAttribute**(“attribute”)
  - si accede all'elemento con uno dei metodi precedenti
  - si ricavano i valori dei suoi attributi

```
let el = document.getElementById(“myimg”);
console.log(el.getAttribute(“alt”));
console.log(el.getAttribute(“src”));
```

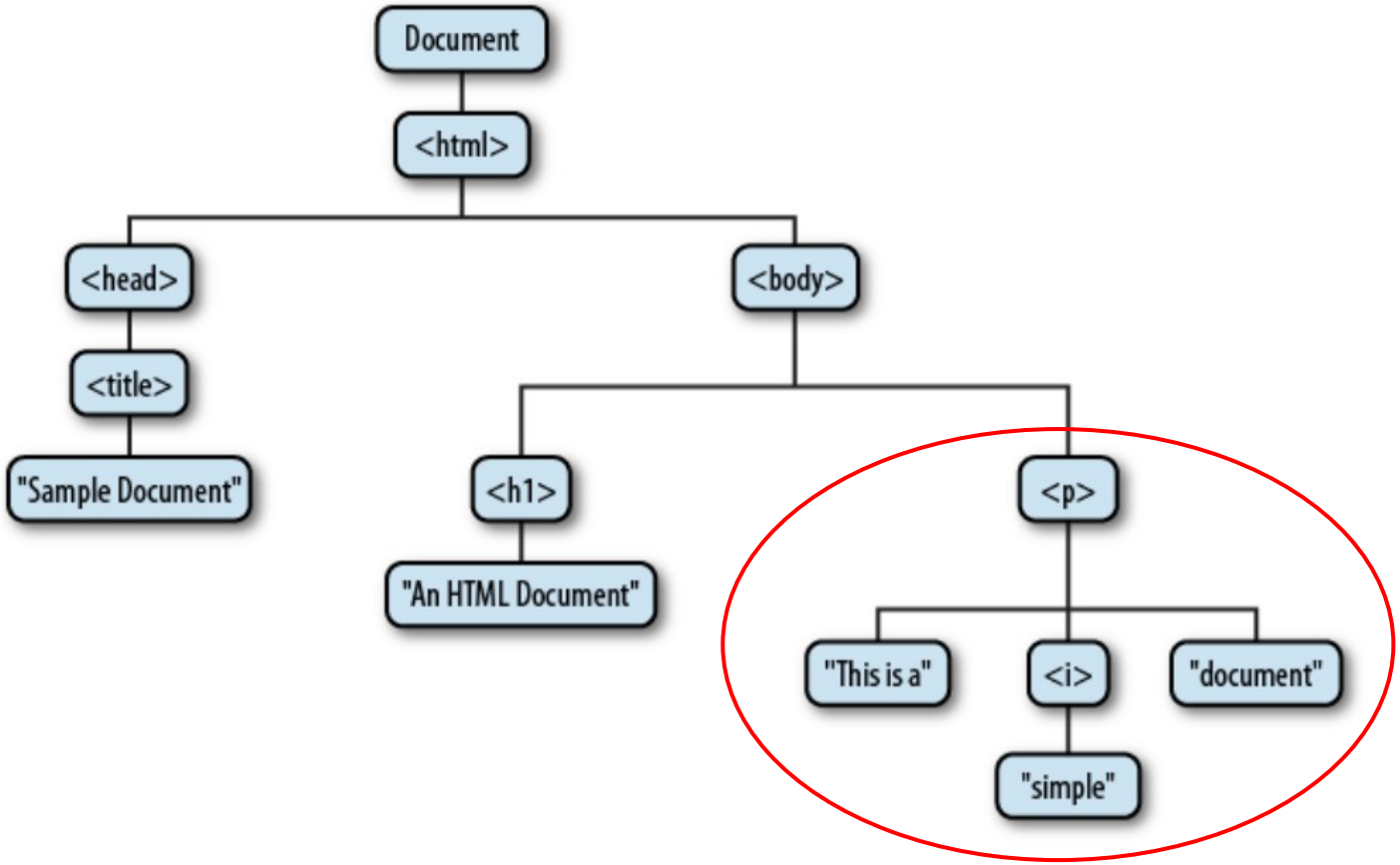
Document Object Model

- **object.setAttribute**(“attribute”,”value”)
  - si accede all'elemento e si modificano i valori dei suoi attributi

- La modifica dei valori degli attributi **non si vede** nel sorgente HTML: agendo sul DOM si modifica il contenuto della pagina in memoria, senza ricaricarla

Document Object Model

- **Contenuto** degli elementi, per es. <p>



Document Object Model

- **Contenuto** degli elementi, per es. <p>
  - La proprietà **innerHTML** ritorna il contenuto di un elemento come stringa con markup
  - La proprietà **innerText** ritorna il contenuto di un elemento come stringa di testo
- Sono **proprietà scrivibili** e permettono di cambiare porzioni della pagina in risposta agli eventi

Document Object Model

- È anche possibile modificare il contenuto di un documento agendo a livello di nodi grazie a metodi quali:
  - createElement()
  - createTextNode()
  - cloneNode()
  - ecc.