

# Mục lục

5.1	GIỚI THIỆU .....	4
5.1.1	Giới thiệu .....	4
5.1.2	Đặc điểm .....	4
5.1.3.	Khả năng của JavaScript.....	4
5.1.4	Các phiên bản(version) của JavaScript .....	5
5.2.	NHÚNG JAVASCRIPT VÀO TRANG HTML.....	6
5.2.1	Nhúng các câu lệnh Javascript trực tiếp vào trong tài liệu HTML.....	6
5.2.2	Sử dụng một file JavaScript riêng .....	7
5.2.3	Kích hoạt javascript từ các sự kiện .....	8
5.3.	CÁC CÂU LỆNH CỦA JAVASCRIPT .....	9
5.3.1	Khởi lệnh .....	9
5.3.2	Lệnh hiển thị dữ liệu ra cửa sổ trình duyệt .....	9
5.3.3	Một sổ hiển thị thông báo (popup box) .....	9
5.3.4	Lời chú thích (comments) .....	11
5.4.	BIẾN TRONG JAVASCRIPT .....	11
5.4.1	Quy tắc đặt tên biến .....	11
5.4.2	Khai báo biến .....	11
5.4.3	Phạm vi của biến.....	11
5.5.	CÁC KIỂU DỮ LIỆU CỦA JAVASCRIPT.....	12
5.5.1.	Giới thiệu .....	12
5.5.2.	Phân loại .....	12
5.5.2.1.	Kiểu dữ liệu dạng số (Number).....	13
5.5.2.2	Dữ liệu kiểu logic (Boolean).....	13
5.5.2.3	null và undefined .....	13
5.5.2.4	Dữ liệu kiểu xâu kí tự (String) .....	14
5.5.2.4	Dữ liệu kiểu Date/Time.....	15
5.5.2. 5.	Dữ liệu kiểu mảng (Array).....	18
5.5.2.5.	Đối tượng Math.....	23
5.6.	CÁC TOÁN TỬ TRONG JAVASCRIPT .....	24

5.6.1	Phép gán .....	24
5.6.2	Phép so sánh.....	24
5.6.3	Các phép toán số học cơ bản.....	24
5.6.4	Các phép toán logic cơ bản.....	25
5.6.5	Các phép toán trên chuỗi kí tự .....	25
5.7.	<b>CÁC CÂU LỆNH ĐIỀU KHIỂN DỮ LIỆU TRONG JAVASCRIPT</b> .....	26
5.7.1.	Các lệnh điều kiện .....	26
5.7.2.	Các lệnh lặp.....	28
5.7.3.	Câu lệnh break .....	29
5.7.4.	Câu lệnh for...in.....	30
5.7.5.	Câu lệnh with .....	30
5.8.	<b>HÀM TRONG JAVASCRIPT (FUNCTIONS)</b> .....	31
5.8.1.	Khái niệm .....	31
5.8.2.	<i>Cú pháp tạo 1 hàm:</i> .....	31
5.8.3.	<i>Một số hàm có sẵn</i> .....	32
5.9.	<b>SỰ KIỆN TRONG JAVASCRIPT</b> .....	33
5.9.1.	<i>Các sự kiện cơ bản của JavaScript</i> .....	33
5.9.2.	<i>Liên kết một sự kiện với 1 hàm javascript</i> .....	34
5.10.	<b>ĐỐI TƯỢNG TRONG JAVASCRIPT</b> .....	34
5.10.1.	<i>Sơ đồ phân cấp</i> .....	34
5.10.2.	<i>Đối tượng Window</i> .....	35
5.10.3.	<i>Location</i> .....	36
5.10.4.	<i>Frame</i> .....	37
5.10.5.	<i>Document</i> .....	38
5.10.6.	<i>Forms</i> .....	39
5.10.7.	<i>History</i> .....	40
5.10.8.	<i>Links</i> .....	40
5.11.	<b>LÀM VIỆC VỚI FORMS</b> .....	41
5.11.1	<i>Thuộc tính type</i> .....	42
a.	<i>Phần tử Button</i> .....	42
5.11.2	<i>Phần tử Text</i> .....	43

5.11.3	Phần tử Checkbox .....	44
5.11.4	Phần tử File Upload.....	46
5.11.5	Phần tử Hidden .....	46
5.11.6	Phần tử Password .....	46
5.11.7	Phần tử Radio.....	46
5.11.8	Phần tử Select .....	48
5.11.9	Phần tử Reset .....	50
5.11.10	Phần tử Submit.....	51
5.11.11	Phần tử Textarea.....	51
	BÀI TẬP CUỐI CHƯƠNG .....	52
	TÀI LIỆU THAM KHẢO.....	57

## Chương 5. NGÔN NGỮ KỊCH BẢN JAVASCRIPT TRONG LẬP TRÌNH WEB

Với HTML, chúng ta đã biết cách tạo ra giao diện Web, tuy nhiên chỉ mới ở mức biểu diễn thông tin chứ chưa phải là các trang Web có khả năng đáp ứng các sự kiện từ phía người dùng. Hãng Netscape đã đưa ra ngôn ngữ kịch bản JavaScript để thực hiện chức năng đáp ứng các sự kiện từ phía người dùng của trang web.

---

Copyright © 2014 Hoàng Thị Hà - Học Viện Nông nghiệp Việt Nam. All Right Reserved.

Lưu hành nội bộ

### 5.1 GIỚI THIỆU

#### 5.1.1 Giới thiệu

JavaScript là một ngôn ngữ lập trình Web được hãng Netscape (nay là Mozilla) và hãng Sun Microsystems (nay là Oracle) tạo ra và phát triển. Đa số các trang web hiện đại đều sử dụng JavaScript. Tất cả các trình duyệt hiện đại chạy trên máy các desktop, máy tính bảng, điện thoại thông minh,...đều có bộ thông dịch JavaScript. Điều đó, làm cho JavaScript là ngôn ngữ lập trình phổ biến nhất trong lịch sử. JavaScript là một phần của bộ ba của công nghệ mà tất cả các nhà phát triển Web phải học: HTML để xác định nội dung của các trang web, CSS để xác định trình bày của các trang web, và JavaScript để xác định hành vi của các trang web.

#### 5.1.2 Đặc điểm

JavaScript được ra đời để cung cấp khả năng tương tác với người dùng và tạo ra các hiệu ứng đặc biệt cho các trang web. JavaScript có một số đặc điểm sau đây:

- ✓ Là ngôn ngữ lập trình kịch bản dựa trên đối tượng.
- ✓ Là một ngôn ngữ được thông dịch và thực hiện tại các trình duyệt.
- ✓ Code JavaScript được nhúng trực tiếp vào trong trang HTML.
- ✓ Là ngôn ngữ không định kiểu
- ✓ Phân biệt chữ hoa, chữ thường.

#### 5.1.3. Khả năng của JavaScript

JavaScript dùng để phát triển các ứng dụng client-side và **server-side**. Tuy nhiên, nó được biết đến nhiều hơn với chức năng là một ngôn ngữ kịch bản phía Client. Các ứng dụng client chạy trong các trình duyệt. Hiện nay, hầu hết các trình duyệt web như: Netscape Navigator, Firefox, Internet Explorer, Chrome...đều hỗ trợ JavaScript. Các ứng dụng server chạy trên Web server. Sau đây là các webserver phổ biến: Internet Information Server của Microsoft, Apache,...

- a. *Trên các trình duyệt:* JavaScript làm tăng cường tính động và khả năng tương tác của các trang web. Ta có thể sử dụng JavaScript để tương tác với người dùng. Cụ thể:

- ✓ Dùng Javascript để đáp lại các sự kiện của người dùng như: click chuột, rê chuột, load Form... nhằm làm thay đổi nội dung trang web và vị trí các phần tử trên trang web.
- ✓ Dùng Javascript để kiểm tra tính hợp lệ dữ liệu khi người dùng nhập vào form phía Client, trước khi trình duyệt gửi dữ liệu đến máy chủ để xử lý. Điều này giúp cho máy chủ không phải xử lý quá nhiều.

*b. Trên Webserver*

Ta có thể nhúng các câu lệnh JavaScript trong các tài liệu HTML để thực hiện trên Sever. Khi các scripts này được gọi từ server, Web server bắt đầu runtime engine để thực hiện các câu lệnh và trả về kết quả dưới dạng HTML để được hiển thị trên các trình duyệt.

Các câu lệnh JavaScript Server-side có thể được dùng để:

- ✓ Kết nối cơ sở dữ liệu
- ✓ Chia sẻ thông tin giữa các người dùng của một ứng dụng
- ✓ Truy cập hệ thống file trên server

#### 5.1.4 Các phiên bản(version) của JavaScript

Các trình duyệt khác nhau và các phiên bản của những trình duyệt này hỗ trợ khác nhau đối với các version của JavaScript. Dưới đây là các version của JavaScript đã được phát hành:

<b>Phiên bản(version)</b>	<b>Năm ra đời</b>	<b>Trình duyệt Navigator hỗ trợ</b>
JavaScript 1.0	3/1996	Navigator 2.0, IE 2.0
JavaScript 1.1	8/1996	Navigator 3.0
JavaScript 1.2	6/1997	Navigator 4.0 - 4.05
JavaScript 1.3	10/1998	Navigator 4.06 – 4.7
JavaScript 1.4	1999	Chạy phía máy chủ của Netscape
JavaScript 1.5	11/2000	Netscape Navigator 6.0, Firefox 1.0.
JavaScript 1.6	9/2005	Firefox 1.5
JavaScript 1.7	10/2006	Firefox 2.0
JavaScript 1.8	6/2008	Firefox 3.0
JavaScript 1.8.1	6/2009	Firefox 3.5
JavaScript 1.8.5	7/2010	Firefox 4.0

## Bảng 5.1. Các version của JavaScript

### 5.2. NHÚNG JAVASCRIPT VÀO TRANG HTML

JavaScript được nhúng vào một file HTML theo 2 cách phổ biến sau:

#### 5.2.1 Nhúng các câu lệnh Javascript trực tiếp vào trong tài liệu HTML

a. Mã javascript có thể được đặt trong phần HEAD

```
<Head>

  <script language= "JavaScript"
  type="text/JavaScript">

      //câu lệnh JavaScript...

  </script>

</head>
```

#### Chú ý:

✓ Có thể sử dụng `language="javascript"` hoặc `type="text/javascript"`, tuy nhiên format theo chuẩn W3C là `type="text/JavaScript"` (và bắt buộc đối với HTML4), cũng có thể bỏ đi nếu không cần.

✓ Có thể chỉ cần gõ tắt `<script> ...</script>`, trình duyệt sẽ tự động nhận dạng mã JavaScript.

#### Ví dụ 5.1:

Source Code:	Submit Code »	Result:
<pre>&lt;html&gt; &lt;head&gt; &lt;script&gt;   function myFunction()   {     document.getElementById("demo").innerHTML="Cảm ơn bạn đã click chuột!";   } &lt;/script&gt; &lt;/head&gt; &lt;body&gt;   &lt;h1&gt;My Web Page&lt;/h1&gt;   &lt;p id="demo"&gt;Chào mừng bạn đã ghé thăm trang web của tôi!&lt;/p&gt;   &lt;input type="button" value="Thử nghiệm" onclick="myFunction()"&gt; &lt;/body&gt; &lt;/html&gt;</pre>		<p><b>My Web Page</b></p> <p>Cảm ơn bạn đã click chuột!</p> <p><input type="button" value="Thử nghiệm"/></p>

b. Mã javascript có thể được đặt trong phần BODY

```
<html>

<head> ... </head>

<body>
```

```

<script type="text/javascript">
//câu lệnh Javascript...
</script>
</body>
</html>

```

### Chú ý:

- ✓ Trong cùng một trang Web có thể có nhiều đoạn mã JavaScript.
- ✓ Nên đặt câu lệnh JavaScript trong cặp thẻ <Head>...</Head>. Tuy nhiên, nếu đặt câu lệnh JavaScript trong phần Body thì nên đặt câu lệnh JavaScript ở cuối phần Body vì như vậy sẽ cải thiện được thời gian load trang web.

### Ví dụ:

Source Code:

Submit Code »

Result:

```

<html>
<head>
</head>
<body>
<h1>My Web Page</h1>
<p id="demo">Chào mừng bạn đã ghé thăm trang web của tôi!</p>
<input type="button" value="Thử nghiệm" onclick="myFunction()">
<script>
function myFunction()
{
document.getElementById("demo").innerHTML="Cảm ơn bạn đã click chuột!";
}
</script>
</body>
</html>

```

## My Web Page

Cảm ơn bạn đã click chuột!

### 5.2.2 Sử dụng một file JavaScript riêng

Cũng giống như CSS bên cạnh việc viết trực tiếp các câu lệnh JavaScript trong một tài liệu HTML, chúng ta có thể tạo ra một file riêng có phần mở rộng .js chứa mã JavaScript. File này được liên kết với một tài liệu HTML khi cần. Đây cũng là phương thức được sử dụng nhiều nhất hiện nay. Để nhúng tập tin .js vào tập tin HTML ta sử dụng cú pháp sau:

#### Cú pháp:

```

<head>

    <script language="javaScript" src = "<Đường dẫn>/tenfile.js"
type="text/javascript">
    </script>

</head>

```

### Chú ý:

- ✓ Một file .js có thể sử dụng được cho nhiều file HTML
- ✓ Đường dẫn có thể là đường dẫn tương đối hoặc tuyệt đối.
- ✓ Trong file .js không chứa các cặp thẻ `<script>...</script>`

**Ví dụ:**

**Bước 1:** Tạo 1 file HTML, đặt tên là test.html

```
<html>
<head>
  <script language="javaScript" src="test.js"></script>
</head>
<body>
  <h1>My Web Page</h1>
  <p id="demo">Chào mừng bạn đã ghé thăm website của tôi!</p>
  <input type="button" value="Thử nghiệm" onclick="myFunction()">
</body>
</html>
```

**Bước 2:** Tạo 1 file test.js tại cùng thư mục

```
function myFunction()
{
  document.getElementById("demo").innerHTML="Cảm ơn bạn đã click chuột!";
}
```

**Bước 3:** Mở file test.html bằng trình duyệt IE, Firefox, Chrome và xem kết quả.

### 5.2.3 Kích hoạt javascript từ các sự kiện

Các đoạn code javascript có thể được viết trực tiếp hoặc kích hoạt qua các sự kiện trên các đối tượng trong trang web, ví dụ như nhấp chuột (onclick), nhấp chuột đôi (ondblclick), di chuột đến (onmouseover), di chuột qua (onmouseout), khi gõ phím (onkeypress)...

**Cú pháp:**

```
<a href="javascript:alert(100)">Hiển thị số 100</a>
Hoặc
<input type="text" onkeypress="kiemtracophailaso()">
Hoặc
<body onload=alert("chào mừng tất cả các bạn đến với trang web!")>
```



### 5.3. CÁC CÂU LỆNH CỦA JAVASCRIPT

Mỗi câu lệnh **JavaScript** có phân biệt chữ hoa, chữ thường và thông thường được kết thúc bằng dấu ;

#### 5.3.1 Khởi lệnh

Bao gồm nhiều dòng lệnh được đặt trong dấu { }.

**Cú pháp:**

```
{  
    Lệnh 1;  
    Lệnh 2;  
    ...  
    Lệnh n;  
}
```

#### 5.3.2 Lệnh hiển thị dữ liệu ra cửa sổ trình duyệt

**Cú pháp:**

```
document.write(dữ liệu cần hiển thị);  
hoặc  
document.writeln(dữ liệu);
```

*Trong đó:*

- ✓ Dữ liệu cần hiển thị có thể là thẻ HTML, có thể là chuỗi kí tự, có thể là tên biến hoặc có thể là biểu thức.
- ✓ Nếu dữ liệu là thẻ HTML hoặc chuỗi kí tự thì phải đặt trong cặp dấu “ ”.

#### 5.3.3 Một số hiển thị thông báo (popup box)

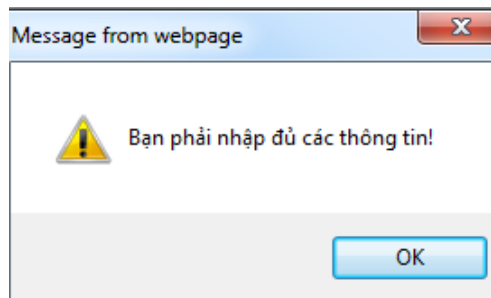
a. **alert** - Hiển thị ra cửa sổ trình duyệt một hộp hội thoại thông báo đơn

**Cú pháp :**

```
alert("Chuỗi thông báo");  
hoặc  
window.alert("Chuỗi thông báo");
```

Cách này được sử dụng để hiện ra một hộp hội thoại thông báo và sẽ chờ cho đến khi người sử dụng nhấn vào nút OK rồi mới tiếp tục thực hiện.

**Ví dụ:** alert("Bạn phải nhập đủ các thông tin");



Thông thường, người ta sử dụng `alert()` trong các trường hợp:

- ✓ Hiển thị cảnh báo cho người dùng
- ✓ Thông tin đưa vào form không hợp lệ
- ✓ Kết quả sau khi tính toán không hợp lệ
- ✓ Khi dịch vụ chưa sẵn sàng để truy nhập dữ liệu
- ✓ Dùng để soát lỗi

**Chú ý:**

✓ *alert có thể hoạt động với mọi kiểu dữ liệu, ví dụ: `alert(100)`, `alert(object)`, `alert(<array>)`, ....*

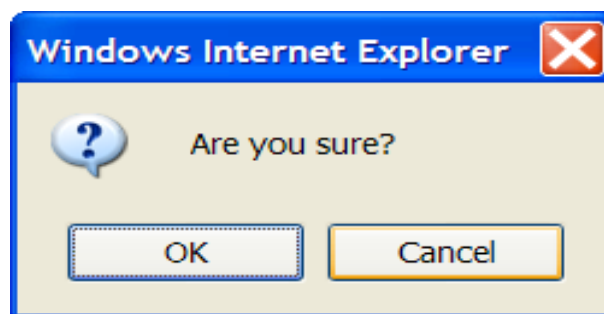
**b. confirm** - Hiển thị ra cửa sổ trình duyệt hộp hội thoại để khẳng định chấp nhận hoặc từ chối

**Cú pháp :**

```
confirm("Chuỗi thông báo?");
```

**Ví dụ:** `confirm("Are you sure?");`

`confirm("Bạn có muốn lưu dữ liệu?");`



**c. prompt** - Hiển thị ra cửa sổ trình duyệt hộp hội thoại cho phép người dùng nhập dữ liệu

**Cú pháp :**

```
prompt("Thông báo", giá trị mặc định)
```

**Ví dụ:** `prompt("Bạn hãy nhập vào tên sinh viên", "Nguyễn Văn A");`

### 5.3.4 Lời chú thích (comments)

Chú thích là những ghi chú của lập trình viên, để diễn giải về các hàm, đối tượng, biến, khối code, thao tác cần ghi nhớ.... Khi gặp lời giải thích trình duyệt sẽ bỏ qua và không dịch.

Chú thích trong javascript giống chú thích trong ngôn ngữ C, C++ và Java.

**Cú pháp:** //Lời chú thích đặt trên 1 dòng:

/\* đây là một chú thích có thể kéo dài trên nhiều dòng \*/

## 5.4. BIẾN TRONG JAVASCRIPT

### 5.4.1 Quy tắc đặt tên biến

- ✓ Tên biến phải bắt đầu bằng một chữ cái hoặc ký tự gạch dưới (" \_")
- ✓ Tên biến có thể chứa chữ hoặc số
- ✓ Không chứa khoảng trắng
- ✓ Có phân biệt chữ hoa và chữ thường

### 5.4.2 Khai báo biến

Ta có thể khai báo biến theo cú pháp sau:

```
Tenbien=giá trị;  
hoặc  
var Tenbien=giá trị;
```

**Lưu ý:** Chúng ta có thể khai báo nhiều biến trên cùng một dòng bằng cách tách tên các biến bởi dấu phẩy.

**Ví dụ:**

```
var hoten="Nga", age=30, nghe="Giaovien";
```

### 5.4.3 Phạm vi của biến

Phạm vi của biến được xác định tại vị trí mà nó được khai báo trong script.

- ✓ Nếu biến ở ngay phần đầu của script, thì nó được xem là một biến global (toàn cục) và có thể truy cập ở bất kỳ nơi đâu trong script, và trong cả trang web.
- ✓ Nếu ta khai báo biến trong một hàm, nó được xem là biến local (cục bộ) và nó chỉ được sử dụng chỉ đối với hàm đó mà thôi. Những hàm khác trong script không thể truy xuất vào biến đó được.

**Ví dụ:** Tạo trang web, cho người dùng nhập vào tên và tuổi. Hãy viết lại tên và tuổi của người đó ra màn hình bằng hàm **document.write**, trong đó tên có màu đậm, tuổi được gạch chân.

```
<HTML>
```

```

<HEAD>

<Title>Ví dụ Java script</Title>

<meta charset="UTF-8">

</HEAD>

<BODY>

<script language = "JavaScript">

    var Ten, Tuoi;          // Khai báo 2 biến để lưu tên và tuổi

    Ten = prompt("Bạn hãy nhập vào tên", "");

    Tuoi = prompt("Bạn hãy nhập vào Tuổi: ");

    document.write("Chào bạn : <B> " + Ten + "</B>");

    document.write("<BR>"); // Xuống dòng

    document.write("Tuổi của bạn là : <I> " + Tuoi + "</i>");

</script>

</BODY>

</HTML>

```

## 5.5. CÁC KIỂU DỮ LIỆU CỦA JAVASCRIPT

### 5.5.1. Giới thiệu

JavaScript là ngôn ngữ lập trình *không định kiểu (untyped)*, nên một biến có thể được sử dụng cho nhiều kiểu dữ liệu khác nhau ở các thời điểm khác nhau. Khi lập trình JavaScript, ta không cần phải chỉ ra kiểu dữ liệu khai báo biến. Khi cần thiết, kiểu dữ liệu có thể tự động chuyển thành một kiểu phù hợp. Như vậy, một biến có thể gán nhiều kiểu dữ liệu khác nhau tại các thời điểm khác nhau.

Ví dụ: Biến có tên là x tại thời điểm này ta có thể gán cho nó là một số, nhưng tại thời điểm khác có thể gán cho nó là một chuỗi kí tự.

```

var x=100;

x = "Hello";

```

### 5.5.2. Phân loại

JavaScript có thể chia thành 2 loại kiểu dữ liệu:

- Kiểu cơ bản
  - o Kiểu số(number)
  - o Kiểu logic(boolean)
  - o null và undefined
  - o Kiểu chuỗi(string)
- Kiểu đối tượng(object): Kiểu đối tượng là kiểu dữ liệu không phải kiểu cơ bản như trên. Đối tượng có thuộc tính(properties) và phương thức(methods). Mỗi thuộc tính có một tên và một giá trị (giá trị này có thể là số, chuỗi, logic hoặc là một đối tượng). Kiểu **mảng** và kiểu **Date** là một trường hợp đặc biệt của kiểu đối tượng.

Để truy cập vào thuộc tính hoặc phương thức (property/method) của đối tượng ta truy cập theo cú pháp sau:

**Tên đối tượng.Tên thuộc tính (hoặc tên phương thức)**

Dưới đây sẽ giới thiệu các kiểu dữ liệu cơ bản và một số đối tượng đặc biệt có sẵn trong JavaScript như (Array, Date, Math,...):

#### 5.5.2.1. Kiểu dữ liệu dạng số (Number)

JavaScript chỉ có một kiểu number chứ không phân nhỏ ra các kiểu: integer; real, double... như các ngôn ngữ lập trình khác. Tất cả các số trong JavaScript đều được biểu diễn dưới dạng dấu phẩy động. Để biểu diễn các số, nó sử dụng 64 – bit theo chuẩn IEEE 754. Như vậy JavaScript có thể lưu được các số trong khoảng  $-(2^{53} - 1)$  ,  $2^{53} - 1$

**Ví dụ:**

```
var x1=34.00;    // Viết dưới dạng số thập phân
var x2=34;        // Số nguyên viết dưới dạng không có số thập phân
```

#### 5.5.2.2 Dữ liệu kiểu logic (Boolean)

Miền giá trị của nó chỉ có 2 giá trị là **true** hoặc **false**

**Ví dụ:**

```
var x=true;
var y=false;
```

#### 5.5.2.3 null và undefined

*a. null:* Khi các biến dạng số, xâu, đối tượng chưa có giá trị thì nó nhận giá trị mặc định là **null**. Như vậy, là giá trị đặc biệt dùng để nhận diện một biến đã được nhập dữ liệu chưa.

*b. undefined:* Tương tự như **null**, trong JavaScript giá trị *undefined* cũng được dùng để chỉ rằng một biến không có giá trị, nhưng undefined ở mức sâu hơn, tức là biến đó chưa được khởi tạo giá trị ban đầu.

#### 5.5.2.4 Dữ liệu kiểu chuỗi ký tự (String)

**String** là một kiểu dữ liệu dùng để lưu trữ và khai thác trên một chuỗi các ký tự. Ký tự đầu tiên (phần tử đầu tiên) có chỉ số là 0, ký tự tiếp theo có chỉ số là 1,... Chiều dài của 1 chuỗi là số ký tự trong chuỗi. Chuỗi rỗng là chuỗi có chiều dài bằng 0. Chuỗi ký tự này được đặt trong cặp dấu “ ” hoặc ‘ ’. Trong JavaScript, chuỗi ký tự cũng là một đối tượng, do đó nó cũng có các phương thức và thuộc tính cơ bản.

Ví dụ:

```
"" // Chuỗi rỗng: không chứa ký tự nào
'testing'
"3.14"
"Wouldn't you prefer O'Reilly's book?"
"Chuỗi này có 2 dòng"
```

a. Khai báo

```
var tenbien = new String("chuỗi ký tự");
```

hoặc:

```
var tenbien = "chuỗi ký tự";
```

b. Truy cập đến các ký tự trong chuỗi

Truy cập đến từng ký tự trong chuỗi ta truy cập theo cú pháp: **Tenbien[chỉ số]**.

Lưu ý: Ký tự đầu tiên có chỉ số là [0], ký tự thứ 2 có chỉ số là [1]...

c. Các thuộc tính và phương thức cơ bản trên chuỗi

**Bảng 5.1. Các thuộc tính và phương thức cơ bản trên chuỗi**

Tên phương thức	Ý nghĩa	Thuộc tính/Phương thức
Length	Trả về chiều dài thực của chuỗi <b>Ví dụ:</b> var txt="Hello world"; document.write(txt.length); Kết quả trả về là 10	Thuộc tính
charAt(k)	Trả về ký tự thứ k của chuỗi. Ký tự đầu tiên có k=0, ký tự thứ 2 có k=1...	Phương thức
charCodeAt(k)	Trả về mã Unicode của ký tự thứ k trong chuỗi.	Phương thức
concat(chuỗi 1, chuỗi 2,...)	Nối 2 hoặc nhiều hơn 2 chuỗi và trả về 1 bản copy của các chuỗi đã nối mà không làm thay đổi các chuỗi cũ.	Phương thức

fromCharCode(giá trị Unicode)	Chuyển 1 giá trị Unicode về 1 ký tự tương ứng.	Phương thức
indexOf(str )	<p>Trả về vị trí đầu tiên tìm thấy xâu str xuất hiện trong một xâu khác. Nếu không tìm thấy thì hàm trả về giá trị 0.</p> <p><b>Ví dụ:</b></p> <pre>var str="Hello world, welcome to the universe."; var n=str.indexOf("welcome");</pre> <p>Kết quả trả về của n sẽ là 13</p>	Phương thức
replace(str1, str2)	Thay thế xâu 2 (str2) cho xâu 1(str1).	Phương thức
toLowerCase()	<p>Chuyển 1 xâu chứa các chữ hoa thành xâu chứa các chữ thường.</p> <p><b>Ví dụ:</b></p> <pre>var txt="Hello World!"; document.write(txt.toLowerCase());</pre> <p>Kết quả trả về là: hello world!</p>	Phương thức
toUpperCase()	<p>Chuyển 1 xâu chứa các chữ thường thành xâu chứa các chữ hoa.</p> <p><b>Ví dụ:</b></p> <pre>var txt="Hello World!"; document.write(txt.toUpperCase());</pre> <p>Kết quả trả về là: HELLO WORLD!</p>	Phương thức
trim()	Xóa các ký tự trống ở đầu xâu và cuối xâu.	Phương thức

#### 5.5.2.4 Dữ liệu kiểu Date/Time

Là dữ liệu kiểu đối tượng được dùng để lưu trữ và thao tác trên ngày tháng và thời gian.

a. Khai báo:

```
var tenbien=new Date()
```

b. Các thuộc tính và phương thức cơ bản

**Bảng 5.2. Các thuộc tính và phương thức cơ bản trên dữ liệu kiểu Date**

Tên phương thức	Ý nghĩa
-----------------	---------

getDate()	Trả về ngày hiện tại của máy tính (từ 1-31)
getDay()	Trả về thứ hiện tại của máy tính (từ 0-6): Chủ nhật có giá trị là 0, thứ 2 có giá trị là 1, thứ 3 có giá trị là 2,..., thứ 7 có giá trị là 6.
getFullYear()	Trả về năm hiện tại máy tính.
getHours()	Trả về giờ hiện tại (từ 0-23)
getMinutes()	Trả về phút hiện tại của máy tính (từ 0-59)
getMonth()	Trả về tháng hiện tại của máy tính (từ 0-11): Tháng 1 có giá trị 0, tháng 2 có giá trị 1,..., tháng 12 có giá trị 11.
getSeconds()	Trả về giây hiện tại của máy tính (từ 0-59)
toString()	Chuyển đổi tượng ngày thành một chuỗi.

**Ví dụ:** Cho người dùng nhập vào năm sinh của họ, sau đó hiển thị tuổi tương ứng.

Minh họa code:

```
<HTML>

<head>

    <meta charset="utf-8"/>

    <title>Tính tuổi</title>

</Head>

<BODY>

<script language="JavaScript">

    var D = new Date();

    var NamSinh, NamHienTai;

    NamHienTai = D.getFullYear(); // Lưu năm hiện tại vào biến NamHienTai

    NamSinh = prompt("Bạn sinh năm bao nhiêu?", "");

    alert("Tuổi của bạn bây giờ là : " + (NamHienTai-NamSinh));

</script>

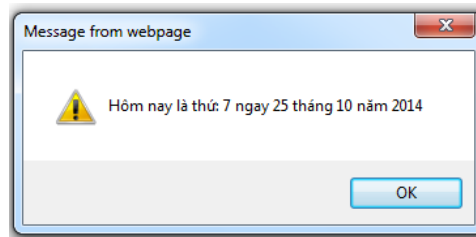
</BODY>

</HTML>
```





**Ví dụ:** Hãy hiển thị ngày và giờ của hệ thống máy tính khi trang web được nạp. Thông tin hiển thị ra có dạng như sau:



Minh họa code:

```
<HTML>

<head>

    <meta charset="utf-8"/>

    <title>Hiển thị ngày tháng năm của hệ thống</title>

</Head>

<BODY>

<script language="JavaScript">

    var D = new Date();

    var thu,ngay, thang,nam;

    thu=D.getDay()+1;

    ngay=D.getDate();

    thang=D.getMonth()+1;

    nam=D.getFullYear();

    alert("Hôm nay là thứ:"+" "+thu+" ngày"+" "+ngay+" tháng "+thang+" năm "+nam);

</script>

</BODY>

</HTML>
```

### 5.5.2. 5. Dữ liệu kiểu mảng (Array)

Trong JavaScript, kiểu mảng dùng để lưu trữ các tập dữ liệu phức hợp. Các phần tử trong mảng có cùng tên nhưng khác chỉ số, mỗi phần tử trong 1 mảng có thể không cùng kiểu dữ liệu. Các phần tử trong mảng có thể có kiểu number, string, logic, object hoặc array.

Chỉ số của mảng dùng để phân biệt các phần tử mảng với nhau. Phần tử đầu tiên của mảng có chỉ số là 0, phần tử thứ 2 có chỉ số là 1,...Chỉ số cao nhất của mảng là 4294967294 (tức  $2^{32}-2$ ). Như vậy, một mảng có thể chứa tối đa 4294967295 phần tử.

Mảng là một đối tượng (object) đặc biệt của JavaScript. Vì vậy, nó có phương thức và thuộc tính. Để truy cập đến thuộc tính hoặc phương thức của mảng ta thực hiện theo cú pháp:

**TendoituongMang.tenphuongthuc**

a. Khai báo mảng

Cách 1:

**Cú pháp:**

```
var Tenmang=[danh sách các phần tử]
```

Lưu ý: Nếu có nhiều phần tử thì các phần tử cách nhau bởi dấu phẩy (,)

**Ví dụ:**

```
var empty = []; // Một mảng không có phần tử
var a = [2, 3, 5, 7, 11]; // Một mảng có 5 phần tử
var misc = [ 1.1, true, "a" ]; // Một mảng có 3 phần tử có kiểu khác nhau
Array literals can contain object literals or other array literals:
var b = [[1,{x:1, y:2}], [2, {x:3, y:4}]]; // Mỗi phần tử trong mảng lại là một
mảng hoặc một đối tượng.
var count = [1,,3]; // Một mảng có 3 phần tử, phần tử ở giữa chưa được
khởi tạo giá trị (undefined).
var undefs = [,]; // Một mảng có 2 phần tử, cả 2 phần tử chưa được khởi
tạo giá trị
```

Cách 2:

**Cú pháp:**

```
Tenmang= new Array();

Tenbien=new Array([arrayLength]);
    hoặc
Tenbien=new Array([element0[, element1[, ..., elementN]]]);
```

Trong đó:

- ✓ *arrayLength* là số phần tử tối đa mà mảng này có thể chứa.
- ✓ *element<i>* là các giá trị phần tử của mảng.

**Ví dụ 1:** khai báo một mảng **tingh** có 3 phần tử với  $a[0] = \text{"Hà Nội"}; a[1] = \text{"Thái Bình"}; a[2] = \text{"Thanh Hóa"}.$

```
var tingh=new Array("Hà Nội","Thái Bình","Thanh Hóa");
```

**Ví dụ 2:** Khai báo một mảng tên là **cars** có tối đa 10 phần tử.

```
var cars = new Array(10);
```

b. Truy cập vào từng phần tử của mảng

**Cú pháp:** `bienmang[chỉ số];`

Trong đó, phần tử đầu tiên có chỉ số là 0, phần tử tiếp theo có chỉ số là 1,...

**Ví dụ:**

```
var a = ["world"]; // Start with a one-element array
var b = a[0]; // Đọc giá trị trong phần tử đầu tiên(phần tử 0) của mảng, gán cho biến có tên là b
a[1] = 3.14; // Gán giá trị 3.14 cho phần tử 1 của mảng có tên là a
i = 2;
a[i] = 3; // Gán giá trị 3 cho phần tử 2 của mảng a
a[i + 1] = "hello"; // Gán giá trị "hello" cho phần tử 3 của mảng a
a[a[i]] = a[0]; // Đọc giá trị trong phần tử 0 và 2, gán giá trị của phần tử 3
```

**Ví dụ:** Cho người dùng nhập vào danh sách tên của sinh viên, sau đó sắp xếp theo vần Alphabet rồi hiển thị danh sách đã sắp xếp đó ra màn hình, mỗi người trên một dòng.

Hướng dẫn: Sử dụng vòng lặp for để cho phép nhập danh sách tên và Lưu danh sách vào một mảng, sau đó sử dụng phương thức sort của đối tượng mảng để sắp xếp, tiếp theo dùng vòng lặp **for...in** để in các phần tử trong danh sách.

Minh họa code:

```
<HTML>

<TITLE>Sắp xếp mảng</TITLE>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<BODY>

<script language="JavaScript">

    var n, x;

    var DS = new Array(100); // Khai báo mảng DS, có thể lưu tối đa là 100 phần tử

    SoLuong = prompt("Bạn cần nhập bao nhiêu người : ", 5);

    for (i=0; i < n; i++)

    {

        DS[i] = prompt("Nhập vào Tên từng người: ", "");

    }

    // Gọi hàm sort của đối tượng mảng DS để sắp xếp

    DS.sort();
```

```

//Hiển thị kết quả sau khi sắp (sort)

document.write("<h1>Danh sách đã sắp xếp là </h1>");

for (x in DS)          /* Nên sử dụng cấu trúc for ... in này để duyệt mảng */
{

    document.write( DS[x] );

    document.write("<BR>");          // Xuống dòng

}

</script>

</BODY>

</HTML>

```

*\*Lưu ý:* Nếu muốn sắp theo chiều giảm dần thì sau khi sort ta gọi hàm reverse.

c. Một số phương thức của mảng

Tên phương thức	Ý nghĩa	Cú pháp	Ví dụ
<b>concat()</b>	Ghép 2 mảng thành 1 mảng mới	<b>tenmang1.concat(tenmang2)</b>	var a = [1,2,3]; a.concat(4, 5) // Trả về [1,2,3,4,5] a.concat([4,5]); // Trả về [1,2,3,4,5] a.concat([4,5],[6,7]) // Trả về [1,2,3,4,5,6,7] a.concat(4, [5,[6,7]]) // Trả về [1,2,3,4,5,[6,7]]
<b>indexOf(giá trị tìm)</b>	Tìm vị trí xuất hiện đầu tiên của 1 phần tử trong mảng	<b>Tenbienmang.indexOf(phantucan tim)</b>	
<b>join()</b>	Chuyển các phần tử của 1 mảng thành 1 chuỗi ký tự	<b>Tenbienmang.join(dauphancach)</b>	var a = [1, 2, 3]; // Tạo 1 mảng với 3 phần tử a.join(); // => "1,2,3" a.join(" "); // => "1 2 3" a.join(""); // => "123" var b = new Array(10); // Tạo một mảng có độ dài là 10 b.join('-') // => '-----': Một chuỗi có 9 dấu '-'
<b>lastIndexOf(Giá trị tìm)</b>	Tìm vị trí xuất hiện của một phần tử	<b>Tenbienmang.LastIndexOf(phantucan tim)</b>	a = [0,1,2,1,0]; a.indexOf(1) // => 1: a[1] is 1 a.lastIndexOf(1) // => 3: a[3] is 1 a.indexOf(3) // => -1: no element has value 3

	trong mảng. Thứ tự tìm kiếm là tìm từ phần tử cuối về phần tử đầu tiên		
<b>toString()</b>	Chuyển mảng thành 1 chuỗi ký tự	<b>Mảng.toString()</b>	[1,2,3].toString() // Yields '1,2,3' ["a", "b", "c"].toString() // Trả về 'a,b,c' [1, [2,'c']].toString() // Trả về '1,2,c'
<b>push()</b>	Thêm một hoặc nhiều giá trị phần tử vào cuối mảng. Trả về chiều dài của mảng mới	<b>Tenbienmang.push()</b>	var stack = []; // tạo một mảng rỗng: [] stack.push(1,2); // mảng stack là [1,2]. Trả về 2
<b>pop()</b>	Xóa phần tử cuối cùng của mảng. Trả về giá trị phần tử đã bị xóa	<b>Tenbienmang.pop()</b>	var stack = []; // tạo một mảng rỗng: [] stack.pop(); // Xóa phần tử 2 khỏi mảng, mảng stack mảng stack bây giờ là: [1]. Giá trị trả về là 2
<b>unshift()</b>	Thêm một giá trị phần tử mới vào vị trí đầu của mảng	<b>Tenbienmang.unshift("giá trị")</b>	var a = []; // a:[] a.unshift(1); // a:[1] Trả về: 1 a.unshift(22); // a:[22,1] Trả về: 2
<b>shift()</b>	Xóa phần tử đầu tiên của mảng ra khỏi mảng. Trả về giá trị đã xóa	<b>Tenbienmang.shift()</b>	var a = []; // a:[] a.unshift(1); // a:[1] Trả về: 1 a.unshift(22); // a:[22,1] Trả về: 2 a.shift(); // a:[1] Trả về: 22 a.unshift(3,[4,5]); // a:[3,[4,5],1] Trả về: 3 a.shift(); // a:[[4,5],1] Trả về: 3 a.shift(); // a:[1] Trả về: [4,5] a.shift(); // a:[] Trả về: 1
<b>sort()</b>	Sắp xếp các phần tử của mảng theo thứ tự tăng dần	<b>Tenbienmang.sort()</b>	var a = new Array("banana", "cherry", "apple"); a.sort(); var s = a.join(", "); // s == "apple, banana, cherry"

	Chú ý: Nếu mảng chứa phần tử undefined chúng sẽ được sắp vào cuối bảng		
<b>reverse()</b>	Đảo ngược thứ tự các giá trị trong mảng	<b>Tenbienmang.reverse()</b>	var a = [1,2,3]; a.reverse() // => Chuyển [1,2,3] thành [3,2,1]

#### 5.5.2.5.Đối tượng Math

Đối tượng Math là đối tượng đã được JavaScript định nghĩa sẵn. Do đó, nó có thuộc tính và phương thức.

#### Các thuộc tính của đối tượng Math

Tên thuộc tính	Mô tả
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

#### Các phương thức cơ bản của đối tượng Math

Phương thức Method	Mô tả Description
abs(x)	Trả về giá trị tuyệt đối của x
sin(x)	Trả về giá trị sin của x(tính bằng radian)
cos(x)	Trả về giá trị cosin của x(tính bằng radian)
tan(x)	Trả về giá trị tang của x(tính bằng radian)
min(x,y)	Trả về giá trị nhỏ nhất của hai số x và y
max(x, y)	Trả về giá trị lớn nhất của hai số x và y
round (x)	Làm tròn đối số x tới số nguyên gần nhất
sqrt(x)	Trả về căn bậc hai của số x

Để truy cập vào thuộc tính hoặc phương thức của đối tượng Math ta truy cập theo cú pháp:

`Math.tenthuoc tinh`

hoặc

`Math.tenphuongthuc`

**Ví dụ:**

a. Hàm sau sẽ trả về giá trị của PI:

```
function getPi()  
{  
    return Math.PI;  
}
```

b. Hàm sau sẽ trả về giá trị lớn nhất của 2 biến x và y:

```
function getMin(x,y)  
{  
    return Math.min(x,y);  
}
```

## 5.6. CÁC TOÁN TỬ TRONG JAVASCRIPT

### 5.6.1 Phép gán

Tên biến=giá trị;

### 5.6.2 Phép so sánh

Người ta sử dụng toán tử so sánh để so sánh hai toán hạng và trả lại giá trị đúng hay sai phụ thuộc vào kết quả so sánh. Sau đây là một số toán tử so sánh trong JavaScript:

Phép toán	Mô tả
<code>==</code>	Bằng
<code>!=</code>	Không bằng
<code>&gt;</code>	Lớn hơn
<code>&gt;=</code>	Lớn hơn hoặc bằng
<code>&lt;</code>	Nhỏ hơn
<code>&lt;=</code>	Nhỏ hơn hoặc bằng

### 5.6.3 Các phép toán số học cơ bản

Giả sử cho  $y=9$ , ta có thể giải thích các phép toán số học như bảng sau:

Phép toán	Mô tả	Ví dụ	Kết quả
<code>+</code>	Phép cộng	$x=x+2$	$x=11$ $y=9$
<code>-</code>	Phép trừ	$x=y-3$	$x=6$ $y=9$
<code>*</code>	Phép nhân	$x=y*2$	$x=18$ $y=9$



/	Phép chia	$x=y/2$	$x=4.5$ $y=9$
%	Phép chia lấy số dư	$x=y\%2$	$y=9$ $x=1$
++	Tăng lên một đơn vị (++y: sẽ trả về giá trị của x sau khi tăng 1; y++ sẽ trả về giá trị của x trước khi tăng 1)	$x=++y$ $x=y++$	$x=10 \quad y=10$ $x=9; y=10$
--	Giảm đi một đơn vị (--y: sẽ trả về giá trị của x sau khi giảm 1; y-- sẽ trả về giá trị của x trước khi giảm 1)	$x=--y$ $x=y--$	$x=8 \quad y=8$ $x=9; y=8$

#### 5.6.4 Các phép toán logic cơ bản

Toán tử	Mô tả
&&	Phép và (And)
	Phép hoặc(Or)
!	Phép phủ định (Not)

#### 5.6.5 Các phép toán trên chuỗi kí tự

Đối với chuỗi kí tự JavaScript chỉ hỗ trợ 1 phép toán +

##### Ví dụ 1:

```
txt1="Học viện";
txt2="Nông nghiệp Việt Nam";
txt3=txt1+" "+txt2;
```

Kết quả: `txt3= "Học viện Nông nghiệp Việt Nam"`

##### Ví dụ 2:

```
x=5+5;           →10
x="5"+"5";       →55
x=5+"5";         →55
x="5"+5;         →55
```

**Lưu ý:** Phép cộng một chuỗi kí tự và một số cho một chuỗi kí tự

## 5.7. CÁC CÂU LỆNH ĐIỀU KHIỂN DỮ LIỆU TRONG JAVASCRIPT

### 5.7.1. Các lệnh điều kiện

#### a. Câu lệnh rẽ nhánh *if*

**Cú pháp:**

```
if (điều kiện)
{
    Các lệnh;
}
```

#### b. Câu lệnh rẽ nhánh *if...else...*

**Cú pháp:**

```
if (điều kiện)
{
    lệnh 1;
}
else
{
    lệnh 2;
}
```

**Ví dụ:**

```
<script type="text/javascript">
/*Nếu thời gian nhỏ hơn 10, sẽ nhận được lời chúc "Good morning", ngược lại sẽ nhận
được lời chúc "Good day" */
```

```
var d = new Date();
var time = d.getHours();
if (time < 10)
{
    alert("Good morning!");
}
else
{
    alert("Good day!");
}
```

</script>

c. *Câu lệnh switch*

Khi ta có nhiều tùy chọn if...else... thì tốt hơn nên lệnh switch. Lệnh này tương đương với lệnh case...of... trong pascal. Lệnh switch thực thi một trong các khối lệnh tùy thuộc vào giá trị của biểu thức. Nếu không tìm thấy một giá trị nào trong danh sách các case của nó, khối lệnh trong phần default sẽ được thực hiện. Lệnh break dùng để thoát ra khỏi câu lệnh switch.

**Cú pháp:**

```
switch(bieu_thuc)
{
    case 1: nhãn1:
        lệnh 1;
        break;
    case 2: nhãn 2:
        lệnh 2;
        break;
    ...
    case n: nhãn n;
        lệnh n;
        break;
    default: lệnh khác;
}
```

**Ví dụ:**

```
<script type="text/javascript">
var d=new Date();
theDay=d.getDay();
switch (theDay)
{
    case 1: document.write("Thứ 2");
        break;
    case 2: document.write("Thứ 3");
        break;
    case 3: document.write("Thứ 4");
```

```

        break;
    case 4: document.write("Thứ 5");
        break;
    case 5: document.write("Thứ 6");
        break;
    case 6: document.write("Thứ 7");
        break;
    case 0: document.write("Chủ nhật");
        break;
    default: document.write("Không phải thứ");
}
</script>

```

### 5.7.2. Các lệnh lặp

#### a. Câu lệnh lặp *for*

##### Cú pháp:

```

for (biến = giá_trị_đầu; điều_kiện; biến=biến + bước_nhảy)
{
    các_lệnh;
}

```

**Ví dụ:** Sử dụng câu lệnh for... để tính tổng  $S=1+2+\dots+10$

Source Code:	Submit Code »	Result:
<pre> &lt;html&gt; &lt;body&gt;   &lt;script type="text/javascript"&gt;     var S=0;     for (i=1;i&lt;=10;i++)       S=S+i;     document.write("Tổng là: ",S);   &lt;/script&gt; &lt;/body&gt; &lt;/html&gt; </pre>		<p>Tổng là: 55</p>

#### b. Câu lệnh vòng lặp *while*

##### Cú pháp:

```

while(dieu_kien)
{
    các_lệnh;
}

```

```
}
```

**Ý nghĩa:** Câu lệnh này, ban đầu máy kiểm tra điều kiện. Nếu điều kiện đúng thì thực hiện khối lệnh trong vòng lặp, nếu sai thì thoát.

**Ví dụ:** Ta cũng có thể sử dụng câu lệnh while để tính tổng  $S=1+2+\dots+10$  như sau:

Source Code:	Submit Code »	Result:
<pre>&lt;html&gt; &lt;body&gt; &lt;script&gt; { var x=0,i=1; while (i&lt;=10) { x=x + i; i++; } document.write("Giá trị của x là:", x); } &lt;/script&gt; &lt;/body&gt; &lt;/html&gt;</pre>		Giá trị của x là:55

c. Câu lệnh vòng lặp do...while

Vòng lặp do...while được dùng để thực thi một khối lệnh cho đến khi điều kiện là false.

**Cú pháp:**

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

**Ý nghĩa:** Câu lệnh này, ban đầu máy thực hiện các lệnh, sau đó kiểm tra điều kiện. Nếu điều kiện đúng thì lại quay về thực hiện các lệnh, nếu sai thì thoát.

**Ví dụ:** Sử dụng câu lệnh while để tính tổng  $S=1+2+\dots+10$ . Code như sau:

Source Code:	Submit Code »	Result:
<pre>&lt;html&gt; &lt;body&gt; &lt;script&gt; { var x=0,i=1; do { x=x + i; i++; } while (i&lt;=10) document.write("Giá trị của x là:", x); } &lt;/script&gt; &lt;/body&gt; &lt;/html&gt;</pre>		Giá trị của x là:55

### 5.7.3. Câu lệnh break

Tác dụng: Phá vỡ vòng lặp hiện tại và thực hiện các lệnh tiếp theo sau vòng lặp đó.

**Ví dụ:**

Source Code:	Submit Code »	Result:
<pre> &lt;html&gt; &lt;body&gt; &lt;script type="text/javascript"&gt; var i=0; for (i=0;i&lt;=10;i++) {     if (i==3)         break;       document.write("Đây là số: " + i);     document.write("&lt;br&gt;"); } &lt;/script&gt; &lt;/body&gt; &lt;/html&gt; </pre>		<p>Đây là số: 0          Đây là số: 1          Đây là số: 2</p>

#### 5.7.4. Câu lệnh for...in

Câu lệnh **for... in** được dùng để duyệt các thuộc tính của một đối tượng hay các phần tử của một mảng.

**Cú pháp:**

```

for (biến in đối_tượng)
{
    các lệnh;
}

```

**Ví dụ:**

Source Code:	Submit Code »	Result:
<pre> &lt;html&gt; &lt;body&gt; &lt;script type="text/javascript"&gt; var x; var mycars = new Array(); mycars[0] = "Saab"; mycars[1] = "Volvo"; mycars[2] = "BMW"; for (x in mycars) {     document.write(mycars[x] + "&lt;br /&gt;"); } &lt;/script&gt; &lt;/body&gt; &lt;/html&gt; </pre>		<p>Saab          Volvo          BMW</p>

#### 5.7.5. Câu lệnh with

Câu lệnh “with” được dùng để thực thi một tập các lệnh cùng tham chiếu đến một đối tượng xác định.

**Cú pháp:**

```

with (object)
{ các lệnh; }

```

**Ví dụ:**

```
with (Math) {  
    a = PI * r*r;  
    y = r*sin(theta);  
    x = r*cos(theta);  
}
```

## 5.8. HÀM TRONG JAVASCRIPT (FUNCTIONS)

### 5.8.1. Khái niệm

Hàm (*Function*) là một khối các lệnh, khối các lệnh này sẽ được thực hiện khi có lời gọi hàm thông qua **Tên hàm** kèm theo các tham số (nếu có).

Cũng như các ngôn ngữ lập trình khác hàm trong Javascript có 2 loại:

- ✓ Các hàm sẵn có của thư viện Javascript
- ✓ Các hàm do người dùng định nghĩa

*Dưới đây sẽ trình bày cách tạo một function trong JavaScript:*

### 5.8.2. Cú pháp tạo 1 hàm:

```
function ten_ham(tham_so_1, tham_so_2, ... tham_so_n)  
{  
    các lệnh;  
    return giá_trị_trả_về;    //nếu có  
}
```

Nếu hàm không có tham số, ta có thể khai báo như sau:

```
function ten_ham()  
{  
    các lệnh;  
    return giá_trị_trả_về; //nếu có  
}
```

**Lưu ý:**

- ✓ Nếu hàm có tham số thì tham số phải có kiểu dữ liệu là: Number, String hoặc Object và số tham số tối đa có thể có là 25
- ✓ Để hàm trả về 1 giá trị thì trong hàm phải có lệnh **return** để trả về giá trị cần lấy. Ví dụ hàm dưới đây trả về bình phương đối số của nó:

```
Function binhphuong(x)  
{
```

```

    return x*x;
}

```

**Ví dụ:** Viết chương trình tính tích của 2 số bằng cách sử dụng chương trình con.

Source Code:	Submit Code »	Result:
<pre> &lt;html&gt; &lt;head&gt; &lt;script type="text/javascript"&gt; function tich(a,b) {     x=a*b;     return x; } &lt;/script&gt; &lt;/head&gt; &lt;body&gt; &lt;script type="text/javascript"&gt; var x=5; y=10; document.write("Tích của x và y là: ",tich(x,y)); &lt;/script&gt; &lt;/body&gt; &lt;/html&gt; </pre>		<p>Tích của x và y là: 50</p>

### 5.8.3. Một số hàm có sẵn

JavaScript có một số hàm có sẵn, gắn trực tiếp vào chính ngôn ngữ và không nằm trong một đối tượng nào:

- ✓ eval
- ✓ parseInt
- ✓ parseFloat

#### a. eval

Hàm này được sử dụng để đánh giá các biểu thức hay lệnh. Biểu thức, lệnh hay các đối tượng của thuộc tính đều có thể được đánh giá. Đặc biệt hết sức hữu ích khi đánh giá các biểu thức do người dùng đưa vào (ngược lại có thể đánh giá trực tiếp).

**Cú pháp:**

**returnval=eval** (bất kỳ biểu thức hay lệnh hợp lệ trong Java)

**Ví dụ:**

```

<HTML>
<HEAD>
<TITLE>Eval Example </TITLE>
<SCRIPT LANGUAGE="JavaScript">
    var string="10+100";
    document.write(eval(string));
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```



### **b. parseInt**

Hàm này chuyển một chuỗi số thành số nguyên với cơ số là tham số thứ hai (tham số này không bắt buộc). Hàm này thông được sử dụng để chuyển các số nguyên sang cơ số 10 và đảm bảo rằng các dữ liệu đọc nhập dưới dạng ký tự được chuyển thành số trước khi tính toán. Trong trường hợp dữ liệu vào không hợp lệ, hàm parseInt sẽ đọc và chuyển dạng chuỗi đến vị trí nó tìm thấy ký tự không phải là số. Ngoài ra hàm này còn cắt dấu phẩy động.

#### ***Cú pháp***

**parseInt (string, [, radix])**

#### ***Ví dụ:***

```
<HTML>
<HEAD>
<TITLE> parseInt Exemple </TITLE>
<SCRIPT LANGUAGE= "JavaScript">
document.write("Converting chuỗi 20 thành số: " + parseInt("20") + "<BR>");
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>
```

### **c. parseFloat**

Hàm này giống hàm parseInt nhưng nó chuyển chuỗi thành số biểu diễn dưới dạng dấu phẩy động.

#### ***Cú pháp***

**parseFloat (string)**

#### ***Ví dụ:***

```
parseFloat("-187.56")
```

## **5.9. SỰ KIỆN TRONG JAVASCRIPT**

Sự kiện là những hành động của người sử dụng tác động lên website của bạn và được phát hiện bởi JavaScript.

### **5.9.1. Các sự kiện cơ bản của JavaScript**

#### ***a. Sự kiện chung***

- ✓ onLoad: Sự kiện nạp một trang web
- ✓ onUnload: Sự kiện đóng một trang web

#### ***b. Sự kiện liên quan đến chuột***

- ✓ Onmouseover: Khi con trỏ chuột chỉ vào một đối tượng nào đó

- ✓ Onmousemove: Khi con trỏ chuột di chuyển trên một đối tượng
- ✓ Onmouseout: Khi con trỏ chuột rời khỏi một đối tượng
- ✓ Onclick: Sự kiện click chuột vào một đối tượng
- ✓ Ondblclick: Sự kiện click kép chuột vào một đối tượng

c. *Sự kiện liên quan đến form*

- ✓ onsubmit: Khi dữ liệu của một form được gửi đi
- ✓ onreset: Khi nhấn vào nút RESET của một form
- ✓ onfocus: Khi con trỏ hoặc chuột chỉ vào một đối tượng.
- ✓ onblur: Khi con trỏ hoặc chuột rời khỏi đối tượng.
- ✓ onselect: Khi người dùng chọn một đoạn văn bản trong một phần tử INPUT hoặc TEXTAREA.
- ✓ onchange: Sự kiện onchange xảy ra bất cứ khi nào một phần tử form thay đổi.

### 5.9.2. *Liên kết một sự kiện với 1 hàm javascript*

**Cú pháp:**

```
<element event="ten_ham(tham_so_ nếu có)">
```

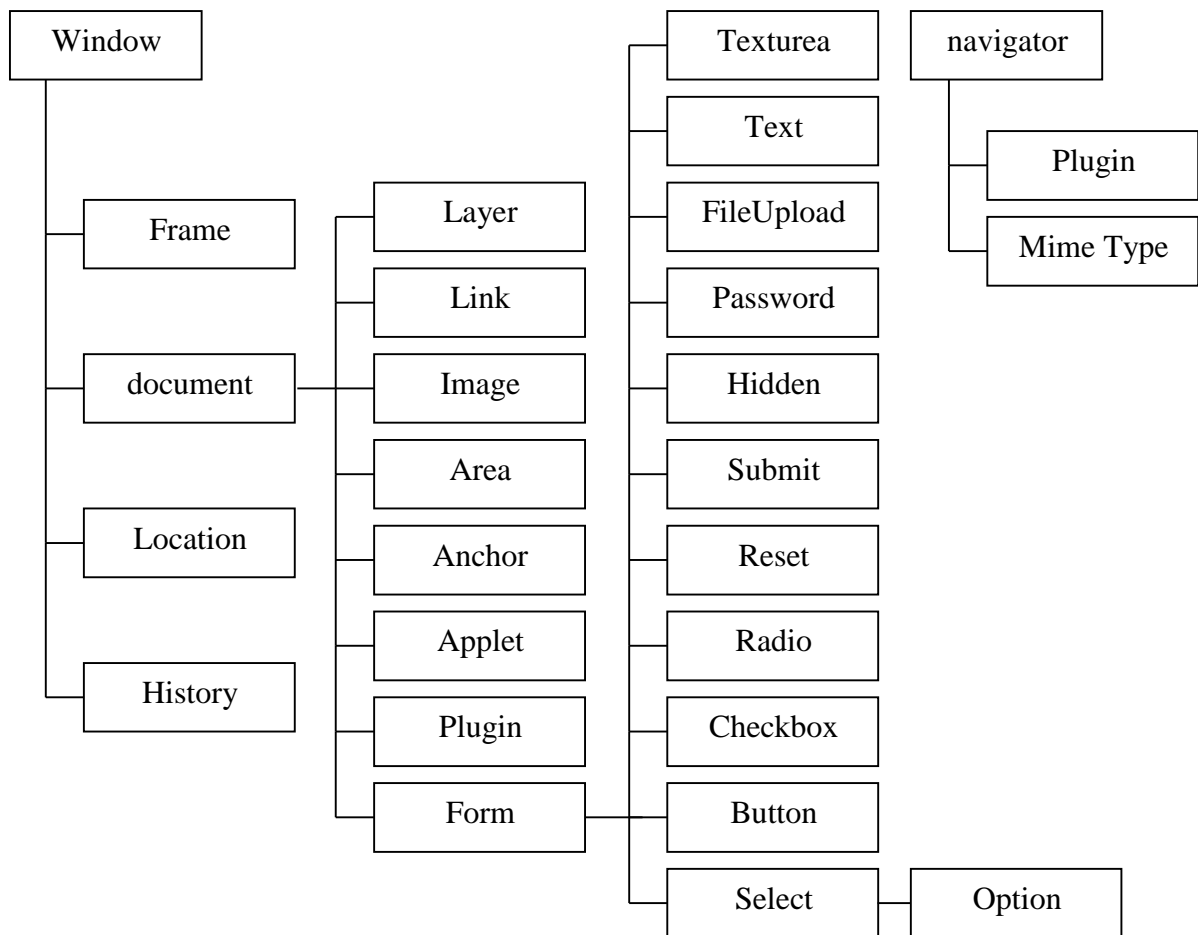
**Ví dụ:**

```
<body onload= "loadForm()">
<input type="submit" onclick="verify_data()">
<input type="checkbox" onchange="ten_ham(tham_so)">
```

## 5.10. ĐỐI TƯỢNG TRONG JAVASCRIPT

### 5.10.1. *Sơ đồ phân cấp*

JavaScript là ngôn ngữ lập trình dựa trên đối tượng, các đối tượng có thuộc tính (properties), Phương thức (methods), và các sự kiện (event handlers) gắn với chúng. Những đối tượng này tồn tại một cách có cấp bậc và phản ánh chính cấu trúc của file HTML đó. Sau đây là sơ đồ phân cấp các đối tượng:



Trong đó:

- ✓ **navigator:** có các thuộc tính tên và phiên bản của trình duyệt đang được sử dụng,
- ✓ **window:** là đối tượng ở mức cao nhất, là toàn bộ cửa sổ trình duyệt.
- ✓ **document:** chứa các thuộc tính dựa trên nội dung trang web và các forms.
- ✓ **location:** là địa chỉ hiện tại (URL)
- ✓ **history:** Chứa các thuộc tính về các URL mà người dùng đã mở trước đó.

### 5.10.2. Đối tượng Window

#### a. Các thuộc tính (Properties)

- ✓ 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 mẹ.
- ✓ Name - Tên của cửa sổ hiện thời.
- ✓ Parent - Đối tượng cửa sổ cha mẹ

- ✓ Self - Cửa sổ hiện thời.
- ✓ Status - Được sử dụng cho thông báo tạm thời hiển thị lên trên thanh trạng thái của sổ. Được sử dụng để lấy hay đặt lại thông báo trạng thái và ghi đè lên defaultStatus.
- ✓ Top - Cửa sổ ở trên cùng.
- ✓ Window - Cửa sổ hiện thời.

#### **b. Các phương thức (Methods)**

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

#### **Ví dụ:**

<HTML>

<BODY>

<FORM>

<INPUT TYPE="button" VALUE="Open Window"  
onClick="msgWindow=window.open('http://google.com.vn','cuaso1','resizable=no,width=300,height=300')">

<INPUT TYPE="button" VALUE="Close Window" onClick="msgWindow.close()">

</FORM>

</BODY>

</HTML>

#### **c. Các sự kiện (events)**

- ✓ onLoad - Khi cửa sổ được load.
- ✓ onUnload - Khi cửa sổ hủy.
- ✓ onClose – Khi đóng cửa sổ.

#### **5.10.3. Location**

**Ví dụ:** <http://www.vnua.com.vn/cntt/search.aspx?query=abc>

#### **Các thuộc tính**

- ✓ hash - Tên anchor của vị trí hiện thời (ví dụ cntt).
- ✓ Host - Phần hostname:port của URL (ví dụ www. hau.com.vn). Chú ý rằng đây thường là cổng ngầm định và ít khi được chỉ ra.
- ✓ Hostname - Tên của host và domain (ví dụ www. hau.com.vn).
- ✓ href - Toàn bộ URL cho document hiện tại.
- ✓ Pathname - Phần đường dẫn của URL (ví dụ /cntt/search.aspx?query=abc).
- ✓ Port - Cổng truyền thông được sử dụng cho máy tính host, thường là cổng ngầm định (80).
- ✓ Protocol - Giao thức được sử dụng (cùng với dấu hai chấm) (ví dụ http:).
- ✓ Search - Câu truy vấn tìm kiếm có thể ở cuối URL cho các script CGI (query=abc)

#### 5.10.4. *Frame*

Một cửa sổ có thể có một vài frame. Các frame có thể cuộn một cách độc lập với nhau và mỗi frame có URL riêng. frame không có các chương trình xử lý sự kiện. Sự kiện onLoad và onUnload là của đối tượng window.

##### *a. 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.

##### *b. Sử dụng frame*

Để tạo một frame, ta sử dụng thẻ **FRAMESET**. Mục đích của thẻ này là định nghĩa một tập các frame trong một trang.

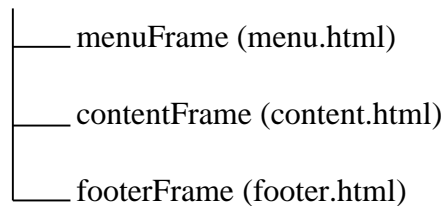
**Ví dụ:** tạo trang Web với frame như sau

```
<HTML>
<HEAD>
<TITLE>Ví dụ Frame</TITLE>
</HEAD>
<FRAMESET ROWS="90%,10%">
<FRAMESET COLS="30%,70%">
    <FRAME SRC=menu.html NAME="menuFrame">
    <FRAME SRC=content.html NAME="contentFrame">
</FRAMESET >
<FRAME SRC=footer.html NAME="footerFrame">
</FRAMESET >
```

</HTML>

Sơ đồ sau hiển thị cấu trúc của các frame: Cả 3 frame đều trên cùng một cửa sổ cha, mặc dù 2 trong số các frame đó nằm trong một frameset khác.

Top

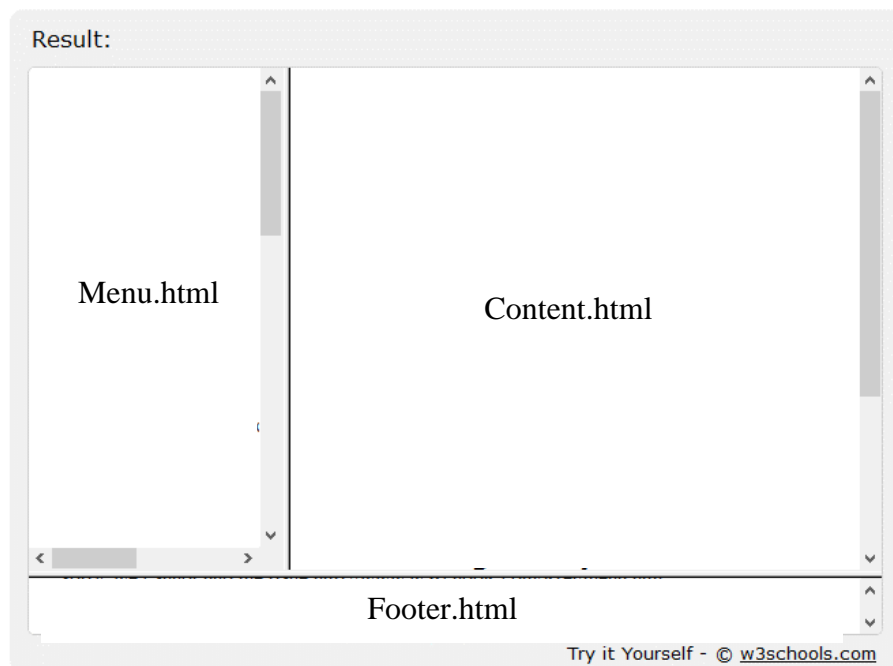


Ta có thể gọi tới những frame trước đó bằng cách sử dụng thuộc tính **frames** như sau:

menuFrame chính là top.frames[0]

contentFrame chính là top.frames[1]

footerFrame chính là top.frames[2]



*Hình: Kết quả việc tạo frame*

#### **5.10.5. Document**

Đối tượng này chứa các thông tin về document hiện thời và cung cấp các phương thức để đưa thông tin ra màn hình.

##### *a. Các thuộc tính*

- ✓ alinkColor - Giống nh thuộc tính ALINK.
- ✓ anchor - Mảng tất cả các anchor trong document.
- ✓ bgColor - Giống thuộc tính BGCOLOR.

- ✓ cookie - Sử dụng để xác định cookie.
- ✓ fgColor - Giống thuộc tính TEXT.
- ✓ forms - Mảng tất cả các form trong document.
- ✓ lastModified - Ngày cuối cùng văn bản được sửa.
- ✓ linkColor - Giống thuộc tính LINK.
- ✓ 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>.
- ✓ vlinkColor - Giống thuộc tính VLINK.

#### *b. 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 (["mimeType"]) - Mở một stream để thu thập dữ liệu vào của các phương 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 Phương thức trên nhưng khi hết mỗi biểu thức lại xuống dòng.

#### **5.10.6. Forms**

Form là các thành phần chứa trong cặp thẻ <FORM> và </FORM>, thông thường 1 form chứa nhiều đối tượng dùng để tương tác dữ liệu với server. Các phần tử (elements) bao gồm:

- ✓ Button : Nút lệnh
- ✓ Reset : Nút lệnh đặc biệt, có tác dụng khởi tạo giá trị tất cả các phần tử trong form về giá trị mặc định
- ✓ Submit : Nút lệnh đặc biệt, cho phép gửi dữ liệu của form về server
- ✓ Checkbox : Lựa chọn cho phép chọn nhiều
- ✓ Radio : Lựa chọn chỉ cho phép chọn một giá trị
- ✓ Text : Phần tử chứa dữ liệu dạng text
- ✓ Hidden : Phần tử chứa dữ liệu ẩn
- ✓ Password : phần tử nhập/hiển thị dạng
- ✓ Select : Danh sách đơn

✓ **Textarea** : Phần tử chứa dữ liệu dạng đoạn văn

Nếu document chứa một vài form, chúng có thể được tham chiếu qua mảng forms. Số lượng các form có thể được xác định như sau:

`document.forms.length`.

Mỗi một form có thể được tham chiếu như sau:

`document.forms[index]`

#### *a. Các thuộc tính*

<b>action</b>	thuộc tính ACTION của thẻ FORM.
<b>elements</b>	Mảng chứa tất cả các thành phần trong một form (như checkbox, text, danh sách lựa chọn)
<b>encoding</b>	Xâu chứa kiểu MIME được sử dụng để mã hoá nội dung của form gửi cho server.
<b>length</b>	Số lượng các thành phần trong một form.
<b>method</b>	Thuộc tính Method.
<b>target</b>	Xâu chứa tên của cửa sổ đích khi submit form

#### *b. Các phương thức*

<b>reset()</b>	Khởi tạo giá trị tất cả các phần tử trong form về giá trị mặc định.
<b>submit()</b>	Gửi dữ liệu trong form tới trang xử lý. Phương thức này tương tự như nút submit trên form.

#### *c. Các sự kiện*

**onSubmit:** Sự kiện này được kích hoạt trước khi chuyển dữ liệu từ form đi.

### **5.10.7. History**

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

#### *a. Các thuộc tính*

**length** - Số lượng các URL trong đối tượng.

#### *b. Các phương thức*

**history.back()** - Được sử dụng để tham chiếu tới URL mới được thăm trước đây.

**history.forward()** - Được sử dụng để tham chiếu tới URL kế tiếp trong danh sách. Nó sẽ không gây hiệu ứng gì nếu đã đến cuối của danh sách.

**history.go (delta | "location")** - Được sử dụng để chuyển lên hay chuyển xuống hoặc di chuyển đến URL xác định bởi location trong danh sách.

### **5.10.8. Links**

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

Mảng link chứa danh sách tất cả các liên kết trong document. Có thể xác định số lượng các link qua:



`document.links.length()`

Có thể tham chiếu tới một liên kết qua

`document.links [index]`

*a. Các thuộc tính*

- ✓ hash - Tên anchor của vị trí hiện thời (ví dụ topic3).
- ✓ Host - Phần hostname:port của URL (ví dụ www.hau.edu.vn). Chú ý rằng đây thường là cổng ngầm định và ít khi được chỉ ra.
- ✓ Hostname - Tên của host và domain (ví dụ ww.abc.com).
- ✓ href - Toàn bộ URL cho document hiện tại.
- ✓ Pathname - Phần đường dẫn của URL (ví dụ /folder/main.html).
- ✓ port - Cổng truyền thông được sử dụng cho máy tính host, thông thường là cổng ngầm định.
- ✓ Protocol - Giao thức được sử dụng (cùng với dấu hai chấm) (ví dụ http:).
- ✓ Search - Câu truy vấn tìm kiếm có thể ở cuối URL cho các script CGI.
- ✓ Target - Giống thuộc tính TARGET của <LINK>.

*b. Các sự kiện*

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

## 5.11. LÀM VIỆC VỚI FORMS

Như ta đã biết, form thường dùng để gửi dữ liệu lên server hoặc hiển thị dữ liệu theo một định dạng nhất định. Trong phần này ta sẽ làm việc cụ thể với các thành phần của form.

***Các phần tử của form***

Phần tử	Mô tả
button	Là một nút bấm thường (cũng có thể là nút submit hay nút reset) (<INPUT TYPE="button">)
checkbox	Một checkbox (<INPUT TYPE="checkbox">)
FileUpload	Là một phần tử tải file lên cho phép người sử dụng gửi lên một file (<INPUT TYPE="file">)
hidden	Một trường ẩn (<INPUT TYPE="hidden">)
password	Một trường text để nhập mật khẩu mà tất cả các ký tự nhập vào đều hiển thị là dấu (*)(<INPUT TYPE="password">)
radio	Một nút bấm (<INPUT TYPE="radio">)

reset	Một nút reset(<INPUT TYPE="reset">)
select	Một danh sách lựa chọn (<SELECT><OPTION>option1</OPTION> <OPTION>option2</OPTION></SELECT>)
submit	Một nút submit (<INPUT TYPE="submit">)
text	Một trường text (<INPUT TYPE="text">)
textArea	Một trường text cho phép nhập vấp nhiều dòng <TEXTAREA>default text</TEXTAREA>)

Mỗi phần tử có thể được đặt tên để JavaScript truy nhập đến chúng qua tên

#### 5.11.1 Thuộc tính type

Trong mỗi phần tử của form đều có thuộc tính type, đó là một xâu chỉ định rõ kiểu của phần tử được đưa vào như nút bấm, một trường text hay một checkbox... Type có các giá trị sau:

- ✓ Text field: "text"
- ✓ Radio button: "radio"
- ✓ Checkbox: "checkbox"
- ✓ Hidden field: "hidden"
- ✓ Submit button: "submit"
- ✓ Reset button: "reset"
- ✓ Password field: "password"
- ✓ Button: "button"
- ✓ Select list: "select-one"
- ✓ Multiple select lists: "select-multiple"
- ✓ Textarea field: "textarea"

##### a. Phần tử Button

Một phần tử button được thể hiện khi sử dụng thẻ INPUT:

```
<INPUT TYPE="button" NAME="name" VALUE="click me">
```

Trong thẻ INPUT, name là tên của button, thuộc tính VALUE chứa nhãn của nút.

Chỉ có một thẻ sự kiện duy nhất đối với button là **onClick**. Phần tử button có khả năng mở rộng cho phép người lập trình JavaScript có thể viết được một đoạn mã lệnh JavaScript để thực thi việc thêm vào một nút bấm trong một script.

Trong ví dụ sau, thay vì sử dụng onChange, ta có thể chỉnh sửa script để định giá biểu thức khi button được bấm.

**Ví dụ:**

```
<HTML>
```

```

<HEAD>
<TITLE>button Example</TITLE>
<SCRIPT LANGUAGE="JavaScript">
function calculate(form)
{
    form.results.value = form.tuoi.value+100;
}
// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</HEAD>
<BODY>
<FORM METHOD=POST>
Giá trị 1:
<INPUT TYPE="text" NAME="tuoi" VALUE="">
<BR>
Giá trị 1 nối với 100 là:
<INPUT TYPE=text NAME="results" onFocus="this.blur();">
<BR>
<INPUT TYPE="button" VALUE="Calculate" onClick="calculate(this.form);">
</FORM>
</BODY>
</HTML>

```

### 5.11.2 Phần tử Text

Phần tử Text hay được sử dụng nhất trong các form HTML. Trường text cho phép nhập vào một dòng đơn và không chứa ký tự xuống dòng.

#### a. Các thuộc tính

- ✓ defaultValue : Chứa giá trị mặc định
- ✓ name : Tên của textbox
- ✓ value : Giá trị của textbox

#### b. Các phương thức

- ✓ focus() : Nhảy vào textbox để nhập dữ liệu
- ✓ blur() : Thoát khỏi trường dữ liệu
- ✓ select() : Bôi đen (select) toàn bộ giá trị trong trường này

#### c. Các sự kiện chính

Sự kiện	Mô tả
---------	-------

onBlur	Khi người dùng thoát khỏi trường thông tin
onFocus	Khi người dùng chọn hoặc click chuột vào trường thông tin
onChange	Khi người dùng thay đổi giá trị của trường thông tin
onSelect	Khi người dùng bôi đen (select) giá trị của trường thông tin
onKeyPress	Khi người dùng gõ phím trong trường thông tin (nhập giá trị)

### 5.11.3 Phần tử Checkbox

Các phần tử checkbox có khả năng bật tắt dùng để chọn hoặc không chọn một thông tin. Bảng dưới đây là danh sách các thuộc tính và các cách thức của phần tử checkbox.

#### *Các thuộc tính và cách thức của phần tử checkbox*

Cách thức và thuộc tính	Mô tả
checked	Cho biết trạng thái hiện thời của checkbox (thuộc tính)
defaultChecked	Cho biết trạng thái mặc định của phần tử (thuộc tính)
name	Cho biết tên của phần tử được chỉ định trong thẻ INPUT (thuộc tính)
value	Cho biết giá trị hiện thời của phần tử được chỉ định trong thẻ INPUT (thuộc tính)
click()	Mô tả một click vào checkbox (Cách thức)

Phần tử checkbox chỉ có một thẻ sự kiện là onClick

Ví dụ: Tạo hộp checkbox để nhập vào một số rồi lựa chọn tính nhân đôi và bình phương:

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>checkbox Example</TITLE>
```

```
<SCRIPT>
```

```
<!-- HIDE FROM OTHER BROWSERS
```

```
function calculate(form,callingField) {
```

```
    if (callingField == "result") {    // if(1)
```

```
    if (form.square.checked) {        // if(2)
```

```
        form.entry.value = Math.sqrt(form.result.value);
```

```
    }
```

```
else {
```

```
    form.entry.value = form.result.value / 2;
```

```
    } //end if(2)
```

```
}
```

```
else{
```

```

    if (form.square.checked) { // if(3)
        form.result.value=form.entry.value*form.entry.value;
    }
    else {
        form.result.value = form.entry.value * 2;
    } //enfzd if(3)
} //end if(1)

} //end function
// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</HEAD>
<BODY>
<FORM METHOD=POST>
Value: <INPUT TYPE="text" NAME="entry" VALUE=0
onChange="calculate(this.form,this.name);">
<BR>
Action (default double): <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>

```

Trong script này, ta thấy cách sử dụng thẻ sự kiện onClick cũng như thuộc tính checked là một giá trị kiểu Boolean có thể dùng làm điều kiện trong câu lệnh if...else

Ta có thể thêm một checkbox tên là square vào form. Nếu hộp này được check, Chương trình sẽ lấy giá trị của nó, nếu không, một thực thi được mặc định sẽ nhân đôi giá trị của nó. Thẻ sự kiện onClick trong checkbox được định nghĩa:

```
(<INPUT TYPE=checkbox NAME=square onClick= "calculate( this.form, this.name);"> )
```

Khi đó nếu người dùng thay đổi một câu lệnh khác, form sẽ được tính toán lại.

Để tạo ra sự mở rộng cho checkbox, ta có thể thay đổi hàm calculate() như sau:

```

function calculate(form,callingField) {
    if (callingField == "result") { // if (1)
    if (form.square.checked) { // if (2)
        form.entry.value = Math.sqrt(form.result.value);
    }
    else {

```

```

        form.entry.value = form.result.value / 2;
    } //end if (2)
}
else {
    if (form.square.checked) {          // if (3)
        form.result.value=form.entry.value*form.entry.value;
    }
else {
    form.result.value = form.entry.value * 2;
    } // end if (3)
} // end if (1)
}

```

#### 5.11.4 *Phần tử File Upload*

Phần tử này cung cấp cho form một cách để người sử dụng đưa một file từ máy tính vào form để gửi lên server xử lý. Phần tử file Upload được chỉ định rõ trong JavaScript bằng đối tượng FileUpload.

Đối tượng chỉ có hai thuộc tính là name và value, cả hai đều là giá trị xâu như các đối tượng khác. Không có cách thức hay thẻ file cho đối tượng này.

#### 5.11.5 *Phần tử Hidden*

Phần tử hidden là phần tử duy nhất trong số tất cả các phần tử của form không được hiển thị trên Web browser. Trường hidden có thể sử dụng để lưu các giá trị cần thiết để gửi tới server song không cần hiển thị ra form. Mọi người có thể sử dụng trong JavaScript để lưu các giá trị trong suốt một script và để tính toán không cần form.

Đối tượng hidden chỉ có hai thuộc tính là name và value, đó cũng là những giá trị xâu giống các đối tượng khác. Không có cách thức hay thẻ sự kiện nào cho đối tượng này.

#### 5.11.6 *Phần tử Password*

Đối tượng Password dùng để che giấu thông tin đối với users, hoặc để tránh lộ mật khẩu: khi gõ bất kỳ ký tự nào vào cũng đều hiển thị dấu sao (\*). Thông thường nó được sử dụng với những thông tin bí mật hoặc đăng ký mật khẩu...

Đối tượng Password có 3 thuộc tính giống trường text là: defaultValue, name và value. Không giống với hai phần tử ở trên, trường Password có nhiều cách thức hơn (focus(), blur(), and select() ) và tương ứng với các thẻ sự kiện: onFocus, onBlur, and onSelect.

#### 5.11.7 *Phần tử Radio*

Đối tượng radio gần giống sự bật tắt checkbox khi có hai nút radio kết hợp thành một nhóm. Khi nhiều radio được kết hợp thành một nhóm, chỉ có một nút được chọn trong bất kỳ một thời điểm nào. Ví dụ dòng lệnh sau tạo ra một nhóm radio có ba nút tên là test:

```
<INPUT TYPE="radio" NAME="test" VALUE="1" checked>1<BR>
```

<INPUT TYPE="radio" NAME="test" VALUE="2">2<BR>

<INPUT TYPE="radio" NAME="test" VALUE="3">3<BR>

Nhóm các nút radio lại bằng cách đặt cho chúng có cùng một tên trong các thẻ INPUT.

Có một vài thuộc tính để kiểm tra trạng thái hiện thời của một nhóm nút radio. Bảng sau hiển thị các thuộc tính và cách thức của đối tượng radio.

***Các thuộc tính và cách thức của đối tượng radio.***

Thuộc tính và cách thức	Mô tả
checked	Mô tả trạng thái hiện thời của phần tử radio (thuộc tính)
defaultChecked	Mô tả trạng thái mặc định của phần tử (thuộc tính)
index	Mô tả thứ tự của nút radio được chọn hiện thời trong một nhóm
length	Mô tả tổng số nút radio trong một nhóm
name	Mô tả tên của phần tử được chỉ định trong thẻ INPUT (thuộc tính)
value	Mô tả giá trị hiện thời của phần tử được định ra trong thẻ INPUT (thuộc tính)
click()	Mô phỏng một click trên nút radio (cách thức)

Cũng như checkbox, radio chỉ có một thẻ sự kiện là onClick.

Không có bất kỳ một đối tượng form nào có thuộc tính index và length. Do một nhóm radio gồm nhiều phần tử radio, nên chúng được đặt trong một mảng các nút radio và được đánh số từ 0. Trong ví dụ nhóm radio có tên test ở trên, nếu nhóm đó nằm trong một form có tên là "testform", bạn có thể gọi tới nút radio thứ hai bằng tên "testform.test[1]" và có thể kiểm tra giá trị của nó bằng "testform.test[1].checked"

Để minh họa cách dùng đối tượng radio, ta xem ví dụ sau:

Ví dụ:

<HTML>

<HEAD>

<TITLE>radio button Example</TITLE>

<SCRIPT>

<!-- HIDE FROM OTHER BROWSERS

function calculate(form,callingField) {

  if (callingField == "result") {

    if (form.action[1].checked) {

      form.entry.value = Math.sqrt(form.result.value);

    } else {

      form.entry.value = form.result.value / 2;

```

    }
  } else {
    if (form.action[1].checked) {
      form.result.value=form.entry.value*form.entry.value;
    } else {
      form.result.value = form.entry.value * 2;
    }
  }
}

// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</HEAD>
<BODY>
<FORM METHOD=POST>
Value: <INPUT TYPE="text" NAME="entry" VALUE=0
      onChange="calculate(this.form,this.name);"> <BR>
Action:<BR>
<INPUT TYPE="radio" NAME="action" VALUE="twice"
      onClick="calculate(this.form,this.name);"> Double<BR>
<INPUT TYPE="radio" NAME="action" VALUE="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>

```

Trong ví dụ này, sự thay đổi từ checkbox ở trên là rất khó nhận biết. Thay cho một checkbox trong ví dụ trước, ở đây ta sử dụng hai nút radio với hai giá trị khác nhau: double và square

Như ta đã biết có thể truy nhập đến các nút radio qua một mảng, do đó hai nút này có thể truy nhập bằng **action[0]** và **action[1]**. Bằng cách này, bạn chỉ cần thay đổi tham chiếu đến hàm *calculate()* từ **form.square.checked** thành **form.action[1].checked**.

#### 5.11.8 *Phân tử Select*

Danh sách lựa chọn dưới dạng menu drop-down hoặc danh sách cuộn có thể được lựa chọn. Các danh sách được hiển thị khi sử dụng hai thẻ SELECT và OPTION. Ví dụ:

```

<SELECT NAME="test">
  <OPTION SELECTED value="1">Lựa chọn 1
  <OPTION value="2">Lựa chọn 2

```



```
<OPTION value="3">Lựa chọn 3
</SELECT>
```

Trong ví dụ trên, ta đã tạo ra ba thành phần của menu thả drop-down với ba lựa chọn Lựa chọn 1, Lựa chọn 2 và Lựa chọn 3. Tuy nhiên khi lựa chọn thì kết quả sẽ cho ra là “1”, “2”, “3”. Nếu trong thẻ `OPTION` không có thuộc tính `value` thì giá trị lựa chọn sẽ chính là giá trị `Text`.

Sử dụng thuộc tính `SIZE` bạn có thể tạo ra một danh sách cuộn với số phần tử hiển thị ở lần thứ nhất. Để bật menu drop-down trong một menu cuộn với hai thành phần hiển thị, bạn có thể sử dụng như sau:

```
<SELECT NAME="test" SIZE=2>
  <OPTION SELECTED>1
  <OPTION>2
  <OPTION>3
</SELECT>
```

### Chú ý:

✓ Mặc định trong cả hai ví dụ trên, người sử dụng chỉ có thể có một lựa chọn. Nếu sử dụng thuộc tính `MULTIPLE`, bạn có thể cho phép người sử dụng lựa chọn nhiều hơn một giá trị trong danh sách lựa chọn:

```
<SELECT NAME="test" SIZE=2 MULTIPLE>
<OPTION SELECTED>1
<OPTION>2
<OPTION>3
</SELECT>
```

Với các thành phần lựa chọn, danh sách các lựa chọn được chứa trong một mảng được đánh số từ 0. Trong trường hợp này, mảng là một thuộc tính của đối tượng **select** gọi là **options**.

Cả việc lựa chọn các option và từng phần tử option riêng biệt đều có những thuộc tính. Bổ sung thêm vào mảng option, phần tử select có thuộc tính **selectedIndex**, có chứa số thứ tự của option được lựa chọn hiện thời.

Mỗi option trong danh sách lựa chọn đều có một vài thuộc tính:

- ✓ **defaultselected**: cho biết option có được mặc định là lựa chọn trong thẻ `OPTION` hay không.
- ✓ **index**: chứa giá trị số thứ tự của option hiện thời trong mảng option.
- ✓ **selected**: cho biết trạng thái hiện thời của option
- ✓ **text**: có chứa giá trị của dòng text hiển thị trên menu cho mỗi option, và thuộc tính `value` mọi giá trị chỉ ra trong thẻ `OPTION`.

Đối tượng select không có các cách thức được định nghĩa sẵn. Tuy nhiên, đối tượng select có ba thẻ sự kiện, đó là `onBlue`, `onFocus`, `onChange`, chúng đều là những đối tượng text.

Ví dụ bạn có danh sách lựa chọn sau:

```
<SELECT NAME="example" onFocus="react();">
  <OPTION SELECTED VALUE="Number One">1
  <OPTION VALUE="The Second">2
  <OPTION VALUE="Three is It">3
</SELECT>
```

Bạn có thể truy nhập tới các thông tin sau:

```
example.options[1].value = "The Second"
```

```
example.options[2].text = "3"
```

```
example.selectedIndex = 0
```

```
example.options[0].defaultSelected = true
```

```
example.options[1].selected = false
```

Nếu người sử dụng kích vào menu và lựa chọn option thứ hai, thì thẻ onFocus sẽ thực hiện, và khi đó giá trị của thuộc tính sẽ là:

```
example.options[1].value = "The Second"
```

```
example.options[2].text = "3"
```

```
example.selectedIndex = 1
```

```
example.options[0].defaultSelected = true
```

```
example.options[1].selected = true
```

### 5.11.9 *Phần tử Reset*

Sử dụng đối tượng reset, bạn có thể tác động ngược lại để click vào nút Reset. Cũng giống đối tượng button, đối tượng reset có hai thuộc tính là name và value, và một cách thức click(), một thẻ sự kiện onClick.

Hầu hết những người lập trình không sử dụng thẻ sự kiện onClick của nút reset để kiểm tra giá trị của nút này, đối tượng reset thông dùng để xoá form.

Ví dụ sau minh hoạ cách sử dụng nút reset để xoá các giá trị của form.

Ví dụ:

```
<HTML>
<HEAD>
<TITLE>reset Example</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!-- HIDE FROM OTHER BROWSERS
function clearForm(form) {
form.value1.value = "Form";
  form.value2.value = "Cleared";
}
// STOP HIDING FROM OTHER BROWSERS -->
//SCRIPT>
```

```

</HEAD>
<BODY>
<FORM METHOD=POST>
<INPUT TYPE="text" NAME="value1"><BR>
<INPUT TYPE="text" NAME="value2"><BR>
<INPUT TYPE="reset" VALUE="Clear Form" onClick="clearForm(this.form);">
</FORM>
</BODY>
</HTML>

```

#### 5.11.10 *Phần tử Submit*

Nút Submit là một trường hợp đặc biệt của button, cũng như nút Reset. Nút này đưa thông tin hiện tại từ các trường của form tới địa chỉ URL được chỉ ra trong thuộc tính ACTION của thẻ form sử dụng cách thức METHOD chỉ ra trong thẻ FORM.

Giống như đối tượng button và reset, đối tượng submit có sẵn thuộc tính name và value, cách thức click() và thẻ sự kiện onClick.

#### 5.11.11 *Phần tử Textarea*

Thẻ TEXTAREA gần giống phần tử Text ở chỗ cung cấp một hộp cho phép dữ liệu dạng text, tuy nhiên nó cho phép xuống dòng với số dòng text do người thiết kế định trước. Ví dụ:

```

<TEXTAREA NAME="fieldName" ROWS=10 COLS=25>
    Default Text Here
</TEXTAREA>

```

Trong ví dụ này, ta tạo ra một trường text cho phép đa vào 10 hàng, mỗi hàng 25 ký tự. Dòng "Default Text Here" sẽ xuất hiện trong trường này vào lần hiển thị đầu tiên.

Cũng như phần tử text, JavaScript cung cấp cho bạn các thuộc tính defaultValue, name, và value, các cách thức focus(), select(), và blur(), các thẻ sự kiện onBlur, onFocus, onChange, onSelect.

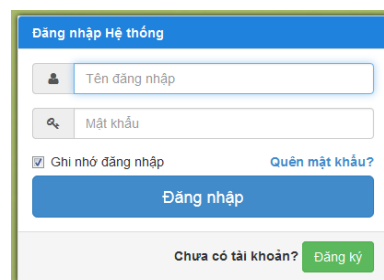
## BÀI TẬP CUỐI CHƯƠNG 5

Dùng JavaScript để thực hiện các yêu cầu sau:

1. Tạo và viết mã lệnh JavaScript mô phỏng 1 máy tính với 4 phép tính, có giao diện như trang sau:

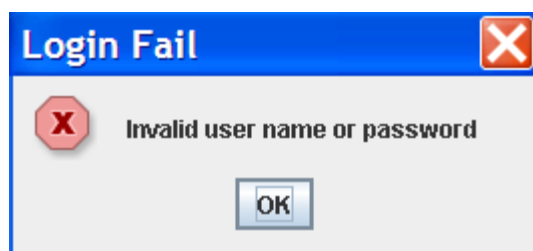


2. Tạo form có tiêu đề **Đăng nhập Hệ thống** như hình dưới và viết mã lệnh JavaScript cho sự kiện click chuột:



Khi click vào nút **Đăng nhập**, ứng dụng này sẽ kiểm tra User name và Password:

- ✓ Nếu User name="admin" và Password="123" thì sẽ hiển thị form quản lý sinh viên trong tệp Qlsv.html.
- ✓ Nếu Username hoặc Password sai thì hệ thống sẽ xuất hiện thông báo: "Sai tên truy cập hoặc mật khẩu"



✓ Nếu Username hoặc Password để trống thì hệ thống sẽ xuất hiện thông báo: “Tên đăng nhập không được để trống”

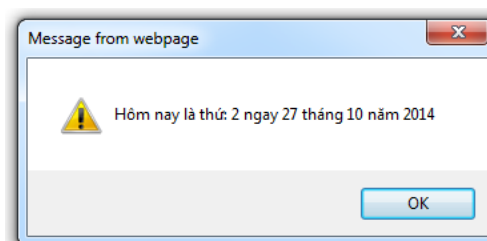
3. Giải phương trình bậc 1:  $ax+b=0$

4. Giải và biện luận phương trình bậc 2:  $ax^2+bx+c=0$

✓ Nhập a, b, c trong 3 textbox tương ứng

✓ Khi click vào nút “OK” thì sẽ tiến hành giải phương trình bậc 2

5. Hãy hiển thị ngày và giờ của hệ thống máy tính khi trang Web được nạp. Thông tin hiển thị ra có dạng như sau:



*Hướng dẫn:* Sử dụng đối tượng Date và sử dụng các hàm lấy thứ, ngày, tháng, năm để in thông tin ra màn hình. Chú ý đến các hàm tính tháng, ngày trong tuần bị giảm một đơn vị.

6. Viết chương trình kiểm tra người dùng nhập địa chỉ email có hợp lệ.

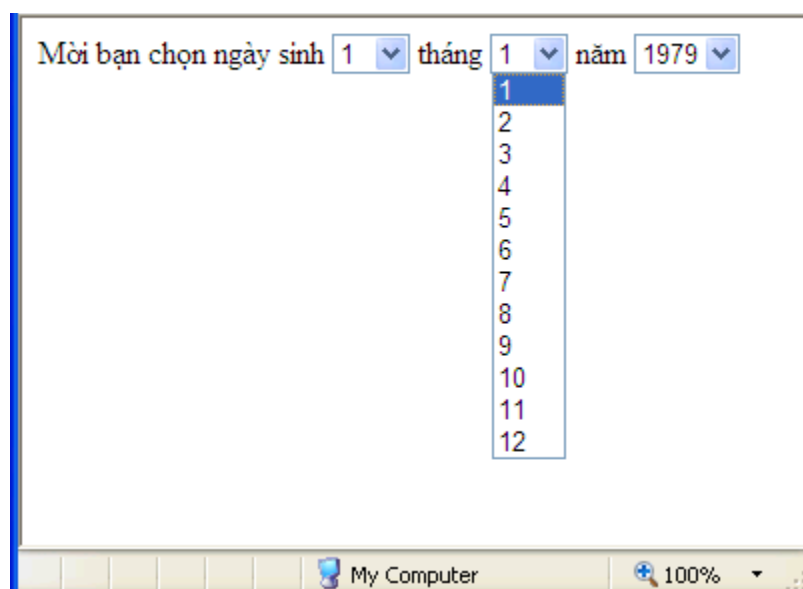
*Hướng dẫn:* Địa chỉ email hợp lệ là địa chỉ có dạng `abc@domain`. Trong đó:

✓ Bắt buộc phải có 1 ký tự @

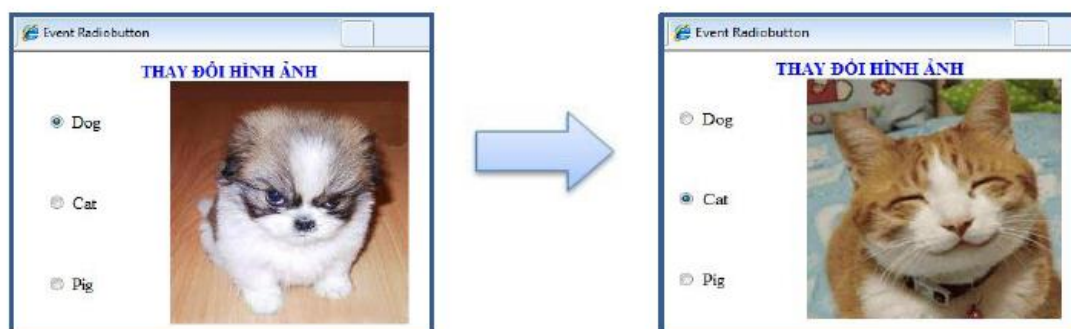
✓ Ký tự @ không được ở đầu hoặc ở cuối

✓ domain phải có ký tự “.”

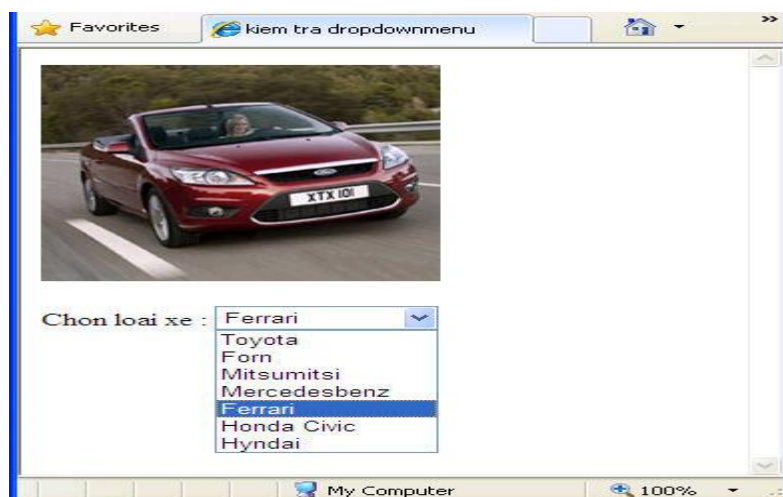
7. Sử dụng JavaScript viết chương trình tạo 3 Dropdown (<select>), trong đó ngày nhận giá trị trong khoảng [1..31], tháng nhận giá trị trong khoảng [1..12], năm nhận giá trị trong khoảng [1900...2100]:







8. Thiết kế 1 trang gồm 3 Radio Button cho phép người dùng thay đổi hình ảnh trong trang sau:



9. Thiết kế trang chứa Dropdown list chứa các hình (khoảng 7 hình), các option có value là đường dẫn các file hình cho trước. Viết sự kiện cho Dropdown (onSelect()) cho phép người sử dụng chọn hình nào thì hình sẽ được thay đổi



10. Thiết kế trang như sau. Khi trang Web load lên hiển thị 1 window với message **“Welcome to Shopping Flower”**. Khi người sử dụng nhập vào trường số lượng và rời khỏi trường này thì ô Tổng thành tiền sẽ được cập nhật (lưu ý nếu nhập vào không phải là số thì thông báo yêu cầu nhập lại). Khi đóng trang Web thì 1 windows có Message **“Bạn có muốn kết thúc không?”**. Nếu OK thì đóng, ngược lại thì không.

CỬA HÀNG BONSAI			
STT	Hình ảnh	Giá	Số lượng mua
1		2000	<input type="text"/>
2		1500	<input type="text"/>
3		1200	<input type="text"/>
4		1900	<input type="text"/>
		Tổng thành tiền	0 VNĐ

Done My Computer

11. Thiết kế Form như sau:

THEM SINH VIÊN	
Họ tên	<input type="text"/>
Lớp	<input type="text"/>
<input type="button" value="Nhập"/>	
DANH SÁCH SINH VIÊN	
HỌ TÊN	LỚP
<input type="text"/>	<input type="text"/>

Done My Computer

Viết chương trình xử lý theo yêu cầu sau:

1. Khi trang load lên thì nút Nhập ở chế độ mờ và con trỏ nhập liệu ở ô Họ tên.
2. Khi rời khỏi trường Lớp nếu trường Họ tên và Lớp khác rỗng thì nút Nhập sáng lên.
3. Bấm nút Nhập thì dữ liệu hiển thị vào bảng “Danh Sách Sinh Viên” ở bên dưới và nút Nhập lại mờ.

THEM SINH VIEN	
Họ tên	<input type="text"/>
Lớp	<input type="text"/>
<input type="button" value="Nhập"/>	
DANH SACH SINH VIEN	
HỌ TÊN	LỚP
Phạm Thanh Sang	CĐTH5A
Dương Yến Phượng	CĐTH5B

Done My Computer



## TÀI LIỆU THAM KHẢO

- 1 Jon Duckett(2014) , *Javascript and JQuery Interactive Front-End Web Development*, Wiley
- 2 <http://www.w3schools.com>
- 3 [www.javascriptsource.com](http://www.javascriptsource.com)

