Cơ bản về HTML DOM

Khóa học: NỀN TẢNG LẬP TRÌNH

Mục tiêu

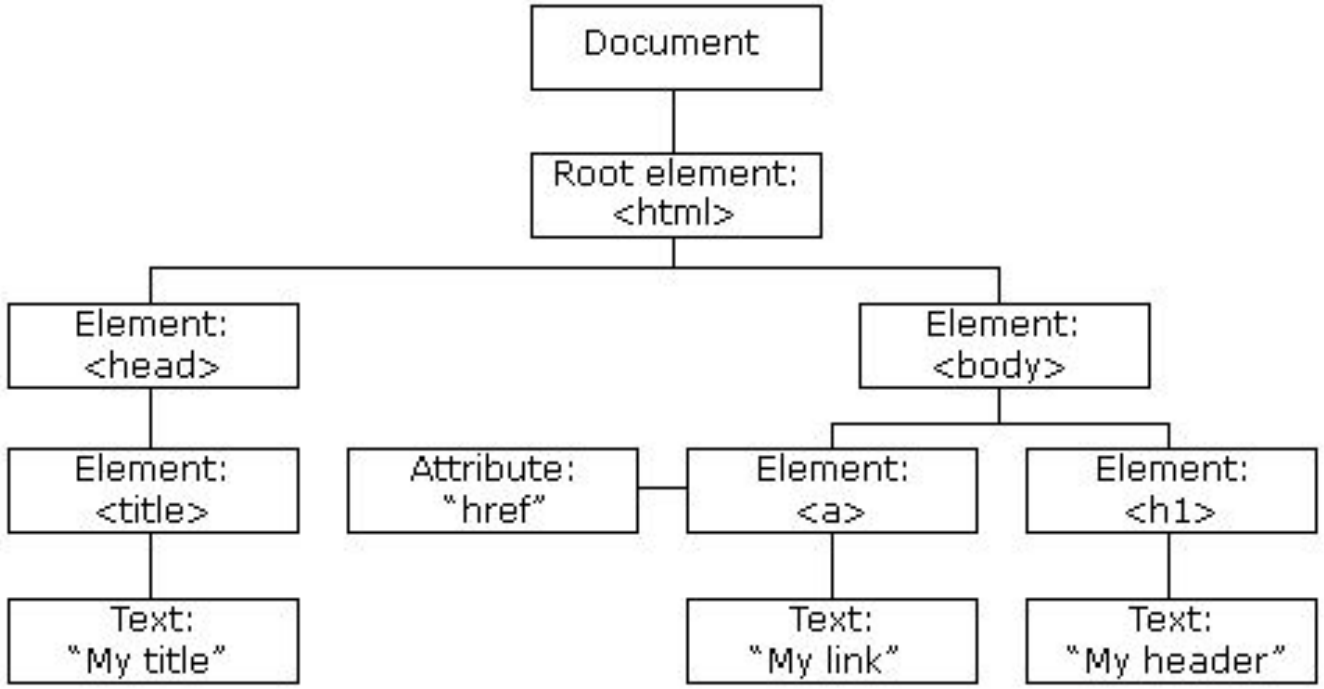
- Trình bày được khái niệm HTML DOM
- Trình bày được khái niệm “Cây đối tượng” (Tree of Objects) trong DOM
- Trình bày được cách sử dụng DOM trong JavaScript
- Trình bày được những thao tác cơ bản với DOM
- Sử dụng JavaScript để truy xuất dữ liệu
- Sử dụng JavaScript để thay đổi nội dung trang web
- Sử dụng JavaScript để thay đổi style trang web
- Sử dụng JavaScript để thêm/xóa phần tử trong nội dung trang web

Khái niệm HTML DOM

- Mô hình đối tượng tài liệu (DOM) biểu diễn dữ liệu của các đối tượng (bao gồm cấu trúc và nội dung) của tài liệu trên web.
- HTML DOM là một tập hợp các câu lệnh lập trình (API) phục vụ những thao tác với đối tượng trong HTML.
  - Định nghĩa cấu trúc logic của trang HTML
  - Cung cấp những phương thức truy cập và làm việc với các đối tượng

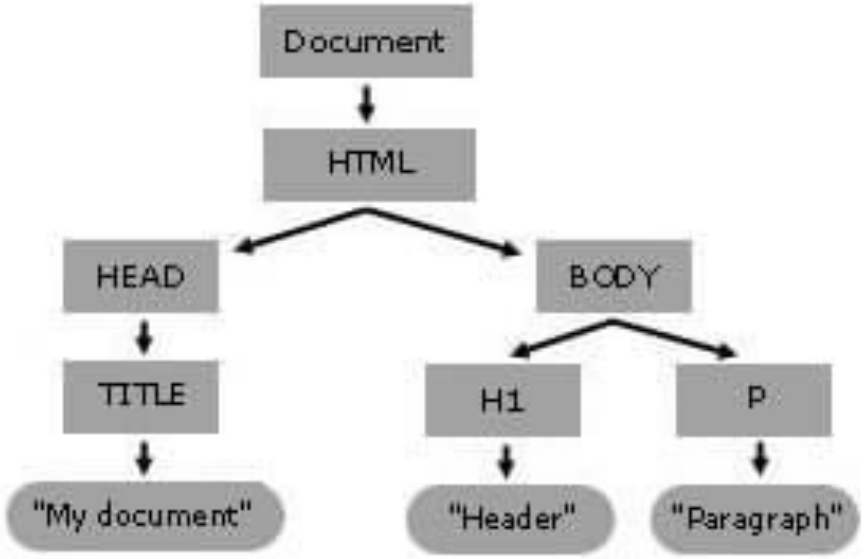
Cây đối tượng

- Khung sườn của một trang HTML là các thẻ (tag).
- Với DOM, mọi thẻ HTML đều là các đối tượng (object).
- Các thẻ lồng nhau là "con" của thẻ bao quanh (Văn bản bên trong thẻ cũng là một đối tượng)



Ví dụ “Cây đối tượng”

```
<html>
<head>
  <title>My Document</title>
</head>
<body>
  <h1>Header</h1>
  <p>Paragraph</p>
</body>
</html>
```



Thao tác cơ bản

Với DOM, chúng ta có thể thực hiện những thao tác sau đây trên một trang web:

- Tham chiếu đến một thành phần (đối tượng) trên trang web. (Nói cách khác, truy cập đến một thành phần trên web)
- Thay đổi nội dung của một thành phần trên trang web
- Thay đổi style của một thành phần trên trang web
- Thêm một thành phần mới vào trang web
- Xóa đi một thành phần đang tồn tại trong trang web
- ...

Truy cập các phần tử trên web

Khi muốn truy cập các phần tử HTML bằng JavaScript, trước tiên chúng ta phải tìm các phần tử đó.

Có những cách dưới đây:

- Tìm các phần tử HTML theo id
- Tìm các phần tử HTML theo tên thẻ (tag)
- Tìm các phần tử HTML theo tên lớp (class name)
- Tìm các phần tử HTML bằng bộ chọn CSS (CSS selector)
- Tìm các phần tử HTML bằng các tập hợp đối tượng HTML

Thuộc tính id trong HTML

- Trong HTML, thuộc tính id được dùng để xác định một id duy nhất cho một phần tử HTML (giá trị phải là duy nhất trong file HTML).
- Giá trị id có thể được dùng trong CSS hoặc JavaScript để thực hiện một tác vụ nhất định cho phần tử duy nhất có giá trị id đó.
- Lưu ý:
  - Thuộc tính **id** có thể dùng với bất kì phần tử HTML nào.
  - Giá trị **id** có phân biệt chữ thường và chữ hoa.
  - Giá trị **id** phải chứa ít nhất 1 kí tự và không chứa khoảng trắng.

Tìm các phần tử HTML theo id

Tìm phần tử HTML theo id là cách dễ nhất để tìm một phần tử HTML trong DOM.

Sử dụng lệnh `document.getElementById(<id-phần tử>)`

Ví dụ: Với mã HTML sau:

```
<p id="demo">Đây là đoạn văn demo trên web</p>
```

Để truy cập vào thẻ <p> có id là “demo”, chúng ta thực thi lệnh:

```
let demoElement = document.getElementById("demo");
```

Thao tác với phần tử HTML

Sau khi tìm được các phần tử HTML, chúng ta có thể thực hiện các thao tác sau với JavaScript:

- Truy xuất nội dung của phần tử HTML
- Thay đổi nội dung của một phần tử HTML
- Thay đổi style của một phần tử HTML

Với một phần tử HTML, chúng ta có thể thao tác với các thuộc tính sau:

- innerHTML
- innerText
- style
- ...

Thuộc tính innerHTML

Thuộc tính *innerHTML* được sử dụng để **truy xuất** hoặc **thay đổi** nội dung bất kỳ phần tử HTML nào, bao gồm <html> và <body>.

Ví dụ: `let demoElement = document.getElementById("demo");`

1. Truy xuất nội dung thẻ có id là “demo”:

```
let demoText = demoElement.innerHTML;
```

2. Thay đổi nội dung thẻ có id là “demo”:

```
demoElement.innerHTML = “Đây là nội dung mới”;
```

Thuộc tính innerText

Tương tự *innerHTML*, thuộc tính *innerText* được sử dụng để **truy xuất** hoặc **thay đổi** nội dung của phần tử HTML.

Sự khác nhau giữa innerHTML và innerText là innerText không xử lý các thẻ HTML bên trong giá trị được gán.

Ví dụ: `let demoElement = document.getElementById("demo");`

1. Thay đổi nội dung thẻ có id là “demo” với innerHTML:

```
demoElement.innerHTML = “Đây là <b>nội dung</b> mới”;
```

2. Thay đổi nội dung thẻ có id là “demo” với innerText:

```
demoElement.innerHTML = “Đây là <b>nội dung</b> mới”;
```

Với *innerHTML*, chữ “nội dung” sẽ được in đậm trên trang web. Trong khi với *innerText*, cặp thẻ và sẽ được hiển thị như một dạng văn bản thông thường.

Thuộc tính style

Thuộc tính *style* được sử dụng để **truy xuất** hoặc **thay đổi** style của phần tử HTML.

Ví dụ: `let demoElement = document.getElementById("demo");`

1. Thay đổi màu chữ cho nội dung trong thẻ có id là “demo”:

```
demoElement.style.color = “red”;
```

2. Thay đổi màu nền cho nội dung trong thẻ có id là “demo”:

```
demoElement.style.background = “yellow”;
```

Demo

Tìm phần từ HTML theo id
Thay đổi nội dung một phần tử HTML
Thay đổi style một phần tử HTML

Thêm phần tử vào trang web (1)

Để thêm một phần tử mới vào HTML DOM, trước tiên phải tạo phần tử, sau đó nối phần tử đó vào một phần tử hiện có.

- Sử dụng **document.createElement()** để tạo phần tử mới.
- Sử dụng **appendChild()** để thêm phần tử hiện có.

Ví dụ:

```
<div id="div1">
    <p id="p1">Đoạn văn thứ nhất.</p>
    <p id="p2">Đoạn văn thứ hai.</p>
</div>

<script>
    const p3 = document.createElement("p");
    p3.innerHTML = "Đoạn văn thứ ba.";

    const div = document.getElementById("div1");
    div.appendChild(p3);
</script>
```

Thêm phần tử vào trang web (2)

Chúng ta có thể chèn một phần tử mới vào trước phần tử tử hiện có trong HTML DOM.

- Sử dụng **insertBefore()** để chèn phần tử mới vào trước phần tử hiện có.

```
<div id="div1">
    <p id="p1">Đoạn văn thứ nhất.</p>
    <p id="p2">Đoạn văn thứ hai.</p>
</div>
```

Ví dụ:

```
<script>
    const p3 = document.createElement("p");
    p3.innerHTML = "Đoạn văn thứ ba.";

    const div = document.getElementById("div1");
    const p1 = document.getElementById("p1");
    div.insertBefore(p3, p1);
</script>
```

Xóa phần tử khỏi trang web (1)

Để xóa một phần tử HTML, hãy sử dụng lệnh **remove()**.

Ví dụ:

```
<div>
    <p id="p1">Đoạn văn thứ nhất.</p>
    <p id="p2">Đoạn văn thứ hai.</p>
</div>

<script>
    const p1 = document.getElementById("p1");
    p1.remove();
</script>
```

Tìm phần tử muốn xóa

Xóa phần tử vừa tìm được

Xóa phần tử khỏi trang web (2)

Lệnh remove() có thể không được hỗ trợ trong một số trình duyệt cũ. Thay vào đó, chúng ta có thể sử dụng lệnh **removeChild()**.

Ví dụ:

```
<div id="div1">
    <p id="p1">Đoạn văn thứ nhất.</p>
    <p id="p2">Đoạn văn thứ hai.</p>
</div>

<script>
    const div1 = document.getElementById("div1");
    const p1 = document.getElementById("p1");
    div1.removeChild(p1);
</script>
```

Tìm phần tử muốn xóa

Xóa phần tử vừa tìm được

Demo

Thêm phần tử vào trang web
Xóa phần tử khỏi trang web

Sự kiện

- JavaScript có thể được thực thi khi một sự kiện nào đó xảy ra, ví dụ như khi người dùng nhấp vào một phần tử HTML.
- Để thực thi mã khi người dùng nhấp vào một phần tử, hãy thêm mã JavaScript vào thuộc tính sự kiện HTML:

thuộc-tính-sự-kiện = **mã-javascript**

- Một số sự kiện phổ biến:
  - *onclick*: Khi người dùng nhấp chuột
  - *onload*: Khi một trang web đã tải xong
  - *onmouseover*: Khi chuột di chuyển qua một phần tử
  - *onchange*: Khi nội dung của <input> được thay đổi
  - *onsubmit*: Khi một biểu mẫu HTML được gửi
  - *onkeypress*: Khi người dùng nhấn phím

Hàm trong JavaScript

Hàm là một khối các mã lệnh được thiết kế để thực hiện một tác vụ cụ thể nào đó.

- Sử dụng từ khóa *function* để định nghĩa (define) một hàm mới
- Hàm sẽ được thực thi khi nó được gọi (invoke).
- Có thể sử dụng với thuộc tính sự kiện nhằm rút gọn các câu lệnh với cú pháp như sau:

thuộc-tính-sự-kiện = **tên-hàm-javascript()**

Lưu ý: Nội dung chi tiết về Hàm sẽ được giới thiệu đầy đủ hơn trong các bài học liên quan. Trong phạm vi bài học này, chúng ta chỉ cần nắm được thao tác định nghĩa hàm và gọi hàm với sự kiện.

Ví dụ về Hàm

Bước 1: Định nghĩa hàm *addElement()* với nhiệm vụ là thêm đoạn văn thứ ba vào nội dung trang web

```
<script>
    function addElement() {
        const p3 = document.createElement("p");
        p3.innerHTML = "Đoạn văn thứ ba.";

        const div = document.getElementById("div1");
        div.appendChild(p3);
    }
</script>
```

Bước 2: Sử dụng nút bấm với thẻ <button>. Khi người dùng click vào nút bấm thì gọi hàm *addElement()* để thực hiện nhiệm vụ.

```
<div id="div1">
    <p id="p1">Đoạn văn thứ nhất.</p>
    <p id="p2">Đoạn văn thứ hai.</p>
</div>

<button onclick="addElement()">Bấm vào đây để thêm đoạn văn!</button>
```

Demo

Sự kiện

Gọi hàm với sự kiện tương ứng

Tóm tắt

- Trình bày khái niệm HTML DOM
- Trình bày khái niệm “Cây đối tượng” (Tree of Objects) trong DOM
- Trình bày cách sử dụng DOM trong JavaScript
- Trình bày những thao tác cơ bản với DOM
- Sử dụng JavaScript để truy xuất dữ liệu
- Sử dụng JavaScript để thay đổi nội dung trang web
- Sử dụng JavaScript để thay đổi style trang web
- Sử dụng JavaScript để thêm/xóa phần tử trong nội dung trang web

Hướng dẫn

Hướng dẫn làm bài thực hành và bài tập

Chuẩn bị bài tiếp theo: *Cấu trúc điều kiện*