Thiết kế Web

Nguyễn Anh Tuấn

Bài 05:

Javascript

Giới thiệu về javascript

- Javascript ra đời với tên gọi LiveScript, sau đó Nescape đổi tên thành Javascript. Tuy nhiên giữa Java và Javascript có rất ít các điểm chung dù rằng cú pháp của chúng có thể có những điểm giống nhau.
- Ngôn ngữ Javascript được tạo bởi Nescape vào năm 1996 và được đưa vào trong trình duyệt Nescape Navigator 2.0 của họ thông qua trình biên dịch để đọc và thực hiện các mã lệnh Javascript được kèm theo trong các trang HTML..
- Javascript là một ngôn ngữ kịch bản (script) để viết kịch bản cho phía client. Client side là những yêu cầu của người sử dụng được xử lý tại máy khách. Thông thường những yêu cầu này là tính tóan, kiểm tra tính hợp lệ của dữ liệu hay các hiệu ứng, các yêu cầu này thường không liên quan đến nguồn cơ sở dữ liệu trên server.

- Đặc điểm của JAVASCRIPT:
  - Javascript là một ngôn ngữ kịch bản được viết chung với HTML.
  - Javascript là trình thông dịch.
  - Javascript là ngôn ngữ động vì các đối tượng có khả năng tương tác với nhau thông qua người sử dụng hoặc các sự kiện.
  - Là ngôn ngữ hướng đối tượng. Phân biệt chữ hoa, chữ thường
  - Được hỗ trợ bởi tất cả các trình duyệt như Nescape và Internet Explorer
  - JavaScript có khả năng tạo và sử dụng các đối tượng(Object)

- **Các đối tượng trong JavaScript gồm 2 nhóm**:
  - **Các object có sẵn trong JavaScript**

    JavaScript cung cấp một bộ các Built–in Object để cung cấp các thông tin về sự hiện hành của các đối tượng được load trong trang Web và nội dung của nó, các đối tượng này gồm phương thức (method) làm việc với các thuộc tính (properties) của nó.
  - **Các Object do người lập trình xây dựng**:

Định nghĩa thuộc tính, phương thức của đối tượng:

Cú pháp:

ObjectName.PropertiesName

ObjectName.Method()

Cấu trúc của đoạn javascript

- **Nhúng Javascript vào tập tin HTML**

```
<Script language="JavaScript">
    Các lệnh Javascript
</script>
```

  - Có thể viết nhiều đọan mã Javascript trong cùng một tập tin HTML.
  - Các khối mã Javascript có thể đặt bất kỳ vị trí nào trong trang HTML.

Vídu1:

```
<HTML>
<HEAD>
<script language="javascript" >
    document.write("What is your name? ");
</script>
</HEAD>
<BODY>
    Nội dung của trang
</BODY>
</HTML>
```

- **Sử dụng tập tin JavaScript bên ngoài:**
  - Có thể viết một tập tin Javascript riêng và sau đó kết nối với một hoặc nhiều tập tin trang web khác nhau.

  ***Cúpháp*:**

```
<HTML>
<BODY>
<Script SRC="fileJavascript.js" Language="javascript" >
    JavaScript program
</Script>
</BODY>
</HTML>
```

Lưu ý: trongthẻ JavaScript ta có thể bỏ thuộc tính ***SRC* và *Language***, khi đó ngôn ngữ mặc định là JavaScript .

- **Môi trường viết JAVASCRIPT:**
  - Frontpage
  - Notepad
  - Visual InterDev
  - Dreamweaver,...

Dreamweaver hỗ trợ phân biệt từ khóa bằng màu chữ, hỗ trợ các hàm, thuộc tính của các tag, giúp người sử dụng thuận tiên trong việc thiết kế và viết chương trình

Cú pháp cơ bản của lệnh

- **Lệnh đơn và khối lệnh:**
  - ***Lệnh đơn***: là một câu lệnh được kết thúc bằng dấu chấm phẩy**(;)**. Trong JavaScript cuối mỗi câu lệnh ta có thể dùng dấu (;) hoặc không dùng dấu gì cả .
  - ***Khối lệnh***: là tập hợp nhiều câu lệnh đơn được bao bọc bởi cặp dấu **{}**
- **Lời chú thích trong chương trình***:* trình duyệt sẽ bỏ qua khi thông dịch chương trình. JavaScript hổ trợ 2 loại chú thích:

  Chú thích trên một dòng: dùng cặp dấu ***//***

  Chú thích trên nhiều dòng: dùng cặp dấu ***/*…*/***

- **Xuất dữ liệu ra trang Web:** JavaScript hỗ trợ 2 phương thức hiển thị dữ liệu ra trang Web là:

  + document.write()

  + document.writeln()

  Nếu dữ liệu là chuỗi phải được đặt trong cặp nháy kép.

  Nếu xuất giá trị của biến thị không cần đặt trong nháy

  Có thể dùng dấu + để nối các chuỗi và biến

  ```
  document.write("String " + variable );
  ```

  Nếu xuất tag HTML thì cặp tag đó cũng phải đặt trong cặp dấu nháy kép

Vídụ:

```
<BODY >
<Script Language="JavaScript"> document.write
("Trường ĐH TĐT TP.HCM");
document.write("<H1> Trường ĐH TĐT TP.HCM
   <H1>");
a ="ĐH TĐT TP.HCM"
document.write("Trường " + a);
</Script>
</BODY>
```

Sử dụng document.writeln() và tag <pre>:

- Dùng với tag <pre>

```
<pre>
 document.writeln()
</pre>
```

xuất dữ liệu và xuống dòng

- Nếu không có cặp tag <pre></pre> document.writeln() tạo một khoảng trắng

Vídu:

```
<body>
<pre>
<script>
   document.writeln("<b>Hello</b>");
   document.writeln("<b>World</b>");
</script>
</pre>
</body>
```

- **Biến**

  Biến là tên của một phần tử trong chương trình, được sử dụng để lưu trữ thông tin do người dùng nhập vào hoặc kết quả trung gian của quá trình tính toán

  Trong Javascript khi khai báo biến không cần xác định kiểu dữ liệu cho biến, do đó khi một biến được khai báo thì nó có thể chứa bất kỳ kiểu dữ liệu nào.

- **Cách khai báo biến**: Trong JavaScript, để khai báo biến dùng từ khoá **var,** cũng có thể bỏ qua từ khóa var.

```
var VariableName;
```

Ví dụ:

```
var a ;
```

Hoặc `a=5;`//khai báo và khởi tạo

Biến

- Lưu ý:
  - Một biến có thể được khai báo và khởi tạo hoặc không khởi tạo giá trị ban đầu
  - Muốn khai báo nhiều biến cùng một lúc thì liệt kê tên biến kế tiếp nhau cách nhau bởi dấu (,)
  - Một biến có thể chứa bất kỳ kiểu dữ liệu nào, giá trị của biến có tác dụng từ vị trí khai báo trở đi

Ví dụ:

```
var a="Hello World";
a=1999 ;
```

Cách xuất giá trị của biến:

document.write(VariableName)

Ví dụ:

```
var a="Hello World";
a=1999 ;
document.write(a)
```

Quy tắc đặt tên biến:

- Tên biến gồm các chữ cái và số, không dùng các ký tự đặc biệt như: (, [, { , # , & …. theo nguyên tắc sau:
  - Tên biến phải bắt đầu bằng ký tự hoặc ký tự gạch dưới(_)
  - Không bắt đầu bằng ký tự số.
  - Không chứa khoảng trắng, tên biến nên gợi nhớ
  - Không trùng với từ khoá của JavaScript

Các từ khoá trong JavaScript

abstract	extends	Int	super
boolean	false	interface	switch
break	final	Long	synchronized
byte	finally	native	this
case	float	new	throw
catch	for	null	throws
char	Function	package	transient
class	goto	private	true
const	if	protected	try
continue	implements	public	var
default	import	return	val
do	In	short	while
double	instanceof	static	with
else			

Biến

- **Tầm vực của biến**: là tầm ảnh hưởng của biến trong chương trình. Có 2 loại biến:
  - **Biến toàn cục** : được khai báo ngoài các hàm. Phạm vi hoạt động của biến là từ vị trí khai báo trở về sau trong chương trình.
  - **Biến cục bộ**: được khai báo trong chương trình con. Phạm vi hoạt động của biến là từ vị trí khai báo đến kết thúc chương trình con.

Lưu ý: Nếu tên biến toàn cục và cục bộ trùng nhau thì biến được sử dụng trong hàm là biến cục bộ.

Kiểu dữ liệu

- **Dữliệu: Có 4 loại dữ liệu**
  - ***Kiểu số***: một biến kiểu số chứa bất kỳ giá trị số nào: số thập phân, số nguyên, số dạng chấm phẩy động.
  - ***Kiểu chuỗi***: một biến kiểu chuỗi có thể chứa một nhóm ký tự (Chữ cái, ký tự số, khoảng trắng, các ký tự đặc biệt, …). Giá trị chuỗi phải đặt trong cặp dấu nháy đôi (“ “) hoặc đơn (‘ ‘)
  - ***Kiểu Boolean***: Là dữ liệu chỉ có 2 giá trị false hoặc true, thường dùng trong trường hợp biến hoặc hàm chỉ nhận một trong 2 trạng thái đúng hoặc sai.
  - ***Kiểu null***: trả về giá trị rỗng

Toán tử: số học

Tóan Tử	Chức Năng	Ví dụ	Kết quả
+	cộng	x=2; x+2	4
-	Trừ	x=2; 5-x	3
*	Nhân	x=4; x*5	20
/	Chia	5/2	2.5
%	Chia lấy phần dư	5%2	1
++	Tăng 1	x=5; x++	6
--	Giảm 1	x=5; x--	x=4

Toán tử: số học

Toán tử gán

Tóan Tử	Ví dụ	Tương đương
=	x = y	x= y
+=	x += y	x = x+y
-=	x -= y	x = x-y
*=	x *= y	x = x*y
/=	x /= y	x= x/y
%=	x%=y	x = x%y

Toán tử gán

Toán tử so sánh

Tóan Tử	Chức Năng	Ví dụ
= =	bằng	5==8 returns false
!=	Không bằng	5!=8 returns true
>	lớn hơn	5>8 returns false
<	nhỏ hơn	5<8 returns true
>=	lớn hợn hoặc bằng	5>=8 returns false
<=	nhỏ hơn hoặc bằng	5<=8 returns true

Toán tử so sánh

Toán tử logic

Tóan Tử	Chức Năng	Ví dụ
&&	Và	x =6; y =3 ; (x < 10 && y > 1) returns true
||	hoặc	x = 6 ; y =3 (x==5 || y==5) returns false
!	not	x=6; y =3; !(x==y) returns true

Toán tử chuỗi

- Ký hiệu **+** : Là phép toán nối hai chuỗi
- Ví dụ:

```
<html>
<script>
    txt1="Welcome to"; txt2="JavaScript!";
    document.write('<h2>'+txt1+txt2+'</h2>');
</script>
</html>
```

Toán tử điều kiện

- **Cú pháp: (Điều kiện) ? value1: value2**
  - Nếu biểu thức điều kiện đúng thì trả về giá trị value 1
  - Nếu biểu thức điều kiện sai thì trả về giá trị value 2
- **Ví dụ:**

```
<html>
<script>
    document.write((day="Saturday")? "Weekend":
    "Not Saturday")
</script>
</html>
```

Một số ký tự đặc biệt

- \n : new line
- \t : tab
- \b : BackSpace
- \& : dấu &
- \”: dấu “

Vídụ:

```
<html>
<script>
    document.write ("You \& i sing \"Happy Birthday\".")
</script>
</html>
```

Function - Hàm

Định nghĩa

- Hàm là một đoạn chương trình có thể được sử dụng nhiều lần trong một chương trình để thực hiện một tác vụ nào đó.
- **Cách xây dựng hàm:** Dùng từ khoá ***function*** để định nghĩa hàm.
- **Cú pháp:**

```
function  Tên_hàm ( [danh sách tham số] )
{
   Khai báo các biến sử dụng trong hàm ;
   Các câu lệnh trong JavaScript thực hiện tác vụ;
[return [giá trị /biểu thức] ];
}
```

- ***Tên_hàm**:* do người lập trình đặt. Qui tắc đặt tên hàm giống như tên biến.
- Sau *Tên_hàm* là cặp dấu ngoặc ( ) chứa danh sách tham số hình thức.
- Nếu hàm không có tham số thì sau Tên_hàm cũng phải có cặp dấu ngoặc ( )
- *[danh sách tham số]:* danh sách các tham số hình thức, nếu có nhiều tham số có thì các tham số phải cách nhau bởi dấu phẩy, các tham số này không chỉ ra kiểu dữ liệu cụ thể và cũng không cần từ khoá var.

- *Câu lệnh **return:*** để kết thúc hàm. nếu hàm có giá trị trả về thì return để trả về giá trị
- Sau Return có thể chứa hoặc không chứa một giá trị cụ thể hoặc một biểu thức tính toán.

Vídu:

```
function  Display(user , pwd)
{
   document.write("Ten cua ban la:" + user) ;
   document.write("Mat khau cua ban la:" + pwd) ;
   return ;
}
```

Gọi hàm

- Hàm sẽ không thực hiện cho đến khi nó được gọi.
- Đối với hàm có đối số ta gọi tên hàm và danh sách các giá trị truyền cho đối số đó

  FunctionName(argument1,argument2,etc)
- Đối với hàm không có đối số ta chỉ cần gọi tên hàm

  FunctionName()
- Đối với hàm không có giá trị trả về :

  NameFunction(parameter) .
- Đối với hàm có giá trị trả về :phải gán giá trị trả về cho biến

  variable= NameFunction(parameter)

Ví dụ:

```
<html>
<head><title>Function</title></head>
<body>
   <script>
    function  tong(a , b)
    {
            c=a+b;
            document.write(c);
    }
    tong(2,3);
   </script>
</body></html>
```

Một số hàm thông dụng trong Javascript

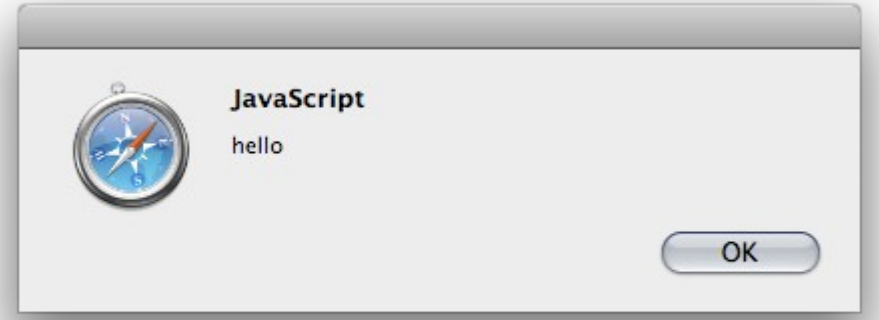
- **Hàmalert():** dùng hiển thị một hộp thông báo có nút OK

  Cúpháp:

  alert("nội dung thông báo")

  Vídu:

```
<html><head><title>Function</title></head>
<body>
<script>
   alert("hello");
</script>
</body></html>
```



Một số hàm thông dụng trong Javascript

Một số hàm thông dụng trong Javascript

- **Hàmprompt():** Tạo hộp thoại chứa 2 nút OK ,Cancel và một textbox để người sd nhập nội dung, giá trị trả về của hàm prompt là nội dung nhập trong textbox

Cúpháp:

prompt("nội dung",giá trị khởi tạo);

vídụ:

```
<script>
a=prompt("Your Lastname:"); b=prompt("Your
FirstName:"); document.write("Your FullName is
:"+ a + ' ' + b)
```

Một số hàm thông dụng trong Javascript

</script>

Một số hàm thông dụng trong Javascript

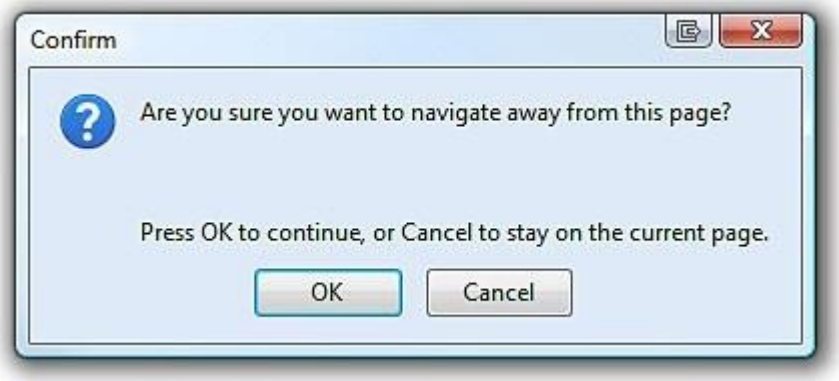
- **Hàm confirm():** Hiển thị hộp thông báo có 2 nút OK và Cancel. Hàm trả về giá trị true nếu người sử dụng click OK và ngược lại thì trả về giá trị false.

Cúpháp:

```
variable=confirm("Chuoi thong bao");
```

Ví dụ:

```
<script>
    a=prompt("nhap so a :");
    b=prompt("nhap so b");
    c=confirm( a +' lon hon '+ b+'?
    if(c == true)
     document.write( a +" > "+b  )
    else
     document.write( a +" < "+b  )
</script>
```



Một số hàm thông dụng trên chuỗi & số

- **Hàm eval():** Trả về giá trị số của một chuỗi số

Cúpháp:

eval(chuỗi số)

Vídụ :

```
<script>
  var str1="123", str2="456";
   str= str1+str2;
  document.write(str); //kết quả :123456
</script>
```

Ví dụ:

```
<script>
    a = eval(prompt("Nhập số a:"));
    b = eval(prompt("Nhập số b:"));
    c = a+b ;
    document.write(c)
</script>
```

Một số hàm thông dụng trên chuỗi & số

Hàm ParseInt(strNum)

- Trả về một số nguyên từ chuỗi **strNum**.
- Nếu **strNum** theo sau là ký tự chữ thì các ký tự này sẽ bị bỏ qua.
- Nếu strNum không bắt đầu bằng số thì hàm này trả về giá trị NaN (Not a Number)

Vídu :

```
var  strNum="123.8" , kq;
kq=parseInt(strNum) //kq=123
strNum="a123"
kq=parseInt(strNum) //kq=NaN
strNum="123.8abc"
kq=parseInt(strNum) //kq=123
```

Một số hàm thông dụng trên chuỗi & số

- **Hàm parseFloat(strNum)**:
  - Hàm trả về một số thực từ chuỗi **strNum**.
  - Nếu chuỗi **strNum** bắt đầu là số và theo sau là các ký tự chữ thì các ký tự này bị bỏ qua.
  - Nếu chuỗi **strNum** bắt đầu từ ký tự chữ thì hàm trả về giá trị NaN.

Vídụ:

```
var  strNum="123.8" , kq;
kq=parseFloat(strNum) //kq=123.8
strNum="a123.8"
kq=parseFloat(strNum) //kq=NaN
strNum="123.8abc"
kq=parseFloat(strNum) //kq=123.8
```

Một số hàm thông dụng trên chuỗi & số

- **Hàm isNaN(str):**

  Hàm trả về giá trị True nếu str là chuỗi, ngược lại là False nếu str là chuỗi số.

Ví dụ :

```
var str="123abc", kq;
kq=isNaN(str); //kq=true
str="123.8"
kq=isNaN(str); //kq=false
```

Các hàm thiết lập thời gian

- **Hàm setTimeout( )**: Báo cho JavaScript thực hiện một lệnh JavaScript sau một khoảng thời gian nào đó.
  - Hàm trả về một ID (duy nhất đối với mỗi hàm setTimeout thực hiện một lệnh)
  - Giá trị ID này dùng để xoá khoảng thời gian đã thiết lập nếu không cần thực hiện hàm Timeout nữa

Cúpháp:

```
IdTime=setTimeout("Command JavaScript",
                  delayTime);
```

- *Command JavaScript* : có thể là lời gọi hàm hoặc là một câu lệnh đơn
- *delayTime* :là khoảng thời gian chờ để thi hành Command JavaScript, được tính bằng mili giây.

Các hàm thiết lập thời gian

Vídụ:

```
Idq=setTimeout("alert('I love you')",1000) ;
```

Cứ 1000 mili giây thì thông báo "I love you" một lần.

- **Hàm clearTimeout()**:Huỷ thời gian đã thiết lập bởi setTimeout().

Cúpháp:

```
clearTimeout(IdTime);
```

Vídụ:

```
clearTimeout(Idq);
```

- **Hàm setInterval() và clearInterval():**

Ý nghĩa và tham số giống như setTimeout() và clearTimeout() .

Thank You!