



Bài giảng Thiết kế, lập trình Front-End

Bài 4. DOM

Giảng viên

Ths. Vũ Đình Thắng

LƯU Ý

**KHÔNG NÓI
CHUYỆN RIÊNG**



**KHÔNG SỬ DỤNG
ĐIỆN THOẠI**



KHÔNG NGỦ GẬT



GHI CHÉP ĐẦY ĐỦ





Bài 4-HTML DOM

(Giảng viên: Vũ Đình Thắng)

Chương I :	Trang 04
Chương II :	Trang 10
Chương III :	Trang 20
Chương IV :	Trang 30
Chương V :	Trang 40
Chương VI :	Trang 50

Javascript là một ngôn ngữ được sử dụng trong các trình duyệt Browser nên nó đóng một vai trò khá quan trọng trong các ứng dụng website.

Nhiệm vụ của Javascript là thao tác với các tài liệu HTML kết hợp với các cú pháp riêng của nó để tạo nên sự ảo diệu của trang web. Để thao tác được với các thẻ HTML thì nó phải thông qua một cơ chế ta gọi là DOM.

DOM là gì?

DOM là viết tắt của chữ **Document Object Model**, dịch tạm ra là mô hình các đối tượng trong tài liệu HTML.

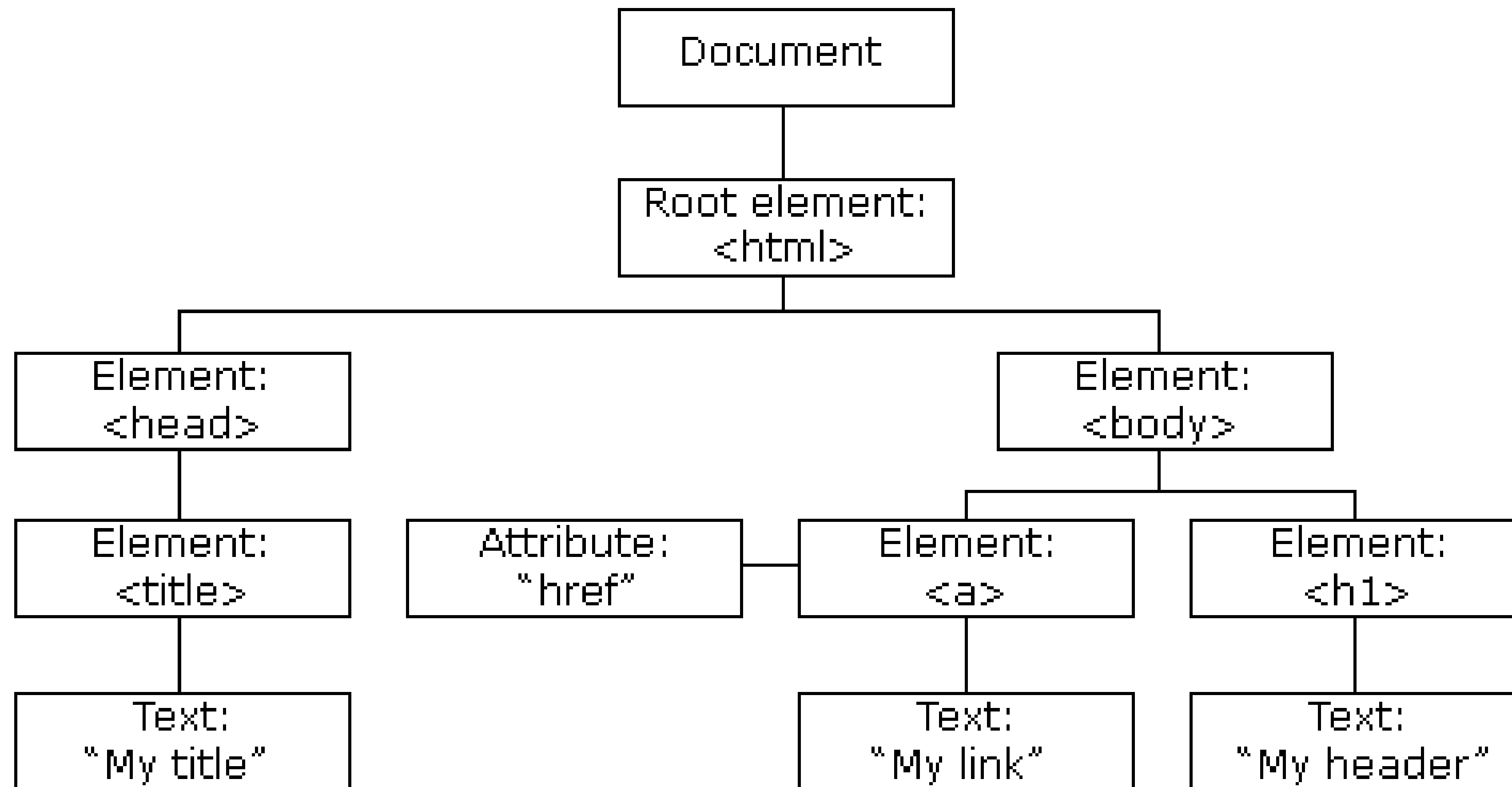
Như các bạn biết trong mỗi thẻ HTML sẽ có những thuộc tính (Properties) và có phân cấp cha - con với các thẻ HTML khác.

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,...

DOM được dùng để truy xuất các tài liệu dạng HTML và XML, có dạng một cây cấu trúc dữ liệu, và thông thường mô hình DOM độc lập với hệ điều hành và dựa theo kỹ thuật lập trình hướng đối tượng để mô tả tài liệu.

DOM là gì?

Cây đối tượng HTML DOM



```
<html>
<body>
  <h1 id="main-content"></h1>
  <script language="javascript">
    document.getElementById("main-content").innerHTML = "Cấu trúc DOM"
  </script>
</body>
</html>
```

Ta thấy sử dụng cấu trúc DOM

```
document.getElementById("main-content").innerHTML = "Cấu trúc DOM"
```

Đoạn code này có ý nghĩa rằng tìm thẻ có id="main-content" và gán nội dung HTML bên trong của thẻ này là dòng chữ "Cấu trúc DOM".

Với mô hình đối tượng, JavaScript có được tất cả sức mạnh cần thiết để tạo HTML động:

JavaScript có thể thay đổi tất cả các thành phần HTML trong trang

JavaScript có thể thay đổi tất cả thuộc tính HTML trong trang

JavaScript có thể thay đổi tất cả các kiểu CSS trong trang

JavaScript có thể xóa các thành phần và thuộc tính HTML hiện có

JavaScript có thể thêm các thành phần và thuộc tính HTML mới

JavaScript có thể phản ứng với tất cả các sự kiện HTML hiện có trong trang

JavaScript có thể tạo các sự kiện HTML mới trong trang

HTML DOM là một mô hình **đối tượng** và **giao diện lập trình** tiêu chuẩn cho HTML. Nó định nghĩa:

- Các phần tử HTML dưới dạng **đối tượng**
- Thuộc **tính** của tất cả các phần tử HTML
- Các **phương pháp** truy cập tất cả các phần tử HTML
- Các **sự kiện** cho tất cả các phần tử HTML

Nói cách khác: **HTML DOM** là một **tiêu chuẩn** về cách lấy, thay đổi, thêm hoặc xóa các phần tử HTML.

Giao diện lập trình DOM

- HTML DOM có thể được truy cập bằng JavaScript (và bằng các ngôn ngữ lập trình khác).
- Trong DOM, tất cả các phần tử HTML được định nghĩa là đối tượng .
- Giao diện lập trình là các thuộc tính và phương thức của từng đối tượng.
- Thuộc tính là một giá trị mà bạn có thể nhận hoặc đặt (như thay đổi nội dung của phần tử HTML).
- Phương thức là một hành động bạn có thể thực hiện (như thêm hoặc xóa phần tử HTML).

Danh sách nhóm DOM:

- DOM document: có nhiệm vụ lưu trữ toàn bộ các thành phần trong tài liệu của website
- DOM element: có nhiệm vụ truy xuất tới thẻ HTML nào đó thông qua các thuộc tính như tên class, id, name của thẻ HTML
- DOM HTML: có nhiệm vụ thay đổi giá trị nội dung và giá trị thuộc tính của các thẻ HTML
- DOM CSS: có nhiệm vụ thay đổi các định dạng CSS của thẻ HTML
- DOM Event: có nhiệm vụ gán các sự kiện như onclick(), onload() vào các thẻ HTML
- DOM Listener: có nhiệm vụ lắng nghe các sự kiện tác động lên thẻ HTML đó
- 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: có nhiệm vụ thao tác với HTML thông qua đối tượng (Object)

Method	Mô tả
<code>document.getElementById(<i>id</i>)</code>	Tìm 1 phần tử theo id
<code>document.getElementsByTagName(<i>name</i>)</code>	Tìm 1 phần tử theo tên thẻ
<code>document.getElementsByClassName(<i>name</i>)</code>	Tìm 1 phần tử theo tên lớp

Thay đổi các phần tử HTML

Thuộc tính	Mô tả
<code>element.innerHTML = new html content</code>	Change the inner HTML of an element
<code>element.attribute = new value</code>	Change the attribute value of an HTML element
<code>element.style.property = new style</code>	Change the style of an HTML element
Phương thức	Mô tả
<code>element.setAttribute(<i>attribute</i>, <i>value</i>)</code>	Change the attribute value of an HTML element

Thêm và xóa các phần tử

Phương thức	Mô tả
<code>document.createElement(<i>element</i>)</code>	Tạo 1 phần tử HTML
<code>document.removeChild(<i>element</i>)</code>	Xóa bỏ 1 phần tử
<code>document.appendChild(<i>element</i>)</code>	Nối thêm 1 phần tử
<code>document.replaceChild(<i>new</i>, <i>old</i>)</code>	Thay thế 1 phần tử con
<code>document.write(<i>text</i>)</code>	Write into the HTML output stream

Thêm trình xử lý sự kiện

Method	Description
<code>document.getElementById(<i>id</i>).onclick = function(){<i>code</i>}</code>	Thêm mã xử lý sự kiện vào sự kiện onClick

Phương thức này giúp ta truy xuất đến thẻ HTML thông qua ID. Nếu trong tài liệu html của bạn bị trùng ID thì nó sẽ lấy thẻ html có ID trùng khớp đầu tiên

```
<body>
  <p id="p1">Tôi là javascript</p>
  <script type="text/javascript" language="javascript">
    var noidung=document.getElementById("p1");
    alert(noidung.innerHTML);
  </script>
</body>
```

Phương thức này lấy tất cả các thẻ html có tên trùng với tham số truyền vào. Ví dụ dưới đây là mình lấy tất cả thẻ **h2**.

```
<body>
  <p id="p1">Tôi là javascript</p>
  <h2>Chào bạn</h2>
  <h2>Tôi là javascript</h2>
  <script type="text/javascript">
    (method) Document.getElementsByTagName<"h2">(qualifiedName: "h2"):
    HTMLCollectionOf<HTMLHeadingElement> (+4 overloads)
    var h=document.getElementsByTagName("h2");
    h[0].style.background="Green";
    h[0].style.color="white";
    h[1].style.background="yellow";
    h[1].style.color="red";
    h[1].style.fontStyle="italic";
    var p1=document.getElementById("p1");
    p1.style.color="blue";
    p1.innerHTML=h[0].innerHTML;
  </script>
```

Chào bạn

Chào bạn

Tôi là javascript

Phương thức này lấy tất cả thẻ html có tên class trùng với tên của tham số truyền vào.

```
8 <body>
9   <p id="p1">Tôi là javascript</p>
10  <h2 class="h">Chào bạn</h2>
11  <script type="text/javascript" language="javascript">
12      var h=document.getElementsByClassName("h");
13      .....var p1=document.getElementById("p1");
14      p1.style.color="blue";
15      p1.innerHTML=h[0].innerHTML;
16  </script>
```

Chào bạn

Chào bạn

Phương thức này lấy tất cả thẻ html có tên **name** trùng với tên của tham số truyền vào.

```
<body>
  <p id="p1">Tôi là javascript</p>
  <h2 name="h">Chào bạn</h2>
  <script type="text/javascript" language="javascript">
    var h=document.getElementsByName("h");
    var p1=document.getElementById("p1");
    p1.style.color="blue";
    p1.innerHTML=h[0].innerHTML;
  </script>
</body>
```

Chào bạn

Chào bạn

Tìm tất cả các phần tử HTML khớp với một bộ chọn CSS được chỉ định (id, tên lớp, loại, thuộc tính, giá trị của thuộc tính, v.v.), hãy sử dụng `querySelectorAll()` phương thức này.

Ví dụ này trả về danh sách tất cả `<p>` các phần tử có `class="intro"`..

```
<body>
  <p>Tìm các phần tử HTML bằng Query Selector</p>
  <p class="intro">Hello World!.</p>
  <p class="intro">Ví dụ này minh họa phương thức <b>querySelectorAll</b>.</p>
  <p id="demo"></p>
  <script type="text/javascript" language="javascript">
    const x = document.querySelectorAll("p.intro");
    document.getElementById("demo").innerHTML = 'Đoạn đầu tiên (index 0) với class="intro" là: ' + x[0].innerHTML;
  </script>
```

Tìm các phần tử HTML bằng Query Selector

Hello World!.

Ví dụ này minh họa phương thức **querySelectorAll**.

Tìm phần tử biểu mẫu có id="frm1", trong bộ sưu tập biểu mẫu và hiển thị tất cả các giá trị phần tử:

JavaScript HTML DOM

Finding HTML Elements Using **document.forms**.

First name:
Last name:

These are the values of each element in the form:

Donald
Duck
Submit

```
<body>
  <h2>JavaScript HTML DOM</h2>
  <p>Finding HTML Elements Using <b>document.forms</b>.</p>
  <form id="frm1" action="/action_page.php">
    First name: <input type="text" name="fname" value="Donald"><br>
    Last name: <input type="text" name="lname" value="Duck"><br><br>
    <input type="submit" value="Submit">
  </form>
  <p>These are the values of each element in the form:</p>
  <p id="demo"></p>
  <script>
    const x = document.forms["frm1"];
    let text = "";
    for (let i = 0; i < x.length ;i++) {
      text += x.elements[i].value + "<br>";
    }
    document.getElementById("demo").innerHTML = text;
  </script>
</body>
```

Lấy các phần tử FORM theo cú pháp:

Cú pháp:

`document.tênForm.Tên_điều_khiển.Value`

Ví dụ:

```
var user=document.dangnhap.username.value;
```

Tất cả các element (phần tử) trên trang web đều có một tập các sự kiện tương ứng

Chú ý: Cùng một element, các browser khác nhau hỗ trợ các tập sự kiện khác nhau

- **onClick:** Được kích hoạt khi nhấn chuột vào một element
- **onLoad và on Unload:** Được kích hoạt khi người dùng vào hoặc thoát khỏi trang web
- **onFocus, onBlur, onChange:** Được kích hoạt khi các trường nhận được focus, mất focus hay được thay đổi giá trị
- **onMouseOver:** Được kích hoạt khi người dùng di chuột lên một element HTML trên form
- **onMouseOut:** Được tạo ra bất cứ khi nào con trỏ chuột di chuyển ra khỏi phần tử đó.

```
<body>
```

```
    <button type="button" onclick="alert('Hello')">Sự kiện</button>
```

```
</body>
```

Sự kiện

127.0.0.1:5500 cho biết

Hello

OK

Một sự kiện có thể gọi đến nhiều hàm
Ví dụ:

```
<body>
```

```
    <button type="button" onclick="ok=confirm('bạn chắc chứ');alert('Bạn đã chọn: '+ok)">
```

Sự kiện

```
</button>
```

```
</body>
```

linh học

127.0.0.1:5500 cho biết

Sự kiện

bạn chắc chứ

OK

Hủy

127.0.0.1:5500 cho biết

Bạn đã chọn: true

OK

LẤY CÁC GIÁ TRỊ TỪ CÁC PHẦN TỬ TRONG FORM

1. TEXTBOX

- a. `document.tên_form.tên textbox.value`
- b. `Document.getElementById(tên-ID).value`

2. VỚI TẤT CẢ CÁC ĐIỀU KHIỂN

- ☐ Sử dụng phương thức `getElementById(id)`
- ☐ Sử dụng phương thức `getElementsByName(name)`
- ☐ Sử dụng phương thức `getElementsByTagName(name)`

3. GÁN SỰ KIỆN

<ten thẻ ...sự kiện="ten_hàm(Đổi số)">

<head>

<script language="javaScript">

function Tên_hàm(Đổi số){

Khai báo biến

//Lấy giá trị từ form

document.tên_form.tên textbox.value hoặc

document.getElementById(tên textbox).value

....

Tính toán

Gán giá trị cho ô kết quả

document.tên_form.tên textbox.value=tên biến kết quả

}

</script>

</head>

```
<body>
  <form name="tên Form"...>
    <ten thẻ ...sự kiện="ten_hàm(Đổi số)">
  </Form>
</Body>
```

Ví dụ

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>su kien</title>
8   <script type="text/javascript">
9     var num =0
10    function showLink(num){
11      var hienthi=document.getElementById("hienthi");
12      if (num==1) {
13        hienthi.value= "Bạn đã chọn CĐNTH FPT";
14      }
15      if (num==2) {
16        hienthi.value = "Bạn đã chọn ĐẠI HỌC FPT";
17      }
18      if (num==3) {
19        hienthi.value = "Bạn đã chọn ĐẠI HỌC CÔNG NGHIỆP";
20      }
21    }
22    function thongbao(){
23      hienthi.value = "ban phải đưa chuột vào 1 trong các liên kết";
24    }
25  </script>
26 </head>
27 <body>
28   <form>
29     <input type="text" size=60 id="hienthi" >
30   </form>
31   <a href="#" onMouseOver="showLink(1)"document.bgcolor= "green" onmouseout="thongbao()">CĐN BCVT</a><br>
32   <a href="#" onMouseOver="showLink(2)" onmouseout="thongbao()">CĐCQ BCVT</a><br>
33   <a href="#" onMouseOver="showLink(3)" onmouseout="thongbao()">CĐ Điện Lực</a><br>
34 </body>
35 </html>
```

Bạn đã chọn CĐTH FPT

CĐTH FPT

ĐẠI HỌC FPT

ĐẠI HỌC CÔNG NGHIỆP

ban phải đưa chuột vào 1 trong các liên kết

CĐTH FPT

ĐẠI HỌC FPT

ĐẠI HỌC CÔNG NGHIỆP

KHÁC BIỆT GIỮA INNERTEXT, INNERHTML, TEXTCONTENT

Để lấy nội dung trong thẻ ta thường thấy người ta dùng `innerText`, `innerHTML` hoặc có khi dùng `textContent`. vậy mấy cái này khác nhau như thế nào.

Trong ví dụ dưới đây ta thấy

Ta thấy `innerHTML` sẽ in ra nội dung text và những thẻ trong div này luôn.

Còn `textContent` sẽ in ra nội dung text và nội dung các thẻ bên trong div này luôn.

Còn `innerText` sẽ in ra nội dung text, không in ra nội dung của thẻ ẩn(do thẻ span có `display: none`, cái này dùng để ẩn thẻ span này).

ví dụ:

```
<div id='hello'>Hello Javascript <span style="display:none;"> 2020</span></div>
```

```
<script>
```

```
var mydiv = document.getElementById('hello');
```

```
console.log(mydiv.innerHTML);
```

```
// Hello Javascript <span style="display:none;"> 2020</span>
```

```
console.log(mydiv.textContent);
```

```
// Hello Javascript 2020
```

```
console.log(mydiv.innerText);
```

```
// Hello Javascript
```

```
</script>
```

```
Hello Javascript <span style="display:none;"> 2020</span>
```

```
Hello Javascript 2020
```

```
Hello Javascript
```




Lời cảm ơn!

HỌ VÀ TÊN NGƯỜI CẢM ƠN

Chức danh

Thông tin liên hệ (nếu cần)