TechMaster
Học là có việc

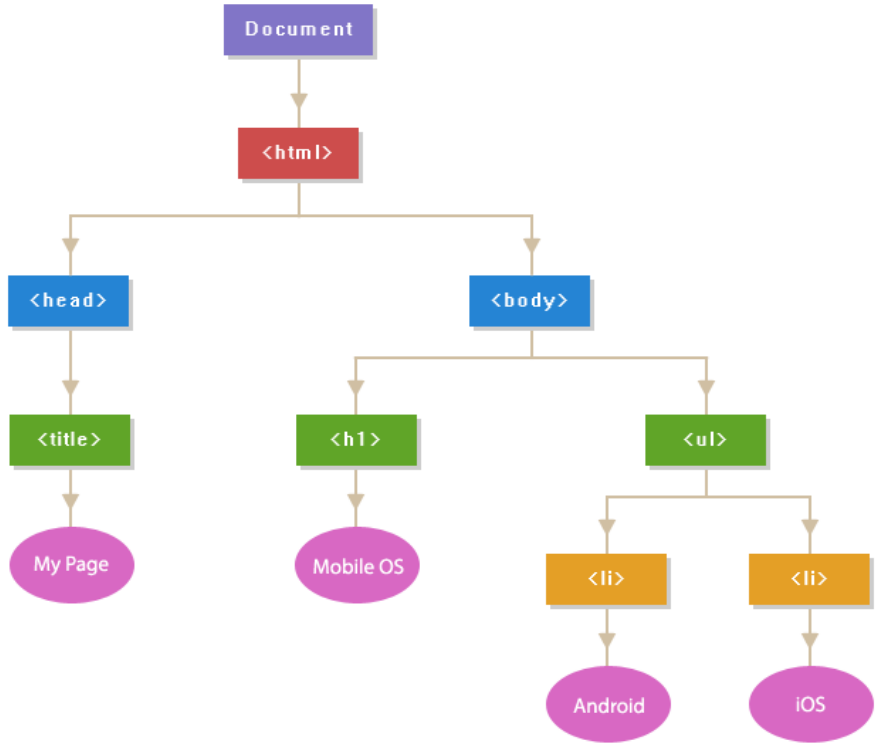
Javascipt DOM

Bùi Văn Hiên
hien@techmaster.vn

Giới thiệu về DOM

Giới thiệu về DOM

- **Document Object Model (DOM)** được tạo ra bởi trình duyệt khi trang web được tải
- DOM được tổ chức theo dạng cây (DOM tree), mỗi thành phần trên cây gọi là một node



Giới thiệu về DOM

Với Javascript, chúng ta có thể thao tác với DOM để:

- Thay đổi phần tử HTML
- Thay đổi thuộc tính của phần tử HTML
- Thay đổi style CSS
- Xóa phần tử và thuộc tính hiện có
- Tạo phần tử và thuộc tính mới
- Phản ứng với các sự kiện
- Tạo sự kiện mới
- …

Truy cập vào các phần tử

Truy cập vào các phần tử trong DOM

Chúng ta có các phương thức sau để có thể truy cập vào các phần tử trong DOM

- document.getElementById: Truy cập thông qua ID
- document.getElementsByTagName: Truy cập thông qua Tag , trả về 1 mảng các phần tử
- document.getElementsByClassName: Truy cập thông qua tên Class , trả về 1 mảng các phần tử
- document.querySelector: Truy cập thông qua CSS Selector, trả về phần tử đầu tiên tìm thấy
- document.querySelectorAll: Truy cập thông qua CSS Selector, trả về 1 mảng các phần tử

Quan hệ giữa các phần tử

Thuộc tính quan hệ:

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

Truy cập gián tiếp

Bởi vì các node trong DOM tree có mối quan hệ với nhau, thay vì truy cập trực tiếp chúng ta có thể truy cập gián tiếp thông qua mối quan hệ này.

- Node.parentNode: tham chiếu đến nút cha của nút hiện tại
- Node.childNodes: tham chiếu đến các nút con trực tiếp của nút hiện tại
- Node.firstChild: tham chiếu đến nút con đầu tiên của nút hiện tại
- Node.lastChild: tham chiếu đến nút con cuối cùng của nút hiện tại
- Node.nextSibling: tham chiếu đến nút anh em nằm liền kề sau với nút hiện tại.
- Node.previousSibling: tham chiếu đến nút anh em nằm liền kề trước với nút hiện tại.

Get/Set nội dung của phần tử

Get/set nội dung phần tử DOM

Chúng ta có các phương thức sau để có thể get/set nội dung cho phần tử DOM

- innerHTML()
- innerText()
- textContent()

Thao tác với các phần tử DOM

Tạo phần tử

Sử dụng phương thức document.createElement([tag_name]) để tạo mới một phần tử HTML

Ví dụ: Tạo thẻ p và gán nội dung cho thẻ p vừa tạo

Thêm phần tử

Để thêm phần tử vào DOM chúng ta có các cách sau

(Khi sử dụng những phương thức này, cần tạo DOM element trước, sau đó mới thêm)

- parentNode.appendChild(newNode)
- parentNode.prepend(newNode)
- parentNode.insertBefore(newNode, referNode)

Ngoài ra còn 1 số cách sau:

- targetElement.insertAdjacentHTML(position, text);
- targetElement.insertAdjacentElement(position, element);
- targetElement.insertAdjacentText(position, text);

Xóa phần tử

Phương thức: parentNode.removeChild(childNode);

- Để xóa phần tử, cần phải biết phần tử cha

VD: Xóa phần tử có **id="child"** nằm trong phần tử có **id="parent"**

Thay thế phần tử

Phương thức: parentNode.replaceChild(newNode, oldNode);

- Để thay thế phần tử, cần phải biết phần tử cha và phần tử cần thay thế

VD: Thay thế phần tử có **id="child"** nằm trong phần tử có **id="parent"** bằng 1 thẻ **p** có nội dung **"New para"**

```
// Truy cập vào phần tử cha và con
const parent = document.getElementById('parent');
const child = document.getElementById('child');
// Tạo thẻ p mới và gán nội dung
const para = document.createElement('p');
para.innerText = 'New para';
// Thực hiện thay thế phần tử con
parent.replaceChild(para, child);
```

Sao chép phần tử

Phương thức: node.cloneNode(deep)

Trong đó:

- **node**: Phần tử được clone
- **deep**: Không bắt buộc
  - **true** - sao chép node, attributes và thành phần con của nó
  - **false**: - chỉ sao chép Node và attributes *(mặc định)*

Thay đổi CSS

Để thay đổi style của 1 phần tử ta sử dụng cú pháp sau

element.style.property = value

Chú ý: Khi **property** có **2 từ trở lên** cần viết theo dạng **camelCase**

Ví dụ: backgroundColor, marginLeft, paddingLeft,…

Attribute

Để thao tác với attribute của phần tử, chúng ta có các phương thức sau:

- *element*.getAttribute(*attributename*): Trả về giá trị của attribute của phần tử được chỉ định
- *element*.setAttribute(*attributename*, *attributevalue*): thêm attribute vào một phần tử và cung cấp cho giá trị cho nó
- *element*.hasAttribute(*attributename*): trả về **true** nếu attribute được chỉ định tồn tại, nếu không thì trả về **false**.
- *element*.removeAttribute(*attributename*) loại bỏ attribute được chỉ định khỏi một phần tử

Truy cập Class

Thuộc tính classList trong HTML DOM

ClassList là thuộc tính **read-only** của một phần tử trả về một tập hợp các class CSS và cho phép chúng ta sử dụng một số phương thức để quản lý và cập nhật các class đó.

Trong đó:

- add(): Thêm các class được chỉ định.
- remove(): Loại bỏ các class được chỉ định.
- contains(): Kiểm tra xem class được chỉ định có tồn tại trong phần tử hay không.
- toggle(): toggles class được chỉ định.

Events

Events

Events là những hành động của người dùng khi tương tác lên phần tử HTML.

Ví dụ:

- Khi người dùng Click chuột vào 1 phần tử
- Khi nhập dữ liệu vào ô input
- Khi submit form để gửi dữ liệu lên server

Ví dụ về một số loại event:

- Sự kiện chuột
- Sự kiện bàn phím
- Sự kiện form
- ...

(Tham khảo link sau: https://www.w3schools.com/jsref/dom_obj_event.asp)

Xử lý sự kiện

- Để xử lý sự kiện, chúng ta có thể thực thi một hàm trong trường hợp xảy ra sự kiện
- Một số cách để thực thi hàm xử lý sự kiện:
  - ❖ Sử dụng **HTML-attribute**
  - ❖ Sử dụng **DOM property**
  - ❖ Sử dụng **addEventListener**

Mouse Events trong Javascript

- **Mouse Events** là đại diện cho các sự kiện khi người dùng tương tác với trang web bằng cách sử dụng chuột

- Một số sự kiện **Mouse Events** tiêu biểu
  - **click** : Được gọi khi click chuột
  - **mousedown** : Được gọi khi ấn chuột xuống
  - **mousemove** : Được gọi khi di chuyển chuột
  - **mouseup** : Được gọi khi nhả chuột ra

Keyboard Events trong Javascript

- **Keyboard Events** là đại diện cho các sự kiện khi người dùng tương tác với trang web bằng cách sử dụng bàn phím

- Một số sự kiện **Keyboard Events** tiêu biểu
  - **keydown** : Được gọi khi ấn phím xuống một phím
  - **keypress** : Được gọi khi ấn và giữ phím
  - **keyup** : Được gọi khi nhả phím ra

Thực hành

Bài thực hành

Học viên xem truy cập vào link để xem nội dung bài thực hành

- Bài thực hành 1: https://codepen.io/buihien1997/pen/wvWmwrm
- Bài thực hành 2: https://codepen.io/buihien1997/pen/mdExbpM

Bài tập về nhà

Bài tập về nhà

- Ôn tập lại các phương thức khi làm việc với DOM
- Làm các bài tập sau:
  - Bài tập 1: https://codepen.io/buihien1997/pen/abZPBPW
  - Bài tập 2: https://codepen.io/buihien1997/pen/ExyGNMm
  - Bài tập 3: https://codepen.io/buihien1997/pen/RwREoXe