

# Bài 03. JavaScript

*Giảng viên: Phạm Nhân Nghĩa*

*Email: [nghiapn89@gmail.com](mailto:nghiapn89@gmail.com)*

## I. NHÚNG JAVASCRIPT VÀO FILE HTML

### 1.1 sử dụng thẻ SCRIPT

Cú pháp:

```
<SCRIPT LANGUAGE="JavaScript">  
// INSERT ALL JavaScript HERE  
</SCRIPT>
```

### 1.2 Sử dụng một file nguồn JavaScript

Cú pháp:

```
<SCRIPT SRC="file_name.js">  
....  
</SCRIPT>
```

### 1.2 Nơi để đặt JavaScript.

- Ta có thể đặt trong thẻ `<head>`
- Hay đặt trong thẻ `<body>`

## 1.1 Biến

- Dùng để lưu giá trị
- Có phân biệt chữ hoa, chữ thường

Ví dụ:

*var x;*

*var y = 123;*

*var st=“NIIT Hanoi ICT”;*



# 1.2 Các phép toán số học

Phép toán	Ví dụ	y	x
+	$x=y+2$	$y=5$	$x=7$
-	$x=y-2$	$y=5$	$x=3$
*	$x=y*2$	$y=5$	$x = 10$
/	$x=y/2$	$y=5$	$x = 2.5$
%	$x=y \% 2$	$y=5$	$x=1$
++	$x = ++y$	$y=6$	$x=6$
	$x = y++$	$y=6$	$x=5$
--	$x = --y$	$y=4$	$x=4$
	$x = y--$	$y=4$	$x=5$



# 1.3 Các phép toán gán

Phép toán	Ví dụ	Ý nghĩa	x
=	$x=y$	$x=y$	$x=5$
+ =	$x += y$	$x=x+y$	$x = 15$
- =	$x -= y$	$x=x-y$	$x=5$
* =	$x *= y$	$x=x*y$	$x = 50$
/ =	$x /= y$	$x=x/y$	$x=2$
% =	$x \% = y$	$x=x\%y$	$x=0$



# 1.4 Các phép toán so sánh

Cho trước  $x=5$

Phép toán	So sánh	Kết quả	
<code>==</code>	<code>x == 8</code>	false	
<code>==</code>	<code>x == 5</code>	true	
<code>====</code>	<code>x === "5"</code>	false	So sánh có xét kiểu
<code>====</code>	<code>x === 5</code>	true	
<code>!=</code>	<code>x != 8</code>	true	
<code>!==</code>	<code>x !== "5"</code>	true	
<code>!==</code>	<code>x !== 5</code>	false	
<code>&gt;</code>	<code>x &gt; 8</code>	false	
<code>&lt;</code>	<code>x &lt; 8</code>	true	
<code>&gt;=</code>	<code>x &gt;= 8</code>	false	
<code>&lt;=</code>	<code>x &lt;= 8</code>	<i>true</i>	



# 1.5 Các phép toán

- Phép toán với chuỗi ký tự:

Phép	Ví dụ	text1	text2	text3
+	text3 = text1 + text2	"Good "	"Morning"	"Good Morning"
+=	text1 += text2	"Good Morning"	"Morning"	""

- Phép toán điều kiện:

Cú pháp: **variable** = (condition) ? value1:value2;

Ví dụ: **voteable** = (age &gt; 18) ? "Too young" : "Old enough";



# 1.5 Các phép toán

## Phép toán

## Ví dụ

<code>&amp;&amp;</code>	<code>(x &lt; 10 &amp;&amp; y &gt; 1) is true</code>
<code>  </code>	<code>(x == 5    y == 5) is false</code>
<code>!</code>	<code>!(x == y) is true</code>

Phép toán	Ví dụ	Bít	Kết quả	Số
<code>&amp;</code>	<code>x=5&amp;1</code>	<code>0101 &amp; 0001</code>	<code>0001</code>	<code>1</code>
<code> </code>	<code>x=5 1</code>	<code>0101   0001</code>	<code>0101</code>	<code>5</code>
<code>~</code>	<code>x=~5</code>	<code>~0101</code>	<code>1010</code>	<code>10</code>
<code>^</code>	<code>x=5^1</code>	<code>0101 ^ 0001</code>	<code>0100</code>	<code>4</code>
<code>&lt;&lt;</code>	<code>x = 5 &lt;&lt; 1</code>	<code>0101 &lt;&lt; 1</code>	<code>1010</code>	<code>10</code>
<code>&gt;&gt;</code>	<code>x = 5 &gt;&gt; 1</code>	<code>0101 &gt;&gt; 1</code>	<code>0010</code>	<code>2</code>



### 2.1 HIỂN THỊ MỘT DÒNG TEXT

- Dòng text kết quả sẽ được dịch như các dòng HTML khác và hiển thị trên trang.
- VD:

```
<HTML>
<HEAD>
<TITLE>Ouputting Text</TITLE>
</HEAD>
<BODY> This text is plain.<BR> <B>
<SCRIPT LANGUAGE="JavaScript">
<!-- HIDE FROM OTHER BROWSERS
document.write("This text is bold.</B>");
document.writeln(" This text .");
// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</BODY>
</HTML>
```



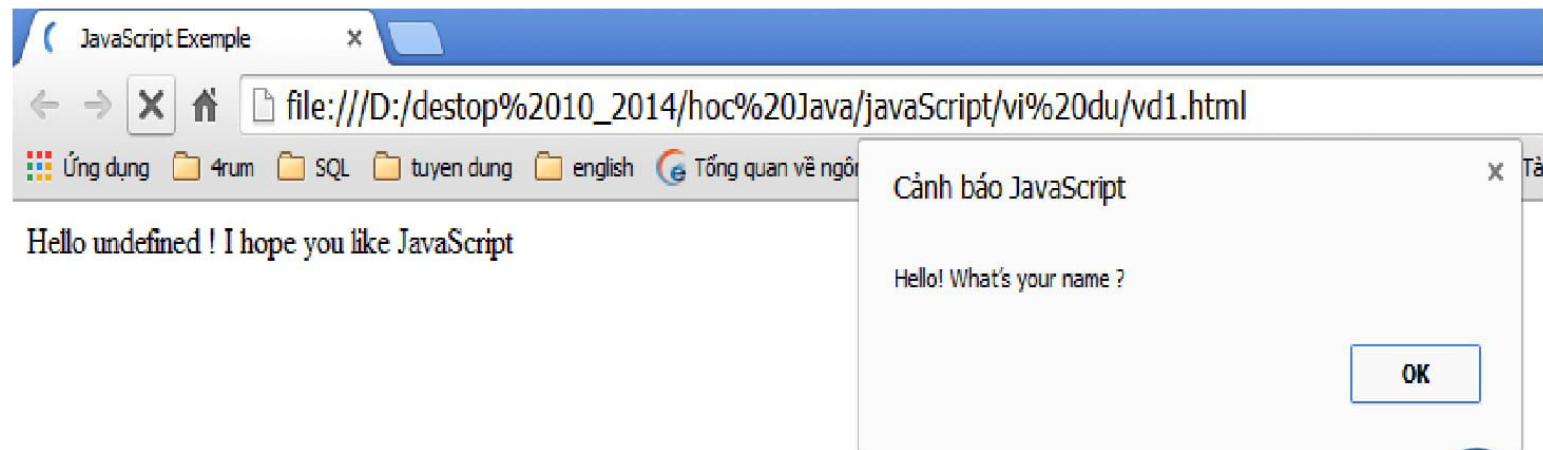
This text is plain.

**This text is bold. This text .**

## 2.2 *Hộp cảnh báo (alert box):*

- Khi alert box xuất hiện, người dùng sẽ click vào nút ok để hoàn thành.
- **Cú pháp:** `alert("sometext");`
- Vd:

```
<HTML>
<HEAD>
<TITLE> JavaScript Exemple </TITLE>
<SCRIPT LANGUAGE= "JavaScript">
var name=window.alert("Hello! What's your name ?","");
document.write("Hello " + name + " ! I hope you like JavaScript ");
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>
```



## 2.3 Hộp nhắc nhở (Prompt box)

- Hộp promptbox được dùng khi ta muốn người dùng nhập vào một giá trị trước khi truy cập một trang. Nếu click **OK** hộp sẽ trả về giá trị nhập vào, nếu click **Cancel** thì hộp trả về **null**.
- VD:

```
<HTML>
```

```
<HEAD>
```

```
<TITLE> JavaScript Exemple </TITLE>
```

```
<SCRIPT LANGUAGE=“JavaScript”>
```

```
var name=window.prompt(“Hello! What’s your name ?”,””);
```

```
document.write(“Hello ” + name + “ ! I hope you like JavaScript ”);
```

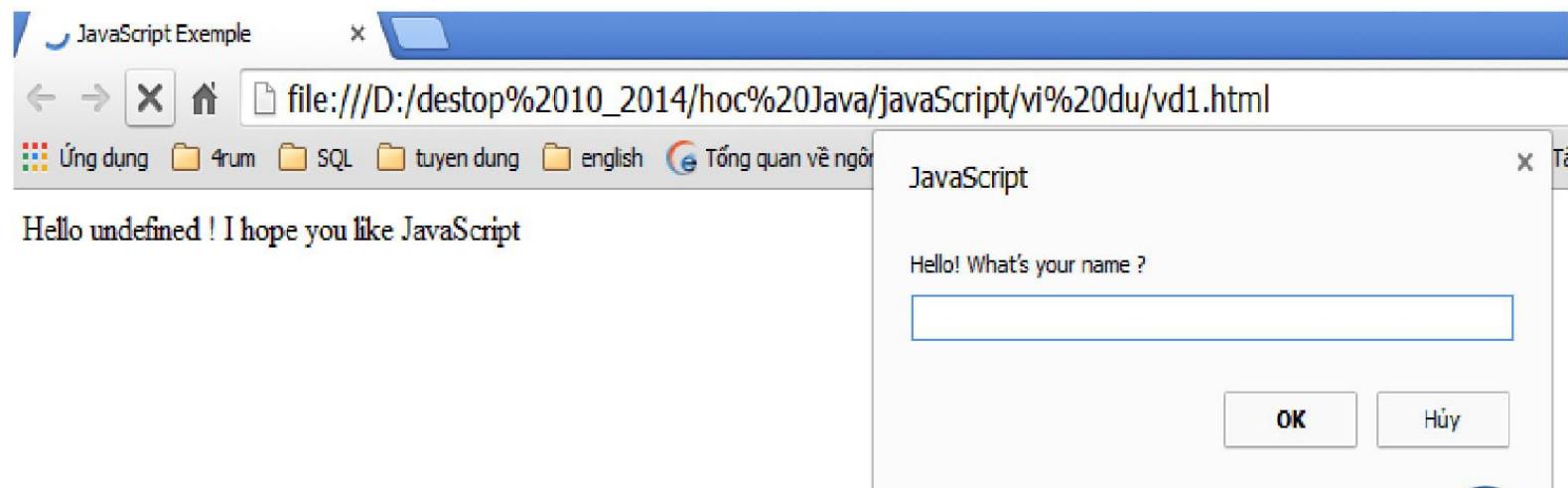
```
</SCRIPT>
```

```
</HEAD>
```

```
<BODY>
```

```
</BODY>
```

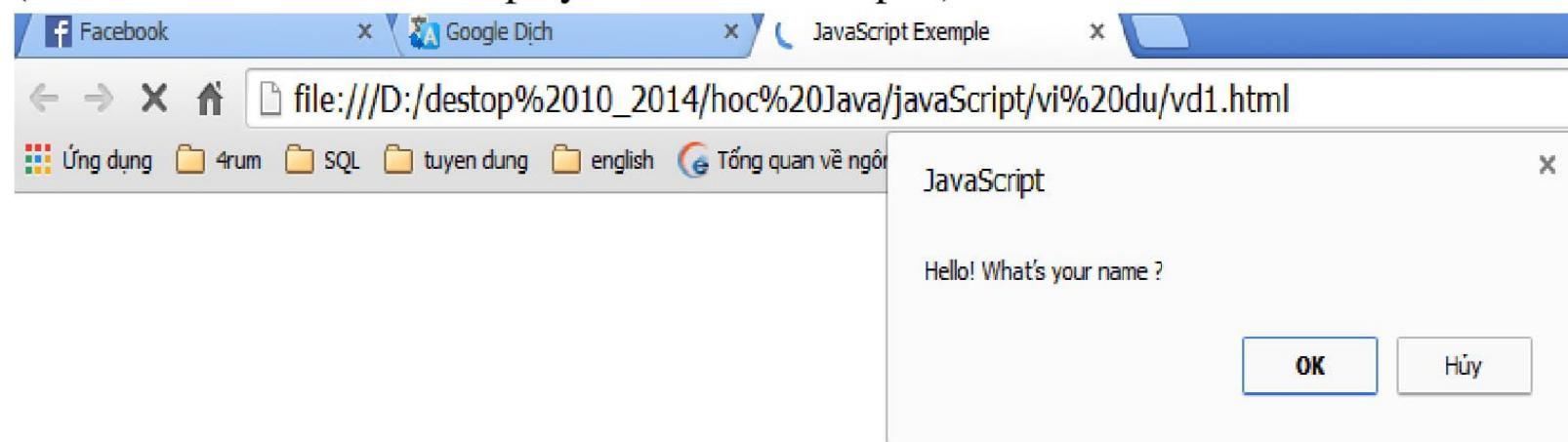
```
</HTML>
```



## 2.4 Hộp xác nhận (confirm box).

- Hộp xác nhận dùng để cho người dùng xác nhận lại một vài thứ. Nếu người dùng click “**OK**”, hộp sẽ trả về giá trị **true**, nếu click vào “**Cancel**” thì trả về **false**.
- Vd:

```
<HTML>
<HEAD>
<TITLE> JavaScript Exemple </TITLE>
<SCRIPT LANGUAGE= "JavaScript">
var name=window.confirm("Hello! What's your name ?","");
document.write("Hello " + name + " ! I hope you like JavaScript ");
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>
```



#### 3. 1. BIẾN VÀ PHÂN LOẠI BIẾN

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

#### 3.2 CÁC TOÁN TỬ

- GÁN: dấu bằng (=)
- SO SÁNH: ==, !=, >, >=, <,
- SỐ HỌC: +, -, \*, /, %, ++, --
- CHUỖI:
- LOGIC: &&, ||, ! expr
- BITWISE: &, |, ^, <<, >>, >>>

- KIỂU NGUYÊN (INTEGER)
- KIỂU DẤU PHẨY ĐỘNG (FLOATING POINT)
- KIỂU LOGIC (BOOLEAN)
- KIỂU CHUỖI (STRING)
- Ta có thể không phải chỉ ra kiểu dữ liệu khi khai báo biến. Kiểu dữ liệu được tự động chuyển thành kiểu phù hợp khi cần thiết.
- VD:

```
<HTML>
<HEAD>
<TITLE> Datatype Example </TITLE>
<SCRIPT LANGUAGE= "JavaScript">
var fruit='apples';
var numfruit=12;
numfruit = numfruit + 20;
var temp ="There are " + numfruit + " " + ".";
document.write(temp);
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>
```



There are 32 .

## IV. CÁC CÂU LỆNH VÀ VÒNG LẶP

### 4.1 if ... else

Cú pháp

```
if ( <điều kiện> )
```

```
{
```

//Các câu lệnh với điều kiện đúng

```
}
```

```
else
```

```
{
```

//Các câu lệnh với điều kiện sai

```
}
```

## 4.2 VÒNG LẶP FOR

Cú pháp:

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

## 4.3 VÒNG LẶP WHILE

Cú pháp:

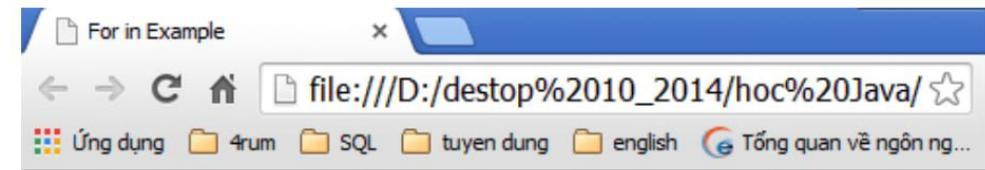
```
while (<điều kiện>){  
    //Các câu lệnh thực hiện trong khi lặp  
}
```

## Cú pháp

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

VD:

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



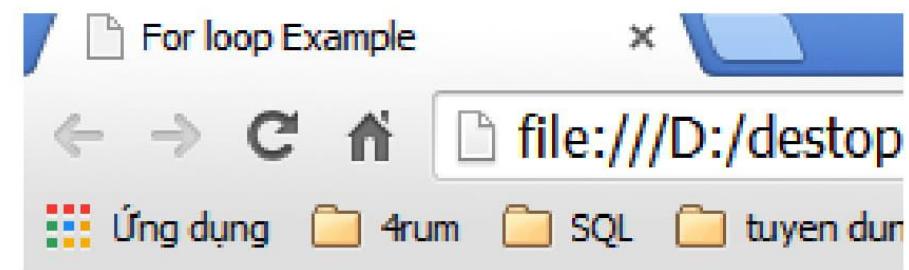
The properties of the Window object are:

top, window, location, external, chrome, document, x, speechSynthesis, localStorage, sessionStorage, applicationCache, webkitStorageInfo, indexedDB, webkitIndexedDB, crypto, CSS, performance, console, devicePixelRatio, styleMedia, parent, opener, frames, self, defaultstatus, defaultStatus, status, name, length, closed, pageYOffset, pageXOffset, scrollY, scrollX, scrollTop, screenLeft, screenY, screenX, innerWidth, innerHeight, outerWidth, outerHeight, offscreenBuffering, frameElement, clientInformation, navigator, toolbar, statusbar, scrollbars, personalbar, menubar, locationbar, history, screen, postMessage,

## a. BREAK, CONTINUE

VD:

```
<HTML> <HEAD>
<TITLE>For loop Example </TITLE>
<SCRIPT LANGUAGE= "JavaScript">
x=1;
while (x<=10){
    y=x*25;
    document.write("x="+x +"; y = "+ y + "<BR>");
    if (x==3)
    {
        x=6;
        continue;
    }
    if (x==8)
    {
        break;
    }
    x++;
}
</SCRIPT>
</HEAD>
<BODY></BODY>
</HTML>
```



## b. NEW

- Dùng để tạo ra một thể hiện mới của đối tượng.
- Cú pháp

`objectvar = new object_type(param1, param2...)`

## c. This

- Dùng để chỉ đối tượng hiện thời.
- Cú pháp

`this [.property]`

### d. WITH

- Dùng để thiết lập đối tượng ngầm định cho một nhóm các lệnh
- Cú pháp: **with (object){**

```
//statement  
}
```

- Vd:

```
<HTML>  
<HEAD>  
<TITLE>With Example </TITLE>  
<SCRIPT LANGUAGE= "JavaScript">  
with (document){  
write("This is an exemple of the things that can be done <BR>");  
write("With the <B>with<B> statement. <P>");  
write("This can really save some typing");  
}  
</SCRIPT>  
</HEAD>  
<BODY>  
</BODY>  
</HTML>
```



This is an exemple of the things that can be done  
With the with statement.

**This can really save some typing**

## V. FUNCTION

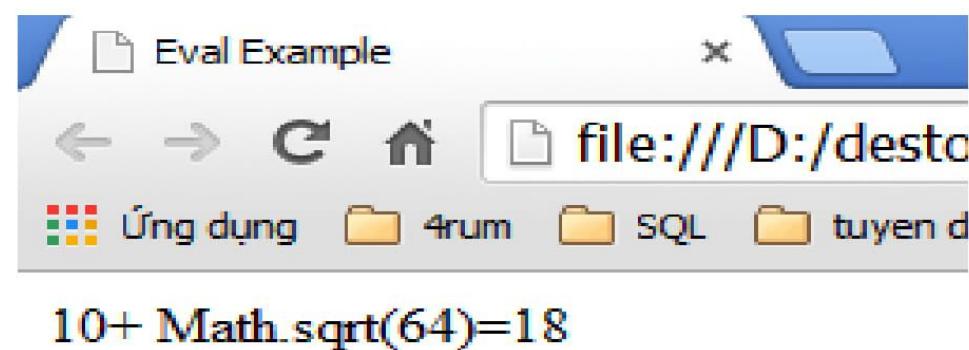
- JavaScript cũng cho phép sử dụng các hàm. Mặc dù không nhất thiết phải có, song các hàm có thể có một hay nhiều tham số truyền vào và một giá trị trả về.
- *Cú pháp*

```
function fnName([param1],[param2],...,[paramN])
{
//function statement
}
```
- Trong JavaScript có một số hàm có sẵn sau: EVAL , PARSEINT , FARSEFLOAT .

- Hàm được dùng để đánh giá biểu thức hay lệnh
- **Cú pháp:**  
`returnval=eval` (bất kỳ biểu thức hay lệnh hợp lệ trong Java)
- VD:
 

```

<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>
```

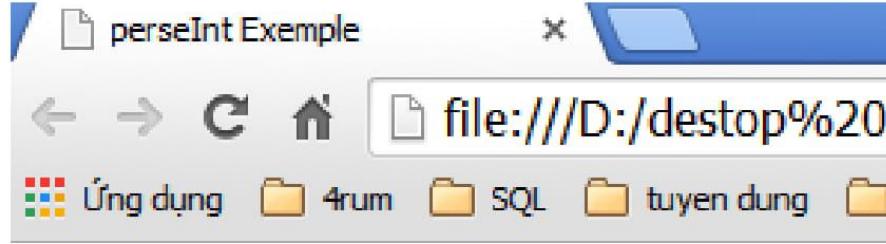


- Hàm chuyển đổi chuỗi thành số nguyên

- **Cú pháp**  
`parseInt (string [, radix])`

- **VD:**

```
<HTML>
<HEAD>
<TITLE> perseInt Exemple </TITLE>
<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>
</HEAD>
<BODY>
</BODY>
</HTML>
```



Converting 0xC hex to base-10: 12  
Converting 1100 binary to base-10: 12

## 5.3 FARSEFLOAT

- Hàm này tương tự như Parseint nhưng là chuyển đổi chuỗi thành số thực
- *Cú pháp*  
**parseFloat (string)**
- VD: tương tự như PARSEINT

- Cú pháp:

- `var array-name = [item1, item2, ...];`
- `var array-name = new Array(item1, item2, ...);`

- Example:

```
var cars = ["Saab", "Volvo", "BMW"];
document.getElementById("demo").innerHTML = cars[0];
```

- Notes:

```
// Creates an array with two elements (40 and 100)
var points = new Array(40, 100);
```

```
// Creates an array with 40 undefined elements
var points = new Array(40);
```

- **Array Properties and Methods:**

// The length property returns the number of elements in cars

```
var x = cars.length;
```

// The sort() method sort cars in alphabetical order

```
var y = cars.sort();
```

// adds a new element (Toyota) to cars, returns set new array length (is 4) to x

```
var x = cars.push("Toyota");
```

// Removes the last element (" Toyota ") from cars and set value of x is Toyota

```
var x = cars.pop();
```

// Removes the first element " Saab " from cars

```
cars.shift();
```

# VII. Sự kiện

## 7.1 OnBlur

- Xảy ra khi con trỏ rời khỏi form
- Vd:

```
<html>
```

```
<body>
```

Enter your name: <input type="text" id="fname" onblur="myFunction()">

<p>When you leave the input field, a function is triggered which transforms the input text to upper case.</p>

```
<script>
```

```
function myFunction() {
```

```
    var x = document.getElementById("fname");
```

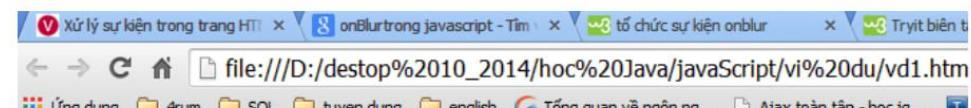
```
    x.value = x.value.toUpperCase();
```

```
}
```

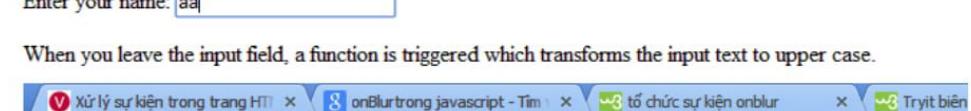
```
</script>
```

```
</body>
```

```
</html>
```



When you leave the input field, a function is triggered which transforms the input text to upper case.

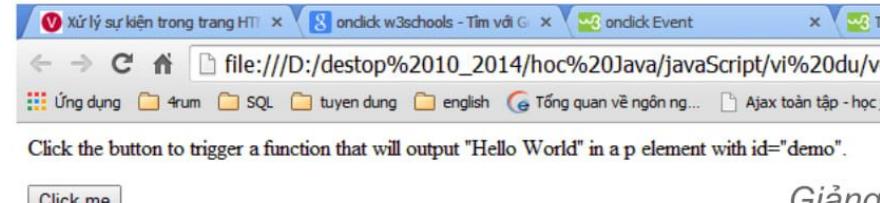
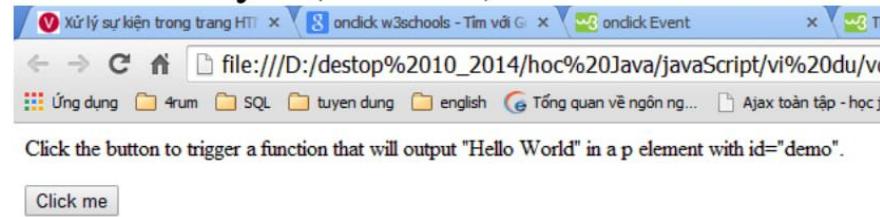


When you leave the input field, a function is triggered which transforms the input text to upper case.

## 7.2 OnClick

- Xảy ra khi người dùng kích vào các thành phần hay liên kết của form.
- **Vd:** <html>

```
<body>
<p>Click the button to trigger a function that will output "Hello World" in a p element
    with id="demo".</p>
<button onclick="myFunction()">Click me</button>
<p id="demo"></p>
<script>
function myFunction() {
    document.getElementById("demo").innerHTML = "Hello World";
}
</script>
</body>
</html>
```



## 7.3 OnChange

- Xảy ra khi giá trị của thành phần được chọn thay đổi
- Vd:

```

<html>
<body>
<p>Select a new car from the list.</p>
<select id="mySelect" onchange="myFunction()">
  <option value="Audi">Audi
  <option value="BMW">BMW
  <option value="Mercedes">Mercedes
  <option value="Volvo">Volvo
</select>
<p>When you select a new car, a function is triggered which outputs the value of the selected car.</p>
<p id="demo"></p>
<script>
function myFunction() {
  var x = document.getElementById("mySelect").value;
  document.getElementById("demo").innerHTML = "You selected: " + x;
}
</script>
</body>
</html>

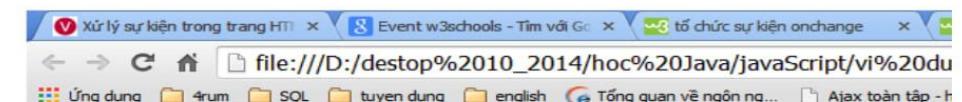
```



Select a new car from the list.

Audi ▾

When you select a new car, a function is triggered which outputs the value of the selected car.



Select a new car from the list.

BMW ▾

When you select a new car, a function is triggered which outputs the value of the selected car.

You selected: BMW

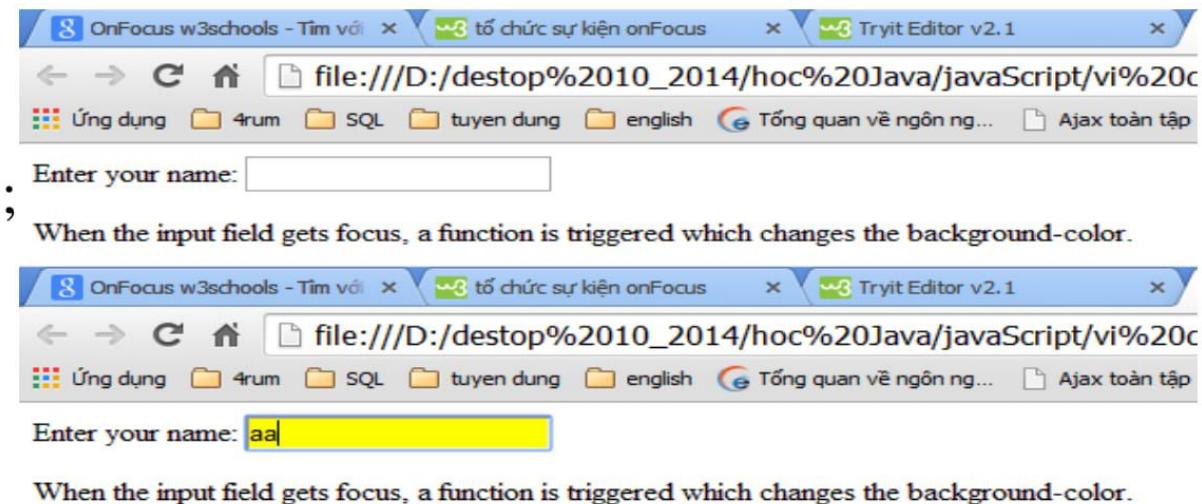
- Xảy ra khi thành phần của form được focus(làm nổi lên).
- Vd:

```
<html>
<body>
```

Enter your name: <input type="text" onFocus="myFunction(this)">

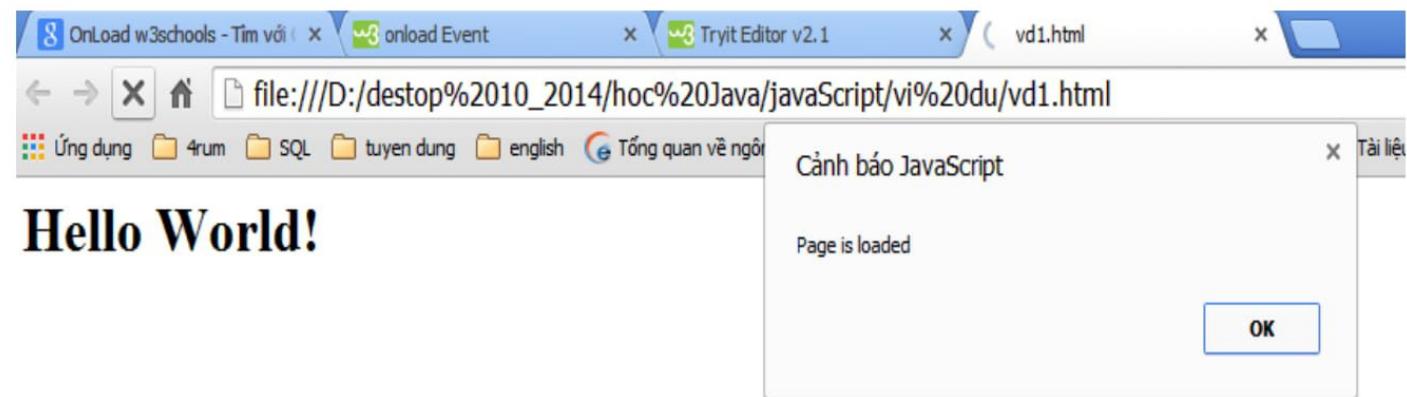
<p>When the input field gets focus, a function is triggered which changes the background-color.</p>

```
<script>
function myFunction(x) {
    x.style.backgroundColor = "yellow";
}
</script>
</body>
</html>
```



- Xảy ra trang Web được tải.
- Vd:

```
<html>
<body onload="myFunction()">
<h1>Hello World!</h1>
<script>
function myFunction() {
    alert("Page is loaded");
}
</script>
</body>
</html>
```



## 7.6 OnMouseOver

- Xảy ra khi di chuột qua kết nối hay anchor.
- Vd:

```
<html>
```

```
<body>
```

```

```

<p>The function bigImg() is triggered when the user moves the mouse pointer over the image.</p>

<p>The function normalImg() is triggered when the mouse pointer is moved out of the image.</p>

```
<script>
```

```
function bigImg(x) {
```

```
    x.style.height = "64px";
```

```
    x.style.width = "64px";
```

```
}
```

```
function normalImg(x) {
```

```
    x.style.height = "32px";
```

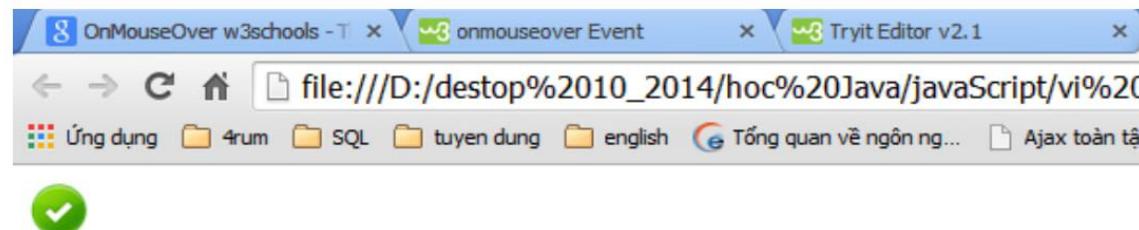
```
    x.style.width = "32px";
```

```
}
```

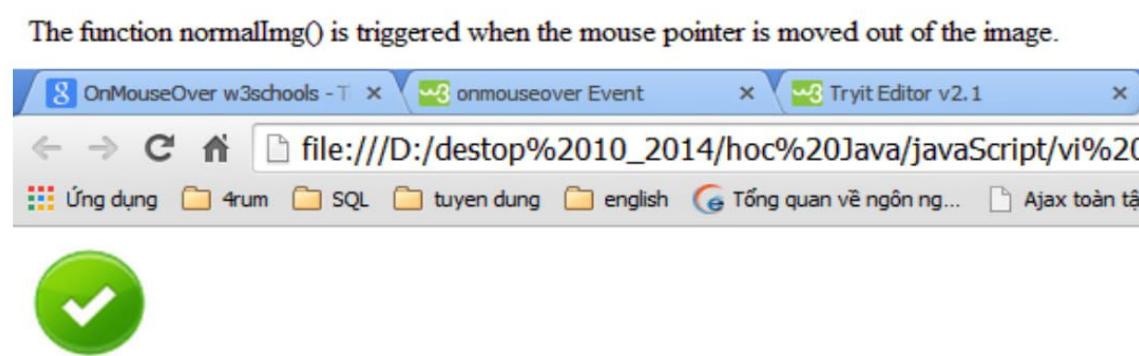
```
</script>
```

```
</body>
```

```
</html>
```



The function bigImg() is triggered when the user moves the mouse pointer over the image.



The function bigImg() is triggered when the user moves the mouse pointer over the image.

The function normalImg() is triggered when the mouse pointer is moved out of the image.

- Xảy ra khi người sử dụng lựa chọn một trường nhập dữ liệu trên form.
- Vd:

```
<html>
```

```
<body>
```

Select some of the text: <input type="text" value="Hello world!"  
onselect="myFunction()">

```
<script>
```

```
function myFunction() {
```

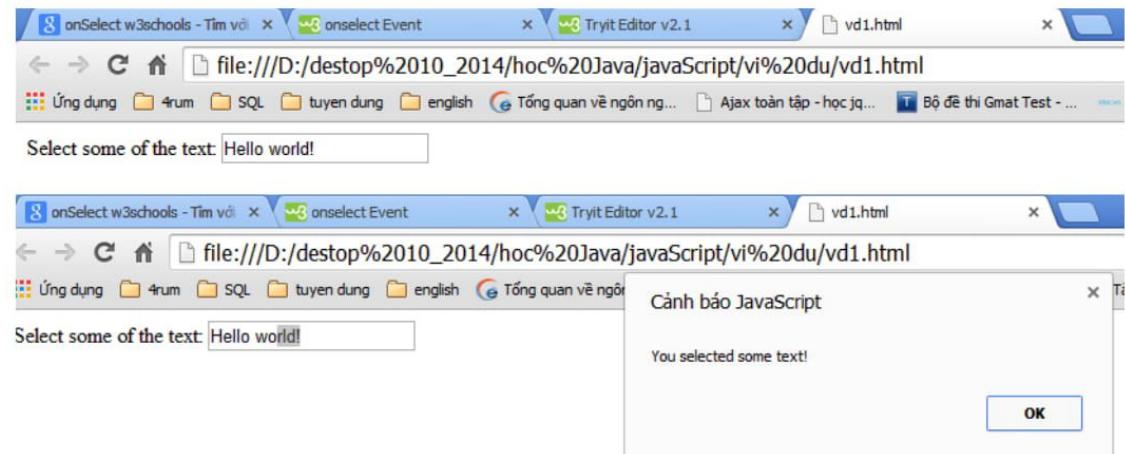
```
    alert("You selected some text!");
```

```
}
```

```
</script>
```

```
</body>
```

```
</html>
```



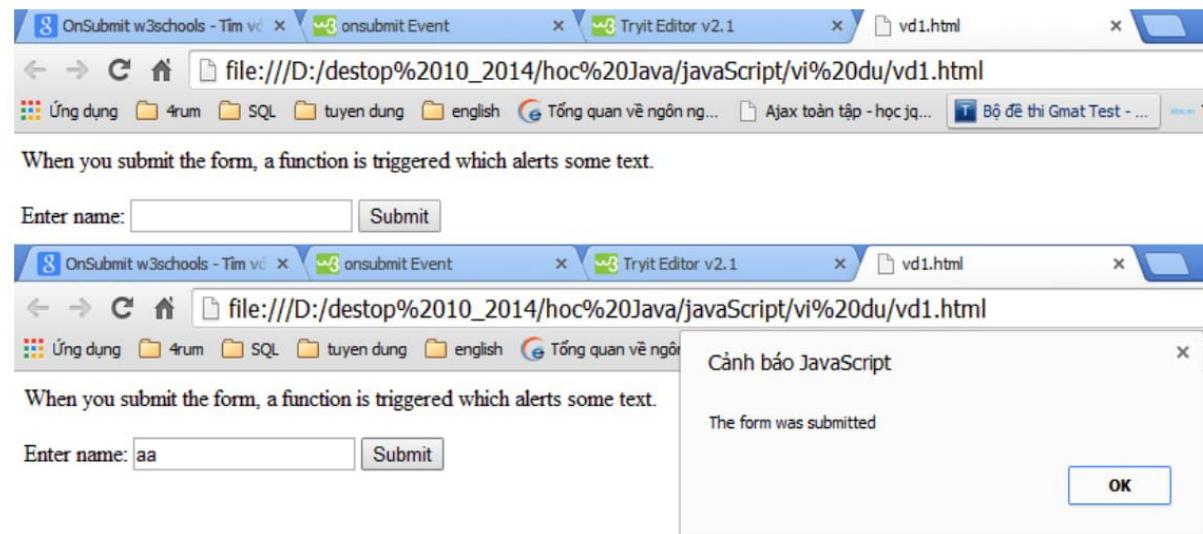
## 7.8 OnSubmit

- Xảy ra khi người dùng đưa ra một form.
- Vd:

```

<html>
<body>
<p>When you submit the form, a function is triggered which alerts some text.</p>
<form action="#" onsubmit="myFunction()">
    Enter name: <input type="text" name="fname">
    <input type="submit" value="Submit">
</form>
<script>
function myFunction() {
    alert("The form was submitted");
}
</script>
</body>
</html>

```



- Xảy ra khi người dùng đóng một trang

- Vd:

```
<html>
```

```
<body onunload="myFunction()">
```

```
<h1>Welcome to my Home Page</h1>
```

```
<p>Close this window or press F5 to reload the page.</p>
```

```
<p><strong>Note:</strong> Due to different browser settings, this event may now  
always work as expected.</p>
```

```
<script>
```

```
function myFunction() {
```

```
    alert("Thank you for visiting W3Schools!");
```

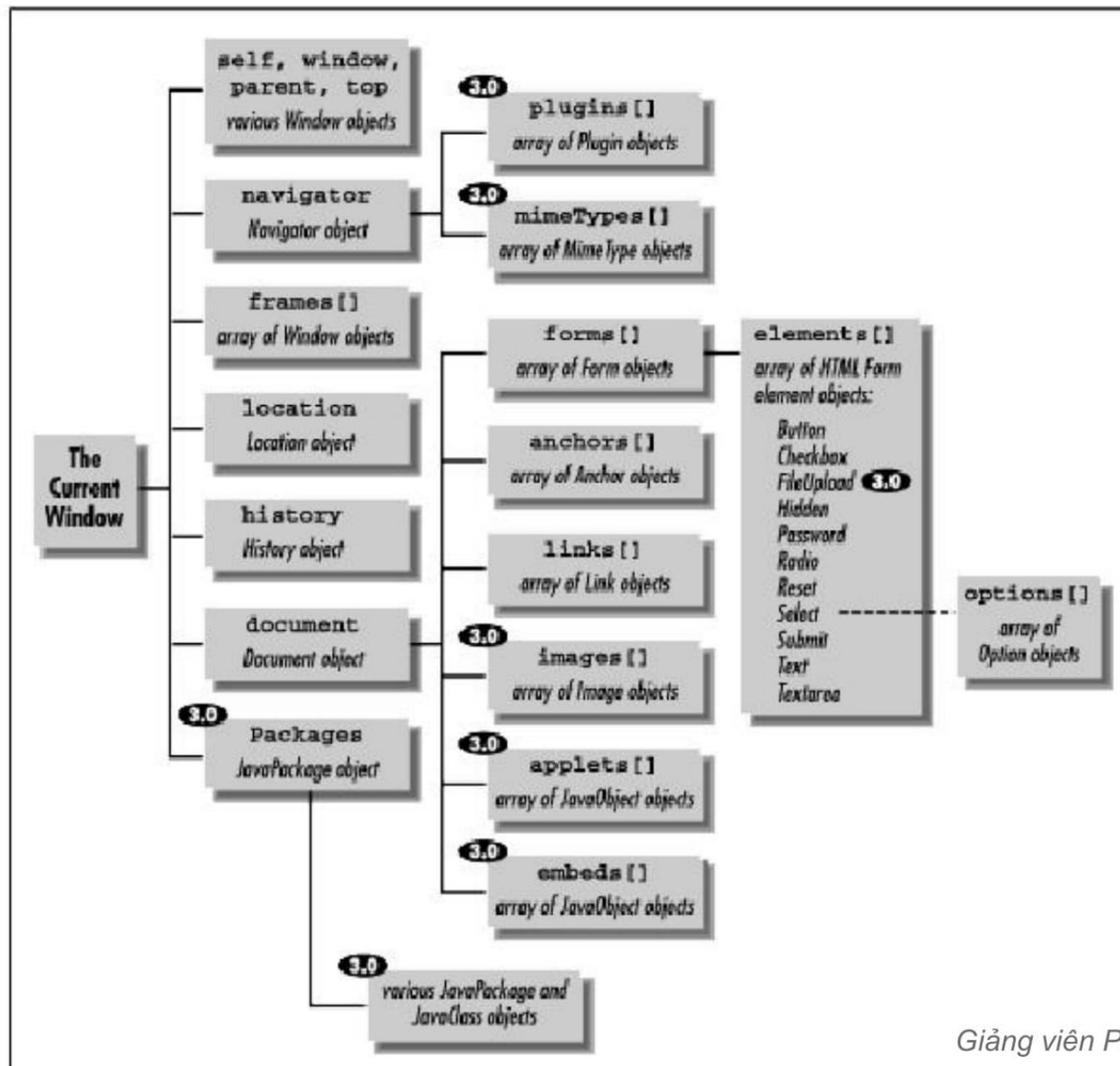
```
}
```

```
</script>
```

```
</body>
```

```
</html>
```

## 8. CÁC ĐỐI TƯỢNG TRONG JAVASCRIPT



### 8.1 ĐỐI TƯỢNG NAVIGATOR

- Đối tượng này được sử dụng để đạt được các thông tin về trình duyệt như số phiên bản.
- Đối tượng này không có phương thức hay chương trình xử lý sự kiện.
- Các thuộc tính
  - appCodeName: Xác định tên mã nội tại của trình duyệt (Atlas).
  - AppName: Xác định tên trình duyệt.
  - AppVersion: Xác định thông tin về phiên bản của đối tượng navigator.
  - userAgent: Xác định header của user - agent.

## 8.2 ĐỐI TƯỢNG WINDOW

### • CÁC THUỘC TÍNH

- DefaultStatus, Frames, Length, Name, Parent, Self , Status, Top, Window

### • CÁC PHƯƠNG THỨC

- alert ("message"); clearTimeout(timeoutID); windowReference.close; confirm("message"); [windowVar = ][window]. open("URL", "windowName", [ "windowFeatures" ] ); prompt ("message" [, "defaultInput"]); TimeoutID = setTimeout(expression,msec).

### • CÁC CHƯƠNG TRÌNH XỬ LÝ SỰ KIỆN

- onLoad - Xuất hiện khi cửa sổ kết thúc việc tải.
- onUnLoad - Xuất hiện khi cửa sổ được loại bỏ.

## 8.3 ĐỐI TƯỢNG LOCATION

- **Các thuộc tính**

- hash - Tên anchor của vị trí hiện thời.
- Host - Phần hostname:port của URL. Chú ý rằng đây thường là cổng ngầm định và ít khi được chỉ ra.
- Hostname - Tên của host và domain .
- href - Toàn bộ URL cho document hiện tại.
- Pathname - Phần đường dẫn của URL .
- Port - Cổng truyền thông được sử dụng cho máy tính host, thường là cổng ngầm định.
- Protocol - Giao thức được sử dụng (cùng với dấu hai chấm) .
- Search - Câu truy vấn tìm kiếm có thể ở cuối URL cho các script CGI.

- **CÁC THUỘC TÍNH**

- frames - Mảng tất cả các frame trong cửa sổ.
- Name - Thuộc tính NAME của thẻ <FRAME>
- Length - Số lượng các frame con trong một frame.
- Parent - Cửa sổ hay frame chứa nhóm frame hiện thời.
- self - frame hiện thời.
- Window - frame hiện thời.

- **CÁC PHƯƠNG THỨC**

- clearTimeout (timeoutID) - Xoá timeout do setTimeout lập. SetTimeout trả lại timeoutID.
- TimeoutID = setTimeout (expression,msec) - Đánh giá expression sau khi hết thời gian msec.

## 8.5 ĐỐI TƯỢNG DOCUMENT

- **CÁC THUỘC TÍNH**

- alinkColor, anchor, bgColor, cookie, fgColor, forms, lastModified, linkColor, links, location, referrer, title, vlinkColor,

- **- CÁC PHƯƠNG THỨC**

- document.clear - Xoá document hiện thời.
- document.close - Đóng dòng dữ liệu vào và đưa toàn bộ dữ liệu trong bộ đệm ra màn hình.
- document.open ("mineType") - Mở một stream để thu thập dữ liệu vào của các phuong thức write và writeln.
- document.write(expression1 [,expression2]...[,expressionN]) - Viết biểu thức HTML lên văn bản trong một cửa sổ xác định.
- document.writeln (expression1 [,expression2] ... [,expressionN] ) - Giống phuong thức trên nhưng khi hết mỗi biểu thức lại xuống dòng.

## 8.6 ĐỐI TƯỢNG HISTORY

- **CÁC THUỘC TÍNH**
  - length - Số lượng các URL trong đối tượng.
- **CÁC PHƯƠNG THỨC**
  - history.back()
  - history.forward()
  - history.go (delta | "location")

- **CÁC THUỘC TÍNH**

- hash - Tên anchor của vị trí hiện thời.
- Host - Phần hostname:port của URL. Chú ý rằng đây thường là cổng ngầm định và ít khi được chỉ ra.
- Hostname - Tên của host và domain.
- href - Toàn bộ URL cho document hiện tại.
- Pathname - Phần đường dẫn của URL.
- port - Cổng truyền thông được sử dụng cho máy tính host, thường là cổng ngầm định.
- Protocol - Giao thức được sử dụng (cùng với dấu hai chấm).
- Search - Câu truy vấn tìm kiếm có thể ở cuối URL cho các script CGI.
- Target - Giống thuộc tính TARGET của <LINK>.

- **CÁC CHƯƠNG TRÌNH XỬ LÝ SỰ KIỆN**

- onClick - Xảy ra khi người sử dụng nhấn vào link.
- onMouseOver - Xảy ra khi con chuột di chuyển qua link.

## 8.8 ĐỐI TƯỢNG MATH

- CÁC THUỘC TÍNH
  - E, LN2, LN10, LOG2E, PI, SQRT1\_2, SQRT2
- CÁC PHƯƠNG THỨC
  - Math.abs (*number*), Math.acos (*number*), Math.asin (*number*), Math.atan (*number*), Math.ceil (*number*), Math.cos (*number*), Math.exp (*number*), Math.floor (*number*), Math.log (*number*), Math.max (*num1,num2*), Math.min (*num1,num2*), Math.pow (*base,exponent*), Math.random (*r*), Math.round (*number*), Math.sin (*number*), Math.sqrt (*number*), Math.tan (*number*).

## • CÁC PHƯƠNG THỨC

- `dateVar.getDate()` - Trả lại ngày trong tháng (1-31) cho `dateVar`.
- `dateVar.getDay()` - Trả lại ngày trong tuần (0=chủ nhật,...6=thứ bảy) cho `dateVar`.
- `dateVar.getHours()` - Trả lại giờ (0-23) cho `dateVar`.
- `dateVar.getMinutes()` - Trả lại phút (0-59) cho `dateVar`.
- `dateVar.getSeconds()` - Trả lại giây (0-59) cho `dateVar`.
- `dateVar.getTime()` - Trả lại số lượng các mili giây từ 00:00:00 ngày 1/1/1970.
- `dateVar.getTimeZoneOffset()` - Trả lại độ dịch chuyển bằng phút của giờ địa phương hiện tại so với giờ quốc tế GMT.
- `dateVar.getYear()`-Trả lại năm cho `dateVar`.
- `Date.parse (dateStr)` - Phân tích chuỗi `dateStr` và trả lại số lượng các mili giây tính từ 00:00:00 ngày 01/01/1970.
- `dateVar.setDay(day)` - Đặt ngày trong tháng là `day` cho `dateVar`.
- `dateVar.setHours(hours)` - Đặt giờ là `hours` cho `dateVar`.
- `dateVar.setMinutes(minutes)` - Đặt phút là `minutes` cho `dateVar`.
- `dateVar.setMonths(months)` - Đặt tháng là `months` cho `dateVar`.
- `dateVar.setSeconds(seconds)` - Đặt giây là `seconds` cho `dateVar`.
- `dateVar.setTime(value)` - Đặt thời gian là `value`, trong đó `value` biểu diễn số lượng mili giây từ 00:00:00 ngày 01/01/1970.
- `dateVar.setYear(years)` - Đặt năm là `years` cho `dateVar`.
- `dateVar.toGMTString()` - Trả lại chuỗi biểu diễn `dateVar` dưới dạng GMT.
- `dateVar.toLocaleString()`-Trả lại chuỗi biểu diễn `dateVar` theo khu vực thời gian hiện thời.
- `Date.UTC (year, month, day [,hours] [,minutes] [,seconds])` - Trả lại số lượng mili giây từ 00:00:00 01/01/1970 GMT.

## • CÁC PHƯƠNG THỨC

- str.anchor(name) - Được sử dụng để tạo ra thẻ <A> (một cách động). Tham số name là thuộc tính NAME của thẻ <A>.
- str.big() - Kết quả giống như thẻ <BIG> trên chuỗi str.
- str.blink() - Kết quả giống như thẻ <BLINK> trên chuỗi str.
- str.bold() - Kết quả giống như thẻ <BOLD> trên chuỗi str.
- str.charAt(a) - Trả lại ký tự thứ a trong chuỗi str.
- str.fixed() - Kết quả giống như thẻ <TT> trên chuỗi str.
- str.fontcolor() - Kết quả giống như thẻ <FONTCOLOR = color>.
- str fontsize(size) - Kết quả giống như thẻ <Fontsize = size>.
- str.indexOf(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 trong chuỗi.
- str.italics() - Kết quả giống như thẻ <I> trên chuỗi str.
- str.lastIndexOf(srchStr [,index]) - Trả lại vị trí trong chuỗi str vị trí xuất hiện cuối cùng của chuỗi srchStr. Chuỗi str được tìm từ phải sang trái. Tham số index có thể được sử dụng để xác định vị trí bắt đầu tìm kiếm trong chuỗi.
- str.link(href) - Được sử dụng để tạo ra một kết nối HTML động cho chuỗi str. Tham số href là URL đích của liên kết.
- str.small() - Kết quả giống như thẻ <SMALL> trên chuỗi str.
- str.strike() - Kết quả giống như thẻ <STRIKE> 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.

## • CÁC THUỘC TÍNH

- action thuộc tính ACTION của thẻ FORM.
- elements Mảng chứa tất cả các thành phần trong một form
- 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

## • CÁC PHƯƠNG THỨC

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

## • CÁC CHƯƠNG TRÌNH XỬ LÝ SỰ KIỆN

- onSubmit - Chương trình xử lý sự kiện này được gọi khi người sử dụng chuyển dữ liệu từ form đi.