JavaScript

Mục tiêu

- Hiểu ngôn ngữ JavaScript trong thiết kế và lập trình web
- Biết cách sử dụng các qui tắc và cú pháp của ngôn ngữ HTML vào việc hỗ trợ thiết kế giao diện và thiết kế xử lý của trang web
- Biết cách gắn kết giữa ngôn ngữ JavaScript, các hàm JavaScript với các thẻ HTML

Nội dung

- Khái niệm về ngôn ngữ Script
- Ngôn ngữ JavaScript

Nội dung

- Khái niệm về ngôn ngữ Script
- JavaScript

Khái niệm ngôn ngữ Script

- Là ngôn ngữ kịch bản
- Giúp trang web có tính tương tác với người dùng
- Các ngôn ngữ script thông dụng:
  - JavaScript (Netscape)
  - Jscript (Microsoft)
  - VBScript (Microsoft)

Script - Ứng dụng Client-Side và Server-Side

Ứng dụng Client-Side:

- Thực hiện tại Browser (Netscape Navigator, IE,Firefox,…)
- Script tại Client-Side (Thực hiện các tương tác với người dùng, thay đổi cấu trúc trang web, kiểm tra dữ liệu được nhập vào của người dùng, …)

Ứng dụng Server-Side:

- Thực hiện tại WebServer (IIS, Netscape Enterprise Server, ….)
- Script tại Server-Side (kết nối CSDL, chia sẽ thông tin giữa các người duyệt web, truy cập hệ thống file trên server, …)

Script - Quá trình thực hiện Script tại server

- Quá trình thực hiện ứng dụng Server-Side gồm 2 giai đoạn:
  - Tạo trang Web có chứa cả Srcipt Client-Side và Script Server-Side
  - Khi Client browser yêu cầu thực hiện, server (run-time engine) sẽ thực hiện các lệnh Server-side Scipts và trả trang Web HTML về browser

Nội dung

- Khái niệm về ngôn ngữ Script
- JavaScript

Ngôn ngữ JavaScript

- Khái niệm JavaScript
- Nhúng JavaScript vào web
- Quy ước trong JavaScript
- Biến và khai báo biến
- Kiểu dữ liệu
- Toán tử
- Cấu trúc điều khiển
- Hàm
- Một số đối tượng dữ liệu
- Lớp đối tượng

Khái niệm JavaScript

- Với HTML chúng ta đã biết cách tạo ra trang Web – tuy nhiên chỉ mới ở mức biểu diễn thông tin chứ chưa phải là các trang Web động có khả năng đáp ứng các sự kiện từ phía người dùng.
- Hãng Netscape đã đưa ra ngôn ngữ script có tên là LiveScript để thực hiện chức năng này.
- Sau đó được đổi tên thành JavaScipt để tận dụng tính đại chúng của ngôn ngữ lập trình Java.

Khái niệm JavaScript

- JavaScript là ngôn ngữ kịch bản dùng để tạo các client-side scripts và server-side scripts.
- JavaScript làm cho việc tạo các trang Web động và tương tác dễ dàng hơn
- Các ứng dụng client chạy trên một trình duyệt như Netscape Navigator hoặc Internet Explorer.

Khả năng của Javascript

- JavaScript có thể tăng cường tính động và tính tương tác của các trang web.
  - Cung cấp sự tương tác người dùng
  - Thay đổi nội dung động
  - Xác nhận tính hợp lệ của dữ liệu

Khái niệm JavaScript

- Java và JavaScript là hai ngôn ngữ hoàn toàn khác nhau
  - Java
    - Là ngôn ngữ lập trình hướng đối tượng
    - Được phát triển bởi hãng Sun Microsystems
  - JavaScript
    - Là ngôn ngữ kịch bản WEB
    - Được phát triển bởi Netscape

Nhúng JavaScript vào web

- Nhúng trực tiếp

```
<script type="text/javascript">
<!--
        // LệnhJavascript
-->
</script>
```

- Nhúng script từ 1 file khác

```
<script src="xxx.js"></script>
```

Nhúng JavaScript vào web

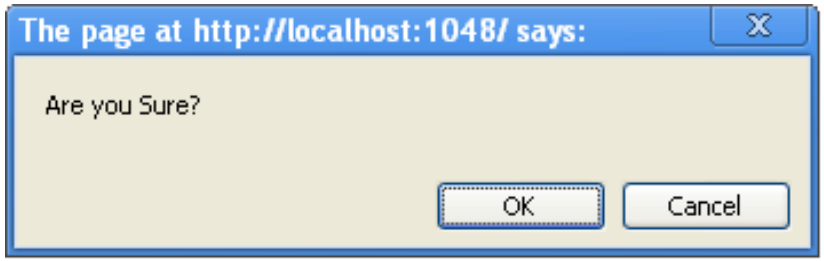
```
<html>
<head>
        <script type = "text/javascript">
                some  statements
        </script>
</head>
<body>
        <script type="text/javascript">
                some statements
        </script>
        <script src = "Tên_file_script.js"> </script>
</body>
</html>
```

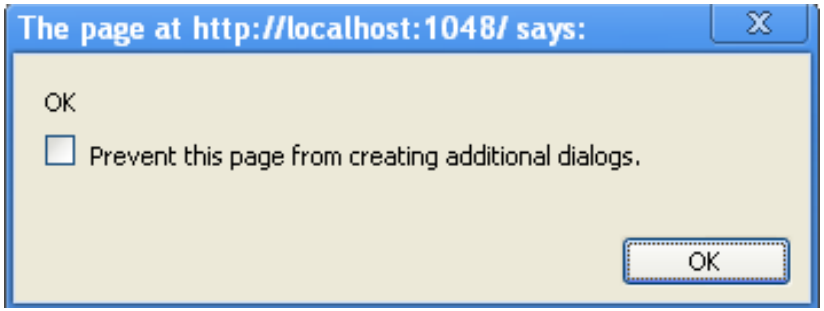
Nhúng JavaScript vào web

- Đặt giữa tag <head> và </head>: đoạn script sẽ thực thi ngay khi trang web được mở.
- Đặt giữa tag <body> và </body>: Đoạn script trong phần body được thực thi khi trang web đang mở (sau khi thực thi các đoạn script có trong phần<head>).
- Số lượng đoạn script không hạn chế.

Ví dụ - Nhúng JavaScript

```
<HTML>
    <HEAD>
     <SCRIPT LANGUAGE = "Javascript">
         confirm("Are you Sure?");
         alert("OK");
         document.write(" Thank You !");
     </SCRIPT>
   </HEAD>
</HTML>
```





Thank You !

Qui tắt trong JavaScript

- JavaScript phân biệt chữ hoa, chữ thường
  - Vídụ: hai biến Java, java là khác nhau
- Câu lệnh JavaScript đều được kết thúc bằng dấu: “ ; ”
- Không phân biệt khoảng trắng, tab, xuống dòng trong câu lệnh.

Qui tắc trong JavaScript

- Chuỗi và dấu nháy
  - Chuỗi trong JavaScript được đặt trong cặp nháy đơn (’) hoặc nháy kép (“”)
  - Vídụ:

    <input value = **‘He said “JavaScriptis good”’>**

    <input type=“button”value=“Click Me!” onclick = **“alert(‘Hello’);”**
- Ghi chú: theo cú pháp ghi chú sau

```
// Đây là ghi chú trên 1 dòng
/* Đây là ghi chú
        trên nhiều dòng*/
```

Qui tắt trong JavaScript

- Tên biến và hàm:
  - Bắt đầu bằng Ký tự (A..Z, a..z), _, $
  - Không bắt đầu bằng ký số (0..9)
  - Không có khoảng trắng trong biến hoặc hàm
  - Không đặt tên trùng từ khóa

Danh sách từ khóa trong

abstract
alert
arguments
Array
blur
boolean
Boolean
break
byte
callee
caller
captureEvents
case
catch
char
class
clearInterval
clearTimeout
close
closed
confirm
const
constructor
continue
Date
debugger
default
defaultStatus
delete
do
document
double
else
enum
escape
eval
export
extends
final
finally
find
float
focus
for
frames
Function
function
goto
history
home
if
implements
import
in
Infinity
innerHeight
innerWidth
instanceof
int
interface
isFinite
isNaN
java
length
location
locationbar
long
Math
menubar
moveBy
moveTo
name
NaN
native
netscape
new
null
Number
Object
open
opener
outerHeight
outerWidth
package
Packages
pageXOffset
pageYOffset
parent
parseFloat
parseInt
personalbar
print
private
prompt
protected
prototype
public
RegExp
releaseEvents
resizeBy
resizeTo
return
routeEvent
scroll
scrollbars
scrollBy
scrollTo
self
setInterval
setTimeout
short
static
status
statusbar
stop
String
super
switch
synchronized
this
throw
throws
toolbar
top
toString
transient
try
typeof
unescape
unwatch
valueOf
var
void
watch
while
window
with
FALSE
TRUE

Biến - JavaScript

- Mỗi biến sẽ có một tên đại diện
- Biến là “vật chứa” thông tin muốn lưu trữ
- Thông tin, dữ liệu của biến có thể thay đổi trong quá trình đoạn lệnh thực hiện
- Sử dụng tên biến để tham chiếu đến dữ liệu của biến đó

Khai báo biến - JavaScript

- Dùng từ khóa var trước tên biến
- Biến có thể khởi tạo giá trị trước hoặc không hoặc không cũng được

  var strname = *value ;*

  - *Tên biến: strname*
  - *Giá trị được gán cho biến: value*
- Ví dụ:

  var strname = *“Hello Word!”;*

Khai báo biến - JavaScript

- Ví dụ:

```
var x ;
var y, sum ;
var x = 1, y = -10, sum = 0;
```

- Biến trong JavaScript có thể lưu **giá trị có kiểu dữ liệu bất kỳ**

Kiểu dữ liệu - JavaScript

Kiểu dữ liệu	Ví dụ	Mô tả
Object	var listBooks = new Array(10) ;	Trước khisử dụng, phải cấp phát bằng từ khóa new
String	“Chúc các bạn thành công.” “10”	Chứa được chuỗi unicode Chuỗi rỗng “”
Number	0.066218 12	Theo chuẩn IEEE 754
boolean	true / false	
undefined	var myVariable ;	myVariable = undefined
null	connection.Close();	connection = null
function	var add = new function("x", "y", "return(x+y)"); add(2, 3);	*functionName* ***= new function([argname1, [... argnameN,]] body);***

Kiểu dữ liệu - JavaScript

- Kiểu số nguyên
  - Các số nguyên có thể được biểu diễn trong hệ thập phân (cơ số 10), hệ thập lục phân (cơ số 16) và hệ bát phân (cơ số 8)
  - Một chữ số nguyên thập phân gồm có một dãy các số mà không có số 0 đứng đầu.
  - Một số 0 đứng đầu trong một chữ số nguyên cho biết nó được biểu diễn trong hệ bát phân
  - Nếu đứng đầu một chữ số nguyên là 0x (hoặc 0X) chỉ ra nó được biểu diễn trong hệ thập lục phân

Kiểu dữ liệu - JavaScript

- Kiểu số nguyên
  - Số nguyên thập phân bao gồm các số từ 0 đến 9.
  - Số nguyên thập lục phân có thể bao gồm các số từ 0 đến 9 và các chữ cái từ a đến f và A đến F.
  - Số nguyên bát phân bao gồm các số từ 0 đến 7.
  - Các chữ số nguyên bát phân không được tán thành và đã bị loại khỏi chuẩn ECMA-262 ấn bản 3.
  - JavaScript vẫn hỗ trợ các chữ số nguyên bát phân để tương thích với các phiên bản trước.

Kiểu dữ liệu - JavaScript

- Kiểu số nguyên
  - Ví dụ về số nguyên:

    42

    0xFFF

    -345

Kiểu dữ liệu - JavaScript

- Kiểu số thực (kiểu số dấu chấm động)
  - Kiểu số thực có thể có các thành phần sau:
    - Phần nguyên thập phân (là một số nguyên thập phân)
    - Một dấu chấm thập phân (“.”)
    - Phần dư (là một số thập phân khác)
    - Phần mũ

Kiểu dữ liệu - JavaScript

- Kiểu số thực (kiểu số dấu chấm động)
  - Trong đó phần số mũ là một chữ “e” hay “E”, theo sau là một số nguyên, có thể được đánh dấu (được đặt trước bởi dấu “+” hoặc “-”).
  - Một số dấu chấm động phải có ít nhất một con số và một dấu chấm thập phân hoặc “e” (hay “E”).

Kiểu dữ liệu - JavaScript

- Kiểu số thực (kiểu số dấu chấm động)
  - Ví dụ về số thực:
    - 3.114
    - -3.1E12
    - .1e12
    - 2E-12

Kiểu dữ liệu - JavaScript

- Kiểu Logical (hay Boolean)
  - Kiểu logic được sử dụng để chỉ hai điều kiện: đúng hoặc sai.
  - Miền giá trị của kiểu này chỉ có hai giá trị:
    - true.
    - false.

Kiểu dữ liệu - JavaScript

- Kiểu chuỗi (String)
  - Một chuỗi chữ gồm không hoặc nhiều ký tự được đặt trong các dấu nháy kép (“”) hoặc nháy đơn (‘’).
  - Một chuỗi phải được phân định bởi các dấu trích dẫn cùng kiểu, tức là cả hai dấu đều phải là dấu nháy đơn hoặc đều là dấu nháy kép

Kiểu dữ liệu - JavaScript

- Kiểu chuỗi (String)
  - Ví dụ về các chuỗi:
    - “Hello”
    - ‘Error!’
    - “12345”
  - Ta có thể gọi bất cứ một phương thức nào của đối tượng String trên một giá trị chuỗi chữ - JavaScript sẽ tự động chuyển đổi chuỗi chữ thành một đối tượng String tạm, gọi phương thức được yêu cầu, sau đó loại bỏ đối tượng String tạm.

Kiểu dữ liệu - JavaScript

- Kiểu chuỗi (String)
  - Khi dùng chuỗi, ngoài các ký tự thông thường, ta cũng có thể chèn các ký tự đặc biệt vào chuỗi đó. Các ký tự đặc biệt sẽ thực hiện một công việc cụ thể nào đó.

    Ví dụ: “one line \n another line”
  - Trong ví dụ trên, dấu “\” kết hợp với ký tự “n” sẽ mang ý nghĩa là sang dòng. Như vậy khi thực hiện câu lệnh trên thì kết quả sẽ hiển thị là:

    one line

    another line

Kiểu dữ liệu - JavaScript

- Kiểu chuỗi (String)

Ký tự	Ý nghĩa
\b	Phím lùi (Backspace)
\f	Sang trang mới (Form feed)
\n	Sang dòng mới (new line)
\r	Đưa con trỏ về đầu dòng hiện tại
\t	Cách một khoảng Tab (Tab)

Kiểu dữ liệu - JavaScript

- Kiểu chuỗi (String)
  - Ngoài ra, có thể chèn một số ký tự đặc biệt khác trong một chuỗi bằng cách đặt trước nó dấu backslash (\).
  - Đây được xem là ký tự thoát (*escaping character*).
  - Dấu backslash được dùng để bỏ qua ý nghĩa sử dụng của ký tự đứng sau nó.
  - Ví dụ nếu muốn hiển thị các ký tự ‘, “ hay \ trong chuỗi thì phải đặt dấu backslash ở phía trước, đó là \’, \” và \\.

Kiểu dữ liệu - JavaScript

- Kiểu null
  - Kiểu null chỉ có duy nhất một giá trị: null.
  - Null mang ý nghĩa là không có dữ liệu, nó thực hiện chức năng là giữ chỗ trong một biến với ý nghĩa là ở đó không có hữu dụng gì.
  - Số 0 hay một xâu rỗng và null là các giá trị khác nhau

Toán tử toán học

Operator	Description	Example	Result
+	Addition	x=2 x+2	4
-	Subtraction	x=2 5-x	3
*	Multiplication	x=4 x*5	20
/	Division	15/5 5/2	3 2.5
%	Modulus (division remainder)	5%2 10%8 10%2	1 2 0
++	Increment	x=5 x++	x=6
--	Decrement	x=5 x--	x=4

Toán tử gán

Operator	Example	Is The Same As
=	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 so sánh

Operator	Description	Example
==	is equal to	5==8 returns false
!=	is not equal	5!=8 returns true
>	is greater than	5>8 returns false
<	is less than	5<8 returns true
>=	is greater than or equal to	5>=8 returns false
<=	is less than or equal to	5<=8 returns true

Toán tử logic

Operator	Description	Example
&&	and	x=6 y=3 (x < 10 && y > 1) returns true
||	or	x=6 y=3 (x==5 || y==5) returns false
!	not	x=6 y=3 !(x==y) returns true
?:	If .. Else	x=6 y=3 x>y ? return x : return y

Toán tử nối chuỗi

- Khi cần nối hai chuỗi ta sử dụng toán tử cộng “+“:
- Ta xét ví dụ sau:

  txt1 = “What a very”

  txt2 = “nice day!”

  txt3 = txt1 + txt2
- Khi đó txt3 có giá trị là: "What a verynice day!"
- Để thêm khoảng trắng vào giữa 2 giá trị, ta nối thêm chuỗi là ký tự khoảng trắng vào giữa

Toán tử nối chuỗi

txt1 = "What a very"

txt2 = "nice day!"

txt3 = txt1 + “ “ + txt2

Hoặc

txt1 = "What a very "

txt2 = "nice day!"

txt3 = txt1 + txt2

Cấu trúc điều khiển

- Rẽ nhánh: if
- Chọn: switch
- Lặp xác định: for
- Lặp không xác định: while, do .. while
- Lặp for … in
- Lệnh break, continue

Rẽ nhánh If

- Cú pháp

if (biểu thức điều kiện) {

các câu lệnh;

//câu lệnh được thực nếu điều kiện là đúng (true)

}

- Ví dụ

```
<script type="text/javascript">
//If the time on your browser is less than 10,
//you will get a "Good morning" greeting.
var d=new Date() ;
var time=d.getHours() ;
if (time<10) {
    document.write("<b>Good morning</b>") ;
}
</script>
```

Rẽ nhánh If … Else

```
if ( biểu thức điều kiện)
{
    …….
    <Lệnh i>
    …….
}
else
{
    ……..
    <Lệnh j>
    ……
}
```

- Nếu *biểu thức điều kiện* cho giá trị là đúng thì Lệnh i được thực hiện, Lệnh j bỏ qua
- Ngược lại Lệnh j được thực hiện, Lệnh i bỏ qua

Ví dụ: Rẽ nhánh If … Else

- Ví dụ 1:

```
<script type="text/javascript">
//If the time on your browser is less than 10,
//you will get a "Good morning" greeting.
//Otherwise you will get a "Good day“
greeting.var d = new Date()
var time = d.getHours()
if (time < 10)
{
    document.write("Good morning!")
}
else
{
    document.write("Good day!")
}
</script>
```

- Ví dụ 2:

```
var x = 5, y = 6, z;
1. if(x == 5) {
            if(y == 6) z = 17;
    }
    else
            z = 20;
2. if(x == 5) {
            z = 7;
            y = 42;
    }
    else
            z = 19;
```

Rẽ nhánh Switch

```
switch (tên biến){
    case giá trị 1:
        các câu lệnh 2;
        break;
    case giá trị 2:
        các câu lệnh 2;
        break;
    defaut:
        các câu lệnh khác;
        break;
}
```

- Chúng ta nên sử dụng switch thay cho if … else khi có nhiều điều kiện để lựa chọn
- Tên biến là biến đã có giá trị sẵn
- Giá trị 1, 2, … là giá trị của biến được truyền vào
- Cách thức xử lý: khi giá trị của biến được truyền vào thì cấu trúc switch sẽ so sánh từng case, nếu giá trị của case nào thỏa thì các câu lệnh của case đó được thực hiện và sau đó gặp break sẽ thoát khỏi switch

Ví dụ Rẽ nhánh Switch

- Ví dụ 1:

```
<script type="text/javascript">
var d=new Date();
theDay=d.getDay();
switch (theDay)
 {
case 5: document.write(“Friday"); break ;
case 6: document.write(“Saturday"); break ;
case 0: document.write(“Sunday“); break ;
default: document.write("I'm looking forward
     to this weekend!") ;
}
</script>
```

- Ví dụ 2:

```
var diem = “G”;
switch(diem)
{
case “Y”:document.write(“Yếu");break;
case “TB”:document.write(“Trungbình");break;
case “K”:document.write(“Khá");break;
case “G”:document.write(“Giỏi");break;
default :document.write(“Xuấtsắc");
}
```

Lặp xác định for

- Cú pháp:

for (<biểu thức khởi tạo>; <biểu thức điều kiện>; <biểu thức thay đổi>)

- biểu thức khởi tạo: giá trị bắt đầu lặp
- biểu thức điều kiện: điều kiện để dừng lặp
- biểu thức thay đổi: giá trị thay đổi

- Cách thức xử lý: biểu thức thay đổi sẽ thay đổi giá trị sau mỗi lần lặp cho đến khi nào giá trị thay đổi này vi phạm biểu thức điều kiện thì vòng lặp dừng

Ví dụ lặp for

```
var  sum = 0, count = 0;
for ( var icount = 0; icount < 10; icount ++)
{
   sum = sum + icount;
   count = count + 1
}
```

Lặp không xác định do … while

- Cú pháp:

```
do {
   các câu lệnh;
} while (biểu thức
   điều kiện)
```

- Các câu lệnh thi hành một lần trước khi điều kiện được kiểm tra.
- Nếu điều kiện là true, thì câu lệnh được thi hành một lần nữa.
- Mỗi lần thi hành vòng lặp, điều kiện được kiểm tra. Khi điều kiện là false, thì ngừng và thoát khỏi lặp do … while.

Lặp không xác định do … while

- Ví dụ 1:

```
var i = 0;
do {
   i += 1;
   document.write (i);
} while (i<5)
```

- Ví dụ 2:

```
var i = 9, total = 0;
do{
   total += i * 3 + 5;
   i = i +5;
} while(i > 10);
```

Lặp không xác định While

- Cú pháp:

```
while (điều kiện) {
    các câu lệnh;
}
```

- Lệnh while được dùng để thực hiện một khối lệnh khi nào điều kiện là true.
- Nếu có nhiều câu lệnh thực hiện trong thân của vòng lặp, phải sử dụng cặp dấu ngoặc móc ({}) để chứa các câu lệnh đó.
- Khác biệt chính giữa vòng lặp while và do…while là các lệnh trong thân vòng lặp while có thể không được thực hiện một lần nào vì nó kiểm tra điều kiện trước, và có thể ngay từ ban đầu điều kiện đã là false.

Lặp không xác định While

- Ví dụ 1:

```
var n = 0;
var x = 0;
while (n <3)
{
   n++;
   x += n;
}
```

- Ví dụ 2:

```
var i = 9, total = 0;
while (i < 10)
{
total += i * 3 + 5;
i = i +5;
}
```

Hàm - Javascript

- Khai báo hàm hay phương thức
  - Mỗi hàm hay phương thức do người dùng định nghĩa được bắt đầu với từ khóa *function*
  - Tham số truyền vào hàm hay phương thức không cần phải khai báo kiểu dữ liệu

Hàm - Javascript

- Cách định nghĩa một hàm:
  - Hàm có tham số:

```
function TenHam(thamso1, thamso2,..)
{
  // Nội dung hàm
}
```

  - Hàm không tham số

```
function TenHam(thamso1, thamso2,..)
{
  // Nội dung hàm
}
```

Hàm - Javascript

– Hàm trả về giá trị:

```
function TenHam(thamso1, thamso2,..)
{
// Nội dung hàm
return (value);
}
```

Ví dụ hàm

- Định nghĩa hàm:

```
function Sum(x, y)
{
    tong = x + y;
    return tong;
}
```

- Gọi hàm:

```
var x = Sum(30, 40);
```

Một số hàm thông dụng trong JavaScript

- Hàm eval
- Hàm isFinite
- Hàm isNaN
- Hàm parseInt và parseFloat
- Hàm Number và String

Một số hàm thông dụng trong JavaScript

- Hàm eval
  - Dùng để đánh giá một chuỗi mà không cần tham chiếu đến bất kì một đối tượng cụ thể nào.
  - Cú pháp:

    eval(string)
  - Với string là chuỗi cần được đánh giá. Chuỗi này có thể là một biểu thức JavaScript, một câu lệnh, hay một nhóm các câu lệnh.
  - Trong biểu thức có thể bao gồm các biến và thuộc tính của một đối tượng.

Một số hàm thông dụng trong JavaScript

- Hàm eval
  - Nếu chuỗi đại diện cho một biểu thức thì hàm eval định giá trị biểu thức đó.
  - Nếu đối số đại diện cho một hoặc nhiều câu lệnh JavaScript, thì hàm eval thực hiện các câu lệnh này.
  - Không dùng hàm eval để định giá trị một biểu thức số học; JavaScript định giá trị các biểu thức số học một cách tự động.

Một số hàm thông dụng trong JavaScript

- Hàm isFinite
  - Hàm isFinite định giá trị một đối số để xác định xem nó có phải là một số hữu hạn hay không.
  - Cú pháp:

    isFinite(number)
  - Với number là số được định giá trị.
  - Nếu đối số là NaN, dương vô cùng hoặc âm vô cùng, phương thức này trả về false, ngoài ra nó trả về true.

Một số hàm thông dụng trong JavaScript

- Đoạn mã nguồn sau kiểm tra đối số ClientInput để xác định xem nó có phải là số hữu hạn không:

```
if (isFinite(ClientInput) == true)
{
/* các bước cụ thể*/
}
```

Một số hàm thông dụng trong JavaScript

- Hàm **isNaN**
  - Hàm isNaN định giá trị một đối số để xác định xem nó có phải là “NaN” (Not a Number) hay không.
  - Cú pháp:

    isNaN(testValue)
  - Với testValue là giá trị bạn muốn định giá trị.
  - Các hàm parseInt và parseFloat trả về “NaN” khi chúng định giá trị một giá trị không phải là một số.
  - Hàm isNaN trả về true nếu nó được truyền giá trị “NaN” và ngược lại là false.

Một số hàm thông dụng trong JavaScript

- Đoạn mã nguồn sau định giá trị floatValue để xác định xem nó có phải là một số hay không và sau đó gọi một thủ tục phù hợp:

```
var floatValue = parseFloat (toFloat)
if (isNaN (floatValue)) {
    notFloat()
}
else {
    isFloat()
}
```

Đối tượng

- Thuộc tính (biến) dùng để định nghĩa đối tượng và các phương thức (hàm) tác động tới dữ liệu đều nằm trong đối tượng.
- Ví dụ: một chiếc xe hơi là một đối tượng. Các thuộc tính của nó là cấu tạo, kiểu dáng và màu sắc. Hầu hết các chiếc xe hơi đều có một vài phương thức chung như go(), brake(), reverse().

Thuộc tính và phương thức

- Để truy cập thuộc tính của đối tượng, chúng ta phải chỉ ra tên đối tượng và thuộc tính của nó:

```
objectName.propertyName
```

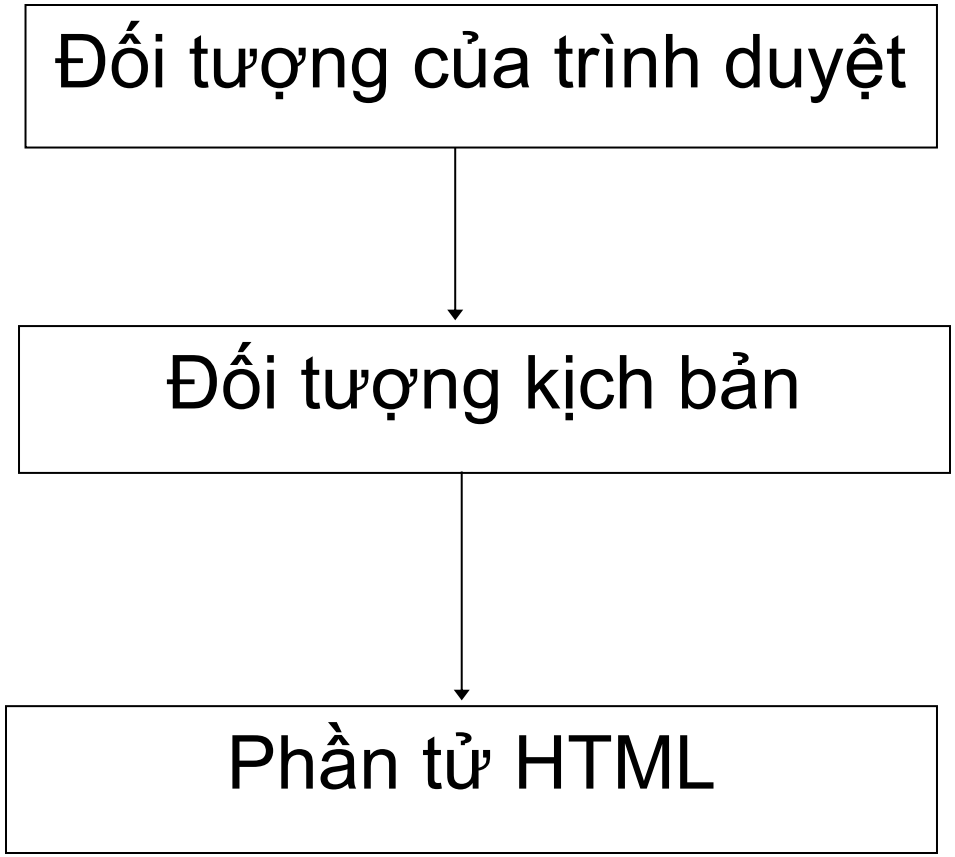
- Để truy cập phương thức của đối tượng, chúng ta phải chỉ ra tên đối tượng và thuộc tính của nó:

```
objectName.method()
```

Cách dùng đối tượng

- Khi tạo trang web, chúng ta cần chèn:
  - Các đối tượng trình duyệt
  - Các đối tượng có sẵn (thay đổi phụ thuộc vào ngôn ngữ kịch bản được sử dụng)
  - HTML elements
- Chúng ta cũng có thể tạo ra các đối tượng để sử dụng theo yêu cầu của mình.

Cây phân cấp đối tượng



Câu lệnh this

- Câu lệnh '**this**' không chỉ là một thuộc tính nội tại.
- Giá trị của nó chỉ ra đối tượng hiện hành và có thể có các thuộc tính chuẩn chẳng hạn như tên, độ dài, và giá trị được áp dụng phù hợp.

Câu lệnh for . . . in

- Câu lệnh For...in được dùng để lặp mỗi thuộc tính của đối tượng hoặc mỗi phần tử của một mảng.
- Cú pháp:

```
for (variable in object)
    { statements; }
```

Câu lệnh with

- Câu lệnh with được dùng để thực thi tập hợp các lệnh mà các lệnh này dùng các phương thức của cùng một loại đối tượng.
- thuộc tính được gán cho đối tượng đã được xác định trong câu lệnh with.
- Cú pháp:

```
with (object) { statements; }
```

Toán tử new

- Toán tử new được dùng để tạo ra một thực thể mới của một loại đối tượng
- Đối tượng có thể có sẵn hoặc do người dùng định nghĩa
- objectName = new objectType (param1 [,param2] ...[,paramN])

  Trong đó:

  objectName là tên của thực thể đối tượng mới.

  ObjectType là một hàm quyết định loại của đối tượng. Ví dụ Array.

  Param[1, 2, . .] là các giá trị thuộc tính của đối tượng.

Đối tượng String

- Đối tượng string được dùng để thao tác và làm việc với chuỗi văn bản.
- Chúng ta có thể tách chuỗi ra thành các chuỗi con và biến đổi chuỗi đó thành các chuỗi hoa hoặc thường trong một chương trình.
- Cú pháp tổng quát:

```
stringName.propertyName     hay
stringName.methodName
```

Cách tạo đối tượng String

- Có 3 phương thức khác nhau để tạo ra chuỗi.
  - Dùng lệnh var và gán cho nó một giá trị.
  - Dùng một toán tử (=) có gán với một tên biến.
  - Dùng hàm khởi tạo String (string).

Đối tượng Math

- Đối tượng Math có các thuộc tính và phương thức biểu thị các phép tính toán học nâng cao.

```
function doCalc(x) {
var a;
a = Math.PI * x * x;
alert ("The area of a circle with a radius of " + x + " is " + a);
}
```

Đối tượng Date

- Date là một đối tượng có sẵn chứa thông tin về ngày và giờ.
- Đối tượng Date không có thuộc tính nào.
- Đối tượng Date có nhiều phương thức dùng để thiết lập, lấy và xử lý các thông tin về thời gian.

Đối tượng Date

- Đối tượng Date lưu trữ thời gian theo số mili giây tính từ 1/1/1970 00:00:00

```
DateObject = new Date(parameters);
```

Tài liệu tham khảo thêm

- http://www.w3school.com/js
- http://vi.wikipedia.org/wiki/JavaScript