THIẾT KẾ WEB

Chương 3
NGÔN NGỮ JAVASCRIPT

1. Tổng quan về Javascript
2. Ngôn ngữ JavaScript
3. Đối tượng và sự kiện

1. TỔNG QUAN VỀ JAVASCRIPT

1.1. Giới thiệu

1.2. Nhúng JavaScript vào File HTML

1.3. Các lệnh cơ bản

1.1. GIỚI THIỆU

- ✓Với HTML chỉ biểu diễn thông tin chưa có khả năng đáp ứng các sự kiện từ phía người dùng.
- ✓Hãng Netscape đã đưa ra ngôn ngữ LiveScript để thực hiện chức năng này. Sau đó đổi tên thành JavaScript để tận dụng tính đại chúng của ngôn ngữ lập trình Java.
- ✓JavaScript là ngôn ngữ dạng script có thể gắn với các file HTML. Được trình duyệt diễn dịch dưới dạng mã nguồn.
- ✓JavaScript là ngôn ngữ dựa trên đối tượng, ví dụ đối tượng ***Math*** với tất cả các chức năng toán học.
- ✓ JavaScript không là ngôn ngữ hướng đối tượng như C++/Java.

1.2. NHÚNG JAVASCRIPT VÀO FILE HTML

Sử dụng một trong các cách sau:

- ✓Sử dụng câu lệnh và hàm trong cặp thẻ <SCRIPT>
- ✓Sử dụng các File nguồn JavaScript
- ✓Sử dụng một biểu thức JavaScript làm giá trị của một thuộc tính HTML
- ✓Sử dụng thẻ sự kiện (event handlers) trong một thẻ HTML nào đó

Trong đó, sử dụng cặp thẻ <Script>...</Script> và nhúng một File nguồn JavaScript là được sử dụng nhiều hơn cả.

1.2. NHÚNG JAVASCRIPT VÀO FILE HTML

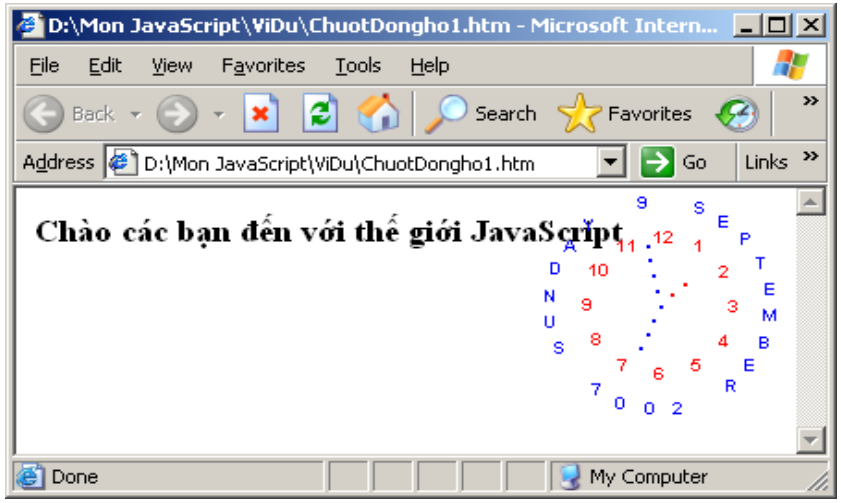
- ✓ **Nhúng JavaScript vào trang HTML**

  Sử dụng cặp thẻ <Script> và </Script>.

  Cú pháp:

```
<Script >
    // Chèn các mã Javacript vào đây
</Script>
```

Ví dụ: Sưu tầm mã JavaScript từ Internet hiệu ứng "Chuột đồng hồ" nhúng vào trang web



1.2. NHÚNG JAVASCRIPT VÀO FILE HTML

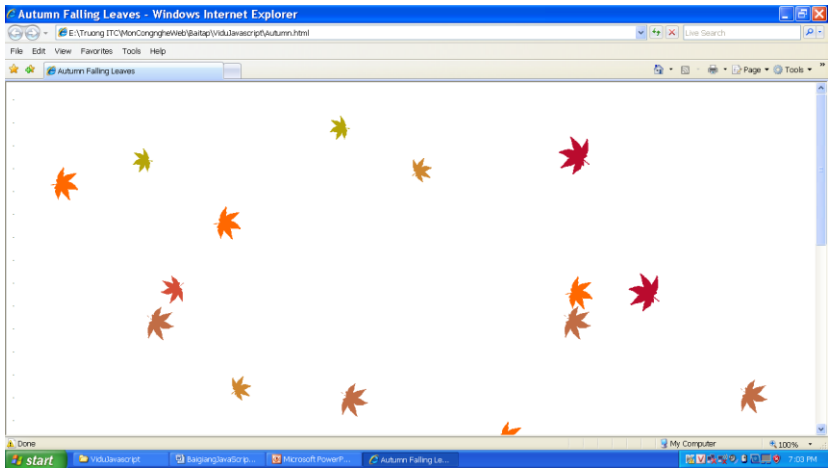
✓ Sử dụng File nguồn JavaScript

Phương pháp này được ưa chuộng hơn bằng cách nhúng file lệnh JavaScript vào trang HTML.

Cú pháp:

```
<Script Src="File_name.js">
</Script>
```

Ví dụ: Sưu tầm mã JavaScript từ Internet hiệu ứng "Ngoài kia lá rơi đầy" nhúng vào trang web



1.3. CÁC LỆNH CƠ BẢN

1.3.1. Cú pháp cơ bản của lệnh:

✓JavaScript xây dựng các hàm, các phát biểu, các toán tử và các biểu thức trên cùng một dòng và kết thúc bằng **;**

✓ Cách gọi một phương thức của một đối tượng như sau:

object_name.property_name;

Ví dụ: document.write("Chào các bạn!
");

1.3. CÁC LỆNH CƠ BẢN

1.3.1. Cú pháp cơ bản của lệnh(tt)

✓**Hiển thị một dòng văn bản**

document.write(“Chuỗi văn bản”);

Ví dụ: document.write("Chào các bạn");

✓**Hiển thị hộp thoại thông báo –Lệnh alert()**

alert("Câu thông báo");

```
<Body>
<Script Language="JavaScript">
alert("Chào mừng bạn đến với JavaScript!. \n Nhấn Ok để tiếp tục");
</Script>
Chúc bạn thành công!!!
</Body
```

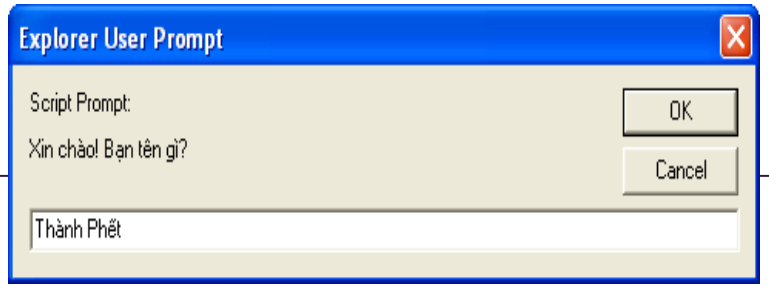
1.3. CÁC LỆNH CƠ BẢN

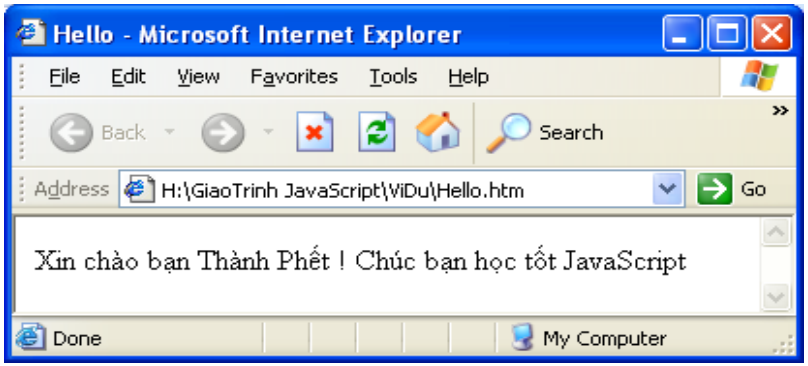
1.3.1. Cú pháp cơ bản của lệnh(tt)

✓Giao tiếp với người sử dụng – Lệnh prompt()

window.prompt("Câu thông báo",”nội dung mặc định”);

```
<Body>
<Script Language="JavaScript">
var   name=window.prompt("Xin chào!Bạn tên gì?","");
document.write("Xin chào bạn " + name + " ! Chúc bạn học tốt JavaScript ");
</Script>
</Body>
```





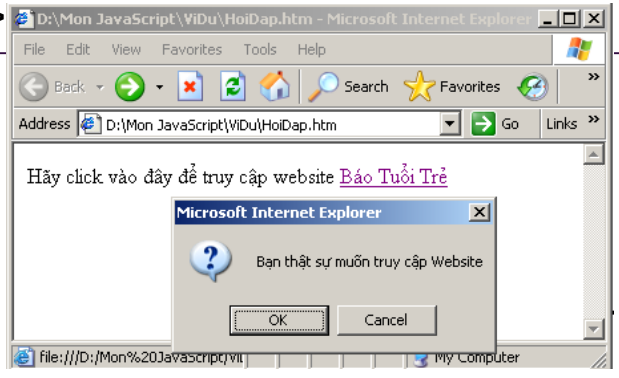
1.3. CÁC LỆNH CƠ BẢN

1.3.1. Cú pháp cơ bản của lệnh(tt)

- ✓ **Hỏi đáp người sử dụng – Lệnh confirm()**

confirm("Câu thông báo hỏi ?");

```
<html><head>
<script>
function Hoidap(){
    question = confirm("Bạn thật sự muốn truy cập Website")
    if (question !="0"){
        top.location = "http://www.tuoitre.com.vn/"
    }
}
</script>
</head><body>
Hãy click vào đây để truy cập website:<a href=""onClick="Hoidap();return false;">Báo Tuổi Trẻ </a>
</body></html>
```



2. NGÔN NGỮ KỊCH BẢN JAVASCRIPT

2.1. Biến

2.2. Kiểu dữ liệu

2.3. Tóan tử & Biểu thức trong JavaScript

2.4. Cấu trúc lập trình

2.5. Mảng - Array

2.6. Hàm - Function

2.1. BIẾN

2.1. Biến

✓ Như các ngôn ngữ lập trình khác javascript dùng biến để lưu trữ các giá trị nhập vào, các giá trị tính toán...

✓Mỗi biến có một tên, tên biến phải bắt đầu bằng ký tự.

✓Phạm vi của biến có thể là một trong hai kiểu sau:

-Biến toàn cục: Có thể được truy cập từ bất kỳ đâu trong ứng dụng. Được khai báo: x = 0;

-Biến cục bộ: Chỉ được truy cập trong phạm vi chương trình mà nó khai báo. Biến cục bộ được khai báo trong một hàm với từ khoá var: var x = 0;

2.2. KIỂU DỮ LIỆU

Khác với C++/Java, JavaScript có tính định kiểu thấp. Nghĩa là không phải chỉ ra kiểu dữ liệu cho biến. Kiểu dữ liệu được tự động chuyển thành kiểu phù hợp khi cần

```
<HTML><Body> <Script Language= "JavaScript">
var a='Trái táo';
var n=12;
n = n + 20;
var tb ="Có tất cả " + n + " " + a;
document.write(tb);
</Script>
```

2.2. KIỂU DỮ LIỆU(TT)

Trong JavaScript, có bốn kiểu dữ liệu sau đây:

- ✓**Kiểu nguyên (Interger)**
- ✓**Kiểu dấu phẩy động (Floating Point)**
- ✓**Kiểu logic (Boolean)**

  Có hai giá trị : true , false.

- ✓**Kiểu chuỗi (String)**

  Một biến kiểu chuỗi biểu diễn bởi không hay nhiều ký tự đặt trong cặp dấu " ... " hay '... '

2.3. LỆNH, KHỐI LỆNH TRONG JAVASCRIPT

- ✓ Các câu lệnh trong JavaScript kết thúc bằng một dấu chấm phẩy (;).
- ✓ Một khối lệnh là đoạn chương trình gồm hai lệnh trở lên và được đặt trong cặp ngoặc nhọn: { . . . }
- ✓ Bên trong một khối lệnh có thể chứa một hay nhiều khối lệnh khác.

2.4. TÓAN TỬ & BIỂU THỨC TRONG JAVASCRIPT

2.4.1. Định nghĩa và phân loại biểu thức

✓ **Biểu thức (expression) có ba kiểu:**

Số học: Nhằm để lượng giá giá trị số.

Ví dụ: (3+4)+(84.5/3) bằng 197.1666666667.

Chuỗi: Nhằm để đánh giá chuỗi.

Ví dụ: "The dog”+”barked!" là “The dog barked!”

Logic: Nhằm đánh giá giá trị logic.

Ví dụ: 23>32 là False.

Biểu thức điều kiện:

(condition) ? valTrue : valFalse

Ví dụ:

ketqua = (diemtb>=5) ? "Đậu" : "Rớt"

➔ Trong ví dụ này biến ketqua được gán giá trị "Đậu"

2.4. TÓAN TỬ & BIỂU THỨC TRONG JAVASCRIPT

2.4.1. Định nghĩa và phân loại biểu thức

✓Các Toán tử:

=	Gán
==	Bằng
!=	Khác
>	Lớn hơn
>=	Lớn hơn hoặc bằng
<	Nhỏ hơn
<=	Nhỏ hơn hoặc bằng
var1 % var2	Chia lấy phần dư
-	Phủ định
var++	Tăng var lên 1
var--	Giảm var đi 1
+	Kết hợp hai chuỗi
expr1 && expr2	Toán tử AND trả về giá trị đúng nếu expr1 và expr2 cùng đúng.
expr1 || expr2	Toán tử OR trả về giá trị đúng nếu ít nhất 1 trong 2 expr1,expr2 đúng.

2.5. CẤU TRÚC LẬP TRÌNH

2.5.1. Cấu trúc lập trình rẽ nhánh (Điều Kiện)

```
if ( <điều kiện> ) {
    //Các câu lệnh với điều kiện đúng
}
else{
    //Các câu lệnh với điều kiện sai
}
```

2.5. CẤU TRÚC LẬP TRÌNH

2.5.2. Vòng lặp For

```
for (initExpr; <điều kiện>; incrExpr)
{
//Các lệnh được thực hiện trong khi lặp
}
```

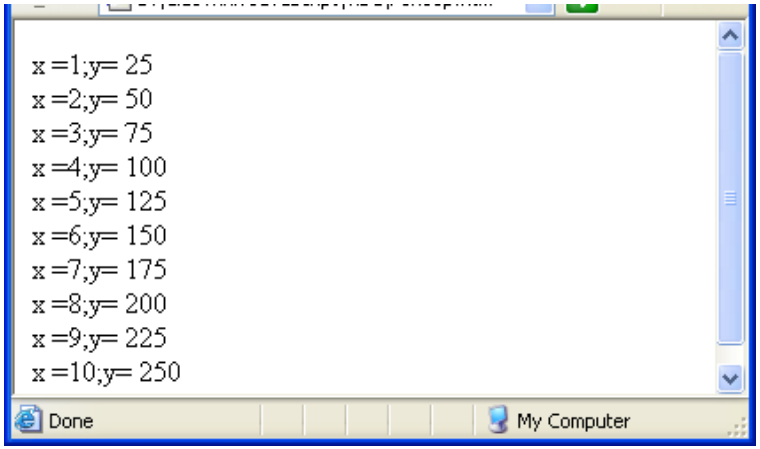
Vòng lặp for thiết lập 1 biểu thức khởi đầu - initExpr, sau đó lặp 1 đoạn mã cho đến khi biểu thức <điều kiện> được đánh giá là đúng. Sau khi kết thúc mỗi vòng lặp, biểu thức incrExpr được đánh giá lại(thường là tăng 1)

2.5. CẤU TRÚC LẶP TRÌNH

2.5.2. Vòng lặp For (tt)

Ví dụ:

```
for (x=1; x<=10 ; x++) {
        y=x*25;
        document.write("x ="+ x +";y= "+ y + "<BR>");
}
```



2.5. CẤU TRÚC LẬP TRÌNH

2.5.3. Vòng lặp While

```
while (<điều kiện>)
{
        //Các câu lệnh thực hiện trong khi lặp
}
```

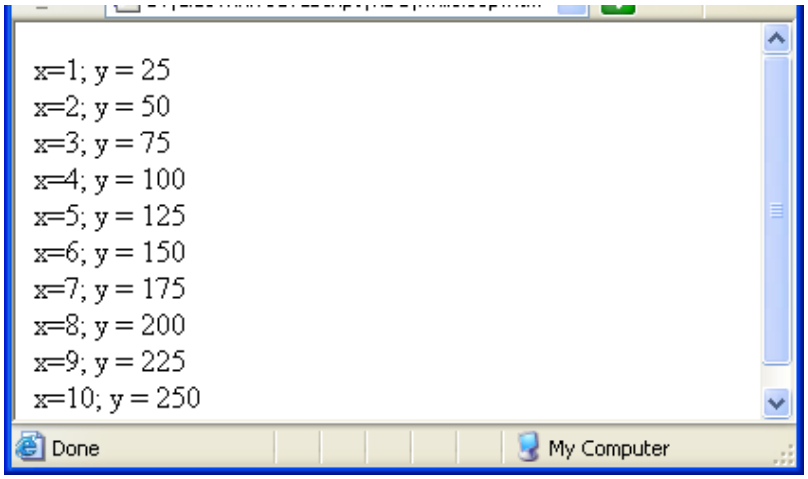
Vòng lặp while lặp khối lệnh khi nào <điều kiện> còn được đánh giá là đúng

2.5. CẤU TRÚC LẶP TRÌNH

2.5.3. Vòng lặp While(tt)

Ví dụ:

```
x=1;
while (x<=10){
        y=x*25;
        document.write("x="+x +"; y = "+ y + "<BR>");
        x++;
}
```



2.5. CẤU TRÚC LẬP TRÌNH

2.5.4. Lệnh Break

Cú pháp: break;

Dùng để kết thúc việc thực hiện của vòng lặp for hay while. Chương trình được tiếp tục thực hiện tại câu lệnh ngay sau chỗ kết thúc của vòng lặp

Ví dụ: Nếu giá trị x đưa vào vòng lặp nhỏ hơn 50, vòng lặp sẽ kết thúc.

```
while (x<100){
        if (x<50) break;
        x++;
}
```

2.5. CẤU TRÚC LẬP TRÌNH

2.5.5. Lệnh Continue

Cú pháp: continue;

Đối với vòng lặp while lệnh continue điều khiển quay lại <điều kiện>; với for lệnh continue điều khiển quay lại incrExpr

Ví dụ: Đoạn mã sau tăng x từ 0 lên 5, nhảy lên 8 và tiếp tục tăng lên 10.

```
x=0;
while (x<=10) {
        document.write("Giá trị của x là:"+ x+"<BR>");
        if (x=5){
                x=8;
                continue;
        }
        x++;
}
```

2.6. MẢNG - ARRAY

JavaScript không hỗ trợ cấu trúc dữ liệu mảng nhưng tạo ra phương thức cho phép tự tạo ra các hàm khởi tạo mảng:

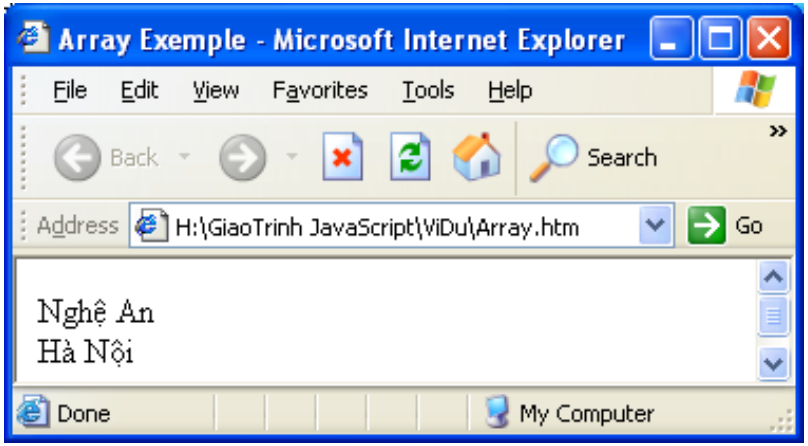
```
function taomang(n) {
        this.length = n;
        for (var i=1; i<=n; i++){
                this[i]=0
        }
        return this;
}
```

Tạo ra 1 mảng với kích thước xác định trước (n) và điền các giá trị 0.

2.6. MẢNG - ARRAY(TT)

Ví dụ: Tạo trang

```
<HTML> <Head>
<Script Language= "JavaScript">
function taomang(n) {
        this.length = n;
        for (var i=1; i<=n; i++){
                this[i]=0
        }
        return this;
}
a = new taomang(10);
a[1] = "Nghệ An";
a[2] = "Hà Nội";
document.write(a[1] + "<BR>");
document.write(a[2] + "<BR>");
</Script>
</Head>
<Body> </Body></HTML>
```



2.7. HÀM - FUNCTION

2.6.1. Giới thiệu

- Trong lập trình sử dụng hàm là để thực hiện một đoạn chương trình nào đó. Và rong Javascript có các hàm được xây dựng sẵn để giúp thực hiện một chức năng nào đó.
- Hàm có thể có 1 hay nhiều tham số truyền vào và 1 giá trị trả về.

2.6.2. Định nghĩa hàm

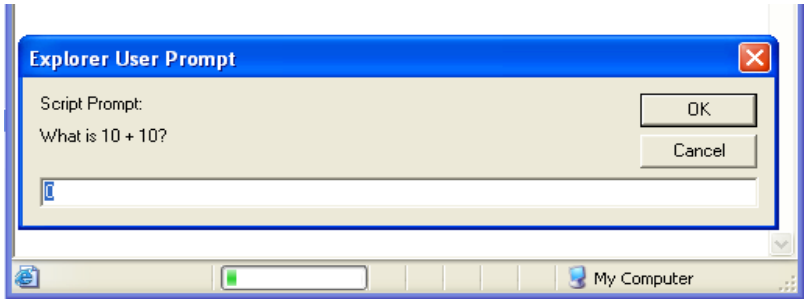
Cú pháp:

```
function fnName([param1],[param2],...,[paramN]){
//function statement
}
```

2.7. HÀM - FUNCTION

Ví dụ:

```
<HTML> <Head> <Title>Function</Title>
<Script Language="JavaScript">
function testQuestion(question){
var answer=eval(question);
var output="What is " + question + "?";
var correct="<IMG SRC='vui.gif'>";
var incorrect="<IMG SRC='buon.gif'>";
var response=prompt(output,"0");
return(response == answer)?correct:incorrect;
}
</Script></Head><Body>
<Script Language="JavaScript">
var result=testQuestion("10 + 10");
document.write(result);
</Script></Body>
</HTML>
```



2.7. HÀM - FUNCTION

2.7.3. Các hàm có sẵn

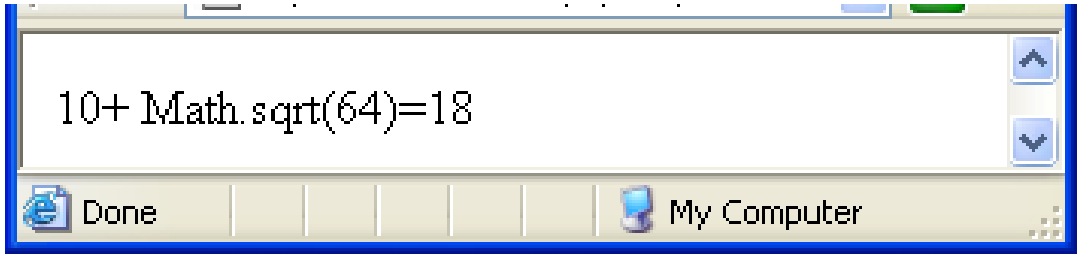
✓**Hàm eval:** Chuyển đổi giá trị chuỗi thành giá trị số.

Cú pháp:

returnval=eval (biểu thức)

Ví dụ: Tạo trang (Eval.htm)

```
<HTML> <Head><Title>Eval Example </Title>
<Script Language= "JavaScript">
        var string=”10+ Math.sqrt(64)”;
        document.write(string+ “=”+ eval(string));
</Script>
</Head>
<Body> </Body>
</HTML>
```



2.7. HÀM - FUNCTION

2.7.3. Các hàm có sẵn(tt)

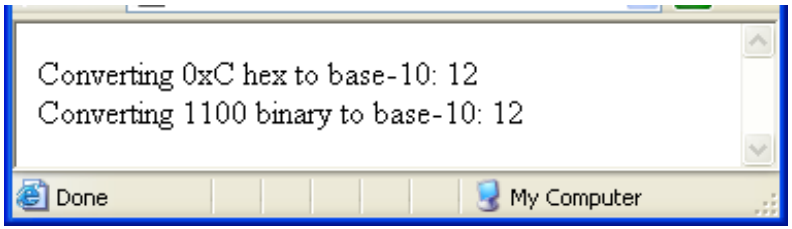
✓ **Hàm parseInt:** Chuyển một chuỗi số thành số nguyên với cơ số là tham số thứ hai.

Cú pháp:

parseInt (string, [, radix])

Ví dụ:

```
<HTML> <Head><Body>
<Script Language= "JavaScript">
document.write("Converting 0xC hex to base-10: " +
                                        parseInt(0xC,10) + "<BR>");
document.write("Converting 1100 binary to base-10:" +
                                        parseInt(1100,2) + "<BR>");
</Script>
</Body></HTML>
```



2.7. HÀM - FUNCTION

2.7.3. Các hàm có sẵn(tt)

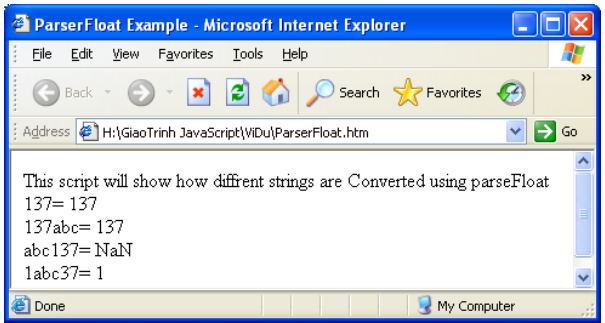
✓ Hàm parseFloat

Chuyển chuỗi thành số biểu diễn dưới dạng số thực.

Cú pháp: parseFloat (string)

Ví dụ:

```
<Body>
<script language= "JavaScript">
document.write("This script will show how diffrent strings are ");
document.write("Converted using parseFloat<BR>");
document.write("137= " + parseFloat("137") + "<BR>");
document.write("137abc= " + parseFloat("137abc") + "<BR>");
document.write("abc137= " + parseFloat("abc137") + "<BR>");
document.write("1abc37= " + parseFloat("1abc37") + "<BR>");
</Script>
</Body>
```



3. ĐỐI TƯỢNGVÀ SỰ KIỆN

3.1. Khái niệm đối tượng

3.1.1. Khái niệm về đôi tượng

3.1.2. Các câu lệnh thao tác trên đối tượng

3.2. Sự kiện & Xử lý sự kiện

3.2.1. Khái niệm sự kiện và xử lý sự kiện

3.2.2. Một số sự kiện trong JavaScript

3.2.3. các sự kiện có sẵn của một số đối tượng.

3.3. Các đối tượng thường dùng

3.3.1. Đối tượng window

3.3.2. Đối tượng forms

3.3.3. Đối tượng Date

3.3.4. Đối tượng Math

3.3.5. Đối tượng String

3.3.6. Đối tượng history

3.3.7. Đối tượng links

3.3.8. Đối tượng Navigator

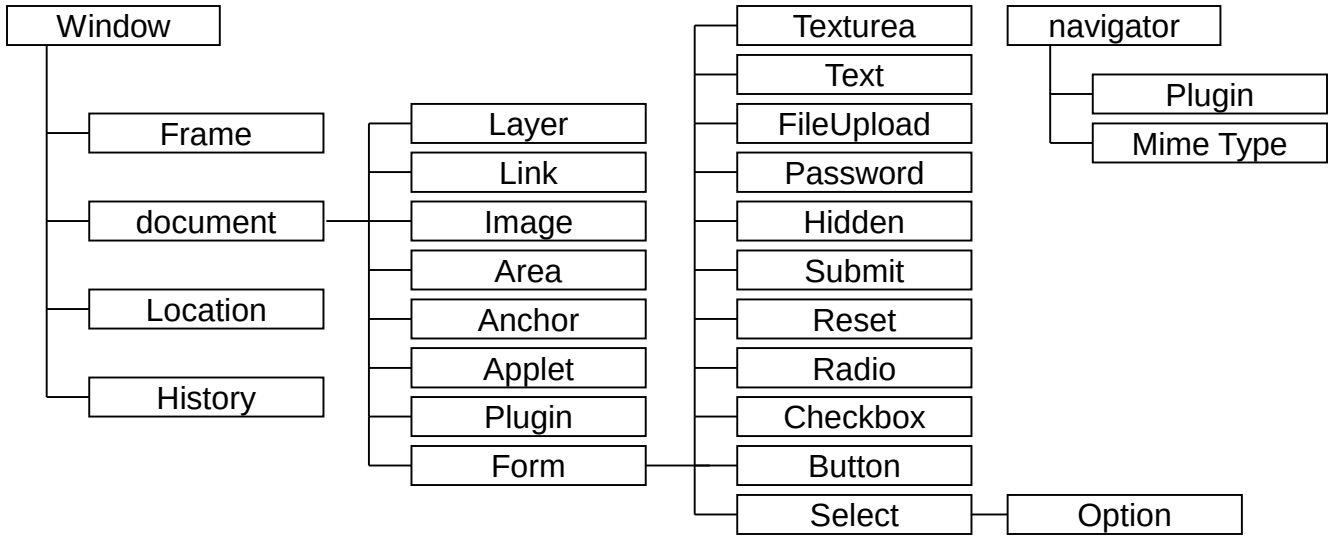
3.3.9. Đối tượng document

3.1. KHÁI NIỆM ĐỐI TƯỢNG

3.1.1. Khái niệm về đôi tượng

JavaScript là ngôn ngữ lập trình dựa trên đối tượng. Trong sơ đồ phân cấp các đối tượng của JavaScript, các đối tượng con thực sự là các thuộc tính của các đối tượng cha.

Vi dụ chương trình xử lý sự kiện trên form tên frmDieutra là thuộc tính của đối tượng document và trường text txtAge là thuộc tính của form frmDieutra. Để tham chiếu đến giá trị của txtAge phải sử dụng: **document.frmDieucha.txtAge.value**



3.1. KHÁI NIỆM ĐỐI TƯỢNG

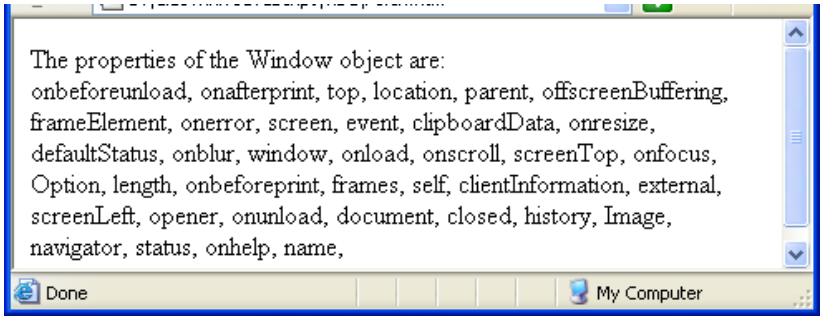
3.1.2. Các câu lệnh thao tác trên đối tượng

✓ Lệnh For...in

Sử dụng để biết tất cả các thuộc tính (properties) của một đối tượng

```
for (<variable> in <object>) {
//Các câu lệnh
}
```

```
<Body><SCRIPT LANGUAGE= "JavaScript"><BODY>
document.write("The properties of the Window  object are: <BR>");
for (var x in window)
        document.write("     "+ x + ", ");
</SCRIPT></Body>
```



3.1. KHÁI NIỆM ĐỐI TƯỢNG

3.1.2. Các câu lệnh thao tác trên đối tượng(tt)

✓Biến new

Được thực hiện để tạo ra một thể hiện mới của một đối tượng

objectvar = new object_type (param1 [,param2]... [,paramN])

✓Từ khóa This

Được sử dụng để chỉ đối tượng hiện thời.

this [.property]

✓ Lệnh With

Sử dụng để thiết lập đối tượng ngầm định cho một nhóm các lệnh.

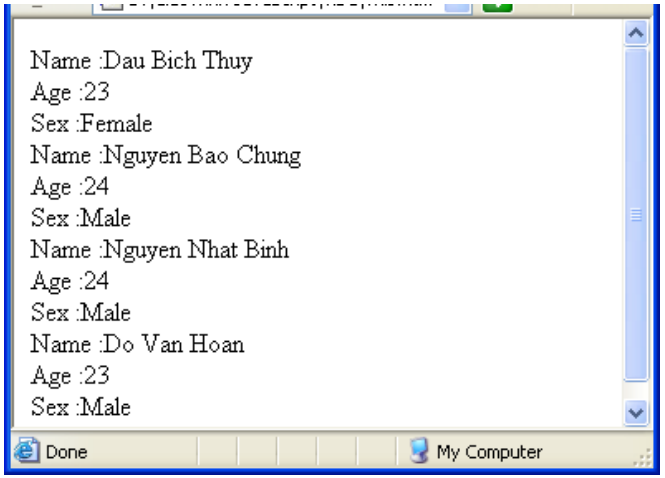
```
with(object){
        // statement
}
```

3.1. KHÁI NIỆM ĐỐI TƯỢNG

3.1.2. Các câu lệnh thao tác trên đối tượng(tt)

✓**Ví dụ:**

```
<HTML> <Script Language= "JavaScript">
function person(first_name, last_name, age, sex){
        this.first_name=first_name;
        this.last_name=last_name;
        this.age=age;
        this.sex=sex;
        this.printStats=printStats;
}
function printStats() {
        with (document) {
        write (" Name :" + this.last_name
                + " " + this.first_name + "<BR>" );
        write("Age :"+this.age+"<BR>");
        write("Sex :"+this.sex+"<BR>");
        }
}
person1= new person("Thuy", "Dau Bich", "23", "Female");
person2= new person("Chung", "Nguyen Bao", "24", "Male");
person3= new person("Binh", "Nguyen Nhat", "24", "Male");
person4= new person("Hoan", "Do Van", "23", "Male");
person1.printStats();
person2.printStats();
person3.printStats();
person4.printStats();
</SCRIPT></HEAD>
<BODY> </BODY></HTML>
```



3.2. SỰ KIỆN & XỬ LÝ SỰ KIỆN

3.2.1. Khái niệm sự kiện và xử lý sự kiện

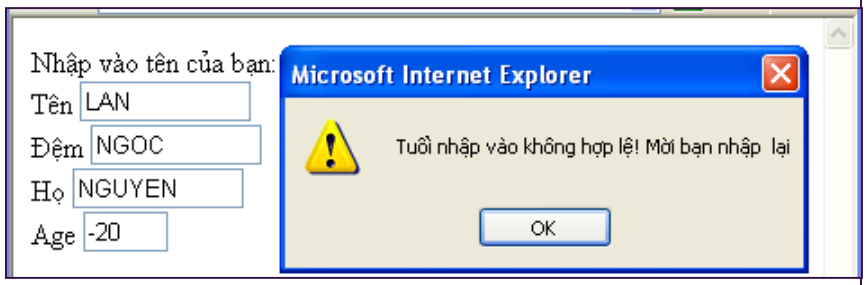
- ✓ JavaScript là ngôn ngữ định hướng sự kiện, nghĩa là sẽ phản ứng trước các sự kiện như: Click chuột . . .
- ✓ Chương trình xử lý sự kiện (Event handler) là 1 đoạn mã hay 1 hàm được thực hiện để phản ứng trước 1 sự kiện được xác định là một thuộc tính của một thẻ HTML:

```
<tagName  eventHandler = "JavaScript Code or Function">
```

3.2. SỰ KIỆN & XỬ LÝ SỰ KIỆN

3.2.1. Khái niệm sự kiện và xử lý sự kiện (tt)

```
<HTML><HEAD>
<Script Language= "JavaScript">
function CheckAge(form) {
if ( (form.AGE.value<0)||(form.AGE.value>120) ) {
alert("Tuổi nhập vào không hợp lệ! Mời bạn nhập  lại");
      form.AGE.value=0;
      }
}
</Script></Head><Body>
<Form NAME="frmDieutra">
Nhập vào tên của bạn:<BR>
Tên <Input Type=Text Name="TEN" ><BR>
Đệm <Input Type=Text Name="DEM" ><BR>
Họ  <Input Type=Text Name="HO" ><BR>
Age <Input Type=Text Name="AGE“
                                  onChange="CheckAge(frmDieutra)"><BR>
<Input Type=Submit Value=”Submit”>
<Input Type=Reset Value=”Reset”>
</Form></Body></HTML>
```



3.2. SỰ KIỆN & XỬ LÝ SỰ KIỆN

3.2.2. Một số sự kiện trong JavaScript

onBlur	Khi input focus bị xoá từ thành phần form
onClick	Khi người dùng kích vào các thành phần hay liên kết của form.
onChange	Khi giá trị của thành phần được chọn thay đổi
onFocus	Khi thành phần của form được focus(làm nổi lên).
onLoad	Khi trang Web được tải.
onMouseOver	Khi di chuyển chuột qua kết nối hay anchor.
onSelect	Khi người sử dụng lựa chọn một trường nhập dữ liệu trên form.
onSubmit	Khi người dùng đưa ra một form.
onUnLoad	Khi người dùng đóng một trang

3.2. SỰ KIỆN & XỬ LÝ SỰ KIỆN

3.2.3. Các sự kiện có sẵn của một số đối tượng.

Đối tượng	Chương trình xử lý sự kiện có sẵn
Selection list	onBlur, onChange, onFocus
Text	onBlur, onChange, onFocus, onSelect
Textarea	onBlur, onChange, onFocus, onSelect
Button	onClick
Checkbox	onClick
Radio button	onClick
Hypertext link	onClick, onMouseOver, onMouseOut
Clickable Imagemap area	onMouseOver, onMouseOut
Reset button	onClick
Submit button	onClick
Document	onLoad, onUnload, onError
Window	onLoad, onUnload, onBlur, onFocus
Framesets	onBlur, onFocus
Form	onSubmit, onReset
Image	onLoad, onError, onAbort

3.2. SỰ KIỆN & XỬ LÝ SỰ KIỆN

3.2.3. Các sự kiện có sẵn của một số đối tượng.(tt)

```
<HTML>
<BODY onLoad="alert('Welcome to my page!');"
                                        onUnload="alert('Goodbye! ');">
<IMG SRC="Logo.jpg">
</BODY>
</HTML>
```

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.1. Đối tượng window

✓Các thuộc tính:

defaultStatus	Thông báo ngầm định hiển thị lên trên thanh trạng thái của cửa sổ
Frames	Mảng xác định tất cả các frame trong cửa sổ.
Length	Số lượng các frame trong cửa sổ cha.
Name	Tên của cửa sổ hiện thời.
Parent	Đối tượng cửa sổ chA
Self	Cửa sổ hiện thời.
Status	Thông báo hiển thị lên trên thanh trạng thái cửa sổ.
Top	Cửa sổ ở trên cùng.
Window	Cửa sổ hiện thời.

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.1. Đối tượng window(tt)

✓Các phương thức:

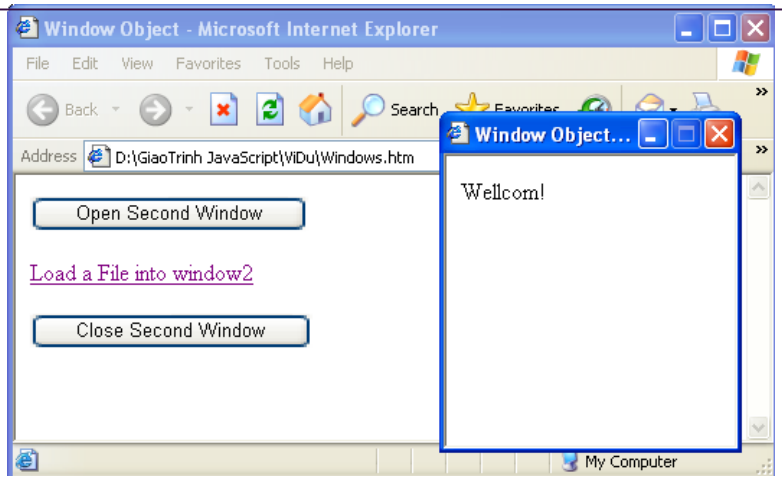
alert ("message")	Hiển thị hộp thoại với chuỗi "message" và nút OK.
clearTimeout(timeoutID)	Xóa timeout do SetTimeout đặt. SetTimeout trả lại timeoutID
WindowReference.close	Đóng cửa sổ windowReference.
confirm("message")	Hiển thị hộp thoại với chuỗi "message", nút OK và nút Cancel. Trả lại trị True cho OK và False cho Cancel.
[windowVar =][window]. open("URL", "windowName", ["windowFeatures"])	Mở cửa sổ mới.
prompt ("message“ [,"defaultInput"])	Mở hộp hội thoại để nhận dữ liệu vào trường text.
TimeoutID = setTimeout(expression,msec)	Đánh giá biểu thức expresion sau thời gian msec.

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.1. Đối tượng window(tt)

✓Các phương thức: (tt)

```
<HTML><Body>
<Form>
<Input Type="button" VALUE="Open Second Window" onClick=
"msgWindow=window.open('','window2','resizable=no,width=200,height=200')">
<BR><A HREF="doc.htm" TARGET="window2"> Load a File into window2 </A>
<Input Type="button" VALUE="Close Second Window“
 onClick="msgWindow.close()">
</Form>
</Body></HTML>
```



3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.2. Đối tượng forms

✓Các thuộc tính:

Action	thuộc tính ACTION của thẻ FORM.
Elements	Mảng chứa các thành phần trong form (như checkbox, textBOX . .
Encoding	Xâu chứa kiểu MIME được sử dụng để mã hoá nội dung của form gửi cho server.
length	Số lượng các thành phần trong một form.
Method	Thuộc tính METHOD.
target	Xâu chứa tên của cửa sổ đích khi submit form

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.2. Đối tượng forms(tt)

✓Các phương thức:

formName.submit () - Xuất dữ liệu của một form tên formName tới trang xử lý. Phương thức này mô phỏng khi click vào nút submit trên form.

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.2. Đối tượng forms(tt)

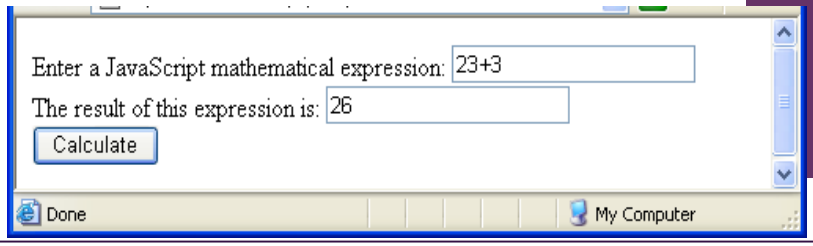
✓Các phần tử của đối tượng Form:

Phần tử	Cú pháp	Mô tả
Button	<Input Type="button">	Một nút
Checkbox	<Input Type="checkbox">	Một checkbox
FileUpload	<Input Type="File">	Một phần tử cho phép sử dụng gửi File
Hidden	<Input Type="hidden">	Một trường ẩn
Password	<Input Type="password">	Một trường text để nhập mật khẩu (*)
Radio	<Input Type="radio">	Một nút chọn
Reset	<Input Type="reset">	Một nút reset
Select	<Select> <Option>option1</Option> <Option>option2</Option> </Select>	Một danh sách lựa chọn
Submit	<Input Type="submit">	Một nút submit
Text	<Input Type="text">	Một trường text
textArea	<Textarea>defaulttext</Textarea>	Một trường text cho nhập nhiều dòng

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.2. Đối tượng forms(tt)

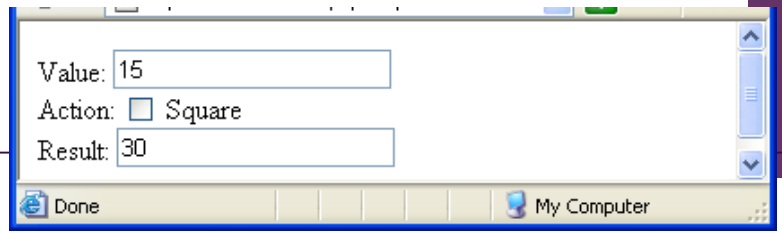
✓**Ví dụ:**




```
<HTML><Script Language="JavaScript">
function calculate(form) {
        form.results.value = eval(form.entry.value);
}
</Script>
</Head>
<Body>
<Form Method=POST>
Enter a JavaScript mathematical expression:
<INPUT TYPE="text" NAME="entry" VALUE=""> <BR>
The result of this expression is:
<INPUT TYPE=text NAME="results" onFocus="this.blur();"> <BR>
<INPUT TYPE="button" VALUE="Calculate" onClick="calculate(this.form);">
</Form>
</Body></HTML>
```

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.2. Đối tượng forms(tt)



✓**Ví dụ:**

```
<HTML><Script>
function calculate(form,callingField) {
if (callingField == "result") {
        if (form.square.checked){
                form.entry.value = Math.sqrt(form.result.value);
        }else{  form.entry.value = form.result.value / 2;}
}else{
        if (form.square.checked){
                form.result.value=form.entry.value*form.entry.value;
        }else {  form.result.value = form.entry.value * 2;
    }
}
}
</Script></Head><Body><Form Method=Post>
Value: <Input Type="text" Name="entry" Value=0
                                        onChange="calculate(this.form,this.name);"><BR>
Action: <Input Type=checkbox NAME=square
                                    onClick="calculate(this.form,this.name);"> Square<BR>
Result: <Input Type="text" Name="result" Value=0
                                            onChange="calculate(this.form,this.name);">
</Form></Body></HTML>
```

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.3. Đối tượng Date

✓Các phương thức

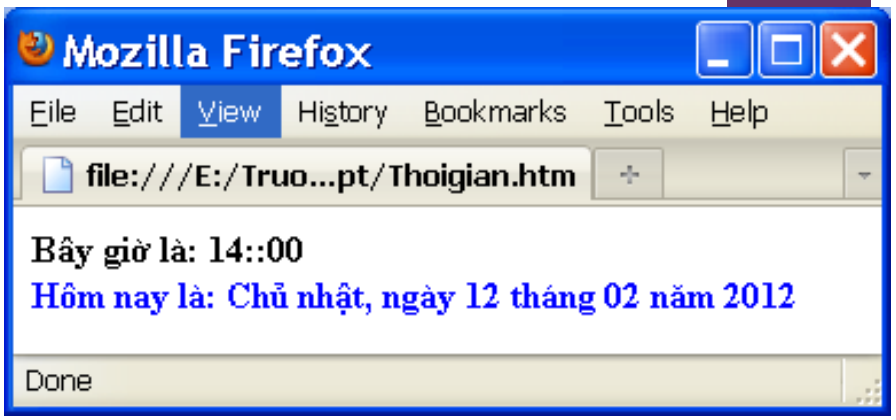
dateVar.getYear()	Trả lại năm
dateVar.getMonth()	Trả lại thang(1-12)
dateVar.getDate()	Trả lại ngày trong tháng (1-31)
dateVar.getDay()	Trả lại ngày trong tuần (0=chủ nhật,...6=thứ bảy)
dateVar.getHours()	Trả lại giờ (0-23) .
dateVar.getMinutes()	Trả lại phút (0-59)
dateVar.getSeconds()	Trả lại giây (0-59)
dateVar.setDay(day)	Đặt ngày trong tháng là day cho dateVar.
dateVar.setMonths(months)	Đặt tháng là months cho dateVar.
dateVar.setYear(years)	Đặt năm là years cho dateVar.
dateVar.setHours(hours)	Đặt giờ là hours cho dateVar.
dateVar.setMinutes(minutes)	Đặt phút là minutes cho dateVar.
dateVar.setSeconds(seconds)	Đặt giây là seconds cho dateVar.

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.3. Đối tượng Date

✓**Ví dụ**

```
<Script Language="JavaScript">
d = new Date();
thu = d.getDay() ;  ngay= d.getDate();
ngay= ((ngay< 10) ? '0' : '') + ngay;
thang= d.getMonth()+1;
thang= ((thang< 10) ? '0' : '') + thang;
nam= 1900  + d.getYear();
gio = d.getHours();
phut = d.getMinutes();
phut= ((phut< 10) ? ':0' : ':') + phut;
if(thu == 0)          thu = " Chủ nhật";
if(thu == 1)          thu = " Thứ hai";
if(thu == 2)          thu = " Thứ ba";
if(thu == 3)          thu = " Thứ tư";
if(thu == 4)          thu = " Thứ năm";
if(thu == 5)          thu = " Thứ sáu";
if(thu == 6)          thu = " Thứ bảy";
</script>
```




```
<body>
<script>
document.write("<b>" +"Bây giờ là: "+
                    gio + ":" + phut +"<br>" ) ;
document.write("<font color=blue>
Hôm nay là:" + thu + ", ngày " + ngay +
 " tháng " + thang + "  năm " + nam +
"</font>");
</script>
</body>
```

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.4. Đối tượng Math

✓Các thuộc tính

E	Hằng số Euler, khoảng 2,718.
LN2	logarit tự nhiên của 2, khoảng 0,693.
LN10	logarit tự nhiên của 10, khoảng 2,302.
LOG2E	logarit cơ số 2 của e, khoảng 1,442.
PI	Giá trị của pi, khoảng 3,14159.
SQRT1_2	Căn bậc 2 của 0,5, khoảng 0,707.
SQRT2	Căn bậc 2 của 2, khoảng 1,414.

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.4. Đối tượng Math(tt)

✓Các phương thức

Math.abs (number)	Trả lại giá trị tuyệt đối của number.
Math.ceil (number)	Trả lại số nguyên nhỏ nhất lớn hơn hoặc bằng number.
Math.cos (number)	Trả lại giá trị cosine của number.
Math.floor (number)	Trả lại số nguyên lớn nhất nhỏ hơn hoặc bằng number.
Math.max (num1,num2)	Trả lại giá trị lớn nhất giữa num1 và num2
Math.min (num1,num2)	Trả lại giá trị nhỏ nhất giữa num1 và num2.
Math.pos (base,exponent)	Trả lại giá trị base luỹ thừa exponent.
Math.round (number)	Trả lại giá trị của number làm tròn tới số nguyên
Math.sqrt (number)	Trả lại căn bậc 2 của number.
……..	……

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.5. Đối tượng String

✓Các phương thức

str.charAt(a)	Trả lại ký tự thứ a trong chuỗi str.
str.fontcolor()	Kết quả giống như thẻ <FONTCOLOR = color>.
str.fontsize(size)	Kết quả giống như thẻ <FONTSIZE = size>.
str.index0f(srchStr [,index])	Trả lại vị trí trong chuỗi str vị trí xuất hiện đầu tiên của chuỗi srchStr. Chuỗi str được tìm từ trái sang phải. Tham số index có thể được sử dụng để xác định vị trí bắt đầu tìm kiếm
str.small()	Kết quả giống như thẻ <SMALL> trên chuỗi str.
str.sub()	Tạo ra một subscript cho chuỗi str, giống thẻ <SUB>.
str.substring(a,b)	Trả lại chuỗi con của str là các ký tự từ vị trí thứ a tới vị trí thứ b. Các ký tự được đếm từ trái sang phải bắt đầu từ 0.
str.sup()	Tạo ra superscript cho chuỗi str, giống thẻ <SUP>.
str.toLowerCase()	Đổi chuỗi str thành chữ thường.
str.toUpperCase()	Đổi chuỗi str thành chữ hoa.
……..	…….

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.6. Đối tượng history

Sử dụng để lưu giữ các thông tin về các URL trước được sử dụng. Danh sách các URL được lưu trữ theo thứ tự thời gian.

✓**Các thuộc tính**

Length - Số lượng các URL trong đối tượng.

✓**Các phương thức**

history.back(): Để tham chiếu tới URL mới được thăm trước đây.

history.forward(): Để tham chiếu tới URL kế tiếp trong danh sách.

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.7. Đối tượng links

Là đoạn văn bản hay ảnh là một liên kết. Các thuộc tính của đối tượng link chủ yếu xử lý về URL của các liên kết.

✓**Các thuộc tính**

Hostname	Tên của host và domain (ww.abc.com).
href	Toàn bộ URL cho document hiện tại.
Pathname	Phần đường dẫn của URL (/chap1/page2.html).
port	Cổng truyền thông được sử dụng cho máy tính host, thường là cổng ngầm định.
Protocol	Giao thức được sử dụng(http:).
Target	Giống thuộc tính target
……..	……

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.8. Đối tượng Navigator

Được sử dụng để biết các thông tin về trình duyệt như số phiên bản.

✓**Các thuộc tính**

AppName	Xác định tên trình duyệt.
AppVersion	Xác định thông tin về phiên bản của đối tượng navigator.
…….	……

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.9. Đối tượng document

Đối tượng này chứa các thông tin về document hiện thời. Được tạo bằng cặp thẻ <BODY> và </BODY>.

✓Các thuộc tính

bgColor	Giống thuộc tính Bgcolor.
fgColor	Giống thuộc tính Text.
forms	Mảng tất cả các form trong document.
links	Mảng tất cả các link trong document.
location	URL đầy đủ của văn bản.
referrer	URL của văn bản gọi nó.
title	Nội dung của thẻ <Title>.
……..	…….

3.3. CÁC ĐỐI TƯỢNG THƯỜNG DÙNG

3.3.9. Đối tượng document (tt)

✓**Các phương thức**

document.clear	Xoá document hiện thời.
document.write(expression1 [,expression2]...[,expressionN])	Viết biểu thức ra một cửa sổ xác định.
……	……

LỜI KẾT:

Nên tham khảo toàn diện JavaScript trên Web của hãng Netscape (http://www.netscape.com) để có các thông tin mới nhất về ngôn ngữ này.

Chương 3
NGÔN NGỮ JAVASCRIPT

THE END.