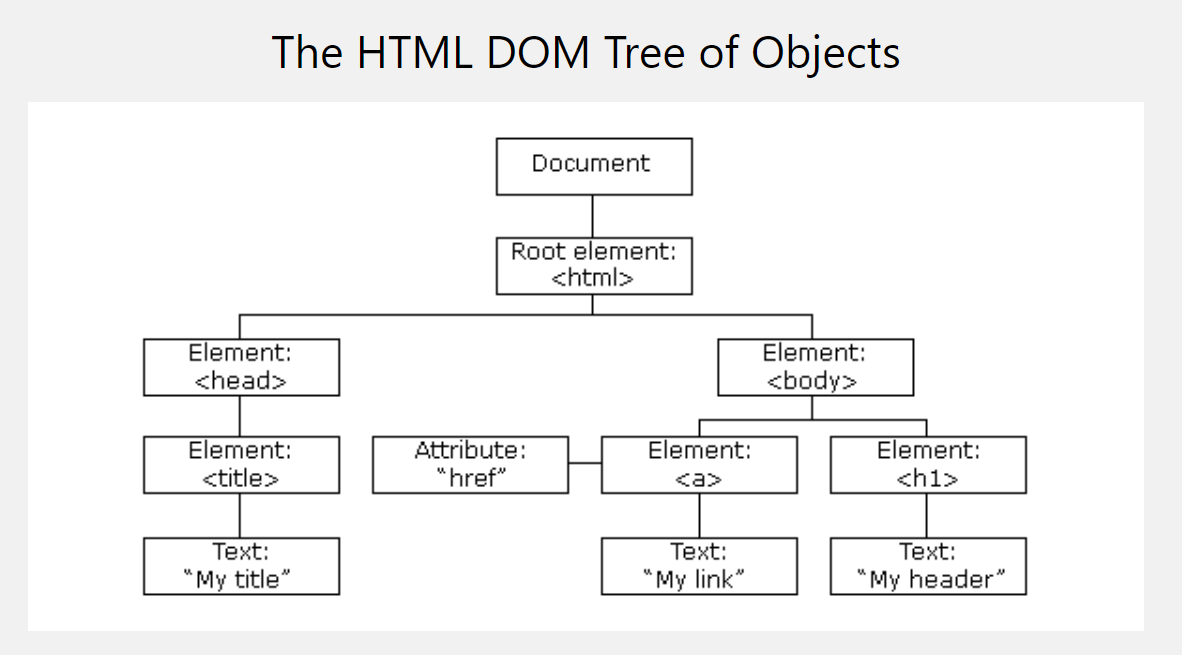
1. DOM

DOM là gì?

**DOM** là tên gọi viết tắt của (**D**ocument **O**bject **M**odel – tạm dịch Mô hình Các Đối tượng Tài liệu). Là một chuẩn được định nghĩa bởi W3C (Tổ Chức Web Toàn Cầu – World Wide Web Consortium). DOM được dùng để truy xuất và thao tác trên các tài liệu có cấu trúc dạng HTML hay XML bằng các ngôn ngữ lập trình thông dụng như Javascript, PHP…



Trong mỗi thẻ HTML sẽ có những thuộc tính (Properties) và có phân cấp cha – con với các thẻ HTML khác. Sự phân cấp và các thuộc tính của thẻ HTML này ta gọi là **selector** và trong DOM sẽ có nhiệm vụ xử lý các vấn đề như đổi thuộc tính của thẻ, đổi cấu trúc HTML của thẻ.

Có thể thấy tất cả các thẻ HTML sẽ được quản lý trong đối tượng **document.** Thẻ cao nhất là thẻ **html**, tiếp theo là phân nhánh **body**và **head**. Bên trong **head**thì có những thẻ như **style**, **title**,… và bên trong **body**thì là vô số các thẻ HTML khác. Như vậy trong **Javascript**, để thao tác với các thẻ HTML ta phải thông qua đối tượng **document.**

## HTML DOM là gì?

HTML DOM giúp thao tác dữ liệu theo mô hình hướng đối tượng. Các phần tử bên trong 1 tài liệu có cấu trúc được định nghĩa thành các *đối tượng*, *phương thức* và *thuộc tính* để có thể truy xuất dễ dàng mà vẫn đảm bảo tính cấu trúc.

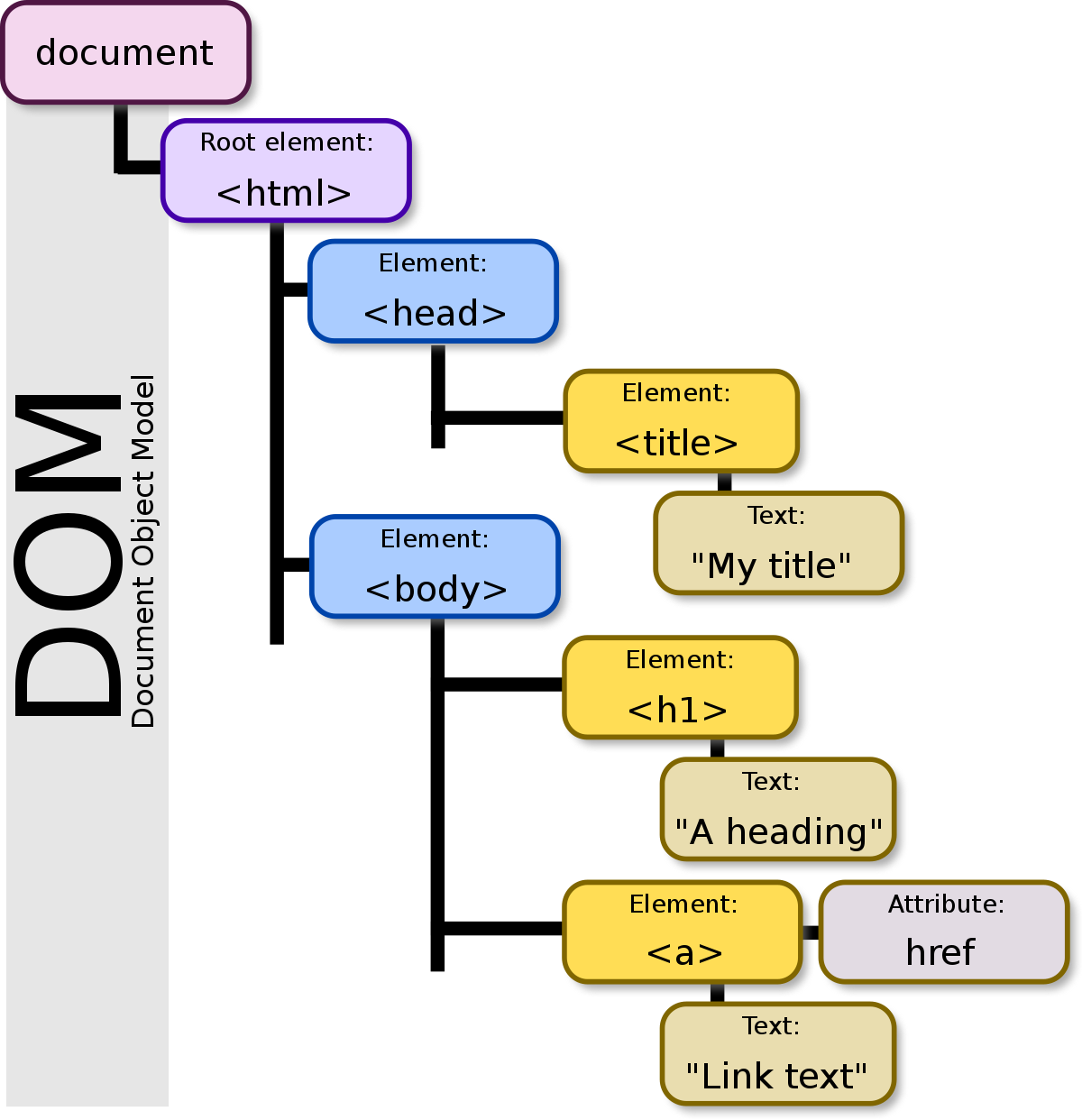
Mỗi phần tử là một đối tượng, sở hữu các thuộc tính và các phương thức để làm việc với các thuộc tính đó như thêm, xóa, sửa, cập nhật. Bên cạnh đó, bạn cũng có thể thêm bớt các phần tử tùy thích, giúp cho nội dung và cấu trúc của trang web luôn cập nhật động. Nó định nghĩa sau đây:

* HTML elements như là objects
* properties của tất cả HTML elements
* methods để truy cập đến tất cả HTML elements
* events cho tất cả HTML elements

### **Node**

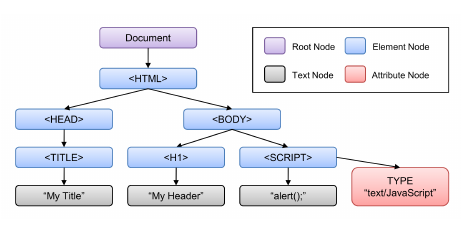
Đối với HTML DOM, mọi thành phần đều được xem là một node (nút), được biểu diễn trên 1 cây cấu trúc dạng cây gọi là DOM Tree. Các phần tử khác nhau sẽ được phân loại node khác nhau nhưng quan trọng nhất là 3 loại: node gốc (document node), node phần tử (element node), node văn bản (text node).

* **Node gốc**: chính là tài liệu HTML, thường được biểu diễn bởi thẻ <html>.
* **Node phần tử**: biểu diễn cho 1 phần tử HTML.
* **Node văn bản**: mỗi đoạn kí tự trong tài liệu HTML, bên trong 1 thẻ HTML đều là 1 node văn bản. Đó có thể là tên trang web trong thẻ <title>, tên đề mục trong thẻ <h1>, hay một đoạn văn trong thẻ <p>.
* Ngoài ra còn có **node thuộc tính** (attribute node) và **node chú thích** (comment node).



### **Quan hệ giữa các node**

* Node gốc (document) luôn là node đầu tiên.
* Tất cả các node không phải là node gốc đều chỉ có 1 node cha (parent).
* Một node có thể có một hoặc nhiều con, nhưng cũng có thể không có con nào.
* Những node có cùng node cha được gọi là các node anh em (siblings).
* Trong các node anh em, node đầu tiên được gọi là con cả (firstChild) và node cuối cùng là con út (lastChild).



## Các loại DOM trong Javascript

Javascript cung cấp cho chúng ta nhiều loại DOM để xử lí HTML và CSS dễ dàng hơn.

* DOM document: lưu trữ toàn bộ các thành phần trong documents của website.
* DOM element: truy xuất tới thẻ HTML nào đó thông qua các thuộc tính như tên class, id, name của thẻ HTML.
* DOM HTML: thay đổi giá trị nội dung và giá trị thuộc tính của các thẻ HTML.
* DOM CSS: thay đổi các định dạng CSS của thẻ HTML.
* DOM Event: gán các sự kiện như onclick(), onload() vào các thẻ HTML.
* DOM Listener: lắng nghe các sự kiện tác động lên thẻ HTML.
* DOM Navigation dùng để quản lý, thao tác với các thẻ HTML, thể hiện mối quan hệ cha – con của các thẻ HTML
* DOM Node, Nodelist: thao tác với HTML thông qua đối tượng (Object).

## Thao tác với DOM

Mọi nội dung đều có thể được cập nhật động thông qua các thuộc tính và phương thức của DOM. Từ thay đổi định dạng chữ, nội dung chữ đến thay đổi cấu trúc các node và cả thêm node mới. Bạn cần hiểu rõ cách thao tác DOM và ý nghĩa của từng thuộc tính, phương thức.

### **Các Thuộc tính và Phương thức thường gặp**

#### **Thuộc tính:**

* **id:** Định danh – là duy nhất cho mỗi phần tử nên thường được dùng để truy xuất DOM trực tiếp và nhanh chóng.
* **className:** Tên lớp – Cũng dùng để truy xuất trực tiếp như id, nhưng 1 className có thể dùng cho nhiều phần tử.
* **tagName:** Tên thẻ HTML.
* **innerHTML:** Trả về mã HTML bên trong phần tử hiện tại. Đoạn mã HTML này là chuỗi kí tự chứa tất cả phần tử bên trong, bao gồm các node phần tử và node văn bản.
* **outerHTML:** Trả về mã HTML của phần tử hiện tại. Nói cách khác, outerHTML = tagName + innerHTML.
* **textContent:** Trả về 1 chuỗi kí tự chứa nội dung của tất cả node văn bản bên trong phần tử hiện tại.
* **attributes:** Tập các thuộc tính như id, name, class, href, title…
* **style:** Tập các định dạng của phần tử hiện tại
* **value:** Lấy giá trị của thành phần được chọn thành một biến.

#### **Phương thức:**

* **getElementById(id):** Tham chiếu đến 1 node duy nhất có thuộc tính id giống với id cần tìm.
* **getElementsByTagName(tagname):** Tham chiếu đến tất cả các node có thuộc tính tagName giống với tên thẻ cần tìm, hay hiểu đơn giản hơn là tìm tất cả các phần tử DOM mang thẻ HTML cùng loại. Nếu muốn truy xuất đến toàn bộ thẻ trong tài liệu HTML thì hãy sử dụng document.getElementsByTagName('\*').
* **getElementsByName(name):** Tham chiếu đến tất cả các node có thuộc tính name cần tìm.
* **getElementByClassName(name):** Tham chiếu đến tất cả các node với className = name
* **getAttribute(attributeName):** Lấy giá trị của thuộc tính.
* **setAttribute(attributeName, value):** Sửa giá trị của thuộc tính.
* **appendChild(node):** Thêm 1 node con vào node hiện tại.
* **removeChild(node):** Xóa 1 node con khỏi node hiện tại.

Mặt khác, các phần tử DOM đều là các node trên cây cấu trúc DOM. Chúng sở hữu thêm các thuộc tính quan hệ để biểu diễn sự phụ thuộc giữa các node với nhau. Nhờ các thuộc tính quan hệ này, chúng ta có thể truy xuất DOM gián tiếp dựa trên quan hệ và vị trí của các phần tử:

#### **Thuộc tính quan hệ:**

* **parentNode:** node cha
* **childNodes:** Các node con
* **firstChild:** node con đầu tiên
* **lastChild:** node con cuối cùng
* **nextSibling:** node anh em liền kề sau
* **previousSibling:** node anh em liền kề trước

Bạn có thể xem danh sách đầy đủ ở [W3SCHOOLS](https://www.w3schools.com/jsref/dom_obj_all.asp)

### **Truy xuất các phần tử trong mô hình DOM**

Chúng ta có thể truy xuất mọi phần tử html bằng cách sử dụng các phương thức :

* document.getElementById()
* document.getElementsByTagName()
* document.getElementsByClass()

Tài liệu tham khảo: <https://codelearn.io/sharing/thao-tac-voi-dom-bang-javascript>

2. Event.

 Sự kiện hay còn lại là event, là một tác động nào đó lên các đối tượng HTML thông qua cấu trúc DOM.

| **STT** | **Event Name** | **Description** |
| --- | --- | --- |
| 1 | onclick | Xảy ra khi click vào thẻ HTML |
| 2 | ondbclick | Xảy ra khi double click vào thẻ HTML |
| 3 | onchange | Xảy ra khi giá trị (value) của thẻ HTML đổi. Thường dùng trong các đối thẻ form input, select |
| 4 | onmouseover | Xảy ra khi con trỏ chuột bắt đầu đi vào thẻ HTML |
| 5 | onmouseout | Xảy ra khi con trỏ chuột bắt đầu rời khỏi thẻ HTML |
| 6 | onmouseenter | Tương tự như onmouseover |
| 7 | onmouseleave | Tương tự như onmouseout |
| 8 | onmousemove | Xảy ra khi con chuột di chuyển bên trong thẻ HTML |
| 9 | onkeydown | Xảy ra khi gõ một phím bất kì vào ô input |
| 10 | onload | Sảy ra khi thẻ HTML bắt đầu chạy, nó giống như hàm khởi tạo trong lập trình hướng đối tượng vậy đó. |
| 11 | onkeyup | Xảy ra khi bạn gõ phím nhưng lúc bạn nhả phím ra sẽ được kích hoạt |
| 12 | onkeypress | Xảy ra khi bạn nhấn môt phím vào ô input |
| 14 | onblur | Xảy ra khi con trỏ chuột rời khỏi ô input |
| 15 | oncopy | Xảy ra khi bạn copy nội dung của thẻ |
| 16 | oncut | Xảy ra khi bạn cắt nội dung của thẻ |
| 17 | onpaste | Xảy ra khi bạn dán nội dung vào thẻ |

Tài liệu tham khảo: https://freetuts.net/su-kien-event-trong-javascript-368.html