

7.2	MÔI TRƯỜNG VIẾT JAVASCRIPT	48
7.2.1	Lệnh đơn và khối lệnh.....	48
7.2.2	Xuất dữ liệu ra trang Web	48
7.3	BIẾN VÀ DỮ LIỆU TRONG JAVASCRIPT.....	49
7.3.1	Biến	49
7.3.2	Dữ liệu: Có 4 loại dữ liệu	50
7.3.3	Toán tử.....	50
7.4	CHƯƠNG 14. HÀM TRONG JAVASCRIPT	53
7.4.1	Định nghĩa	53
7.4.2	Cách gọi hàm	53
7.4.3	CÁC HÀM THÔNG DỤNG TRONG JAVASCRIPT	54
7.5	CHƯƠNG 14. CÁC CẤU TRÚC ĐIỀU KIỆN.....	57
7.5.1	Cấu trúc lựa chọn	57
7.5.2	CẤU TRÚC LẮP	59
7.6	CHƯƠNG 14. MÔ HÌNH ĐỐI TƯỢNG	64
8.1.1	MÔ HÌNH DOM ((Document Object Model).....	64
8.1.2	Xây dựng một đối tượng mới	66
8.1.3	CÁC ĐỐI TƯỢNG CÓ SẴN TRONG JAVASRIPT.....	67
8.1.4	Đối tượng trình duyệt (Navigator Object).....	76
8.1.5	THAY ĐỔI NỘI DUNG ĐỘNG TRÊN TRANG.....	88

CHƯƠNG 1. GIỚI THIỆU VỀ WEB

1.1 CÁC KHÁI NIỆM CƠ BẢN

- *Internet* là một mạng máy tính toàn cầu trong đó các máy truyền thông với nhau theo một ngôn ngữ chung là TCP/IP.
- *Intranet* đó là mạng cục bộ không nối vào Internet và cách truyền thông của chúng cũng theo ngôn ngữ chung là TCP/IP.
- *Mô hình Client-Server*: là mô hình khách-chủ. Server chứa tài nguyên dùng chung cho nhiều máy khách(Client) như các tập tin, tài liệu, máy in... Ưu điểm của mô hình này là tiết kiệm về thời gian, tài chính, dễ quản trị hệ thống... Cách hoạt động của mô hình này là máy Server ở trạng thái hoạt động(24/24) và chờ yêu cầu từ phía Client. Khi Client yêu cầu thì máy Server đáp ứng yêu cầu đó.
- *Internet Server* là các Server cung cấp các dịch vụ Internet(Web Server, Mail Server, FTP Server...)
- *Internet Service Provider(ISP)*: Là nơi cung cấp các dịch vụ Internet cho khách hàng. Mỗi ISP có nhiều khách hàng và có thể có nhiều loại dịch vụ Internet khác nhau.
- *Internet Protocol* : Các máy sử dụng trong mạng Internet liên lạc với nhau theo một tiêu chuẩn truyền thông gọi là Internet Protocol (IP). IP Address-địa chỉ IP: để việc trao đổi thông tin trong mạng Internet thực hiện được thì mỗi máy trong mạng cần phải định danh để phân biệt với các máy khác. Mỗi máy tính trong mạng được định danh bằng một nhóm các số được gọi là địa chỉ IP. Địa chỉ IP gồm 4 số thập phân có giá từ 0 đến 255 và được phân cách nhau bởi dấu chấm. Ví dụ 192.168.0.1 Địa chỉ IP này có giá trị trong toàn mạng Internet. Ủy ban phân phối địa chỉ IP của thế giới sẽ phân chia các nhóm địa chỉ IP cho các quốc gia khác nhau. Thông thường địa chỉ IP của một quốc gia do các cơ quan bưu điện quản lý và phân phối lại cho các ISP. Một máy tính khi thâm nhập vào mạng Internet cần có một địa chỉ IP. Địa chỉ IP có thể cấp tạm thời hoặc cấp vĩnh viễn. Thông thường các máy Client kết nối vào mạng Internet thông qua một ISP bằng đường điện thoại. Khi kết nối, ISP sẽ cấp tạm thời một IP cho máy Client.
- *Phương thức truyền thông tin trong Internet*: Khi một máy tính có địa chỉ IP là x(máy X) gửi tin đến máy tính có địa chỉ IP là y (máy Y) thì phương thức truyền tin cơ bản diễn ra như sau: Nếu máy X và máy Y cùng nằm trên một mạng con thì thông tin sẽ được gửi đi trực tiếp. Còn máy X và Y không cùng nằm trong mạng con thì thông tin sẽ được chuyển tới một máy trung gian có đường thông với các mạng khác rồi mới chuyển tới máy Y. Máy trung gian này gọi là Gateway.
- *World Wide Web(WWW)*: là một dịch vụ phổ biến nhất hiện nay trên Internet. Dịch vụ này đưa ra cách truy xuất các tài liệu của các máy phục vụ dễ dàng thông qua các giao tiếp đồ họa. Để sử dụng dịch vụ này máy Client cần có một chương trình gọi là Web Browser.
- *Web Browser(trình duyệt)*: là trình duyệt Web. Dùng để truy xuất các tài liệu trên các Web Server. Các trình duyệt hiện nay là Internet Explorer, Netscape
- *Home page*: là trang web đầu tiên trong web site
- *Hosting provider*: là công ty hoặc tổ chức đưa các trang của chúng ta lên web
- *Hyperlink* : tên khác của hypertextlink
- *Publish*: làm cho trang web chạy được trên mạng
- *URL(Uniform resource locator)*: một địa chỉ chỉ đến một file cụ thể trong nguồn tài nguyên mạng.

- Mỗi nguồn trên web có duy nhất một địa chỉ rất khó nhớ. Vì vậy, người ta sử dụng URL là một chuỗi cung cấp địa chỉ Internet của một web site hoặc nguồn trên World Wide Web. Định dạng đặc trưng là:

www.nameofsite.typeofsite.countrycode

Ví dụ:

207.46.130.149 được biểu diễn trong URL là www.microsoft.com

- URL cũng nhận biết giao thức của site hoặc nguồn được truy cập. Giao thức thông thường nhất là “http”, một vài dạng URL khác là “gopher”, cung cấp địa chỉ Internet của một thư mục Gopher, và “ftp”, cung cấp vị trí mạng của nguồn FTP.
 - Có hai dạng URL:
 - ✧ *URL tuyệt đối* – là địa chỉ Internet đầy đủ của một trang hoặc file, bao gồm giao thức, vị trí mạng, đường dẫn tùy chọn và tên file.
Ví dụ, [http:// www.microsoft.com/ms.htm](http://www.microsoft.com/ms.htm).
 - ✧ *URL tương đối* - mô tả ngắn gọn địa chỉ tập tin kết nối có cùng đường dẫn với tập tin hiện hành, URL tương đối đơn giản bao gồm tên và phần mở rộng của tập tin.
Ví dụ: [index.html](#)
- *Web server* là một chương trình đáp ứng lại các yêu cầu truy xuất tài nguyên từ trình duyệt.

1.2 GIỚI THIỆU KHÁI QUÁT VỀ WEB

- *Web* là một ứng dụng chạy trên mạng(Client-Server), được chia sẻ khắp toàn cầu.
- *Trang web* là một file văn bản chứa những tag HTML hoặc những đoạn mã đặc biệt mà trình duyệt web (Web browser) có thể hiểu và thông dịch được, file được lưu với phần mở rộng là .html hoặc htm.
- *HTML (HyperText Markup Language)*, gồm các đoạn mã chuẩn được quy ước để thiết kế Web và được hiển thị bởi trình duyệt Web (Web Browser)
 - *Hypertext* (Hypertext link), là một từ hay một cụm từ đặc biệt dùng để tạo liên kết giữa các trang web
 - *Markup*: là cách định dạng văn bản để trình duyệt hiểu và thông dịch được.
 - *Language*: đây không là ngôn ngữ lập trình, mà chỉ là tập nhỏ những quy luật để định dạng văn bản trên trang web.
- *Trình soạn thảo trang web* :Có thể soạn thảo web trên bất kỳ trình soạn thảo văn bản nào. Các trình soạn thảo phổ biến hiện nay là: Notepad, FrontPage hoặc Dreamweaver.

1.3 TAG HTML

Tag HTML là những câu lệnh nằm giữa cặp tag “<” và “>”, dùng để định dạng các văn bản trên trang web. Dạng chung của một tag HTML là:

<tagName ListProperties> Object </tagName>

Trong đó:

- *TagName* : là tên một tag HTML, viết liền với dấu “<”, không có khoảng trắng
- *Object* : là đối tượng cần định dạng trong trang Web
- *ListPropeties* là danh sách thuộc tính của Tag, là những đặc điểm bổ sung vào cho một tag, thứ tự các thuộc tính trong một tag là tùy ý. Nếu có từ 2 thuộc tính trở lên thì mỗi thuộc tính cách nhau bởi khoảng trắng.

- <TagName property1='value1' property2='value2'...>Object</TagName>**
- Giá trị của thuộc tính được đặt trong nháy đơn ‘ hoặc nháy đôi “.(có thể bỏ qua)
 - **<TagName>**: gọi là tag mở
 - **</TagName>**: gọi là tag đóng. Thông thường thì các tag đều có tag đóng. Tuy nhiên có một số tag không có tag đóng
- Ví dụ :** <body BGCOLOR="RED">nội dung </body>
- ↑ ↑ ↑
 TagName(mở) Properties TagName(đóng)
- Có thể có nhiều tag lồng vào nhau, theo nguyên tắc tag nào mở trước thì tag đó đóng sau
- Ví dụ:**
- ```

<Tag1><Tag2>Object</Tag2></Tag1>
Object1<I>Object2 </I>

```
- Trong trang HTML, nếu một tag bị sai thì nội dung bên trong Tag đó không hiển thị trên trình duyệt

## 1.4 CẤU TRÚC CƠ BẢN CỦA TRANG WEB

### 1.4.1 Cấu trúc trang web

- Phần đầu(<Head></Head>): là phần chứa thông tin của trang Web.
- Phần thân (<Body></Body>): là phần chứa nội dung của trang Web.
- Phần đầu và phần thân được đặt trong cặp tag <HTML></HTML>

**<HTML>**

**<HEAD>**

**Nội dung thông tin của trang web**

**</HEAD>**

**<BODY>**

**Nội dung hiển thị trên trình duyệt**

**</BODY>**

**</HTML>**

#### 1. Hiển thị trang web:

- Khởi động trình duyệt Internet Explorer
- Chọn menu file,open, dùng browse tìm tập tin html mới tạo
- Hoặc double click vào tên tập tin .htm

### 1.4.2 Các tag HTML cơ bản

**<Title>:**

**Hiển thị nội dung tiêu đề của trang web trên thanh tiêu đề của trình duyệt.**

- Cặp tag <Title> được đặt trong phần <Head> của trang HTML
- **Cú pháp:**

**<TITLE> Nội dung tiêu đề </TITLE>**

**<Hn>:**

**Tạo header, gồm 6 cấp header, được đặt trong phần BODY**

- **Cú pháp:**

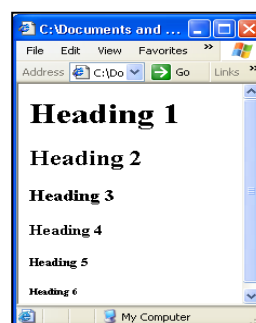
**<Hn ALIGN= "Direction"> Nội dung của Header </Hn>**

Trong đó:

- Direction: gồm các giá trị left, right, center, dùng để canh lề cho header, mặc định là canh trái

- Ví dụ:

```
<H1>Heading 1</H1>
<H2>Heading 2</H2>
<H3>Heading 3</H3>
<H4>Heading 4</H4>
<H5>Heading 5</H5>
<H6>Heading 6</H6>
```



### <P>:

- Dùng để ngắt đoạn và bắt đầu đoạn mới

- **Cú pháp:**

**<P ALIGN = "Direction"> Nội dung của đoạn </P>**

- Tag <P> không bắt buộc.
- Tag <P> kế tiếp sẽ tự động bắt đầu một đoạn mới.

### <BR>:

- Ngắt dòng tại vị trí của của tag.

Ví dụ:

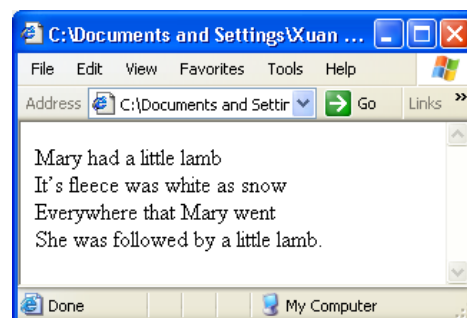
```
<P>
Mary had a little lamb

It's fleece was white as snow

Everywhere that Mary went

She was followed by a little lamb

</p>
```



### <HR>:

- Dùng để kẻ đường ngang trang, không có tag đóng

- **Cú pháp:**

**<HR Align="direction" Width= "Value" Size=value color=#rrggbb>**

Trong đó:

- ✧ Direction: gồm các giá trị left, right, center
- ✧ Width: độ dài đường kẻ, tính bằng Pixel hoặc %
- ✧ Size: độ dày của đường kẻ, tính bằng pixel
- ✧ Color: màu đường kẻ, có thể dùng tên màu hoặc dùng mã #rrggbb

Ví dụ:

```
<HTML>
<HEAD><TITLE>Welcome to HTML </TITLE></HEAD>
<BODY>
<H3> My first HTML document</H3>
<HR size = 5 align = center width = 50%>
<HR size = 15 align = left width = 80%>
<P> This is going to be real fun
</BODY>
</HTML>
```

### <FONT>:

- Dùng định dạng font chữ
- Định dạng Font chữ cho cả tài liệu thì đặt tag <Font> trong phần <Body>
- Định dạng từng phần hoặc từng từ thì đặt tại vị trí muốn định dạng
- **Cú pháp:**

**<FONT Face="fontName1, fontName2, fontName3" size="value"  
Color="rrggbg">**

**Nội dung hiển thị**

**</FONT>**

Ví dụ:

```
<HTML>
 <HEAD>
 <TITLE>Welcome to HTML</TITLE>
 </HEAD>
 <BODY>

 My first HTML document
 <P> This is going
 to be real fun
 </BODY>
</HTML>
```

**<BODY>:**

- Chứa nội dung của trang web
- Cú pháp:

**<BODY>**

**Nội dung chính của trang web**

**</BODY>**

- Các thuộc tính của <Body>
  - ✧ BgColor: thiết lập màu nền của trang
  - ✧ Text: thiết lập màu chữ
  - ✧ Link: màu của siêu liên kết
  - ✧ Vlink: màu của siêu liên kết đã xem qua
  - ✧ Background: dùng load một hình làm nền cho trang
  - ✧ LeftMargin: Canh lề trái
  - ✧ TopMargin: Canh lề trên của trang

Ví dụ:

```
<HTML>
 <HEAD><TITLE> Learning HTML</TITLE></HEAD>
 <BODY BGCOLOR="#0000FF" text="yellow">
 Welcome to
 HTML
 </BODY>
</HTML>
```

- ✧ **Màu sắc:** Internet Explorer có thể xác lập 16 màu theo tên như sau:
  - Black, Silver, Gray, White, Maroon, Red, Purple, Fuchsia, Green, Lime, Olive, Yellow, Navy, Blue, Teal, Aqua.
  - Một số mã thập lục phân của màu :#RRGGBB

Mã thập lục phân	Màu
#FF0000	RED
#00FF00	GREEN
#0000FF	BLUE
#000000	BLACK

#FFFFFF	WHITE
---------	-------

### <IMG>:

- Dùng để chèn một hình ảnh vào trang Web
- **Cú pháp:**

**<Img src="URL" alt="Text" width=value height=value border=value>**

- o Src: xác định đường dẫn tập tin cần load, sử dụng đường dẫn tương đối  
<Img src="../images/h1.gif"> .
- o Alt: chứa nội dung văn bản thay thế cho hình ảnh khi hình không load về được, nếu load về được thì sẽ xuất hiện nội dung trong textbox mỗi khi người dùng đưa chuột tới hình.
- o Width, Height: dùng để xác định chế độ phóng to thu nhỏ hình ảnh.
- o Align =" left/ right/top/bottom": so hàng giữa hình ảnh và text

### <BgSound>:

- Dùng để chèn một âm thanh vào trang Web. Âm thanh này sẽ được phát mỗi khi người sử dụng mở trang Web.
- **Cú pháp:**

**<BgSound src="filenhat" Loop=value>**

- o Src chứa địa chỉ file nhạc, file này có phần mở rộng .mp3 , midi, ...
- o Loop xác định chế độ lặp đi lặp lại của bài hát, nếu value< 0 thì lặp vô hạn, value=n thì lặp lại n lần rồi tự động tắt.

### <EMBED>:

- Cho phép đưa âm thanh trực tiếp vào trang WEB.
- **Cú pháp:**

**<EMBED SRC="URL" >**

**Ví dụ:**

**<EMBED SRC="clouds.mid" WIDTH="145" HEIGHT="61">**

### <Marquee></Marquee>:

- Dùng để điều khiển đối tượng chạy một cách tự động trên trang Web
- **Cú pháp:**

**<Marquee >Object</Marquee>**

- Các thuộc tính của Marquee :
  - o *Direction* = up/ down / left / right dùng để điều khiển hướng chạy.
  - o *Behavior* = alternate: đối tượng chạy từ lề này sang lề kia và ngược lại.

**Ví dụ:**

**<Marquee direction=up>Đối tượng chạy lên </Marquee>**

**<!-- Ghi chú -->:** Nội dung trong cặp tag này không hiển thị trong trang

**Cú pháp: <!-- Nội dung lời chú thích -->**

**Tag <B>: định dạng chữ đậm**

- **Cú pháp**  
**<B> Nội dung chữ đậm</B>**

**Ví dụ:**

**<P><B> This is good fun</B></P>**

**Tag <I>: Định dạng chữ nghiêng**

- **Cú pháp:**

**<I> Nội dung chữ nghiêng </I>**

**Tag <U>: Gạch chân văn bản**

– Cú pháp:

**<U> Nội dung chữ gạch chân </U>**

Ví dụ:

Định dạng khối văn bản vừa đậm, nghiêng và gạch chân

**<B><I><U> Trường ĐHCN TP HCM </U></I></B>**

**Tag <BIG> và <SMALL>:**

– Chính cỡ chữ to hoặc nhỏ hơn cỡ chữ xung quanh

– Cú pháp:

**<BIG>Nội dung chữ to </BIG>**

**<SMALL>Nội dung chữ nhỏ </SMALL>**

**Tag <SUP> và <SUB>:**

– Đưa chữ lên cao hoặc xuống thấp so với văn bản bình thường

– Cú pháp:

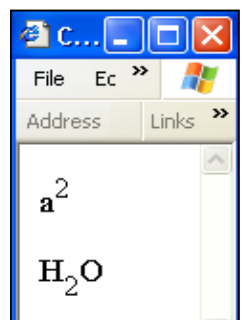
**<SUP>Nội dung chữ đưa lên cao </SUP>**

**<SUB>Nội dung chữ đưa xuống thấp </SUB>**

Ví dụ:

a<SUP>2</SUP>

H<SUB>2</SUB>O



**<STRIKE>:**

– Gạch ngang văn bản

– Cú pháp:

**<STRIKE>Nội dung văn bản bị gạch ngang </STRIKE>**

**<CODE>...</CODE>:**

– Dùng để nhập một dòng mã có định dạng ký tự riêng. Dòng mã này không được thực hiện mà được hiển thị dưới dạng văn bản bình thường

– Cú pháp:

**<CODE>**

**Nội dung văn bản muốn định dạng**

**</CODE>**



Ví dụ:

```
<CODE>
 If (x > 0)

 x = x + 1

 else

 y = y + 1
</CODE>
```

**<EM>:** Văn bản được nhấn mạnh (giống tag <I>)

– **Cú pháp:**

**<EM>Văn bản được nhấn mạnh</EM>**

**<STRONG>:** Định dạng chữ đậm (giống <B>)

– **Cú pháp:**

**<STRONG>Văn bản được nhấn mạnh</STRONG>**

**<BLOCKQUOTE>:**

– Dùng phân cách một khối văn bản để nhấn mạnh, đoạn văn bản này được tách ra thành một paragraph riêng, thêm khoảng trắng trên và dưới đoạn đồng thời thụt vào so với lề trái (tương đương chức năng của phím tab)

– **Cú pháp:**

```
<BLOCKQUOTE>
 Nội dung khối văn bản nhấn mạnh
</BLOCKQUOTE>
```

Ví dụ:

```
<HTML>
 <HEAD><TITLE>Learning HTML</TITLE><HEAD>
 <BODY>
 <BLOCKQUOTE>
 Humpty Dumpty sat on a wall
 Humpty Dumpty had a great fall
 All the King's horses
 And all the King's men
 Could not put Humpty Dumty together again

 </BLOCKQUOTE>
</BODY>
</HTML>
```

**<PRE>:**

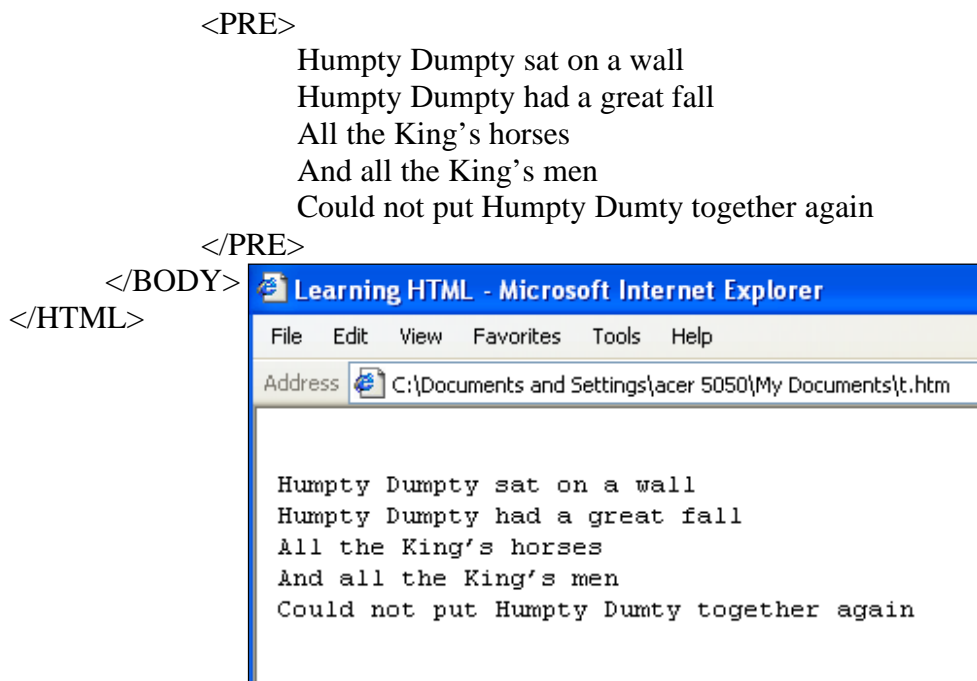
– Giữ nguyên các định dạng như: ngắt dòng, khoảng cách, thích hợp với việc tạo bảng

– **Cú pháp:**

```
<PRE>
 Nội dung văn bản cần định dạng trước với tất cả định dạng khoảng
 cách, xuống dòng và ngắt hàng
</PRE>
```

Ví dụ:

```
<HTML>
 <HEAD><TITLE>Learning HTML</TITLE><HEAD>
 <BODY>
```



### <DIV> <SPAN>:

- Chia văn bản thành các khối, có chung một định dạng
  - <DIV> chia văn bản thành một khối bắt đầu từ một dòng mới.
  - <SPAN> tách khối nhưng không bắt đầu từ một dòng mới
- **Cú pháp:**
  - <DIV>Nội dung của khối bắt đầu từ một dòng mới </DIV>
  - <SPAN>Nội dung của khối trong 1 dòng </SPAN>

Ví dụ:

```
<HTML>
<HEAD><TITLE> Learning HTML</TITLE></HEAD>
<BODY>
 <DIV>Division 1
 <P> The DIV element is used to group elements.
 <P>Typically, DIV is used for block level elements
 </DIV>
 <DIV align = right>
 Division 2
 <P>This is a second division

 <H2>Are you having fun?</H2>

 </DIV>
 <P> The second division is right aligned.
 Common
 formatting
 is applied to all the elements in the division
</BODY>
</HTML>
```

### Các ký tự đặc biệt:

- a. Lớn hơn (>): &gt;
- Ví dụ:
- ```
<CODE>
```

```
If A > B  
Then <BR>  
A = A + 1
```

</CODE>

- b. Nhỏ hơn (<): <

Ví dụ:

```
<CODE>
```

```
If A < B  
Then <BR>  
A = A + 1
```

</CODE>

- c. Cặp nháy"": "

Ví dụ:

```
<BODY>
```

" To be or not to be? " That is the question

```
</BODY>
```

- d. Ký tự và &: &

Ví dụ:

```
<P> William & Graham went to the fair
```

- e. Ký tự khoảng trắng:

CHƯƠNG 2. SIÊU LIÊN KẾT-HÌNH ẢNH

2.1 GIỚI THIỆU SIÊU LIÊN KẾT

2.1.1 Siêu liên kết

Khả năng chính của HTML là hỗ trợ các siêu liên kết. Một siêu liên kết cho phép người truy cập có thể đi từ trang web này đến trang web khác. Một liên kết gồm 3 phần:

- *Nguồn*: chứa nội dung hiển thị khi người dùng truy cập đến, có thể là một trang web khác, một đoạn film, một hình ảnh hoặc một hộp thoại để gửi mail...
- *Nhãn*: có thể là dòng văn bản hoặc hình ảnh để người dùng click vào khi muốn truy cập đến liên kết, nếu nhãn là văn bản thì thường được gạch dưới
- *Đích đến* (target): xác định vị trí để nguồn hiển thị.

2.1.2 Các loại liên kết

- *Internal Hyperlink*: (Liên kết trong) là các liên kết với các phần trong cùng một tài liệu hoặc liên kết các trang trong cùng một web site.
- *External Hyperlink* (Liên kết ngoài) là các liên kết với các trang trên web site khác.

2.2 TẠO SIÊU LIÊN KẾT

Cú pháp:

** Nhãn **

- Dùng URL tương đối để liên kết đến các trang trong cùng một website

Ví dụ:

```
<HTML>
```

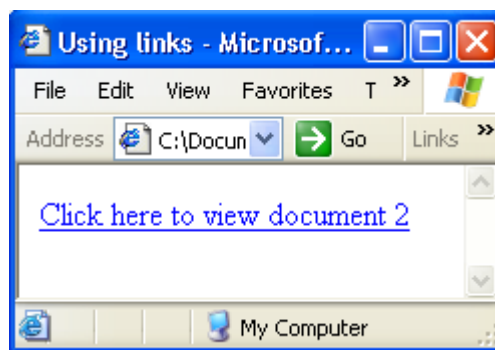
```
<HEAD> <TITLE> Using links</TITLE></HEAD>
```

```
<BODY>
```

```
<A HREF = "Doc2.htm">Click here to view document 2</A>
```

```
</BODY>
```

```
</HTML>
```



- Dùng URL tuyệt đối để liên kết đến các trang trong website khác

Ví dụ:

```
<A href="http://www.google.com"> liên kết đến Google</A>
```

Liên kết với các phần trong cùng một trang web

- Nếu nội dung của trang quá dài thì nên tạo các Bookmark để nhảy đến một phần cụ thể nào đó trên chính trang web hiện hành.
- Cách tạo liên kết đến các phần trong cùng trang: gồm 2 bước
- ✧ Tạo Bookmark:

** Nhãn Nội dung**

- ✧ Tạo liên kết đến Bookmark:

Nhãn của text liên kết

Ví dụ :

<HTML>

<HEAD><TITLE> Using htm links</TITLE> </HEAD>

<BODY>

Internet

Introduction to HTML

Internet

Internet là một mạng của các mạng. Nghĩa là, các mạng máy tính được liên kết với các mạng khác, nối các nước và ngày nay là toàn cầu. Giao thức truyền thông là TCP/IP cung cấp liên kết với tất cả các máy tính trên thế giới

Introduction to HTML

Ngôn ngữ đánh dấu siêu văn bản là ngôn ngữ chuẩn mà web sử dụng để tạo và nhận ra tài liệu. Mặc dù không phải là một tập con của ngôn ngữ nâng cấp tiêu chuẩn tổng quát (SGML), ngôn ngữ đánh dấu siêu văn bản cũng có liên quan với SGML. SGML là một phương pháp trình bày các ngôn ngữ định dạng tài liệu. HTML là ngôn ngữ đánh dấu được sử dụng để tạo tài liệu HTML. Các hướng dẫn chỉ rõ một trang web nên được hiển thị như thế nào trong trình duyệt

</BODY>

</HTML>→ Kết quả trên trình duyệt



Liên kết với một Bookmark ở một tài liệu khác

Cú pháp:

Ví dụ:

– Trang main.htm

<HTML>

<HEAD><TITLE> Main document</TITLE></HEAD>

```

<BODY>
    <A HREF = "C:\Doc1.htm#Internet">Internet</A><br>
    <A HREF = "C:\Doc1.htm#HTML">Introduction to
HTML</A><br>
</BODY>
</HTML>
- Trang Doc1.htm
<HTML>
<HEAD><TITLE>Using Links</TITLE></HEAD>
<BODY>
    <A name = "Internet">Internet</A><BR>
        Internet là một mạng của các mạng. Nghĩa là, mạng máy tính
        được liên kết với các mạng khác, nối với các nước và ngày nay
        là toàn cầu. Giao thức truyền TCP/IP cung cấp liên kết với tất
        cả các máy tính trên thế giới.
    <A name = "HTML">Introduction to HTML</A><BR>
        Ngôn ngữ đánh dấu siêu văn bản là ngôn ngữ chuẩn mà web sử
        dụng để tạo và nhận ra tài liệu. Mặc dù không phải là một tập
        con của ngôn ngữ nâng cấp tiêu chuẩn tổng quát (SGML), ngôn
        ngữ đánh dấu siêu văn bản cũng có liên quan với SGML.
        SGML là một phương pháp trình bày các ngôn ngữ định dạng
        tài liệu. HTML là ngôn ngữ đánh dấu được sử dụng để tạo tài
        liệu HTML.
</BODY>
</HTML>

```

Liên kết đến hộp thư e-mail

Cú pháp:

```
<A href="mailto:địa chỉ Email">Nhãn</A>
```

- Nếu siêu liên kết đặt ở cuối trang thì dùng tag <ADDRESS>

Cú pháp:

```
<Address><A href="mailto:địa chỉ Email">Nhãn</A></Address>
```

2.3 HÌNH ẢNH TRÊN TRANG WEB

2.3.1 Các loại ảnh

- Ảnh .Gif** (Graphics Interchange Format): được sử dụng phổ biến nhất trong các tài liệu HTML, dễ chuyển tải, ngay cả các kết nối sử dụng MODEM tốc độ chậm, hỗ trợ 256 màu GIF. Các file GIF được định dạng không phụ thuộc phần nền
- Ảnh JPEG** (Joint PhotoGraphic Expert Group) có phần mở rộng .JPG, là loại ảnh nén mất thông tin, nghĩa là ảnh sau khi bị nén không giống như ảnh gốc. Tuy nhiên, trong quá trình phát lại thì ảnh cũng tốt gần như ảnh gốc. JPEG hỗ trợ hơn 16 triệu màu và thường được sử dụng cho các ảnh có màu thực.
- Ảnh PNG** (Portable Network Graphics) nén không mất dữ liệu

2.3.2 Chèn hình ảnh

Cú pháp:

```
<IMG Src=URL Border=n Alt="Nội dung thay thế">
```

URL: địa chỉ của tập tin hình ảnh, thường sử dụng địa chỉ tương đối, ví dụ:

```

```

không phụ thuộc vị trí của tập tin ảnh trên đĩa

n: độ dày của đường viền, tính bằng pixel

Alt: Nội dung thay thế sẽ hiển thị khi hình không load được, hoặc khi đưa chuột ngang qua hình

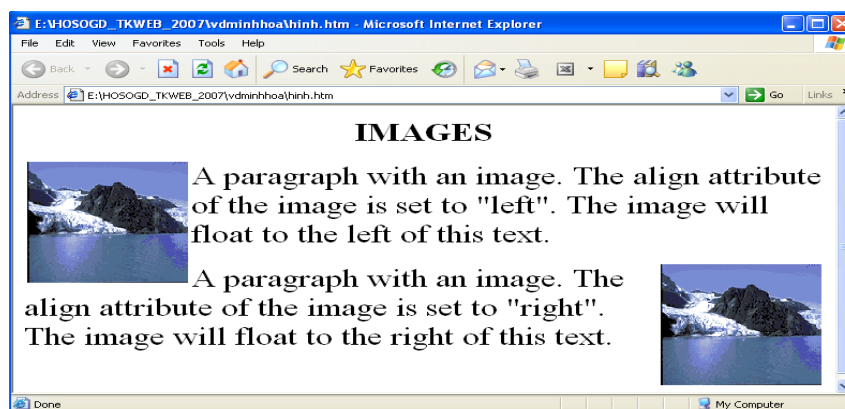
2.3.3 Các thuộc tính của ảnh

a) Dàn văn bản quanh hình ảnh:

** Nội dung văn bản quanh hình ảnh**

** Nội dung văn bản quanh hình ảnh**

Ví dụ:



b) Kích thước ảnh:

Ví dụ:

```
<html>
  <head><title>Image</title></head>
  <body>
    
  </body>
</html>
```

c) Chặn văn bản bao quanh hình:

Canh lề khi dàn văn bản xung quanh một ảnh sẽ tác động đến tất cả các văn bản sau đó nếu không chèn vào một dòng kẻ đặc biệt. Thuộc tính CLEAR trong tag BR làm cho văn bản không bắt đầu nếu lề cụ thể không bị xóa đi (nghĩa là tại cạnh dưới của ảnh)

Cú pháp:

<BR CLEAR=Right> : Ngăn chặn văn bản dàn bên lề phải của ảnh

<BR CLEAR=Left> : Ngăn chặn văn bản dàn bên lề trái của ảnh

<BR CLEAR=All> : Ngăn chặn văn bản dàn hai bên lề của ảnh

d) Thêm khoảng trống xung quanh ảnh

Nếu không muốn văn bản dàn xung quanh lề trái của ảnh thì ta có thể thêm khoảng trắng xung quanh ảnh

Cú pháp:

HSPACE=n: Khoảng trắng được tính bằng pixel sẽ thêm vào cả bên phải và bên trái của ảnh

VSPACE=m: Khoảng trắng được tính bằng pixel sẽ thêm vào cả bên trên và bên dưới của ảnh

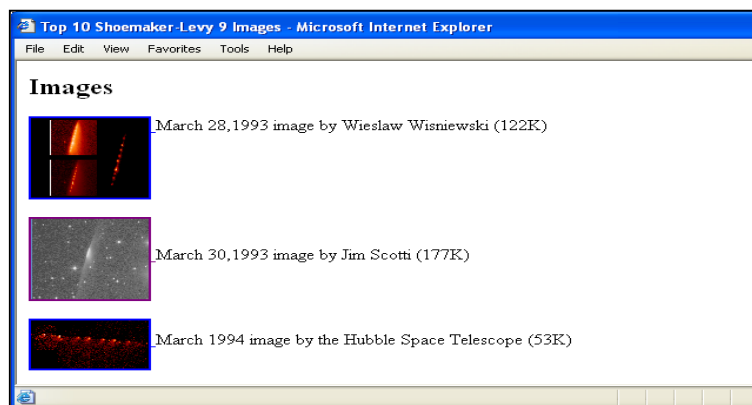
- e) **Canh lề cho ảnh:** Có thể canh lề cho ảnh so với một dòng văn bản trong một đoạn

Cú pháp:

****Văn bản muốn canh lề so với ảnh

Direction: gồm các giá trị: top, bottom, middle, texttop

Ví dụ:



2.3.4 Dùng ảnh làm liên kết

Có thể dùng hình ảnh để tạo một liên kết đến một trang khác, hoặc nếu có một ảnh lớn, bạn có thể tạo ảnh nhỏ hơn hoặc một biểu tượng cho nó để nó có thể hiển thị nhanh chóng trên trang web, sau đó tạo liên kết để đưa người truy cập đến ảnh có kích thước thật

Cú pháp:

Nhãn

2.3.5 Bản đồ ảnh

Bản đồ ảnh là một ảnh trong trang web được chia ra làm nhiều vùng, mỗi vùng khi click vào sẽ liên kết đến một địa chỉ URL

Cách tạo: Trước hết phải chèn vào trang một ảnh và đặt nhãn cho ảnh

<Map Name="Label">

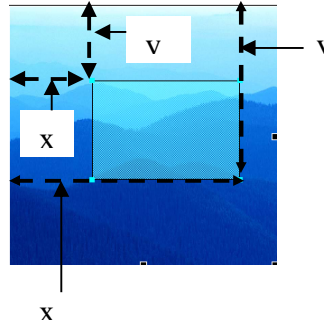
<Area Shape= "type" coords="x1,y1,x2,y2, ..." href="URL">

</Map>

Trong đó:

- Label: tên của bản đồ ảnh
- Type: hình dạng của các vùng trên ảnh, gồm các loại:
 - Rect: Vùng hình chữ nhật
 - Circle: Vùng hình tròn
 - Poly: Vùng hình đa giác

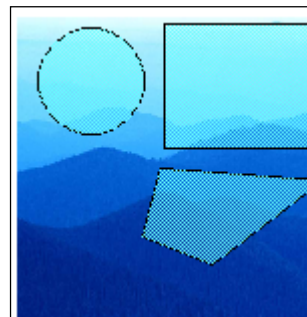
- Coords: toạ độ các đỉnh của hình
 - Rect: (x1, y1, x2, y2) là toạ độ 2 đỉnh chéo của vùng hình CN
 - Circle: (x, y, r) lần lượt là toạ độ tâm và bán kính của vùng hình tròn
 - Poly: (x1, y1, x2, y2, x3, y3, ...) là các đỉnh của vùng hình đa giác



Ví dụ:

```
<html>
<head>
<title>Image</title>
</head>
<body>

<map name="Map1">
<area shape="rect" coords="73,3,149,66" href="B1.htm">
<area shape="poly" coords="152,81,71,75,62,109,97,123"
href="B3.htm">
<area shape="circle" coords="37,32,27" href="b2.htm">
</map>
</body>
</html>
```



2.3.6 Hình nền

Trong hầu hết các trang web thường sử dụng nền màu, với mục đích là làm nổi bật nội dung trang đó. Tuy nhiên cũng có thể sử dụng hình ảnh để làm nền bằng thuộc tính BACKGROUND của thẻ BODY.

<BODY BACKGROUND= "bgimage.gif">

CHƯƠNG 3. DANH SÁCH

3.1 DANH SÁCH KHÔNG CÓ THỨ TỰ

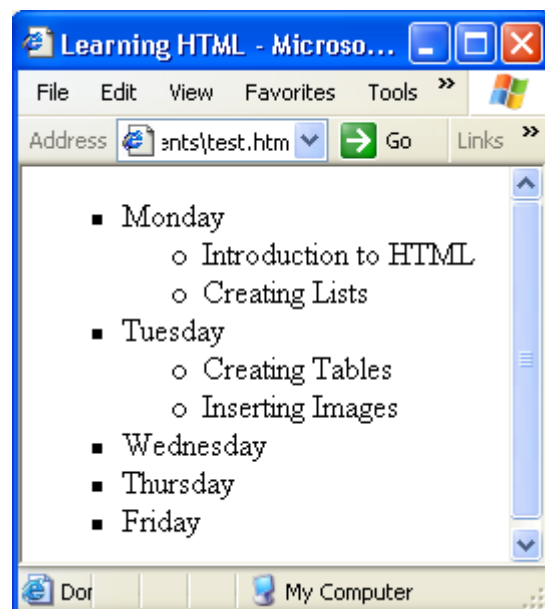
Cú pháp:

```
<UL Type= Shape1>
  <LI Type= Shape 2> Nội dung 1
  <LI Type= Shape 2> Nội dung 2
  ...
</UL>
```

- Shape 1, Shape 2 là loại bullet tự động đặt ở đầu dòng trong danh sách
- Shape 1: ảnh hưởng đến toàn danh sách
- Shape 2: ảnh hưởng đến một mục trong danh sách
- Các loại shape:
 - Circle: Bullet tròn, rỗng
 - Square: Bullet vuông
 - Disc: Bullet tròn không rỗng

Ví dụ:

```
<HTML>
<HEAD><TITLE>Learning HTML</TITLE>
<BODY>
  <UL type="Square">
    <LI>Monday
      <UL>
        <LI>Introduction to HTML
        <LI>Creating Lists
      </UL>
    <LI>Tuesday
      <UL>
        <LI>Creating Tables
        <LI>Inserting Images
      </UL>
    <LI>Wednesday
    <LI>Thursday
    <LI>Friday
  </UL>
</BODY>
</HTML>
```



3.2 DANH SÁCH CÓ THỨ TỰ

Cú pháp:

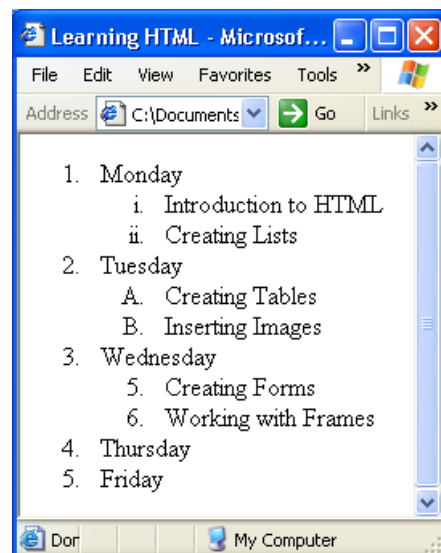
```
<OL Type=x Start =n >
  <LI Type =x1 Value=m> Nội dung 1
  <LI Type =x1 Value=m> Nội dung 2
  ...
</OL>
```

- x: loại ký tự muốn sử dụng trong danh sách gồm :
 - ✧ A: Chữ hoa
 - ✧ a: Chữ thường

- ✧ I: Số la mã hoa
- ✧ i: Số la mã thường
- ✧ 1: Cho số mặc định
- **n**: giá trị đầu tiên của danh sách
- **x1**: là loại ký tự sử dụng cho dòng này và dòng tiếp theo, làm mất ảnh hưởng của x
- **m**: giá trị đầu tiên của dòng này, làm thay đổi giá trị của n

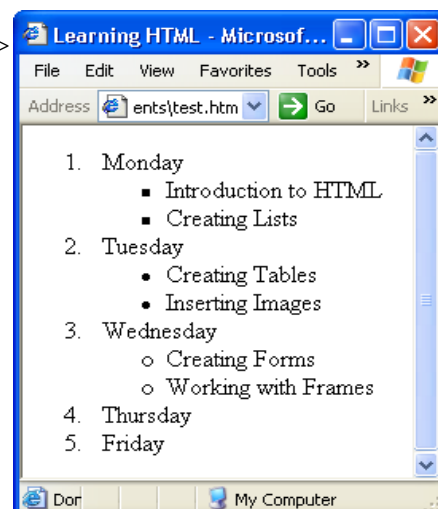
Ví dụ 1:

```
<HTML>
<HEAD><TITLE>Learning HTML</TITLE>
<BODY>
  <OL>
    <LI>Monday
      <OL>
        <LI TYPE = i>Introduction to HTML
        <LI TYPE = i>Creating Lists
      </OL>
    <LI>Tuesday
      <OL>
        <LI TYPE = A>Creating Tables
        <LI TYPE = A>Inserting Images
      </OL>
    <LI>Wednesday
      <OL START = 5>
        <LI >Creating Forms
        <LI >Working with Frames
      </OL>
    <LI>Thursday
    <LI>Friday
  </OL>
</BODY>
</HTML>
```



Ví dụ 2: Có thể lồng 2 loại danh sách có thứ tự và không có thứ tự vào nhau

```
<HTML>
<HEAD><TITLE>Learning HTML</TITLE></HEAD>
<BODY>
  <OL>
    <LI>Monday
      <UL>
        <LI >Introduction to HTML
        <LI >Creating Lists
      </UL>
    <LI>Tuesday
      <UL type='Disc'>
        <LI >Creating Tables
        <LI >Inserting Images
      </UL>
    <LI>Wednesday
      <UL type='cycle'>
        <LI >Creating Forms
        <LI >Working with Frames
      </UL>
  </OL>
```



```

        </UL>
    <LI>Thursday
    <LI>Friday
</OL>
</BODY>
</HTML>

```

3.3 DANH SÁCH ĐỊNH NGHĨA

Trong HTML có một tag đặc biệt dùng để tạo danh sách định nghĩa dành riêng cho việc tra cứu, nhưng cũng thích hợp cho danh sách nào để nối một từ với một diễn giải dài.

Cú pháp:

```

<DL>
    <DT>Nhập từ muốn định nghĩa
    <DD>Nhập nội dung định nghĩa
    ...
</DL>

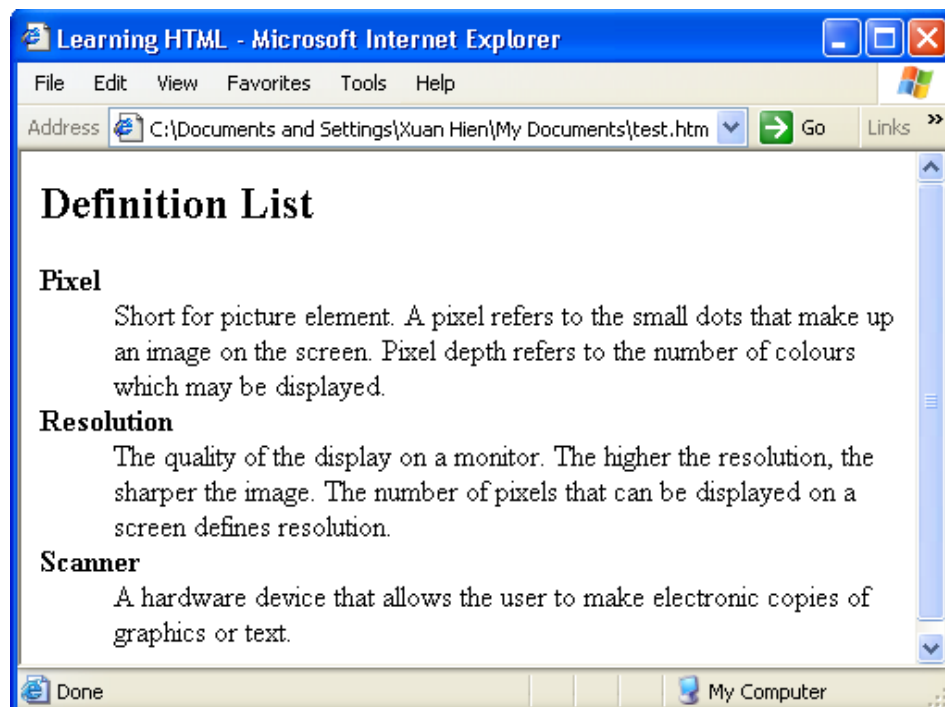
```

Ví dụ:

```

<HTML>
  <HEAD>
    <TITLE>Learning HTML</TITLE>
  </HEAD>
  <BODY>
    <h2> Definition List</h2>
    <DL>
      <DT>Pixel
      <DD> Short for picture element. A pixel refers to the small dots
        that make up an image on the screen. Pixel depth refers to the
        number of colours which may be displayed.
      <DT>Resolution
      <DD>The quality of the display on a monitor. The higher the
        resolution, the sharper the image. The number of pixels that can
        be displayed on a screen defines resolution.
      <DT>Scanner
      <DD> A hardware device that allows the user to make electronic
        copies of graphics or text.
    </DL>
  </BODY>
</HTML>

```



CHƯƠNG 4. BẢNG VÀ TRÌNH BÀY TRANG

4.1 BẢNG

Bảng thường được sử dụng để tạo các văn bản nhiều cột hoặc phân chia trang thành nhiều vùng khác nhau rất tiện lợi trong thiết kế và trình bày trang web

Cú pháp:

```

<TABLE >
  {
    <TR>
      {
        <TD>Nội dung trong ô 1</TD>  ← Cột 1
        <TD>Nội dung trong ô 2</TD>  ← Cột 2
        ...
        <TD>Nội dung trong ô n</TD>
      }
    </TR>
    {
      <TR>
        <TD>Nội dung trong ô 1</TD>
        <TD>Nội dung trong ô 2</TD>
        ...
        <TD>Nội dung trong ô n</TD>
      }
    </TR>
    ...
  }
</TABLE>
  
```

- Tag <table> </table> : chỉ thị một bảng
- Tag <tr>.....</tr> : xác định một dòng của bảng
- Tag <td>.....</td>: xác định một ô chứa dữ liệu của bảng. Dữ liệu trong ô có thể là văn bản hoặc hình ảnh...

Ví dụ 1:

```

<HTML>
  <HEAD>
    <TITLE>TABLE</TITLE>
  </HEAD>
  <BODY >
    <table border="1">
      <TR>
        <TD>Cell 1</TD>
        <TD>Cell 2</TD>
        <TD>Cell 3</TD>
        <TD>Cell 4</TD>
      </TR>
    </table>
  </BODY>
</HTML>
  
```

→

Cell 1	Cell 2	Cell 3	Cell 4
--------	--------	--------	--------

Ví dụ 2:

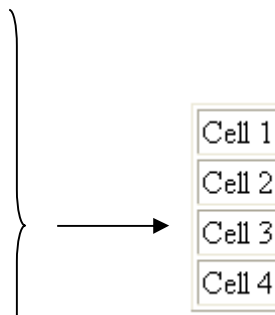
```

<HTML>
  
```

```

<HEAD>
  <TITLE>TABLE</TITLE>
</HEAD>
<BODY >
  <table border="1">
    <TR>
      <TD>Cell 1</TD>
    </TR>
    <TR>
      <TD>Cell 2</TD>
    </TR>
    <TR>
      <TD>Cell 3</TD>
    </TR>
    <TR>
      <TD>Cell 4</TD>
    </TR>
  </table>
</BODY>
</HTML>

```



Ví dụ 3:

```

<HTML>
  <HEAD>
    <TITLE>TABLE</TITLE>
  </HEAD>
  <BODY >
    <table border="1">
      <TR>
        <TD>Cell 1</TD>
        <TD>Cell 2</TD>
      </TR>
      <TR>
        <TD>Cell 3</TD>
        <TD>Cell 4</TD>
      </TR>
    </table>
  </BODY>
</HTML>

```

Cell 1	Cell 2
Cell 3	Cell 4

4.2 CÁC THUỘC TÍNH

a) Thêm khung viền:

<Table Border =n>...</Table>

n: độ dày của khung viền tính bằng Pixel

b) Định màu của khung viền và màu nền:

<Table BorderColor= "Color" BgColor="Color">...</Table>

c) Tạo bóng :

<Table BorderColorDark= "Color"> : Bóng đổ ở cạnh dưới và phải của bảng

<Table BorderColorLight= "Color"> : Bóng đổ cạnh trên trái của bảng

d) Định chiều rộng và chiều cao của bảng:

<Table Width =n height=m>, n là chiều rộng tính bằng Pixel

e) Canh lề bảng:

<Table Align= left/ right/ center>...</table>

f) Thuộc tính Cellpadding và CellSpacing:

<Table CellSpacing ="value">: Khoảng cách giữa đường viền của các ô

<Table CellPadding="Value">: Khoảng cách giữa đường viền của ô với văn bản

g) Tag tiêu đề của Table:

<Caption> tiêu đề </Caption>

- Tag <Caption> nằm trong cặp Tag <Table>...</Table>

4.3 THUỘC TÍNH CỦA CỘT

a) Canh lề theo chiều ngang:

<Td Align=left/ right/center>...</Td>

b) Canh lề theo chiều đứng:

<Td Valign= Top/ Bottom/ Middle>...</Td>

c) Trộn ô:

<Td Colspan=n>: trộn n cột

<Td RowSpan=n>: trộn n dòng

d) Tag <th>:

Có tác dụng như <td> nhưng làm cho dữ liệu trong ô được in đậm và canh giữa

```
<tr>
  <th> Nội dung </th>
</tr>
```

Ví dụ:

```
<TABLE border=2>
  <tr>
    <th> Cell 1 </th>
  </tr>
  <tr>
    <th> Cell 2 </th>
  </tr>
</TABLE>
```

Cell 1
Cell 2

Ví dụ:

<HTML>

<HEAD>


```

<TITLE>TABLE</TITLE>
</HEAD>
<BODY >
  <table border="5" CellSpacing =10 BorderColorDark=red width=50%>
    <TR>
      <TD>Cell 1</TD>
      <TD>Cell 2</TD>
    </TR>
    <TR>
      <TD>Cell 3</TD>
      <TD>Cell 4</TD>
    </TR>
  </table>
</BODY>
</HTML>

```

Cell 1	Cell 2
Cell 3	Cell 4

Ví dụ:

```

<Table border="1" bgcolor= "fuchsia" bordercolor="red" align="center"
Width=50% Height=30%>

```

```

  <caption> Properties of Table</caption>
  <tr>
    <th colspan="3"> Colspan</th>
  </tr>
  <tr>
    <th Rowspan="2"> Rowspan</th>
    <td align=center>Cell 1</td>
    <td align=center>Cell 2</td>
  </tr>
  <tr>
    <td align=center> Cell 3</td>
    <td align=center> Cell 4</td>
  </tr>
</table>

```

Properties of Table

Colspan		
Rowspan	Cell 1	Cell 2
	Cell 3	Cell 4

Ví dụ: Thiết kế một trang web như mẫu

Computer Model	
Tin tức	
Giai tri	
Quang cao	
The thao	

```

<html>
<head>
  <title> Trình bay trang</title>

```

```

</head>
<body>
<Table width="68%" height="135" border="1" cellspacing="0"
bordercolor="#990033">
  <tr>
    <th colspan="2" bgcolor="#FFCCFF">
      <div align="center">Computer Model </div>
    </th>
  </tr>
  <tr>
    <td width="24%" height="98" valign="top">
      <table width="100%" border="1" cellspacing="0">
        <tr>
          <td>Tin tức</td>
        </tr>
        <tr>
          <td>Giải trí</td>
        </tr>
        <tr>
          <td>Quang cáo</td>
        </tr>
        <tr>
          <td height="23">Thao</td>
        </tr>
      </table>
    </td>
    <td width="76%" align="center">
      
    </td>
  </tr>
</table>
</body>
</html>

```

4.4 TRÌNH BÀY TRANG

Trong thực tế, bảng thường được sử dụng để trình bày bố cục cho toàn bộ trang web. Nếu muốn thiết kế một trang thể hiện văn bản trong cột dạng báo chí hoặc phân trang thành những vùng có chủ đề khác nhau, thì bảng là một công cụ cần thiết. Một trong những nét đặc trưng hữu dụng của bảng đó là trong mỗi table cell bạn có thể sử dụng bất kỳ tag HTML nào, ví dụ bạn có thể chèn một tag <H1> trong một cell hoặc một danh sách có thứ tự các mục hoặc có thể chèn một bảng con trong một bảng khác...

Ví dụ :

Cần thiết kế trang web gồm nhiều vùng với những chủ đề khác nhau như hình dưới đây, thì bảng là công cụ hữu hiệu



Bước 1: Tạo một table thứ nhất gồm 1 dòng và 2 cột

```
<table>
  <tr>
    <td>
      <!--Danh sách các mục liên kết-->
    </td>
    <td>
      <!--Table 2 -->
    </td>
  </tr>
</table>
```

Danh sách các mục liên kết	table 2
----------------------------------	---------

Bước 2: tạo table thứ 2 gồm 3 dòng và 2 cột

```
<table>
  <tr>
    <td colspan =2>
      <!--Chèn hình logo-->
    </td>
  </tr>
  <tr>
    <td rowspan =2>
      <!--Nội dung 1 -->
    </td>
    <td>
      <!--Nội dung 2 -->
    </td>
  </tr>
  <tr>
    <td>
      <!--Nội dung 3 -->
    </td>
  </tr>
</table>
```

danh sách các mục liên kết ~~~~~ ~~~~~ ~~~~~ ~~~~~	Logo	
	Nội dung 1 ~~~~~ ~~~~~ ~~~~~ ~~~~~ ~~~~~	Nội dung 2 ○ ~~~~~ ○ ~~~~~ ○ ~~~~~ ○ ~~~~~
	Nội dung 3 ~~~~~ ~~~~~ ~~~~~	

CHƯƠNG 5. FORM

5.1 GIỚI THIỆU FORM

Form được sử dụng khi cần:

- Thu thập thông tin tên, địa chỉ, số điện thoại, email, ... để đăng ký cho người dùng vào một dịch vụ hoặc một sự kiện
- Tập hợp thông tin để mua hàng
- Thu thập thông tin phản hồi về một Website
- Cung cấp công cụ tìm kiếm trên website

Cách tạo:

Cú pháp:

<Form Method=(Post Get) Action=script.url>

Nội dung của Form

</Form>

Trong đó:

-*Method*: xác định phương thức đưa dữ liệu lên máy chủ, có 2 giá trị :Post và Get

- Nếu giá trị là GET thì trình duyệt sẽ tạo một câu hỏi chứa trang URL, một dấu hỏi và các giá trị do biểu mẫu tạo ra. Trình duyệt sẽ đổi script của câu hỏi thành kiểu được xác định trong URL để xử lý.
- Nếu giá trị là POST thì dữ liệu trên biểu mẫu sẽ được gửi đến script như một khối dữ liệu

-*Action*: là địa chỉ của script sẽ thực hiện khi form được submit

5.2 CÁC PHẦN TỬ CỦA FORM

Các phần tử của form thường sử dụng trên trang web gồm

- **Input boxes**: nhập dữ liệu dạng text và number
- **Radio buttons**: dùng để chọn một tùy chọn trong danh sách
- **Selection lists**: dùng cho một danh sách dài các lựa chọn, thường là trong Drop-down list box
- **Check boxes**: chỉ định một item được chọn hay không
- **Text area**: một text box có thể chứa nhiều dòng
- **Submit và Reset button**: để gửi form đến CGI script vừa để reset form về trạng thái ban đầu

5.2.1 Input boxes

Là một hộp dòng đơn dùng để nhập văn bản hoặc số. Để tạo các input boxes, sử dụng tag <INPUT>, tag <INPUT> còn được sử dụng cho nhiều loại field khác trên form.

Cú pháp:

<FORM>

<INPUT TYPE=Object NAME=Text>

</FORM>

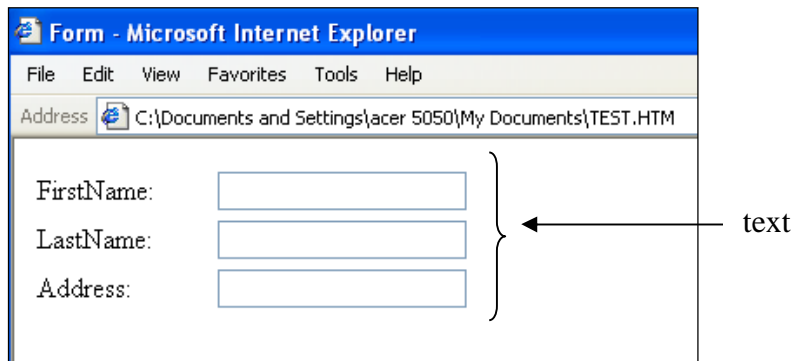
Các giá trị của thuộc tính TYPE: Mặc định giá trị của TYPE là text, nếu trong tag <INPUT> không nhập thuộc tính TYPE thì loại input boxes là text

- TEXT
- PASSWORD
- CHECKBOX

- RADIO
- HIDDEN
- RESET
- SUBMIT
- TEXTAREA
- BUTTON
- IMAGE

Ví dụ:

```
<html>
<head><title>Form</title></head>
<body>
  <form>
    <table>
      <tr>
        <td width=100>FirstName: </td>
        <td><input name =Firstname></td>
      </tr>
      <tr>
        <td>LastName: </td>
        <td><input name =Lastname></td>
      </tr>
      <tr>
        <td>Address: </td>
        <td><input name =Address></td>
      </tr>
    </table>
  </form>
</body>
</html>
```



1. **Text box**: Hộp văn bản, do người sử dụng nhập vào

Cú pháp:

<Input Type="Text" Value="Value" Name="name" Size=n Maxlength=m>

- Name : tên dữ liệu đầu vào server
- Value: Dữ liệu ban đầu có sẵn trong text box
- Size: chiều rộng của text box tính bằng số ký tự (mặc định là 20)
- Maxlength: số ký tự tối đa có thể nhập vào text box

2. **Tạo hộp Password**: Những ký tự nhập vào hiển thị dưới dạng dấu chấm , thông tin sẽ không bị mã hoá khi gửi lên server

Cú pháp:

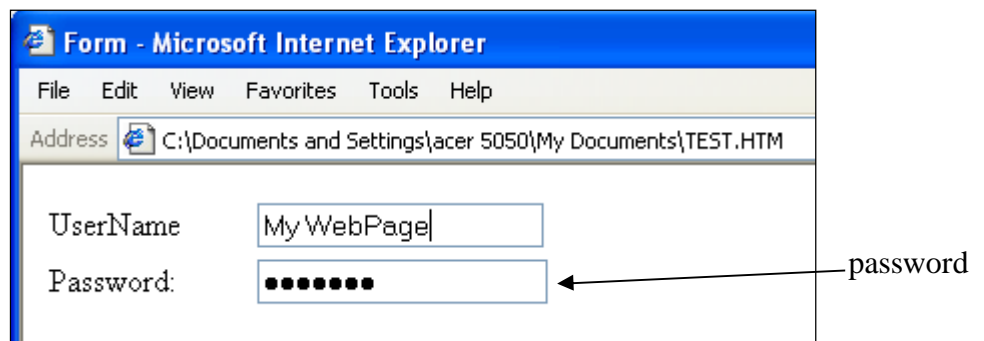
<Input Type="password" Name="name" size=n maxlength=n>

- Size: chiều rộng của hộp Password, tính bằng ký tự

- Maxlength: Số ký tự tối đa có thể nhập vào hộp Password

Ví dụ:

```
<html>
  <head><title>Form</title></head>
  <body>
    <form><table>
      <tr>
        <td width=100>UserName </td>
        <td><input name =UserName></td>
      </tr>
      <tr>
        <td>Password: </td>
        <td><input Type='password' name =Password></td>
      </tr>
    </table></form>
  </body>
</html>
```



3. **Checkbox:** Hộp chọn, người xem có thể đánh dấu nhiều checkbox trong cùng 1 bộ

Cú pháp:

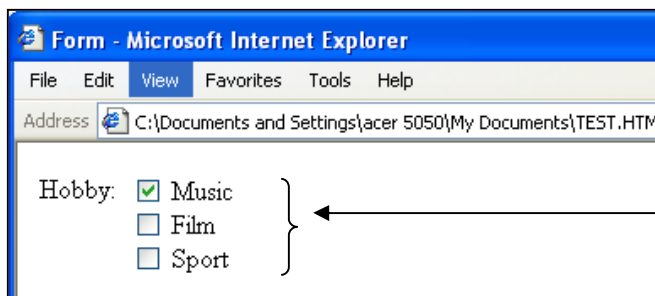
<Input Type="Checkbox" Name="Name" Value="Value" Checked> Nhãn

- Name: tên của checkbox
- Value: xác định mỗi giá trị cho mỗi hộp checkbox được gửi cho server khi người xem đánh dấu vào checkbox
- Checked: thuộc tính để hộp check box được chọn mặc định

Ví dụ:

```
<html>
  <head><title>Form</title></head>
  <body>
    <form><table>
      <tr>
        <td width=50 valign=top>Hobby: </td>
        <td>
          <Input Type='Checkbox' Name='st' Value='nhac' Checked> Music<br>
          <Input Type='Checkbox' Name='st' Value='film'> Film<br>
          <Input Type='Checkbox' Name='st' Value='thethao' > Sport
        </td>
      </tr>
    </table></form>
```

</body>
</html>



4. **Radio button:** Cho phép người xem chỉ chọn một tùy chọn tại mỗi thời điểm

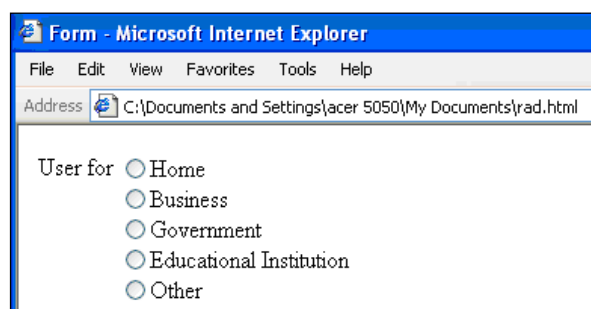
Cú pháp:

<input type="radio" name="name" value="Value" checked>Nhãn

- Name: tên của radio, kết nối các radio button với nhau
- Value: Những dữ liệu sẽ gửi đến server khi radio button được chọn
- Checked: thuộc tính để radio button được chọn mặc định

Ví dụ:

```
<html>
<head><title>Form</title></head>
<body>
  <form><table>
    <tr>
      <td valign=top>User for</td>
      <td>
        <input type='radio' name=use value=home>Home<br>
        <input type='radio' name=use value=bus>Business<br>
        <input type='radio' name=use value=gov>Government<br>
        <input type='radio' name=use value=ed>Educational Institution<br>
        <input type='radio' name=use value=other>Other<br>
      </td>
    </tr>
  </table></form>
</body>
</html>
```



5. **Submit Button:** Tất cả thông tin của người xem nhập vào sẽ được gửi đến server khi người xem click nút Submit

Cú pháp:

<Input Type="Submit" Value="Submit Message" Name="Name">

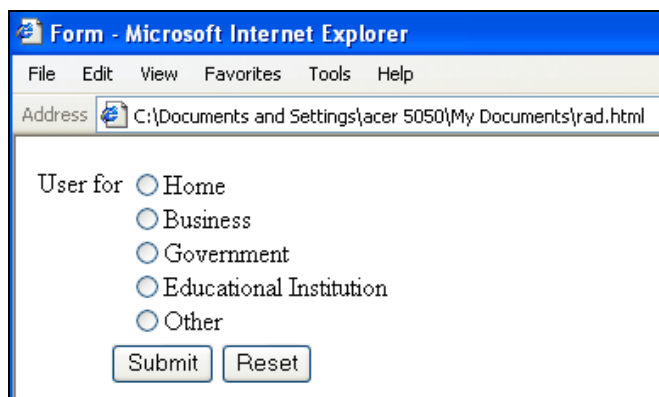
- Submit Message: Là chữ xuất hiện trên Button
- Name: tên của button

6. **Reset Button:** Thiết lập giá trị ban đầu của tất cả các điều khiển trên form

Cú pháp:

<Input Type="reset" Value="Reset Message" Name="Name">

Có thể tạo nút Reset và Submit bằng hình ảnh với cú pháp:
<Button Type="reset" Name="reset" Value="reset">
Nhãn chữ lẽ trái
<Image src="Image.gif">Nhãn chữ lẽ phải</Button>
Ví dụ:



7. **Button:** Nút dùng để thực hiện các lệnh do người sử dụng đưa ra
Cú pháp:

<input type="button" name="Button" value="Button">

8. **Hidden:** là các field mà người xem không nhìn thấy trên trình duyệt, nhưng vẫn là một phần tử trên form. Hidden field dùng để lưu trữ thông tin trong các form trước, các thông tin này cần đi kèm với các dữ liệu trong form hiện hành mà không muốn người xem nhập lại

Cú pháp:

<Input Type='hidden' Name='Name' Value='Value'>

Name: tên mô tả ngắn gọn thông tin cần lưu trữ

Value: Thông tin cần lưu trữ

5.2.2 Selection List

1. **Drop down menu:**

Cú pháp:

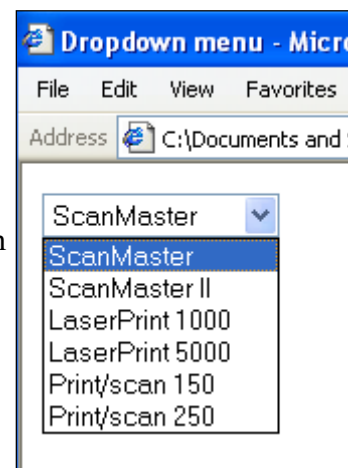
<Select Name="Name" Size=n Multiple>
<Option Value="Value" selected> Option 1
<Option Value="Value"> Option 2
 ...

</Select>

- Nhãn:Giới thiệu Menu
- Name: tên dữ liệu đầu vào server
- Size: là chiều cao của menu tính bằng hàng chữ
- Multiple: là thuộc tính cho phép chọn nhiều đề mục (listbox)
- Selected: đề mục được chọn mặc định
- Value: xác định dữ liệu gởi cho server nếu đề mục được chọn

Ví dụ:

```
<html>
  <head><title>Dropdown menu</title></head>
  <body>
    <form>
      <select Name=Product>
        <option value=1>ScanMaster
```



```

<option value=3>ScanMaster II
<option value=4>LaserPrint 1000
<option value=5> LaserPrint 5000
<option value=6>Print/scan 150
<option value=2> Print/scan 250

```

```
</Select>
```

```
</form>
```

```
</body>
```

```
</html>
```

2. Nếu thêm thuộc tính Multiple thì ta được dạng listbox

```
<html>
```

```
<head><title>Dropdown menu</title></head>
```

```
<body>
```

```
<form>
```

```
<select Name=Product size=5 Multiple>
```

```
<option value=1>ScanMaster
```

```
<option value=3>ScanMaster II
```

```
<option value=4>LaserPrint 1000
```

```
<option value=5> LaserPrint 5000
```

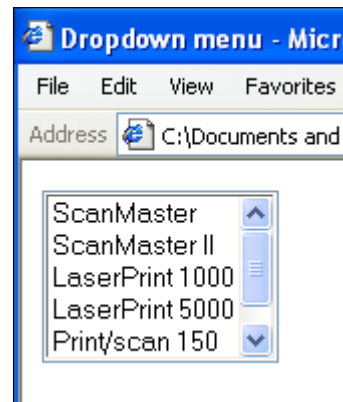
```
<option value=6>Print/scan 150
```

```
<option value=2> Print/scan 250
```

```
</Select>
```

```
</form>
```

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



3. Phần tử OPTGROUP: được sử dụng để nhóm các chọn lựa thành các nhóm riêng.

Ví dụ:

```
<HTML>
```

```
<HEAD><Title>Using the Option Group</Title></head>
```

```
<BODY>
```

```
<FORM action= 'http: // somesite.com / processform' method= 'post'>
```

```
<SELECT name= 'course'>
```

```
<OPTGROUP>
```

```
<OPTION value= "Internetintro">Introduction to the Internet
```

```
<OPTION value= "Introhtml">Introduction to HTML
```

```
<OPTION value= "Introweb">Introduction to the web page designing
```

```
</OPTGROUP>
```

```
<OPTGROUP>
```

```
<OPTION value= "vbintro">Visual Basic – An Introduction
```

```
<OPTION value= "vbdev">Visual Basic – Application Development
```

```
</OPTGROUP>
```

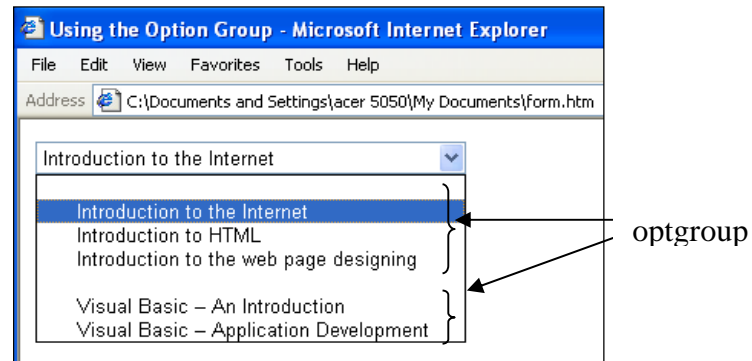
```
</SELECT>
```

```
</FORM></BODY></HTML>
```

5.2.3 TextArea

Cú pháp:

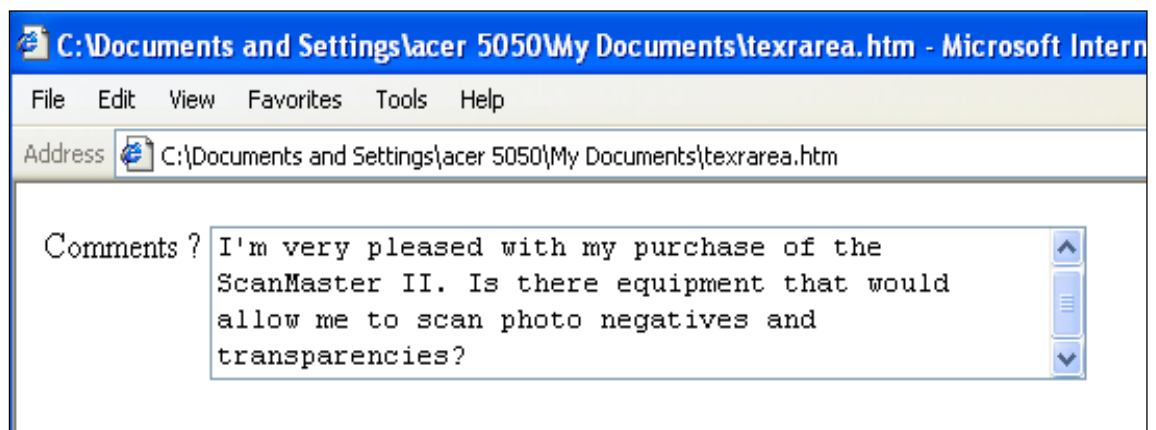
**<TextArea Name="name" Rows=n
Cols=m Wrap>Default text</textarea>**



- *Rows*: số dòng có thể nhập vào TextArea (mặc định là 4)
- *Cols*: độ rộng của textarea (tính bằng số ký tự, mặc định là 40)
- *Wrap*: các dòng chữ tự động tràn ra trong lề của vùng text area, Value: virtual, physical

Ví dụ:

```
<html>
<head><title>Textarea</title></head>
<body>
<table>
<tr>
<td valign=top> Comments ?</td>
<td><textarea rows=4 cols=50 name=comments
wrap=virtual></textarea>
</td>
</tr>
</table>
</body>
</html>
```



5.2.4 Nhãn

Cú pháp:

<Label For="idname"> Nội dung label</label>

Idname: là giá trị của thuộc tính ID trong thành phần Form tương ứng

Ví dụ:

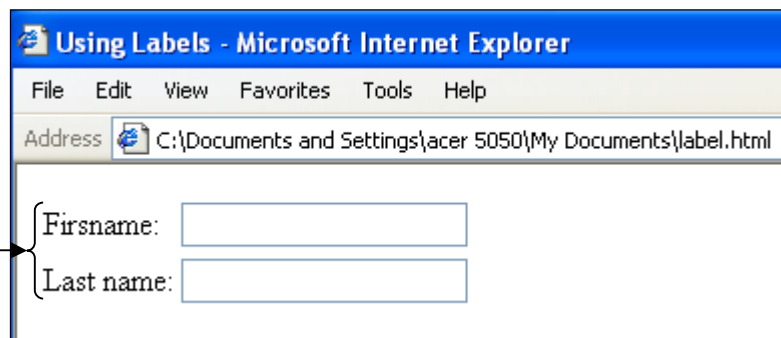
<HTML>

```

<HEAD><TITLE>Using Labels</TITLE></HEAD>
<BODY>
  <FORM action= 'http: // somesite.com' method = 'post'>
    <table>
      <tr>
        <td><LABEL for= 'firstname'>Firstname: </LABEL></td>
        <td><INPUT type='text' id='firstname'></td>
      </tr>
      <tr>
        <td><LABEL for= 'lastname'> Last name: </LABEL></td>
        <td><INPUT type= 'text' id= 'lastname'><BR></td>
      </tr>
    </table>
  </FORM>
</BODY>
</HTML>

```

Label



5.2.5 Fieldset

Cú pháp:

```

<Fieldset>
  <Legend Align="left, right">Chú thích
</Legend>
  Các thành phần trong nhóm
</Fieldset>

```

-Tag<legend>: tạo chú thích cho nhóm

-Align=left, right: chỉ vị trí của chú thích

Ví dụ:

```

<HTML>
  <HEAD><TITLE>Job application</TITLE></HEAD>
  <BODY >
    <H1><CENTER><FONT SIZE = 4 COLOR= Forestgreen>Application Form
    </CENTER></FONT></H1>
    <HR><BR><FORM action= "http: // somesite.com / processform" method =
    "post"><P>
    <FIELDSET>
      <LEGEND>Position</LEGEND>
      Application for the post of: <INPUT name= 'name' type= 'text' tabindex= '1'>
    </FIELDSET>
    <FIELDSET>
      <LEGEND>Sex</LEGEND>
      <INPUT name= 'sex' type= 'radio' value= 'Male' tabindex '4' >Male
      <INPUT name= 'sex' type= 'radio' value= 'Female' tabindex '4'> Female
    </FIELDSET>
  </BODY>
</HTML>

```

```

</FIELDSET>
<FIELDSET>
  <LEGEND>Educational Qualifications</LEGEND>
  <INPUT name= 'qualif' type='radio' value= 'grad' tabindex= '5'> Graduate
  <INPUT name= 'qualif' type='radio' value='postgrad' tabindex='5'>
Postgraduate
</FIELDSET>
<FIELDSET>
  <LEGEND>Language known</LEGEND>
  <INPUT name= 'lang' type='checkbox' value= 'english' tabindex= '6'> English
  <INPUT name= 'lang' type='checkbox' value= 'french' tabindex= '7'> French
  <INPUT name= 'lang' type='checkbox' value= 'german' tabindex= '8'>
German
</FIELDSET>
<FIELDSET>
  <LEGEND> Personal Information</LEGEND>
  Name: <INPUT name = 'name' type= 'text' tabindex= '2'><BR>
  <TEXTAREA name = 'address' rows= '3' cols = '30' tabindex = "3">
    Enter address</TEXTAREA>
</FIELDSET>
</FORM>
</BODY></HTML>

```

The screenshot shows a web browser window titled "Job application - Microsoft Internet Explorer". The address bar shows the file path "C:\Documents and Settings\acer 5050\My Documents\fieldset.htm". The main content area displays an "Application Form" with the following sections:

- Position:** A text input field with the label "Application for the post of:".
- Sex:** Two radio buttons labeled "Male" and "Female".
- Educational Qualifications:** Two radio buttons labeled "Graduate" and "Postgraduate".
- Language known:** Three checkboxes labeled "English", "French", and "German".
- Personal Information:** A text input field labeled "Name:" and a text area labeled "Enter address".

5.3 ĐIỀU KHIỂN CÁC PHẦN TỬ TRÊN FORM

1. Định thứ tự Tab:

Dùng phím tab để di chuyển giữa các đối tượng trong form, mặc định theo thứ tự của mã HTML, muốn định lại thứ tự ta dùng thuộc tính **TabIndex=n** trong tag tạo các thành phần của form, trong đó n là thứ tự của tab, có giá trị từ 0 đến 32767

Trong một form ta thường định thứ tự tab cho các thành phần : textbox, password, checkbox, radio button, textarea, menu và button

2. **Tạo phím tắt:**

– Cách tạo:

Trong tag tạo các phần tử của form ta dùng thuộc tính **Accesskey="Phím tắt"**

– Sử dụng phím tắt: Nhấn tổ hợp phím **Alt+Phím tắt**

CHƯƠNG 7. CASCADING STYLE SHEET-CSS

6.1 GIỚI THIỆU

- Bảng kiểu (style sheet) nhằm thoả mãn nhu cầu thẩm mỹ, tiện dụng nhưng giữ tính thống nhất cho trang HTML. CSS cho phép định dạng một số tính chất thông thường cùng một lúc cho tất cả các đối tượng trên trang được đánh dấu bằng tag đặc biệt
- Tiềm ích của CSS là :
 - Tiết kiệm thời gian
 - Khi thay đổi định dạng chỉ cần thay đổi CSS, các trang khác sẽ tự động cập nhật sự thay đổi đó
 - Có thể dùng các CSS cùng với JavaScript để tạo các hiệu ứng đặc biệt
- Bất lợi của CSS:
 - Không một trình duyệt nào chấp nhận nó hoàn toàn
 - Phải mất thời gian để học cách sử dụng

6.2 PHÂN LOẠI VÀ CÁCH TẠO

Có 3 loại :

- Inline style
- Internal style
- External style

a. Inline style:

Là kiểu được gán cho một dòng hoặc một đoạn văn bản, bằng cách sử dụng thuộc tính style bên trong tag muốn định dạng

Cú pháp:

`<TagName Style="property1:value1;property2: value2;...">`

Nội dung văn bản muốn định dạng

`</TagName>`

Ví dụ : `<HTML>`

`<HEAD>`

`<TITLE>Setting Properties</TITLE>`

`</HEAD>`

`<BODY>`

`<P style = "color:aqua ; font- Style:italic, text- Align:center">`

This paragraph has an inline style applied to it

`<P>` This paragraph is displayed in the default style.

`<P>` Can you see the ``difference

`` in this line

`</BODY>`

`</HTML>`

b. Internal style :

Là bảng mẫu thích hợp cho trang riêng lẻ với nhiều văn bản, bằng cách tạo bảng mẫu chung trên đầu trang và dùng cho cả trang HTML

Cú pháp:

`<Head>`

`<Style>`

`TagName{property1: value 1; property2: value 2...}`

(lặp lại cho mỗi tag có thuộc tính cần định dạng)

</Style>

</Head>

Ví dụ :

```
<HTML>
  <HEAD>
    <STYLE TYPE="text/css">
      H1,H2 { color: limegreen; font-family: Arial }
    </STYLE>
  </HEAD>
  <BODY>
    <H1>This is the H1 element</H1>
    <H2>This is the H2 element</H2>
    <H3>This is the H3 element with its default style as displayed
      in the browser</H3>
  </BODY>
</HTML>
```

c. External style :

Là một bảng kiểu được lưu trữ thành một file bên ngoài và được liên kết với trang HTML. Bảng kiểu này sẽ được áp dụng và ảnh hưởng cho tất cả các trang của một website.

– Cách tạo:

- Tạo một tập tin văn bản mới
- Nhập tên các tag muốn định dạng thuộc tính theo mẫu:

TagName{property1: value1; property2:value2;...}

- Lưu tập tin với định dạng Text Only và có phần mở rộng .css

– Cách dùng External style:

Cú pháp:

<Head>

<Link Rel=StyleSheet Type="text/css" Href="tên tập tin.css">

</Head>

Ví dụ:

Tạo tập tin **Sheet1.css**

```
H2 {color:blue; font-style:italic}
P{text-align:justify; text-indent:8pt; font:10pt/15pt "Myriad Roman","Verdana"}
```

Trang1.htm

```
<HTML>
  <HEAD><TITLE> Changing the rules</TITLE>
    <LINK REL=stylesheet HREF="sheet1.css"
      TYPE="text/css">
```



```

        </HEAD>
        <BODY>
            <H2> Changing the rules is fun</H2>
            <P> Changing the rules may not be such fun
            <H2>The H2 element again</H2>
        </BODY>
    </HTML>
Trang2.htm
    <HTML>
        <HEAD><TITLE> Changing the rules</TITLE>
        <LINK REL=stylesheet HREF="sheet1.css"
        TYPE="text/css">
        </HEAD>
        <BODY>
            <H2> This document uses the sheet1 style sheet</H2>
            <P>Selecting this option could delete all your files
            <H2>The H2 element again</H2>
        </BODY>
    </HTML>

```

6.3 ĐỊNH BẢNG MẪU CHO LỚP (CLASS)

Có thể chia các yếu tố trong HTML thành các lớp để áp dụng kiểu mẫu hiệu quả hơn

Cú pháp:

- Trong phần <Style > nhập cú pháp:

```

<STYLE>
    .ClassName{thuộc tính1:giá trị1;thuộc tính2:giá trị2;...}
</STYLE>

```

- Trong phần <Body>, đánh dấu phần nằm trong lớp bằng cú pháp:

```

<Body>

    <TagName Class="ClassName">Nội dung </TagName>

</Body>

```

Ví dụ:

```

<HTML>
    <HEAD>
        <STYLE>
            .water{color:blue}
            .danger{color:red}
        </STYLE>
    </HEAD>
    <BODY>
        <p class=water>test water
        <P class=danger>test danger
    </BODY></HTML>

```

6.4 ĐỊNH CÁC TAG RIÊNG BIỆT

Dùng áp dụng cho một phần tử riêng biệt trên trang Web

Cú pháp:

- Trong Tag Style nhập :

TagName#IDName{th/tính1: giá trị1; thuộc tính2: giá trị 2;...}

- Trong tag Body nhập :

<TagName ID=IDName> Nội dung</TagName>

Ví dụ 1:

<HTML>

<HEAD><TITLE> ID Selectors</TITLE>

<STYLE>

#control { color: red ;FONT-WEIGHT:BOLD}

</STYLE>

</HEAD>

<BODY>

Fire is this colorThis paragraph
has no style applied

</BODY></HTML>

Ví dụ 2:

<HTML>

<HEAD><TITLE> combining ID and class Selector</TITLE>

<STYLE>

.forest { color: green;font-weight:bold }

.danger { color: red;font-weight:bold }

#control{ color: blue;font-weight:bold }

</STYLE>

</HEAD>

<BODY>

<P class='forest'>green things

<P class='danger'>fire hazards

<EM class='forest'> more green things

<EM class='danger'>more fire hazards

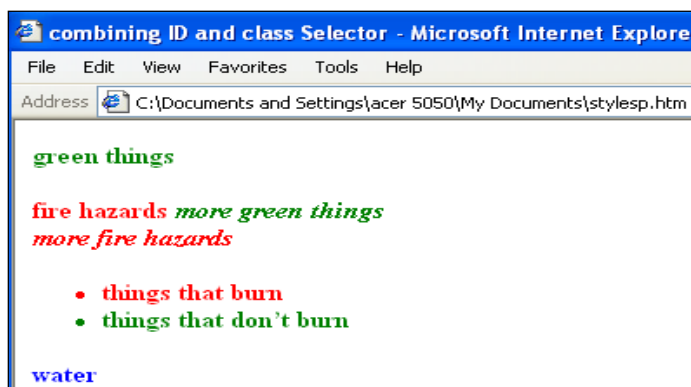
<LI class='danger'>things that burn

<LI class='forest'>things that don't burn

<P id='control'> water </P>

</BODY>

</HTML>



6.5 TẠO CÁC TAG TÙY Ý

Có 2 loại tag chung có thể kết nối Class hay các ID để tạo các tag tùy ý . cần phân biệt đối tượng cấp khối và cấp hàng:

- Đối tượng cấp khối như một đoạn văn, thường bắt đầu một dòng mới và có thể chứa các đối tượng cấp khối khác gồm các tag: P, H1, Body, table
- Đối tượng cấp hàng thường không tạo dòng mới, thường chứa văn bản và các yếu tố trong hàng khác gồm các tag: B, Font
- Các tag DIV và SPAN: có thể kết nối với các phần tử cấp khối và ID để tạo ra các tag tùy ý. Trong đó DIV phù hợp với các đối tượng cấp khối, SPAN phù hợp với các đối tượng cấp hàng

1. Tạo tag cấp khối tùy ý:

Cú pháp: Bằng cách thêm một lớp hoặc ID vào tag DIV và định mẫu cần có

- Trong phần Style hoặc một bảng mẫu bên ngoài ta nhập:
DIV.ClassName{th/tính1:giá trị 1; thuộc tính 2: giá trị 2...}
với ClassName là tên lớp sẽ sử dụng. hoặc:
DIV#IDname{thuộc tính1:g trị 1; thuộc tính 2: giá trị 2...}
với IDName là tên cá biệt của tag DIV
- Áp dụng tag cấp khối tùy ý vào trang HTML: Tại đầu phần văn bản muốn định dạng, nhập cú pháp
<DIV Class="ClassName" IDname="IDname">Nội dung </DIV>
(bên trong có thể chứa các tag <P> hoặc <H1>)

2. Tạo các tag trong hàng tùy ý:

Kết nối nhiều kiểu định dạng văn bản trong một tag

Cú pháp:

- Trong phần Style, nhập cú pháp:
SPAN.Classname {th/tính1:giá trị1; th/tính2: giá trị 2...}
Hoặc:
SPAN#IDname {th/tính1:giá trị 1; th/tính 2: giá trị 2...}
- Áp dụng tag trong hàng tùy ý vào trang HTML: Tại đầu đoạn văn bản muốn định dạng, nhập cú pháp:
** nội dung văn bản**
Hoặc:
** Nội dung văn bản**

3. Các thuộc tính định dạng văn bản:

- Chọn bộ font:
font-family: familyname1, familyname2...
- Tạo chữ nghiêng:
Font-style: italic
- Tạo chữ đậm:
Font-weight: bold
- Định cỡ chữ:
Font-size: xx-smallhoặc x-small, small, medium, large, x-large, xx-large
hoặc Font-size:12pt (giá trị cụ thể)

Có thể định dạng các thuộc tính chữ nghiêng, đậm và cỡ chữ cùng một lúc:

Font: italic bold size

- e) Màu của chữ:
Color: colorName/#rrggbb
- f) Màu nền của chữ:
Background:colorName/#rrggbb
- g) Định khoảng các giữa các từ, các ký tự:
Word-spacing:n (n: khoảng cách giữa các từ, tính bằng pixel)
Letter-spacing:n (n: khoảng cách giữa các từ, tính bằng pixel)
- h) Canh lề cho văn bản:
Text-Align: left, right, center, justify
- i) Thay đổi dạng chữ:
Text-transform: capitalize, uppercase, lowercase

4. Định dạng danh sách:

List-style:circle chấm tròn rỗng
List-style: disc chấm tròn đen
List-style: square chấm đen vuông
List-style: decimal đánh số ả rập
List-style: lower-alpha thứ tự alpha
List-style: upper-alpha thứ tự alpha chữ in hoa
List-style: upper-roman số la mã hoa
List-style: lower-roman số la mã thường

5. Định dạng màu nền:

Body{color:#rrggbb}
blockquote{ background-color:#rrggbb}
background:bacground-color
background :background-image
background: background –position
background: background-repeat
background: background-attachment

6. Định dạng Hypertext link

A{Text-Decoration:none}: không gạch dưới
A:visited{color:#rrggbb}
A:link{styles cho vị trí chưa được xem}
A:active{style cho những link đang click}
A:hover{style khi trỏ lút qua link}

CHƯƠNG 7. TỔNG QUAN VỀ JAVASCRIPT

7.1 GIỚI THIỆU VỀ JAVASCRIPT

Javascript ra đời với tên gọi LiveScript, sau đó Netscape đổi tên thành Javascript. Tuy nhiên giữa Java và Javascript có rất ít các điểm chung dù rằng cú pháp của chúng có thể có những điểm giống nhau.

Ngôn ngữ Javascript được tạo bởi Netscape vào năm 1996 và được đưa vào trong trình duyệt Netscape Navigator 2.0 của họ thông qua trình biên dịch để đọc và thực hiện các mã lệnh Javascript được kèm theo trong các trang HTML..

Javascript là một ngôn ngữ kịch bản (script) để viết kịch bản cho phía client. Client side là những yêu cầu của người sử dụng được xử lý tại máy khách. Thông thường những yêu cầu này là tính toán, kiểm tra tính hợp lệ của dữ liệu hay các hiệu ứng, các yêu cầu này thường không liên quan đến nguồn cơ sở dữ liệu trên server.

7.1.1 Đặc điểm của JAVASCRIPT

- Javascript là một ngôn ngữ kịch bản được viết chung với HTML.
- Không biên dịch như các ngôn ngữ khác. Khi trang web load xuống nó được trình duyệt thông dịch.
- Javascript là ngôn ngữ thiết kế động vì các đối tượng có khả năng tương tác với nhau thông qua người sử dụng hoặc các sự kiện.
- Là ngôn ngữ hướng đối tượng. Phân biệt chữ hoa, chữ thường
- Được hỗ trợ bởi tất cả các trình duyệt như Netscape và Internet Explorer
- JavaScript có khả năng tạo và sử dụng các đối tượng(Object), các đối tượng gồm 2 nhóm:
 - Các Object do người sử dụng tạo ra gồm :
 - Định nghĩa thuộc tính cho đối tượng
 - Cú pháp: ***Object Name.Properties***
 - Thêm phương thức cho đối tượng
 - Tạo một instance của đối tượng
 - Các object có sẵn. JavaScript cung cấp một bộ các Built-in Object để cung cấp các thông tin về sự hiện hành của các đối tượng được load trong trang Web và nội dung của nó, các đối tượng này gồm phương pháp (method) làm việc với các thuộc tính (properties) của nó.

7.1.2 Cấu trúc của đoạn Javascript

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

7.1.3 JAVASCRIPT trong một trang HTML

- Đặt các dòng mã lệnh của Javascript giữa cặp tag <script></script>
- Có thể viết nhiều đoạn mã lệnh Javascript trong cùng một tập tin HTML. Các khối mã lệnh Javascript có thể đặt bất kỳ nơi nào của trang HTML. Có thể đặt trong cặp tag <head></head> hoặc trong cặp tag <body> </body> tuy nhiên ta nên đặt trong cặp tag <head> để dễ kiểm soát mã lệnh và cũng dễ sửa đổi chương trình.
- Có thể viết một tập tin Javascript riêng và sau đó kết nối với một hoặc nhiều tập tin trang web khác nhau.

❖ **Cách 1:** Viết đoạn mã script trong cùng trang HTML

Ví dụ 1:

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

Ví dụ 2:

```
<HTML>
  <BODY>
    <script language="javascript">
      document.write("Hello World!")
    </script>
  </BODY>
</HTML>
```

Ví dụ 3:

```
<html>
  <head>
    <script type="text/javascript">
      some statements
    </script>
  </head>
  <body>
    <script type="text/javascript">
      some statements
    </script>
  </body>
</html>
```

❖ **Cách 2:**

Mở trình soạn thảo notepad, Viết đoạn chương trình Javascript. Lưu lại với phần mở rộng là **.js** (lưu ý trong tập tin này không chứa bất kỳ một thẻ nào của ngôn ngữ HTML).

- Liên kết với một **file JavaScript.js** đã được xây dựng trước

Cú pháp:

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

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

7.2 MÔI TRƯỜNG VIẾT JAVASCRIPT

Có thể dùng chương trình soạn thảo: Frontpage, Notepad, Visual InterDev, Dreamweaver để viết mã Javascript, trong giáo trình này sẽ sử dụng môi trường Dreamweaver, chọn chế độ code, Dreamweaver hỗ trợ phân biệt từ khóa bằng màu chữ, hỗ trợ các hàm, thuộc tính của các tag, giúp người sử dụng thuận tiện trong việc thiết kế và viết chương trình

7.2.1 Lệnh đơn và khối lệnh

a) *Lệnh đơn:*

Lệnh đơn là một câu lệnh được kết thúc bằng dấu chấm phẩy(;). Trong JavaScript cuối mỗi câu lệnh ta có thể dùng dấu (;) hoặc không dùng dấu gì cả .

b) *Khối lệnh:*

Khối lệnh là tập hợp nhiều câu lệnh đơn được bao bọc bởi cặp dấu {}

c) *Lời chú thích trong chương trình:*

Lời chú thích này trình duyệt sẽ bỏ qua khi thông dịch chương trình. JavaScript hỗ trợ 2 loại chú thích:

- Chú thích trên một dòng: dùng cặp dấu //
- Chú thích trên nhiều dòng: dùng cặp dấu /*...*/

7.2.2 Xuất dữ liệu ra trang Web

- JavaScript hỗ trợ 2 phương thức hiển thị dữ liệu ra trang Web là:
+ **document.write("Text")**
+ **document.writeln("Text")**
- Text là chuỗi dữ liệu muốn hiển thị ra trang Web, phải được đặt trong cặp nháy kép.
- Nếu xuất giá trị của biến thì không cần đặt trong nháy. Có thể dùng dấu + để nối các chuỗi và biến
document.write("String " + variable);
- Nếu xuất tag HTML thì cặp tag đó cũng phải đặt trong cặp dấu nháy kép
- **document.writeln:** nếu đặt trong cặp tag

```
<pre>
```

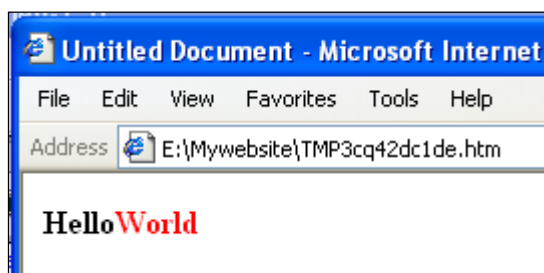
 thì lệnh document.writeln xuất dữ liệu và xuống dòng. Nếu không có cặp tag

```
<pre>
```

 thì nó cách ra một khoảng trắng

Ví dụ:

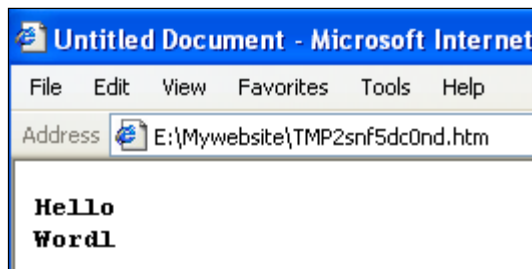
```
<BODY >
    <Script Language="JavaScript">
        document.write("<H1>Hello<H1>");
        document.write("<font color=red>World</font>");
    </Script>
</BODY>
```



Ví dụ:

```
<body>
```

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



7.3 BIẾN VÀ DỮ LIỆU TRONG JAVASCRIPT

7.3.1 Biến

- Khái niệm:** Biến là tên của một phần tử trong chương trình, được sử dụng để lưu trữ thông tin do người dùng nhập vào hoặc kết quả trung gian của quá trình tính toán. Khi khai báo biến trong Javascript không cần xác định kiểu dữ liệu cho biến cho nên khi một biến được khai báo xong nó có thể chứa bất kỳ kiểu dữ liệu nào.
- Cách khai báo biến:** Trong JavaScript, để khai báo biến dùng từ khóa **var**, cũng có thể bỏ qua từ khóa var.

var NameVariable ;

- Một biến có thể được khai báo và khởi tạo hoặc không khởi tạo giá trị ban đầu
- Muốn khai báo nhiều biến cùng một lúc thì liệt kê tên biến kế tiếp nhau cách nhau bởi dấu (,)

Ví dụ: Var x = 7 ;

var y,z = "19" ;

- Trong JavaScript, 1 biến có thể chứa bất kỳ kiểu dữ liệu gì

Ví dụ:

var a="Hello World";

a=1999 ;

- Cách xuất giá trị của biến:**

document.write(NameVariable)

- Quy tắc đặt tên biến:**

Tên biến gồm các chữ cái và số, không dùng các ký tự đặc biệt như: (, [, { , # , & theo nguyên tắc sau:

- Tên biến phải bắt đầu bằng ký tự hoặc ký tự gạch dưới(_)
- Không bắt đầu bằng ký tự số.
- Không chứa khoảng trắng, tên biến phải gọi nhớ
- Không trùng với từ khóa của JavaScript
- Các từ khóa trong JavaScript

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

5. Tầm vực của biến: là tầm ảnh hưởng của biến trong chương trình. Có 2 loại biến:

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

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

7.3.2 Dữ liệu: Có 4 loại dữ liệu

- *Kiểu số*: một biến kiểu số chứa bất kỳ giá trị số nào: số thập phân, số nguyên, số dạng chấm phẩy động.
- *Kiểu chuỗi*: một biến kiểu chuỗi có thể chứa một nhóm ký tự (Chữ cái, ký tự số, khoảng trắng, các ký tự đặc biệt, ...). Giá trị chuỗi phải đặt trong cặp dấu nháy đôi (" ") hoặc đơn (' ')

Ví dụ:

```
var s1, s2, s3 ;
s1="Hello World" ;
s2='Hello World ' ;
```

- *Kiểu Boolean*: Là dữ liệu chỉ có 2 giá trị False hoặc True thường dùng trong trường hợp biến hoặc hàm chỉ nhận một trong 2 trạng thái đúng hoặc sai.

Ví dụ: var bl;

bl=true ;

- *Kiểu Null*: là biến không gán cho giá trị

7.3.3 Toán tử

1. Toán tử số học

T toán tử	Chức Năng	Ví dụ	Kết quả
+	cộng	x=2 x+2	4
-	Trừ	x=2 5-x	3
*	Nhân	x=4 x*5	20

/	Chia	15/5 5/2	3 2.5
%	Lấy phần dư	5%2 10%8 10%2	1 2 0
++	Tăng giá trị lên 1	x=5 x++	x=6
--	Giảm giá trị xuống 1	x=5 x--	x=4

2. Toán Tử Gán

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

3. Toán Tử so sánh

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

4. Toán Tử logic

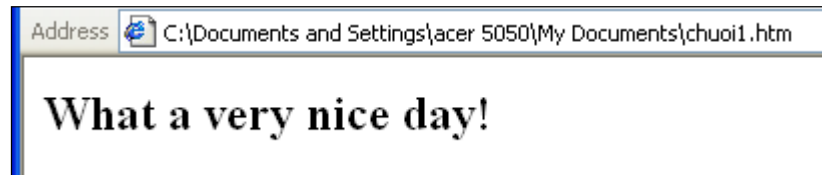
T toán Tử	Chức Năng	Ví dụ
&&	Và	x =6; y =3 ; (x < 10 && y > 1) returns true
	hoặc	x = 6 ; y =3 (x==5 y==5) returns false
!	not	x=6; y =3; !(x==y) returns true

5. Toán tử chuỗi

- Ký hiệu: + : Là phép toán nối hai chuỗi với nhau

Ví dụ:
<html>

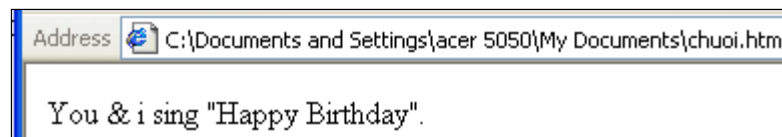
```
<script>
    txt1="What a very";
    txt2="nice day!";
    document.write('<h2>'+txt1+txt2+'</h2>');
</script>
</html>
```



- Một số ký tự đặc biệt: \n (new line), \t (tab), \b (BackSpace), \& (dấu &), \"(“)

Ví dụ:

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



6. Toán tử Điều kiện:

Cú pháp:

(Điều kiện) ? value1: value2

- Nếu biểu thức điều kiện đúng thì trả về giá trị value 1
- Nếu biểu thức điều kiện sai thì trả về giá trị value 2

Ví dụ:

```
<html>
<script>
    a=5;   b=6;
    document.write((a>b)? 'a lon hon b': 'b lon hon a');
</script>
</html>
```

7.4 CHƯƠNG 14. HÀM TRONG JAVASCRIPT

7.4.1 Định nghĩa

Hàm là một đoạn chương trình có thể được sử dụng nhiều lần trong một chương trình để thực hiện một tác vụ nào đó.

Xây dựng hàm:

Trong JavaScript, dùng từ khoá function để định nghĩa hàm.

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

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

Ví dụ:

```
function Display(user , pwd)
{
    document.write("UserName của bạn là:" + user) ;
    document.write("Password của bạn là:" + pwd) ;
    return ;
}
```

- **Câu lệnh return**: là câu lệnh kết thúc hàm. Câu lệnh này là tùy chọn. Có thể bỏ qua, nếu hàm có giá trị trả về thì cần có câu lệnh Return để trả về giá trị. Sau Return có thể chứa hoặc không chứa một giá trị cụ thể hoặc một biểu thức tính toán.

Ví dụ:

```
Function total(a,b)
{
    C=a+b;
    Return c;
}
```

7.4.2 Cách gọi hàm

- Hàm sẽ không thực hiện cho đến khi nó được gọi.
- Đối với hàm có đối số ta gọi tên hàm và danh sách các giá trị truyền cho đối số đó
FunctionName(argument1,argument2,etc)
- Đối với hàm không có đối số ta chỉ cần gọi tên hàm là được.

FunctionName()

- Đối với hàm không có giá trị trả về :
NameFunction(parameter) .
- Đối với hàm có giá trị trả về :
variable= NameFunction(parameter) .

Ví dụ:

```
<html>
  <head><title>Function</title></head>
  <body>
    <script>
      function Area(Width, Length)
      {
        size=Width*Length;
        return size;
      }
      x=eval(prompt("Nhập x: "));
      y= eval(prompt("Nhập y: "));
      document.write(Area(x,y))
    </script>
  </body>
</html>
```

7.4.3 CÁC HÀM THÔNG DỤNG TRONG JAVASCRIPT

- 1) **Hàm alert():** dùng hiển thị một hộp thông báo có nút OK

Cú pháp:

alert("nội dung thông báo")

ví dụ:

```
<html>
  <head><title>Function</title></head>
  <body>
    <script>
      alert("Hello World")
    </script>
  </body>
</html>
```



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

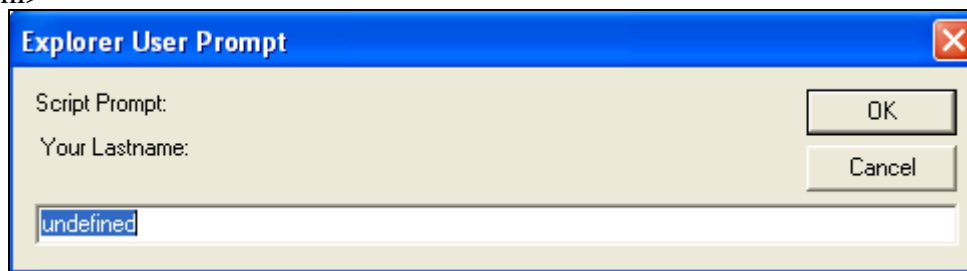
Cú pháp:

variable= prompt("nội dung đối thoại", giá trị khởi tạo);

ví dụ:

```
<html>
  <head><title>Function</title></head>
  <body>
    <script>
      a=prompt("Your Lastname:");
      b=prompt("Your FirstName");
      document.write("Your FullName is :"+ a + ' ' + b)
    </script>
  </body>
```

</html>



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

Cú pháp:

variable=confirm("Chuoi thong bao");

Ví dụ:

<html>

<head><title>Function</title></head>

<body>

<script>

a=prompt("nhap so a :");

b=prompt("nhap so b");

c=confirm(a +' lon hon '+ b+'?')

if(c= true)

document.write(a +" > "+b)

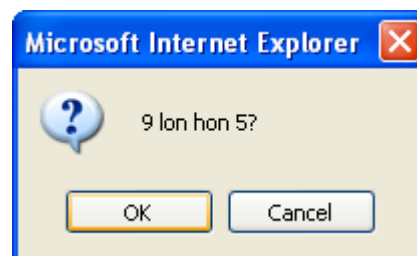
else

document.write(a +" < "+b)

</script>

</body>

</html>



- 4) **Các hàm thông dụng của chuỗi và số:**

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

Cú pháp:

eval(chuỗi số)

Ví dụ:

<script>

var str1="123", str2="456";

str= str1+str2;

document.write(str);→ kết quả :123456

</script>

<script>

var str1="123", str2="456";

str=eval(str1)+eval(str2) ;

document.write(str)→kết quả: 579

</script>

- b. **Hàm parseInt(strNum)**

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

Ví dụ :

var strNum="123.8" , kq;

```
kq=parseInt(strNum) =>kq=123
strNum="a123"
kq=parseInt(strNum) =>kq=NaN
strNum="123.8abc"
kq=parseInt(strNum)=>kq=123
```

c. Hàm parseFloat(strNum):

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

Ví dụ:

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

d. Hàm isNaN(str):

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

Ví dụ :

```
Var str="123abc", kq;
kq=isNaN(str) =>kq=true;
str="123.8"
kq=isNaN(str) =>kq=false ;
```

5) Các hàm thiết lập thời gian:

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

Cú pháp:

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

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

Ví dụ:

```
Idq=setTimeout("alert('Da het gio')",1000) ;
Cứ 1000 mili giây thì thông báo đã hết giờ một lần.
```

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

Cú pháp:

clearTimeout(IdTime);

Ví dụ:

```
clearTimeout(Idq);
```

- c. Hàm setInterval() và clearInterval()** với ý nghĩa và tham số giống như setTimeout() và clearTimeout() .

7.5 CHƯƠNG 14. CÁC CẤU TRÚC ĐIỀU KIỆN

7.5.1 Cấu trúc lựa chọn

7.5.1.1 Câu lệnh if

- **Mẫu 1:** Áp dụng cho trường hợp có 1 điều kiện và 1 công việc xử lý

Cú pháp:

if (<Biểu thức điều kiện>)

Khối lệnh 1;

Khối lệnh 2;

Nguyên tắc hoạt động: Nếu biểu thức điều kiện đúng thì thực hiện khối lệnh 1, sau đó thực hiện khối lệnh 2, ngược lại nếu biểu thức điều kiện sai thì bỏ qua khối lệnh 1 và thực hiện khối lệnh 2

- **Mẫu 2:** Áp dụng cho trường hợp có 1 điều kiện và 2 lựa chọn công việc xử lý

Cú pháp:

if(<biểu thức điều kiện>)

Khối lệnh1;

else

Khối lệnh 2 ;

Khối lệnh 3;

Nguyên tắc hoạt động: Nếu biểu thức điều kiện đúng thì thực hiện khối lệnh 1, sau đó thực hiện khối lệnh 3, ngược lại thì thực hiện khối lệnh 2, sau đó thực hiện khối lệnh 3

- **Mẫu 3 (if ...else lồng nhau):** Áp dụng cho trường hợp có nhiều lựa chọn khác nhau

Cú pháp:

if(<biểu thức điều kiện1>)

Khối lệnh 1;

else

if (<biểu thức điều kiện 2>)

Khối lệnh 2 ;

else

...

khối lệnh 3

Để áp dụng mẫu 3, cần phải xác định biểu thức điều kiện của bài toán rồi sắp xếp thứ tự lồng nhau cho hợp lý.

Ví dụ: Viết chương trình nhập 3 cạnh của tam giác sau đó xuất ra màn hình đó là tam giác gì?

```
<Body><script>
```

```
a=eval(prompt("Nhập cạnh a"));
```

```
b=eval(prompt("Nhập cạnh b"));
```

```
c=eval(prompt("Nhập cạnh c"));
```

```
if(a==b && b==c && c==a)
```

```
    Tam giác đều ;
```

```
else
```

```
    if(a==b || b==c || c==a)
```



```

        Tam giác cân
    Else
        Tam giác thương
</script></Body>

```

7.5.1.2 Cấu trúc chọn lựa switch...case

Áp dụng trong trường hợp muốn chọn một trong các giá trị của biểu thức để thực hiện lệnh. Giá trị của biểu thức có thể là một chuỗi hoặc một số

- **Mẫu 1:**

```

switch(Biểu thức)
{
    case value1:
        Khởi lệnh 1;
        break;
    case value2:
        Khởi lệnh 2 ;
        break;
    .....
    case valuek:
        Khởi lệnh k ;
        break;
}

```

- **Mẫu 2:**

```

switch(biểu thức)
{
    case value1:
        khởi lệnh 1 ;
        break;
    case value2:
        khởi lệnh 2 ;
        break;
    .....
    case valuek:
        khởi lệnh k ;
        break;
    default :
        khởi lệnh k+1 ;}

```

Nguyên tắc hoạt động:

- Trình thông dịch sẽ tính giá trị của biểu thức rồi so sánh với các value, nếu bằng giá trị nào thì thực hiện khối lệnh đó.
- Sự khác nhau giữa mẫu 1 và 2 là: ở mẫu 2 khi so sánh giá trị của biểu thức với các value, nếu không khớp thì thực hiện lệnh trong default
- Trong trường hợp có nhiều value khác nhau mà cùng thực hiện một khối lệnh thì liệt kê các value liên tiếp nhau và cách nhau dấu phẩy.

case valuej1 ,valuej2 ,...,valuejk : khởi lệnh; break;

Ví dụ:

```

<body>
<script>
    t=prompt("nhập thang: ");
    switch(eval(t))

```

```

    {
        case 1: case 3: case 5: case 7: case 8 : case 10: case 12:
            alert("Thang "+ t+ " co 31 ngay");
            break;
        case 2:
            alert("Thang "+t + " co 28 ngay");
            break;
        case 4: case 6: case 9: case 11:
            alert("Thang "+t +" co 30 ngay");
            break;
        default:
            alert("Khong co thang nay");
    }
</script>
</body>

```

7.5.2 CẤU TRÚC LẶP

Được áp dụng khi một công việc nào đó muốn thực hiện lặp đi lặp lại nhiều lần với một điều kiện nào đó. Có 2 loại cấu trúc lặp là : lặp với số lần lặp biết trước và lặp với số lần lặp không biết trước

7.5.2.1 Vòng lặp For

Thường áp dụng cho số lần lặp biết trước

Cú pháp:

for(biểu thức 1; biểu thức 2; biểu thức 3)

```

{
    Khối lệnh 1;
}
khối lệnh 2;

```

Trong đó :

biểu thức 1: chứa giá trị khởi tạo của biến điều khiển

biểu thức 2 : chứa biểu thức điều kiện lặp.

biểu thức 3: chứa biểu thức tăng hoặc giảm biến điều khiển .

Nguyên tắc hoạt động::

- Trình thông dịch gán giá trị khởi tạo cho biến điều khiển, Kiểm tra biểu thức 2, nếu đúng thì thực hiện khối lệnh 1, chuyển lên thực hiện biểu thức 3, tiếp tục kiểm tra biểu thức 2, và tiếp tục ...
- Nếu biểu thức 2 có giá trị sai thì chương trình thoát khỏi vòng lặp và thực hiện khối lệnh 2.
- Nếu khối lệnh 1 có chứa câu lệnh Break thì chương trình sẽ thoát khỏi vòng lặp for và thực hiện khối lệnh 2

Ví dụ: Viết chương trình tạo một table m dòng n cột.

```

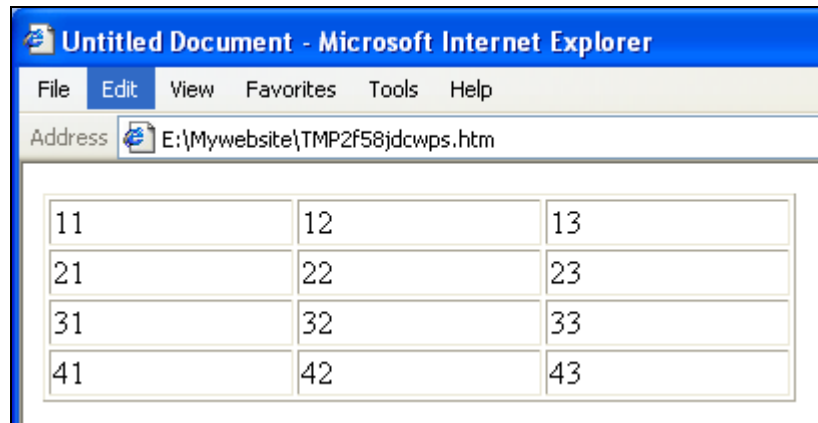
<body>
    <Script language="javascript">
        var n, m, i, j;

```

```

m=prompt("Nhap so dong");
n=prompt("Nhap so cot");
document.write("<table width=50% border=1>");
for(i=1;i<=m;i++)
{
    document.write("<tr>");
    for(j=1;j<=n;j++)
        document.write("<td>" + i + j + "</td>");
    document.write("</tr>");
}
document.write("</table>");
</Script>
</body>

```



7.5.2.2 Vòng lặp while

thường áp dụng cho số lần lặp không xác định

1. **Vòng lặp While:** Kiểm tra điều kiện trước khi thực hiện lệnh

Cú pháp:

```

while(biểu thức điều kiện)
{
    Khối lệnh 1;
}
Khối lệnh 2;

```

Nguyên tắc hoạt động:

- Trình thông dịch kiểm tra biểu thức điều kiện, nếu đúng thì thực hiện khối lệnh 1, sau đó quay lại kiểm tra biểu thức điều kiện, và tiếp tục ..., nếu sai thì thực hiện khối lệnh 2.
- Như vậy khối lệnh 1 có thể không được thực hiện lần nào nếu ngay từ đầu biểu thức điều kiện sai
- Thường khối lệnh 1 chứa lệnh làm thay đổi giá trị của biểu thức điều kiện để có thể thoát ra khỏi vòng lặp, hoặc chứa lệnh break để thoát khỏi vòng lặp *while*

Ví dụ:

```
<script language="javascript">
```

```
var userInput;
while ((userInput!=99 )
{
    userInput=prompt("Nhập vào một số bảy kỳ, nhập 99 để thoát")
    if(isNaN(userInput))
    {
        document.write("Dữ liệu không hợp lệ, nhập số ");
        break;
    }
}
</script>
```

2. Vòng lặp do ...while: Thực hiện lệnh trước sau đó kiểm tra biểu thức điều kiện

Cú pháp:

```
do
{
    khối lệnh 1;
} While(biểu thức điều kiện);
khối lệnh 2;
```

Nguyên tắc hoạt động: trình thông dịch thực hiện khối lệnh 1, sau đó kiểm tra biểu thức điều kiện, nếu đúng thì thực hiện lại khối lệnh 1, nếu sai thì thoát khỏi vòng lặp và thực hiện khối lệnh 2

Ví dụ:Viết chương trình yêu cầu người dùng nhập vào một số, kiểm tra xem giá trị nhập có phải là số không, nếu không yêu cầu nhập lại.

```
<script language="javascript">
var userInput;
do
{
    userInput=prompt("Nhập vào một số bảy kỳ, nhập 99 để thoát")
    if(isNaN(userInput))
    {
        document.write("Dữ liệu không hợp lệ, nhập số ");
        break;
    }
}while ((userInput!=99 )
</script>
```

3. Vòng lặp for ...in: dùng để duyệt qua các thuộc tính của một đối tượng hay giá trị của các phần tử trong mảng

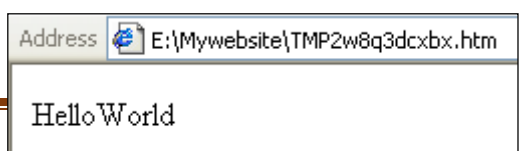
Cú pháp:

```
for ( variable in Object)
{
    khối lệnh 1 ;
}
khối lệnh 2;
```

Nguyên tắc hoạt động: trình thông dịch sẽ duyệt qua tất cả các phần tử trong Object.

Ví dụ:

```
<body>
<script>
```



```

obj= new Array() ;
obj[0]="Hello";
obj[1]="World" ;
for(i in obj)
    document.write(obj[i]);
</script>
</body>

```

7.5.2.3 Câu lệnh try ...catch và throw

Cú pháp:

```

try
{
    khối lệnh ;
}
catch(objErr)
{
    Xử lý lỗi ;
}

```

Nguyên tắc hoạt động:

- Trình thông dịch thực thi các lệnh trong khối lệnh, nếu trong quá trình thực thi có lỗi xảy ra thì trình thông dịch truyền đối tượng lỗi cho catch.
- Câu lệnh catch tự động gửi vào tham số có chứa đối tượng lỗi, đối tượng này có 2 thuộc tính number và description. mỗi dạng lỗi trong mã kịch bản sẽ được gán cho một con số lỗi duy nhất. thuộc tính Number chứa một số nguyên lỗi, thuộc tính description chứa một mô tả dạng văn bản về lỗi.

Ví dụ:

```

<head><title>Chương trình kiểm tra lỗi</title>
<Script language="JavaScript">
    var str ;
    try
    {
        document.write("Hello World");
        Math.r();
    }
    catch(objerr)
    {
        str="Lỗi thu " + objerr.number + "<br>";
        str="Và lỗi do là " + objerr.description;
        alert(str);
    }
</Script>
</head>

```

- Câu lệnh throw được dùng để truyền một thông báo lỗi đến một câu lệnh catch. Nó cũng có thể được dùng để truyền một lỗi lên
- Bộ xử lý lỗi mức cao hơn trong trường hợp có nhiều câu lệnh try...catch lồng nhau

Ví dụ:

```

<Html><head><title>Chương trình kiểm tra lỗi</title>
<Script language="JavaScript">

```

```
var str , m=4 ,kq;
try
{
    try
    {
        document.write("Hello World");
        kq=m/n;
    }
    catch(objerr)
    {
        str="Loi thu " + objerr.number + "<br>";
        str="Va loi do la " + objerr.description;
        if (kq= =4)
            alert("n=1") ;
        else
            throw (objerr) ;
    }
    catch (objerr)
    {
        alert(objerr.number + objerr.description) ;
    }
}
</Script></head></html>
```

7.6 CHƯƠNG 14. MÔ HÌNH ĐỐI TƯỢNG

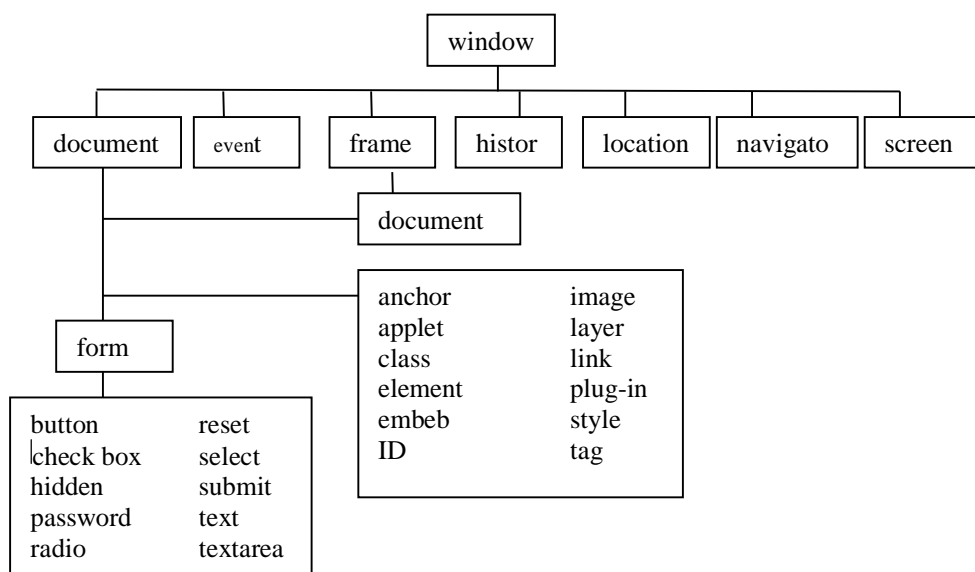
8.1.1 MÔ HÌNH DOM ((Document Object Model)

1. Đối tượng – Mô hình đối tượng:

- Mỗi thành phần trên trang web được xem như một đối tượng, mỗi đối tượng đều có các thuộc tính, phương thức và sự kiện của nó. Ví dụ hình ảnh, văn bản, button, một tag trong HTML cũng được xem như là một đối tượng và các thuộc tính của tag đó được xem như là đối tượng con của nó. Tất cả các đối tượng trong Javascript được tổ chức phân cấp dạng hình cây gọi là mô hình DOM (Document Object Model), Ở mức trên cùng là đối tượng **window** biểu thị cho khung hay cửa sổ của trình duyệt, các phần tử còn lại là đối tượng con của window
- Dùng thuộc tính ID để truy xuất đến một đối tượng trong IE và thay đổi dữ liệu cho chính phần tử đó, tất cả các đối tượng trên trang đều có một ID duy nhất

Ví dụ:

```
<tr><td Id=IdName1>Data</td></tr>
<img Id=Id Name2>
<Iframe Id=IdName3></Iframe>
```



- Muốn truy cập vào đối tượng nào thì phải truy cập vào đối tượng chứa nó trước, dùng toán tử dấu chấm (.) để phân cách giữa các đối tượng. Tuy nhiên ta có thể bỏ qua đối tượng window nếu đang thao tác trên cửa sổ hiện hành

Ví dụ: window.location

- Ngoài các đối tượng do chương trình xây dựng sẵn, có thể tạo thêm những đối tượng mới cần thiết cho nhu cầu sử dụng.
- Mỗi đối tượng đều có *thuộc tính*, *sự kiện* và *phương thức*, nhờ các thành phần này mà có thể truy cập và thay đổi nội dung của chúng.
- o Thuộc tính (Properties): là nơi chứa các mô tả thông tin của đối tượng.

Ví dụ:

Trong tag Img có 3 thuộc tính src, width, height, để thay đổi kích thước của hình thì thay đổi giá trị của thuộc tính width, height hoặc đổi hình khác thì

thay đổi giá trị của thuộc tính, để thực hiện các việc trên ta dựa vào thuộc tính Id là Idh1.

```
Ví dụ: function ZoomIn()
{
    Idh1.width=Idh1.width +10;
    Idh1.height=Idh1.height + 5;
}
function ChangeImg(file)
{
    Idh1.src=file ;
}
```

- *Sự kiện (event)*: là các hành động, sự việc xảy ra trên trang web: click chuột, di chuyển chuột, ...gọi là sự kiện. Sự kiện được xử lý bởi các đoạn mã kịch gọi là bộ xử lý sự kiện

- *Các sự kiện thường sử dụng:*

Tên sự kiện	Ý nghĩa
Onmousedown	Phát sinh khi người sd dùng nhấn chuột
Onmouseover	Phát sinh khi người sd dùng d/chuyển chuột lên đối tượng
Onmouseout	Phát sinh khi người sd dùng d/chuyển chuột ra ngoài đtượng
Onkeypress	Phát sinh khi người sử dụng nhấn một phím
Onfocus	Phát sinh khi đối tượng nhận tiêu điểm
Onblur	Phát sinh khi rời khỏi đối tượng
Onclick	- Người dùng click chuột vào đối tượng - Một đối tượng đang có tiêu điểm, người sd nhấn enter - Một checkbox hoặc nút chọn đang có tiêu điểm, người sd nhấn phím Spacebar
Onload	- Phát sinh khi đối tượng được tải xuống
OnUnload	- Phát sinh khi đối tượng được nạp trở lại hoặc chuyển trang
Onresize	- Phát sinh khi cửa sổ bị thay đổi kích thước
Onselect	- Phát sinh khi đối tượng được chọn
Onchange	- Phát sinh khi đối tượng thay đổi giá trị
Onsubmit	- Phát sinh khi Form được Submit

- *Cách sử dụng các sự kiện*: muốn điều khiển sự kiện, ta thêm sự kiện đó vào trong thẻ HTML.

- Cú pháp:< TagName event_handler="JavaScript Command">

- TagName: tên tag
- event_handler: tên sự kiện
- JavaScript Command: gọi hàm xử lý sự kiện

- *Phương thức*: Là các hàm được xây dựng trước có tác dụng làm thay đổi thuộc tính của đối tượng.

2. Mục đích của mô hình DOM:

- Để định nghĩa 1 tổ chức phân cấp thể hiện các phần của 1 hồ sơ web.
- Cho phép thay đổi cấu trúc đó thông qua việc thêm bớt nội dung
- Cung cấp cách thức quan sát, thao tác các đặc tính của nội dung trên trang web
- Cung cấp thông tin về cách tương tác giữa các mục trên trang web với người dùng
- Nó cho phép thông báo các sự kiện gây ra do chuột và bàn phím

8.1.2 Xây dựng một đối tượng mới

Ngoài các đối tượng có sẵn trong Javascript, ta có thể tạo các đối tượng mới với các phương thức và thuộc tính riêng cho đối tượng đó.

1. Cách xây dựng một đối tượng mới: Gồm 2 bước

- a. **Bước 1:** Định nghĩa đối tượng bằng cách xây dựng hàm cho đối tượng đó gồm các phương thức và thuộc tính cho đối tượng đó.

```
function Object(List Parameter)
{
    this.property1= Parameter1;
    this.property2= Parameter2;
    ...
    this.method1=functionName1;
    this.method2=functionName2;
    ...
}
```

Trong đó

- Từ khoá **this** để tham chiếu đến đối tượng đang được tạo. Khi xây dựng đối tượng có bao nhiêu thuộc tính thì dùng từ khoá this để tham chiếu đến bấy nhiêu thuộc tính của nó
- Câu lệnh **this.property1= Parameter1**: gán giá trị Parameter1 cho thuộc tính property1
- Tương tự: muốn xây dựng phương thức cho đối tượng thì gán phương thức cho hàm đã định nghĩa sẵn

```
this.method1=FunctionName1;
```

- b. **Bước 2:** Tạo instance cho đối tượng, dùng từ khoá **new**

```
var obj=new Object();
```

- Truy cập hoặc thay đổi g/ trị của thuộc tính ta sử dụng: **obj.property**
- Muốn sử dụng phương thức method1 thì dùng **obj.method()**

Ví dụ:

Xây dựng một đối tượng Student gồm các thuộc tính IdStudent, Name, Address và phương thức Display() hiển thị thông tin của Student

Bước 1: Xây dựng đối tượng Student

```
function Student(masv,hten,dchi) // Đối tượng
{
    this.IdStudent=masv;
    this.Name=hten;
    this.Address=dchi;
    this.Display=Information;
}
function Display() //Phương thức
{
    document.write("Ma SV "+this.IdStudent + "<br>");
    document.write("Ho ten SV "+this.Name + "<br>");
    document.write("Dia chi SV "+this.Address + "<br>");
}
```

- Muốn xuất thông tin SV ta gọi phương thức Display()

- Sử dụng đối tượng Student: tạo instance cho đối tượng

```
var st=new Student() ; //Tạo thể thiện cho đối tượng
st.IdStudent="TH01";
st.Name="Truong Tam Phong" // Khởi tạo giá trị cho đtượng
st.Address="12 Nguyen Cuu Van"
```
- Hiện thị thông tin của đối tượng thì gọi đến phương thức Display()
st.Display()
- Có thể khai báo và khởi tạo đối tượng bằng cách:

```
var st=new("TH01","Truong Tam Phong","12 Nguyen Cuu Van")
```

8.1.3 CÁC ĐỐI TƯỢNG CÓ SẴN TRONG JAVASRIPT

8.1.3.1 Đối tượng Array()

Đối tượng Array dùng để lưu trữ nhiều giá trị với cùng một tên gọi. Trong Javascript đối tượng mảng có thể chứa các thành phần mang kiểu giá trị khác nhau. Một mảng có n phần tử được đánh chỉ số từ 0 đến n-1.

Mỗi phần tử mảng được phân biệt nhau qua chỉ số, dựa vào chỉ số này ta có thể truy cập hoặc thay đổi giá trị của từng phần tử trong mảng

1. Khởi tạo một mảng:

Dùng từ khóa **new** để khởi tạo một mảng

var Variable = new Array(size)

Ví dụ: <script>

```
var arr= new Array()
arr[0]= "thu hai";
arr[1]= "Thu ba";
arr[2]= "Thu tu";
arr[3]= "Thu nam";
arr[4]= "Thu sau";
arr[5]= "Thu bay";
for(i=0; i<=5;i++)
document.write(arr[i]+ "<br>")
```

</script>

2. Các thuộc tính của Array()

- **length** : để xác định số phần tử trong mảng

Ví dụ: <script>

```
var arr= new Array()
arr[0]= "thu hai";
arr[1]= "Thu ba";
arr[2]= "Thu tu";
arr[3]= "Thu nam";
arr[4]= "Thu sau";
arr[5]= "Thu bay";
document.write("So phan tu trong mang la: " +arr.length)
```

</script>

3. Các phương thức của đối tượng Array()

Phương thức	Ý nghĩa	Ví dụ
concat()	Dùng để nối 2 mảng	a=a.concat(b)
join(separator)	để ghép các phần tử trong mảng lại với nhau cách nhau bởi dấu separator	a=a.join("+")
slice(start,end)	Dùng tách một mảng bắt đầu từ vtrí start đến vtrí end-1.	str=a.slice(i,j)
reverse()	Dùng để đảo ngược chuỗi	a.reverse()
valueOf()	Dùng để lấy tất cả các đối tượng trong chuỗi	a.valueOf()
pop()	Lấy phần tử cuối của mảng	
push()	Thêm 1 hoặc nhiều phần tử vào cuối mảng	
Shift()	lấy phần tử và trả về phần tử đầu tiên của mảng	
Sort()	sắp xếp các phần tử của mảng	
valueOf()	Trả về tất cả các giá trị ban đầu của mảng	

Ví dụ 1:

```
<script>
    var a=new Array()
    a[0]= "Thang gieng";
    a[1]= "Thang hai";
    a[2]= "Thang ba";
    var b =new Array();
    b[0]= "Thang tu";
    b[1]= "Thang nam";
    b[2]= "Thang sau";
    a=a.concat(b);
    document.write(a);
</script>
```

Ví dụ 2

```
<script type="text/javascript">
    var arrName = new Array(3)
    arrName [0] = "Jani"
    arrName [1] = "Tove"
    arrName [2] = "Hege"
    document.write(arrName.length + "<br>")
    document.write(arrName.join(".") + "<br>")
    document.write(arrName.reverse() + "<br>")
    document.write(arrName.sort() + "<br>")
    document.write(arrName.push("Ola","Jon") + "<br>")
    document.write(arrName.pop() + "<br>")
    document.write(arrName.shift() + "<br>")
</script>
```

8.1.3.2 Đối tượng Date()

1. Cách khai báo: Có 2 cách khai báo

Cách 1: Khai báo và khởi tạo

var variableName= new Date("month, day, year , hours : minutes : seconds")

hoặc:

```
var variableName= new Date(year,month,day,hours,minutes,seconds)
```

hoặc:

```
var variableName= new Date(year,month, day)
```

```
var variableName= new Date("Month dd, yyyy hh:mm:ss")
```

```
var variableName= new Date("Month dd, yyyy")
```

```
var variableName= new Date(yy,mm,dd,hh,mm,ss)
```

```
var variableName= new Date(yy,mm,dd)
```

```
var variableName= new Date(milliseconds)
```

- variableName là biến dùng để lưu trữ thông tin ngày tháng năm, giờ phút giây.
- Trường hợp 1: giá trị khởi tạo là 1 chuỗi. Trong trường hợp này month là chuỗi,
- Trường hợp 2 và 3, giá trị là một số.

Ví dụ:

```
var objday =new Date("November,1,2003,7:30:9") // Khai báo hợp lệ
```

```
var objday= new Date("10,1,2003,7:30:9") //Khai báo không hợp lệ
```

Cách 2: Khai báo ngày hiện hành (Không khởi tạo)

```
var variableName=new Date()
```

- Trong trường hợp này giá trị trả về là ngày tháng năm giờ phút giây hiện hành của hệ thống.

2. Các phương thức của đối tượng Date():

- Để truy xuất phương thức của đối tượng dùng cú pháp

```
variableName.Method()
```

Phương thức	Mô tả
Date()	trả về đối tượng date
getDate()	Trả về giá trị ngày (số nguyên từ 1-31) trong tháng
getDay()	Trả về giá trị ngày trong tuần (số nguyên từ 0-6 Sunday=0)
getMonth()	Trả về tháng trong năm (from 0-11. 0=January, 1=February)
getFullYear()	Trả về giá trị năm (bốn số)
getYear()	Trả về giá trị năm (hai số)
getHours()	Trả về giờ của hệ thống (từ 0-23)
getMinutes()	Trả về phút của hệ thống (từ 0-59)
getSeconds()	Trả về giây của hệ thống (từ 0-59)
getMilliseconds()	Trả về giá trị millisecond from 0-999)
setFullYear(years)	Thiết lập lại năm cho ngày hệ thống (4 số)
setHours(hours)	Thiết lập lại giờ cho hệ thống (từ 0-24)
setMinutes(minutes)	Thiết lập lại phút cho hệ thống (từ 0-59)
setMonth(months)	Thiết lập lại tháng cho hệ thống (từ 0-11)
setSeconds(seconds)	Thiết lập lại giây cho hệ thống (from 0-59)
toGMTString()	Chuyển ngày giờ hệ thống sang ngày giờ quốc tế.

toString()	Chuyển ngày giờ hệ thống sang chuỗi
------------	-------------------------------------

Ví dụ : Hiển thị giờ trên thanh trạng thái 9:12:48 P.M.

```
<html>
  <head><title>Digital Clock - Status Bar</title>
  <script Language="JavaScript">
    var timerID = null;
    var timerRunning = false;
    function stopclock ()
    {
      if(timerRunning)
        clearTimeout(timerID);
        timerRunning = false;
    }
    function showtime ()
    {
      var now = new Date();
      var hours = now.getHours();
      var minutes = now.getMinutes();
      var seconds = now.getSeconds();
      var timeValue = "" + ((hours >12) ? hours -12:hours);
      timeValue += ((minutes < 10) ? ":0": ":") + minutes
      timeValue += ((seconds < 10) ? ":0": ":") + seconds
      timeValue += (hours >= 12) ? " P.M.": " A.M."
      window.status = timeValue;
      timerID = setTimeout("showtime()",1000);
      timerRunning = true;
    }
    function startclock ()
    {
      stopclock();
      showtime();
    }
  </script>
</head>
<body BGCOLOR="#FFFFFF" TEXT="#000000" LiNK="#FF0000"
VLiNK="#000080" ALiNK="#000080"
onLoad="startclock()">
</body>
</html>
```

Ví dụ:

```
<html>
  <head><title>Hiển thị ngày giờ lên trang web </title>
  <script language="javascript">
    function Ngay()
    {
      var day=new Date();
      var w,m,d,y;
      var arrday=new Array();
      arrday[0]= "chu nhật";
```

```

        arrday[1]= "Thu hai ";
        arrday[2]= "Thu ba ";
        arrday[3]= "Thu tu";
        arrday[4]= "Thu nam ";
        arrday[5]= "Thu sau";
        arrday[6]= "Thu Bay";
        w=day.getDay();
        d=day.getDate();
        m=day.getMonth()+1;
        y=day.getFullYear();
        document.write("Hom nay:"+arrday[w]+" ngay "+d+" thang "+m+ "
        nam "+y);
    }
</script>
</head>
<body>
<script language="Javascript">Ngay()</script>
</body>
</html>

```

8.1.3.3 Đối tượng String

Mỗi chuỗi trong JavaScript là một đối tượng, gồm các thuộc tính và phương thức thực hiện trên chuỗi, đó là các phương thức tìm kiếm chuỗi, trích chuỗi con và áp dụng các thẻ HTML vào nội dung của chuỗi.

1. Cách khai báo đối tượng String

```
var stringVariable=new String()
```

2. Thuộc tính của String():

- **Length:** dùng để xác định chiều dài của chuỗi. Các ký tự trong chuỗi được đánh chỉ số từ 0 đến Length-1. Tất cả các thành phần có giá trị chuỗi đều dùng được thuộc tính length. Cách tham chiếu đến thuộc tính length của đối tượng String().
 - Cách 1: StringLength=stringVariable.length
 - Cách 2:
 - var st=new Stringt()
 - StringLength=stName.length
 - Cách 3: StringLength="This is a string".length

3. Các phương thức của String: Các phương thức của String để thực hiện các thao tác trên nội dung của chuỗi:

Phương thức	Mô tả	Ví dụ
anchor("anchormame")	Trả về một chuỗi liên kết anchormame trở thành 1 liên kết	str.anchor("anchormame")) This is a string
big()	Trả về một chuỗi đặt trong cặp thẻ <big>	str.big() <big>This is a string </big>

Bold()	Trả về một chuỗi in đậm	str.bold() This is a string
charAt(index)	Trả về ký tự thứ index trong chuỗi. index từ 0 đến str.length-1	str.charAt(0)="T"
concat()	Trả về hai chuỗi nối nhau	
fontcolor()	Trả về một chuỗi với màu đã được xác lập.	str.fontcolor("red") This is a string
fontsize()	Trả về một chuỗi với kích thước đã được xác lập.	str.fontsize("5") This is a string
indexOf(searchvalue, [fromindex])	Trả về vị trí của đầu tiên được tìm thấy của chuỗi searchvalue bắt đầu tìm từ vị trí fromindex. Nếu không có fromindex thì tìm từ vị trí 0. Nếu không tìm thấy thì hàm trả về giá trị -1	Pos=str.indexOf("is") Pos=2
italics()	Trả về một chuỗi in nghiêng	
lastIndexOf(searchvalue)	Trả về vị trí của cuối cùng được tìm thấy của chuỗi searchvalue bắt đầu tìm từ phải qua trái. Nếu không tìm thấy thì hàm trả về giá trị -1	
link()	Trả về một chuỗi liên kết	
match()	Tương tự như hàm indexOf và lastIndexOf, nhưng phương thức này trả về một chuỗi cụ thể nếu không tìm thấy thì trả về giá trị "null".	
replace()	Thay thế một vài ký tự bằng một vào ký tự mới	
search()	Trả về giá trị là số chuỗi được tìm thấy trong chuỗi cha, nếu không tìm thấy thì trả về giá trị -1	
slice()	Trả về một chuỗi con được cắt từ chuỗi mẹ tại vị trí cắt	
small()	Trả về một chuỗi nhỏ hơn	
strike()	Trả về một chuỗi được gạch ngang qua thân chuỗi	
sub()	Trả về một chuỗi kiểu subscript	Str.sub() _{This is a string}
substr(start,length)	Trả về chuỗi con bắt đầu từ vị trí start và có chiều dài length. nếu không có start xem như start=0	Str.substr(0,2)="Th"
substring(Start,end)	Tách ra một chuỗi con từ một chuỗi. Bắt đầu từ chỉ số start đến end. Nếu Start<end, chuỗi trả về từ start đến end-1 Nếu end<start, chuỗi trả về từ end đến	

	start Nếu start=end chuỗi trả về là null.	
sup()	Trả về chuỗi kiểu superscript	
toLowerCase()	Chuyển chuỗi thành chữ thường	
toUpperCase()	Chuyển chuỗi thành chữ hoa	

Ví dụ: Tính chiều dài chuỗi sử dụng phương thức length

```
<script type="text/javascript">
    var str="Nguyễn Thị Hoa Hồng !"
    document.write("<p>" + str + "</p>")
    document.write("Chiều dài của chuỗi là : "+ str.length)
</script>
```

Ví dụ: Phương thức fontcolor() dùng để định màu của chuỗi

```
<script type="text/javascript">
    var txt="Nguyễn Thị Bảo Nhi "
    document.write("<p>" + txt.fontcolor() + "</p>")
    document.write("<p>" + txt.fontcolor('red') + "</p>")
    document.write("<p>" + txt.fontcolor('blue') + "</p>")
    document.write("<p>" + txt.fontcolor('green') + "</p>")
</script>
```

Ví dụ Phương thức indexOf. Phương thức này trả về vị trí của chuỗi con được tìm thấy trong một chuỗi

```
<script type="text/javascript">
    var str="This is my Schools "
    var pos=str.indexOf("School")
    if (pos>=0)
    {
        document.write("School found at position: ")
        document.write(pos + "<br />")
    }
    Else
    {document.write("School not found!")}
</script>
```

8.1.3.4 Đối tượng Math()

Đối tượng math() cung cấp các hàm và các phương thức cần thiết để thực hiện các phép toán số học. Không cần phải tạo đối tượng Math() mà chúng ta có thể sử dụng trực tiếp đối tượng này

1. Các phương thức của Math():

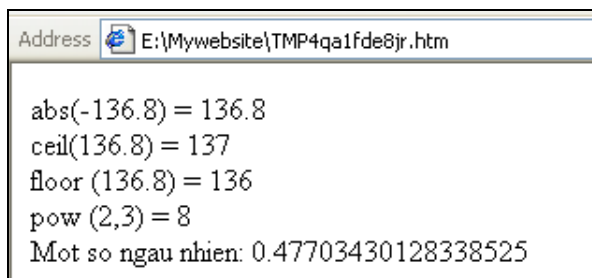
Cú pháp chung:

Math.method([value])

Ví dụ:

```
<script>
    var n= -136.8 , m=136.8
    Document.write("abs(-136.8) = " + Math.abs(n) + "<br>")
    Document.write("ceil(136.8) = " + Math.ceil(m) + "<br>")
    Document.write("floor (136.8) = " + Math.floor(m) + "<br>")
    Document.write("pow (2,3) = " + Math.pow(2,3)+ "<br>")
    Document.write("Một số ngẫu nhiên: " + Math.random()*5 + "<br>")
```


</script>



– Danh sách các phương thức của Math()

Phương thức	Mô tả
abs(x)	Trả về giá trị tuyệt đối của biến x
acos(x)	Trả về giá trị arccosine của x
ceil(x)	Trả về số nguyên lớn hơn hoặc bằng x
floor(x)	Trả về số nguyên nhỏ hơn hoặc bằng x
log(x)	Trả về giá trị log của x
max(x,y)	Trả về giá trị lớn nhất trong hai số x và y
min(x,y)	Trả về giá trị nhỏ nhất trong hai số x và y
pow(x,y)	Trả về giá trị x lũy thừa y
random()	Trả về giá trị một số ngẫu nhiên từ 0 đến 1
round(x)	Làm tròn số x
sqrt(x)	Trả về giá trị căn bậc 2 của x

Ví dụ:

Viết chương trình tạo một nút đổi hình (play) và nút stop để ngưng

<html><head>

```

<script>
var idq;
function play()
{
    var arrhinh= new Array();
    arrhinh[0]= "h1.jpg";
    arrhinh[1]= "h2.jpg";
    arrhinh[2]= "h3.jpg";
    arrhinh[3]= "h4.jpg";
    arrhinh[4]= "h5.jpg";
    arrhinh[5]= "h6.jpg";
    var i=Math.round(Math.random()*6)
    idhinh.src=arrhinh[i];
    idq=setTimeout("play()",1000);
}
function Stop()
{
    clearTimeout(idq);
}
</script>

```

```

</head>
<body>

<form>
    <input type=button value= "Play" onClick= "Play()">
    <input type=button value= "Stop" onClick= "Stop()">
</form></body></html>

```

8.1.3.5 Đối tượng document

Đối tượng document cung cấp các thuộc tính và phương thức để làm việc với toàn bộ tài liệu hiện hành gồm: form, liên kết, hình ảnh, tiêu đề, vị trí hiện hành, màu hiện hành...

Đối tượng document được định nghĩa khi tag body được xử lý trong trang HTML và nó vẫn tồn tại nếu trang được nạp. Các thuộc tính của document phản ánh thuộc tính của tag body. Trong body có 2 sự kiện OnLoad và Unload

1. Các thuộc tính của đối tượng document

Thuộc tính	Mô tả
alinkcolor	Thiết lập hoặc trả về giá trị màu của liên kết đang xem của tài liệu
bgcolor	Thiết lập hoặc trả về giá trị màu nền của tài liệu
cookie	Chứa giá trị các cookies dành cho tài liệu hiện hành
domain	Trả về giá trị tên miền máy chủ chứa document
fgcolor	Thiết lập hoặc trả về giá trị màu chữ của tài liệu
lastmodified	Trả về giá trị ngày giờ cuối cùng mà tài liệu được cập nhật
linkcolor	Thiết lập hoặc trả về giá trị màu của liên kết trong tài liệu
location	mở một trang web mới
referrer	Returns the URL of the document that loaded the current document
title	Trả về giá trị của tựa đề của tài liệu
url	Trả về đường dẫn củ tài liệu hiện hành
vlinkcolor	Thiết lập hoặc trả về giá trị màu của liên kết đã xem của tài liệu

2. Phương thức:

Phương thức	Mô tả
clear()	Xóa tài liệu
close()	Đóng một tài liệu
focus()	Đưa trở về một đối tượng trong trang
open("mimetype",replace)	
write("str")	viết một chuỗi vào một tài liệu
writeln("str")	viết một chuỗi vào một tài liệu và xuống dòng

3. Sự kiện

- Cú pháp:

document.event_name="someJavaScriptCode"

- **Danh sách các sự kiện tác động đối tượng document**

Event
 OnBlur
 OnClick
 OnDblClick
 OnFocus
 OnKeyDown
 OnKeyPress
 OnKeyUp
 OnMouseDown
 OnMouseMove
 OnMouseOut
 OnMouseOver
 OnMouseUp
 OnMouseUp

8.1.4 Đối tượng trình duyệt (Navigator Object)

Đối tượng trình duyệt chứa đựng những thông tin về trình duyệt web của client. Có hai trình duyệt web lớn là Netscape Navigator và Internet Explorer. Mặc dù cả hai đều hỗ trợ mô hình đối tượng trên ngôn ngữ Javascript nhưng cũng có một số đối tượng và cách truy cập vào thành phần thuộc tính trên hai trình duyệt cũng có một cái khác nhau. Muốn cho ứng dụng chạy hoàn chỉnh trên mọi trình duyệt thì người lập trình phải xác định ra chương trình đang chạy trên trình duyệt nào và ở version nào để xử lý đoạn code tốt hơn.

1. Thuộc tính

Thuộc tính	Mô tả
appName	Tên trình duyệt
appVersion	Phiên bản trình duyệt
cookieEnabled	
platform	Nền của hệ điều hành

2. Phương thức

Phương thức	Mô tả
javaEnabled()	Trả về giá trị true nếu trình duyệt có hỗ trợ Javascript
refresh()	
preference()	

8.1.4.1 Đối tượng Window

Là đối tượng cao nhất trong mô hình DOM, là nơi chứa tất cả các thành phần của trang web.

1. Thuộc tính của đối tượng Window:

Thuộc tính	Mô tả	Giá trị
defaultStatus	Thiết lập chuỗi t/báo trên thanh trạng thái	Text
status	Thiết lập thông báo tại thời điểm hiện hành	Text
location	Xác định vị trí trang hiện tại trong cửa sổ	URL
history	Xác định các phần tử trong history	

alwaysLowered	hiển thị cửa sổ bên dưới các cửa sổ khác	Yes/no
alwaysRaised	hiển thị cửa sổ trên tất cả các cửa sổ khác	Yes/no
Dependent	Cửa sổ này sẽ đóng khi cửa sổ cha bị đóng	Yes/no
directories	Hiển thị Button thư mục	Yes/no
fullscreen	hiển thị chế độ đầy màn hình	Yes/no
height	thiết lập chiều cao của cửa sổ	số nguyên
hotkeys	Cho phép dùng phím nóng	Yes/no
left	Thiết lập k/cách từ văn bản đến cạnh cửa sổ	số nguyên
location	hiển thị hộp location	Yes/no
menubar	hiển thị thanh menu bar	Yes/no
resizable	Cho phép thay đổi kích thước cửa sổ	Yes/no
scrollbars	xuất hiện /không xuất hiện thanh cuộn	Yes/no
status	Hiển thị thanh trạng thái	Yes/no
titlebar	hiển thị thanh tiêu đề	Yes/no
toolbar	hiển thị thanh công cụ	Yes/no
width	Xác định độ rộng của cửa sổ	số nguyên
closed	trả về giá trị true, false. True khi cửa sổ đóng	true, false

Ví dụ:

```

window.defaultStatus="String"
window.status="String"
window.location="URL"

```

- Ta cũng có thể mở một trang web mới bằng lệnh:

```

window.location.href="URL"

```

2. Phương Thức

Cú pháp: window.method_name()

Phương Thức	Mô tả
alert("msg")	Hiển Thị hộp thoại thông báo
blur()	Di chuyển con trỏ đến cửa sổ hiện hành
clearInterval(ID)	Hủy thời gian đã thiết lập bằng setInterval()
clearTimeout(ID)	Hủy thời gian đã thiết lập bằng setTimeout()
close()	Đóng cửa sổ hiện hành
confirm("msg")	Hiển thị hộp thoại xác nhận với hai nút Cancel và OK
focus()	Đưa con trỏ về cửa sổ hiện hành
MoveBy(x,y)	Di chuyển cửa sổ đến một vị trí mới một đoạn pixel so với cửa sổ hiện hành
MoveTo(x,y)	Di chuyển cửa sổ qua trái và lên trên một đoạn pixel cụ thể so với cửa sổ hiện hành.
open(URL,"windowname","FeatureList")	URL : là địa chỉ trang web muốn nạp vào cửa sổ.

	WindowName: là tên cửa sổ . FeatureList : là danh sách các thuộc tính của cửa sổ: thanh công cụ, thanh menu, thanh status
print()	in nội dung trong cửa sổ hiện hành.
prompt("msg","reply")	Hiển thị hộp thoại nhập liệu
setTimeout(func,millisec)	Thiết lập thời gian mili giây để gọi một hàm
stop()	Hủy việc download một cửa sổ. Tương tự như việc đóng một cửa sổ trình duyệt.
resizeBy(dx,dy)	Thay đổi kích thước cửa sổ sang phải dx, dưới dy pixel
resizeTo(x,y)	Thay đổi kích thước x, y pixel
scrollBy(dx,dy)	cuộn nội dung sang phải dx, xuống dưới dy pixel
scrollTo(x,y)	cuộn nội dung trên trang đến vị trí x,y

Ví dụ: Objwindow.close()

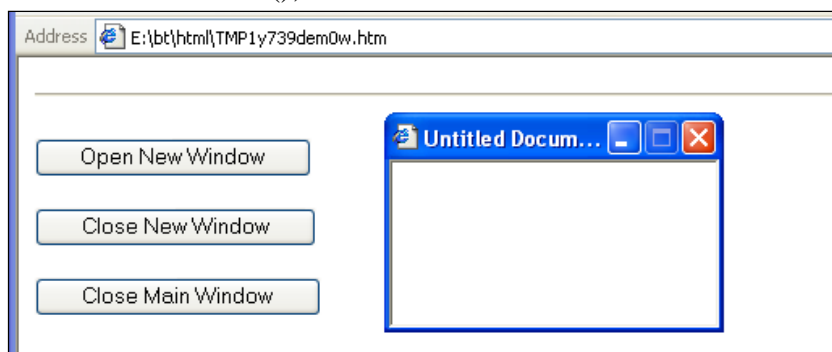
- ❖ **Từ khoá self:** trong trường hợp muốn thao tác trên cửa sổ hiện hành ta dùng từ khoá self thay thế cho đối tượng window

Ví dụ : đóng cửa sổ hiện hành:

Self.close() hoặc window.close()

Ví dụ: Đoạn Script đóng, mở cửa sổ trình duyệt

```
<html>
<head></head>
<body>
  <FORM NAME="winform">
    <INPUT TYPE="button" VALUE="Open New Window"
    onClick="NewWin=window.open('blank1.htm','NewWin',
    'toolbar=no,status=no,width=200,height=100'); ">
    <P><INPUT TYPE="button" VALUE="Close New Window"
    onClick="NewWin.close();" >
    <P><INPUT TYPE="button" VALUE="Close Main Window"
    onClick="window.close();">
  </FORM>
</body>
</html>
```



Ví dụ: Viết hàm mở một cửa sổ mới khi click nút điều khiển

```
function openwindow()
{
    window.open("http://www.yahoo.com","my_new_window",
        "toolbar=yes, location=yes, directories=no, status=no, menubar=yes,
        scrollbars=yes, resizable=no, copyhistory=yes, width=400,
        height=400")
}
```

3. Sự kiện

Sự kiện	Mô tả
onBlur	Thực thi đoạn mã lệnh khi sự kiện blur xảy ra
onError	Thực thi đoạn mã lệnh khi sự kiện Error xảy ra
onFocus	Thực thi đoạn mã lệnh khi sự kiện Focus xảy ra
onLoad	Thực thi đoạn mã lệnh khi sự kiện Load xảy ra
onResize	Thực thi đoạn mã lệnh khi sự kiện resizer xảy ra
onUnload	Thực thi đoạn mã lệnh khi sự kiện Unload xảy ra

Ví dụ: Viết hàm trở tới một trang web khác

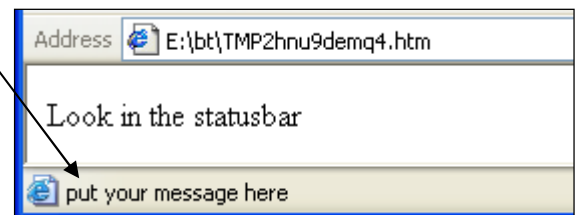
```
function locate()
{
    location="http://www.yahoo.com/"
}
```

Ví dụ:Hiển thị một số thông tin lên thanh trạng thái

```
<head>
    <script type="text/javascript">
        function load()
        { window.status = "put your message here" }
    </script>
</head>
<body onload="load()">
    <p>Look in the statusbar</p>
</body>
```

Ví dụ: Viết hàm in trang web:

```
function printpage()
{
    window.print()
}
```



8.1.4.2 Đối tượng form

Form là một thành phần trên trang web dùng để thu thập dữ liệu, thông tin từ người dùng. Mỗi phần tử trong form là một đối tượng trong DOM. Do đó mỗi phần tử trên form cũng có những sự kiện.

1. Các sự kiện của các phần tử trên form

Phần tử	Tên sự kiện
Button	onClick
Checkbox	onClick

Form	OnSubmit, onReset
Textbox	OnBlur, OnChange, OnFocus, Onselect
Radio	OnClick
Reset button	OnClick
Dropdown menu	OnBlur, onChange, onFocus, onSelect
Submit button	OnClick
Textarea	OnBlur, OnChange, OnFocus, Onselect

2. Truy cập giá trị các phần tử trên form

Cú pháp:

document.formName.formelement.properties

document.formName.formelement.method

Ví dụ 1: Tạo một form chứa một field để nhập địa chỉ email. Kiểm tra dữ liệu nhập vào có phải là địa chỉ E-mail không bằng cách kiểm tra ký tự @ trong địa chỉ nhập vào

```
<html>
<head><Title> Kiem tra</title></head>
<body>
    <form name="form1" method="post" action="">
        <table>
            <tr>
                <td>Enter your E-mail address: </td>
                <td><input type="text" name="MyEmail"></td>
            </tr>
            <tr>
                <td align="center" colspan="2">
                    <input type="submit" name="Submit" value="Send Input"
                    onclick= "validate()">
                </td>
            </tr>
        </table>
    </form>
    <script>
        function validate()
        {
            ad= document.form1.MyEmail.value.indexOf("@")
            if (ad == -1)
            {
                alert("Not a valid e-mail")
                return false
            }
        }
    </script>
</body></html>
```

Ví dụ 2: Kiểm tra tính hợp lệ của giá trị nhập vào textfield

```
<html>
  <head>
    <script>
      function validate()
      {
        txt= document.myForm.myinput.value
        if (txt>=1 && txt<=5)
        {
          return true
        }
        else
        {
          Alert("Must be between 1 and 5")
          return false
        }
      }
    </script>
  </head>
  <body>
    <form name="myForm" onsubmit="return validate()">
      Enter a value from 1 to 5:
      <input type="text" name="myinput">
      <input type="submit" value="Send input">
    </form>
  </body>
</html>
```

Ví dụ 3: Tạo một form chứa 1 field để nhập giá trị. Viết một hàm dùng để kiểm tra số ký tự nhập vào field này (dùng thuộc tính length để kiểm tra số ký tự nhập vào)

```
<html>
  <head>
    <script language="javascript">
      function validateform()
      {
        input= document.myForm.myinput.value
        if (input.length>5)
        {
          alert("Do not insert more than 5 characters")
          return false
        }
        else { return true }
      }
    </script></head>
  <body>
    <form name="myForm" onsubmit="return validateform()">
      in this input box you are not allowed to insert more than 5 characters:
      <input type="text" name="myinput">
      <input type="submit" value="Send input">
    </form>
```



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

3. Các thuộc tính trên đối tượng form:

Thuộc tính	Mô tả	Ví dụ
Action	Trả về đường dẫn (URL) đến tập tin xử lý của form thứ i	Document.forms[i].action
Length	Trả về số form trên trang web Hoặc trả về số phần tử trên form thứ i	Countform=document.forms.length Countfield=document.forms[i].length
Name	Trả về giá trị tên của form thứ i	Nameform=document.forms[i].name
Method	Các định phương thức của form thứ i	Methodform=document.forms[i].method
elements	mảng element chứa các phần tử trên form	document.form[i].elements[j].value

4. Các thuộc tính trên mảng element

Thuộc tính	Mô tả	Ví dụ
Name	Xác định tên của một phần tử trên form thứ i.	document.form[i].elements[j].name
Type	Xác định loại của đối tượng	document.form[i].elements[j].type
Value	Xác định giá trị của phần tử thứ j trong form i.	document.form[i].elements[j].value
Checked	Xác định phần tử thứ j có được checked không. Nếu có trả về giá trị true còn không trả về giá trị false	document.form[i].elements[j].checked
Disable	Thiết lập chế độ mờ (gán giá trị true không cho phép người sử dụng chọn lựa và ngược lại)	document.form[i].elements[j].disable
isDisable	Kiểm tra phần tử có mờ không (true là mờ và ngược lại)	document.form[i].elements[j].isDisable
readOnly	Cho phép/không thay đổi nội dung của phần tử	document.forms[i].elements[j].readOnly

5. Phương thức trên mảng element:

Phương thức	Mô tả	Ví dụ
Focus ()	Đưa con trỏ về lại text box hoặc dropdownmenu	document.form[i].elements[j].focus()

Lưu ý: Nếu ta đang làm việc trên document hiện hành, biết tên cụ thể của form và tên của thành phần trên form ta có thể truy cập trực tiếp mà không cần qua mảng form và element:

nameForm.nameField.property

hoặc

nameForm.nameField.method

Ví dụ 1: về set focus trên một field.

```
<html>
```

```

<head>
    <script type="text/javascript">
        function setfocus()
        {
            document.forms[0].field.focus()
        }
    </script>
</head>
<body>
    <form>
        <input type="text" name="field" size="30">
        <input type="button" value="Get Focus" onclick="setfocus()">
    </form>
</body>
</html>

```

Ví dụ 2: Viết một hàm để đưa con trỏ về textbox đã tạo trước (sử dụng phương thức focus())

```

<html>
    <head>
        <script language="javascript">
            function setfocus()
            {
                document.forms[0].field.select()
                document.forms[0].field.focus()
            }
        </script>
    </head>
    <body>
        <form>
            <input type="text" name="field" size="30" value="input text">
            <input type="button" value="Selected" onclick="setfocus()">
        </form>
    </body>
</html>

```

Các phần tử trên form:

1. Thao tác trên trường radio

Muốn lấy giá trị của trường radio ta phải sử dụng đến mảng element. Duyệt qua tất cả các phần tử và kiểm tra phần tử đó có được checked không ?

Cú pháp:

Countfield=nameform.length

for(var i=0;i<Countfield;i++)

if(nameform.elements[i].type== "radio" &&nameform.elements[i].checked==true)

Giatri=nameform.elements[i].value

Ví dụ 1: Tạo form có chứa các radio:

```

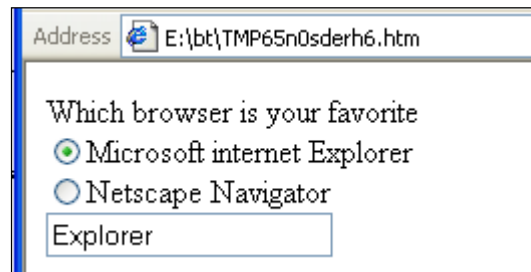
<html>
    <head>

```

```

<script type="text/javascript">
function check(browser)
{
    document.forms[0].answer.value=browser
}
</script>
</head>
<body>
<form>
    Which browser is your favorite<br>
    <input type="radio" name="browser"
    onclick="check(this.value)"
    value="Explorer">Microsoft internet Explorer<br>
    <input type="radio" name="browser"
    onclick="check(this.value)"
    value="Netscape">Netscape Navigator<br>
    <input type="text" name="answer">
</form></body></html>

```



Ví dụ 2: Thao tác trên trường checkbox

```

<html>
<head>
<script type="text/javascript">
function check()
{
    coffee=document.forms[0].coffee
    answer=document.forms[0].answer
    txt=""
    for (i = 0; i<coffee.length; ++ i)
    {
        if (coffee[i].checked)
        {txt=txt + coffee[i].value + " "}
    }
    answer.value="You ordered a coffee with " + txt
}
</script>
</head>
<body>
<form>
    How would you like your coffee?<br>
    <input type="checkbox" name="coffee" value="cream">With
    cream<br>
    <input type="checkbox" name="coffee" value="sugar">With sugar<br>

```

```

<input type="text" name="answer" size="30">
<input type="button" name="test" onclick="check()" value="Order">
</form>
</body>
</html>

```

2. Thao tác trên dropdownmenu

Các thuộc tính và phương thức của dropdownmenu

a) Thuộc tính

Thuộc tính	Mô tả	Ví dụ
length	Trả về số phần tử trong danh sách dropdownmenu.	spt=nameform.namefield.length
selectedIndex	Trả về chỉ số của phần tử được chọn trong danh sách	spt=nameform.namefield.selectedIndex
options	mảng option chứa các phần tử trong danh sách được đánh chỉ số 0->spt-1	

b) Phương thức

Phương thức	Mô tả	Ví dụ
Focus()	Đưa con trỏ về lại dropdownmenu	nameform.namefield.focus()

c) Các thuộc tính của mảng option

Thuộc tính	Mô tả	Ví dụ
DefaultSelected	Trả về giá trị true nếu phần tử thứ i được chọn	nameform.namefield.option[i].defaultSelected
Selected	Trả về giá trị true nếu phần tử thứ i được chọn	nameform.namefield.option[i].selected
Value	Trả về giá trị value của option thứ i.	nameform.namefield.option[i].value
Text	Trả về giá trị text của option thứ i.	nameform.namefield.option[i].text

Ví dụ 1:

```

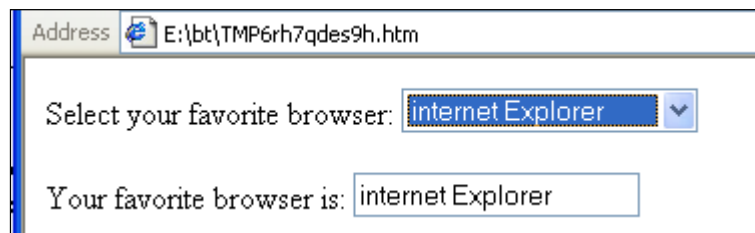
<html>
<head>
<script type="text/javascript">
function put()

```

```

    {
        txt=document.forms[0].dropdown.options[document.forms[0].
        dropdown.selectedIndex].text
        document.forms[0].favorite.value=txt
    }
</script>
</head>
<body>
    <form>
        Select your favorite browser:
        <select name="dropdown" onchange="put()">
            <option>internet Explorer
            <option>Netscape Navigator
        </select>
        <p>
        Your favorite browser is:
        <input type="text"
        name="favorite" value="internet Explorer">
    </form>
</body>
</html>

```

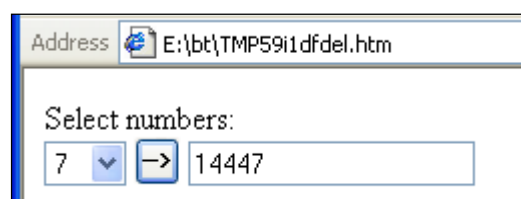


Ví dụ 2 Tạo form có dropdown menu

```

<html>
<head>
    <script type="text/javascript">
        function put()
        {
            option=document.forms[0].dropdown.options[document.forms[0].
            dropdown.selectedIndex].text
            txt=document.forms[0].number.value
            txt=txt + option
            document.forms[0].number.value=txt
        }
    </script>
</head>
<body>
    <form>
        Select numbers:<br>
        <select name="dropdown">
            <option>1
            <option>2
            <option>3
            <option>4
            <option>5

```

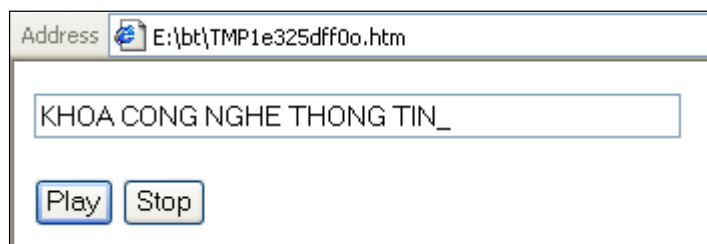


```

        <option>6
        <option>7
        <option>8
        <option>9
        <option>10
    </select>
    <input type="button" onclick="put()" value="--"> <input type="text"
    name="number">
</form>
</body>
</html>
Ví dụ 3:
<html>
<head></head>
<body>
    <script language="JavaScript">
        var max=0;
        function textlist()
        {
            max=textlist.arguments.length;
            for (i=0; i<max; i++)
                this[i]=textlist.arguments[i];
        }
        tl=new textlist("KHOA CONG NGHE THONG TIN", "TRUONG DAI HOC
        CONG NGHIEP TPHCM", "SO 12 NGUYEN VAN BAO ", "DIEN THOAI:
        8940390")
        var x=0; pos=0;
        var l=tl[0].length;
        function textticker()
        {
            document.tickform.tickfield.value=tl[x].substring(0,pos)+"_";
            if(pos++==l)
            {
                pos=0; setTimeout("textticker()",1000);
                x++;
                if(x==max) x=0; l=tl[x].length;
            }
            else
                iq=setTimeout("textticker()",50);
        }
        function stop()
        {
            clearTimeout(iq)
        }
    </script>
    <form action="" method="post" name="tickform" id="tickform">
        <input name="tickfield" type="text" id="tickfield" size="50">
        <p>
            <input type="button" name="Button" value="Play" onClick="textticker()">
            <input type="button" name="Button" value="Stop" onClick="stop()">

```

```
</p>
</form>
</body>
</html>
```



8.1.5 THAY ĐỔI NỘI DUNG ĐỘNG TRÊN TRANG

Trên Firefox:

Để thay đổi nội dung động trên trang dùng cặp tag <Layer> </Layer>

```
<Layer ID=IdName properties>
    Document content
</Layer>
```

ID :là tên của Layer, dựa vào IdName để thay đổi nội dung động trên trang,

Properties: là danh sách các thuộc tính của Layer như xác định vị trí xuất hiện hay liên kết đến trang web nào đó.

Danh sách các thuộc tính:

Thuộc tính	Mô tả
Clip=top_x, left_y,bottom_x,right_y	Xác định tọa độ xuất hiện của layer trong cửa sổ
Height=value	Xác định chiều cao của layer
Left=value	Qui định khoảng trắng trái từ văn bản đến layer
PageX=value	Xác định khoảng cách của layer so với cạnh trên của cửa sổ
PageY=value	Xác định khoảng cách của layer so với cạnh bên của cửa sổ
SRC=URL	Xác định tập tin nạp vào layer
Top=value	Qui định khoảng trắng trên từ văn bản đến layer
Visibility=option(Hide show)	Xác định layer xuất hiện (show)hay không xuất hiện (hide)
Width=value	Xác định chiều rộng layer
Z-index=value	Vị trí tương đối của layer so với các phần tử khác

Ví dụ:

```
<layer ID=Idlayer SRC="filename.htm"></layer>
```

Trong một trang ta có thể viết một hoặc nhiều layer. Như vậy dựa vào thuộc tính SRC để có thể nạp vào một hoặc nhiều trang web khác nhau .

Trên Internet Explorer

Tag <Iframe></Iframe> trên Internet Explorer, có công dụng giống như Layer trên NetsCape.

```
<Iframe Id=IdName properties>
</Iframe>
```

Các thuộc tính của Iframe

Thuộc tính	Mô tả
Align={left,center,right}	Xác định vị trí xuất hiện của của iframe
frameborder	Xác định đường viền
Height=value	Xác định chiều cao
SRC	Xác định địa chỉ trang web nạp vào iframe
width	Xác định chiều rộng của iframe

- Nếu như nội dung trong iframe quá lớn thì tự nó tự động xuất hiện thanh trượt để cuộn nội dung.

- Viết nội dung vào iframe

Idname.document.write(content)

- Thay đổi nội dung trên trang iframe

Document.all.idName.SRC="URL"

Ví dụ :

```
<iframe id=page frameborder=0 src= "page2.htm"></iframe>
```

Thay đổi nội dung trên trang dựa vào inner và outer

Ta dùng đặc tính inner và outer để thay đổi nội dung hoặc lấy giá trị của một vùng nào đó trên trang web.

1. Phân biệt giữa inner và outer

- Inner là những gì chứa bên trong của đối tượng chứa ID. Inner gồm có
 - InnerHTML lấy nội dung text và tag HTML bên trong đối tượng ID
 - innerText: chỉ lấy nội dung text bên trong đối tượng ID
- Outer là phần inner và bản thân đối tượng chứa ID. Outer gồm có
 - outerHTML lấy nội dung text và tag HTML của cả đối tượng ID
 - outerText : lấy nội dung text

Ví dụ:

```
<Div ID=Intro>Monitor<B> SAMSUNG</B></Div>
```

2. Lấy dữ liệu từ một đối tượng

- Nếu lấy dữ liệu từ một vùng nào đó thì innerText và outerText hoàn toàn giống nhau, chỉ lấy phần văn bản không lấy các thẻ HTML bao quanh chúng

Ví dụ:

```
Var s1,s2
```

```
s1=Intro.outerText
```

```
s2=Intro.innerText
```

thì s1 và s2 đều nhận giá trị Monitor SAMSUNG

Ví dụ

```
s1=Intro.outerHTML
```

```
s2=Intro.innerHTML
```

Thì s1=<Div Id=Intro>Monitor SAMSUNG</Div>

Và s2=Monitor SAMSUNG

- Trong trường hợp ta viết nội dung nó ra ngoài thì nó sẽ định dạng những thẻ HTML chứa trong phần nội dung

Ví dụ:

Intro.innerHTML="<I>CPU Pentium IV"</I>

Khi đó vùng Intro sẽ được thay thế bằng chuỗi *CPU Pentium IV* Chứa trong thẻ <DIV>

<Div Id=Intro>CPU Pentium IV</Div>

Intro.outerHTML="<I>CPU Pentium IV"</I>

Khi đó vùng Intro sẽ được thay thế bằng chuỗi *CPU Pentium IV* chứa trong thẻ <DIV> đồng thời thẻ <DIV> không còn trên trang. Do đó nếu chương trình tiếp tục xử lý các lệnh còn liên quan đến ID có tên là Intro thì nó sẽ báo lỗi. Vì chuỗi *CPU Pentium IV* thay thế cho <Div Id=Intro>MonitorSAMSUNG</Div>

Ví dụ: Dùng Inner Outer

```
<html>
  <head><title>Untitled Document</title>
    <script language="JavaScript">
      var count;
      count=1;
      function Add()
      {
        stt.innerText=count;
        ma.innerText=form1.msv.value;
        ten.innerText=form1.tsv.value;
        dc.innerText=form1.dcsv.value;

        r.id="r" + count;
        stt.id="stt"+count;
        ma.id="ma"+count;
        ten.id="ten"+count;
        dc.id="dc"+count;

        pg=t.innerHTML
        pd='<table id=t border="1" cellspacing="1" style=" BORDER-
        COLLAPSE: collapse" align="center" width=80%>'
        pt='<tr id=r>'
        pt=pt+'<td id=stt width="14%"></td>'
        pt=pt+'<td id=ma width="26%"></td>'
        pt=pt+'<td id=ten width="30%"></td>'
        pt=pt+'<td id=dc width="30%"></td>'
        pt=pt+'</tr>'
        pc="</table>"
        t.outerHTML=pd+pg+pt+pc
        count++
      }
    </script>
  </head>

  <body>
    <form name="form1" method="post" action="">
      <table width="60%" border="1" align="center" cellpadding="0"
        cellspacing="0">
```

```

        <tr>
            <td width="30%">MA SVIEN</td>
            <td width="70%"><input name="msv" type="text"
            id="msv" SIZE=30></td>
        </tr>
        <tr>
            <td>HO TEN SV</td>
            <td><input name="tsv" type="text" id="tsv"
            SIZE=30></td>
        </tr>
        <tr>
            <td>DIA CHI</td>
            <td><input name="dcsv" type="text" id="dcsv"
            SIZE=30></td>
        </tr>
        <tr>
            <td colspan="2"><div align="center">
                <input type="button" name="Button" value="Add"
                onClick="Add()"></div>
            </td>
        </tr>
    </table>
</form>
<table id="t" width="75%" border="1" align="center" cellpadding="0"
cellspacing="0" align="center">
    <tr bgcolor="#FFCCCC">
        <td WIDTH=14%><div align="center">STT</div></td>
        <td WIDTH=26%><div align="center">MA SVIEN</div></td>
        <td WIDTH=30%><div align="center">HO TEN</div></td>
        <td WIDTH=50%><div align="center">DIA CHI</div></td>
    </tr>
    <tr id="r" bgcolor="#FFCCCC">
        <td id="stt" width="14%"></td>
        <td id="ma" width="26%"></td>
        <td id="ten" width="30%"></td>
        <td id="dc" width="50%"></td>
    </tr>
</table>
</body>
</html>

```

MA SVIEN	A02
HO TEN SV	Nguyen Nam
DIA CHI	Le Loi
Add	

STT	MA SVIEN	HO TEN	DIA CHI
1	A01	Nguyen Hoang	2- Bui thi Xuan
2	A02	Nguyen Nam	Le Loi

Ví dụ tham khảo : Thiết kế form bán vé tàu

<HTML>

<HEAD><TITLE></TITLE>

<script>

```
var t1="",t2="",t3="",t4="";
```

```
var objw;
```

```
focus();
```

```
function nhap()
```

```
{
```

```
var i;
```

```
if(DK.T1.value=="")
```

```
{
```

```
    window.showModalDialog("massege.htm","Phai nhap vao ho ten","status=no;help=no;scrollbar=no")
```

```
    DK.T1.focus();
```

```
    return;
```

```
}
```

```
if(DK.D1.options[DK.D1.selectedIndex].text==DK.D2.options[DK.D2.selectedIndex].text)
```

```
{
```

```
    window.showModalDialog("massege.htm","Noi di khong duoc trung noi den","status=no;help=no;scrollbar=no");
```

```
    return;
```

```
}
```

```
if(DK.T2.value=="")
```

```
{
```

```
    window.showModalDialog("massege.htm","Phai nhap vao gia tien","status=no; help=no;scrollbar=no")
```

```
    DK.T2.focus();
```

```
    return;
```

```
}
```

```
if (isNaN(DK.T2.value )==true )
```

```
{
```

```

        window.showModalDialog("massege.htm","Gia tri phai co kieu so",
        "status=no; help=no;scrollbar=no");
        DK.T2.value="";
        DK.T2.focus();
        return;
    }
    objw=window.open("danhsachdangky.htm","DanhSachDangKy")
    t1= t1 + DK.T1.value + "<br>" ;
    objw.c1.innerHTML = t1
    i=DK.D1.selectedIndex ;
    t2=t2+DK.D1.options[i].text+"<br>";
    objw.c2.innerHTML=t2;

    i=DK.D2.selectedIndex ;
    t3=t3+DK.D2.options[i].text+"<br>";
    objw.c3.innerHTML=t3;
    gia= eval(DK.T2.value);
    if(DK.co.checked)
    {
        t4 = t4 + gia*2*0.8 + "<br>"
        objw.c4.innerHTML= t4
    }
    else
    {
        t4 = t4 + gia + "<br>"
        objw.c4.innerHTML= t4
    }
    blur();
    objw.focus();
}
</script>
</HEAD>
<BODY>
<FORM method="post" name="DK" >
<TABLE border="1" width="79%">
<TR>
<TH colspan="2">
<font size="5" face="Arial, Helvetica, sans-serif">
DANG KY VE TAU
</font>
</TH>
</TR>
<TR>
<TD width="54%">Ho ten khách hàng: </TD>
<TD width="46%"><INPUT name="T1" ></TD>
</TR>

<TR>
<TD width="54%">Noi di: </TD>

```

