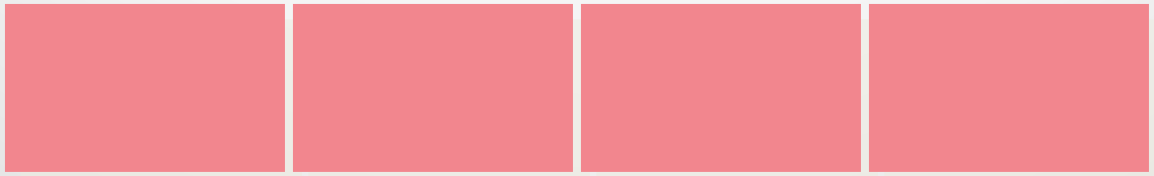
Bài giảng

PHÁT TRIỂN ỨNG DỤNG WEB

Lê Đình Thanh
Khoa Công nghệ Thông tin
Trường Đại học Công nghệ, ĐHQGHN
E-mail: thanhld@vnu.edu.vn Mobile: 0987.257.504

Chương 4

Quản lý trang web bằng JavaScript



Nội dung

- JavaScript
- DOM
- BOM
- Quản lý trang web
- SOP
- Vấn đề của trình duyệt

JavaScript

Tại sao sử dụng JavaScript?

- HTML:
  - Cung cấp các thẻ tạo (khai báo) đối tượng tài liệu nhưng không cung cấp khả năng quản lý (hủy, thay đổi thuộc tính, triệu gọi phương thức) chúng.
    - Ví dụ: thẻ *<input type = "button" …>* tạo một nút bấm nhưng HTML không xử lý sự kiện khi nút được bấm (onclick).
- JavaScript (Scripts):
  - Quản lý (tạo, hủy bỏ, thay đổi thuộc tính, triệu gọi phương thức) các đối tượng.

JavaScript

- Được sử dụng rộng rãi.
- Tựa C, Java.
  - *Khác C ở các điểm*:
    - Định kiểu không tường minh
    - Khai báo biến bằng từ khóa var;
    - Định nghĩa hàm bằng từ khóa function.
    - Mảng là ánh xạ
    - first-class function
    - prototype-based
- Sử dụng cùng HTML:
  - Viết lệnh JavaScript trong cặp thẻ *<script type= "text/javascript"> </script>* - phân đoạn Javascript.
  - Có thể đặt (nhiều) phân đoạn JavaScript tại bất kỳ đâu trong trang HTML.
  - Gọi hàm JavaScript trong các thuộc tính sự kiện của các đối tượng HTML.

Khai báo, sử dụng biến

Trang2 - Notepad

File Edit Format View Help

```
l>
<head>
      <title>Trang thứ 2</title>
</head>
<body>
       <h3>Ví dụ sử dụng Javascript</h3>
       <form method = "POST" action = "Default.htm">
                 <script language = "JavaScript">
                           var a = 10, b = 12;
                 </script>
                 <table>
                           <tr>
                                     <td>a =<script language = "JavaScript">document.write(a);</script></td>
                           </tr>
                           <tr>
                                     <td>b =<script language = "JavaScript">document.write(b);</script></td>
                           </tr>
                           <tr>
                                     <td>a+b =<script language = "JavaScript">document.write(a+b);</script></td>
                           </tr>
                 </table>
       </form>
</body>
il>
```

Khai báo, sử dụng biến

Trang2 - Notepad

File Edit Format View Help

```
l>
<head>
      <title>Trang thứ 2</title>
</head>
<body>
       <h3>Ví dụ sử dụng Javascript</h3>
       <form method = "POST" action = "Default.htm">
                  <script language = "JavaScript">
                             var a = 10, b = 12;
                  </script>
                  <table>
                             <tr>
                                        <td>a =<script language = "JavaScript">document.write(a);</script></td>
                             </tr>
                             <tr>
                                        <td>b =<scr
                             </tr>
                             <tr>
                                        <td>a+b =<
                             </tr>
                  </table>
       </form>
</body>
l>
```

Trang thơ 2 - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites

Address D:\Web_Vidu\Trang2.htm Go Links

Ví dụ sử dụng Javascript

a =10

b =12

a+b =22

start LAC... Tran... Tran... KHG... 10:14 PM

Khai báo, sử dụng hàm

Trang2 - Notepad

File Edit Format View Help

```
<h3>Ví dụ sử dụng Javascript</h3>
<form method = "POST" action = "Default.htm">
        <script language = "JavaScript">
                var a = 10, b = 12;

                //Ham nay cong hai so
                function Cong(x, y)
                {
                        return (x+y);
                }
        </script>
        <table>
                <tr>
                        <td>a =<script language = "JavaScript">document.write(a);</script></td>
                </tr>
                <tr>
                        <td>b =<script language = "JavaScript">document.write(b);</script></td>
                </tr>
                <tr>
                        <td>a+b =<script language = "JavaScript">document.write(Cong(a, b));</script></td>
                </tr>
        </table>
 </form>
y>
```

Mảng

- Khai báo
  - Chính tắc
    - var myCars=new Array();
      myCars[0]="Saab";
      myCars[1]="Volvo";
  - Rút gọn
    - var myCars=new Array("Saab", "Volvo", "BMW");
  - Nguyên thủy
    - var myCars=["Saab", "Volvo“, "BMW"];
- Truy cập phần tử
  - var name=myCars[0];
  - myCars[0]="Saab";

Mảng

```
var a = ["Hoàng", 'M', new Date('1998-3-18'), [5, 9, 7]];
a[100] = "CLC";                    //Thêm mới
document.write(a.length);          //101
document.write(a[1]);              //M
a[1] = 'F';                        //Thay đổi giá trị
document.write(a[1]);              //F
document.write(a[2]);              //Wed Mar 18 1998
                                   07:00:00 GMT+0700 (ICT)
document.write(a[3][1]);           //9
document.write(a[15]);             //undefined
```

Khai báo đối tượng

- Đối tượng là sưu tập (collection)/kết hợp (association) động của các thuộc tính và phương thức: Thêm và bớt thuộc tính/phương thức bất kỳ.
- Khai báo trực tiếp

```
var person={fullname:"John", sayHello:function() {
            document.write(this.fullname + " says hello!");}};
person.sayHello();
person.dob = "22/2/2012";
delete person.fullname;
```

- Khai báo đối tượng Object rồi gắn thuộc tính và phương thức cho nó

```
person = new Object();
person.fullname = "John";
person.sayHello = function() { document.write(this.fullname +
                                                " says hello!");}
person.sayHello();
```

Sử dụng hàm tạo

- Để tạo hàng loạt đối tượng có kiểu giống nhau

```
function Person(fn, al) {
        this.fullname = fn;
        this.alias = al;
        this.sayHello = function() {
                document.write(this.fullname + " " + this.alias);
        }
}
person = new Person("Hoàng Tùng", "Bolero");
person.sayHello(); //Hoàng Tùng Bolero
papa = new Person("Hoàng Bách", "Sava");
papa.sayHello(); //Hoàng Bách Sava

Person.prototype.sayGoodbye = function () {
        document.write(this.fullname +  " good bye everyone!");
};
person.sayGoodbye(); //Hoàng Tùng goodbye everyone!
papa.sayGoodbye(); // Hoàng Bách goodbye everyone!
```

Phạm vi truy cập thuộc tính, phương thức

```
function Person(fn, al) {
        var fullname = fn;          //private
        var alias = al;             //private
        function getAllNames() {return (fullname + " " + alias);} //private
        this.sayHello = function() { //public
                document.write(getAllNames.apply(this));
        }
}
person = new Person("Hoàng Tùng", "Bolero");
person.sayHello(); //Hoàng Tùng Bolero
person.getAllNames(); //Lỗi
```

Getters, setters

```
var o = {
  a: 7,
  get b() {
    return this.a + 1;
  },
  set c(x) {
    this.a = x / 2
  }
};

console.log(o.a); // 7
console.log(o.b); // 8
o.c = 50;
console.log(o.a); // 25
```

Kế thừa

- JavaScript là ngôn ngữ lập trình dựa trên nguyên mẫu (prototype-based)
- Mỗi đối tượng tham chiếu đến một đối tượng khác (gọi là đối tượng nguyên mẫu) và kế thừa các thuộc tính, phương thức của đối tượng nguyên mẫu.
- Nguyên mẫu của các đối tượng Object là null.
- Mặc định, nguyên mẫu của đối tượng là Object.prototype
  - `var a = {p: 1};  // a ---> Object.prototype ---> null`
  - `console.log(a.toString());`
- Sử dụng Object.create() để chỉ định nguyên mẫu
  - `var b = Object.create(a); // b ---> a ---> Object.prototype ---> null`
  - `console.log(b.p); // 1 (inherited)`

Kế thừa

- Lấy nguyên mẫu của đối tượng
  - Object.getPrototypeOf(g);
- Đối tượng nguyên mẫu lại có nguyên mẫu của nó. Quan hệ nguyên mẫu tạo nên chuỗi nguyên mẫu (prototype chain).
  - a--→b--→c --→ …-→ Object.prototype –→ null

Kế thừa

```
function Person(fn, al) {
        var fullname = fn;
        var alias = al;
        function getAllNames() {return (fullname + " " + alias);}
        this.sayHello = function() {
                document.write(getAllNames.apply(this));
        }
}
person = new Person("Hoàng Tùng", "Bolero");
faculty  = Object.create(person);
faculty.department = "Khoa CNTT";
faculty.sayHello = function() {
        person.sayHello();
        document.write(" " + this.department);
};
faculty.sayHello(); //Hoàng Tùng Bolero Khoa CNTT
```

Kế thừa theo hàm tạo

```
function Person(fn, al) {
        var fullname = fn;                      //private
        var alias = al;                         //private
        function getAllNames() {return (fullname + " " + alias);} //private
        this.sayHello = function() { //public
                document.write(getAllNames.apply(this));
        };
        this.getFullname = function() {return fullname;}; //public
}
function Staff(fn, al, sa) {
        Person.call(this, fn, al);
        var salary = sa;
        var parentHello = this.sayHello;
        this.sayHello = function() {  //overriding
                parentHello.apply(this);
                document.write(" with salary " + salary);
        };
        this.sayGoodbye = function() {
            document.write(this.getFullname() + " good bye everyone!");
        };
}
staff = new Staff("Hoàng Ngân", "Diamon", 1000);
staff.sayHello(); //Hoàng Ngân Diamon with salary 1000
staff.sayGoodbye(); //Hoàng Ngân good bye everyone!
```

Xâu

- Chuỗi ký tự nằm giữa nháy đơn (‘) hoặc nháy kép (“)
  - var carname="Volvo XC60";
    var carname='Volvo XC60';
- Truy cập ký tự trong xâu
  - var character=carname[7];
- Độ dài xâu
  - var txt="Hello World!";
    document.write(txt.length);

Xâu

- Tìm xâu con
  - var str="Hello world, welcome to the universe.";
    var n=str.indexOf("welcome");
- Thay thế xâu con
  - str="Please visit Microsoft!"
    var n=str.replace("Microsoft","W3Schools");

Xâu

- str.substring(begin, end): Trả về xâu con bao gồm các ký tự có chỉ mục từ begin đến end-1 của xâu str
- str.substring(begin): Trả về xâu con bao gồm các ký tự có chỉ mục từ begin đến hết của xâu str
- str.split(deli): Tách xâu str bởi sử dụng xâu ngăn cách deli. Trả về mảng các xâu kết quả

Xâu

- str. toUpperCase(): Trả về xâu viết hoa của str
- str. toLowerCase(): Trả về xâu viết thường của str
- isNaN(s): true nếu s không là biểu diễn số
- parseInt(s): Giá trị nguyên của biểu diễn s
- parseFloat(s): Giá trị thực của biểu diễn s

Chú thích

```
// Chú thích trên một dòng JavaScript


/*
Chú thích trên nhiều
dòng JavaScript
*/
```

ECMAScript

- ECMAScript là đặc tả (JavaScript là một cài đặt cụ thể của ECMAScript)
- Các phiên bản ECMAScript
  - 1st edition ES1 6/1997
  - 2nd edition ES2 6/1998
  - 3rd edition ES3 12/1999
  - 4th edition ES4 abandoned 2003
  - 5th edtion ES5 12/2009
  - **6th edition ES2015 6/2015**
  - 7th edition ES2016 6/2016
  - 8th edition ES2017 6/2017
  - 9th edition ES2018 6/2018
  - 10th edition ES2019 6/2019
  - 11th edition ES2020 6/2020
  - 12th edition ES2021 6/2021

ECMAScript (tiếp)

Browser Support for ES6 (2015)

Safari 10 and Edge 14 were the first browsers to fully support ES6:

				
Chrome 58	Edge 14	Firefox 54	Safari 10	Opera 55
Jan 2017	Aug 2016	Mar 2017	Jul 2016	Aug 2018

- Nếu trình duyệt chưa hỗ trợ (chưa cài đặt) các features của ECMAScript
  - Có thể dùng polyfill như Babel https://babeljs.io

ES2015

- class
- module

class

- Cho phép định nghĩa lớp tương tự lập trình hướng đối tượng
  - dễ dàng hơn sử dụng hàm tạo
- Lưu ý:
  - class/lớp chỉ là cách viết khác thay cho hàm tạo, hay class là "sugar syntax" cho function
  - **class/lớp là một loại hàm**

Ví dụ: Sử dụng function

```
function PersonF(fn, al) {
    var fullname = fn;        // private
    var alias = al;           // private
    function getAllNames() { //private
        return (fullname + " " + alias);
    };
    this.getFullname = function() {return fullname;}; //public
    this.sayHello = function() { //public
        console.log(getAllNames.apply(this));
    };
}
```

Source code: https://itest.com.vn/lects/webappdev/classes/

Ví dụ: Viết lại bằng class

```
class PersonC {
    #fullname;  // private
    #alias;     // private

    constructor(fn, al) {
        this.#fullname = fn;
        this.#alias = al;
    }

    #getAllNames() { // private
        return (this.#fullname + " " + this.#alias);
    }

    getFullname() {return this.#fullname;}; //public

    sayHello() { //public
        console.log(this.#getAllNames());
    }
}
```

Ví dụ: Khai báo đối tượng

```
function PersonF(fn, al) {
    var fullname = fn;      // private
    var alias = al;         // private
    function getAllNames() { //private
        return (fullname + " " + alias);
    };
    this.getFullname = function() {return fullname;}; //public
    this.sayHello = function() { //public
        console.log(getAllNames.apply(this));
    };
}
```

```
class PersonC {
    #fullname;  // private
    #alias;     // private

    constructor(fn, al) {
        this.#fullname = fn;
        this.#alias = al;
    }

    #getAllNames() { // private
        return (this.#fullname + " " + this.#alias);
    }

    getFullname() {return this.#fullname;}; //public

    sayHello() { //public
        console.log(this.#getAllNames());
```

```
let p1 = new PersonF("Hoàng Tùng", "Bolero");
p1.sayHello(); //Hoàng Tùng Bolero

let p2 = new PersonC("Trần Bình", "Ballad");
p2.sayHello(); // Trần Bình Ballad


console.log(typeof(PersonF)); // function
console.log(typeof(PersonC)); // function
```

Kế thừa: Sử dụng hàm tạo

```
function StaffF(fn, al, sa) {
    PersonF.call(this, fn, al); // kế thừa
    var salary = sa;
    var parentHello = this.sayHello;
    this.sayHello = function() {  // overriding
        parentHello.apply(this);
        console.log(" with salary " + salary);
    };
    this.sayGoodbye = function() {
        console.log(this.getFullname() + " good bye everyone!");
    };
}
staff = new StaffF("Hoàng Ngân", "Diamon", 1000);
staff.sayHello(); // Hoàng Ngân Diamon with salary 1000
staff.sayGoodbye(); // Hoàng Ngân good bye everyone!
```

Kế thừa: Viết lại bằng class

```
class StaffC extends PersonC { // kế thừa
    #salary;

    constructor(fn, al, sa) {
        super(fn, al);
        this.#salary = sa;
    }

    sayHello() {  // overriding
        super.sayHello();
        console.log(" with salary " + this.#salary);
    };

    sayGoodbye() {
        console.log(this.getFullname() + " good bye everyone!");
    };
}

staff = new StaffC("Trần Bình", "Ballad", 1000);
staff.sayHello(); // Hoàng Ngân Diamon with salary 1000
staff.sayGoodbye(); // Hoàng Ngân good bye everyone!
```

Kế thừa: Từ hàm tạo

```
class StaffD extends PersonF { // kế thừa từ function
    #salary;

    constructor(fn, al, sa) {
        super(fn, al);
        this.#salary = sa;
    }

    sayHello() {  // overriding
        super.sayHello();
        console.log(" with salary " + this.#salary);
    };

    sayGoodbye() {
        console.log(this.getFullname() + " good bye everyone!");
    };
}

staff = new StaffD("Trần Bình", "Ballad", 1000);
staff.sayHello(); // Hoàng Ngân Diamon with salary 1000
staff.sayGoodbye(); // Hoàng Ngân good bye everyone!
```

modules

	Chrome	Edge	Firefox	Internet Explorer	Opera	Safari	WebView Android	Chrome Android	Firefox for Android	Opera Android	Safari on iOS	Samsung Internet	Deno	Node.js
`import`	✓ 61	✓ 16	✓ 60	✗ No	✓ 48	✓ 10.1	✓ 61	✓ 61	✓ 60	✓ 45	✓ 10.3	✓ 8.0	✓ 1.0	✓ 13.2.0 *
Dynamic import	✓ 63	✓ 79	✓ 67	✗ No	✓ 50	✓ 11.1	✓ 63	✓ 63	✓ 67	✓ 46	✓ 11.3	✓ 8.0	✓ 1.0	✓ 13.2.0 *
Available in workers	✓ 80	✓ 80	✗ No *	✗ No	✗ No	✓ 15	✓ 80	✓ 80	✗ No *	✗ No	✓ 15	✓ 13.0	✓ 1.0	✗ No

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Modules

modules

- Mỗi tệp .js là một module, mỗi module trong một tệp.
- Định nghĩa module và export các features trong module
- import và sử dụng các features từ module khác
- Áp dụng module trong HTML

export

f.js × c.js × lib.js

```
1    function Person(fn, al) {
2        var fullname = fn;       // private
3        var alias = al;          // private
4        function getAllNames() { //private
5            return (fullname + " " + alias);
6        };
7        this.getFullname = function() {return fullname;}; //public
8        this.sayHello = function() { //public
9            console.log(getAllNames.apply(this));
10       };
11   }
12
13   function Staff(fn, al, sa) {
14       Person.call(this, fn, al); // kế thừa
15       var salary = sa;
16       var parentHello = this.sayHello;
17       this.sayHello = function() {  // overriding
18           parentHello.apply(this);
19           console.log(" with salary " + salary);
20       };
21       this.sayGoodbye = function() {
22           console.log(this.getFullname() + " good bye everyone!");
23       };
24   }
25
26   export {Person, Staff};
```

export (tiếp)

```
22    class Person { ... }
23    class Staff extends Person { // kế thừa
24        #salary;
25
26        constructor(fn, al, sa) {
27            super(fn, al);
28            this.#salary = sa;
29        }
30
31        sayHello() {  // overriding
32            super.sayHello();
33            console.log(" with salary " + this.#salary);
34        };
35
36        sayGoodbye() {
37            console.log(this.getFullname() + " good bye everyone!");
38        };
39    }
40
41
42    export {Person, Staff};
```

import, aggregating export

lib.js

```
1 import {Person as PersonF} from "./f.js";
2
3 class StaffD extends PersonF { // kế thừa từ function
4         #salary;
5         PersonF
6         constructor(fn, al, sa) {
7             super(fn, al);
8             this.#salary = sa;
9         }
10
11        sayHello() {  // overriding
12            super.sayHello();
13            console.log(" with salary " + this.#salary);
14        };
15
16        sayGoodbye() {
17            console.log(this.getFullname() + " good bye everyone!");
18        };
19    }
20
21
22 export {Person as PersonF, Staff as StaffF} from './f.js';
23 export {Person as PersonC, Staff as StaffC} from './c.js';
24 export {StaffD};
```

Sử dụng module trong HTML

```
1 <!DOCTYPE html><html><head>
2     <title>JavaScript module</title>
3     <meta charset="utf-8">
4 </head><body>
5 <b>Hướng dẫn:</b> Mở xem nguồn trang (View Page Source) c
  duyệt để xem kết quả (output).
6 <script type="module">
7     import *  as HRM from './lib.js';
8
9
10    let p1 = new HRM.PersonF("Hoàng Tùng", "Bolero");
11    p1.sayHello(); //Hoàng Tùng Bolero
12
13    let p2 = new HRM.PersonC("Trần Bình", "Ballad");
14    p2.sayHello(); // Trần Bình Ballad
15
16
17    console.log(typeof(HRM.PersonF)); // function
18    console.log(typeof(HRM.PersonC)); // function
19
```

DOM – Document Object Model

Giới thiệu về DOM

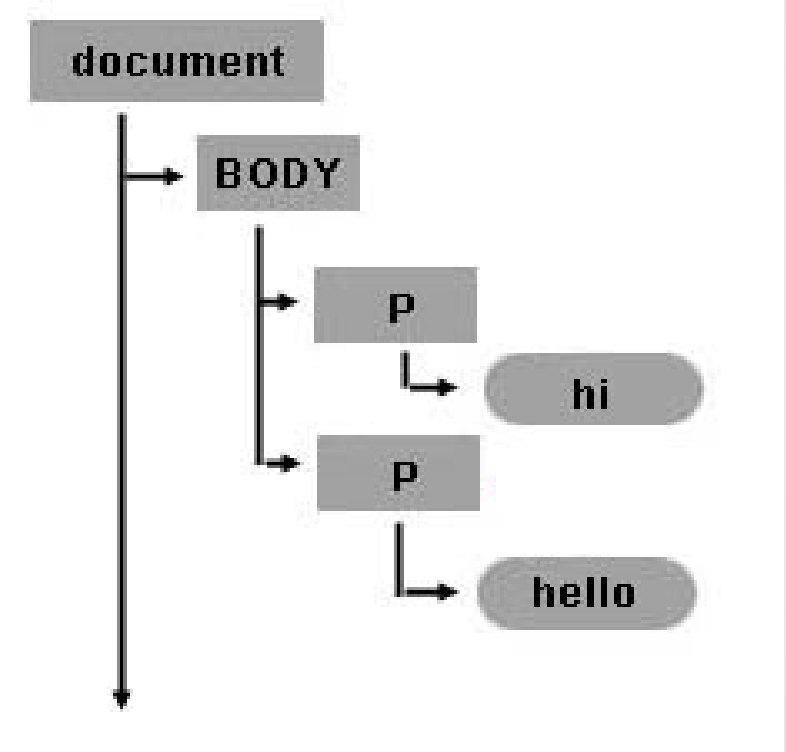
- Một trang web bao gồm một tập các đối tượng được tổ chức theo cấu trúc cây có gốc là đối tượng document
  - Đối tượng document đại diện cho chính nội dung trang web.

Giới thiệu về DOM

- HTML được dùng để khai báo các đối tượng (thuộc các lớp dựng sẵn)
- CSS được dùng để định nghĩa thuộc tính/kiểu trình diễn cho các đối tượng
- (java)script được dùng để quản lý (tạo, hủy bỏ, thay đổi thuộc tính, triệu gọi phương thức) các đối tượng, định nghĩa lớp mới.

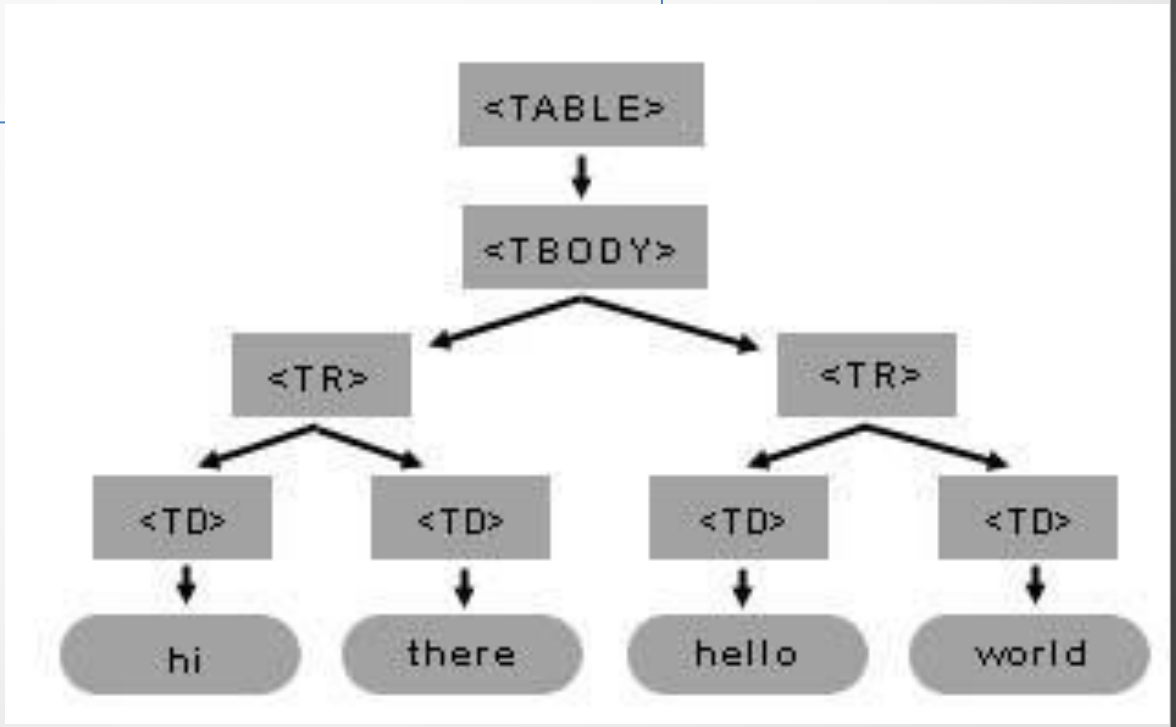
DOM – Ví dụ 1

```
<html>
        <body>
                <p>hi</p>
                <p>hello</p>
        </body>
</html>
```



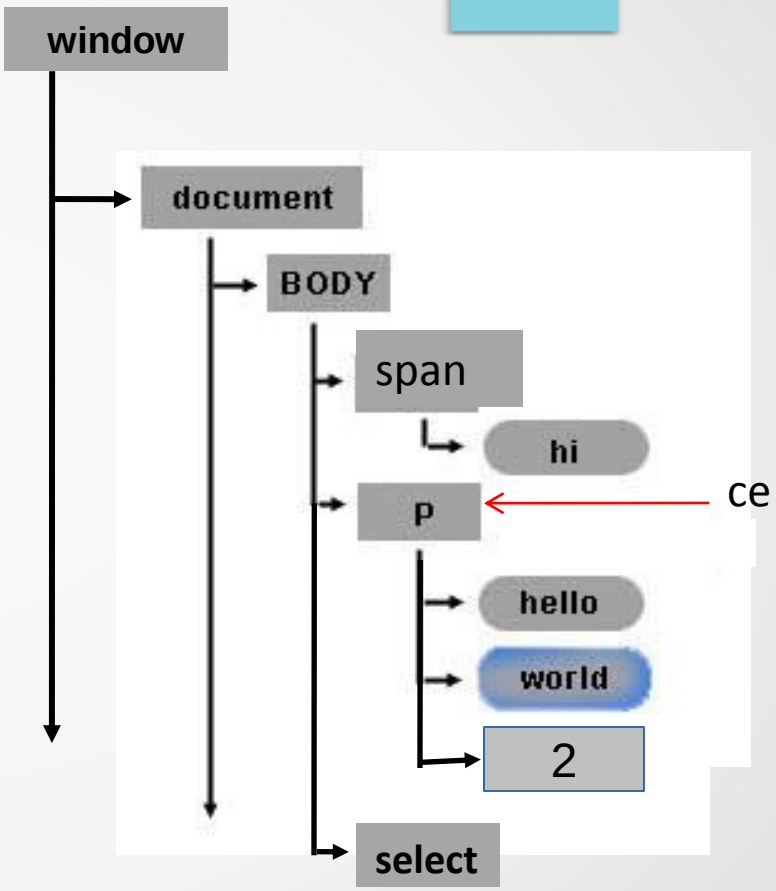
DOM – Ví dụ 2

```
<table border="2">
    <tbody><tr><td> hi </td><td> there </td></tr>
    <tr><td> hello </td><td> world
    </td></tr></tbody>
</table>
```



DOM – Ví dụ 3

- ce.parentNode == body
- ce.childNodes[0] == “hello”
- ce.childNodes[1] == “world”
- ce.childNodes[2] == “2”
- ce.firstChild == “hello”
- ce.lastChild == “2”
- ce.previousSibling == span
- ce.nextSibling == select
- ce.hasChildNodes() == true;



Tất cả đối tượng: Thuộc tính

Properties	Description
attributes[]	Returns an array (NamedNodeMap) containing all the attributes defined for the element in question, including custom attributes. IE6 returns not just attributes explicitly defined by the webmaster, but those of the element's internal DTD as well. In Firefox, attributes[] work more as expected, returning only user defined attributes, and even reflect changes done by scripting to an attribute. Each attribute[] element returned supports a name and value property to retrieve additional information about the attribute. **Example(s):** `var imageattributes=document.getElementById("myimage").attributes` `imageattributes[0].name //name of the first attribute of "myimage"` `imageattributes[0].value //value of the first attribute of "myimage"` `imageattributes.getNamedItem("src").value //value of the "src" property of "myimage"`

Tất cả đối tượng: Thuộc tính

childNodes[]	Returns an array of all of the child nodes of an element as objects. Use the properties "nodeName" and "nodeType" to retrieve additional information about a node. **Example(s):** `//access some <ul> element` `var mylist=document.getElementById("mylist")` `for (i=0; i<mylist.childNodes.length; i++){` `if (mylist.childNodes[i].nodeName=="LI")` `//do something` `}`
className	Returns the CSS class attribute of an element. Read/write. **Example(s):** `document.getElementById("test").className="class1" //Assign the class "class1" to element` `document.getElementById("test").className+=" class2" //Assign an additional "class2" class to element`
clientWidth	A cross browser (NS7+, IE4+) property that returns the viewable width of the content on the page, not including borders, margins, or scrollbars (overflowing content). **Example(s):** `var pagewidth=document.body.clientWidth`

Tất cả đối tượng: Thuộc tính

clientHeight	A cross browser (NS7+, IE4+) property that returns the viewable height of the content on the page, not including borders, margins, or scrollbars (overflowing content).
dir	Read/write property that returns the text direction of the element. Valid values are "ltr" (left to right) and "rtl" (right to left). Default is "ltr." **Example(s):** `document.getElementById("mydiv").dir="rtl" //change text direction`
firstChild	Returns a reference to the first child of an element.
id	Read/write property that reflects the ID attribute of an element. Use this property to access any element on the page easily.
innerHTML	A cross browser (non W3C DOM) property that lets you easily change the HTML contents of an element. Generally this property can only be invoked after the document has fully loaded. **Example(s):** `<p><b>Old paragraph text</b></p>` `<script type="text/javascript">` `window.onload=function(){` `document.getElementsByTagName("p")[0].innerHTML="<b>new paragraph text</b>"` `}` `</script>`

Tất cả đối tượng: Thuộc tính

lang	Read/write property that specifies the language of an element's attribute values and text content. Commonly invoked on the body level to determine the base language of the document.
lastChild	Returns a reference to the last child of an element.
localName	Returns the name of the node of an XML element. Equivilant to the nodeName property for regular HTML elements.
namespaceURI	Returns the URI string assigned to the xmlns attribute of an XML element.
nextSibling	Returns the next node following the current one. Returns null if there are none or for text nodes inside an element.
nodeName	Returns a string indicating the name of the node, in the case of elements, its tag name. Returned value is in uppercase. **Example(s):** `if (document.getElementById("test").firstChild.nodeName=="DIV")` `alert("This is a DIV")`
nodeType	Returns an integer indicating the type of a node. See chart for what each integer value represents. **Example(s):** `alert(document.getElementById("adiv").nodeType) //DIV element. Alerts 1`

Tất cả đối tượng: Thuộc tính

nodeValue	Read/write property that reflects the value of a node. For text nodes, the content of the node is returned, while for attribute nodes, the attribute value. Null is returned for Document and element nodes. Use this property to alter the contents of a text or attribute node. **Example(s):** `<body>` `<div id="test">Old text</div>` `<script type="text/javascript">` `if (document.getElementById("test").firstChild.nodeName=="#text")` `document.getElementById("test").firstChild.nodeValue="New text"` `</script>`
offsetLeft	Gets the horizontal offset position of the current element relative to its offset container. Read only. Use "offsetParent" to determine what an element's container is. Typically a positioned DIV or TABLE will act as an offset container to any element contained inside them.
offsetTop	Gets the vertical offset position of the current element relative to its offset container. Read only. Use "offsetParent" to determine what an element's container is.

Tất cả đối tượng: Thuộc tính

offsetParent	Returns the offset container of the current element. For most elements on the page, the "BODY" is their offset container. However, a positioned DIV for example creates its own offset container space. **Example(s):** `<body>` `<div id="master" style="position: relative">` `<div id="slave" style="position: relative">test</div>` `</div>` `<script type="text/javascript">` `alert(document.getElementById("slave").offsetParent.id) //alerts "master"` `</script>`
offsetWidth	A cross browser (non W3C DOM) property that returns the width of the element, including borders and padding if any, but not margins. In IE6, if a valid doctype is not specified, margins/padding are NOT included in the returned value.
offsetHeight	A cross browser (non W3C DOM) property that returns the height of the element, including borders and padding if any, but not margins. In IE6, if a valid doctype is not specified, margins/padding are NOT included in the returned value.
ownerDocument	Returns the document object that contains the current node.
parentNode	References the node that is the parent of the current one in the document tree.
prefix	Returns the namespace prefix of the current XML node, or `null` if not available.
previousSibling	Returns the previous node relative the current one in the document tree. Returns null if there are none or for text nodes inside an element.

Tất cả đối tượng: Thuộc tính

scrollLeft	A cross browser (NS7+, IE4+) property that returns the distance between the actual left edge of the element and its left edge currently in view. In a horizontally scrollable DIV for example, as you drag the scrollbar to the right, the DIV's scrollLeft property increases as the distance between its actual left edge and visible left edge increases. Applicable to scrollable elements, such as a DIV with scrollbars, a textarea, the BODY etc.
scrollTop	A cross browser (NS7+, IE4+) property that returns the distance between the actual top edge of the element and its top edge currently in view. Most commonly invoked on the BODY element for the purpose of positioning an element on the page so it's always visible. **Example(s):** `<div id="static" style="width:150px; height:150px; position: absolute; border:1px solid yellow; left: 5px;">Some text</div>` `<script type="text/javascript">` `//Keep "static" always in view on the page` `setInterval("document.getElementById` `('static').style.top=document.body.scrollTop+10+'px'", 50)` `</script>`
scrollHeight	A cross browser (NS7+, IE4+) property that returns the entire height of an element, including any area that may be hidden due to scrollbars. When the element does not contain vertical scrollbars, its scrollHeight is equal to its clientHeight.
scrollWidth	A cross browser (NS7+, IE4+) property that returns the entire width of an element, including any area that may be hidden due to scrollbars. When the element does not contain horizontal scrollbars, its scrollWidth is equal to its clientWidth.

Tất cả đối tượng: Thuộc tính

style	References the style object of an element, in turn accessing and modifying individual style attributes' values. **Example(s):** `document.getElementById("test").style.backgroundColor="yellow"`
tabIndex	Gets/sets the tab order of the current element.
tagName	Returns the name of the tag of an element as a string and in uppercase.
title	Read/write property that returns the title of the document or "title" attribute of an element when invoked on an element.

Tất cả đối tượng: Phương thức

DOM Element methods

Properties	Description
addEventListener(eventType, listener, useCapture)	Associates a function with a particular event and binds the event to the current node. **NS/Firefox only.** addEventListener() accepts the following 3 parameters: **1) EventType:** A string representing the event to bind, without the "on" prefix. For example, "click", "mousedown" etc. **2) listener:** The function or method to associate with the event. **3) useCapture:** Boolean indicating whether to bind the event as it is propogating towards the target node, (event Capture), or as the event bubbles upwards from the target (event bubble). Set to true or false, respectively. The advantage of using the DOM to bind an event is that you can assign multiple functions to a node for the same event (ie: window.onload) without running into event handler conflicts. **Example(s):** `function statusreport(){` `alert("document has loaded")` `}` `if (window.addEventListener)` `window.addEventListener("load", statusreport, false) //invoke function` `window.onload=statusreport() //function invoked again, since no event handler conflicts`

Tất cả đối tượng: Phương thức

attachEvent(eventType, function)	The IE5+ proprietary equivalent of addEventListener(). Note that for the parameter eventType, the even string should include the "on" prefix (ie: "onload", "onclick" etc). **Example(s):** `if (window.attachEvent)` `window.attachEvent("onload", statusreport) //invoke function`
appendChild(node)	Inserts the specified node at the end of the current node object. A frequently used method for dynamically appending a new element or text to the document. **Example(s):** `<div id="test"></div>` `<script type="text/javascript">` `var newdiv=document.createElement("div")` `var newtext=document.createTextNode("A new div")` `newdiv.appendChild(newtext) //append text to new div` `document.getElementById("test").appendChild(newdiv) //append new div to another div` `</script>`
blur()	Removes keyboard focus from the current element. Used for example to fire the onBlur event handler of an element via scripting.
click()	Executes a click on a element. Used for example to fire the onClick event handler of an element via scripting.

Tất cả đối tượng: Phương thức

cloneNode(deepBoolean)	Duplicates and returns a copy of the current node as a standalone node (not part of document tree). Cloning a node copies both the original's attributes and values, including the ID attribute, so be sure to alter the cloned ID attribute's value so it's unique before introducing it to the document tree. This method supports a single Boolean parameter, "deepBoolean" that when set to true, clones all the sub nodes of the current node as well, such as any text contained within. **Example(s):** `p=document.getElementById("mypara")` `pclone = p.cloneNode(true)`
detachEvent(eventType, function)	Removes an event handler and its function previously associated with a node in IE5+, via attachEvent() for example. The IE5+ proprietary equivalent of DOM2's removeEventListener(). **Example(s):** `if (window.detachEvent)` `window.detachEvent("onload", statusreport) //invoke function`

Tất cả đối tượng: Phương thức

dispatchEvent(eventObject)	Dispatches an event to fire on a node artificially. This method returns a Boolean indicating whether any of the listeners which handled the event called preventDefault (false if called, otherwise, true). IE's equivalent of dispatchEvent() is fireEvent(). **Example(s):** `<div id="test" onclick="alert('hi')">Sample DIV.</div>` `<script type="text/javascript">` `//Generate an artificial click event on "test". Fires alert("hi")` `var clickevent=document.createEvent("MouseEvents")` `clickevent.initEvent("click", true, true)` `document.getElementById("test").dispatchEvent(myevent)` `</script>`
focus()	Sets focus on the current node.
getAttribute(attributeName)	Returns the value of the attribute named attribute of the current node. **Example(s):** `document.getElementById("test").getAttribute("align")`
getAttributeNS(namespace, localname)	Returns the value of the attribute with the given local name and namespace. Applicable in XML documents.

Tất cả đối tượng: Phương thức

getAttributeNode (attributename)	Returns/references the attribute of the current element as a stand only node (not part of document tree). **Example(s):** `var attributeobj=document.getElementById("nav").getAttributeNode("align")` `attributeobj.value="center"`
getAttributeNodeNS (namespace, localname)	Returns/references the attribute of the current element with the given local name and namespace. Applicable in XML documents.
getElementsByTagName (tagName)	Returns as an array all the child elements of the current element matching the "tagName" parameter (ie: "li"). In Firefox/ IE6+, you may enter an asterisk ("*") for the method's parameter to retrieve a list of all elements within the current. **Example(s):** `var mylist=document.getElementById("navlist")` `var listitems= mylist.getElementsByTagName("li")` `for (i=0; i<listitems.length; i++)` `//manipulate each li element` `var alltags=document.getElementsByTagName("*") //returns all elements on page`
getElementsByTagNameNS (namespace, localname)	Returns as an array all the child elements of the current element with the given local name and namespace. Applicable in XML documents.

Tất cả đối tượng: Phương thức

hasAttribute(attributename)	Returns a Boolean value indicating whether the current element contains an attribute (ie: "align"). **Example(s):** `if (document.getElementById("mytable").hasAttribute("style"))` `//manipuate the element's style`
hasAttributeNS(namespace, localname)	Returns a Boolean value indicating whether the current element contains an attribute with the given local name and namespace. Applicable in XML documents.
hasAtrributes()	Returns a Boolean value indicating whether the current element has any explicit attributes defined.
hasChildNodes()	Returns a Boolean value indicating whether the current element contains any child nodes.

Tất cả đối tượng: Phương thức

insertBefore(newElement, targetElement)	Inserts a new node "newElement" as a child node of the current node. The "targetElement" property dictates where "newElement" is inserted within the list of child nodes. If set to null, the new element is inserted as the last child node; otherwise, it's inserted right before "targetElement". **Example(s):** `<div id="employees">` `<div id="george">George Doe: Human resources department</div>` `</div>` **To insert a new DIV directly above "george", so the outcome becomes:** `<div id="employees">` `<div id='kevin">Kevin Lin: Main system administrator</div>` `<div id="george">George Doe: Human resources department</div>` `</div>` **You would do the following:** `<script type="text/javascript">` `var newemployee=document.createElement("div")` `var oldemployee=document.getElementById("george")` `newemployee.setAttribute("id", "kevin")` `newemployee.innerHTML="Kevin Lin: Main system administrator"` `document.getElementById("employees").insertBefore(newemployee, oldemployee)` `</script>`
insertAfter(newElement, targetElement)	

Tất cả đối tượng: Phương thức

item(index)	Retrieves a node based on its index within the document tree. IE4+ and FireFox1+. **Example(s):** `<div id="div1"></div>` `<div id="div2"></div>` `<script type="text/javascript">` `var mydivs=document.getElementsByTagName("div")` `alert(mydivs.item(1).id) //alerts "div2"` `</script>`
normalize()	Normalizes the current node and its sub tree. See here for more info.
querySelector(selectors, [NSResolver]) *Note: Currently supported in FF3.1+, IE8+ (only in IE8 standards mode), and Safari 3.1+*	Accepts a CSS selector(s) and returns the first matching element (based on the document tree) within the invoking element, or `null`. **Example:** `<ul id="mylist">` `<li>Item 1</li>` `<li>Item 2</li>` `<li>Item 3</li>` `</ul>` `<script type="text/javascript">` `var item2=document.getElementById("mylist").querySelector('li:nth-of-type(2)')` `alert(item2.innerHTML) //alerts "Item 2"` `</script>`

Tất cả đối tượng: Phương thức

removeAttribute (attributename)	Removes an attribute by its name. **Example(s):** `document.getElementById("test").removeAttribute("href")`
removeAttributeNode (attributereference)	Remove an attribute by passing in as parameter a reference to the attribute object to remove. It offers an alternate way to removeAttribrute()"for removing attributes, when all you have is a reference to the attribute object in your script. **Example(s):** `var hrefattr=document.getElementById("test").getAttributeNode("href")` `document.getElementById("test").removeAttributeNode(hrefattr)`
removeAttributeNS (namespace, localname)	Removes an attribute with the specified namespace and localname.

Tất cả đối tượng: Phương thức

removeChild(childreference)	Removes the child node of the current node. The removed node can then be reinserted elsewhere in the document tree. **Example(s):** `<div id="father"><div id="child">A child</div></div>` `<script type="text/javascript">` `var childnode=document.getElementById("child")` `var removednode=document.getElementById("father").removeChild(childnode)` `</script>`
removeEventListener (eventType, listener, useCapture)	Removes the specified event from being binded to the current node: **1) EventType:** A string representing the event to unbind, without the "on" prefix. For example, "click", "mousedown" etc. **2) listener:** The function or method to associate with the event. **3) useCapture:** Boolean indicating whether to unbind the event as it is propagating towards the target node, (event Capture), or as the event bubbles upwards from the target (event bubble). Set to true or false, respectively. **NS6/Firefox method.**

Tất cả đối tượng: Phương thức

replaceChild(newChild, oldChild)	Replaces one child node of the current node with another child node. **Example(s):** `<div id="adiv"><span id="innerspan"></span></div>` `<script type="text/javascript">` `var oldel=document.getElementById("innerspan")` `var newel=document.createElement("p")` `document.getElementById("adiv").replaceChild(newel, oldel)` `</script>`
scrollIntoView([Boolean])	Firefox/IE4+ proprietary method that scrolls an element into view. It accepts an optional Boolean parameter that when set to true (default), scrolls the element so its top left corner touches the top of the viewable window. If false, the element's bottom left corner touches the bottom of the window.

Tất cả đối tượng: Phương thức

setAttribute(attributename, value, [iecaseflag])	Sets an attribute's value for the current element. If the attribute doesn't exit yet, it creates the attribute first. Otherwise, the existing attribute is modified with the new value. In **IE**, the following two pitfalls exist: • To set the "class" attribute, use "className" instead. • The "attributename" parameter is case sensitive by default in IE . This means if you attempt to set the "align" attribute and "Align" already exists on the element, both will be present as a result. To turn off case sensitivity, set the IE-only 2nd parameter of setAttribute() to 0 (instead of default, which is 1). **Example(s):** `document.getElementById("test").setAttribute("title", "JavaScript Kit")`
setAttributeNS(namespace, qualifiedname, value)	Sets or creates an attribute for the current node with the given local name and namespace. Applicable in XML documents.

Tất cả đối tượng: Phương thức

setAttributeNS(namespace, qualifiedname, value)	Sets or creates an attribute for the current node with the given local name and namespace. Applicable in XML documents.
setAttributeNode (attributereference)	Sets or creates an attribute for the current node. "attributereference" should be a reference to a attribute you wish to insert. If an attribute of the same name (as referenced) already exists on the node, it is replaced with the newly inserted one. **Example(s):** `<div id="brother" style="border:1px solid black; padding: 2px">Brother</div>` `<div id="sister">Sister</div>` `<script type="text/javascript">` `var bro=document.getElementById("brother")` `var sis=document.getElementById("sister")` `var brostyle=bro.getAttributeNode("style")` `var clonebrostyle=brostyle.cloneNode(false) //clone attribute first. Required.` `sis.setAttributeNode(clonebrostyle)` `</script>`
supports(feature, [version])	Tests to see if this DOM implementation supports a particular feature.

Tất cả đối tượng: Con trỏ sự kiện

- Con trỏ sự kiện
  - được gọi khi sự kiện tương ứng xảy ra trên đối tượng
- Một số con trỏ sự kiện
  - onClick – Kích chuột lên đối tượng
  - onDblClick – Kích đúp chuột lên đối tượng
  - onMouseOver – Di chuyển chuột trên đối tượng
  - onMouseOut – Di chuyển chuột ra ngoài đối tượng
  - onKeyUp – Nhả phím khi đối tượng đang được đặt làm tâm điểm
  - onKeyDown – Nhấn phím khi đối tượng đang được đặt tâm điểm

Tất cả đối tượng: Con trỏ sự kiện

- Ví dụ

```
<script type=”text/javascript”>
  function showUp() {
     alert(“Hello!”);
  }
</script>
<input type=”button” value=”Try it”
onclick=”javascript:showUp();”/>
```

Đối tượng document: Thuộc tính

Properties	Description
body	References the body element of the page. From there, you can then access other nodes contained within the body.
body.offsetWidth, body.offsetHeight	Returns the width and height of the entire document, respectively.
compatMode	Returns the compatibility mode of the current document, specifically, whether the page is rendered in Quirks or Stricts mode. The two possible values returned are "BackCompat" for Quirks and "CSS1Compat" for Strict. Useful for determining the doctype setting of the page and executing different code accordingly. **Example(s):** `if (document.compatMode=="CSS1Compat")` `// execute code for page with a valid doctype`
doctype	Read-only property that returns the Document Type Definition (DTD) of the current document, or null if the page doesn't contain a DTD. Not supported in IE as of IE6.
documentElement	References the root element of the document, in the case of HTML documents, the html element. This read only property is useful for accessing all elements on the page, such as the HEAD. **Example(s):** `entiredoc = document.documentElement;` `var docnodes=entiredoc.childNodes` `for (i=0; i<docnodes.length; i++)` `alert(docnodes[i].tagName)`

Đối tượng document: Thuộc tính

domain	Gets/sets the domain of the current document. Useful in cross domain scripting when one domain is to communicate with another. **Example(s):** `document.domain="javascriptkit.com"`
implementation	Returns the DOM implementation of the current document.
ownerDocument	Returns a reference to the document object that contains the current element/node. `var ownerdoc=document.getElementById("adiv").ownerDocument`
readyState	IE exclusive property that specifies the loading status of the document. It returns one of the below 4 values: 1) uninitialized- The document hasn't started loading yet. 2) loading- The document is loading. 3) interactive- The document has loaded enough whereby user can interact with it. 4) complete- The document has fully loaded.
styleSheets[]	An array referencing all stylesheet objects on the page, whether they are defined using the <style> or <link> tag.
title	Specifies the title of the document. Read/write in modern browsers.
URL	A string that specifies the complete URL of the document.

Đối tượng document: Phương thức

Methods	Description
createAttribute ("attributename")	Creates a new attribute, ready to be inserted somewhere in the document. It returns a reference to the created attribute. **Example(s):** `var styleattr=document.createAttribute("align")` `styleattr.nodeValue="center"` `document.getElementById("sister").setAttributeNode(styleattr)`
createComment (commenttext)	Creates an instance of the comment node. Once created, you can then insert it into the document tree using appendChild(), for example. **Example(s):** `var commentnode=document.createComment("Copyright JavaScript Kit")` `document.getElementById("mydiv").appendChild(commentnode)`

Đối tượng document: Phương thức

createDocumentFragment()	Creates an empty document fragment. The result is a temporary container for creating and modifying new elements or attributes before introducing the final result to your document tree. This is a very useful method when you're performing multiple operations that add to or modify the document tree. Instead of directly modifying the document tree each time (very inefficient), it's much better to use a temporary "whiteboard" that is created by createDocumentFragment() to perform all your operations on first before finally inserting the result to the document tree. **Example(s):** `<div id="sister"></div>` `<script type="text/javascript">` `var docfrag=document.createDocumentFragment()` `var mydiv=document.createElement("div")` `var divtext=document.createTextNode("This is div text")` `mydiv.appendChild(divtext)` `docfrag.appendChild(mydiv)` `document.getElementById("sister").appendChild(docfrag) //div now reads "this is div text"` `</script>`

Đối tượng document: Phương thức

createDocumentFragment()	Creates an empty document fragment. The result is a temporary container for creating and modifying new elements or attributes before introducing the final result to your document tree. This is a very useful method when you're performing multiple operations that add to or modify the document tree. Instead of directly modifying the document tree each time (very inefficient), it's much better to use a temporary "whiteboard" that is created by createDocumentFragment() to perform all your operations on first before finally inserting the result to the document tree. **Example(s):** `<div id="sister"></div>` `<script type="text/javascript">` `var docfrag=document.createDocumentFragment()` `var mydiv=document.createElement("div")` `var divtext=document.createTextNode("This is div text")` `mydiv.appendChild(divtext)` `docfrag.appendChild(mydiv)` `document.getElementById("sister").appendChild(docfrag) //div now reads "this is div text"` `</script>`

Đối tượng document: Phương thức

createElement(tagName)	Creates an instance of the element object, which can then added to the document tree using appendChild(), for example. **Example(s):** `var textblock=document.createElement("p")` `textblock.setAttribute("id", "george")` `textblock.setAttribute("align", "center")` `document.body.appendChild(textblock)`
createTextNode(text)	Creates a new text node, which can then be added to an element in the document tree. **Example(s):** `var headertext=document.createTextNode("Welcome to JavaScript Kit")` `document.getElementById("mytitle").appendChild(headertext)`
getElementById(id)	Accesses any element on the page via its ID attribute. A fundamental method within the DOM for accessing elements on the page.

Đối tượng document: Phương thức

getElementsByName(name)	Returns an array of elements with a name attribute whose value matches that of the parameter's. In IE6, elements with an ID attribute of the matching value will also be included in the array, and getElementsByName() is limited to retrieving form objects such as checkboxes and INPUT. In Firefox, nither of these "pitfalls" apply. `<div name="george">f</div>` `<div name="george">f</div>` `<script type="text/javascript">` `var georges=document.getElementsByName("george")` `for (i=0; i< georges.length; i++)` `// do something with each DIV tag with name="george". Firefox only.` `</script>`
getElementsByTagName (tagname)	Returns an array of elements whose tag name matches the parameter. In Firefox/ IE6+, you may enter an asterisk ("*") as the parameter to retrieve a list of all elements within the document. `var ptags=document.getElementsByTagName("p")` `var alltags=document.getElementsByTagName("*") //returns all elements on page`

Đối tượng document: Phương thức

querySelector(selectors, [NSResolver]) *Note: Currently supported in FF3.1+, IE8+ (only in IE8 standards mode), and Safari 3.1+*	Accepts a CSS selector(s) and returns the first matching element (based on the document tree) within the document, or `null`. **Example:** `<ul id="mylist">` `<li>Item 1</li>` `<li>Item 2</li>` `<li>Item 3</li>` `</ul>` `<script type="text/javascript">` `var item2=document.querySelector('#mylist li:nth-of-type(2)')` `alert(item2.innerHTML) //alerts "Item 2"` `</script>` You can enter multiple CSS selectors each separated by a comma (,), in which case the first matching element of any of the CSS selectors entered will be returned: `//returns either element "#leftcolumn" or "#rightcolumn", depending on which one is found first:` `document.querySelector("#leftcolumn, #rightcolumn")` querySelector() supports an optional 2nd "`NSResolver`" parameter to resolve namespace prefixes in XHTML documents. Not supported in IE8.

Đối tượng document: Phương thức

querySelectorAll(selectors, [NSResolver])

Note: Currently supported in FF3.1+, IE8+ (only in IE8 standards mode), and Safari 3.1+

Accepts a CSS selector(s) and returns all matching elements (based on the document tree) within the document as a `staticNodeList`, or `null`. A `staticNodeList` is a static collection of elements that are not affected by any subsequent changes occuring on the document tree.

Example:

```
<ul id="mylist">
<li>Item 1</li>
<li>Item 2</li>
<li>Item 3</li>
</ul>

<script type="text/javascript">
var odditems=document.querySelectorAll('#mylist li:nth-of-type(odd)')
for (var i=0; i<odditems.length; i++)
 alert(odditems[i].innerHTML) //alerts "Item 1", "Item 3" etc.
</script>
```

You can enter multiple CSS selectors each separated by a comma (,), in which case **all matching elements** found using the entered CSS selectors will be returned:

```
//returns both elements "#leftcolumn" or "#rightcolumn", or one of them
if only one defined:
document.querySelectorAll("#leftcolumn, #rightcolumn")
```

`querySelectorAll()` supports an optional 2nd "`NSResolver`" parameter to resolve namespace prefixes in XHTML documents. Not supported in IE8.

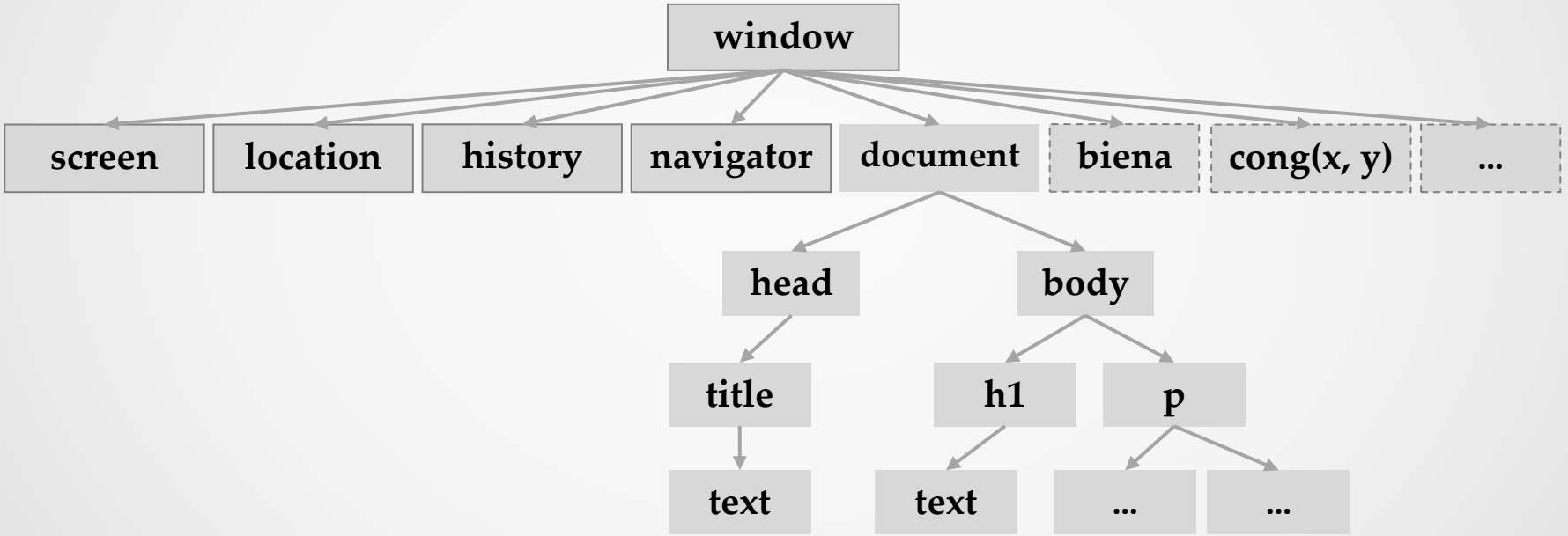
See "Overview of CSS3 Structural puesdo-classes" for advanced CSS selectors you can use with the query selector methods.

BOM – Browser Object Model

Giới thiệu về BOM

- Cho phép Javascript tương tác với trình duyệt.
- Không có chuẩn nhưng các trình duyệt cài đặt BOM như nhau.
- Gốc của cây BOM là đối tượng window.
  - Đối tượng window đại diện cho cửa sổ/khung hiển thị trang web

BOM với DOM và các hàm, biến toàn cục



Đối tượng window

- Các hàm, biến toàn cục được khai báo trở thành phương thức, thuộc tính của đối tượng window.
- Có thể truy cập các phương thức và thuộc tính của đối tượng window mà không cần tiền tố *window.*
- Ví dụ:

alert("some text"); tương đương
window.alert("some text");
document tương đương **window.document**

Đối tượng window: Thuộc tính

innerWidth, innerHeight	Read/write property that specifies the width and height, in pixels, of the window's content area respectively. Does not include the toolbar, scrollbars etc. **NS/Firefox exclusive properties.** **Note:** IE equivalents are "document.body.clientWidth" and "document.body.clientHeight"
length	Returns the number of frames contained in the window.
outerWidth, outerHeight	Read/write property that specifies the total width and height, in pixels, of the window's content area respectively, including any toolbar, scrollbars etc. **NS/Firefox exclusive properties** with no IE4+ equivalent.
pageXOffset, pageYOffset	Returns an integer representing the pixels the current document has been scrolled from the upper left corner of the window, horizontally and vertically, respectively. Typically used to provide the needed calculations to keep an element in view even when the page is scrolled. **NS/Firefox exclusive properties.** **Note:** IE equivalents are "document.body.scrollLeft" and "document.body.scrollTop"
window.screen	References the screen object, which provides information about the user's screen/ monitor.
screen.availWidth	Returns the height of the screen, in pixels, minus interface features such as the taskbar in Windows. In other words, the usable height available to your browser window.
screen.availHeight	Returns the width of the screen, in pixels, minus interface features such as the taskbar in Windows. In other words, the usable width available to your browser window.
screen.colorDepth	The bit depth of the color palette available for displaying images in bits per pixel.

Đối tượng window: Thuộc tính

screen.height	The total height of the screen, in pixels.
screen.pixelDepth	Display screen color resolution (bits per pixel). **NS/Firefox exclusive property.**
screen.width	The total width of the screen, in pixels.
screenX, screenY	Read/write property that specifies the x and y coordinates of the window relative to the user's monitor screen. **NS/Firefox exclusive properties.**
screenLeft, screenTop	Specifies the x and y coordinates of the window relative to the user's monitor screen. **IE only.**
scrollX, scrollY	Returns an integer representing the pixels the current document has been scrolled from the upper left corner of the window, horizontally and vertically, respectively. **NS/Firefox exclusive properties.** Equivalent to pageXOffset and pageYOffset, and in IE, "document.body.scrollLeft" and "document.body.scrollTop"

Đối tượng window: Phương thức

Methods	Description
addEventListener(eventType, listener, useCapture)	Associates a function with a particular event and binds the event to the current node. **NS/Firefox method.** addEventListener() accepts the following 3 parameters: **1) EventType:** A string representing the event to bind, without the "on" prefix. For example, "click", "mousedown" etc. **2) listener:** The function or method to associate with the event. **3) useCapture:** Boolean indicating whether to bind the event as it is propagating towards the target node, (event Capture), or as the event bubbles upwards from the target (event bubble). Set to true or false, respectively. The advantage of using the DOM to bind an event is that you can assign multiple functions to a node for the same event (ie: window.onload) without running into event handler conflicts. **Example(s):** `function statusreport(){` `alert("document has loaded")` `}` `if (window.addEventListener)` `window.addEventListener("load", statusreport, false) //invoke function` `window.onload=statusreport() //function invoked again, since no event handler conflicts`

Đối tượng window: Phương thức

attachEvent(eventType, function)	**IE5+ proprietary** equivalent of addEventListener(). For the parameter eventType, the event string should include the "on" prefix (ie: "onload", "ondick" etc). **Example(s):** `if (window.attachEvent)` `window.attachEvent("onload", statusreport) //invoke function`
detachEvent(eventType, function)	Removes an event handler and its function previously associated with a node in IE5+, via attachEvent() for example. The **IE5+ proprietary equivalent** of DOM2's removeEventListener (). **Example(s):** `if (window.detachEvent)` `window.detachEvent("onload", statusreport) //invoke function`

Đối tượng window: Phương thức

dispatchEvent(eventObject)	Dispatches an event to fire on a node artificially. **NS/Firefox method.** This method returns a Boolean indicating whether any of the listeners which handled the event called preventDefault (false if called, otherwise, true). IE's equivalent of dispatchEvent() is fireEvent (). **Example(s):** `<div id="test" onclick="alert('hi')">Sample DIV.</div>` `<script type="text/javascript">` `//Generate an artificial click event on "test". Fires alert("hi")` `var clickevent=document.createEvent("MouseEvents")` `clickevent.initEvent("click", true, true)` `document.getElementById("test").dispatchEvent(myevent)` `</script>`
getComputedStyle (elementRef, pseudoElementName)	**NS/Firefox only** method that returns the style object for an element with its current active style settings, whether they're set using external/global or inline CSS. Use styleref.getPropertyValue() in conjunction to retrieve the value of a CSS attribute regardless of how it was set. The following shows how to get the "background-color" property value of a DIV set using global CSS: `<script type="text/javascript">` `var mydiv=document.getElementById("test")` `var mydivstyle=window.getComputedStyle(mydiv, "")` `var divbgcolor=mydivstyle.getPropertyValue("background-color") //contains "yellow"` `</script>`

Đối tượng window: Phương thức

releaseEvents(eventType)	Releases the window object from trapping events of a specific type: `window.releaseEvents(Event.KEYPRESS) //single event` `window.releaseEvents(Event.KEYPRESS | Event.KEYDOWN | Event.KEYUP) //multiple events`
removeEventListener (eventType, listener, useCapture)	Removes the specified event from being binded to the current node: **1) EventType:** A string representing the event to unbind, without the "on" prefix. For example, "click", "mousedown" etc. **2) listener:** The function or method to associate with the event. **3) useCapture:** Boolean indicating whether to unbind the event as it is propagating towards the target node, (event Capture), or as the event bubbles upwards from the target (event bubble). Set to true or false, respectively. **NS6/Firefox method.**
resizeBy(dx, dy)	Resizes a window by the specified amount in pixels.
resizeTo(x y)	Resizes a window to the specified pixel values.
scrollBy(dx, dy)	Scrolls a window by the specified amount in pixels.
scrollByLines(lines)	Scrolls the document by the number of lines entered as the parameter. **NS/Firefox method.**
scrollByPages(pages)	Scrolls the document by the number of pages entered as the parameter. **NS/Firefox method.**
scrollTo(x, y)	Scrolls a window to the specified pixel values.
sizeToContent()	Sizes the window to fit the content contained within. Useful, for example, with popup windows that contain small amounts of content. **NS6/Firefox method.**

Đối tượng window.screen

- Mang thông tin về màn hình thiết bị
- Thuộc tính
  - *width*
  - *height*
  - *availWidth*
  - *availHeight*
  - *colorDepth*

Đối tượng window.location

- Cho biết URL của trang hiện tại và có thể được sử dụng để chuyển hướng
- Thuộc tính
  - *href*
  - *hostname*
  - *pathname*
  - *protocol*
- Phương thức
  - *assign(url)*

Đối tượng window.history

- Lưu và chuyển lịch sử duyệt của cửa sổ
- Phương thức
  - *back()*
  - *forward()*

Đối tượng window.navigator

- Mang thông tin về trình duyệt
- Thuộc tính
  - *appName*
  - *appCodeName*
  - *appVersion*
  - *cookieEnabled*

Quản lý trang web

Tác vụ chung

- Lấy tham chiếu các đối tượng tài liệu
- Đọc và thay đổi thuộc tính các đối tượng tài liệu
- Thay đổi kiểu trình diễn đối tượng tài liệu
- Xử lý sự kiện trên đối tượng tài liệu
- Thêm mới, loại bỏ đối tượng tài liệu
- Mở cửa sổ mới và tương tác giữa các đối tượng thuộc các cửa sổ khác nhau
- Hộp thoại, in ấn

Lấy tham chiếu các đối tượng tài liệu

- *document.**getElementById**(v)*
- *document.**getElementsByName**(v)*
- *document.**getElementsByTagName**(v)*
- *document.**querySelector**(v)*
- *document.**querySelectorAll**(v)*
- *obj.parentNode*
- *obj.childNodes*,
- *obj.previousSibling*
- *obj.nextSibling*

Lấy tham chiếu các đối tượng tài liệu

```
<ul id=”mylist”>
        <li>Item 1</li>
        <li>Item 2</li>
        <li>Item 3</li>
</ul>
<script type=”text/javascript”>
        var list = document.getElementById(“mylist”);
        alert(list.innerHTML); // <li>Item 1</li><li>Item 2</li><li>Item 3</li>
        var oddi = document.querySelectorAll(“#mylist li:nth-of-type(odd)”);
        for (var i = 0; i < oddi.length; i++)
                console.log(oddi[i].innerHTML); //Item 1   Item 3
</script>
```

Đọc và thay đổi thuộc tính đối tượng tài liệu

- *obj.innerHTML*
- *obj.attributes*
- *obj.value*
- *obj.checked*

Đọc và thay đổi thuộc tính đối tượng tài liệu

```
<label>Họ và tên:</label>
<input type=”text” id=”fullname”/>
<span id=”fullnameErr”></span>
<script type=”text/javascript”>
        var fn = document.getElementById(“fullname”);
        var fnerr = document.getElementById(“fullnameErr”);
        if (fn.value == “”) fnerr.innerHTML = “Chưa nhập họ tên”;


        var attrs = document.getElementById(“fullname”).attributes;
        for (var i = 0; i < attrs.length; i++) {
                console.log(attrs[i].name + ": " + attrs[i].value);
        }
</script>
```

Thay đổi CSS của đối tượng tài liệu

- *obj.style*
- *obj.className*
- *obj.classList*
- *obj.classList.add(v1, v2, …)*
- *obj.classList.remove(v1, v2, …)*
- *obj.classList. toggle(v, true|false)*
- *obj.classList.contains(v)*

Thay đổi CSS của đối tượng tài liệu

```
<!DOCTYPE html><html><head>
        <style type=”text/css”>
                .err {color:red;}
                .required {font-weight:bold;}
                .critical {font-size:x-large;}
                .highlight {border: solid 1px green;}
        </style>
</head><body>
        <label>Họ và tên:</label>
        <input type="text" id="fullname"/>
        <span id="fullnameErr"></span>
        <script type="text/javascript">
                var fn = document.getElementById("fullname");
                var fnerr = document.getElementById("fullnameErr");
                if (fn.value == "") {
                        fnerr.innerHTML = "Chưa nhập họ tên";
                        fnerr.style.backgroundColor = "yellow";
                        fnerr.className = "err required";
                        fnerr.classList.add("critical", "highlight");
                }
        </script>
</body></html>
```

Xử lý sự kiện trên đối tượng tài liệu

- *obj.onclick*
- *obj.ondblclick*
- *obj.onmouseover*
- *obj.onmouseout*
- *obj.onfocus*
- *obj.onblur*
- *obj.onkeydown*
- *obj.onkeyup*

Xử lý sự kiện trên đối tượng tài liệu

```
<!DOCTYPE html><html><head>
        <style type="text/css"> .err {color:red;} </style>
</head><body>
        <label>Họ và tên:</label>
        <input type="text" id="fullname"/>
        <span id="fullnameErr" class="err"></span>
        <br/>
        <button onclick="checkInput();">Chấp nhận</button>
        <button id="btncancel">Bỏ qua</button>
        <script type="text/javascript">
                function checkInput() {
                        var fn = document.getElementById("fullname");
                        var fnerr = document.getElementById("fullnameErr");
                        if (fn.value == "") {
                                fnerr.innerHTML = "Chưa nhập họ tên";
                        } else {
                                //đệ trình ở đây
                        }
                }
                document.getElementById("btncancel").onclick = function() {
                        window.location = "Page2.htm";
                }
```

Xử lý sự kiện trên đối tượng tài liệu

```
<!DOCTYPE html><html><head>
        <title>Event and This</title>
        <meta charset="utf-8">
        <meta http-equiv="Content-Type" content="text/html;"/>
</head><body>
        <label>Họ và tên:</label>
        <input type="text" id="fullname"/>
        <br/>
        <label>Địa chỉ:</label>
        <input type="text" id="address"/>

        <script type="text/javascript">
                function checkInput(ctrl) { return (ctrl.value == "" ?  false : true);   }
                document.getElementById("fullname").onkeyup = function(e) {
                        if (!checkInput(this)) this.style.backgroundColor = "white";
                        else this.style.backgroundColor = "yellow";

                        var kc = window.event ? window.event.keyCode : e.keyCode;
                        if (kc == 13) {//Enter
                                document.getElementById("address").focus();
                        }
                };
        </script>
</body></html>
```

Kích hoạt sự kiện trên đối tượng tài liệu

- *obj.click()*
- *obj.dblclick()*
- *obj.mouseover()*
- *obj.mouseout()*
- *obj.focus()*
- *obj.blur()*
- *obj.keydown()*
- *obj.keyup()*

Kích hoạt sự kiện trên đối tượng tài liệu

```
<!DOCTYPE html><html><head>
        <title>Event Triger</title>
</head><body>
        <label>Họ và tên:</label>
        <input type="text" id="fullname"/>
        <span id="fullnameErr"></span>

        <script type="text/javascript">
                document.getElementById("fullname").focus();
        </script>
</body></html>
```

Thêm mới, loại bỏ đối tượng tài liệu

- *obj.innerHTML*
- *document.createElement(tagname)*
- *document.createTextNode(txt)*
- *document.createComment(txt)*
- *document.createDocumentFragment()*
- *obj.appendChild(node)*
- *obj.removeChild(node)*
- *obj.replaceChild(newN, oldN)*
- *obj.insertBefore(newN, targetN)*
- *obj.insertAfter(newN, targetN)*
- *obj.hasChildNodes()*

Thêm mới, loại bỏ đối tượng tài liệu

```
<!DOCTYPE html><html><head>
        <title>HTML DOM</title>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
</head><body>
        <h1>Phần 1</h1>
        <div id="container1"><p>Đoạn văn thứ nhất</p><p>Đoạn văn thứ hai</p></div>
        <h1>Phần 2</h1>
        <div id="container2"><p>Đoạn văn sẽ được thay thế</p></div>

        <script>
                var p3 = document.createElement("p");
                var p4 = document.createElement("p");
                var p5 = document.createElement("p");
                var txt3 = document.createTextNode("Đoạn văn thứ ba");
                var txt4 = document.createTextNode("Đoạn văn thứ tư");
                var txt5 = document.createTextNode("Đoạn văn thứ năm");
                p3.appendChild(txt3);
                p4.appendChild(txt4);
                p5.appendChild(txt5);

                var div1 = document.getElementById("container1");
                div1.appendChild(p4);
                div1.insertBefore(p3, p4);

                var div2 = document.getElementById("container2");
                var rp = document.querySelector("#container2 p");
                div2.replaceChild(p5, rp);
```

Tương tác giữa các đối tượng trong và ngoài iframe

- iframe là một nút thuộc cây DOM bên ngoài
- iframe có cửa sổ riêng của nó là contentWindow

- *window (trang ngoài)*
  - *document (trang ngoài)*
    - *ifr (trang ngoài)*
      - contentWindow (trang trong)
        - document (trang trong)

Tương tác giữa các đối tượng trong và ngoài iframe

```
<!—OutterPage.htm -->
<!DOCTYPE html><html><head><title>Outer Page</title></head><body>
        <input id="txt" type="text">
        <iframe id="ifr" src="innerPage.htm"></iframe>
        <script>
                document.getElementById("txt").onkeyup = function() {
                        var iwnd = document.getElementById("ifr").contentWindow;
                        iwnd.document.getElementById("txt").value = this.value;
                };
        </script>
</body></html>

<!—InnerPage.htm -->
<!DOCTYPE html><html><head><title>Inner Page</title></head><body>
        <input id="txt" type="text">
        <script>
                document.getElementById("txt").onkeyup = function() {
                        parent.document.getElementById("txt").value = this.value;
                };
        </script>
</body></html>
```

Tương tác giữa các đối tượng thuộc cửa sổ cha và con

- Mở một cửa sổ mới, trả về tham chiếu của cửa sổ được mở
  - *childWnd = window.open(url, …);*
- *window.opener* của cửa sổ con tham chiếu đến window của cửa sổ cha

Tương tác giữa các đối tượng thuộc cửa sổ cha và con

```
<!—OpeningPage.htm -->
<!DOCTYPE html><html><head><title>Opening Page</title></head><body>
        <input id="txt" type="text">
        <script>
                var cwnd = window.open("openedPage.htm", "", "");
                document.getElementById("txt").onkeyup = function() {
                        cwnd.document.getElementById("txt").value = this.value;
                };
        </script>
</body></html>

<!—OpenedPage.htm -->
<!DOCTYPE html><html><head><title>Opened Page</title></head><body>
        <input id="txt" type="text">
        <script>
                document.getElementById("txt").onkeyup = function() {
                        opener.document.getElementById("txt").value = this.value;
                };
        </script>
</body></html>
```

SOP

- Cùng nguồn gốc
  - URL của hai trang có cùng lược đồ, tên miền, và số hiệu cổng.
- **Chính sách cùng nguồn gốc**, viết tắt là **SOP** (Same origin policy).
  - Chỉ cho phép kịch bản ở trang này truy cập dữ liệu ở trang khác nếu hai trang có cùng nguồn gốc.

Hộp thoại, in ấn

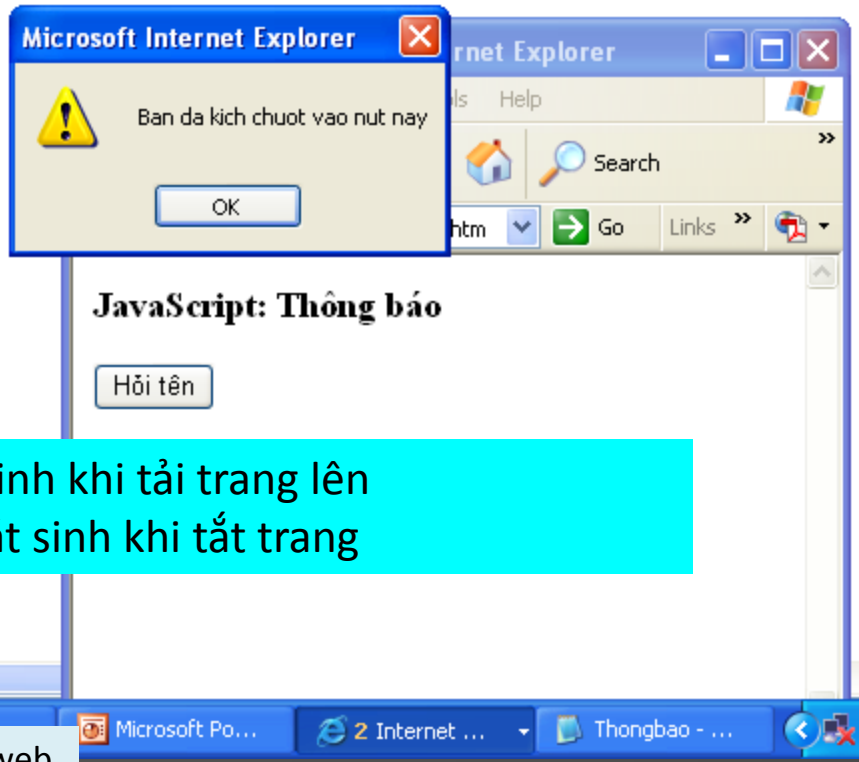
- *window.alert(msg)*
- *window.confirm(msg)*
- *window.print()*

Ví dụ: Hiển thị thông báo

Thongbao - Notepad

File Edit Format View Help

```
<html>
    <head>
        <title>Trang thứ 2</title>
    </head>
    <body onload = "alert('Xin chao')" onUnload = "alert('Tam biet')">
        <h3>JavaScript: Thông báo</h3>
        <form method = "POST">
            <input type = "button" value= "Hỏi tên" onclick = "alert('Ban da kich chuot vao n
        </form>
    </body>
</html>
```



Onload: Sự kiện được phát sinh khi tải trang lên

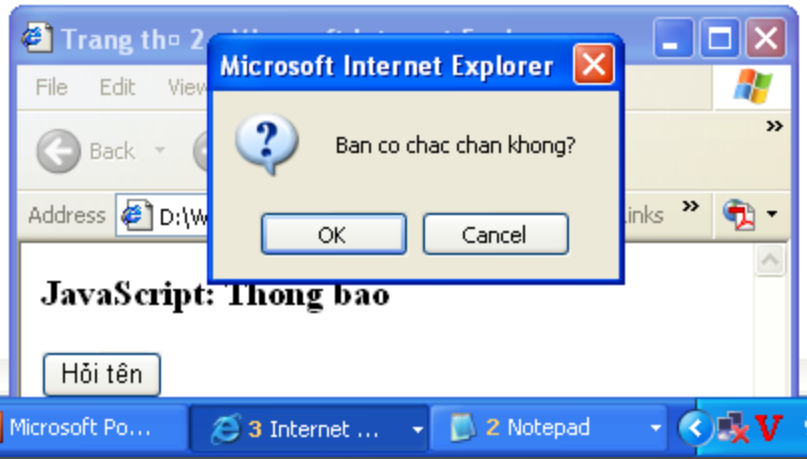
OnUnload: Sự kiện được phát sinh khi tắt trang

Ví dụ: Thông báo hỏi sự đồng ý

Hoi chap nhan - Notepad

File Edit Format View Help

```
<html>
    <head>
        <title>Trang thứ 2</title>
    </head>
        <script language = "JavaScript">
            function Hoi()
            {
                if (confirm("Ban co chac chan khong?")) alert("Ban da chap nhan");
                else alert("Ban da tu choi");
            }
        </script>
    <body>
        <h3>JavaScript: Thông báo</h3>
        <form method = "POST">
            <input type = "button" value= "Hỏi tên" onclick = "javascript:Hoi();">
        </form>
    </body>
</html>
```

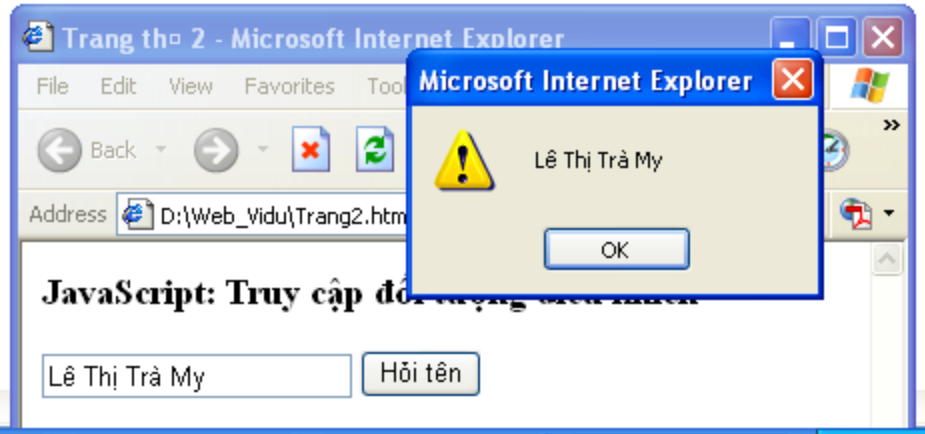


Ví dụ: Truy cập điều khiển text

Trang2.htm - Notepad

File Edit Format View Help

```
<html>
    <head>
        <title>Trang thứ 2</title>
        <script language = "JavaScript">
            function HienthiTen()
            {
                var ht = document.getElementById("hoten").value;
                alert(ht);
            }
        </script>
    </head>
    <body>
        <h3>JavaScript: Truy cập đối tượng điều khiển</h3>
        <form method = "POST">
            <input type = "text" id = "hoten" width = "20">
            <input type = "button" value= "Hỏi tên" onclick = "javascript:HienthiTen();">
        </form>
    </body>
</html>
```



Ví dụ: Truy cập bảng

Truy cap bang - Notepad

File Edit Format View Help

```
<html>
        <head>


                <script language = "javascript">
                        function sua()
                        {
                                //Neu xoa mot hang thi dung cau nay
                        //      document.getElementById("Table1").deleteRow(0);

                                //Neu muon them hang va o
                        //      r = document.getElementById("Table1").insertRow(document.getElementById("
                        //      c = r.insertCell(0);
                        //      c.innerHTML = "vc fff";
                        //      c.bgColor = "#ffffff";

                                //Thay doi noi dung o
                                r = document.getElementById("Table1").rows[0];
                                r.cells[0].innerHTML = "abc";
                                r.cells[1].innerHTML = "<a href = ">cba</a>";
                        }

                </script>
        </head>
        <body>
                <table id = "Table1" border = 1>
```

start Micr... We... Micr... Tru... 11:24 PM

Ví dụ: Truy cập bảng

```
<script language="javascript">
function dechuot(i) {
        document.getElementById("table1").rows[i].cells[0].background="cuoc
    song.gif" ;
}

function chuotRa(i) {
        document.getElementById("table1").rows[i].cells[0].background="cuoc
    song1.gif" ;
}
</script>

<td  style="cursor:pointer" onMouseOver="javascript:dechuot(<%=i%>);"
    onMouseOut="javascript:chuotRa(<%=i%>);" background="cuocsong1.gif"
    align="center" bgcolor="#CCCCFF" onClick="javascript:Chuyentrang(i);“
</td>
```

Ví dụ: Mở cửa sổ mới

Cua so moi - Nhan gia tri - Notepad

File Edit Format View Help

```
<html>
    <head>
        <title>Trang thu 2</title>
        <script language = "JavaScript">

            function moTrang()
            {
                window.open("Cuaso.htm", "df", "width = 300 height = 400");
            }
        </script>
    </head>
    <body>
        <form>
            <input type  = "text" id = hoten width = "200" disabled>
            <input type = "button" onclick = "moTrang();" value = "Trang 1">
        </form>
    </body>
</html>
```

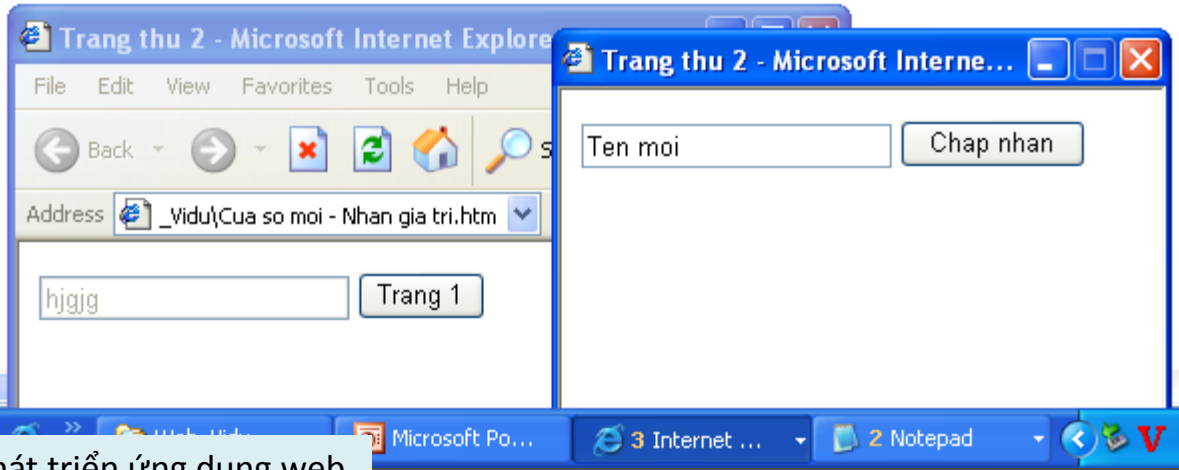
Ví dụ: Mở cửa sổ mới – Nhập tham số

Cuaso - Notepad

File Edit Format View Help

```
<html>
    <head>
        <title>Trang thu 2</title>
        <script language = "JavaScript">

            function nhap()
            {
                opener.document.forms[0].hoten.value = document.getElementById("h
                window.close();
            }
        </script>
    </head>
    <body>
        <input type  = "text" id = ht width = "200">
        <input type = "button" onclick = "nhap();" value = "Chap nhan">
    </body>
</html>
```



Ví dụ: Mở cửa sổ con

EditPlus - [C:\lect\LT Web\Web_tinh\dialog.html]

File Edit View Search Document Project Tools Browser Window Help

```
1  <script type="text/javascript">
2      function inputName()
3      {
4          ht = window.showModalDialog("Nhapten.aspx", "Nhapten", "clientWidth:100px; clientHeight:200px
5          alert(ht);
6      }
7  </script>
8
9
10 <a href="javascript:inputName();">Nhap ten</a>
11
12
13
14
```

Cửa sổ con trả về giá trị bằng **returnValue**

dialog.html

For Help, press F1 — ln 4 | col 36 | 14 | 28 | UNIX | UTF

start | 2 Windows Explorer | Microsoft PowerPoint ... | EditPlus - [C:\lect\LT ... | Source of: http://tint... | (2:39)

Ví dụ: Thay nội dung frame

Menu - Notepad

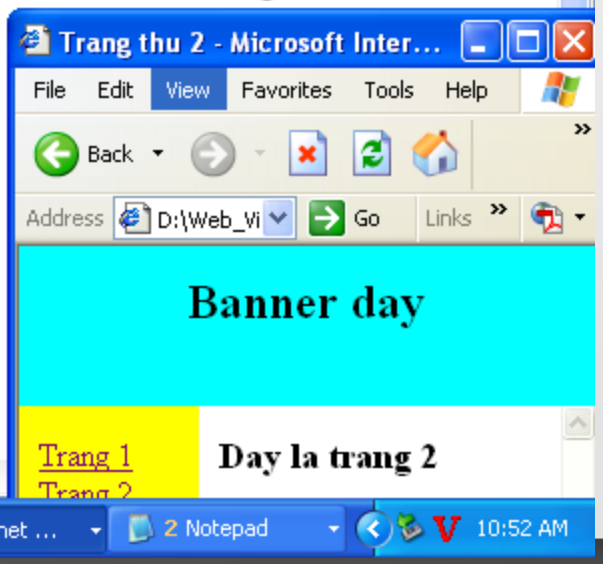
File Edit Format View Help

```
<html>
    <head>
        <title>Trang thứ 2</title>
        <script language = "JavaScript">

                function thayTrang(contentID)
                {
                        if (contentID == 1)
                                parent.frames[2].location.href = "Noidung1.htm";
                        else if (contentID == 2)
                                parent.frames[2].location.href = "Noidung2.htm";
                        else if (contentID == 3)
                                parent.frames[2].location.href = "Noidung3.htm";
                }
        </script>

    </head>
    <body bgcolor = "#ffff00">
        <a href = "javascript:thayTrang(1);">Trang 1</a> <br>
        <a href = "javascript:thayTrang(2);">Trang 2</a> <br>
        <a href = "javascript:thayTrang(3);">Trang 3</a> <br>

    </body>
```



Ví dụ: Thay nội dung iframe

EditPlus - [C:\lect\LT Web\Web_tinh\iframe1.html]

File Edit View Search Document Project Tools Browser Window Help

```
1  <script type="text/javascript">
2      function SelectDict(dictid)
3      {
4          if (dictid == 1) {
5              document.getElementById("dict_detail").src =
               "http://www.vietgle.vn/tratu/tu-dien-truc-tuyen.aspx?t=2";
6          }
7          else if (dictid == 2) {
8              document.getElementById("dict_detail").src = "http://vdict.com/";
9          }
10         else if (dictid == 3) {
11             document.getElementById("dict_detail").src =
               "http://translate.google.com/translate_t?sl=en&tl=vi#en|vi|welcome";
12         }
13     }
14 </script>
15
16 <table>
17     <tr>
18         <td><a href="javascript:SelectDict(1);">LạcViệt</a></td>
19         <td><a href="javascript:SelectDict(2);">Vdict</a></td>
20         <td><a href="javascript:SelectDict(3);">Google</a></td>
21     </tr>
22 </table>
23 <iframe id="dict_detail" src="http://www.vietgle.vn/tratu/tu-dien-truc-tuyen.aspx?t=2" style="width:100%;
   height:300px; " ></iframe>
24
```

iframe1.html

For Help, press F1 | ln 16 | col 8 | 25 | 00 | UNIX | UTF-8

start | 2 Windows Ex... | Microsoft Power... | EditPlus - [C:\lec... | Mozilla Firefox | Source of: http:... | (2:54)

Ví dụ: Truy cập điều khiển và hàm javascript trong frame và iframe

EditPlus - [C:\lect\LT Web\Web_tinh\iframe2.html]

File Edit View Search Document Project Tools Browser Window Help

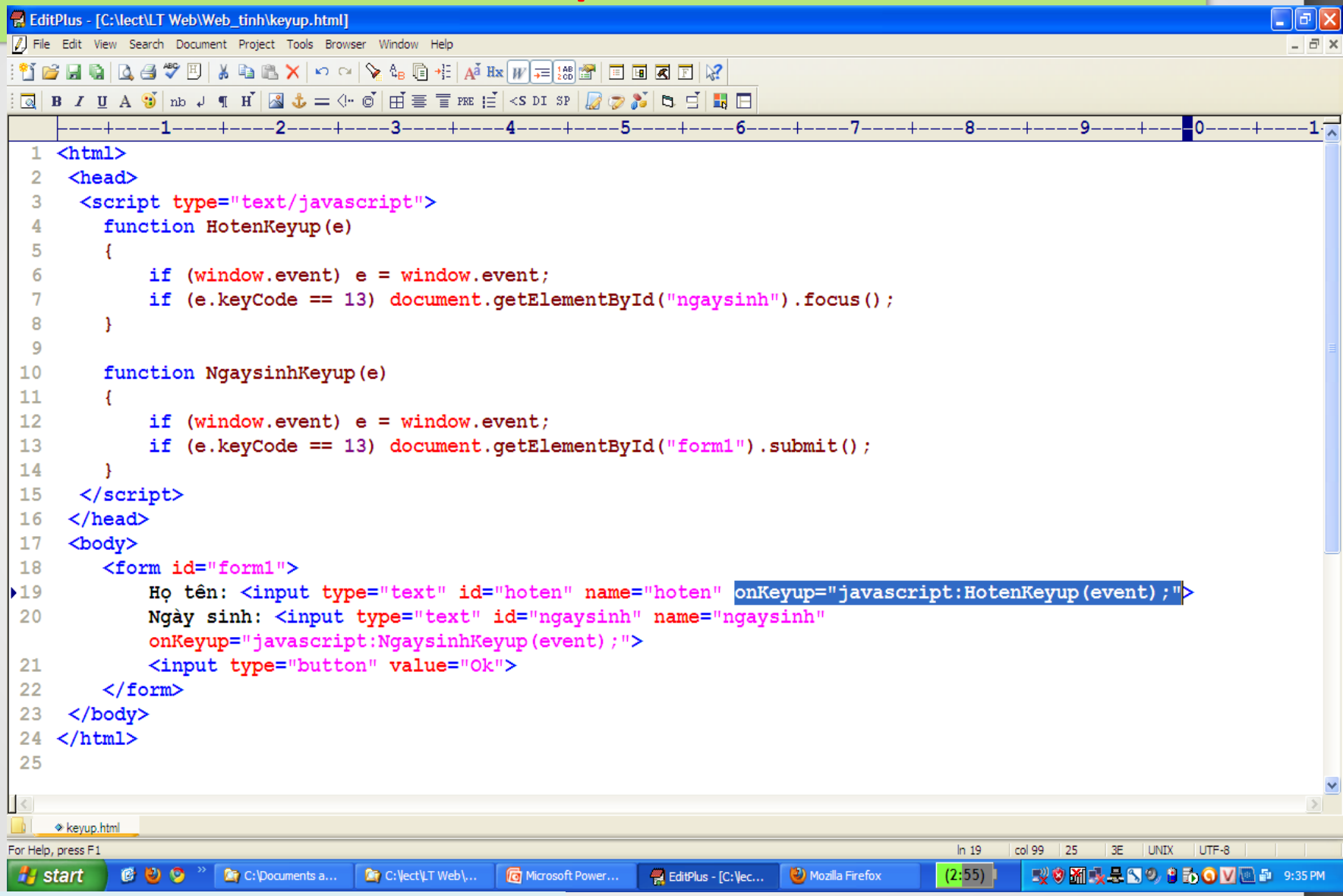
```
1  <script type="text/javascript">
2      function AccessControl()
3      {
4          ht = document.getElementById("someframe").contentWindow.document.getElementById("hoten").value;
5          alert(ht);
6      }
7
8      function AccessJavascriptFunction()
9      {
10         ht = document.getElementById("someframe").contentWindow.AFunction();
11     }
12
13 </script>
14
15 <table>
16     <tr>
17         <td><a href="javascript:AccessControl();">Truy cap dieu khien</a></td>
18         <td><a href="javascript:AccessJavascriptFunction();">Goi ham javascript</a></td>
19     </tr>
20 </table>
21 <iframe id="someframe" src="apage.aspx"></iframe>
22
23
```

iframe2.html

For Help, press F1 — ln 4 | col 64 | 23 | 2E | UNIX | UTF-8

start | 2 Windows Ex... | Microsoft Power... | EditPlus - [C:\lec... | Mozilla Firefox | Source of: http:... | (2:55)

Ví dụ: Nhận phím được bấm



Ví dụ: Đặt điều khiển được chọn

Focus - Notepad

File Edit Format View Help

```
</head>
    <script language="javascript">
        function TenSD_focus(){
            document.getElementById("tensd").focus();
        }

        function tensd_keydown()
        {

            if (window.event.keyCode == 13)
            {
                document.getElementById("matkhau").focus();
                //form1.matkhau.focus();

            }
        }

    </script>

<body onload = "TenSD_focus();">
    <form id = "form1">
        Ten dang nhap: <input type = "text" width = "200"  id = "tensd" onkeydown = "t
```

Microsoft PowerPoint... Focus - Notepad 10:54 AM

Ví dụ: In trên web

In tren Web - Notepad

File Edit Format View Help

```
<html>
    <head>
        <title>Trang thu 2</title>
                    <script language="javascript">
                        function Print()
                        {
                            document.all["idToolPrint"].style.display='none';
                            document.all["idPreview"].style.overflow='hidden';
                            window.print();
                            document.all["idPreview"].style.overflow='auto';
                            document.all["idToolPrint"].style.display='';
                        }
                    </script>
    </head>
    <body>

<div id="idToolPrint" style="BORDER-BOTTOM: steelblue 1px solid; BACKGROUND-COLOR: #99ccff">
                    <table id="toolbar" cellSpacing="0">
                        <tr><td>  <table cellSpacing="0"><tr><td>  <input type="button" id="btnPrin
                                                                    <td><input type="button" id="b
                                                                </tr></table></td></tr></table>
</div>
                <br>
<div id="idPreview" style="OVERFLOW:auto; WIDTH:100%; HEIGHT:90%">
        <h2> Danh sách sinh viên</h2>
        <table border="1"  borderColor="steelblue"
```

Microsoft PowerPoint... | In tren Web - Notepad | 10:58 AM

Vấn đề của trình duyệt

Vấn đề

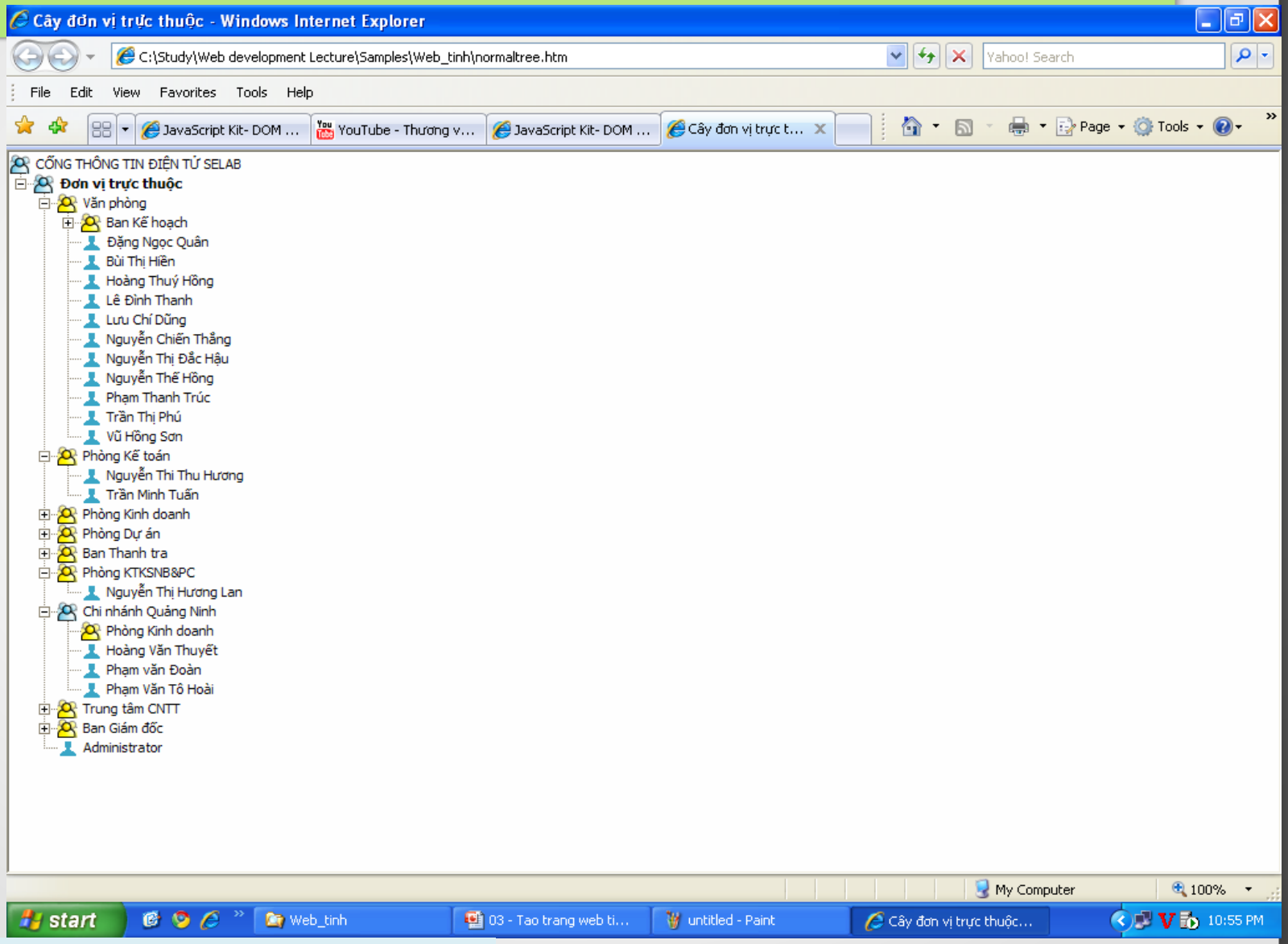
- Vấn đề: Một trang web (cùng nội dung, cùng mã nguồn) có thể hiện (hiển thị và tương tác) khác nhau trên các trình duyệt khác nhau.
- Nguyên nhân: Có nhiều hãng tạo ra nhiều trình duyệt khác nhau như MS-IE, Moz.-FF, Google-Chrome, …. Các hãng không thống nhất với nhau cách xử lý (bản chất trình duyệt chính là trình thông dịch)
- Ví dụ:
  - window.event chỉ được thể hiện ở IE
  - addEventListener hỗ trợ bởi FF tương đương với attachEvent hỗ trợ bởi IE
  - style.opacity chỉ được hỗ trợ từ IE 9.0, FF 2.0, Chrome 4.0 trở lên.

Mong muốn và giải pháp

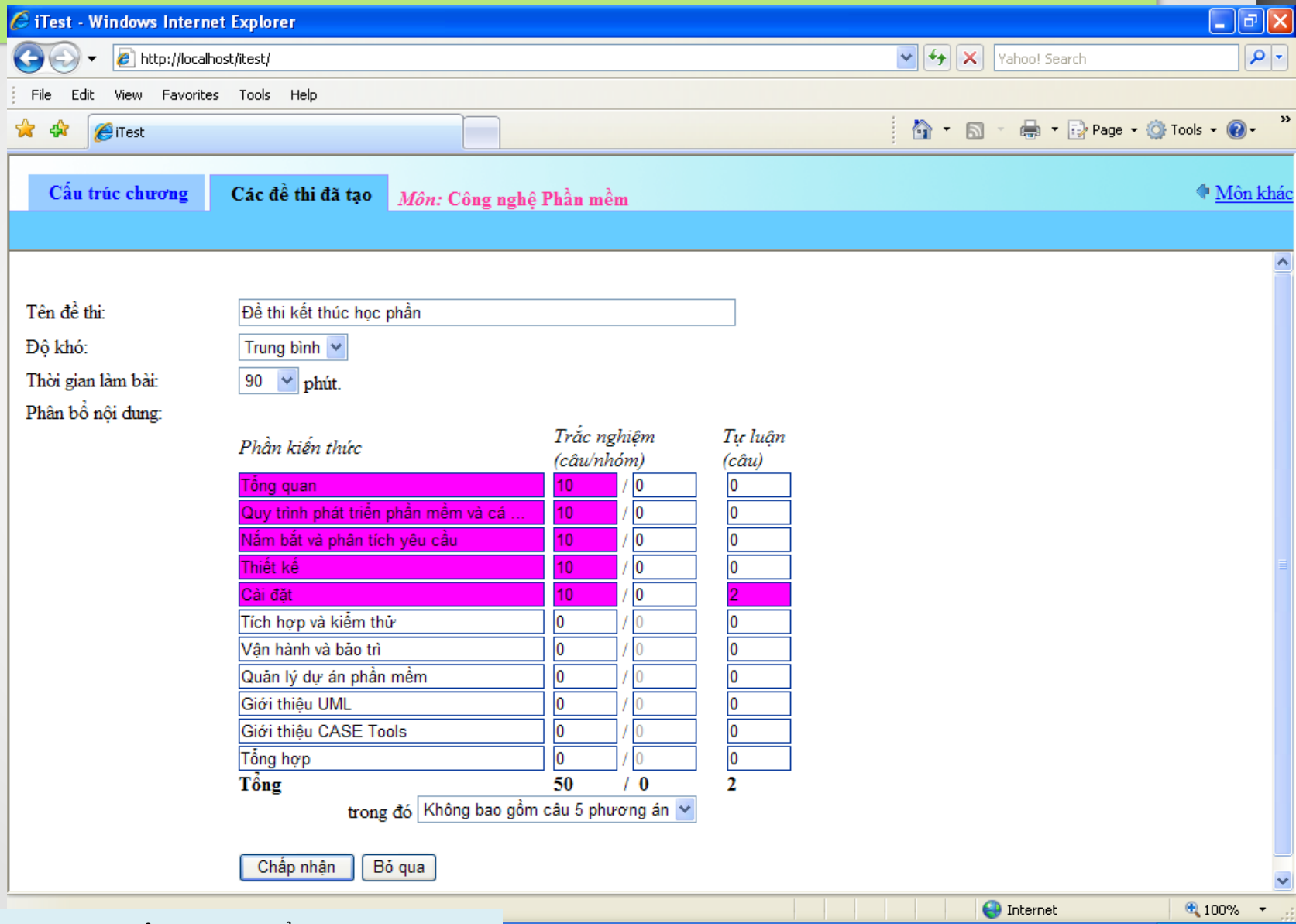
- Mong muốn: Trang web được thể hiện như nhau ở tất cả các trình duyệt (cross-browser).
- Giải pháp:
  - **Sử dụng các đối tượng, thuộc tính, phương thức được hỗ trợ và xử lý như nhau trên các trình duyệt**
  - **Tùy trình duyệt mà sinh mã cho phù hợp**
    - Lấy thông tin trình duyệt: window.clientInformation.appName/.appVersion/. Platform
  - **Sử dụng polyfill**
    - Babel
  - **Khuyến cáo sử dụng trình duyệt được ưu tiên**

Ứng dụng mẫu: Table Sorter

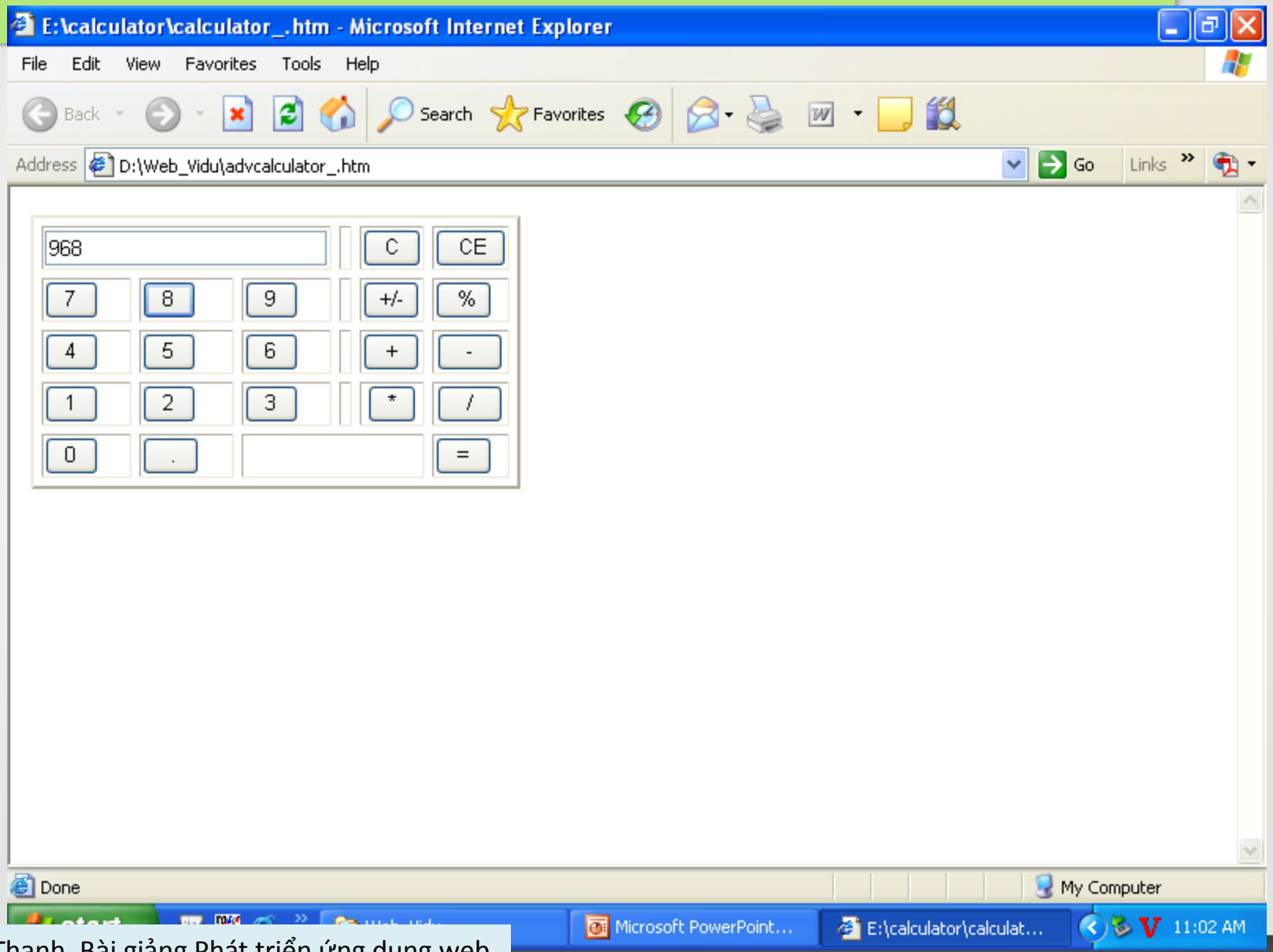
Ứng dụng mẫu: Tree



Ứng dụng mẫu: Tabbed Content



Ứng dụng mẫu: Calculator



Tiếp theo

AJAX và JSON