

HỒ HOÀNG TRIẾT

CĂN BẢN THIẾT KẾ

WEB

HTML

JAVASCRIPT

ASP

NHÀ XUẤT BẢN THỐNG KÊ

HỒ HOÀNG TRIẾT

CĂN BẢN THIẾT KẾ

WEB

**HTML
JAVASCRIPT
ASP**

NHÀ XUẤT BẢN THỐNG KÊ

Mục Lục

MỤC LỤC.....	5
PHẦN I HTML HYPER TEXT MARKUP LANGUAGE	9
BẮT ĐẦU VỚI HTML.....	9
<i>Bạn cần có gì để bắt đầu với HTML.....</i>	<i>9</i>
SỬ DỤNG CÁC PHƯƠNG TIỆN	9
<i>Sử dụng Notepad và trình duyệt web.....</i>	<i>9</i>
CĂN BẢN ĐỊNH DẠNG HTML.....	14
<i>Giới thiệu các tag HTML và định dạng cơ bản của một trang HTML</i>	<i>14</i>
<i>Cách dùng các tag HTML để vận dụng với văn bản.....</i>	<i>17</i>
<i>Sử dụng các đề mục, các đoạn văn bản, và ngắt dòng.....</i>	<i>19</i>
<i>Cách thay đổi kích thước và màu sắc văn bản.....</i>	<i>24</i>
<i>Cách dùng các ký tự đặc biệt trong HTML.....</i>	<i>27</i>
<i>Cách liên kết với các trang khác trong HTML.....</i>	<i>30</i>
<i>Cách đặt các hình ảnh vào trang web của bạn.....</i>	<i>32</i>
<i>Thử dùng một số tag văn bản khác.....</i>	<i>34</i>
<i>Sử dụng ghi chú (comment) trong code html.....</i>	<i>38</i>
<i>Cách thêm danh sách (list) vào trang web của bạn.....</i>	<i>39</i>
<i>Cách dùng các thuộc tính của tag BODY để làm đẹp trang web.....</i>	<i>42</i>
<i>Cách thay đổi màu mặc nhiên của phông chữ.....</i>	<i>43</i>
<i>Cách thay đổi màu của link trong trang web của bạn.....</i>	<i>44</i>
<i>Thêm hoặc thay đổi màu nền của trang web.....</i>	<i>45</i>
<i>Cách cho một hình ảnh làm hình ảnh nền của trang web.....</i>	<i>46</i>
<i>Cách định lại kích thước của hình ảnh.....</i>	<i>47</i>
<i>Làm thế nào để link dùng một hình ảnh.....</i>	<i>50</i>
<i>Cách xuống dòng tự động (wrap) văn bản bao quanh hình ảnh.....</i>	<i>52</i>
<i>Một thuộc tính (alternative - alt) khác của hình ảnh.....</i>	<i>53</i>
<i>Nhảy đến đầu trang, cuối trang hay một vị trí bất kỳ.....</i>	<i>55</i>
<i>Cách để tạo một email link.....</i>	<i>55</i>
<i>Các mẹo khác với email link.....</i>	<i>58</i>

Căn bản thiết kế Web

<i>Cách thêm các form vào trang web.</i>	60
<i>Cách để chơi nhạc và nghe âm thanh.</i>	68
<i>Dùng META tag để các search engine thấy trang web của bạn.</i>	71
<i>Cách tự động chuyển hướng người sử dụng.</i>	73
<i>Sử dụng đường gạch ngang <HR></i>	74
<i>Sử dụng các phông chữ khác nhau trên trang web.</i>	77
<i>Cách tạo một link mở một cửa sổ trình duyệt mới.</i>	78
<i>Cách cho một ảnh nền cố định vào trang web.</i>	79
<i>Cách thêm một marquee (dòng chữ chạy) vào trang web.</i>	80
BẢNG TRONG HTML	83
<i>Cách cho một bảng (table) vào trang web.</i>	83
<i>Cách dùng các thuộc tính khác của bảng</i>	90
<i>Đặt các bảng trong một bảng lớn hơn</i>	97
<i>Canh lề các bảng với một ảnh ẩn</i>	99
<i>Dùng các bảng lồng nhau thay vì dùng mẹo hình ảnh.</i>	101
SỬ DỤNG CÁC FRAME	103
<i>Cách thêm các khung (frame) vào trang web.</i>	103
<i>Cách liên kết giữa các frame, thay đổi thuộc tính của frame, và sử dụng tag noframes.</i>	107
<i>Làm cho các thanh cuộn và các đường viền biến mất.</i>	110
<i>Cách chèn frame nội (inline) trong trang HTML.</i>	111
<i>Cách cho một image map vào trang web.</i>	113
<i>Cách bỏ các đường gạch dưới của các link</i>	115
PHẦN II JAVASCRIPT	118
CẦN CÓ GÌ ĐỂ SỬ DỤNG JAVASCRIPT	118
PHẦN CĂN BẢN JAVASCRIPT	118
<i>Các căn bản để chèn JavaScript vào trang Web.</i>	118
<i>Cách thêm JavaScript đầu tiên vào trang web</i>	120
<i>Cách làm một nút bấm (button) phản ứng với JavaScript</i>	122
<i>Tạo các nút tương tác với history.</i>	126
<i>Thông báo, cảnh báo</i>	127
<i>Thêm các biến và các hàm vào trong các script.</i>	131
<i>Các toán tử logic và các câu lệnh có điều kiện.</i>	137

<i>JavaScript Prompts</i>	140
<i>Các hộp thoại xác nhận</i>	143
<i>Lấy thông tin trình duyệt</i>	146
<i>Chuyển hướng - Redirection</i>	150
<i>Cách dùng JavaScript để mở một cửa sổ mới</i>	154
<i>Làm thế nào để gọi hàm mà không dùng các nút bấm</i>	161
PHẦN JAVASCRIPT NÂNG CAO	162
<i>Hàm setTimeout</i>	162
<i>Sử dụng mảng trong JavaScript</i>	165
<i>Sử dụng mảng liên hợp trong JavaScript</i>	169
<i>Tải trước các hình ảnh</i>	170
<i>Thay đổi hình ảnh khi di chuyển chuột qua nó</i>	172
<i>Làm hiệu ứng với nhiều hình ảnh</i>	178
<i>Tạo hiệu ứng cho một ảnh khác</i>	179
<i>Kiểm tra các đối tượng thay vì kiểm tra trình duyệt</i>	187
FRAME VÀ JAVASCRIPT	188
<i>Kết hợp dùng JavaScript và Frame</i>	188
<i>Thay đổi hai Frame cùng lúc</i>	191
<i>Soạn thảo HTML cho frame bên trái</i>	192
LÀM VIỆC VỚI FORM	195
<i>Cách xử lý các phần tử của form</i>	195
<i>Dùng các hộp danh sách để di đến trang khác</i>	198
<i>Dùng hộp danh sách không cần nhấn nút</i>	200
<i>Sử dụng các hộp danh sách với các frame</i>	203
<i>Tạo các hình ảnh "nhấn được" giống như button của form</i>	206
<i>Cách làm cho một file JavaScript làm việc trên nhiều trang html</i>	210
ĐỐI TƯỢNG NGÀY THÁNG (DATE OBJECT)	212
<i>Sử dụng date object và các phương thức (method) của nó để tạo một đồng hồ</i>	212
<i>Thêm một vài tính năng mới cho đồng hồ</i>	218
ĐỐI TƯỢNG TOÁN HỌC (MATH OBJECT)	221
<i>Các phương thức hữu dụng cho việc tính toán</i>	221
<i>Lấy các số và các chuỗi ngẫu nhiên</i>	223

XỬ LÝ CHUỖI	226
<i>Sử dụng charAt và indexOf</i>	<i>226</i>
<i>Sử dụng phương thức split</i>	<i>231</i>
PHẦN III ASP ACTIVE SERVER PAGES	234
<i>Active Server Pages là gì?</i>	<i>234</i>
<i>Bạn có thể làm gì với Active Server Pages?</i>	<i>235</i>
<i>Trang ASP trông như thế nào?</i>	<i>235</i>
<i>Các server-side script trông như thế nào?</i>	<i>235</i>
<i>Hiển thị ngày giờ hiện hành</i>	<i>236</i>
<i>Hiển thị văn bản</i>	<i>241</i>
<i>Sử dụng các hàm ngày tháng, thời gian và văn bản</i>	<i>244</i>
<i>Sử dụng biến, và form trong Active Server Pages</i>	<i>248</i>
<i>If...Then...Else</i>	<i>250</i>
<i>For...Next</i>	<i>251</i>
<i>Do While...Loop</i>	<i>252</i>
<i>Do Until...Loop</i>	<i>252</i>
<i>Select Case...End Select</i>	<i>254</i>
<i>Viết các thủ tục</i>	<i>258</i>
<i>Include/virtual</i>	<i>260</i>
<i>Phương pháp Session</i>	<i>263</i>
<i>Phương pháp Application</i>	<i>266</i>
<i>Đối tượng Dictionary</i>	<i>270</i>
<i>Cookies</i>	<i>276</i>
<i>Mở file và đọc nội dung từ một file văn bản</i>	<i>277</i>
<i>Tạo và ghi vào file văn bản</i>	<i>280</i>
<i>Global.asa</i>	<i>282</i>
<i>Các hàm và các thủ tục</i>	<i>284</i>
<i>Các toán tử</i>	<i>287</i>

Phần I HTML

Hyper Text Markup Language

Bắt đầu với HTML

Bạn cần có gì để bắt đầu với HTML

Để bắt đầu dùng HTML, bạn cần có một số thứ. Sau đây là danh sách các phần cần có:

1. Cần có một trình soạn thảo văn bản. Nếu sử dụng hệ điều hành Windows, *Notepad* là đủ.
2. Cần có một trình duyệt. Nếu có khả năng, dùng Netscape Navigator hay Microsoft Explorer phiên bản mới nhất để hiển thị các trang HTML.

Sử dụng các phương tiện

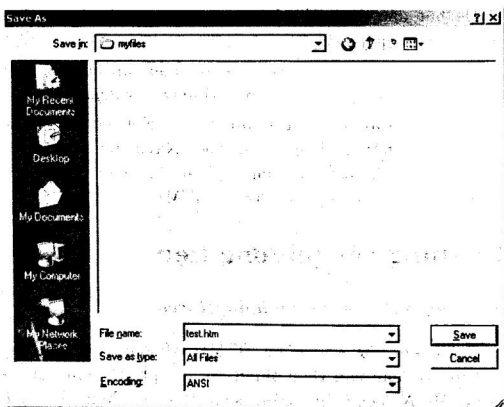
Sử dụng Notepad và trình duyệt web.

Trước khi bắt đầu dùng HTML (Hyper Text Markup Language), bạn cần thân thiện với các phần mềm bạn sẽ dùng. Mở *Notepad* (hay trình soạn thảo của bạn). Bây giờ

nhập vào các dòng code sau vào *Notepad*. Chúng ta sẽ dùng nó với trình duyệt sau này.

```
<HTML>
<HEAD>
<TITLE>Test Page</TITLE>
</HEAD>
<BODY>
Hi there, you have just written your first web page!
</BODY>
</HTML>
```

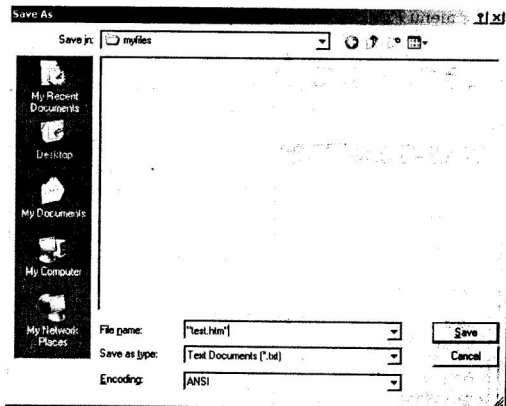
Sau khi nhập vào xong, chọn menu “File” và nhấn “Save As”. Thao tác này sẽ hỏi bạn tên bạn đặt cho file. Nhập vào *test.htm*. Tại vị trí cuối của hộp thoại bạn thấy một hộp danh sách “Save file as Type” hay “Save as Type”. Giá trị mặc nhiên là *.txt*. Bạn chọn *All File (*.*)*. Có thể bạn thấy tương tự như hình bên dưới:



Căn bản thiết kế Web

Bạn có thể chọn ổ đĩa và thư mục bạn muốn lưu trữ vào (ví dụ: `c:\myfiles`), và nhấn “Save”. Bạn làm tương tự như trên cho các file HTML bạn sẽ tạo sau này.

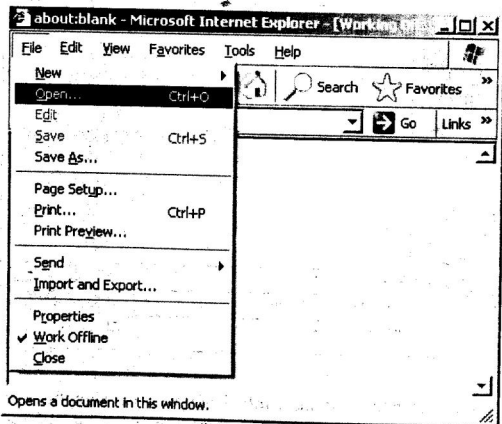
Trong trường hợp trong hộp thoại không có *All Files(*.*)*, hệ điều hành lưu thành *test.htm.txt*. Để tránh trường hợp này, bạn nên đặt tên file trong dấu ngoặc kép “*test.htm*”, như sau:



Bây giờ bạn sẽ dùng trình duyệt web để xem file bạn vừa tạo. Chạy trình duyệt. Nếu trình duyệt hỏi kết nối vào Internet, nhấn “Cancel” và bạn sẽ thấy một trang trắng. Hay cũng có thể thấy một trang thông báo lỗi trình duyệt không thể kết nối vào <http://somepage.htm> hoặc cái gì đó

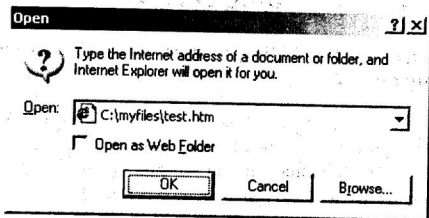
tương tự. Bạn có thể nhấn OK để nhận được một trang trắng. Trong hộp nhập địa chỉ, nhập vào đường dẫn của file html, ví dụ: e:\mypages\test.htm. Sau đó nhấn ENTER. Trình duyệt sẽ hiển thị trang HTML này lên.

Một cách khác là dùng hộp thoại *Open* thay vì nhập vào địa chỉ. Chọn menu "*File*" trên trình duyệt, chọn "*Open*". Hộp thoại *Open* mở ra, bạn tìm và chọn file *test.htm* và nhấn *Open*. Hình bên dưới cho thấy mục *Open* trong menu *File* của Internet Explorer:

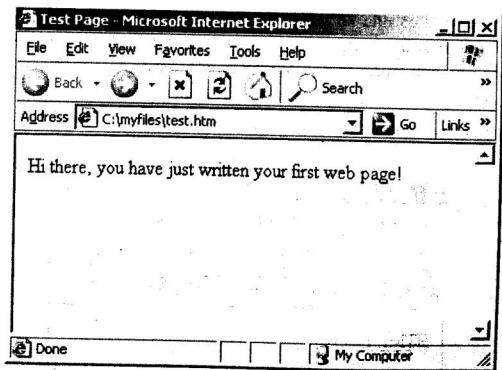


Căn bản thiết kế Web

Sau khi chọn mục Open:



Sau khi mở file html này trên trình duyệt, bạn sẽ thấy như sau:



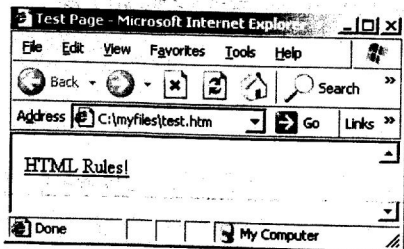
Căn bản định dạng HTML

Giới thiệu các tag HTML và định dạng cơ bản của một trang HTML

Đến đây ta bắt đầu tìm hiểu về các thẻ HTML (HTML tag). Một tag (thẻ) luôn luôn bắt đầu với dấu nhỏ hơn, như sau: <. Các tag kết thúc với dấu lớn hơn, >. Một ví dụ về tag dùng để gạch dưới văn bản, <U>. Bạn đặt tag này trước dòng văn bản bạn muốn gạch dưới. Tag này gọi là tag mở, dùng nó để bắt đầu một hành động mà bạn muốn thực hiện. Để kết thúc việc gạch dưới văn bản, bạn phải dùng một tag đóng. Một tag đóng thường là giống với tag mở, nhưng có dấu sổ đứng trước tên lệnh, ví dụ: </U>. Do đó, nếu bạn muốn gạch dưới cụm từ "HTML Rules!", bạn sẽ viết như sau trong trình soạn thảo văn bản:

<U>HTML Rules!</U>

Kết quả xuất hiện trên trình duyệt như sau:



Không phải tất cả các tag đều bắt buộc có một tag đóng. Ví dụ như là tag hình ảnh, dùng để đặt một hình ảnh lên trang HTML. Nó giống như sau: `<IMG SRC="myimage.gif">`. Một số ví dụ khác như là tag xuống dòng: `<BR>`, đường ngang: `<HR>`, và tag đoạn văn bản: `<P>`.

Bạn không cần phải viết hoa các tag. Tag `<P>` và `<p>` là như nhau. Và bạn có thể dùng bao nhiêu khoảng trống cũng được. Ví dụ:

```
<U> Underline Me! </U>
```

tương đương với:

```
<U> Underline Me! </U>
```

và cũng tương đương với:

```
<U>
Underline Me!
</U>
```

Một file HTML cơ bản sẽ có định dạng như bên dưới:

```
<HTML>
<HEAD>
<TITLE>I Love HTML</TITLE>
</HEAD>
<BODY>
Everything displayed on your page will be in here.
</BODY>
</HTML>
```

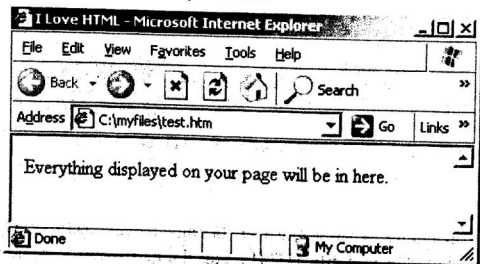
Cái chúng ta thấy đầu tiên là tag `<HTML>`. Đây là dấu hiệu bắt đầu một file HTML. Tag đóng, `</HTML>`, nằm ở vị trí cuối của file HTML. Bạn cũng có thể đoán ra đây là

dấu hiệu kết thúc tài liệu HTML (*HTML document*). Mọi thứ còn lại (các tag, văn bản..) sẽ nằm giữa hai tag này.

Tag tiếp theo là tag <HEAD>. Nó mở ra một phần để bạn đặt tựa đề của trang web, sử dụng các từ khoá, và thêm các thông tin mô tả của trang web. Phần này kết thúc bằng tag </HEAD>. Trong ví dụ trên, phần HEAD chỉ chứa TITLE.

Tag <TITLE> cho phép bạn tạo tựa đề cho trang Web. Các tựa đề này chỉ được sử dụng cho các động cơ tìm kiếm, bookmark, và tựa đề được hiển thị trên thanh tiêu đề của cửa sổ trình duyệt. Để kết thúc tựa đề, dùng tag </TITLE>. Như trong ví dụ trên, tựa đề là "I Love HTML".

Tag <BODY> mở phần thân của trang Web, phần này sẽ được hiển thị trên trình duyệt web. Hầu hết các công việc được làm ở đây. Để kết thúc phần thân, dùng tag </BODY>. Ví dụ trên tạo một trang web rất đơn giản. Trình duyệt sẽ hiển thị như sau:



Tất cả những gì ta có là một dòng văn bản được canh lề bên trái màn hình.

Cách dùng các tag HTML để vận dụng với văn bản

Bây giờ là lúc làm cho các dòng văn bản xuất hiện với nhiều kiểu dáng khác nhau. Hãy bắt đầu với một số tag như sau:

`<B></B>` tag dùng để in đậm (bold) văn bản.
ví dụ:

`<B>Howdy</B>`

Dòng này hiển thị trên trang web như sau:

Howdy

Và đây là một số tag khác:

`<U></U>` Gạch dưới (Underline) văn bản.

`<U>Underline Me!</U>`

Underline Me!

`<I></I>` in nghiêng (Italic) văn bản.

`<I>Isn't this fun?</I>`

Isn't this fun?

`<STRIKE></STRIKE>` gạch giữa.

`<STRIKE>You're Out!</STRIKE>`

~~You're Out!~~

`<CENTER></CENTER>` canh chính giữa

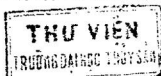
`<CENTER>This centers text on the page</CENTER>`

This centers text on the page

Bạn có thể kết hợp dùng nhiều hơn một tag cùng một lúc. Giả sử bạn muốn văn bản in đậm và in nghiêng, chỉ cần đặt cả hai tag trước văn bản và nhớ đóng chúng với hai tag, tương tự như sau:

`<B><I>I am bold AND Italic, which makes me cool!</I></B>`

08006



Sẽ hiển thị dòng giống như sau:

I am bold AND Italic, which makes me cool!

Thứ tự của các tag có ảnh hưởng đến kết quả hay không? Trong trường hợp này, không quan trọng về cách bạn mở và đóng các tag. Tuy nhiên, làm việc theo thứ tự từ trong ra ngoài sẽ giúp bạn thấy code tốt hơn, và cũng giúp bạn tránh được các trường hợp gây ra lỗi do không đúng thứ tự tag. Sau đây là một cách khác để làm việc theo thứ tự trong ra ngoài. Ta có thể viết HTML theo cách sau:

```
<B>  
  <I>  
    I am bold AND Italic, which makes me cool!  
  </I>  
</B>
```

Đoạn code trên dài dòng. Bạn chỉ cần nhớ là văn bản bạn viết ra bị ảnh hưởng bởi mọi tag đứng trước nó mà chưa được đóng. Ảnh hưởng này chỉ kết thúc khi từng tag mở được đóng bởi tag đóng của nó.

Hãy thử với một dòng văn bản: đậm, nghiêng và gạch dưới!

```
<B><I><U>Would you stop tagging me!</B></I></U>
```

Dòng trên cho kết quả như sau:

Would you stop tagging me!

Nhưng dòng này:

```
<U><I><B>Would you stop</B></I>tagging me!</U>
```

sẽ cho kết quả là:

Would you stop tagging me!

Như bạn thấy, các tag *bold* và *italic* đã được đóng trước từ "tagging" nhưng tag *underline* vẫn còn mở cho đến dấu

chấm than. Điều này làm cho "*tagging me!*" vẫn còn gạch dưới, và không chịu ảnh hưởng các tag *bold* và *italic*.

Bây giờ thử dùng tag CENTER. Bởi vì lẽ mặc nhiên của mọi thứ là bên trái, do đó nếu bạn muốn đặt cái gì đó ở giữa trang web, dùng tag <CENTER>. Mọi thứ bạn đặt giữa <CENTER> và </CENTER> sẽ được canh giữa trang web. Đây là một ví dụ:

```
<CENTER>I'm in the middle!</CENTER>
```

Dòng trên cho kết quả như sau:

I'm in the middle!

Bạn có thể dùng nó với một hay nhiều tag khác, ví dụ:

```
<CENTER><B><I>Look at me now!</I></B></CENTER>
```

Look at me now!

Sử dụng các đề mục, các đoạn văn bản, và ngắt dòng.

Bắt đầu với các tag đề mục (*Heading*). Các tag này hữu dụng cho việc tạo các tiêu đề hay các đề mục của các phần. Sau đây là vài ví dụ:

```
<H1>Large Heading!</H1>
```

cho kết quả:

Large Heading!

<H2>Heading 2</H2>

Heading 2

<H3>Heading 3</H3>

Heading 3

<H4>Getting Small</H4>

Getting Small

<H5>Smaller Still...</H5>

Smaller Still...

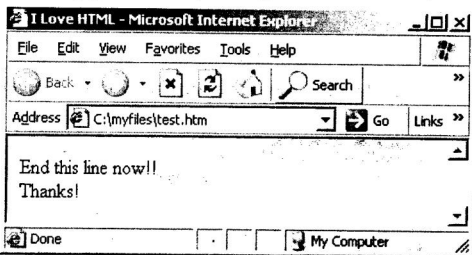
<H6>You must have good vision...</H6>

You must have good vision...

Bây giờ ta tìm hiểu tag ngắt dòng. Tag dùng để ngắt dòng là tag
. Khi cho tag này vào tài liệu của bạn, nội dung đứng sau nó sẽ qua dòng kế tiếp. Tag
 không cần tag đóng đứng sau nó. Ví dụ:

End this line now!!
Thanks!

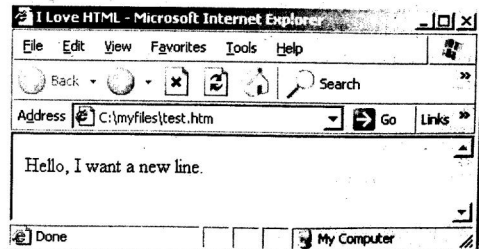
Trình duyệt sẽ hiển thị như sau:



Về cơ bản, một ngắt dòng giống như bạn nhấn phím “Enter” khi bạn viết văn bản. Trình duyệt sẽ chuyển qua dòng mới khi phía sau văn bản vẫn không còn chỗ trống, hay thấy tag ép nó phải qua dòng mới. Do đó, nếu bạn gõ các dòng sau vào trình soạn thảo, trình duyệt chỉ thể hiện trên một dòng:

```
Hello,  
I want  
a new line.
```

Kết quả là:

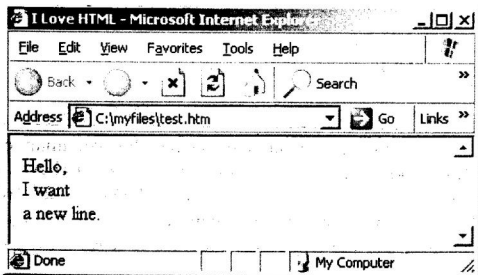


Căn bản thiết kế Web

Để văn bản chuyển qua dòng mới, sử dụng tag
:

```
Hello,<BR>
I want<BR>
a new line.
```

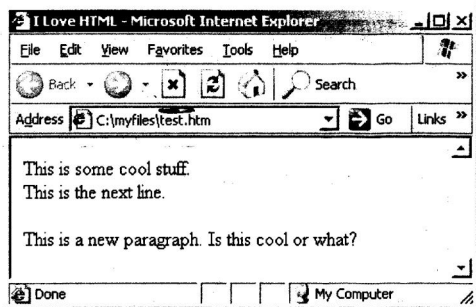
Bây giờ thì trình duyệt hiển thị theo ý muốn của bạn:



Hãy chuyển qua tìm hiểu tag đoạn văn bản (*paragraph*), <P>. Tag này sẽ nhảy qua một khoảng trống theo chiều dọc sau khi chuyển qua dòng mới, giống như là bạn dùng
 hai lần. Tag này tốt trong trường hợp muốn nhảy qua một dòng trước khi bắt đầu một đoạn văn bản mới. Ví dụ:

```
This is some cool stuff.
<BR>
This is the next line.
<P>
This is a new paragraph. Is this cool or what?
```

Sẽ cho kết quả như sau:



Tag `<P>` không yêu cầu có một tag đóng, nhưng có một tag đóng, `</P>`, sẽ rõ ràng hơn.

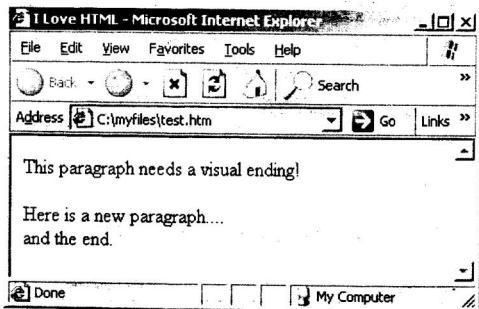
```
<P>
This paragraph needs a visual ending!
</P>
<P>
Here is a new paragraph....<BR>
and the end.
</P>
```

Đoạn code trên cho cùng một kết quả như chỉ dùng tag

`<P>`:

```
<P>
This paragraph needs a visual ending!
<P>
Here is a new paragraph....<BR>
and the end.
```

Cả hai đoạn code trên cho kết quả như sau:



Cách thay đổi kích thước và màu sắc văn bản

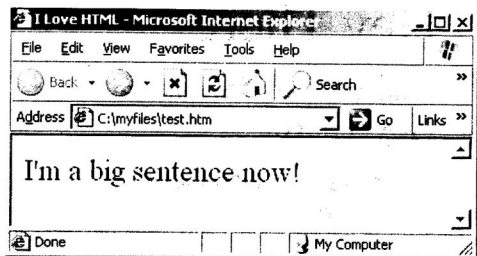
Bạn muốn thay đổi kích thước của văn bản. Làm điều này với tag sau:

```
<FONT SIZE="x">text to change</FONT>
```

"x" sẽ thay thế bằng một số và kết hợp với dấu + hay - phía trước nó. Giả sử bạn muốn kích thước văn bản lớn hơn. Bạn có thể dùng tag trên với +2, như sau:

```
<FONT SIZE="+2">I'm a big sentence now!</FONT>
```

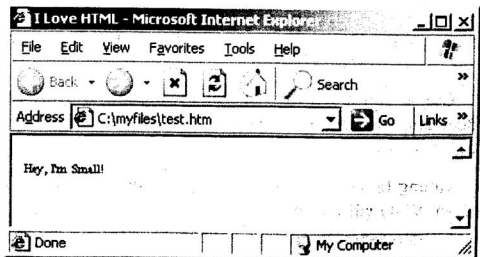
Sẽ cho kết quả như sau:



Tương tự, bạn có thể làm cho kích thước nhỏ hơn với dấu -:

```
<FONT SIZE="-2">Hey, I'm Small!</FONT>
```

Sẽ cho dòng sau:



Sau đây là một số ví dụ khác về kích thước văn bản:

Hey There

Hey There

Hey There

Hey There

Hey There

Hey There

Hey There

Hey There

Hey There

Hey There

Hey There

Hey There

Can you read this?

Can you read this?

Giả sử bạn muốn thay đổi màu chữ (*font color*). Nó cũng được làm với cùng một cách. Tag này như sau:

Chúng ta thay thế color với tên màu hoặc giá trị màu hexa. Ví dụ với tên màu:

I'm red!

I'm red!

Căn bản thiết kế Web

Thử dùng giá trị hexa của màu đỏ. Các giá trị hexa phải bắt đầu bằng dấu # và theo sau là sáu ký tự hoặc số. Ví dụ:

```
<FONT COLOR="#FF0000">I'm red!</FONT>
```

I'm red!

Thường thì tên màu dễ nhớ hơn, nhưng cần dùng các giá trị hexa cho các màu phức tạp hơn.

Giả sử bạn muốn thay đổi cả kích thước và màu sắc. Bạn có thể dùng hai tag FONT và nhớ đóng chúng với hai tag:

```
<FONT SIZE="+2"><FONT COLOR="gold">I am  
gold!</FONT></FONT>
```

Nhưng bạn có thể dùng SIZE và COLOR trong cùng một tag, và chỉ cần một tag đóng. Giống như sau:

```
<FONT SIZE="+2" COLOR="gold">I am gold!</FONT>
```

Cách dùng các ký tự đặc biệt trong HTML

Giả sử bạn muốn thêm các khoảng trắng hay hiển thị ký tự ® (copyright) trên trang web của bạn. Đặt các ký tự đặc biệt lên trang web của bạn bằng cách dùng các tham chiếu đặc biệt đến các ký tự bạn muốn dùng. Ký hiệu tham chiếu bắt đầu bằng dấu &, theo sau là vài ký tự hay ký số, và kết thúc bằng dấu chấm phẩy (;). Ví dụ, bạn muốn đặt thêm một vài khoảng trắng giữa hai ký tự. Để làm điều này bạn đặt tham chiếu vào vị trí bạn muốn thêm khoảng trắng. Ví dụ:

Hello there!

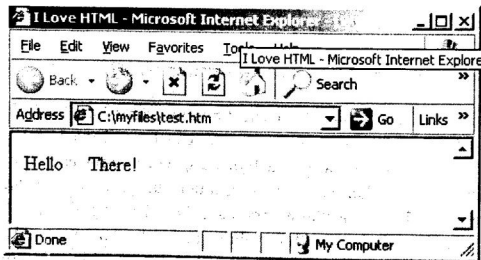
Dòng trên chèn hai khoảng trắng giữa "Hello" và "there!", như sau:

Hello there!

Khoảng trắng đầu tiên được thêm vào dòng thanh “space” trên bàn phím. Trình duyệt web sẽ thấy khoảng trắng đầu tiên, nhưng sau đó có bao nhiêu khoảng trắng cũng không có gì khác biệt – nghĩa là bạn chỉ thấy khoảng trắng đầu tiên trong trình duyệt. Bằng cách thêm , chúng ta ép trình duyệt thêm một khoảng trắng giữa hai từ. Bạn có thể thêm bao nhiêu khoảng trắng nếu bạn muốn, ví dụ:

Hello There!

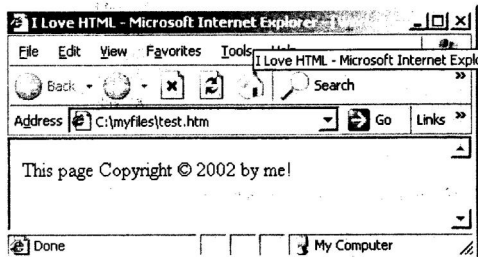
Dòng trên tạo một khoảng trắng và bốn khoảng trắng phụ giữa hai từ, tổng cộng có năm khoảng trắng. Trình duyệt sẽ hiển thị như sau:



Ta tìm hiểu một ký tự đặc biệt khác là ký tự *copyright*. Dùng nó giống như dùng ký tự khoảng trắng, bằng cách đặt tham chiếu của nó tại vị trí bạn muốn thấy ký hiệu

trên trang web. Tham chiếu của *copyright* là ©. Sau đây là một ví dụ:

This page Copyright © 2002 by me!



Các ký tự đặc biệt khác cũng được dùng theo cách này. Chỉ cần đặt tham chiếu của nó tại vị trí muốn nó xuất hiện trên trang web. Sau đây là danh sách một số các ký tự đặc biệt:

Ký tự	Ký hiệu
Space	
"	"
<	<
>	>
&	&
©	©
®	®
¢	¢
°	°
²	²
»	»

Căn bản thiết kế Web

«	&lquo;
¼	¼
½	½
¾	¾
±	±
ũ	ü
Ø	Ø
¡	¡
ñ	ñ

Cách liên kết với các trang khác trong HTML

Bây giờ là lúc học cách liên kết (link) với các trang khác. Hãy bắt đầu bằng cách xem tag gì chúng ta dùng để liên kết:

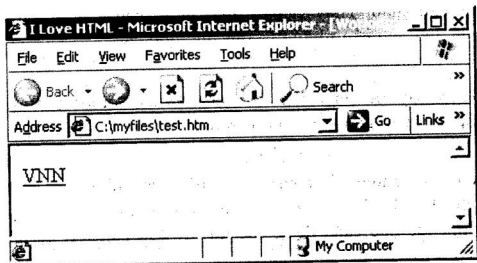
```
<A HREF="http://www.someplace.com">Display Text</A>
```

Tag <A> là tag dùng để liên kết, HREF="" là nơi để chỉ định địa chỉ link tới. Tag là tag đóng. Văn bản nằm giữa hai tag là những gì hiển thị trên màn hình đóng vai trò là một liên kết (link). Do đó, nếu bạn muốn liên kết đến trang web, bạn chỉ cần đặt địa chỉ bên trong dấu ngoặc kép. Ví dụ, tạo một liên kết đến trang web vnn có địa chỉ là *http://www.vnn.vn*, đặt dòng code sau vào trang web mà bạn muốn đặt link:

```
<A HREF="http://www.vnn.vn">VNN</A>
```

Căn bản thiết kế Web

Nó sẽ hiển thị trong trang web như sau:



Dòng này có màu và được gạch dưới. Trong đa số trường hợp, văn bản dạng này là một liên kết. Nếu bạn di chuyển chuột qua đầu link, bạn sẽ thấy con trỏ chuột chuyển sang hình bàn tay.

Xét một ví dụ khác, tạo một liên kết đến một trang cụ thể. Giả sử đến địa chỉ sau: http://www.vnn.vn/vnn_news/listweb.html. Để tạo link, đặt địa chỉ vào trong tag như sau:

```
<A  
HREF="http://www.vnn.vn/vnn_news/listweb.html">Linking  
to Other Pages</A>
```

Sẽ cho một link như sau:

Linking to Other Pages

Nếu bạn nhấn chuột vào link trên, trình duyệt sẽ hiển thị trang này. Bây giờ bạn tạo các liên kết đến các trang của bạn bằng cách thay thế địa chỉ trên thành địa chỉ của bạn. Nếu bạn đặt tất cả các file trong một thư mục, bạn có thể dùng một địa chỉ ngắn hơn, gọi là "địa chỉ cục bộ".

Trước khi bắt đầu, bạn phải bảo đảm là file mà bạn muốn liên kết tới nằm trong cùng một thư mục với trang bạn đang soạn thảo. Thay vì bạn phải nhập vào một địa chỉ đầy đủ, bạn chỉ cần nhập vào tên file, như sau:

```
<A HREF="linking.htm">Linking to Other Pages</A>
```

Dòng này tạo một liên kết đến trang "linking.htm", nhưng nó ngắn gọn hơn link ở trên.

Linking to Other Pages

Nếu bạn không chắc chắn, thì luôn luôn nhập vào các địa chỉ đầy đủ. Nhập vào các địa chỉ đầy đủ sẽ cho phép các link lúc nào cũng đúng cho dù file của bạn đặt ở chỗ nào trên Internet.

Cách đặt các hình ảnh vào trang web của bạn

Xem xét tag hình ảnh sau:

```
<IMG SRC="image.gif">
```

IMG viết tắt của image (hình ảnh). SRC viết tắt của source (nguồn). Source của hình ảnh là địa chỉ của hình ảnh đó. Thông thường chỉ cần nhập vào tên file hình ảnh ở đây, giống như sau:

```
<IMG SRC="image.gif">
```

Tên file không phải bắt buộc với phần mở rộng .gif. Bạn có thể dùng một file .jpg. Đây là hai loại file hình ảnh được dùng phổ biến trên Internet. Nếu bạn có một hình ảnh ở dạng khác, bạn nên chuyển chúng qua dạng .gif hay .jpg. Thao tác này có thể được thực hiện với hầu hết các chương trình xử lý ảnh.

Nếu hình ảnh của bạn trong một thư mục khác với thư mục chứa trang html, bạn có thể dùng một địa chỉ đầy đủ của hình ảnh. Nếu ảnh được đặt tại

Căn bản thiết kế Web

<http://www.coolness.com/pictures/image.jpg> , bạn dùng địa chỉ của ảnh như sau:

```
<IMG  
SRC="http://www.coolness.com/pictures/image.jpg">
```

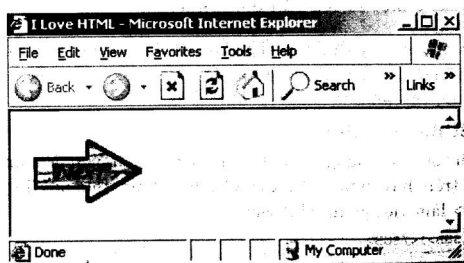
Nếu bạn không chắc chắn, thì dùng một địa chỉ đầy đủ lúc nào cũng chính xác. Ví dụ, một hình ảnh trong web site của bạn có tên "*next.jpg*". Địa chỉ của hình ảnh này là <http://www.yoursite.com/myfiles/next.jpg> . Nếu trang html cùng thư mục với hình này, bạn nhập vào như sau:

```
<IMG SRC="next.jpg">
```

Nếu khác, nên dùng một địa chỉ đầy đủ như sau:

```
<IMG  
SRC="http://www.yoursite.com/myfiles/next.jpg">
```

Cả hai cách trên đều hiển thị một hình ảnh trên trang web, cạnh lề bên trái, như sau:



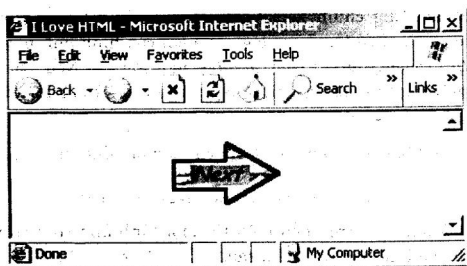
Nếu bạn muốn hình ảnh xuất hiện ở chính giữa, bạn đặt tag CENTER bọc tag hình ảnh, như sau:

Căn bản thiết kế Web

<CENTER>

</CENTER>

Đoạn code trên hiển thị một hình ảnh ở giữa màn hình:



Thử dùng một số tag văn bản khác.

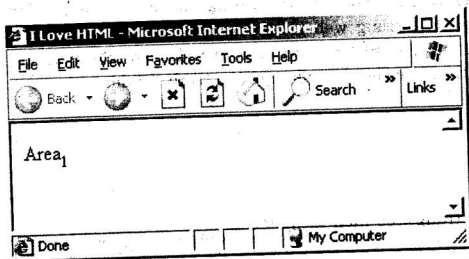
Có một số tag văn bản khác bạn sẽ dùng trong khi tạo trang web của bạn. Các tag này là <SUB>, <SUP>, <PRE>, <NOBR>, và <WBR>.

<SUB> và <SUP>

Các tag này dùng để đặt một số hay một vài từ nằm phía trên hay phía dưới các văn bản bình thường. Tag <SUB> làm việc giống như sau:

Area₁

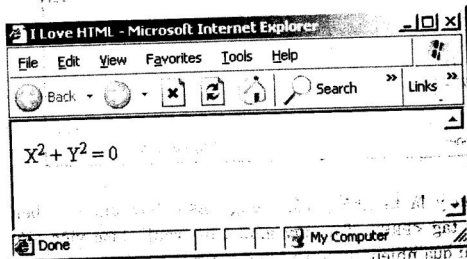
Cho ra kết quả:



Và tag <SUP> cho phép bạn viết các ký hiệu số mũ:

$$X² + Y² = 0$$

Dòng code này cho kết quả:



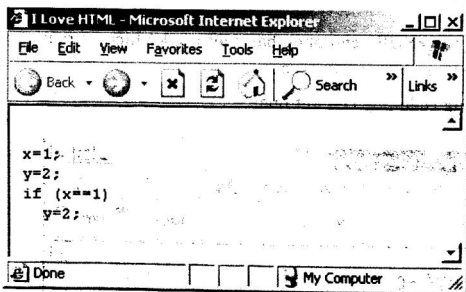
Các tag này hữu dụng khi viết các công thức toán học trên trang web của bạn.

<PRE> Tag

Tag <PRE> được sử dụng khi bạn muốn giữ các khoảng trống trong trang web giống với những gì bạn soạn thảo trong trang html trong trình soạn thảo. Nó hữu dụng khi hiển thị các code của ngôn ngữ lập trình. Ví dụ:

```
<PRE>
x=1;
y=2;
if (x==1)
    y=2;
</PRE>
```

Và trang web hiển thị như sau:



Chú ý là ta không cần dùng
 hay bên trong tag <PRE>. Tag này giúp bạn tránh được việc viết lặp lại quá nhiều
 và khoảng trắng.

<NOBR> và <WBR>

Bất kỳ văn bản nào đặt bên trong các tag <NOBR> và </NOBR> sẽ không ngắt qua dòng mới, ví dụ:

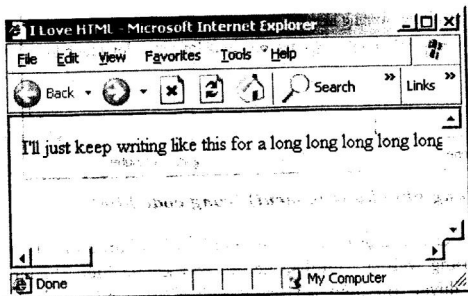
Căn bản thiết kế Web

<NOBR>

I'll just keep writing like this for a long long long
long long long long long long long long long long
long long long long long long long long long long
long long long long long long long long long long
long long long long long long long long long long
long long long long long long long long long long
long long long long long long long long long long
long time.

</NOBR>

Khi hiển thị trên trình duyệt của bạn, nó thực sự là một dòng dài, như sau:



Trừ khi bạn có màn hình đủ rộng để đọc dòng này, thường thì phải dùng thanh cuộn sang phải để xem phần cuối của dòng văn bản. Bạn có thể dùng tag <WR> bên trong tag NOBR để ép ngắt dòng nếu bạn muốn như vậy:

<NOBR>

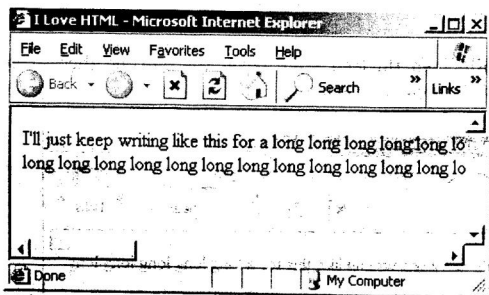
I'll just keep writing like this for a long long long
long long long long long long long long long long
long long long long long long long long long long
long long long long long long long long long long
long long long long long long <WR>

Cần bản thiết kế Web

[illegible]

</NOBR>

Bây giờ bạn có hai dòng không dài bằng nhau:



Sử dụng ghi chú (comment) trong code html

Ghi chú là một cách tốt để giúp bạn khi bạn đang thiết kế trang web của mình. Các ghi chú không hiển thị lên trình duyệt khi hiển thị trang web. Chỉ có một cách để xem ghi chú là nhìn trong mã html nguồn. Ghi chú cho phép bạn ghi lại các lời chú ý để bạn dễ dàng nâng cấp trang web sau này. Để ghi một ghi chú, bắt đầu bằng một dấu nhỏ hơn (<) theo sau là một dấu chấm than (!) và hai dấu trừ (--). Sau đó gõ vào ghi chú của bạn. Để kết thúc ghi chú, dùng hai dấu trừ (--) và một dấu lớn hơn (>).

```
<!-- I am a comment. I feel invisible though. -->
```

Bạn có thể ghi chú nhiều dòng, và nhớ có dấu kết thúc ghi chú:

Căn bản thiết kế Web

<!--

You can't see me....

unless you look at the page source code.

-->

Bạn có thể dùng ghi chú để nhắc cho chính mình biết là một đoạn code làm công việc gì:

<!-- This image should be aligned to the right, and have alt text -->

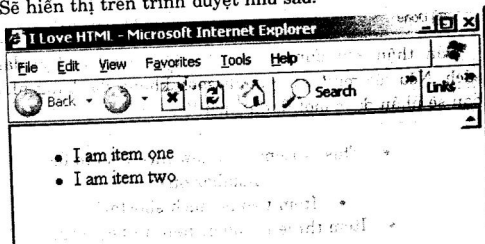
Cách thêm danh sách (list) vào trang web của bạn.

Để bắt đầu chúng ta cần một tag để bắt đầu và kết thúc toàn bộ danh sách. Trước tiên, dùng một danh sách không có thứ tự. Đoạn code này tạo một danh sách có các *bullet* bên cạnh các mục trong danh sách. Tag mở là , và tag đóng là . Để đặt các phần tử vào danh sách, bạn dùng tag , theo sau là dòng văn bản. Sau đây là một ví dụ với hai mục trong danh sách:

I am item one

I am item two

- Sẽ hiển thị trên trình duyệt như sau:

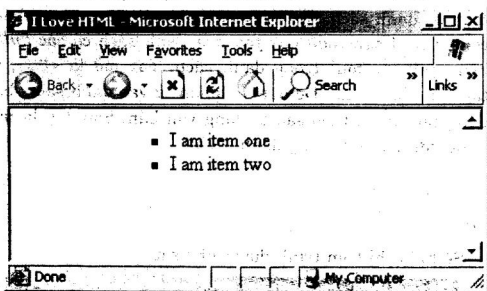


Căn bản thiết kế Web

Chú ý là tag không yêu cầu một tag đóng. Và danh sách được thụt vào so với các văn bản khác. Bạn có thể thụt vào tùy ý bằng cách thêm các tag , và nhớ đóng tất cả chúng lại. Ví dụ:

```
<UL>  
<UL>  
<UL>  
<LI>I am item one  
<LI>I am item two  
</UL>  
</UL>  
</UL>
```

Danh sách được thụt vào như sau:



Cẩn thận khi dùng tag <CENTER> để bao bọc danh sách. Nếu các mục trong danh sách không dài bằng nhau, bạn sẽ nhận được một danh sách lộn xộn:

- This is item one, how nice it is to be number one.
 - Item two is much shorter!
- Item three is somewhere in between..

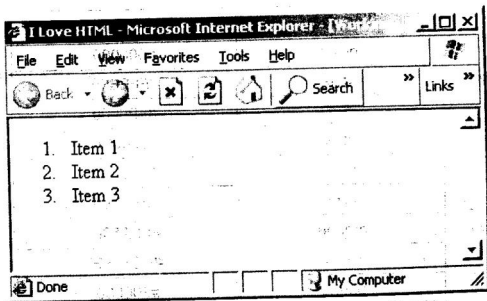
Căn bản thiết kế Web

Nếu bạn muốn danh sách ở giữa trang, bạn có thể dùng các tag (nhưng nó có thể lộn xộn trên các màn hình có độ phân giải khác nhau), hoặc bạn có thể dùng các bảng (table) để canh lề văn bản.

Bạn cũng có thể dùng một danh sách có thứ tự giống như là sử dụng danh sách không có thứ tự. Thay vì dùng , bạn dùng , như sau:

```
<OL>
<LI>Item 1
<LI>Item 2
<LI>Item 3
</OL>
```

Đoạn code trên cho bạn một danh sách có đánh số thứ tự thay vì các bullet:



Cách dùng các thuộc tính của tag BODY để làm đẹp trang web.

Khi tạo một trang web, bạn có thể thay đổi nhiều thứ trong phần thân của document bằng cách thêm các thuộc tính phụ cho tag <BODY>. Sau đây là một ví dụ về tag BODY với nhiều thuộc tính:

```
<BODY bgcolor="black" text="red" link="yellow"  
alink="orange" vlink="white" background="image.gif">
```

bgcolor="color"

Thuộc tính này thay đổi màu nền của trang web. Bạn có thể đặt nó là bất kỳ màu gì bạn thích. Chỉ cần thay thế *color* bằng tên của màu hoặc giá trị hexa. Sau đây là danh sách một số màu phổ biến.

Tên màu	Giá trị hexa
black	#000000
white	#FFFFFF
red	#FF0000
blue	#0000FF
green	#008000
yellow	#FFFF00
orange	#FFA500
violet	#EE82EE
purple	#800080
pink	#FFC0CB
silver	#C0C0C0
gold	#FFD700
gray	#808080

Căn bản thiết kế Web

aqua	#00FFFF
skyblue	#87CEEB
lightblue	#ADD8E6
fuchsia	#FF00FF
khaki	#F0E68C

text="color"

Thay đổi màu văn bản mặc nhiên của trình trình duyệt. Màu mặc nhiên của trình duyệt là màu đen.

link="color"

Thay đổi màu của tất cả các đầu link chưa được đến trên trang web của bạn. Mặc mặc nhiên của các đầu link là màu xanh.

alink="color"

Thay đổi màu của link vừa được người sử dụng nhấn chuột vào.

vlink="color"

Thay đổi màu của các đầu link đã đến. Màu mặc nhiên là màu tím.

background="image.gif"

Đặt ảnh nền cho trang web của bạn. Nếu bạn dùng thuộc tính này, ảnh nền sẽ trải hết khoảng trống của nền. Nếu không dùng ảnh nền, trình duyệt sẽ dùng màu nền hoặc màu nền mặc nhiên.

Cách thay đổi màu mặc nhiên của phong chữ

Trong trang html của bạn, tìm tag BODY, và xem có thuộc tính text="" hay chưa. Có thể đặt thuộc tính này như sau:

Căn bản thiết kế Web

```
<BODY text="black">
```

Bạn cũng có thể đặt giá trị là một số hexa:

```
<BODY text="#000000">
```

Nếu trong tag BODY có nhiều thuộc tính, phải có ít nhất một khoảng trắng giữa các thuộc tính:

```
<BODY bgcolor="blue" text="black">
```

Bạn có thể thay thế các màu trên bằng bất kỳ màu gì bạn thích.

Cách thay đổi màu của link trong trang web của bạn

Nếu bạn muốn thay đổi màu của các đầu link trong trang web của bạn, tìm tag <BODY>. Xem có thuộc tính link="color" hay chưa, nó có thể như sau:

```
<BODY link="blue">
```

hay tag <BODY> có thêm các thuộc tính khác:

```
<BODY bgcolor="#000000" text="#FFFFFF" link="#0000FF">
```

Nếu bạn chỉ thấy tag BODY mà không có thuộc tính nào, bạn có thể thêm một thuộc tính vào như sau:

```
<BODY link="blue">
```

Để thay đổi màu của đầu link, bạn thay thế color trong đầu ngoặc kép bằng một tên màu hay một giá trị hexa.

Giả sử, bạn muốn thay đổi màu của đầu link sang màu đỏ, bạn làm như sau:

```
<BODY link="red">
```

hoặc dùng một giá trị hexa:

```
<BODY link="#FF0000">
```

Bây giờ tất cả các đầu link trên trang web của bạn đều có màu đỏ. Bạn cũng có thể làm tương tự như trên với màu

Căn bản thiết kế Web

của các đầu link đã đến và vừa mới đến bằng cách thay đổi hoặc thêm các thuộc tính. Các thuộc tính đó là:

alink="color" dùng cho màu của đầu link vừa nhấn vào

vlink="color" dùng cho các đầu link đã vào

Có thể làm như sau:

```
<BODY link="blue" alink="blue" vlink="violet">
```

Bạn có thể thay đổi thành bất kỳ màu nào bạn muốn. Nếu bạn muốn thêm các thuộc tính khác, chỉ việc gõ thêm vào và giữa chúng phải có ít nhất một khoảng trắng.

Thêm hoặc thay đổi màu nền của trang web

Nếu bạn muốn sử dụng một màu nền trên trang web, bạn phải thêm thuộc tính bgcolor="color". Có thể như sau:

```
<BODY bgcolor="gray">
```

hay có nhiều thuộc tính khác

```
<BODY text="#000000" link="#A6CAF0" bgcolor="#808080">
```

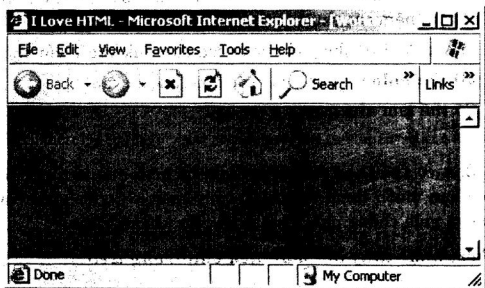
Nếu không có thuộc tính này bạn có thể thêm chúng vào, như sau:

```
<BODY bgcolor="gray" text="#000000" link="#A6CAF0">
```

Hay tag BODY không có thuộc tính nào, thì ngắn gọn như sau:

```
<BODY bgcolor="gray">
```

Tương tự, chỉ cần thay thế màu vào trong dấu ngoặc kép.



Cách cho một hình ảnh làm hình ảnh nền của trang web

Bạn tìm đến tag `<BODY>`, và dùng thuộc tính `background=" "`, nếu không có thì thêm vào. Tag BODY bây giờ có thể tương tự như sau:

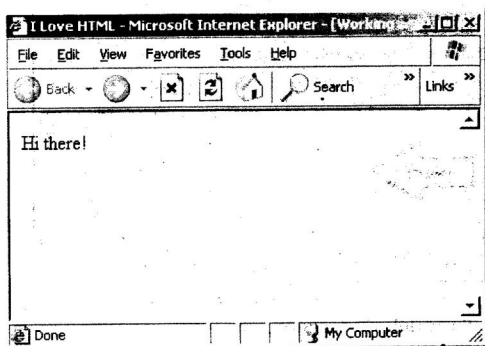
```
<BODY background=" ">
```

Chúng ta sẽ đặt địa chỉ của hình ảnh bên trong dấu ngoặc kép. Giả sử hình ảnh bạn muốn dùng có địa chỉ là `http://www.mysite.com/image1.gif`, bạn chèn nó vào dấu ngoặc kép như sau:

```
<BODY background="http://www.mysite.com/image1.gif">
```

Nếu file hình ảnh và file html ở cùng một thư mục, bạn chỉ cần nhập vào tên file hình ảnh. Nếu bạn muốn `image2.gif` làm hình ảnh nền, và nó cùng thư mục với file bạn đang soạn thảo, tag BODY có thể như sau:

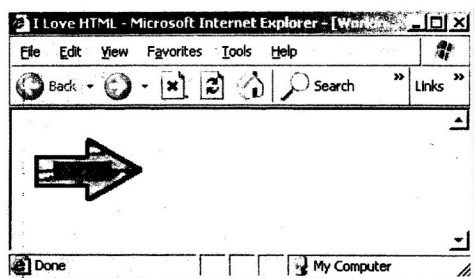
```
<BODY background="image2.gif">
```



Cách định lại kích thước của hình ảnh

Bạn có một hình ảnh muốn sử dụng, nhưng nó không đúng với kích thước bạn mong muốn. Bạn có thể định lại kích thước của hình ảnh bằng cách thêm các thuộc tính width và height vào tag hình ảnh. Hãy xét một ví dụ, ta có một hình ảnh có tên "next.jpg". Giả sử bạn muốn làm cho nó nhỏ hơn. Trước tiên, bạn cần biết kích thước gốc của hình ảnh. Chiều rộng (width) và chiều cao (height) thường được tính bằng pixel (điểm ảnh). Một màn hình tiêu chuẩn thường có chiều rộng là 800 pixel và chiều cao là 600 pixel. Hình "next.jpg" có chiều rộng là 106 pixel và chiều cao là 65 pixel. Hình gốc trông như sau:

Căn bản thiết kế Web



Do đó để thay đổi kích thước của hình ảnh, bạn sẽ đặt các thuộc tính vào tag hình ảnh:

`width=" "` và
`height=" "`

Tag hình ảnh sẽ như sau:

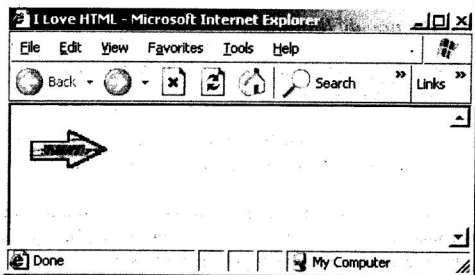
```
<IMG SRC="next.jpg" width=" " height=" ">
```

Để làm cho hình ảnh nhỏ hơn, bạn đặt các con số vào các dấu ngoặc kép. Giả sử bạn muốn nó rộng 75 pixel và cao 40 pixel. Bạn chỉ cần đặt các con số này vào trong các dấu ngoặc kép của các thuộc tính:

```
<IMG SRC="next.jpg" width="75" height="40">
```

Bây bạn xem lại trang này, hình ảnh sẽ có chiều rộng và chiều cao mới với các giá trị do bạn chỉ định. Đây là kết quả:

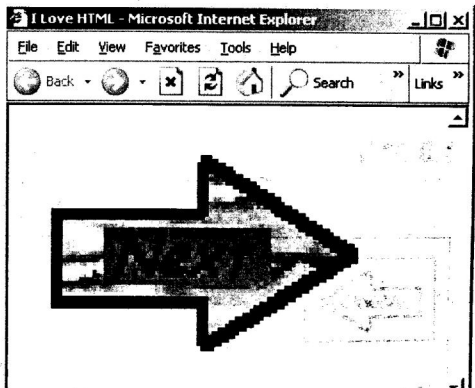
Căn bản thiết kế Web



Để có một ảnh lớn hơn cũng làm tương tự, chỉ cần chỉ định các số lớn hơn:

```
<IMG SRC="next.jpg" width="300" height="200">
```

Kết quả như sau:



Làm thế nào để link dùng một hình ảnh

Để sử dụng một hình ảnh như là một đầu link, bạn cần kết hợp hai thứ đã học:

1. Cách tạo một liên kết (link).
2. Cách cho một hình ảnh vào trang web.

Trước hết bạn phải tạo một liên kết đến địa chỉ mà bạn muốn người sử dụng sẽ đến đó nếu họ nhấn vào hình ảnh. Ví dụ, tạo một tag liên kết mở như sau:

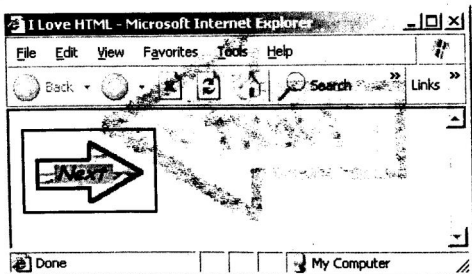
```
<A HREF="http://www.vnn.vn">
```

Tiếp theo, không gõ văn bản và cũng chưa đóng tag. Chúng ta sẽ đặt một tag hình ảnh bên phải tag link mở. Hình ảnh ta dùng ở đây là "next.jpg". Code sẽ như sau:

```
<A HREF="http://www.vnn.vn">  
<IMG SRC="next.jpg">
```

Bây giờ ta đóng tag link, như sau:

```
<A HREF="http://www.vnn.vn">  
<IMG SRC="next.jpg">  
</A>
```



Hình ảnh được đặt trong tag link, do đó nó làm việc như là các link bình thường, nhưng bây giờ nó là một hình ảnh trực quan. Kết quả là một hình ảnh có thể link.

Di chuyển con trỏ chuột qua hình ảnh, nó sẽ trở thành hình bàn tay. Nếu bạn nhấn vào hình ảnh, trình duyệt sẽ mở trang web vnn.

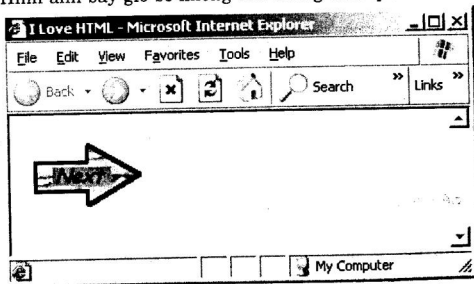
Lúc này hình ảnh có một đường viền bao quanh nó. Đường viền này mặc nhiên được các trình duyệt thêm vào. Để bỏ nó, thêm thuộc tính sau vào tag hình ảnh:

`border="0"`

ví dụ:

```
<A HREF="http://www.pageresource.com">  
<IMG SRC="next.jpg" border="0">  
</A>
```

Hình ảnh bây giờ sẽ không còn đường viền quanh nó:



Bạn cũng có thể tăng kích thước của đường viền với cùng cách trên, chỉ cần đặt một số lớn hơn vào đó, ví dụ:

`border="5"`

Cách xuống dòng tự động (wrap) văn bản bao quanh hình ảnh

Để cho phép văn bản bao quanh hình ảnh, bạn chỉ cần cho các thuộc tính sau vào tag hình ảnh:

```
align="left"  
align="right"
```

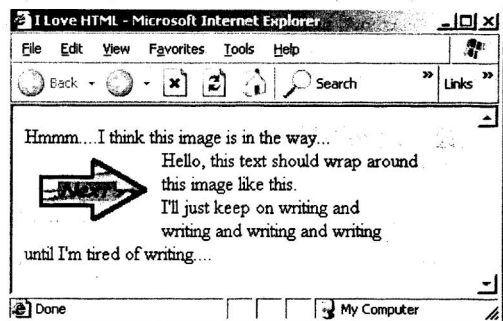
Tag IMG sẽ giống như sau:

```
<IMG SRC="next.jpg" align="left">
```

Bây giờ chỉ cần nhập vào văn bản và nó sẽ bao quanh hình ảnh, không phải là nhảy đến phía dưới của hình ảnh. Ví dụ:

```
Hmmm....I think this image is in the way...<br>  
<IMG SRC="next.jpg" align="left">  
Hello, this text should wrap around<br>  
this image like this.<br>  
I'll just keep on writing and<br>  
writing and writing and writing<br>  
until I'm tired of writing....<br>
```

Sẽ cho kết quả như sau:



Nếu bạn muốn văn bản trải ngang hết màn hình, bạn không cần các tag ngắt dòng. Ta dùng nó ở đây để văn bản đủ dài. Bạn có thể làm tương tự với thuộc tính `align="right"`.

Một thuộc tính (alternative - alt) khác của hình ảnh

Thuộc tính `alt` được sử dụng để hiển thị văn bản trong trường hợp trình duyệt không thể hiển thị hình ảnh, hay trình duyệt tắt chức năng hiển thị hình ảnh để trang web tải nhanh hơn. Nếu bạn dùng hình ảnh như là một link, đây là cách cho người sử dụng biết hình ảnh này sẽ làm gì. Đây là thuộc tính `alt`:

`alt="something you want to write"`

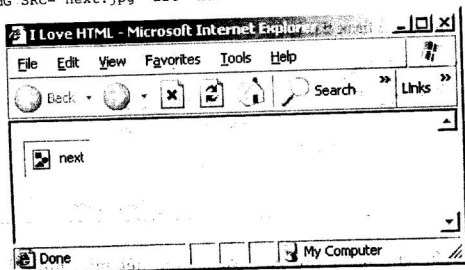
Đặt nó bên trong tag hình ảnh:

`<IMG SRC="image.gif" alt="something you want to write">`

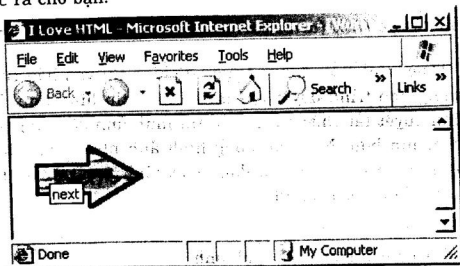
Căn bản thiết kế Web

Ví dụ bạn có thể hiển thị "next" với thuộc tính alt của hình "next.jpg", người sử dụng sẽ thấy "next" tại vị trí của hình ảnh trên trình duyệt không hỗ trợ chức năng hiển thị hình ảnh:

```
<IMG SRC="next.jpg" alt="next">
```



Nếu bạn có một trình duyệt đủ mới, bạn di chuyển chuột đến hình ảnh và nó sẽ hiển thị các văn bản trong alt ra cho bạn.



Nhảy đến đầu trang, cuối trang hay một vị trí bất kỳ.

Giả sử bạn có một trang web khá dài. Và bạn muốn cho người sử dụng một cách trở về đầu trang khi họ đang xem ở cuối trang. Hay bạn muốn chia nó ra thành nhiều phần và dùng mục lục tại phía trên của trang web. Cách để làm điều này là đặt tên một đầu link (móc neo), đó là nơi mà bạn muốn tạo liên kết đến. Bạn có thể link đến bất kỳ phần nào trong trang web, bây giờ thử tạo một link đến đầu trang. Để làm điều này, đến phần đầu của phần BODY (ngay sau tag BODY), nhập dòng code sau vào:

```
<A NAME="top"></A>
```

Bạn có thể đặt bất kỳ tên nào bạn muốn trong dấu ngoặc kép. Và đến một vị trí nào đó trong phần BODY và nhập link sau vào:

```
<A HREF="#top">Back to the Top</A>
```

Nếu bạn nhấn vào link này, bạn sẽ trở về đầu trang.

Dấu # báo cho trình duyệt biết đích đến là một móc neo trong trang. Và nếu bạn đặt tên của móc neo là "cool", thì bạn link đến đó bằng cách dùng "#cool".

Bạn có thể đặt một móc neo ở bất kỳ nơi nào trong trang web. Và có thể link đến bất kỳ phần nào trong trang web.

Cách để tạo một email link

Để tạo một email link, bạn chỉ cần dùng tag link. Điểm khác nhau là những gì đặt trong địa chỉ để link tới. Để ép trình duyệt hiểu nó như là một email link, bạn dùng "mailto:" thay cho "http://" trong phần đầu của địa

Căn bản thiết kế Web

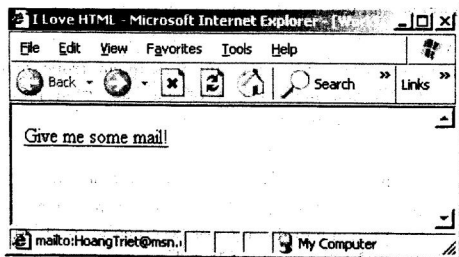
chỉ. Tiếp theo sau "mailto:", bạn dùng một địa chỉ email thay vì địa chỉ web:

```
<A HREF="mailto:your_email_address">E-mail Me!</A>
```

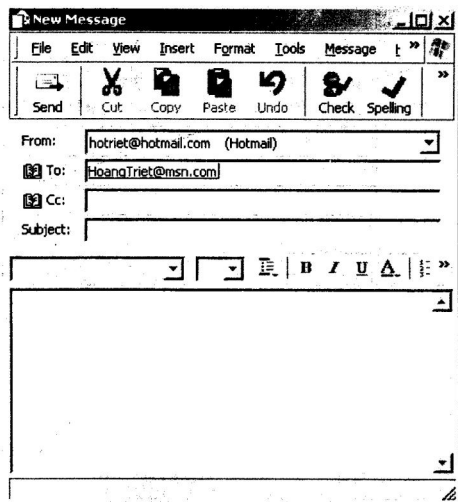
Để người sử dụng gửi mail cho bạn khi nhấn vào link này, chỉ cần thay thế `your_email_address` bằng địa chỉ mail của bạn, ví dụ:

```
<A HREF="mailto:HoangTriet@msn.com">Give me some mail!</A>
```

kết quả như sau:



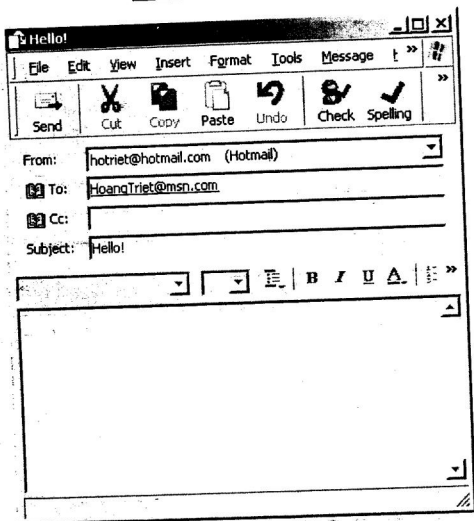
Nếu bạn nhấn vào link trên, trình duyệt sẽ mở một cửa sổ để cho bạn gửi mail, cửa sổ này đã được nhập sẵn địa chỉ của người nhận ở trên.



Bạn cũng có thể tạo sẵn chủ đề (Subject) của thông điệp cho người sử dụng. Làm điều này bằng cách thêm "?" phía sau địa chỉ email và tiếp theo là chủ đề:

```
<A HREF="mailto:HoangTriet@msn.com?subject=Hello!">Mail Me!</A>
```

Khi nhấn vào link trên, cửa sổ gửi mail đã có sẵn địa chỉ người nhận và chủ đề của email.



Các mẹo khác với email link

Trong phần trước, chúng ta đã biết cách thêm chủ đề cho một thông điệp thông qua e-mail:

```
<A HREF="mailto:you@you.com?subject=Hi">Mail Me</A>
```

Ngoài việc thêm chủ đề, có hai thứ có thể thêm là cc và bcc, tương tự như thêm chủ đề.

Bạn có thể tạo các email link như sau:

```
<A HREF="mailto:you@you.com?cc=friend@friend.com">  
Mail Me</A>
```

Căn bản thiết kế Web

Mail Me

Bạn có thể gửi đến nhiều địa chỉ bằng cách phân cách chúng bằng dấu phẩy (,):

Mail Me

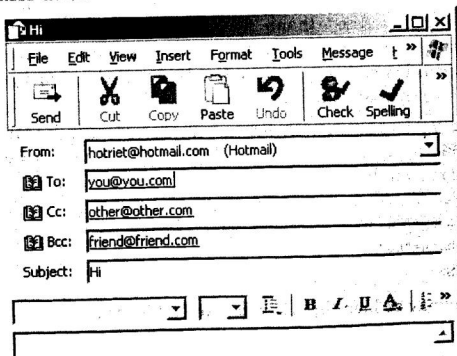
Nếu bạn muốn dùng nhiều thuộc tính, bạn nối chúng lại bằng dấu &:

<A
HREF="mailto:you@you.com?subject=Hi&bcc=friend@friend.co
m">

Mail Me

<A
HREF="mailto:you@you.com?subject=Hi&bcc=friend@friend.co
m&cc=other@other.com">

Mail Me



Cách thêm các form vào trang web.

Để sử dụng một form trong trang web, bạn cần hai thứ: code HTML để tạo form và một chương trình CGI hoặc script trên server (ví dụ: ASP, PHP, PERL...) để xử lý khi nó được gửi lên server.

Xét tag tạo form sau:

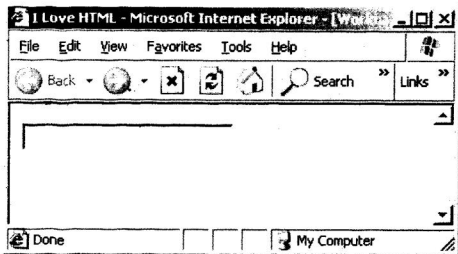
```
<FORM method="post" action="/cgi-bin/mailme">
...Your form stuff...
</FORM>
```

Thuộc tính method hầu như lúc nào cũng đặt là "post". Một giá trị khác là "get", nhưng nếu bạn dùng một chương trình trên server để xử lý, thì dùng "post". Thuộc tính action yêu cầu địa chỉ của chương trình CGI (hay ASP script, PHP...) dùng để xử lý form khi nó được gửi lên server (submit). Bạn thay thế chuỗi "/cgi-bin/mailme" thành địa chỉ của chương trình của bạn trên server. Trước khi đưa các chương trình cgi hoặc các script lên server, phải bảo đảm là server cho phép bạn dùng các chương trình này.

Ngay khi bạn mở tag form, tiếp theo là code HTML. Bắt đầu bằng việc xét tag cho phép người dùng nhập vào một dòng văn bản:

```
<FORM method="post" action="test.asp">
<INPUT type="text" name="yourname" size="25"
maxlength="60">
</FORM>
```

Tag này sẽ hiển thị một hộp textbox như sau:



Tên của loại tag này là INPUT

Thuộc tính type=" " xác định kiểu nhập vào từ người sử dụng. Bởi vì bạn muốn chỉ nhập vào một dòng văn bản, nên type là "text".

Thuộc tính name=" " đặt tên cho dòng văn bản để bạn có thể nhận biết nó khi viết JavaScript hoặc khi được chuyển lên server. Trong ví dụ này đặt tên là "yourname".

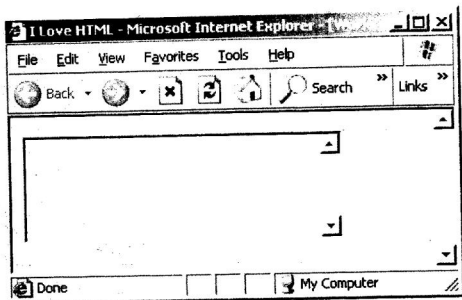
Thuộc tính size cho phép bạn quy định chiều dài của hộp textbox tính theo ký tự.

Thuộc tính maxlength = "x" ép hộp văn bản chỉ cho phép chứa x ký tự. Trong trường hợp này là 60.

Làm thế nào để có một không gian lớn hơn cho phép người sử dụng nhập vào nhiều dòng? Bạn dùng tag <TEXTAREA>. Ví dụ:

```
<TEXTAREA name="comments" rows="5" cols="30"></TEXTAREA>
```

Sẽ hiển thị trên trình duyệt như sau:



Tag TEXTAREA đòi hỏi có một tag đóng để kết thúc.

Thuộc tính `name` = " " cho phép bạn đặt tên cho TEXTAREA.

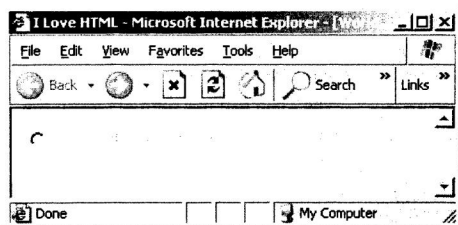
Thuộc tính `rows` = " " cho phép bạn chỉ định số dòng mà TEXTAREA sẽ hiển thị.

Thuộc tính `cols` = " " quy định bề rộng của TEXTAREA tính bằng ký tự.

Tạo một radio button thì sao?, tag của nó là:

```
<INPUT type="radio" name="foodchoice" value="pizza">
```

Sẽ cho một nút nhỏ cho phép người dùng nhấn vào để đánh dấu (chọn) nó:



Thuộc tính `type = "radio"` báo cho trình duyệt hiển thị một nút radio.

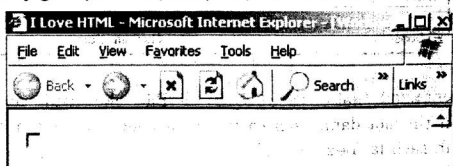
Thuộc tính `name = " "` cho phép bạn đặt tên cho các phần chọn lựa. Nếu bạn có một nhóm các nút radio, bạn phải đặt chúng có cùng một tên. Trong trường hợp này là "foodchoice".

Thuộc tính `value = " "` cho phép bạn gán các giá trị cụ thể cho từng *radio button* khi nó được chọn. Đó là giá trị bạn nhận được khi người sử dụng chọn. Trong ví dụ trên là "pizza".

Để tạo hộp kiểm (*checkbox*), dùng tag như sau:

```
<INPUT type="checkbox" name="foodchoice" value="pizza">
```

Bây giờ bạn sẽ nhận được một hình như sau:



Căn bản thiết kế Web

`type="checkbox"` báo cho trình duyệt hiển thị một *checkbox*.

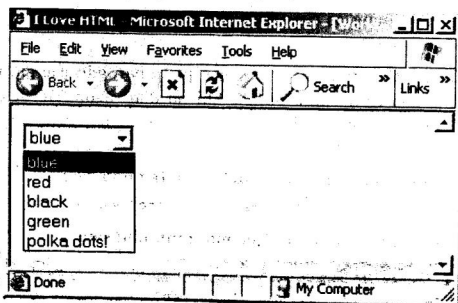
`name=" "` cho phép bạn đặt tên cho checkbox.

`Value = " "` cho phép bạn quy định giá trị cụ thể.

Để tạo một danh sách đổ xuống, dùng tag `<SELECT>`:

```
<SELECT name="bestcolor">  
<OPTION SELECTED>blue  
<OPTION>red  
<OPTION>black  
<OPTION>green  
<OPTION>polka dots!  
</SELECT>
```

Sẽ cho bạn một danh sách như sau, nhấn vào dấu mũi tên ở phía bên phải để thấy các mục chọn lựa:



`<SELECT name="bestcolor">` Báo cho trình duyệt hiển thị một danh sách có thể chọn được, và đặt tên cho danh sách là "bestcolor".

Căn bản thiết kế Web

<OPTION SELECTED> Cho phép bạn chỉ định mục nào được hiển thị như là giá trị mặc nhiên trên màn hình.

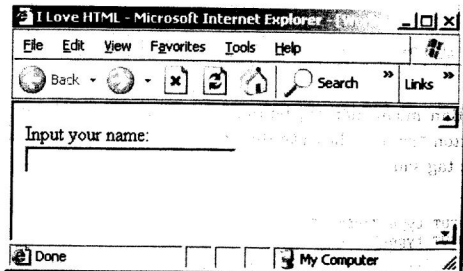
<OPTION> cho phép bạn thêm các mục chọn lựa khác vào danh sách. Bạn có thể dùng bao nhiêu tag này cũng được.

</SELECT> kết thúc danh sách.

Làm thế nào để hiển thị văn bản bên cạnh các INPUT trên để người sử dụng biết được họ nhập gì vào. Chỉ cần đặt văn bản vào vị trí bạn muốn nó xuất hiện. Dùng khoảng trắng hay ngắt dòng để tách biệt văn bản và các INPUT. Nếu bạn muốn ghi "Input your name:" và có một hộp nhập tên bên cạnh, làm như sau:

```
Input your name: <BR>
<INPUT type="text" name="yourname" size="25"
maxlength="50">
```

Bạn sẽ thấy như sau:



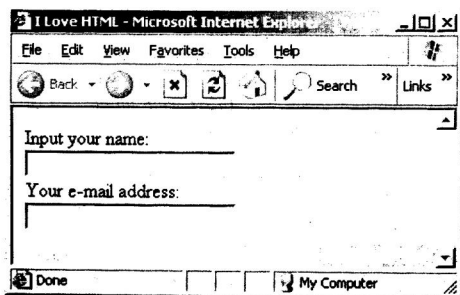
Đừng quên dùng các ngắt dòng để tách văn bản và hộp nhập liệu:

```
Input your name: <BR>
<INPUT type="text" name="yourname" size="25"
```

Căn bản thiết kế Web

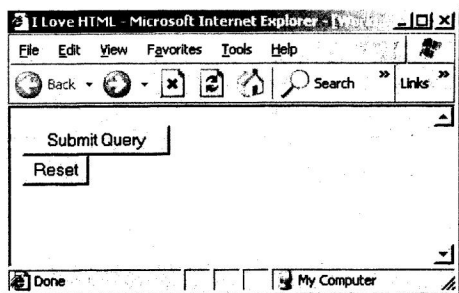
```
maxlength="50">
<BR>
Your e-mail address: <BR>
<INPUT type="text" name="emailadd" size="25"
maxlength="40">
```

Với cách này các phần nằm trên các dòng tách biệt với nhau:



Bạn muốn hiển thị button “submit” (gửi dữ liệu đi) và button “reset” (hủy các thay đổi trên form), chỉ cần thêm các tag sau:

```
<INPUT type="submit"> <BR>
<INPUT type="reset">
```



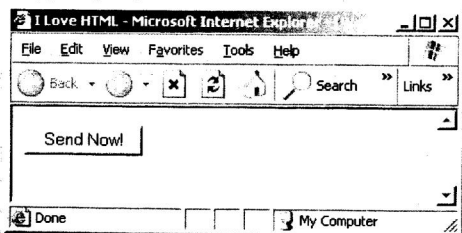
Button submit gửi các dữ liệu người sử dụng nhập vào cho chương trình cgi xử lý và gửi kết quả ngược lại cho người sử dụng.

Button "reset" xóa toàn bộ các thay đổi của người sử dụng và trở về trạng thái ban đầu.

Nếu bạn muốn các nút hiển thị khác với "Submit Query" và "Reset", bạn có thể thêm thuộc tính `value=" "` vào trong cả hai tag. Giả sử bạn muốn thay "Submit Query" bằng "Send Now!", làm như sau:

```
<INPUT type="submit" value="Send Now!">
```

Và bạn được một button như sau:



Cuối cùng, nhớ đóng form bằng tag `</FORM>` tại cuối form.

Cách để chơi nhạc và nghe âm thanh

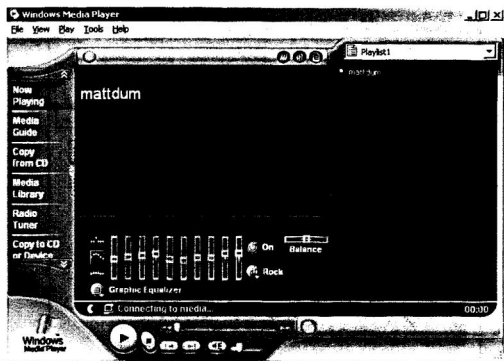
Để có âm thanh và âm nhạc trên trang web của bạn, bạn có thể làm với một trong hai cách sau: bạn có thể tạo một link đến file âm thanh mà bạn cho người sử dụng tải về và nghe nó, hoặc bạn có thể nhúng trực tiếp file âm thanh vào trang web của bạn.

Phương pháp dùng link

Đây là một ví dụ về link đến một file âm thanh:

```
<A HREF="http://www.mysite.com/sounds/mattdum.mid">A  
Cool MIDI Song</A>
```

Bạn có thể cho link tới một file âm thanh bất kỳ, như .wav, .aud, và các file âm thanh phổ biến khác. Người sử dụng sẽ tải file này về khi nhấn vào link này. Đây là những gì bạn có thể thấy:

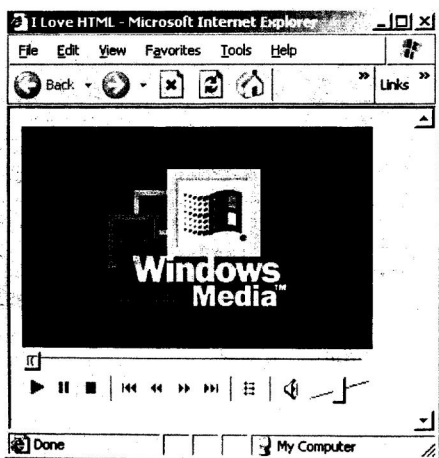


Sử dụng phương pháp nhúng

Phương pháp này là cách rất phổ biến để nghe âm thanh hay âm nhạc bởi vì nó cho phép nhiều chọn lựa hơn là một đầu link làm. Bạn có thể cho nó tự động bắt đầu, lặp lại liên tục, hay chỉ nghe một lần. Để làm điều này, bạn dùng tag <EMBED>. Tag làm việc hơi giống tag hình ảnh. Bạn cần chỉ định địa chỉ của file âm thanh và thêm các thuộc tính khác nếu cần. Ví dụ:

```
<EMBED src="http://www.mysite.com/sounds/mattdum.mid">
```

Tag này báo cho trình duyệt đặt file này lên trang web tại vị trí bạn đặt tag. Không nhất thiết phải là file .mid, bạn có thể dùng các loại file khác như .wav, .aud... Như bạn thấy, thuộc tính src="" chỉ định địa chỉ nguồn của file âm thanh, Đây là những gì bạn có thể thấy trên trang web:



Bây giờ bạn có thể thêm một số thuộc tính vào tag để điều khiển giao diện và cách nghe file âm thanh. Một số thuộc tính phổ biến:

1. **width="144"**
Chiều rộng để hiển thị màn hình file âm thanh. Tính bằng pixel.
2. **height="60"**
Chiều cao của màn hình file âm thanh hiển thị trên trang web. Tính bằng pixel, giá trị thường dùng là 60.
3. **autostart="true"**
Ra lệnh cho trình duyệt tự động phát

bài nhạc ngay khi nó được tải trên trang web. Bạn có thể đặt nó là true hay false. Giá trị mặc nhiên là false.

4. **loop="true"**

Ra lệnh cho trình duyệt lặp lại bài nhạc liên tục cho đến khi nhấn nút stop trên màn hình hiển thị. Giá trị mặc nhiên là false.

5. **hidden="true"**

Thuộc tính này báo cho trình duyệt ẩn màn hình âm thanh để người sử dụng không thấy màn hình nghe file âm thanh có các nút điều khiển. File âm nhạc sẽ được nghe như là nhạc nền. Giá trị mặc nhiên là false.

Ví dụ dùng nhiều thuộc tính:

```
<EMBED src="http://www.myplace.com/coolound.wav"
width="144" height="60" autostart="true" loop="true"
hidden="true">
```

Dùng META tag để các search engine thấy trang web của bạn.

Các tag META thường được đặt giữa các tag <HEAD> và </HEAD>. Một tag META thường có dạng như sau:

```
<HEAD>
<TITLE>META examples</TITLE>
<META name=" " content=" ">
</HEAD>
```

Thuộc tính name=" " cho biết loại tag meta nào bạn sẽ dùng. Thuộc tính content=" " sẽ là nội dung của tag bạn cung cấp cho các search engine. Xét các loại META sau:

Keywords (từ khóa)

`<META name="keywords" content="meta tags, search, homepage, web sites">`

1. name="keywords"

Nhắc nhở cho các search engine biết là một nhóm các từ khóa sẽ có trong tag này.

2. content="meta tags, search, homepage, web sites"

Đây là danh sách các từ khóa và cụm từ khóa. Trong tag này, bạn tách biệt các từ khóa hay các cụm từ khóa bằng cách đặt dấu phẩy nằm giữa chúng.

Description (Mô tả)

Tag META này mô tả trang web của bạn. Ví dụ:

`<META name="description" content="How to add META tags to your page.">`

1. name="description"

Báo cho search engine là một tả của trang web trong tag này.

2. content="How to add META tags to your page."

Đây là mô tả trang web của bạn. Có thể các search engine thể hiện mô tả của bạn trong danh sách tìm kiếm.

Cách tự động chuyển hướng người sử dụng

Nếu trang web của bạn đã di chuyển đến địa chỉ khác, hay bạn muốn tự động chuyển hướng người xem trang web, bạn có thể dùng tag meta refresh để đưa họ đến địa chỉ bạn muốn.

Nó hoạt động như thế nào, sau đây là đoạn code chuyển hướng:

```
<HTML>
<HEAD>
<META HTTP-EQUIV="refresh"
content="2;URL=http://www.yoursite.com/newpage.htm">
<TITLE>Page has moved</TITLE>
</HEAD>
<BODY>
whatever you want to write
</BODY>
</HTML>
```

Chú ý là tag refresh đứng trước tag title. Thuộc tính content=" " làm hai việc. Nó báo cho trình duyệt đợi bao nhiêu giây trước khi thực hiện *refresh* (tải lại trang web), và địa chỉ để nó chuyển hướng đến. Chú ý là dấu ngoặc kép bao quanh toàn bộ chuỗi, không phải chỉ bao quanh số giây hay địa chỉ và chúng được tách biệt bằng dấu chấm phẩy (;).

Dĩ nhiên bạn có thể hiển thị gì trong phần body cũng được. Nếu người sử dụng dùng các trình duyệt cũ, bạn nên thêm một link trong phần body để link đến trang mới:

```
<HTML>
<HEAD>
<META HTTP-EQUIV="refresh"
content="2;URL=http://www.yoursite.com/newpage.htm">
<TITLE>Page has moved</TITLE>
</HEAD>
<BODY>
This page has moved. If your browser does not
```

Căn bản thiết kế Web

automatically redirect you in a few seconds, click here to go to the new page.

</BODY>

</HTML>

Sử dụng đường gạch ngang <HR>

Giả sử bạn muốn dùng một đường ngang trong trang web, và bạn muốn nó dài bằng một hình ảnh, hoặc một dòng văn bản.... Sau đây là danh sách các thuộc tính bạn có thể dùng để thay đổi tag <HR>:

1. width="100"

Chỉ định chiều rộng của đường ngang. Có thể dùng pixel hoặc phần trăm bề rộng của trang web. Giá trị mặc nhiên là 100%.

2. align="left"

Dùng thuộc tính này để canh lề cho đường ngang là bên trái, bên phải hay chính giữa của trang web.

3. size="3"

Sử dụng thuộc tính này để chỉ định chiều cao của đường ngang. Giá trị mặc nhiên là 2.

4. noshade

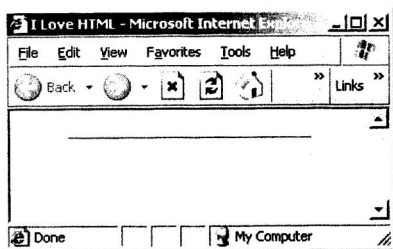
Thuộc tính này làm cho đường ngang không có bóng.

Sử dụng thuộc tính chiều rộng, chỉ cần cho nó vào tag <HR>:

<HR width="200">

Căn bản thiết kế Web

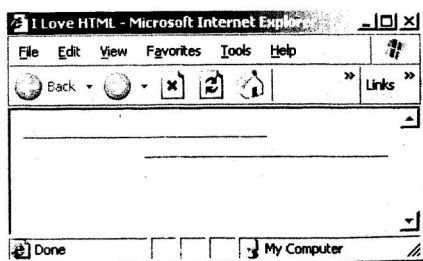
Trình duyệt sẽ cho một đường ngang như sau:



Nếu bạn muốn nó canh lề bên trái hay bên phải, cho thuộc tính vào tag:

```
<HR width="200" align="left">  
<HR width="200" align="right">
```

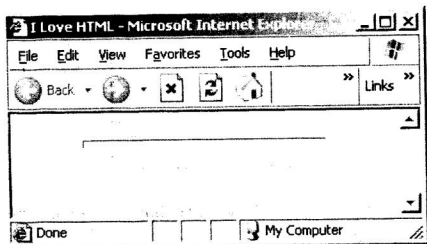
Sẽ cho hai đường ngang như sau:



Thử thay đổi kích thước của đường ngang, bằng cách thêm thuộc tính size:

```
<HR width="200" size="7">
```

Sẽ cho một đường như sau:

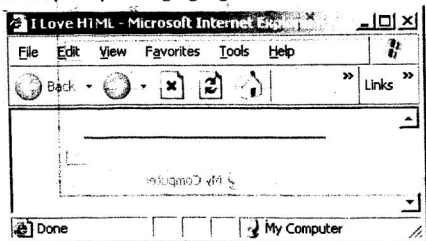


Bạn thử thay đổi size thành các số khác, 70 hoặc lớn hơn, xem có gì xảy ra.

Và cuối cùng, dùng thuộc tính noshade, sẽ cho một đường ngang không có bóng:

```
<HR width="200" noshade>
```

Sẽ nhận được đường ngang sau:



Căn bản thiết kế Web

Với trình duyệt IE4, cho phép bạn dùng thuộc tính `color=" "`. Nếu bạn dùng IE4, bạn có thể có một đường ngang màu đỏ với thuộc tính `color="red"`:

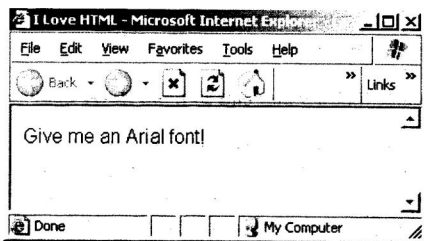
```
<HR width="200" color="red" noshade>
```

Sử dụng các phong chữ khác nhau trên trang web.

Nếu bạn muốn thay đổi font chữ trên trang web mà không cần dùng hình ảnh, bạn có thể làm điều đó với tag **FONT** trong HTML. Thay đổi font bằng cách thêm thuộc tính `face=" "` vào tag **FONT**, như sau:

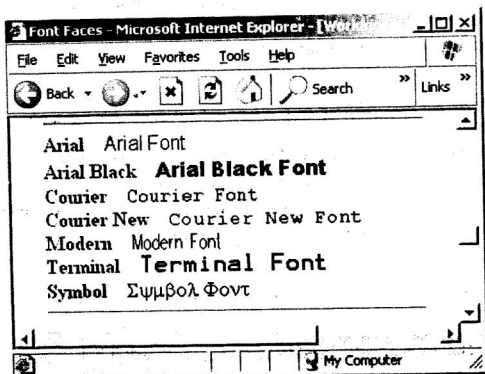
```
<FONT face="Arial">Give me an Arial font!</FONT>
```

Dòng code trên sẽ hiển thị văn bản với font Arial:



Chú ý là người sử dụng phải có font Arial được cài đặt trên máy của họ để có thể thấy được.

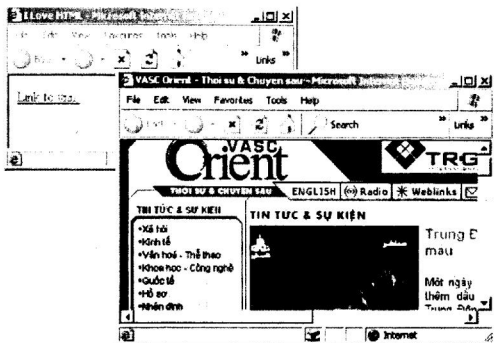
Sau đây là danh sách các font phổ biến, hầu như có trên tất cả các máy tính, sử dụng tag font `face=" "`. Bạn thử dùng một font có trên máy tính của bạn xem có gì xảy ra...



Cách tạo một link mở một cửa sổ trình duyệt mới

Để một đầu link mở một cửa sổ trình duyệt mới, bạn chỉ cần thêm thuộc tính `target= " "` vào tag link, ví dụ:

```
<A HREF="http://www.vnn.vn/" target="Resource Window">  
Link to vnn!</A>
```



Nhấn chuột vào nó. Trình duyệt sẽ mở một cửa sổ mới có nội dung của địa chỉ đặt trong link. Bạn có thể làm như trên với một hình ảnh:

```
<A HREF="http://www.vnn.vn/" target="Resource Window">  
<IMG SRC="scare.jpg">  
</A>
```

Có một cách khác để mở một cửa sổ mới với nhiều chọn lựa hơn, đó là dùng JavaScript.

Cách cho một ảnh nền cố định vào trang web.

Một ảnh nền cố định là một ảnh nền đứng một chỗ trong khi các văn bản và hình ảnh di chuyển bên trên nó khi người sử dụng cuộn trang web. Thuộc tính này chỉ ảnh hưởng trên Internet Explorer.

Để tạo một ảnh nền cố định, bạn cần dùng một ảnh nền. Sau khi chỉ định nó là ảnh nền, dùng một thuộc tính phụ trong tag BODY:

Căn bản thiết kế Web

bgproperties="fixed"

Tag BODY có thể giống như sau:

```
<BODY background="image.gif" bgproperties="fixed">
```

Cách thêm một *marquee* (dòng chữ chạy) vào trang web.

Tag *marquee* cho phép bạn cuộn văn bản ngang qua màn hình mà không cần dùng JavaScript. Nó chỉ hoạt động trên Internet Explorer. Netscape chỉ hiển thị văn bản như là các văn bản bình thường trên màn hình.

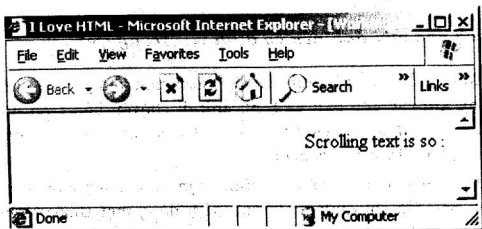
Nếu bạn dùng IE, tag để chạy *marquee* là:

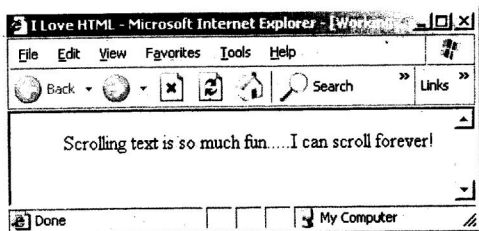
```
<MARQUEE LOOP="infinite">Your choice of text  
here</MARQUEE>
```

Thuộc tính LOOP="infinite" cho biết là văn bản được cuộn liên tục trong lúc trang web đang thể hiện trên trình duyệt. Xét ví dụ sau:

```
<MARQUEE LOOP="infinite">  
Scrolling text is so much fun.....I can scroll forever!  
</MARQUEE>
```

Đoạn code trên sẽ tạo một dòng văn bản cuộn ngang.





Bạn có thể dùng nhiều thuộc tính để thú vị hơn:

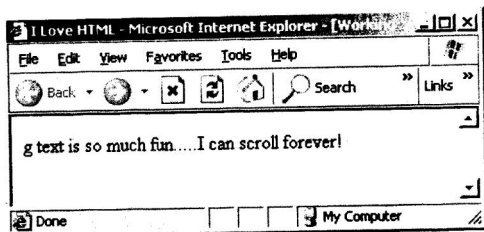
LOOP=" "

Chúng ta dùng LOOP="infinite" trong ví dụ này, bạn có thể thay đổi thành LOOP="1", LOOP="2",.... Thuộc tính này báo cho trình duyệt cuộn dòng văn bản bao nhiêu lần. LOOP="1" chỉ cuộn 1 lần.

BGCOLOR=" "

Thuộc tính này cho phép bạn thay đổi màu nền của vùng văn bản đang cuộn. Ví dụ:

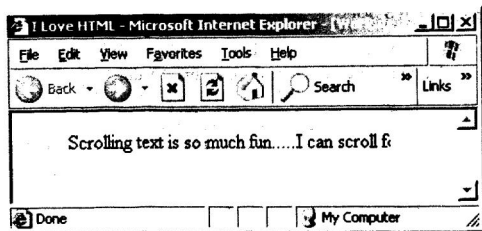
```
<MARQUEE LOOP="infinite" BGCOLOR="yellow">
Scrolling text is so much fun.....I can scroll forever!
</MARQUEE>
```



WIDTH=" "

Thuộc tính này quy định chiều rộng của marquee. Có thể là pixel, hoặc phần trăm của chiều rộng trang web. Ví dụ:

```
<MARQUEE LOOP="infinite" BGCOLOR="yellow" WIDTH="300">  
Scrolling text is so much fun.....I can scroll forever!  
</MARQUEE>
```



Bảng trong HTML

Cách cho một bảng (table) vào trang web.

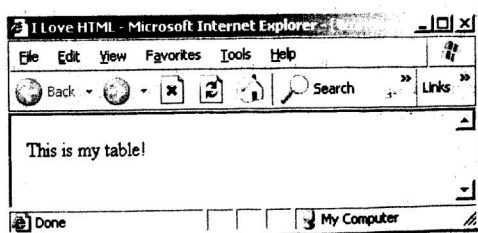
Bạn muốn tạo một bảng, hãy dùng các tag TABLE:

```
<TABLE>
.....contents of table.....
</TABLE>
```

Tag <TABLE> bắt đầu một bảng, đặt những gì bạn muốn bên trong nó, và kết thúc bằng tag </TABLE>. Để bắt đầu cho nội dung vào bảng, bạn cần tag <TD> (table data, là dữ liệu đặt sau tag này). Và kết thúc vùng dữ liệu bằng tag </TD>. Sau đây là một bảng chỉ có một ô:

```
<TABLE>
<TD>
This is my table!
</TD>
</TABLE>
```

Trình duyệt sẽ hiển thị như sau:

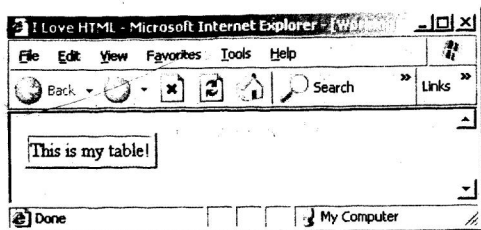


Căn bản thiết kế Web

Bảng này không có đường viền, để thêm đường viền chỉ cần thêm thuộc tính border vào tag <TABLE>, ví dụ:

```
<TABLE border="2">  
<TD>  
This is my table!  
</TD>  
</TABLE>
```

Và bây giờ bảng có một đường viền bao quanh nó:

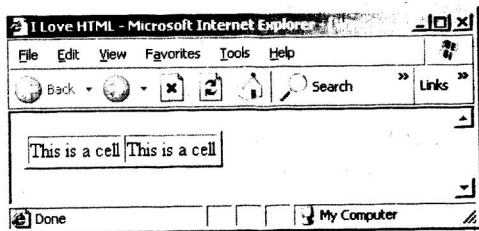


Bạn có thể đặt đường viền to hay nhỏ tùy ý, chỉ cần thay đổi số bên trong dấu ngoặc kép. Nếu bạn đặt border="0", bạn sẽ có một bảng không có đường viền.

Bạn có thể muốn bảng có nhiều ô. Để thêm một ô trên cùng một dòng, chỉ cần thêm một tag <TD>, như sau:

```
<TABLE border="2">  
<TD>  
This is a cell  
</TD>  
<TD>  
This is a cell  
</TD>  
</TABLE>
```


và ta có hai ô:



Nếu bạn muốn có một dòng mới trong bảng, dùng tag

`<TR>` và `</TR>`:

```
<TABLE border="2">
```

```
<TD>
```

```
This is a cell
```

```
</TD>
```

```
<TD>
```

```
This is a cell
```

```
</TD>
```

```
<TR>
```

```
<TD>
```

```
This is the new row
```

```
</TD>
```

```
<TD>
```

```
I'm on the new row, too!
```

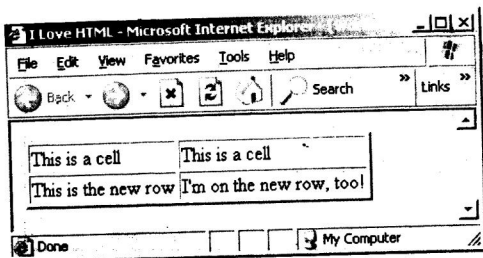
```
</TD>
```

```
</TR>
```

```
</TABLE>
```

Căn bản thiết kế Web

Bây giờ có hai dòng, mỗi dòng có hai ô:



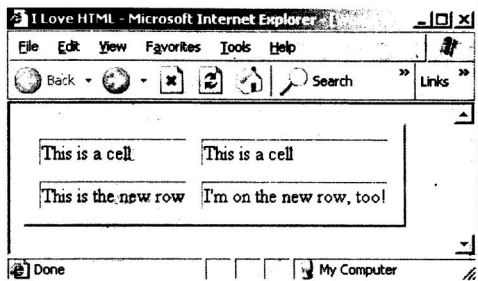
Có hai thuộc tính bạn có thể thêm vào tag `<TABLE>` để tăng khoảng cách giữa các ô :

1. **cellspacing=" "**
Dùng thuộc tính này để thêm không gian quanh mỗi ô. Đặt một số bên trong dấu ngoặc kép.
2. **cellpadding=" "**
Dùng thuộc tính này để tăng thêm không gian trong mỗi ô.

Ví dụ, đặt thuộc tính `cellspacing` vào bảng trên, và cho nó giá trị 12:

```
<TABLE border="2" cellspacing="12">
```

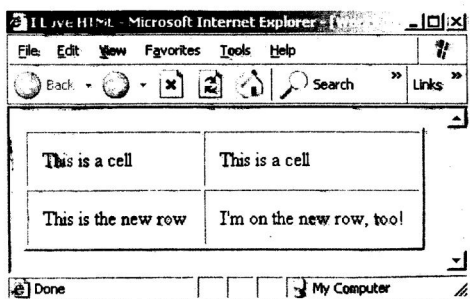
Bạn sẽ nhận được một bảng như sau:



Và bây giờ đặt thuộc tính cellpadding là 12:

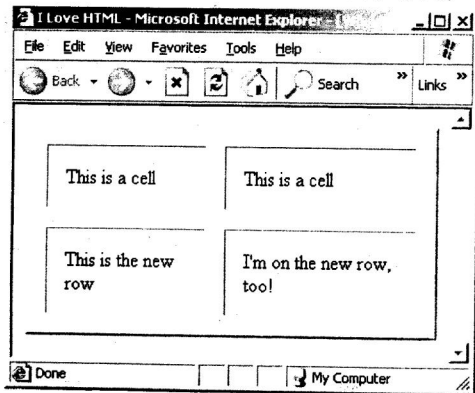
```
<TABLE border="2" cellpadding="12">
```

Bảng sẽ trông như sau:



Và bạn có thể dùng cả hai thuộc tính cùng lúc:

```
<TABLE border="2" cellspacing="15" cellpadding="15">
```



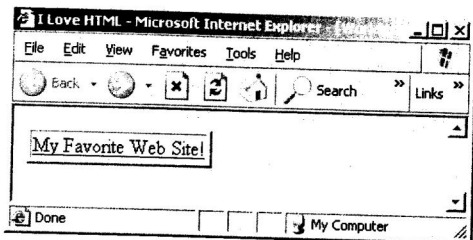
Bạn có thể đặt bất cứ thứ gì vào trong các ô. Có thể là link, hình ảnh, tiêu đề, đoạn văn bản...

Để dùng một link bên trong một ô, chỉ cần đặt tag link bên trong tag `<TD>` và `</TD>`:

```
<TD>  
<A HREF="http://www.vnn.vn">My Favorite Web Site!</A>  
</TD>
```

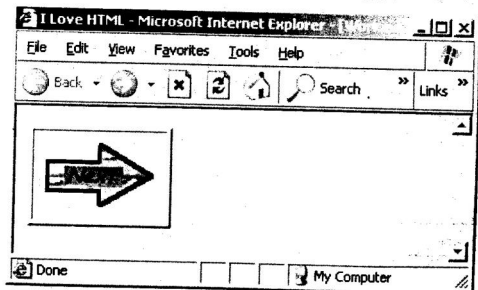
Căn bản thiết kế Web

Bạn sẽ có một link như sau:



Để đặt một hình ảnh vào trong một ô, làm tương tự:

```
<TD>  
<IMG SRC="next.jpg">  
</TD>
```



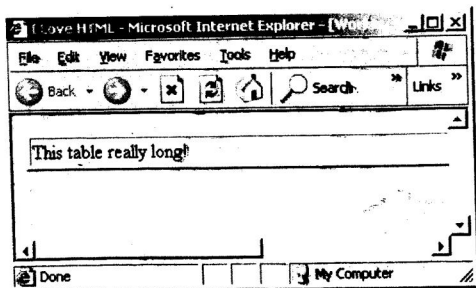
Chú ý là mọi thứ đều được canh lề bên trái của mỗi ô, đây là canh lề mặc nhiên.

Cách dùng các thuộc tính khác của bảng

Trước khi tìm hiểu các thuộc tính khác, hãy xem cách định nghĩa chiều rộng của bảng theo kích thước bạn muốn. Để làm điều này, thêm thuộc tính `width=""` vào tag `<TABLE>`. Đặt số pixel chiều rộng của bảng vào trong dấu ngoặc kép. Nếu bạn muốn bảng rộng 600 pixel, làm như sau:

```
<TABLE width="600" border="2">  
<TD>  
This table really long!  
</TD>  
</TABLE>
```

Bảng này chỉ chứa một dòng văn bản, nhưng nó dài 600 pixel vì bạn qui định như vậy.



Để canh lề cho nội dung bên trong các ô, bạn thêm thuộc tính `align=""` vào tag `<TD>`. Bạn có thể dùng thuộc tính này với ba giá trị:

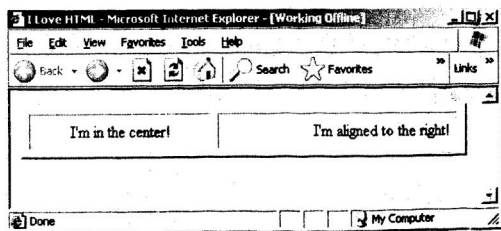
Căn bản thiết kế Web

1. **align="left"**
Canh nội dung theo lề trái.
2. **align="center"**
Canh nội dung chính giữa.
3. **align="right"**
Canh nội dung bên phải.

Xem một bảng có dùng các thuộc tính này:

```
<TABLE width="450" border="2" cellspacing="7"  
cellpadding="7">  
<TD align="center">  
I'm in the center!  
</TD>  
<TD align="right">  
I'm aligned to the right!  
</TD>  
</TABLE>
```

Ô đầu tiên có văn bản nằm chính giữa, ô thứ hai có văn bản nằm bên phải:



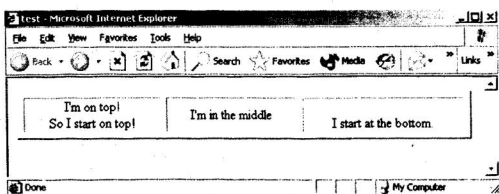
Bạn cũng có thể dùng một thuộc tính khác cho các ô, canh lề theo chiều dọc, thuộc tính `valign=" "`. Các giá trị của nó:

1. **valign="top"**
Canh nội dung trên đầu ô.
2. **valign="middle"**
Canh nội dung nằm giữa phần đầu và phần cuối.
3. **valign="bottom"**
Canh nội dung nằm cuối ô.

Sau đây là một ví dụ dùng các thuộc tính canh lề theo chiều dọc:

```
<TABLE width="550" border="2" cellspacing="7" cellpadding="0">
<TD align="center" valign="top">
I'm on top! <br>
So I start on top! </TD>
<TD align="center" valign="middle">
I'm in the middle
</TD>
<TD align="center" valign="bottom">
I start at the bottom.
</TD>
</TABLE>
```

Kết quả như sau:



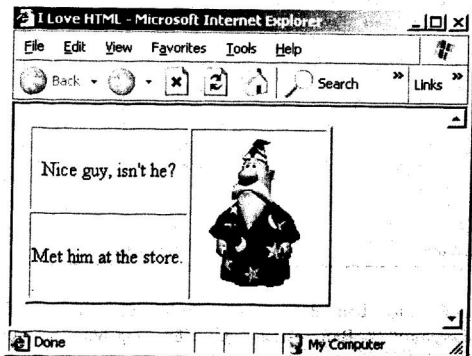
Căn bản thiết kế Web

Canh lề chiều dọc chỉ hữu dụng đối với bảng có các ô không đều nhau. Nếu ô đầu tiên trong bảng trên chỉ có một dòng, thì bạn sẽ không thấy gì khác biệt với bảng không có các thuộc tính valign.

Làm thế nào để tạo các bảng không đều nhau, nối các dòng và các cột? Chúng ta sẽ xét thêm một số thuộc tính của tag <TD>, rowspan và colspan:

1. **rowspan="** "
 Định nghĩa số dòng mà ô sẽ nối ra. Đặt số vào dấu ngoặc kép.
2. **colspan="** "
 Định nghĩa số cột mà ô sẽ nối ra.

Giả sử bạn muốn tạo một bảng như sau:



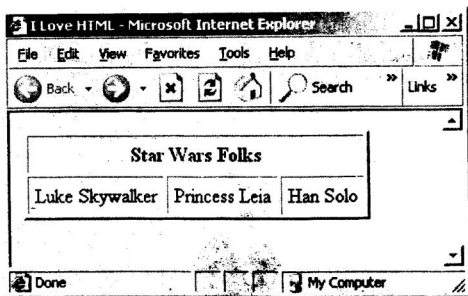
Cần bản thiết kế Web

Để tạo bảng như trên, bạn phải dùng thuộc tính **rowspan** trong ô chứa ảnh. Bạn đặt **rowspan="2"** và ô chứa hình ảnh sẽ nói ra hai dòng. Code như sau:

```
<TABLE border="2">
<TD align="center">
Nice guy, isn't he?
</TD>
<TD rowspan="2" align="center">
<IMG SRC="../../images/scare.jpg">
</TD>

<TR>
<TD align="center">
Met him at the store.
</TD>
</TR>
</TABLE>
```

Giả sử bạn muốn tạo bảng như sau:



Lúc này bạn dùng thuộc tính **colspan** và đặt **colspan="3"**. Bạn sẽ nói ô đầu tiên rộng bằng ba ô của dòng dưới:

Căn bản thiết kế Web

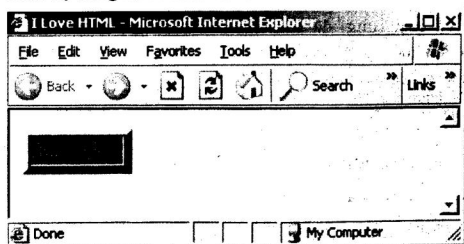
```
<TABLE border="2" cellpadding="5">
<TD colspan="3" align="center">
<B>Star Wars Folks</B>
</TD>

<TR>
<TD align="center">
Luke Skywalker
</TD>
<TD align="center">
Princess Leia
</TD>
<TD align="center">
Han Solo
</TD>
</TR>
</TABLE>
```

Giả sử bạn muốn thay đổi màu nền của các ô trong bảng. Bạn có thể thay đổi màu nền của toàn bộ bảng, của mỗi dòng, hay của từng ô. Để thay đổi màu nền của toàn bộ bảng, thêm thuộc tính bgcolor=" " vào tag TABLE:

```
<TABLE border="5" bgcolor="gray">
<TD>
Gray Rules!
</TD>
</TABLE>
```

Toàn bộ bảng sẽ có màu xám.

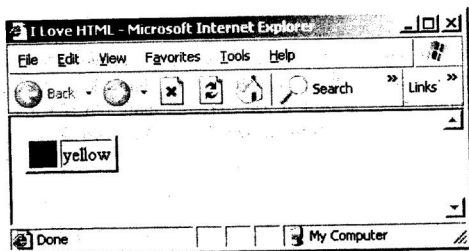


Căn bản thiết kế Web

Để thay đổi màu của rưng ô, thêm thuộc tính bgcolor vào mỗi tag <TD> mà bạn muốn thay đổi:

```
<TABLE width="75" border="2">
<TD bgcolor="red">
red
</TD>
<TD bgcolor="yellow">
yellow
</TD>
</TABLE>
```

Bây giờ có hai ô với hai màu khác nhau:

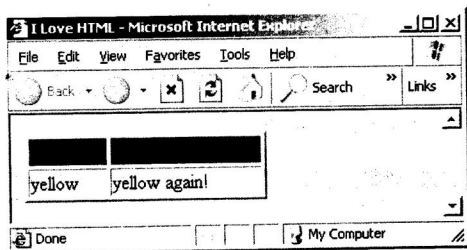


và cuối cùng, để thay đổi màu của toàn bộ một dòng, thêm thuộc tính bgcolor vào tag <TR>:

```
<TABLE width="200" border="2" >
<TR bgcolor="red">
<TD> red </TD>
<TD> red again... </TD>
</TR>
<TR bgcolor="yellow">
<TD> yellow </TD>
<TD> yellow again! </TD>
</TR>
</TABLE>
```

Căn bản thiết kế Web

Ta có hai dòng với hai màu khác nhau:



Đặt các bảng trong một bảng lớn hơn

Giả sử bạn muốn đặt hai bảng trên cùng một dòng. Bạn có thể làm điều này bằng cách đặt hai bảng vào một bảng lớn hơn:

Ta bắt đầu bằng việc đặt một bảng trong một bảng khác:

```
<TABLE width="400" border="6"> <!--starts the big table-->
```

```
<TD align="center"> <!--starts the first cell of the big table-->
```

```
<TABLE width="300" border="2"> <!--we use another table as the cell data for the big table-->
```

```
<TD align="center"> <!--starts the small table inside-->
```

```
I'm inside the small table! <!--the contents of the first cell of the small inside table-->
```

```
</TD> <!--ends the table cell of the small inside
```

Căn bản thiết kế Web

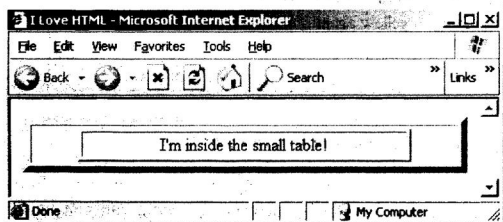
table---->

</TABLE> <!--ends the small inside table----->

</TD> <!--ends the cell of the big table which contained the small table----->

</TABLE> <!--ends the big table----->

Sau đây là kết quả nhận được:

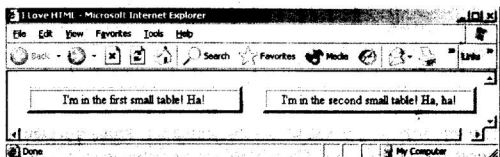


Để ẩn đi bảng bên ngoài, bạn đặt thuộc tính border của bảng lớn bằng 0. Ví dụ:

```
<TABLE width="600" border="0">
<TD align="center">
    <TABLE width="275" border="4">
    <TD align="center">
        I'm in the first small table! Ha!
    </TD>
    </TABLE>
</TD>
<TD align="center">
    <TABLE width="275" border="4">
    <TD align="center">
        I'm in the second small table! Ha, ha!
    </TD>
    </TABLE>
</TD>
</TABLE>
```

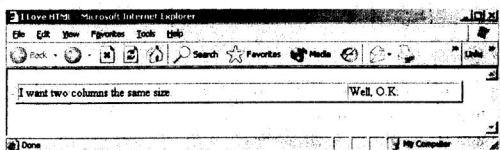
Căn bản thiết kế Web

Bây giờ bạn có hai bảng trên cùng một dòng:



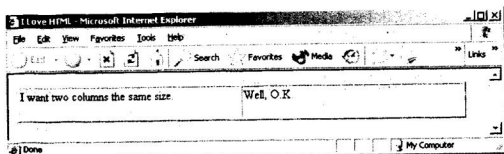
Canh lề các bảng với một ảnh ẩn

Vẫn còn một vấn đề với bảng, có thể gây rắc rối cho bạn. Vấn đề đó là chiều rộng và chiều cao của các cột hay các dòng được canh theo đối tượng dài nhất. Kết quả là khi bạn tạo một bảng và hiển thị nó trên trình duyệt và thấy có cột lớn, cột nhỏ, ví dụ:



Như bạn thấy, cột bên trái chiếm quá nhiều không gian, và cột bên phải thì thiếu. Một phương pháp để giải quyết vấn đề này là mượn dùng hình ảnh có kích thước 1x1 pixel. Để làm điều này, trước tiên bạn tạo một ảnh 1x1 pixel trong suốt hay có cùng màu với màu nền. Sau đó ép các ô của bảng có chiều dài bằng nhau, giống như sau:

Căn bản thiết kế Web



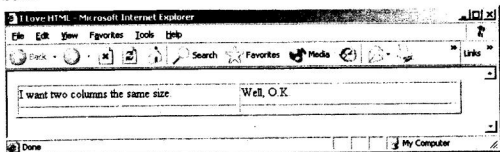
Đoạn code tạo bảng trên như bên dưới. Như bạn thấy, *spacer.jpg* là ảnh 1x1. Chúng ta thay đổi bề rộng của ảnh trong code html thành 290 pixel trong mỗi ô, sẽ cho ta bảng như trên:

```
<TABLE width="600" border="2">
<TD align="left">
I want two columns the same size.
<BR>
<IMG SRC="spacer.jpg" width="290" height="1">
</TD>
<TD align="left">
Well, O.K.
<BR>
<IMG SRC="spacer.jpg" width="290" height="1">
</TD>
</TABLE>
```

Có một cách khác để làm như trên, nhưng có một đường ngang trong ô, đó là dùng tag <hr>:

```
<TABLE width="600" border="2">
<TD align="left">
I want two columns the same size.
<HR width="290">
</TD>
<TD align="left">
Well, O.K.
<HR width="290">
</TD>
</TABLE>
```


Dĩ nhiên là bạn sẽ có một đường ngang bên dưới văn bản:



Dùng các bảng lồng nhau thay vì dùng mẹo hình ảnh

Bạn tạo một bảng và muốn các cột của chúng có chiều rộng bằng nhau. Bạn có thể dùng hình ảnh 1x1 pixel, nhưng nó có một vấn đề khác là có một dòng trống bên dưới. Bạn có thể dùng các bảng lồng nhau, và thêm các thuộc tính cho các bảng bên trong để tạo bảng bên ngoài có các cột bằng nhau:

Trước tiên tạo một bảng lớn, có bề rộng 500 pixel, và nhớ đặt các giá trị cellspacing và cellpadding bằng 0. Đoạn code sau tạo một bảng bên ngoài với độ rộng đường viền bằng 2:

```
<TABLE width="500" border="2" cellspacing="0"
cellpadding="0">
<TD align="left">
</TD>
</TABLE>
```

Bây giờ chúng ta cần tạo hai bảng bên trong, sẽ chứa nội dung. Và chúng ta sẽ thêm các thuộc tính width cho các tag TABLE của các bảng bên trong. Ta sẽ đặt cả hai bảng này đều có chiều rộng là 250 pixel, và border bằng 0:

Căn bản thiết kế Web

```
<TABLE width="500" border="2" cellspacing="0"
cellpadding="0">
<TD align="left">

<!-------start your first inside table here----->
<TABLE width="250" border="0">
<TD align="left">
I want two columns the same size.
</TD>
</TABLE>
<!-------end of your first inside table----->

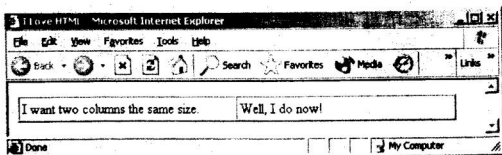
</TD> <!-------this ends your first table cell in the
outside table----->

<TD align="left"> <!-------start your second table cell
in the outside table----->

<!-------start your second inside table here----->
<TABLE width="250" border="0">
<TD align="left">
Well, I do now!
</TD>
</TABLE>
<!-------end of your second inside table----->

</TD> <!-------this ends your second table cell in the
outside table----->
</TABLE>
```

Kết quả là một bảng có hai cột bằng nhau:



Bạn có thể dùng cách này để tạo bộ khung cho trang web của bạn, bằng cách dùng các bảng có border bằng 0.

Sử dụng các Frame

Cách thêm các khung (frame) vào trang web

Trước khi sử dụng các frame trên trang web của bạn, bạn cần biết các frame làm việc như thế nào. Một trang web có các frame thực sự là một trang web được chia ra thành hai hay nhiều phần, mỗi phần chứa một trang html của riêng nó. Sau đây là một ví dụ về một trang có hai frame:

Dây là một trang html.	Dây là một trang html khác.
------------------------	-----------------------------

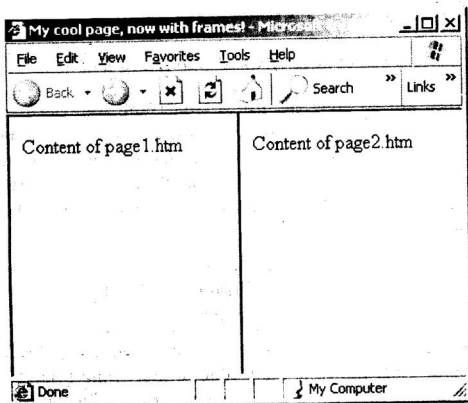
Như đã nói, cả hai frame thực sự là các trang html. Trang để chứa hai frame này cũng là một trang html. Ta bắt đầu xem cách trang chính tạo các frame như thế nào. Tạo các frame bằng cách dùng tag <FRAMESET> thay vì dùng tag <BODY>. Sau đây là một ví dụ html tạo ra hai frame, chia trang web thành hai phần:

```
<HTML>
<HEAD>
<TITLE>My cool page, now with frames!</TITLE>
</HEAD>

<FRAMESET cols="50%,50%">
<FRAME SRC="page1.htm">
<FRAME SRC="page2.htm">
</FRAMESET>

</HTML>
```

Kết quả như sau:



Ta xét ví dụ trên:

1. <FRAMESET>

Tag này báo cho trình duyệt web là tạo các frame thay vì tạo một trang web bình thường.

2. cols="50%,50%"

Thuộc tính này trong tag FRAMESET báo cho trình duyệt chia trang web thành hai cột. Trong trường hợp này mỗi cột chiếm 50% không gian của màn hình. Bạn có thể thay đổi thành phần trăm nếu muốn. Bạn cũng có thể dùng pixel thay vì dùng phần trăm. Nếu dùng

Cần bản thiết kế Web

phần trăm phải thêm dấu % phía sau số, nếu không trình duyệt xem nó như là giá trị pixel.

3. **<FRAME SRC="page1.htm">**

Tag này báo cho trình duyệt địa chỉ của trang web trong frame trái nhất.

4. **<FRAME SRC="page2.htm">**

Tag này chỉ định địa chỉ của frame kế tiếp, tính từ trái sang phải.

Giả sử bạn muốn có ba frame trên trang web, bạn cần thay đổi tag FRAMESET và thêm một tag FRAME SRC khác cho frame thứ ba, giống như sau:

```
<FRAMESET cols="33%,33%,33%">  
<FRAME SRC="page1.htm">  
<FRAME SRC="page2.htm">  
<FRAME SRC="page3.htm">  
</FRAMESET>
```

Đoạn code trên tạo ba cột trên trang web, mỗi cột chiếm 33% chiều rộng của trang web.

Làm sao để tạo các frame theo thứ tự từ trên xuống dưới. Giả sử bạn muốn các frame giống như sau:

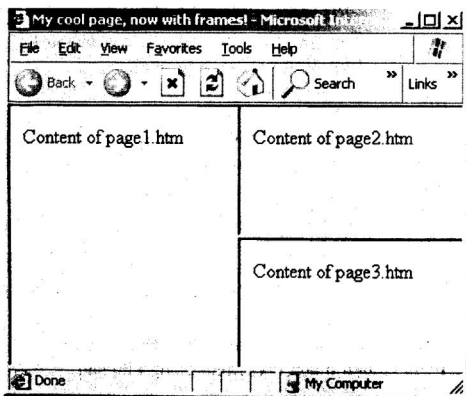
Frame 1	Frame2
	Frame 3

Ta cần thêm một tag FRAMESET nữa, nhưng lần này ta dùng thuộc tính "rows". Sau đây là đoạn code tạo một trang giống như trên:

Căn bản thiết kế Web

```
<FRAMESET cols="50%,50%">
  <FRAME SRC="page1.htm">

  <FRAMESET rows="50%,50%">
    <FRAME SRC="page2.htm">
    <FRAME SRC="page3.htm">
  </FRAMESET>
</FRAMESET>
```



Thuộc tính `rows` được hiểu từ trên xuống dưới, giống như là thuộc tính `cols` hiểu từ trái qua phải. Bạn có thể tạo bao nhiêu cột hay dòng bạn muốn, nhưng nhớ lồng các tag `frameset` đúng theo cách mà bạn muốn trang web thể hiện. Trong ví dụ trên:

- Tag `<FRAMESET>` đầu tiên báo cho trình duyệt chia trang web thành hai cột.

- Tag FRAME SRC tiếp theo báo cho trình duyệt là cột đầu tiên sẽ hiển thị file *page1.htm*.
- Tag FRAMESET tiếp theo được lồng trong tag FRAMESET thứ nhất. Tag này báo cho trình duyệt chia cột thứ hai thành hai dòng, thay vì hiển thị một trang html khác.
- Hai tag FRAME SRC tiếp theo báo cho trình duyệt là frame phía trên hiển thị trang *page2.htm* và hiển thị trang *frame3.htm* trong frame bên dưới.
- Nhớ đóng tất cả các tag FRAMESET.

Cách liên kết giữa các frame, thay đổi thuộc tính của frame, và sử dụng tag noframes.

Bây giờ bạn có một trang với các frame. Nhưng khi bạn đặt một link bên trong một frame, nội dung của địa chỉ bạn link tới chỉ xuất hiện trong frame đó. Đúng vậy, bởi vì một link trong một frame sẽ mặc nhiên hiển thị trang kế tiếp trong frame có link đó. Bạn muốn có các link hiển thị các trang ở các frame khác. Mẹo để làm điều này là đặt tên cho các frame, và chỉ định frame đích (target) trong tag link. Sau đây là cách làm:

Trước tiên cho thuộc tính name = " " vào mỗi frame, như sau:

```
<FRAMESET cols="20%,80%">  
<FRAME SRC="page1.htm" name="left_frame">  
<FRAME SRC="page2.htm" name="right_frame">  
</FRAMESET>
```

Mỗi frame có một tên, và bạn có thể dùng tên của frame như là đích (target) trong tag link. Giả sử, bạn có

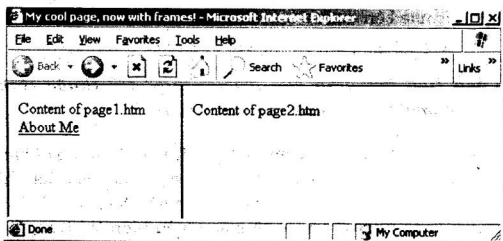
Căn bản thiết kế Web

một link ở left_frame (*page1.htm*). Nếu bạn muốn hiển thị địa chỉ của link ở right_frame, bạn cần thêm thuộc tính `target="frame_name"` bên trong tag link. Ví dụ sau là một link bên trái, khi nhấn vào nó sẽ hiển thị ở frame bên phải:

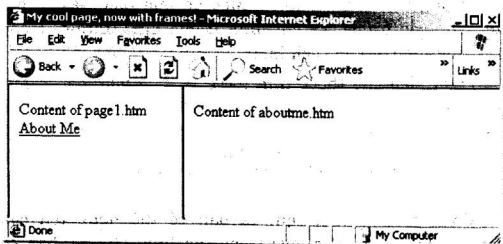
```
<A HREF="aboutme.htm" target="right_frame">About Me</A>
```

Frame bên trái vẫn không thay đổi, trong khi frame bên phải hiển thị nội dung.

Trước khi nhấn vào link:



Sau khi nhấn vào link:



Làm sao để một link từ một frame con thay đổi toàn bộ trang web, mà không phải là thay đổi trong chính nó? Bạn chỉ cần thêm thuộc tính target như sau:

```
<A HREF="http://www.someplace.com" target="_top">See my friend's Homepage!</A>
```

target="_top"

Báo cho trình duyệt hiển thị trang web trong màn hình chính. Bạn nhớ đặt dấu gạch ngang bên dưới từ top.

Sau đây là một số thuộc tính bạn có thể thêm vào các tag <FRAME> cụ thể để giúp bạn điều khiển dao diện của frame:

1. scrolling="no"

Thuộc tính này chỉ định frame có hiển thị thanh cuộn bên phải hay không. Nếu không có thuộc tính này frame sẽ tự động có hiển thị hay không, phụ thuộc vào nội dung của frame. Nếu đặt là "yes", frame luôn luôn có thanh cuộn. Nếu đặt là "no" frame sẽ không bao giờ hiển thị thanh cuộn.

2. border="2"

Cho phép chỉ định độ rộng của đường viền.

3. resize="no"

Thuộc tính này quyết định có cho phép người sử dụng định lại kích thước của frame hay không. Nếu đặt là "no" frame sẽ không thể thay đổi kích thước. Giá trị mặc nhiên là "yes".

4. noresize

Bạn có thể dùng thuộc tính này với

Căn bản thiết kế Web

resize="no", bởi vì không phải tất cả trình duyệt đều hỗ trợ thuộc tính resize="no".

5. **marginwidth="2"** và **marginheight="2"**

Các thuộc tính này cho phép bạn quyết định lề của frame và nội dung trong frame.

Làm cách nào để hiển thị thông báo cho những người dùng trình duyệt không hỗ trợ frame?. Sau tag đóng của FRAMESET, thêm tag như sau:

```
<NOFRAMES>
```

Hi, your browser is really old. If ya wanna view this page, get a newer browser or click [here](#) to view my alternate page.

```
</NOFRAMES>
```

Làm cho các thanh cuộn và các đường viền biến mất

Để tạo các frame không thấy được, bạn cần thêm một số thuộc tính vào tag FRAMESET. Sau đây là code tạo hai frame, và đường viền giữa chúng sẽ biến mất:

```
<FRAMESET cols="20%,80%" border="0" framespacing="0"
frameborder="0">
```

```
<FRAME SRC="page1.htm" name="left_frame" scrolling="no">
```

```
<FRAME SRC="page2.htm" name="right_frame">
```

```
</FRAMESET>
```

Như bạn thấy, có ba thuộc tính mới trong tag FRAMESET:

1. **border="0"**

Đặt giá trị border bằng 0 để nó không xuất hiện.

Căn bản thiết kế Web

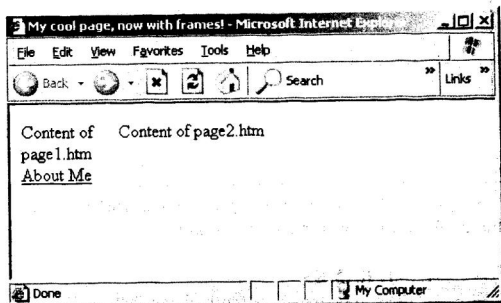
2. **frameborder="0"**

Đặt frameborder bằng 0 để chúng không hiển thị.

3. **framespacing="0"**

Đặt khoảng cách giữa các frame bằng 0, để không có khoảng cách giữa chúng.

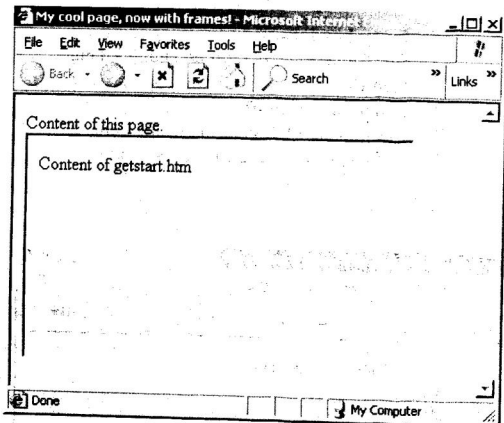
Kết quả:



Cách chèn frame nội (inline) trong trang HTML.

Để tạo một inline frame, dùng tag sau:

```
<IFRAME src="getstart.htm" width="350" height="200"  
frameborder="10"> </IFRAME>
```



Nó giống như tất cả các frame thông thường khác, nhưng nó nằm trong trang web của bạn.

1. **<IFRAME>**

Tag này báo cho trình duyệt biết một inline frame sẽ nằm ở đây.

2. **src="getstart.htm"**

Thuộc tính này chỉ định địa chỉ của trang web mà frame này sẽ hiển thị.

3. **width="350"**

Quy định chiều rộng của frame, tính bằng pixel.

4. **height="200"**

Quy định chiều cao của frame, tính bằng pixel.

5. **frameborder="10"**

Cho phép bạn cho frame một đường viền to hay nhỏ tùy ý.

Bạn cũng có thể dùng các link như các frame khác. Đặt cho inline frame một cái tên, và qui định target của link trong trang bên ngoài.

Trước tiên, thêm thuộc tính name vào tag IFRAME:

```
<IFRAME src="getstart.htm" name="inside_frame"
width="350" height="200" frameborder="10">
</IFRAME>
```

Đặt link trong trang bên ngoài, cho phép thay đổi frame bên trong:

```
<A HREF="table1.htm" target="inside_frame">See how to
build a table!</A>
```

Bạn cũng có thể thêm thuộc tính scrolling="no" để frame không hiển thị thanh cuộn.

Cách cho một image map vào trang web.

Ví dụ, bạn muốn tạo một hình ảnh mà khi người sử dụng nhấn chuột vào phía bên trái sẽ mở một trang web, và khi người sử dụng nhấn chuột bên phải thì sẽ mở một trang khác. Cả hai liên kết này đều nằm trên cùng một hình ảnh, chúng ta gọi là image map.

Làm sao để tạo hình ảnh như trên? Trước tiên bạn phải có một hình ảnh bạn muốn sử dụng. Chọn bất cứ hình nào bạn thích. Và tiếp theo là bạn phải biết chiều rộng và chiều cao của hình ảnh mà bạn sẽ dùng.

Bây giờ bạn cần đặt hình ảnh vào trang web. Để làm điều này, bạn dùng tag hình ảnh, nhưng với một thuộc tính mới: USEMAP.

Căn bản thiết kế Web

```
<IMG SRC="eximapl.gif" width="200" height="40"  
border="0" alt="imagemap" USEMAP="#mymap">
```

Thuộc tính `USEMAP="#mymap"` báo cho trình duyệt dùng một ảnh xạ (map) trên hình ảnh, ảnh xạ có tên "mymap". Chủ ý là cách dùng dấu # trước tên của ảnh xạ. Và cũng chú ý là chúng ta có định nghĩa chiều rộng và chiều cao của hình ảnh. Bây giờ xem cách định nghĩa một map. Chúng ta đặt định nghĩa này ở trong trang html, nên đặt ngay sau tag hình ảnh để dễ hiểu.

```
<MAP name="mymap">  
<AREA SHAPE="RECT" COORDS="0,0,99,40" HREF="page1.htm">  
<AREA SHAPE="RECT" COORDS="100,0,200,40"  
HREF="page2.htm">  
<AREA SHAPE="DEFAULT" HREF="http://www.vnn.vn">  
</MAP>
```

Xét các thuộc tính:

<MAP name="mymap">

Định nghĩa phần ảnh xạ của hình ảnh, và đặt tên cho ảnh xạ.

**<AREA SHAPE="RECT" COORDS="0,0,99,40"
HREF="page1.htm">**

Tag AREA định nghĩa một vùng của hình ảnh sẽ được dùng như là một link. Thuộc tính SHAPE báo cho trình duyệt biết là vùng đó có dạng hình học gì. Để đơn giản, ta dùng "RECT", là hình chữ nhật. Thuộc tính COORDS là nơi định nghĩa các đỉnh của vùng link. Bởi vì hình chữ nhật bắt đầu tại đỉnh trên bên trái của hình ảnh nên tọa độ là (0,0) pixel. Hai số tiếp theo định nghĩa đỉnh kết thúc của hình chữ nhật. Đỉnh này là đỉnh dưới bên phải của hình chữ nhật. Trong trường hợp này, hình ảnh của chúng ta có

kích thước 200x40. Chúng ta muốn đỉnh kết thúc của hình chữ nhật này là nằm giữa cạnh dưới của hình ảnh. Một nửa của 200 là 100, nhưng ta dùng 99 bởi vì 100 chỉ được dùng 1 lần. Chúng ta sẽ dùng 100 cho hình chữ nhật thứ hai. Do đó góc dưới của hình chữ nhật thứ nhất là 99 và 40. Thuộc tính HREF báo cho trình duyệt biết vị trí đi đến khi người sử dụng nhấn chuột vào hình chữ nhật này.

```
<AREA SHAPE="RECT" COORDS="100,0,200,40"
```

```
HREF="page2.htm">
```

Tương tự như hình chữ nhật trước, nhưng nó tạo hình chữ nhật thứ hai.

```
<AREA SHAPE="DEFAULT"
```

```
HREF="http://www.vnn.vn ">
```

Giá trị DEFAULT thực sự không phải là một hình học, nó chỉ dùng cho các vùng chưa được ánh xạ. Chúng ta không để vùng trống nào trong ví dụ trên.

```
</MAP>
```

Kết thúc phần định nghĩa map.

Cách bỏ các đường gạch dưới của các link

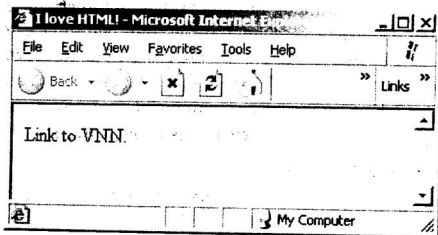
Bạn muốn có một link không có gạch dưới, hoặc một trang web có các link không có gạch dưới. Để tạo một link không có dưới, bạn cần dùng thuộc tính style="" trong tag link. Và thuộc tính bên trong ta sẽ dùng là text-decoration, và đặt giá trị của nó là none:

```
style="text-decoration:none"
```

Cho thuộc tính trên vào tag link, như sau:

```
<A HREF="http://www.vnn.vn" style="text-decoration:none">Link to VNN.</A>
```

Bạn sẽ có một link như sau:



Đối với toàn bộ link trong trang web

Nếu bạn muốn tất cả các link trong trang web không có gạch dưới, bạn có thể dùng tag *style sheet* bên trong tag <HEAD> </HEAD>. Đoạn code như sau:

```
<STYLE type="text/css">
<!--
A { text-decoration:none }
-->
</STYLE>
```

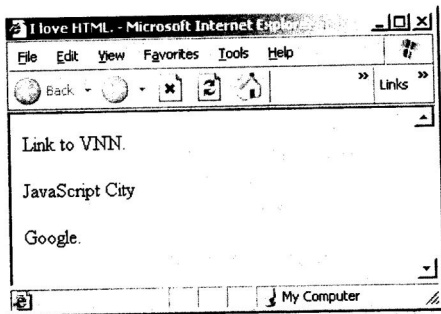
Đặt đoạn code trên giữa hai tag <HEAD> </HEAD>, và sau đó tạo các link bình thường. Style sheet trong phần head sẽ làm cho các link không có gạch dưới:

```
<HTML>
<HEAD>
<STYLE type="text/css">
<!--
A { text-decoration:none }
-->
</STYLE>
</HEAD>
```



```
<BODY>  
<P>  
<A HREF="http://www.vnn.vn">Link to VNN. </A>  
<P>  
<A HREF="http://www.javascriptcity.com">JavaScript  
City</A>  
<P>  
<A HREF="http://www.google.com //">Google. </A>  
</BODY>  
</HTML>
```

Kết quả trông như sau, các link không có gạch dưới:



Phần II JavaScript

Cần có gì để sử dụng JavaScript

Để bắt đầu học JavaScript, cần có một số điều kiện như sau:

1. Cần biết cách sử dụng HTML và có khả năng soạn thảo các tài liệu HTML.
2. Cần có một trình duyệt Web (Web Browser) hỗ trợ JavaScript. Nên dùng Netscape 3 hay mới hơn, hay Microsoft Internet Explorer 3.02 trở lên. Nếu dùng IE, nên dùng IE 4 bởi vì IE 3.x không tương thích với tất cả các script.

Phần căn bản JavaScript

Các căn bản để chèn JavaScript vào trang Web

Để bắt đầu với JavaScript, bạn sẽ muốn biết tag dùng để quy định một script là một phần của trang HTML. Các tag được sử dụng để bắt đầu và kết thúc script là `<SCRIPT>` và `</SCRIPT>`. Tag mở sẽ có định dạng giống như sau:

```
<SCRIPT language="JavaScript">
```

Thuộc tính language = "JavaScript" ở đây báo cho trình duyệt biết là phần code là JavaScript và không phải là một ngôn ngữ script khác, chẳng hạn như VBScript. Code JavaScript sẽ nằm sau tag này, và kết thúc với tag </SCRIPT>.

```
<SCRIPT language="JavaScript">  
.....JavaScript Code.....  
</SCRIPT>
```

Có thể đặt bao nhiêu tag <SCRIPT> trong phần thân của trang HTML cũng được, nó cũng như các tag thông thường khác. Hãy nhớ đóng mỗi tag trước khi tiếp tục. Nếu sẽ dùng các hàm JavaScript (sẽ học sau) cần phải đặt chúng trong các tag <HEAD> </HEAD> của tài liệu. Với cách này, các hàm JavaScript được tải trước khi trang HTML được hiển thị, và sẽ không thấy tất cả các loại lỗi trong JavaScript. Sau đây là một ví dụ:

```
<HEAD>  
<TITLE>My World</TITLE>  
  
<SCRIPT language="JavaScript">  
  
function cool()  
{  
  //JavaScript Stuff...  
}  
  
</SCRIPT>  
  
</HEAD>
```

Đến đây, còn một thứ nên xem trước khi bắt đầu viết các script. Bởi vì có một số trình duyệt cũ không hiểu tag <SCRIPT>. Thay vì thực hiện đoạn JavaScript, chúng thể hiện các script này ra màn hình. Để tránh được vấn đề này, bạn phải lừa trình duyệt để bỏ qua tag <SCRIPT>

nếu trình duyệt không hiểu. Làm điều này bằng cách dùng ghi chú trong HTML. Các trình duyệt cũ sẽ bỏ qua văn bản nằm trong các dấu ghi chú, nhưng các trình duyệt có hỗ trợ JavaScript sẽ thực hiện script nằm trong các dấu ghi chú. Sau đây là cách để làm điều này:

```
<SCRIPT language="JavaScript">
```

```
<!--
```

```
.....Javascript Statements.....
```

```
//-->
```

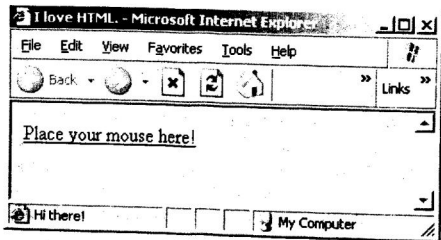
```
</SCRIPT>
```

Cách thêm JavaScript đầu tiên vào trang web

Đây là một đoạn script nhỏ sẽ viết thông báo trên thanh trạng thái của trình duyệt khi di chuyển chuột qua một liên kết (link). Hãy xem ví dụ sau:

```
<A HREF="jmouse.htm" onmouseover="window.status='Hi there!'; return true">Place your mouse here!</A>
```

Hãy thực hành và xem nó làm gì. Đặt con trỏ chuột phỉ trên liên kết, nhưng không nhấn vào nó. Và nhìn xuống thanh trạng thái tại vị trí cuối của trình duyệt. Sẽ thấy "Hi there!"



Đây là những gì xảy ra với lệnh onmouseover:

1. onmouseover=""

Đây là định dạng của lệnh onmouseover. Đặt bên trong dấu ngoặc kép là các lệnh khác hay là các hàm.

2. window.status='Hi there!'

Câu lệnh này ra lệnh cho trình duyệt viết ra thanh trạng thái. Bạn có thể đặt những gì muốn viết ra thanh trạng thái trong dấu móc đơn.

3. return true

Trả về giá trị True để trình duyệt hiển thị văn bản.

Lý do dùng dấu móc đơn trong trường hợp này là do lệnh window.status được đặt trong lệnh onmouseover, lệnh này đã sử dụng dấu ngoặc kép. Nếu dùng dấu ngoặc kép thêm một lần nữa sẽ gây nhầm lẫn cho trình duyệt.

```
onmouseover=""window.....
```

Khi bạn đã di chuyển chuột đi chỗ khác nhưng "Hi There!" vẫn còn trên thanh trạng thái. Có hai cách để giải quyết vấn đề này. Một là dùng lệnh onmouseout, hai là gọi một hàm để xóa văn bản sau một khoảng thời gian xác định. Cách thứ hai phải dùng đến hàm và các tag script, bây giờ chỉ trình bày cách đơn giản nhất, dùng lệnh onmouseout:

```
<A HREF="jmouse.htm" onmouseover="window.status='Hi there!'; return true" onmouseout="window.status=' '; return true">Place your mouse here!</A>
```

Đặt đoạn code trên trong một dòng. Nó sẽ làm biến mất văn bản khỏi thanh trạng thái khi di chuyển chuột ra khỏi liên kết.

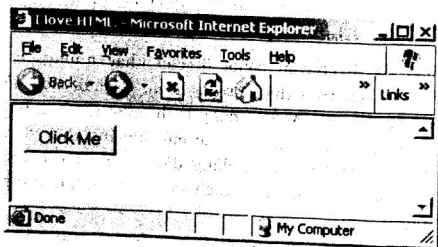
Script này rất hữu dụng, nó giúp cho người sử dụng trang web, nếu như bạn đặt vào đó dòng văn bản mô tả là liên kết này sẽ đi đến đâu.

Cách làm một nút bấm (button) phản ứng với JavaScript

Để viết các script sử dụng các nút bấm (button), trước hết phải biết cách đặt các button vào trang Web. Để làm điều này, cần sử dụng tag <FORM> bọc tag của button. Sau đây là một ví dụ HTML đặt một nút vào trang web:

```
<FORM>  
<INPUT type="button" value="Click Me" name="button1">  
</FORM>
```

Đoạn script này đặt một button vào trang web của bạn, nhưng nếu nhấn vào nó thì không có gì xảy ra.



Giải thích những gì trong đoạn code trên:

1. **<FORM>**

Tạo một form để có thể sử dụng button.

2. **<INPUT>**

Tag này cho phép tạo một vùng nhập theo một kiểu nào đó.

type="button"

Thuộc tính này khai báo vùng nhập là một button.

3. **value="Click Me"**

Đặt văn bản mà người sử dụng sẽ thấy trên button. Viết bất cứ thứ gì mà bạn muốn người vào trang web của bạn thấy.

4. **name="button1"**

Đặt tên cho button để liên hệ đến nó trong script.

Bây giờ xem một lệnh JavaScript khiến cho trình duyệt làm điều gì đó khi người xem nhấn vào button:

`onClick="javascript commands"`

Hãy đặt lệnh này bên trong tag INPUT của button, như sau:

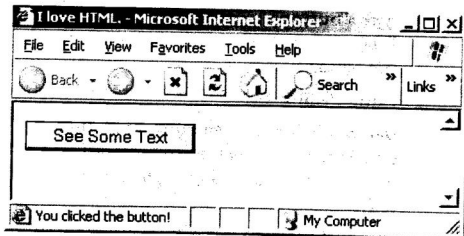
```
<INPUT type="button" value="Click Me" name="button1"
onClick=" " >
```

Cũng có thể dùng một button để hiển thị văn bản trên thanh trạng thái giống như đã làm với onMouseover.

Căn bản thiết kế Web

```
<FORM>  
<INPUT type="button" value="See Some Text"  
name="button2" onClick="window.status='You clicked the  
button!'; return true">  
</FORM>
```

Nếu nhấn vào button sẽ thấy văn bản hiển thị trên thanh trạng thái.



Bạn cũng có thể cho phép những người xem trang web thay đổi màu nền của trang web. Chỉ cần dùng lệnh sau, cũng tương tự như lệnh `window.status`:

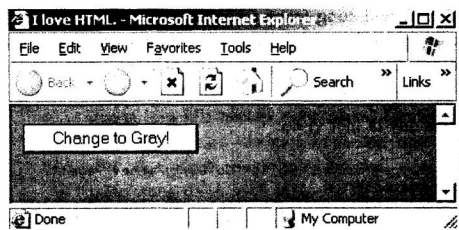
```
document.bgColor='color'
```

Chèn dòng này vào lệnh `onClick`, như sau:

```
<FORM>  
<INPUT type="button" value="Change to Gray!"  
name="button3" onClick="document.bgColor='gray'">  
</FORM>
```

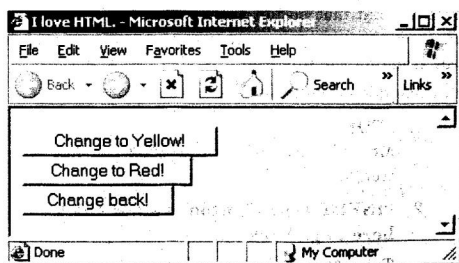
Khi nhấn vào nút này, sẽ thấy màu nền chuyển sang màu xám.

Căn bản thiết kế Web



Bạn có thêm bao nhiêu button như thế này cũng được, và nhớ là có một button chuyển về màu gốc. Đoạn script kế tiếp sẽ cho ba chọn lựa: yellow, red, và màu gốc.

```
<FORM>
<INPUT type="button" value="Change to Yellow!"
name="button3" onClick="document.bgColor='yellow'"> <br>
<INPUT type="button" value="Change to Red!"
name="button4" onClick="document.bgColor='red'"> <br>
<INPUT type="button" value="Change back!" name="button5"
onClick="document.bgColor='white'"> </FORM>
```



Script cuối cùng trong phần này cho phép sử dụng các button (nút bấm) như là các link (liên kết đến trang khác). Chỉ cần dùng lệnh sau trong lệnh onClick:

```
window.location='url'
```

Sau đây là một ví dụ:

```
<FORM>
<INPUT type="button" value="Go to my other Page!"
name="button6"
onClick="window.location='http://www.vnn.vn'">
</FORM>
```

Tạo các nút tương tác với history

Khi bạn muốn tạo một nút back – nghĩa là tạo một button đưa người sử dụng trở về trang mà đã đến trang hiện hành. Loại nút back này giống với nút back trên trình duyệt. Nếu muốn tạo một nút như vậy, chỉ cần một đoạn nhỏ script như sau:

```
<FORM>
<INPUT type="button" value="Click here to go back"
onClick="history.back()">
</FORM>
```

Script này đặt một button vào trang web của bạn dùng để đưa người sử dụng trở về trang cuối cùng trong danh sách các trang đã vào (history).

Giải thích đoạn code trên:

1. **<FORM>**

Mở một form để có thể dùng các nút trong trang web.

2. **<INPUT type="button" value="Click here to go back".....**

Tạo một nút để dùng script.

3.onClick="history.back()">

Đây là lệnh làm cho mọi thứ xảy ra. Lệnh onClick = " " báo cho trình duyệt thực hiện các lệnh trong dấu ngoặc kép khi button được nhấn. Lệnh history.back() là một hàm làm như sau: đưa người xem quay lui lại một bước trong danh sách history.

Có thể đổi hàm history.back() ở trên với một trong các lệnh sau để làm các công việc khác nhau:

1. **history.forward()**

Đưa người xem bước về phía trước một bước trong danh sách history.

2. **history.go(-1)** hay **history.go(1)**

Lệnh này cho phép bạn quyết định số trang bước tới hoặc quay lui. Dùng số âm để quay lui và số dương để đi tới.

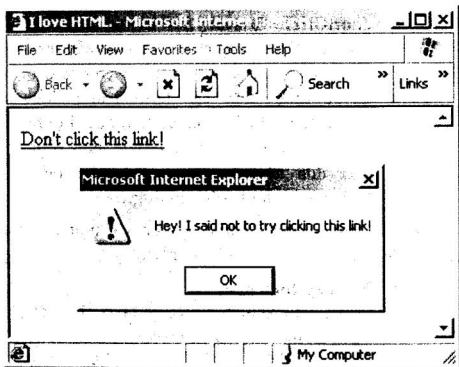
Thông báo, cảnh báo

Bắt đầu với hộp thoại cảnh báo (alert box) để thông báo cho người xem cái gì đó mà bạn muốn họ biết. Đây là lệnh cảnh báo:

```
alert('your choice of text')
```

Để sử dụng nó, cần phải đặt nó trong một lệnh khác. Hãy dùng cái ta đã quen là lệnh onMouseover. Đoạn script sau sẽ cảnh báo người sử dụng là bạn không muốn họ cố nhấn chuột vào liên kết cá biệt khi họ di chuyển chuột qua nó.

```
<A HREF="noplace" onMouseover="alert('Hey! I said not to  
try clicking this link!')"> Don't click this link!</A>
```



1. `onMouseover=""`

Báo cho trình duyệt thực hiện lệnh hay hàm bên trong dấu ngoặc kép khi người sử dụng di chuyển chuột qua đầu link.

2. `alert('Hey! I said not to try clicking this link!')`

Ra lệnh cho trình duyệt hiển thị văn bản trong dấu móc đơn trên một hộp thoại.

Có thể làm khác đi một chút, tạo một script cảnh báo ngay khi mở trang web. Đặt lệnh cảnh báo trong tag `<HEAD></HEAD>`. Bây giờ ta bắt đầu sử dụng tag `<SCRIPT> </SCRIPT>`. Sau đây là đoạn code:

```
<HEAD>  
<TITLE>Cool JavaScripts</TITLE>
```

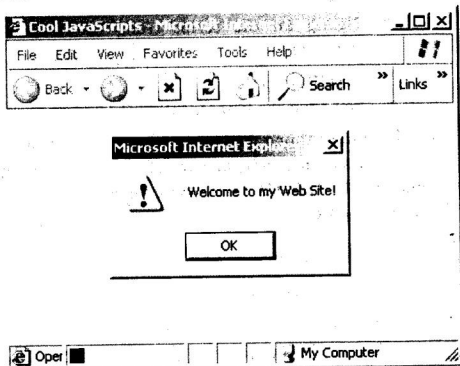
Căn bản thiết kế Web

```
<SCRIPT language="JavaScript">
<!-- hide from old browsers

alert('Welcome to my Web Site!');

//-->
</SCRIPT>

</HEAD>
```



Đoạn code này hiển thị một cảnh báo trước khi trang web được tải lên. Khi nhấn "OK" trang web sẽ tiếp tục bình thường.

1. **<SCRIPT language="JavaScript">**

Tag này cho trình duyệt biết bạn sẽ dùng các lệnh JavaScript ở đây.

2. **<!-- hide script from old browsers**

Dòng này bảo đảm là các trình duyệt cũ không hiển thị script lên như các dòng văn bản.

3. **alert('Welcome to my Web Site!');**

Đây là thông báo của bạn. Đặt thông điệp bên trong dấu móc đơn. Chú ý đến dấu chấm phẩy ở cuối dòng lệnh, nó dùng để tách biệt các lệnh JavaScript.

4. **//-->**

Ngừng ẩn các script trong các trình duyệt cũ.

5. **</SCRIPT>**

Kết thúc các lệnh JavaScript.

Thử đặt nhiều thông báo như thế này bên trong tag **SCRIPT**, theo sau mỗi lệnh là một dấu chấm phẩy. Người xem phải trả lời "OK" tất cả các thông báo trước khi trang web bắt đầu hiển thị.

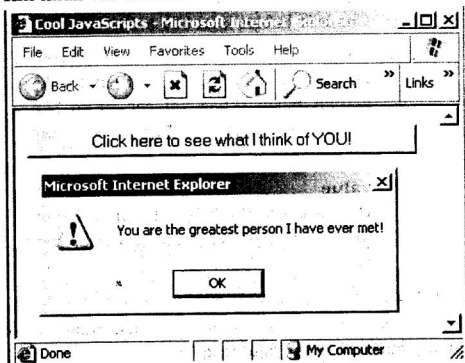
```
<HEAD>
<TITLE>JavaScript Example 2</TITLE>

<SCRIPT language="JavaScript">
<!--
alert('Please Sign My Guestbook, NOW!');
alert('I mean it, NOW!!!');
alert('Did I mention I had a guestbook? Well SIGN IT!');
alert('Oh, remember....THE GUESTBOOK! O.K.?!!');
//-->
</SCRIPT>
</HEAD>
```

Bạn cũng có thể dùng thông báo với các button. Đặt code để có button trên trang web, và không cần dùng tag **SCRIPT** trong trường hợp này.

```
<FORM>
<INPUT type="button" value="Click here to see what I
think of YOU!"
onClick="alert('You are the greatest person I have ever
met!')">
</FORM>
```

Khi nhấn vào button:



Thêm các biến và các hàm vào trong các script

Bây giờ là lúc bắt đầu với một số món thú vị. Các biến và các hàm. Hãy bắt đầu với việc khai báo biến. Chúng ta giữ tất cả các biến trong phần HEAD. Đặt các khai báo giữa các tag SCRIPT bên trong phần HEAD của tài liệu.

Để khai báo các biến trong JavaScript, bạn có thể viết tương tự như sau:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--hide from old browsers
```

```
var name=value;
```

```
//-->
```

```
</SCRIPT>
```

```
</HEAD>
```

1. **var**

Cho biết là bạn khai báo biến.

2. **name**

Đây là tên đặt cho biến. Đặt bất kỳ tên gì bạn muốn (khác với các từ có sẵn trong JavaScript như "function" hay "onmouseover")

3. **value**

Đây là giá trị khởi tạo cho biến. Nó có thể là số, cụm từ, true, false, hay null.

Sử dụng số

Có thể gán một giá trị số vào một biến bằng cách đặt số mong muốn phía sau dấu bằng (=):

```
var cars=3;
```

Cũng có thể dùng số có dấu chấm thập phân.

```
var cost=9.95;
```

Sử dụng chuỗi

Một chuỗi chỉ là một nhóm các ký tự, như là một câu hay một từ. Để định nghĩa một chuỗi, bạn cần đặt chuỗi trong dấu ngoặc kép hoặc dấu móc đơn, như sau:

```
var movie="The Lost World";
```

hay

```
var movie= 'The Lost World';
```

Nếu đặt số trong dấu ngoặc kép, chúng được xem như là một chuỗi không phải là giá trị số.

Các giá trị Boolean

Chỉ cần gán cho biến một giá trị true hoặc false.
Không cần dấu ngoặc kép. Ví dụ:

```
var story=true;
```

Giá trị null

Nếu khai báo một biến có giá trị null, có nghĩa là không có gì cả. Không phải là không (zero). Định dạng như sau:

```
var mymoney=null;
```

Chữ hoa và chữ thường

Không giống như HTML, JavaScript là ngôn ngữ script phân biệt chữ hoa và chữ thường. Cho nên một biến tên "john" khác với biến có tên "John" hay "JOHN". Quy luật này cũng áp dụng đối với các lệnh và các hàm. Nếu bạn không chú ý đến chữ hoa và chữ thường trong script có thể gây ra lỗi.

Dấu chấm phẩy

Đừng quên các dấu chấm phẩy. Chúng được dùng để tách biệt các lệnh JavaScript và các khai báo, và cũng gây ra lỗi nếu bỏ nó.

Các hàm!

Các hàm được sử dụng để làm cho mọi thứ được tổ chức và dễ đọc hơn. Một hàm là một tập các câu lệnh JavaScript được đặt chung với nhau để thực hiện một mục đích. Nên đặt các hàm bên trong các tag SCRIPT đặt trong phần HEAD. Đây là định dạng của khai báo một hàm:

```
<HEAD>  
<SCRIPT language="JavaScript">  
<!--hide from old browsers
```

Cần bản thiết kế Web

```
function name (parameter1, parameter2)
{
    JavaScript Statements and declarations
}

//-->
</SCRIPT>
</HEAD>
```

1. **function**

Cho biết là bạn chuẩn bị tạo một hàm.

2. **name**

Đây là tên đặt cho hàm. Cũng như trước, có thể đặt tên bất kỳ.

3. **(parameter1, parameter2)**

Các tham số (*parameter*) là các biến được gửi cho hàm khi nó được gọi. Có thể không có tham số hoặc có một, hai, ba tham số ...

4. **{**

Ký hiệu này cho phép bắt đầu viết các câu lệnh JavaScript và các khai báo.

5. **}**

Ký hiệu này cho biết kết thúc các câu lệnh, và kết thúc hàm.

Để sử dụng một hàm, bạn phải gọi hàm khi muốn dùng nó. Gọi một hàm bằng tên và bất kỳ tham số nào muốn gửi vào, và dấu chấm phẩy, như sau:

```
function dosomething (mymoney, cost);
```

Tới đây, ta viết một script khác dùng các biến và hàm để ghi văn bản ra thanh trạng thái. Di chuyển chuột qua đầu link, bạn sẽ thấy dòng chữ trên thanh trạng thái.

```
<HEAD>
<SCRIPT language="JavaScript">
<!--hide

var text=" ";

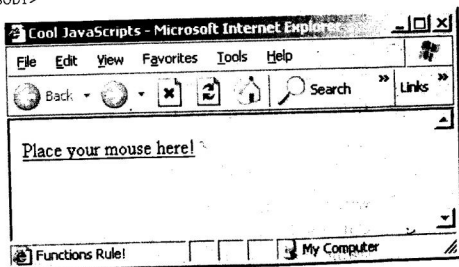
function overlink (text)
{
window.status=text;
}
function offlink (text)
{
window.status=text;
}

//-->
</SCRIPT>
</HEAD>

<BODY>

<A HREF="jvar.htm" onmouseover="overlink('Functions
Rule!');return true"
onmouseout="offlink(' ');return true"> Place your mouse
here!</A>

</BODY>
```



Trong trường hợp này script dài hơn và mất một ít thời gian để hiểu nó.

Trong phần HEAD

1. **var text="";**

Khai báo một biến có tên text, và cho nó một giá trị chuỗi rỗng.

2. **function overlink(text)**

Khai báo một hàm tên overlink. Hàm này đòi hỏi truyền vào nó biến text để nó làm việc đúng.

3. **{**

Bắt đầu các câu lệnh JavaScript của hàm overtext.

4. **window.status=text;**

Đặt giá trị của biến text vào thanh trạng thái. Giá trị của text được gửi cho hàm khi nó được gọi trong tag link, là chuỗi "Functions Rule!".

5. **}**

Kết thúc hàm overlink.

6. **function offlink (text)**

Khai báo hàm offlink. Hàm này đòi hỏi biến truyền vào nó biến text.

7. **window.status=text;**

Đặt giá trị của biến text vào thanh trạng thái. Giá trị của text được gửi

vào hàm khi nó được gọi trong tag link là "".

Trong phần BODY

```
<A HREF="jvar.htm"  
onMouseover="overlink('Functions  
Rule!');return true"  
onMouseout="offlink(' ');return true">  
Place your mouse here!</A>
```

Tag này gọi cả hai hàm và truyền vào một chuỗi được gán cho biến text. Hàm đầu tiên, overlink, được gọi bên trong lệnh onMouseover(). Có nghĩa là khi người sử dụng đặt con trỏ chuột phía trên đầu link, các câu lệnh bên trong hàm overlink sẽ được thực hiện. Chuỗi "Function Rule!" được truyền vào hàm overlink, và nó xuất hiện trên thanh trạng thái. Lệnh onMouseout gọi hàm offlink khi con trỏ chuột được di chuyển ra khỏi đầu link. Một chuỗi trống được truyền vào và nó ghi lên thanh trạng thái, xóa chuỗi "Function Rule!" trước đó. Cụm từ return true bảo đảm là các lệnh này được làm việc bằng cách trả về true.

Các toán tử logic và các câu lệnh có điều kiện

Phần này học một nhóm các toán tử và câu lệnh if/else. Đầu tiên chúng ta tìm hiểu một số toán tử toán học.

Toán tử	Ý nghĩa
+	Cộng
-	Trừ
*	Nhân
/	Chia
%	Chia lấy dư

Và đây là các toán tử so sánh:

Toán tử	Ý nghĩa
>	Lớn hơn
<	Nhỏ hơn
>=	Lớn hơn hoặc bằng
<=	Nhỏ hơn hoặc bằng
==	Bằng (khác với "=" dùng để gán)
!=	Không bằng (khác)

Các toán tử logic:

Toán tử	Ý nghĩa
&&	AND
	OR
!	NOT

Các toán tử này thường kết hợp làm việc với câu lệnh if/else:

```
if (conditional statement)
{
    JavaScript statements...
}
else
{
    JavaScript Statements.....
}
```

Câu lệnh này hữu dụng nếu bạn muốn kiểm tra giá trị của một biến hay một số biến. Nếu câu lệnh trong dấu ngoặc đơn () là đúng (true), thì các lệnh theo sau lệnh if bên trong dấu ngoặc nhọn {} sẽ được thực hiện. Nếu sai, các lệnh phía sau lệnh else sẽ được thực hiện. Ví dụ:

```
var number=3;
var mymoney=0;
if (number > 2)
{
  mymoney=100;
}
else
{
  mymoney= -100;
}
```

Đoạn code này quyết định giá trị của biến mymoney, dựa trên giá trị của biến number. Nếu number lớn hơn 2, thì giá trị của mymoney sẽ được đổi thành 100. Nếu number không lớn hơn hai, thì mymoney là -100.

Giả sử bạn muốn thực hiện một tập các câu lệnh một số lần, và kết thúc khi nào bạn có thứ bạn cần. Để làm điều này, có thể dùng một vòng lặp for hay một vòng lặp while để lặp lại. Trước tiên là vòng lặp for:

```
for ( condition1; condition2; command)
{
  JavaScript Statements....
}
```

Những gì nó làm là bắt đầu vòng lặp với condition1, và kết thúc với condition2, và thực hiện từng lệnh (command) khi nó lặp lại. Do đó, nếu bạn muốn thay đổi giá trị của biến 10 lần, có thể làm giống như sau:

```
var count=1;
var mymoney=0;
for (count=1; count<11; count=count+1)
{
  mymoney=mymoney+100;
}
```

Vòng lặp bắt đầu với count=1. Sau đó câu lệnh mymoney=mymoney+100 được thực hiện lần đầu tiên. Sau đó lệnh count=count+1 được thực hiện, làm cho count

bằng 2. Ngay khi nó hoàn tất, giá trị của count được kiểm tra để chắc chắn là nó nhỏ hơn 11 (condition2). Bởi vì count là 2, nhỏ hơn 11, vòng lặp được thực hiện thêm lần nữa, và count tăng lên 3. Quá trình này cứ tiếp tục cho đến khi count bằng với 11. Sau khi kết thúc vòng lặp, mymoney từ 0 đã tăng lên thành 1000.

Bây giờ ta xét vòng lặp while. Nó làm công việc tương tự như vòng lặp for, nhưng nó được viết khác đi:

```
var count=1;
var mymoney=0;
while (count<11)
{
mymoney=mymoney+100;
count=count+1;
}
```

Ví dụ này làm cùng một công việc giống như ví dụ trước. Nó kiểm tra count có nhỏ hơn 11 hay không. Nếu có, các câu lệnh sẽ được thực hiện. Nếu không, vòng lặp kết thúc. Nhớ cộng thêm 1 vào count trong phần chứa các dòng lệnh, nếu không bạn sẽ có một vòng lặp vô định.

JavaScript Prompts

Giả sử bạn muốn lấy tên của người sử dụng trước khi họ thấy được trang web, và sau đó ghi tên của họ trên trang web của bạn. Bạn có thể làm điều này với JavaScript prompt. Đây là dòng lệnh:

```
prompt('Your Question', '');
```

Lệnh này làm xuất hiện một cửa sổ có câu hỏi của bạn, và một khoảng trống để người sử dụng trả lời câu hỏi. Cặp dấu móc đơn thứ hai cho phép bạn nhập câu trả lời mặc nhiên. Nếu để nó trống, người sử dụng chỉ thấy một hộp

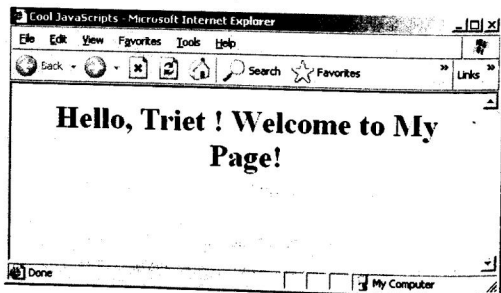
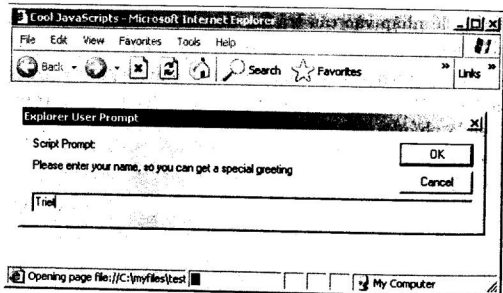
trống để nhập vào câu trả lời. Sau đây là đoạn script làm điều này.

Chú ý cách của prompt và câu lệnh if/else trong phần HEAD, trong khi nơi thực sự ghi thông tin ra trang web là phần BODY:

```
<SCRIPT language="JavaScript">
<!--hide
var yourname= prompt('Please enter your name, so you \
                        can get a special greeting', ' ');
if ( (yourname==' ') || (yourname==null) )
{
    yourname="Dude";
}
//-->
</SCRIPT>
</HEAD>
<BODY>

<SCRIPT language="JavaScript">
<!--hide
document.write("<CENTER><H1>Hello, " + yourname +
                " ! Welcome to My Page! <\/H1><\/CENTER>");
//-->
</SCRIPT>
</BODY>
```

Căn bản thiết kế Web



Cái xảy ra đầu tiên là biến `yourname` được gán cho giá trị nhận được từ người sử dụng nhập vào từ lệnh prompt. Do đó biến `yourname` sẽ là một chuỗi các ký tự chứa tên của một người. Câu lệnh `if/else` gán "Dude" vào `yourname` nếu người sử dụng không nhập vào gì cả. Nó kiểm tra `yourname` có phải là "" hay null, cả hai đều có

nghĩa là không có gì. Trong phần BODY, dùng tag SCRIPT một lần nữa để đặt các lệnh script vào giữa trang HTML. Và bạn cũng thấy có một lệnh mới đó là `document.write("");`. Đây là lệnh cho phép ghi biến `yourname` ra trang HTML. Bạn sẽ ghi ra hai chuỗi cộng với một biến `yourname`. Biến `yourname` không cần dấu ngoặc kép bởi vì nó là một giá trị và nó không phải là một chuỗi (giá trị của nó là một chuỗi). Đó là lý do có hai dấu cộng bao quanh nó. Lệnh này làm cho trình duyệt ghi ra chuỗi đầu tiên tiếp theo là giá trị của biến và chuỗi thứ hai. Chú ý các tag HTML đặt bên trong chuỗi. Bởi vì đây là JavaScript, cách duy nhất để viết thêm các tag HTML vào trang web là bằng cách đặt chúng trong dấu ngoặc kép như là một phần của chuỗi. Tương tự, chú ý đến cách các tag đóng được ghi ra khác đi. (`</H1>`). Dấu sổ ngược (`\`) ở đây đóng vai trò như là một ký tự chuyển đổi. Nó cho phép ghi dấu sổ (`/`) ra màn hình mà không nhầm lẫn nó với phép chia. (Trong JavaScript dấu `/` là phép chia.)

Các hộp thoại xác nhận

Một hộp thoại xác nhận (confirmation box) cho phép người sử dụng chọn có thực hiện hay không một thao tác đã chọn. Hộp thoại xác nhận xuất hiện giống như một hộp thoại cảnh báo (alert box), nhưng cho phép người sử dụng nhấn nút "OK" hoặc "Cancel". Dạng cơ bản của lệnh như sau:

```
confirm("Text or question you want to use");
```

Vấn đề là nếu bạn chỉ dùng như vậy sẽ không hữu dụng. Hộp thoại xác nhận sẽ trả về giá trị `true` hoặc `false`, đó là lý do ta dùng hộp thoại xác nhận. Một cách

dễ dàng để nhận giá trị trả về để dùng lại là gán giá trị trả về vào một biến, như sau:

```
var where_to= confirm("Do you really want to go to \
this page??");
```

Bây giờ, bạn có thể dùng biến `where_to` để đưa người sử dụng từ một trang đến một trang khác, phụ thuộc vào giá trị trả về của hộp thoại xác nhận. Có thể làm điều đó với lệnh `if/else`:

```
if (where_to== true)
{
    window.location="http://yourplace.com/yourpage.htm";
}
else
{
    window.location="http://www.barbie.com";
}
```

Trong trường hợp này, nếu người xem nhấn nút "OK", trình duyệt sẽ mở trang do bạn chỉ định. Nếu người xem nhấn nút "Cancel", trình duyệt sẽ mở trang web Barbie.

Để sử dụng nó, viết một hàm JavaScript trong phần HEAD và sau đó gọi hàm này trong phần BODY. Nên đặt các phần trên trong một hàm đặt trong tag HEAD, như sau:

```
<SCRIPT language="JavaScript">
```

```
<!--
```

```
function go_there()
```

```
{
```

```
    var where_to= confirm("Do you really want to go to \
this page??");
```

```
if (where_to== true)
{
    window.location="http://yourplace.com/yourpage.htm";
}
else
{
    window.location="http://www.barbie.com";
}
//-->
</SCRIPT>
```

Bây giờ bạn cần tìm một cách để gọi hàm trong phần thân của trang HTML để bạn có thể hỏi người xem có đi đến trang chỉ định. Có thể dùng một button để gọi hàm khi nó được nhấn, do đó có thể đặt đoạn code sau trong phần thân:

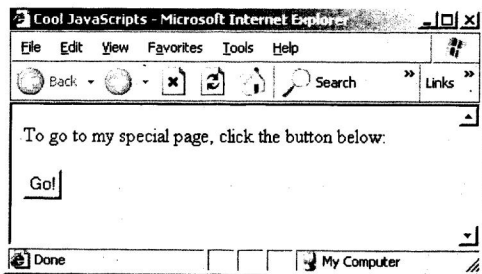
To go to my special page, click the button below:

<FORM>

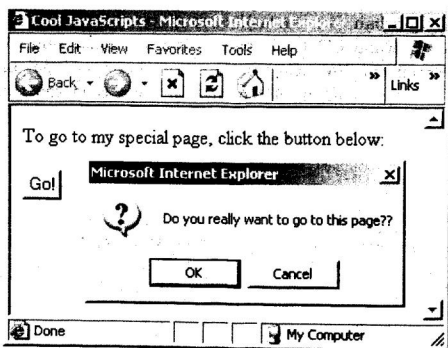
<INPUT TYPE="button" value="Go!" onClick="go_there()">

</FORM>

Căn bản thiết kế Web



Khi nhấn "Go":



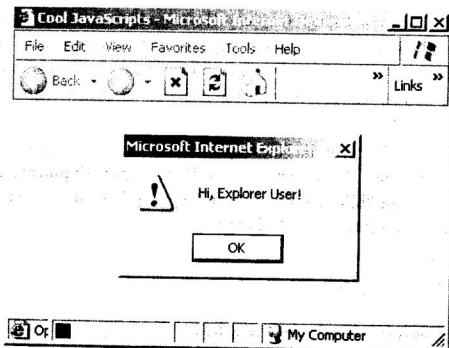
Lấy thông tin trình duyệt

Tìm loại trình duyệt (Browser detection) cho phép tìm ra trình duyệt nào đang được người xem dùng, và sau đó thực hiện script dựa trên nó.

Có hai đối tượng (object) thường được sử dụng cho chuyện này, là các đối tượng navigator.appName và navigator.appVersion. Cái đầu tiên trả về tên của trình duyệt, và đối tượng thứ hai trả về phiên bản của trình duyệt.

Nếu trình duyệt là Netscape, navigator.appName trả về "Netscape". Nếu trình duyệt là Internet Explorer, nó trả về chuỗi "Microsoft Internet Explorer". Ví dụ sau chào người sử dụng theo trình duyệt:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
var browserName=navigator.appName;
if (browserName=="Netscape")
{
    alert("Hi Netscape User!");
}else
{
    if (browserName=="Microsoft Internet Explorer")
    {
        alert("Hi, Explorer User!");
    } else
    {
        alert("What ARE you browsing with here?");
    }
}
//-->
</SCRIPT>
</HEAD>
```



Có thể làm tương tự với `navigator.appVersion`, thông thường bạn muốn nhận một số nguyên từ thông tin phiên bản (2,3,4 ...). Để làm điều này, ta dùng hàm `parseInt()`:

```
var browserVer=parseInt(navigator.appVersion);
```

Nó chỉ trả về số nguyên và không phải là số thực như 4.51. Nó chỉ trả về giá trị 4 trong trường hợp này. Script sau kiểm tra trình duyệt có đủ mới để sử dụng hay không và thông báo cho người sử dụng:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
var browserVer=parseInt(navigator.appVersion);
if (browserVer >= 4)
{
    alert("Your browser is new enough for my site.");
}
```



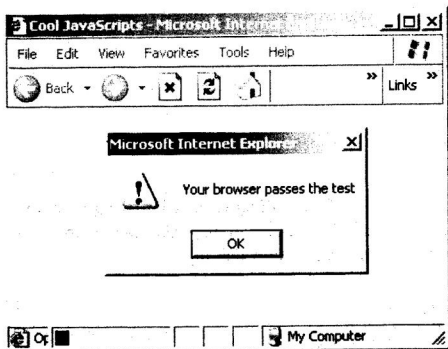
```
else
{
    alert("I think you need a new browser!");
}
//-->
</SCRIPT>
</HEAD>
```

Đĩ nhiên, bạn có thể dùng cả hai đối tượng để chính xác hơn. Bạn muốn bảo đảm với một tập các trình duyệt và chỉ với trên một phiên bản nào đó:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
var browserName=navigator.appName;
var browserVer=parseInt(navigator.appVersion);
if ((browserName=="Netscape" && browserVer>=3) ||
(browserName=="Microsoft Internet Explorer" &&
browserVer>=4))
    version="n3";
else
    version="n2";

if (version=="n3")
    alert("Your browser passes the test");
else
    alert("You need an upgrade, I think.");
//-->
</SCRIPT>
</HEAD>
```

Căn bản thiết kế Web



Như bạn có thể thấy, đoạn code này sử dụng nhiều toán tử logic đã nêu trong phần trước. Giải nghĩa như sau: “nếu trình duyệt là Netscape và có phiên bản 3 hoặc lớn hơn, HOẶC (OR) nếu trình duyệt là Microsoft Internet Explorer và có phiên bản 4 hoặc lớn hơn, sau đó gán biến phiên bản giá trị ‘n3’. Nếu khác, biến phiên bản sẽ là ‘n2’. Sau đó tiếp tục với phần còn lại”

Một trong những ứng dụng của việc phát hiện trình duyệt là để chuyển hướng người sử dụng đến trang được làm phù hợp với loại và phiên bản của trình duyệt.

Chuyển hướng - Redirection

Chuyển hướng thường được dùng để đưa người sử dụng đến một trang web khác tùy thuộc vào tên và phiên bản của trình duyệt. Để chuyển hướng người xem ngay lập tức, chỉ cần thêm một lệnh ngắn trong phần HEAD:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
window.location="http://someplace.com";
//-->
</SCRIPT>
</HEAD>
```

Đôi khi chuyển hướng còn được dùng trong trường hợp web site đã di chuyển đến vị trí khác. Cách dùng khác là lấy thông tin của trình duyệt và chuyển hướng đến trang dành cho Netscape, hay trang dành cho Internet Explorer:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
var browserName=navigator.appName;
if (browserName=="Netscape")
{
window.location="http://www.someplace.com/ns.html";
}
else
{
if (browserName=="Microsoft Internet Explorer")
{
window.location="http://www.someplace.com/ie.html";
}
else
{
window.location="http://www.someplace.com/other.html";
}
```

```
    }  
  }  
  //-->  
</SCRIPT>  
</HEAD>
```

Đoạn code này dùng phương pháp lấy thông tin trình duyệt trong phần trước. Thay vì hiển thị thông báo với hộp thoại alert, ta chuyển hướng đến trang web thích hợp với trình duyệt đang sử dụng.

Nếu bạn có một phiên bản dành cho trình duyệt IE 4 và các phiên bản khác cho các trình duyệt khác, chúng ta có thể tạo một script khác từ phần trước:

```
<HEAD>  
<SCRIPT language="JavaScript">  
<!--  
var browserVer=parseInt(navigator.appVersion);  
if (browserVer >= 4)  
{  
    window.location="http://www.someplace.com/v4.html";  
}  
else  
{  
    window.location="http://www.someplace.com/other.html";  
}  
//-->  
</SCRIPT>  
</HEAD>
```

Có thể dùng `top.frames.length` để kiểm tra xem trang web có trong frame hay không. Nếu có giá trị là 0,

trang web không có trong frame. Nếu nó lớn hơn không, trang web được đặt trong frame. Do đó, bạn có thể đưa người xem trở về trang frameset chính của bạn bằng cách dùng script giống như sau:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
function getgoing()
{
    top.location="http://someplace.com";
}

if (top.frames.length==0)
{
    alert("You will be redirected to our main page in \
10 seconds!");
    setTimeout('getgoing()',10000);
}
//-->
</SCRIPT>
</HEAD>
```

Đoạn code này thông báo cho người sử dụng và đưa họ về trang chính sau 10 giây. Bạn có thể thay đổi con số trong hàm `setTimeout` để áp đặt thời gian bạn muốn.

Bạn cũng có thể dùng nó đưa người xem ra khỏi một frame của trang khác, do họ đi theo các liên kết của trang khác có frame.

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
function getgoing()
{
    top.location="http://someplace.com";
}

if (top.frames.length > 0)
{
    alert("The frames will be broken in ten seconds.");
    setTimeout('getgoing()',10000);
}
//-->
</SCRIPT>
</HEAD>
```

Cách dùng JavaScript để mở một cửa sổ mới

Để mở một cửa sổ mới, bạn cần dùng một hàm xây dựng sẵn của JavaScript. Nó như sau:

```
window.open('url to open','window name','attribute1,attribute2')
```

Đây là hàm cho phép mở một cửa sổ trình duyệt mới cho người xem dùng. Chú ý là tất cả tên và thuộc tính được tách biệt nhau bằng dấu phẩy. Các tham số bên trong gồm:

1. 'url to open'

Đây là địa chỉ web của một trang mà bạn muốn xuất hiện trong cửa sổ mới.

2. **'window name'**

Bạn có thể đặt tên cho cửa sổ tùy ý, để trong trường hợp bạn cần liên hệ đến cửa sổ trong script.

3. **'attribute1,attribute2'**

Các thuộc tính của cửa sổ mới.

Các thuộc tính của cửa sổ

Dưới đây là danh sách các thuộc tính mà bạn có thể dùng:

1. **width=300**

Chiều rộng của cửa sổ.

2. **height=200**

Chiều cao của cửa sổ.

3. **resizable=yes hoặc no**

Cho phép người sử dụng thay đổi kích thước của cửa sổ hay không.

4. **scrollbars=yes hoặc no**

Cho phép bạn quyết định cửa sổ có thanh cuộn hay không.

5. **toolbar=yes hoặc no**

Cửa sổ có thanh công cụ hay không.
(thanh chứa các nút back, forward, stop...)

6. **location=yes hoặc no**

Có hiển thị hộp nhập địa chỉ hay không. (Nơi để gõ vào http://...)

7. **directories=yes hoặc no**

Có hiển thị các nút bấm mở rộng hay không.

8. **status=yes hoặc no**

Có hiển thị thanh trạng thái ở cuối cửa sổ hay không.

9. **menubar=yes hoặc no**

Có hiển thị thanh menu hay không (File, Edit...)

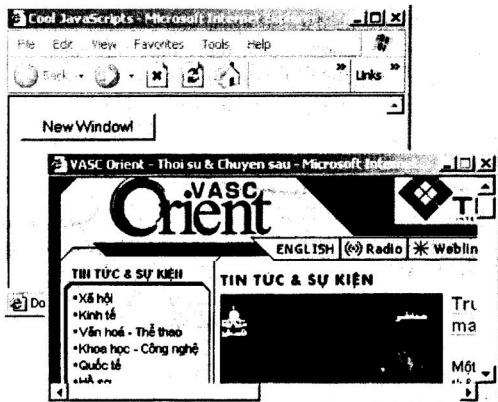
10. **copyhistory=yes hoặc no**

Có sao chép danh sách history từ trình duyệt cũ qua cửa sổ mới hay không.

Sau đây là một ví dụ về mở cửa sổ mới:

```
<FORM>  
<INPUT type="button" value="New Window!"  
      onClick="window.open('http://www.vnn.vn/',  
        'mywindow', 'width=400,height=200') ">  
</FORM>
```

Sau khi nhấn nút sẽ xuất hiện một cửa sổ 400x200.



Một số quy luật quan trọng

Trước khi tiếp tục, chúng ta chú ý một số quy luật sau:

1. Khi dùng tag INPUT, giữ mọi thứ nằm trên cùng một dòng, kể cả các lệnh JavaScript.
2. Khi viết lệnh onClick=" ", không để bất kỳ khoảng trắng nào nằm giữa mọi thứ. Chỉ dùng dấu phẩy và dấu ngoặc kép. Bất kỳ khoảng trắng nào cũng có thể làm cho nó không chạy được trên Netscape.

3. Không đặt dấu ngoặc kép bao yes, no, hay các số trong thuộc tính. Chỉ dùng dấu móc đơn cho toàn bộ các thuộc tính.
4. Trong một số trình duyệt, có thể cần phải dùng 1 cho yes, và 0 cho no trong phần thuộc tính

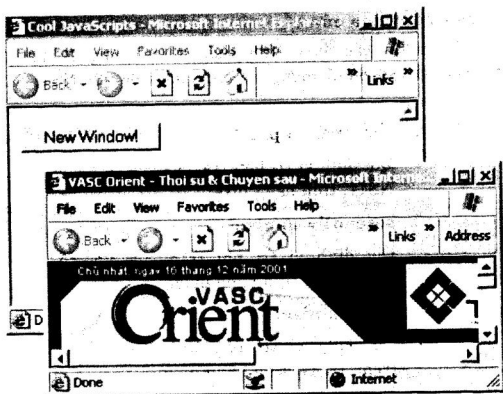
Một cửa sổ trình duyệt mới

Hãy xem đoạn code tạo một cửa sổ trình duyệt hoàn toàn mới. Chỉ cần dùng yes cho tất cả các thuộc tính. Đây là đoạn code:

```
<FORM>
<INPUT type="button" value="New Window!"
onClick="window.open('http://www.vnn.vn','mywindow',
    'width=400,height=200,toolbar=yes, location=yes,
    directories=yes,status=yes,menubar=yes,
    scrollbars=yes,copyhistory=yes, resizable=yes')">
</FORM>
```

Hãy nhớ là đặt mọi thứ trên cùng một dòng, một dòng dài.

Thử nhấn vào nút này, sẽ có một cửa sổ với đầy đủ tính năng.



Đóng cửa sổ mới

Chúng ta tạo một button "Close Window" trong cửa sổ mới để đóng cửa sổ. Để được như vậy, ta dùng hàm `window.close()` trong trang HTML của cửa sổ mới. Chỉ cần đặt đoạn code này vào trang HTML của cửa sổ mới (phải là các trang của bạn):

```
<FORM>
<INPUT type="button" value="Close Window"
onClick="window.close()" >
</FORM>
```

Dĩ nhiên, cửa sổ có thể đóng bằng dấu x phía trên bên phải của cửa sổ.

Đặt vị trí của cửa sổ

Có một tập các thuộc tính khác có thể dùng để đặt vị trí của cửa sổ mới, nhưng nó chỉ làm việc với NS4+ và IE4+

1. screenX=số pixel

Qui định vị trí bên trái của cửa sổ trong NS4+, tính bằng pixel.

2. screenY= số pixel

Qui định vị trí phía trên của cửa sổ, trong NS4, tính bằng pixel.

3. left= số pixel

Qui định vị trí bên trái của cửa sổ, trong IE 4+, tính bằng pixel.

4. top= số pixel

Qui định vị trí phía trên của cửa sổ, trong IE 4+, tính bằng pixel.

Nhưng bạn làm thế nào nếu không biết là người xem dùng trình duyệt nào? Trong trường hợp này, bạn có thể dùng cả hai tập thuộc tính – trình duyệt sẽ bỏ qua thuộc tính nào mà nó không nhận biết được. Sau đây là một ví dụ:

```
<FORM>
<INPUT type="button" value="New Window!"
onClick="window.open('jex5.htm','mywindow','width=400,
height=200,left=0,top=100,screenX=0,screenY=100')">
</FORM>
```

Làm thế nào để gọi hàm mà không dùng các nút bấm

Thay vì dùng các nút bấm để gọi các hàm, có thể dùng các đầu link. Có hai cách để làm điều này. Các thứ nhất dễ hiểu hơn.

Phương pháp thứ nhất là xử lý một hàm JavaScript trong thuộc tính HREF của tag link. Nếu bạn muốn liên kết (link) đến trang khác, chỉ cần viết:

```
<A HREF="nextpage.htm">Click here</A>
```

Bạn có thể gọi một hàm JavaScript đã viết sẵn thay vì dùng cách trên:

```
<A HREF="javascript:myfunction()">Click Here</A>
```

Bây giờ bạn có thể mở một cửa sổ mới mà không cần dùng nút. Sau đây là một đoạn script mở một cửa sổ mới:

Trước hết, tạo một hàm trong phần HEAD, như sau:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--hide

function newwindow()
{
window.open('jex5.htm','jav','width=300,height=200,
resizable=yes');
}
//-->
</SCRIPT>
```

Script trên sẽ mở trang "jex5.htm" trong một cửa sổ mới. Có thể thay thế trang này thành trang của bạn muốn mở và thêm các thuộc tính mà bạn muốn.

Đến phần BODY nơi bạn muốn xuất hiện đầu link, và viết tag link như sau:

Căn bản thiết kế Web

```
<A HREF="javascript:newwindow()" >Click Here!</A>
```

Nếu bạn muốn dùng dạng này với hình ảnh để liên kết, chỉ cần đặt tag IMG bên trong tag A như sau:

```
<A HREF="javascript:newwindow()" ><IMG SRC="scare.jpg"
border="0"></A>
```

Cách thứ hai để làm điều này khó hơn một chút. Sử dụng `onClick=""` trong tag link. Như ví dụ sau:

```
<A HREF="newpage.htm" onClick="newwindow();return
false">Click Here!</A>
```

Chú ý là có dấu chấm phẩy sau gọi hàm, và "return false" tại cuối dòng lệnh. "return false" giữ cho trình duyệt không đi đến "newpage.htm" sau khi mở cửa sổ mới. Bạn có thể đặt bất kỳ địa chỉ nào trong HREF, nhưng liên kết này không đưa bạn đến đó (ngoại trừ một số trình duyệt cũ). Bạn cũng có thể đặt một hình ảnh bên trong tag link để tạo một hình ảnh có thể nhấn vào được.

Phần JavaScript Nâng Cao

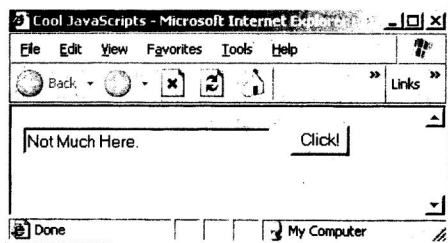
Hàm setTimeout

Bạn muốn làm một số công việc sau một khoảng thời gian xác định. Hàm `setTimeout` có thể giúp bạn tạo một số script dùng thời gian chờ để làm cho việc gì đó xảy ra. Xem cách gọi hàm `setTimeout`:

```
setTimeout("yourfunction()",1000);
```

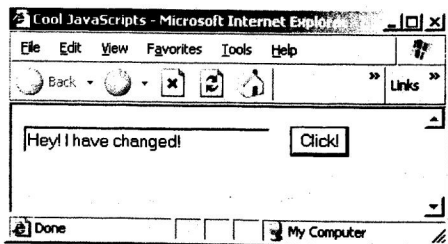
Tham số đầu tiên là một chuỗi, chứa tên của hàm muốn sử dụng. Trong trường hợp này, hàm này đã được đặt tên là "myfunction()". Tham số thứ hai là một giá trị số. Số này cho biết số mili giây mà trình duyệt phải đợi trước

Căn bản thiết kế Web



Dùng phần HEAD để định nghĩa hai hàm. Hàm đầu tiên là "newtext()". Chúng ta gọi hàm này khi người sử dụng nhấn vào button. Như bạn thấy, ta đã thay đổi giá trị trong hộp văn bản bằng cách dùng tên của form và tên của textbox:

```
document.myform.mytext.value="Hey! I have changed!";
```



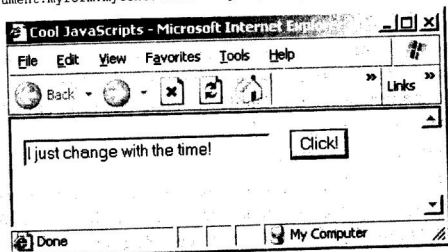
Lệnh tiếp theo là những gì làm cho sự thay đổi lần thứ hai xảy ra. Gọi hàm "moretext()" sau 1000 mili giây:

```
setTimeout("moretext()",1000);
```


Căn bản thiết kế Web

Đến đây, hàm "moretext()" thay đổi giá trị trong textbox một lần nữa, sau khi đợi một giây.

```
document.myform.mytext.value="I just change with the time!";
```



Trong phần BODY, viết code cho form và textbox rất đơn giản. Nhớ đặt tên cho form và cho textbox. Đó là cách để có thể thay đổi văn bản trong textbox bằng cách dùng các hàm. Tên cho form ở trên là "myform", và tên cho textbox là "mytext".

Sử dụng mảng trong JavaScript

Mảng là một cách để làm cho chương trình có tổ chức hơn. Chúng cho phép giải quyết các vấn đề khó khăn khi không có mảng. Mảng là một nhóm các biến có cùng kiểu dữ liệu và cùng tên, dùng chỉ số để phân biệt chúng với nhau. Giả sử bạn muốn viết ra 5 lời trích dẫn, và dùng mỗi biến cho mỗi trích dẫn. Bạn phải khai báo 5 biến:

```
quote1="I like JavaScript.";
quote2="I used to like Java.";
quote3="JavaScript rules.";
quote4="Help! JavaScript Error!";
quote5="Just Kidding.";
```

Tuy nhiên, có một cách tốt hơn cách trên là sử dụng mảng. Để định nghĩa một mảng, chúng ta cần làm theo dạng sau:

```
var Name_of_Array= new Array(number_of_elements);
```

Name_of_Array là tên bạn muốn đặt cho mảng, number_of_elements là số phần tử (số biến) mà bạn muốn mảng này lưu trữ. Do đó, để dùng cho 5 lời chỉ dẫn, có thể viết như sau:

```
var quote= new Array(5);
```

Để gán các giá trị, chúng ta sử dụng một thứ gọi là chỉ số, và đặt nó trong dấu móc vuông. Đặt nó bên phải tên của mảng, ta có phần tử đầu tiên của mảng quote là:

```
quote[0]
```

Phần tử đầu tiên của mảng luôn có chỉ số là 0, không phải 1, và phần tử cuối cùng luôn nhỏ hơn một đơn vị so với số phần tử đã khai báo, phần tử đầu tiên là quote[0], phần tử cuối cùng là quote[4]. Để gán các chuỗi vào các phần tử của mảng, chúng ta sử dụng một cách gán mới, như sau:

```
var quote= new Array(5)
quote[0]="I like JavaScript.";
quote[1]="I used to like Java.";
quote[2]="JavaScript rules.";
quote[3]="Help! JavaScript Error!";
quote[4]="Just Kidding.";
```

Chú ý là khi làm như vậy, chúng ta bỏ dấu chấm phẩy tại vị trí cuối dòng khai báo mảng, và dùng các dấu chấm phẩy ở cuối các dòng lệnh gán.

Mảng này có thể được dùng kết hợp với một vòng lặp, sử dụng một biến làm chỉ số của mảng:

```
var x=0;
for (x=0; x<5; x++)
{
    alert(quote[x]);
}
```

Đoạn code này dùng biến `x` trong vị trí của chỉ số, và chạy thông báo tất cả 5 phần tử của mảng. Lần đi qua đầu tiên, sẽ nhận được thông báo với `quote[0]`, lần thứ hai là `quote[1]`, và tiếp tục.

Toán tử `++` chưa được giải thích cho đến bây giờ. `x++` nghĩa là cộng thêm 1 vào giá trị của `x`. Trong cách tương tự, chúng ta có thể trừ 1 từ `x` với `--`. Đây là cách có thể để `x` đi từ 0 đến 4. Nó dừng tại `x=4` bởi vì phần điều kiện của vòng lặp là chỉ chạy trong khi `x` nhỏ hơn 5.

Bạn có thể viết script này trong phần HEAD, đặt nó trong một hàm, và gọi nó từ một đầu link:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
function display_quote()
{
    var quote= new Array(5)
    quote[0]="I like JavaScript.";
    quote[1]="I used to like Java.";
    quote[2]="JavaScript rules.";
    quote[3]="Help! JavaScript Error!";
    quote[4]="Just Kidding.";
    var x=0;
    for (x=0; x<5; x++)
    {
        alert(quote[x]);
    }
}
//-->
</SCRIPT>
</HEAD>
```

```
<BODY>
<A HREF="javascript:display_quote()">Click Here, I dare
you!</A>
</BODY>
```

Bạn cũng có thể gán các giá trị vào mảng bằng cách dùng vòng lặp. Cách này cho phép người xem trang web gán giá trị vào thông qua prompt, và sau đó bạn có thể thông báo cho họ biết các câu họ đã nhập vào:

```
var quote= new Array(5);
var y=0;
for (y=0; y<5; y++)
{
    quote[y]=prompt('Enter a Quote!', ' ');
}
```

Chú ý là ta chỉ khai báo mảng, không khởi tạo giá trị cho chúng, dùng quen dấu chấm phẩy. Sau đó nhận giá trị từ người sử dụng thông qua prompt. quote[y] dùng giá trị của y mỗi khi lặp lại. Do đó, bạn sẽ gán giá trị vào quote[0] trước tiên, tiếp theo là quote[1], và tiếp tục. script đầy đủ như sau:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
function display_quote2()
{
    var quote= new Array(5);
    var y=0;
    for (y=0; y<5; y++)
    {
        quote[y]=prompt('Enter a Quote!', ' ');
    }
    var x=0;
    for (x=0; x<5; x++)
    {
        alert(quote[x]);
    }
}
//-->
</SCRIPT>
</HEAD>
```

```
<BODY>
```

```
<A HREF="javascript:display_quote2()">Click Here, I dare  
you!</A>
```

```
</BODY>
```

Sử dụng mảng liên hợp trong JavaScript

Mảng liên hợp (Associative array) cho bạn một cách khác để lưu trữ thông tin. Sử dụng các mảng liên hợp, bạn có thể gọi đến các phần tử của mảng bằng một chuỗi, nó thường dễ nhớ hơn là số. Nhưng nó không hữu dụng với các vòng lặp bởi vì không dùng số làm chỉ số.

Một mảng liên hợp được khai báo giống như mảng thông thường, nhưng chúng ta thay thế chỉ số bằng các chuỗi:

```
var my_cars= new Array()  
my_cars["cool"]="Mustang";  
my_cars["family"]="Station Wagon";  
my_cars["big"]="SUV";
```

Bây giờ chúng ta có thể lấy được từ "Mustang" bằng cách nhớ nó là "cool", do đó ta dùng my_cars["cool"]. Đoạn code sau hỏi người sử dụng nhập vào loại xe họ thích, sau đó in ra tên loại xe nên mua:

```
<SCRIPT language="JavaScript">
```

```
<!--
```

```
function which_car()
```

```
{  
    var my_cars= new Array()
```

```
    my_cars["cool"]="Mustang";
```

```
    my_cars["family"]="Station Wagon";
```

```
    my_cars["big"]="SUV";
```

```
var car_type=prompt("What type of car do you like?","");

if ((car_type=="cool") || (car_type=="family") ||
(car_type=="big"))
    alert("I think you should get a(n) " +
        my_cars[car_type]+".");
else
    alert("I don't really know what you should get.Sorry.");
}

//-->
</SCRIPT>
<FORM>
<INPUT TYPE="button" onClick="which_car()" value="Go!">
</FORM>
```

Tải trước các hình ảnh

Tải trước các hình ảnh là kỹ thuật thường được sử dụng trong các hiệu ứng hình ảnh, đòi hỏi các hình ảnh phải thay đổi nhanh chóng, như hiệu ứng thay đổi hình ảnh khi di chuyển chuột qua (*rollover button*).

Những gì JavaScript làm là cho phép bạn bắt đầu tải các hình ảnh trong phần HEAD của trang web, nghĩa là với kỹ thuật này người sử dụng có các hình ảnh cần thiết trong bộ đệm của trình duyệt trước khi script bắt đầu chạy. Với cách này, một hình ảnh *rollover* sẽ không làm cho người xem phải đợi trình duyệt tải hình ảnh thứ hai vì nó đã có sẵn trong bộ đệm của trình duyệt.

Để tải trước một hình ảnh, bạn cần có một phần script nhỏ trong phần HEAD như sau:

```
<SCRIPT language="JavaScript">
<!--
pic1= new Image(100,25);
pic1.src="http://someplace.com/imagel.gif";
//-->
</SCRIPT>
```

Dòng lệnh đầu định nghĩa một đối tượng hình ảnh mới, cho nó chiều rộng là 100 và chiều cao là 25. Bạn sẽ thay thế các số này với chiều rộng và chiều cao của hình ảnh bạn định tải trước. Dòng lệnh thứ hai định nghĩa địa chỉ của hình ảnh. Thay thế nó bằng địa chỉ hình ảnh của bạn.

Tuy nhiên, có một số trình duyệt cũ không hỗ trợ đối tượng hình ảnh (*image object*). Sau đây là đoạn code kiểm tra có tồn tại đối tượng image hay không với "document.images". Chỉ cần kiểm tra nó với lệnh *if*, và sau đó chạy phần code tải trước ảnh:

```
<SCRIPT language="JavaScript">
<!--
if (document.images)
{
    pic1= new Image(100,25);
    pic1.src="http://someplace.com/imagel.gif";
}
//-->
</SCRIPT>
```

Nếu bạn muốn tải nhiều hơn một ảnh, chỉ cần gán cho chúng nhiều tên khác nhau, đặt chiều rộng, chiều cao, và địa chỉ của ảnh. Đoạn code như sau:

Căn bản thiết kế Web

```
<SCRIPT language="JavaScript">
<!--
if (document.images)
{
    pic1= new Image(100,25);
    pic1.src="http://someplace.com/image1.gif";

    pic2= new Image(240,55);
    pic2.src="http://someplace.com/image2.gif";

    pic3= new Image(88,31);
    pic3.src="http://someplace.com/image3.gif";
}
//-->
</SCRIPT>
```

Thay đổi hình ảnh khi di chuyển chuột qua nó

Sử dụng JavaScript để thay đổi một hình ảnh khi người sử dụng di chuyển chuột qua nó đã trở nên khá phổ biến. Nhưng gì xảy ra là chúng ta dùng hai hình ảnh, một đầu link, và một JavaScript để tạo một hình ảnh có thể nhấn vào được và được làm nổi bật khi con trỏ chuột đi qua nó.

Để bắt đầu, bạn cần tạo cho mình hai hình ảnh có cùng kích thước, làm bất kỳ thay đổi nào cho hình thứ hai. Bên dưới là một ví dụ, một hình với chữ màu xám, một hình với chữ màu đen. Hình màu đen là hình bạn muốn người sử dụng thấy khi họ di chuyển con trỏ chuột qua hình màu xám.

Shoe Page	Shoe Page
------------------	------------------

Bây giờ ta sẽ nhờ một script làm công việc này. Chúng ta đưa toàn bộ script và giải thích từng phần bên dưới. Sau đây là đoạn code của phần HEAD:

```
<HEAD>
```

```
<SCRIPT language="JavaScript">
```

```
<!--
```

```
    if (document.images)
```

```
    {
```

```
        piclon= new Image(100,25);
```

```
        piclon.src="shoes2.gif";
```

```
        picloff= new Image(100,25);
```

```
        picloff.src="shoes1.gif";
```

```
    }
```

```
function lightup(imgName)
```

```
{
```

```
    if (document.images)
```

```
    {
```

```
        imgOn=eval(imgName + "on.src");
```

```
        document[imgName].src= imgOn;
```

```
    }
```

```
}
```

```
function turnoff(imgName)
```

```
{
```

```
    if (document.images)
```

```
    {
```

Căn bản thiết kế Web

```
imgOff=eval(imgName + "off.src");  
document[imgName].src= imgOff;  
}  
}  
//-->  
</SCRIPT>  
</HEAD>
```

Phần I: Kiểm tra trình duyệt

Phần đầu tiên là kiểm tra trình duyệt có hỗ trợ tất cả các hàm và các khai báo chúng ta sẽ dùng.

```
if (document.images)  
{  
  
}
```

Kiểm tra này đảm bảo là trình duyệt có hỗ trợ đối tượng *image* trước khi tải các hình ảnh. Cách này tránh được các lỗi JavaScript trong các trình duyệt cũ.

Phần II: Định nghĩa và tải các hình ảnh

Tải các hình ảnh theo khung giải thích sau:

```
if (document.images)
```

```
{
```

Kiểm tra này đảm bảo trình duyệt có hỗ trợ đối tượng *image*:

```
piclon= new Image(100,25);
```

Dòng này định nghĩa một ảnh mới tên "piclon" với chiều rộng 100 và chiều cao 25.

```
pic1on.src="shoes2.gif";
```

Dòng này định nghĩa địa chỉ của hình ảnh. Nó cho phép tải ngay tức thời cho nên người sử dụng không phải đợi tải hình ảnh mới khi họ di chuyển chuột qua hình ảnh lúc sau. Bảo đảm đây là địa chỉ của hình thứ hai. Đây là hình mà bạn muốn người sử dụng thấy khi họ di chuyển chuột qua ảnh.

```
pic1off= new Image(100,25);
```

Dòng này định nghĩa ảnh mặc nhiên, ảnh thứ nhất.

```
pic1off.src="shoes1.gif";
```

Cho địa chỉ của ảnh thứ nhất để nó được tải.

} : Kết thúc định nghĩa các hình ảnh và tải ảnh.

Hàm 1: hàm mouseover

Xét hàm sẽ được thực hiện khi di chuyển chuột qua hình ảnh:

```
function lightup(imgName)
```

```
{
```

Dòng này định nghĩa hàm "lightup". Nó có một tham số nhận vào là "imgName". "imgName" sẽ là 'pic1'. Tham số này được truyền vào hàm khi gọi trong phần sau của trang HTML. Dấu ngoặc nhọn "{" bắt đầu nhóm các dòng lệnh của hàm.

```
if (document.images)
```

```
{
```

Kiểm tra trình duyệt có hỗ trợ đối tượng image và bắt đầu các dòng lệnh.

```
imgOn=eval(imgName + "on.src");
```

Dòng này định nghĩa biến "imgOn" như là tổng của biến "imgName" và chuỗi "on.src". Nhớ là hàm đọc giá trị "imgName" vào là "pic1", do đó tổng sẽ là "pic1on.src", chuỗi này đã được định nghĩa trước như là địa chỉ của hình ảnh thứ hai.

```
document[imgName].src= imgOn;
```

Bây giờ chúng ta định nghĩa imgOn cho trang HTML, để nó tải đúng hình. Bằng cách thay thế giá trị của imgOn, và imgName, ta có dòng sau:

```
document[pic1].src=pic1on.src.
```

Đây là lý do tại sao ta dùng thuộc tính name='pic1' trong tag IMG trong trang HTML. Nó cho phép liên hệ đến hình ảnh bằng tên "pic1". Địa chỉ của ảnh pic1 bây giờ là giá trị của "pic1on.src", chúng ta đã định nghĩa trước là "shoes2.gif".

|) : Kết thúc lệnh if, và kết thúc hàm.

Hàm 2: Khi con trỏ chuột rời khỏi hình ảnh

```
function turnoff(imgName)
```

```
{
```

```
    if (document.images)
```

```
    {
```

```
        imgOff=eval(imgName + "off.src");
```

```
        document[imgName].src= imgOff;
```

```
    }
```

```
}
```

```
//---->
```

```
</SCRIPT>
```

```
</HEAD>
```

Căn bản thiết kế Web

Hàm này làm cùng công việc như hàm trên, ngoại trừ là nó thực hiện khi con trỏ chuột di chuyển ra khỏi hình ảnh. Cuối cùng là các tag đóng của SCRIPT và của HEAD.

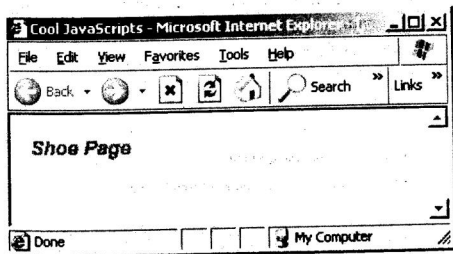
Phần cuối cùng: HTML!

Tiếp tục là phần BODY. Bạn sẽ đặt các code của hình ảnh liên kết vào trang HTML. Nên đặt chúng trên một dòng:

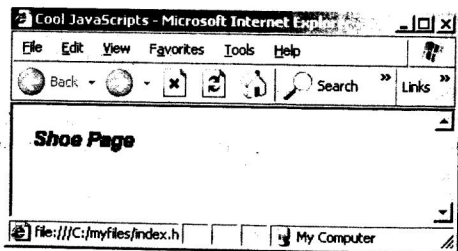
```
<A HREF="index.html" onMouseover="lightup('pic1')"  
onMouseout="turnoff('pic1')">  
<IMG SRC="shoes1.gif" name="pic1" width="100"  
height="25" border="0"></A>
```

Thuộc tính border chọn bằng 0 để dấu đường viền bên ngoài ảnh.

Ta có kết quả như sau:



Khi di chuyển chuột qua hình ảnh :



Làm hiệu ứng với nhiều hình ảnh

Bạn có thể dùng cho bao nhiêu nút nếu bạn muốn. Chỉ cần nhớ định nghĩa các ảnh khác như ảnh đầu tiên. Nhớ là bạn cần hai hình ảnh cho tất cả các ảnh bạn muốn thay đổi, ví dụ như sau:

```
if (document.images)
{
    piclon= new Image(100,25);
    piclon.src="../images/shoes2.gif";
    pic2on= new Image(100,25);
    pic2on.src="../images/story2.gif";

    picloff= new Image(100,25);
    picloff.src="../images/shoes1.gif";
    pic2off= new Image(100,25);
    pic2off.src="../images/story1.gif";
}
```

Bạn nhớ đặt tên đúng cho các hình ảnh trong phần BODY của trang HTML. Với một ảnh mới ta đặt tên là "pic2". Và cho nó link đến trang khác:

```
<A HREF="newpage.html" onmouseover="lightup('pic2')"  
onmouseout="turnoff('pic2')">  
<IMG SRC="../../images/story1.gif" name="pic2" width="100"  
height="25" border="0"></A>
```

Bây giờ bạn có hai nút thay đổi được trong trang HTML.

Tạo hiệu ứng cho một ảnh khác

Bạn muốn thay đổi một ảnh khác khi đặt con trỏ chuột trên ảnh thứ nhất, hay khi đặt con trỏ chuột trên một đầu link.

Chúng ta dùng lại script trên, và thay đổi một chút ít.

```
<SCRIPT language="JavaScript">  
<!--  
  
if (document.images)  
{  
    pic2on= new Image(100,25);  
    pic2on.src="story2.gif";  
  
    pic2off= new Image(100,25);  
    pic2off.src="story1.gif";  
}
```

Căn bản thiết kế Web

```
function lightup(imgName)
{
    if (document.images)
    {
        imgOn=eval(imgName + "on.src");
        document[imgName].src= imgOn;
    }
}

function turnoff(imgName)
{
    if (document.images)
    {
        imgOff=eval(imgName + "off.src");
        document[imgName].src= imgOff;
    }
}

//-->
</SCRIPT>
</HEAD>

<BODY>

<A HREF="newpage.html" onmouseover="lightup('pic2')"
onmouseout="turnoff('pic2')">
<IMG SRC="shoes1.gif" name="pic1" width="100"
height="25" border="0">
</A>

<P>

<IMG SRC="story1.gif" name="pic2" width="100"
height="25" border="0">
</BODY>
```


Căn bản thiết kế Web

Nếu muốn dùng một link thông thường để thay đổi hình ảnh, chỉ cần bỏ tag IMG thứ nhất và thay vào đó là văn bản bạn muốn, như sau:

```
<A HREF="newpage.html" onmouseover="lightup('pic2')"  
onmouseout="turnoff('pic2')">
```

New Page

```
</A>
```

```
<P>
```

```
<IMG SRC="story1.gif" name="pic2" width="100"  
height="25" border="0">
```

Làm thế nào để thay đổi nhiều hình ảnh cùng lúc? Sau đây là một ví dụ, di chuyển con trỏ chuột qua hình ảnh bên trên, và cả hai hình ảnh đều thay đổi.

```
<BODY>
```

```
<SCRIPT language="JavaScript">
```

```
<!--
```

```
if (document.images)  
{  
    pic1on= new Image(100,25);  
    pic1on.src="shoes2.gif";  
    pic2on= new Image(100,25);  
    pic2on.src="story2.gif";  
  
    pic1off= new Image(100,25);  
    pic1off.src="shoes1.gif";  
    pic2off= new Image(100,25);  
    pic2off.src="story1.gif";  
}
```

Căn bản thiết kế Web

```
function lightup(imgName,imgName2)
{
    if (document.images)
    {
        imgOn=eval(imgName + "on.src");
        document[imgName].src= imgOn;
        imgOn2=eval(imgName2 + "on.src");
        document[imgName2].src= imgOn2;
    }
}

function turnoff(imgName,imgName2)
{
    if (document.images)
    {
        imgOff=eval(imgName + "off.src");
        document[imgName].src= imgOff;
        imgOff2=eval(imgName2 + "off.src");
        document[imgName2].src= imgOff2;
    }
}

//-->
</SCRIPT>
</HEAD>

<BODY>
<A HREF="newpage.html"
onMouseover="lightup('pic1','pic2')"
onMouseout="turnoff('pic1','pic2')">
```

```
<IMG SRC="shoes1.gif" name="pic1" width="100"  
height="25" border="1">
```

```
</A>
```

```
<BR>
```

```
<IMG SRC="story1.gif" name="pic2" width="100"  
height="25" border="1">
```

```
</BODY>
```

Mỗi hàm bây giờ nhận hai tham số, và trong trang HTML, bạn cũng phải gọi nó với hai tham số.

Bạn có thể làm với nhiều hình ảnh, bằng cách thêm hai dòng cho từng ảnh trong mỗi hàm, và cũng phải thêm một tham số cho mỗi hàm. Ví dụ bạn muốn làm với ba hình ảnh, các hàm sẽ giống như sau:

```
function lightup(imgName,imgName2,imgName3)  
{  
  if (document.images)  
  {  
    imgOn=eval(imgName + "on.src");  
    document[imgName].src= imgOn;  
    imgOn2=eval(imgName2 + "on.src");  
    document[imgName2].src= imgOn2;  
    imgOn3=eval(imgName3 + "on.src");  
    document[imgName3].src= imgOn3;  
  }  
}
```

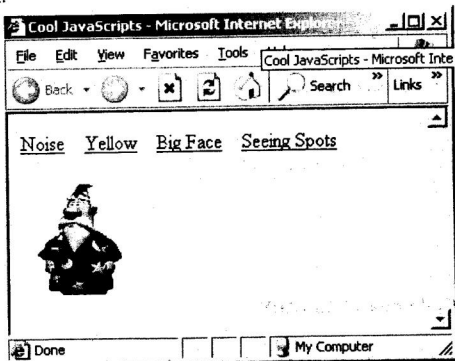
```
function turnoff(imgName, imgName2, imgName3)
{
    if (document.images)
    {
        imgOff=eval(imgName + "off.src");
        document[imgName].src= imgOff;
        imgOff2=eval(imgName2 + "off.src");
        document[imgName2].src= imgOff2;
        imgOff3=eval(imgName3 + "off.src");
        document[imgName3].src= imgOff3;
    }
}
```

và cũng phải nhớ load các ảnh này trước trong phần đầu của script. Trong trang HTML, thay đổi cách gọi hàm, và thêm vào một hình ảnh:

```
<A HREF="newpage.html"
onmouseover="lightup('pic1','pic2','pic3')"
onmouseout="turnoff('pic1','pic2','pic3')">
<IMG SRC="shoes1.gif" name="pic1" width="100"
height="25" border="1">
</A>
<BR>
<IMG SRC="story1.gif" name="pic2" width="100"
height="25" border="1">
<BR>
<IMG SRC="another1.gif" name="pic3" width="100"
height="25" border="1">
```

Cách dùng nhiều link để thay đổi một hình ảnh

Chúng ta sẽ tạo một trang giống như sau, di chuyển con trỏ chuột qua các đầu link để thấy hình ảnh bên dưới thay đổi.



Code của phần HEAD

```
<SCRIPT language="JavaScript">
```

```
<!--
```

```
if (document.images)
```

```
{
```

```
    image_off= new Image(100,100);
```

```
    image_off.src="scare.jpg";
```

```
    image2= new Image(100,100);
```

```
    image2.src="scare2.jpg";
```

```
    image3= new Image(100,100);
```

```
    image3.src="scare3.jpg";
```

Cần bản thiết kế Web

```

        image4= new Image(100,100);
        image4.src="scare4.jpg";
        image5= new Image(100,100);
        image5.src="scare5.jpg";
    }

function change1(picName,imgName)
{
    if (document.images)
    {
        imgOn=eval(imgName + ".src");
        document[picName].src= imgOn;
    }
}

//-->

</SCRIPT>

```

Code của phần BODY

[illegible]

Kiểm tra các đối tượng thay vì kiểm tra trình duyệt

Bạn đã biết cách dùng JavaScript để kiểm tra trình duyệt để chạy các script đúng với trình duyệt. Thay vì kiểm tra các trình duyệt, chúng ta kiểm tra các đối tượng (object) JavaScript chúng ta cần để script làm việc. Ví dụ, để dùng hiệu ứng với các hình ảnh cần kiểm tra đối tượng `document.images`:

```
if (document.images)
{ ...code... }
```

Bạn có thể dùng cặp object mới để kiểm tra các phiên bản trình duyệt mới. Nếu chỉ cần kiểm tra NS4, kiểm tra đối tượng `document.layers`:

```
if (document.layers)
{ ...code... }
```

Nếu chỉ muốn IE4, kiểm tra `document.all`:

```
if (document.all)
{ ...code... }
```

Dĩ nhiên, nếu bạn muốn kiểm tra một trong hai phiên bản trình duyệt 4, kiểm tra như sau:

```
if (document.all || document.layers)
{ ...code... }
```

Với IE 5:

```
if (document.getElementById && document.all)
{ ...code... }
```

Với NS 6:

```
if (document.getElementById && !document.all)
{ ...code... }
```

Cho cả hai IE5 và NS6 (hoặc cao hơn):

```
if (document.getElementById)
{ ...code... }
```

Bạn dùng các kiểm tra trên để tránh các thông báo lỗi của JavaScript trong các trình duyệt không thích hợp.

Frame và JavaScript

Kết hợp dùng JavaScript và Frame

Cái đầu tiên bạn muốn biết là “Cách dùng JavaScript để xử lý các biến trong một frame từ một frame khác”. Câu trả lời là dùng theo luật sau:

```
parent.frameName.attributes_to_change
```

Thay thế *frameName* bằng tên bạn đặt cho frame trong trang frame chính (frameset), ví dụ:

```
<FRAME src="jex14.htm" name="right_frame">
```

Trong trường hợp này, bạn xử lý cái gì đó trong frame như sau:

```
parent.right_frame.attributes_to_change
```

Hãy bắt đầu bằng việc thay đổi giá trị của textbox trong frame bên phải từ frame bên trái. Trước tiên, tạo một trang frameset. Công việc này đòi hỏi 3 trang html khác nhau. Một trang dành cho frameset, và hai trang html cho hai frame. Trang frameset như sau:

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>Frames Values</TITLE>
```

```
</HEAD>
```

```
<FRAMESET cols="20%,80%">
```

```
<FRAME SRC="page1.htm" name="left_frame">
```

```
<FRAME SRC="page2.htm" name="right_frame">
```



```
</FRAMESET>
```

```
</HTML>
```

Hai trong dùng trong hai frame là *"page1.htm"* và *"page2.htm"*. Trang bên trái (*page1.htm*) như sau:

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>JavaScript Example 13</TITLE>
```

```
</HEAD>
```

```
<BODY>
```

```
<FORM>
```

```
<INPUT type="button" value="What is cool?"
```

```
onClick="parent.right_frame.document.form1.text1.value='Me!'">
```

```
</FORM>
```

```
</BODY>
```

```
</HTML>
```

Lệnh `onClick = " "` là lệnh dùng để thực hiện thay đổi đã mô tả.

```
onClick="parent.right_frame.document.form1.text1.value='Me!'"
```

Như bạn thấy, chúng ta xử lý frame bên phải với *"parent.right_frame."* và sau đó thêm những gì muốn thay đổi vào phần đuôi. Bởi vì muốn thay đổi một vài thứ trong đối tượng `document`, nên ta phải dùng *"parent.right_frame.document."* nếu bạn bỏ *"document"* trình duyệt sẽ dùng đối tượng mặc nhiên là *"window"*. Để truy xuất đến form, dùng tên của form, *"parent.right_frame.document.form1."* Sau đó xử lý đến textbox bằng cách thêm tên của textbox vào nó: *"parent.right_frame.document.form1.text1."* Cuối cùng, thay đổi thuộc tính giá trị (`value`) của textbox bằng

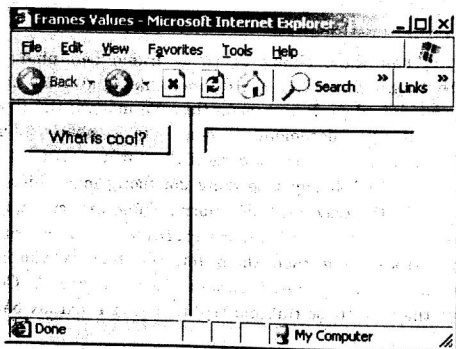
cách thêm "value" vào cuối và gán cho nó một giá trị bằng dấu bằng:

```
"parent.right_frame.document.form1.text1.value='Me!'"
```

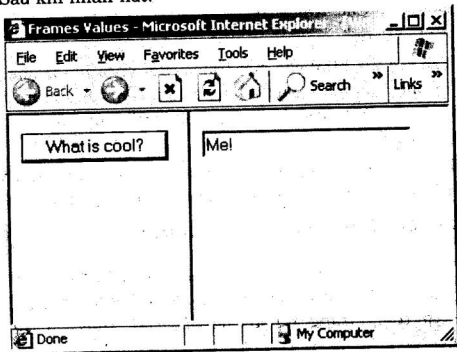
Trang HTML của frame bên phải với một form có một textbox:

```
<HTML>
<HEAD>
<TITLE>JavaScript Example 14</TITLE>
</HEAD>
<BODY>
<FORM name="form1">
<INPUT type="text" name="text1" size="25" value="">
</FORM>
</BODY>
</HTML>
```

Trước khi nhấn nút:



Sau khi nhấn nút:



Thay đổi hai Frame cùng lúc

Đầu tiên là tạo một frameset. Frameset này có hai frame như sau:

```
<HTML>
<HEAD>
<TITLE>Frames Example 5</TITLE>
</HEAD>
<FRAMESET cols="20%,80%">
<FRAME SRC="page1.htm" name="left_frame">
<FRAME SRC="page2.htm" name="right_frame">
</FRAMESET>
</HTML>
```

Bảo đảm là các frame có đặt tên với thuộc tính name=" ". Các frame trên có tên là "left_frame" và "right_frame".

Tạo các file html cho mỗi frame. Bởi vì có hai frame, và muốn thay đổi hai frame cùng lúc, nên cần 4 file html. Đặt tên chúng là "page1.htm", "page2.htm", "page3.htm" và "page4.htm".

Frame bên trái là nơi để ta đặt các link, khi nhấn vào nó cả hai frame đều thay đổi thành các trang khác.

Soạn thảo HTML cho frame bên trái

Trong trường hợp này, frame bên trái là "page1.htm". Đây là file ta sẽ làm việc trên nó. Sau đây là những gì ta cần có trong trang HTML để đầu link thay đổi nhiều frame:

```
<HTML>
<HEAD>
<TITLE>FRAME 10</TITLE>
<SCRIPT language="JavaScript">
<!--hide
function change2()
{
parent.left_frame.location="page3.htm";
parent.right_frame.location="page4.htm";
}
//-->
</SCRIPT>
</HEAD>
```

```
<BODY bgcolor="#FFFFFF" text="#000000">
<CENTER>
Click the link below to change both frames.
<BR>
<A HREF="javascript:change2()">Change 2 Frames</A>
</CENTER>
</BODY>
</HTML>
```

Xét hàm change2()

Các dòng quan trọng là :

```
parent.left_frame.location="page3.htm";
parent.right_frame.location="page4.htm";
```

Hai dòng này bảo cho trình duyệt các trang html nào bạn muốn thay đổi cho các frame. Và đây cũng là nơi bạn dùng các tên của frame đã đặt trước. Dạng tổng quát như sau:

```
parent.your_frame_name.location="url";
parent.your_other_frame_name.location="url";
```

Bạn có thể dùng một địa chỉ đầy đủ của một trang web khác. Ví dụ:

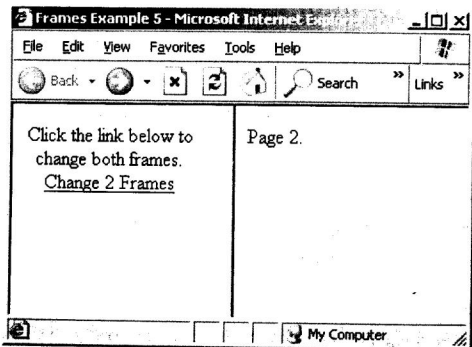
```
parent.left_frame.location="http://www.vnn.vn";
```

Xét tag Link

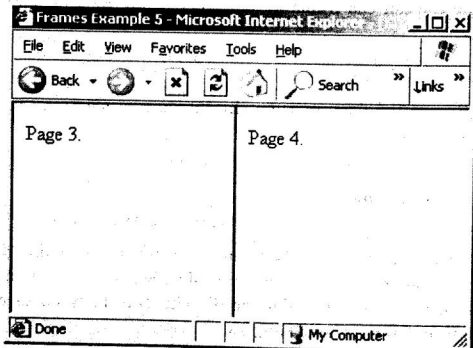
```
<A HREF="javascript:change2()">Change 2 Frames</A>
```

Chú ý là đầu link này không liên kết đến một địa chỉ bình thường. Bởi vì nó gọi hàm JavaScript của bạn, nơi mà sẽ thực hiện thay đổi các địa chỉ. Đây là lý do ta có thể thay đổi hai frame cùng lúc.

Trước khi nhấn vào link:



Sau khi nhấn vào link :



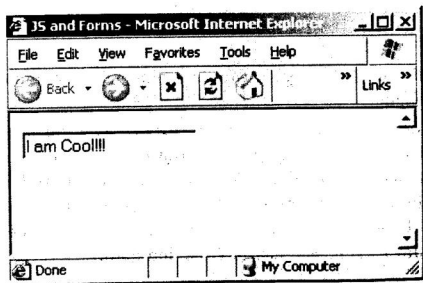
Làm việc với Form

Cách xử lý các phần tử của form

Bạn có thể dùng JavaScript để làm nhiều thứ với form. Bạn có thể xử lý nhiều phần của form bằng cách đặt tên cho form và đặt tên cho các phần tử của form. Xem form sau:

```
<FORM name="coolform">  
<INPUT type="text" name="cooltext" value="I am Cool!!!"  
size="20">  
</FORM>
```

Đoạn code này tạo một hộp nhập văn bản. Thuộc tính value="I am Cool!!!" cho hộp này một giá trị khởi đầu, và có thể được thay đổi bởi người sử dụng, hoặc bạn thay đổi nó với JavaScript.



Bạn có thể vào đó và gõ vào câu gì bạn muốn.

Cần/bản thiết kế Web

Xem lại đoạn code trên, ta đã đặt tên cho form và textbox. Đây là cách chúng ta truy xuất đến chúng dùng JavaScript. Hộp nhập văn bản textbox là một thành phần của form, và form là một phần của document. Nếu bạn nhớ đối tượng document.location trong phần trước, bạn sẽ biết cách truy xuất đến các phần của document và thay đổi chúng. Do đó, để xử lý form, chúng ta dùng:

document.coolform

Nhớ là tên của form là "coolform" có từ thuộc tính name="coolform" bên trong tag form. Để xử lý textbox, thêm tên của text box:

Bây giờ, chúng ta có thể xử lý giá trị của textbox, là những gì bên trong thuộc tính value. "I'm Cool!!!". Do đó, nếu bạn muốn lấy giá trị của text box (I'm Cool!!!), ta thêm từ "value" vào cuối:

document.coolform.cooltext.value

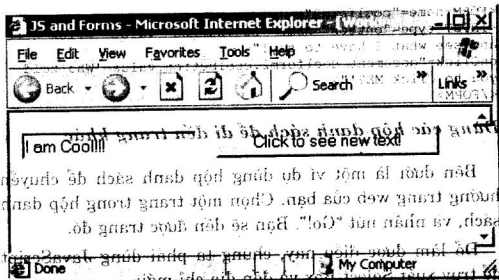
Chú ý là các đối tượng và các tên phải tách biệt nhau bằng dấu chấm.

Chúng ta đã lấy được giá trị của textbox, bây giờ hãy thử thay đổi văn bản trong text box. Một cách dễ làm điều này là cho người xem nhấn vào một button và thay đổi văn bản khi button được nhấn. Với form giống như trên, ta thêm một button bên dưới, và dùng lệnh "onClick" để thực hiện JavaScript. Sau đây là ví dụ đơn giản:

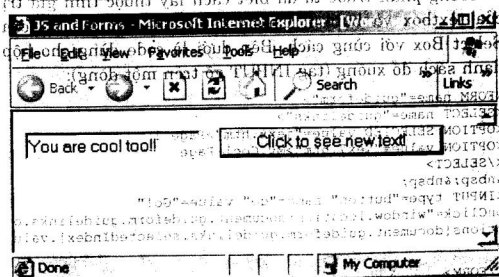
Bạn có thể vào đó và gõ vào câu gì bạn muốn.

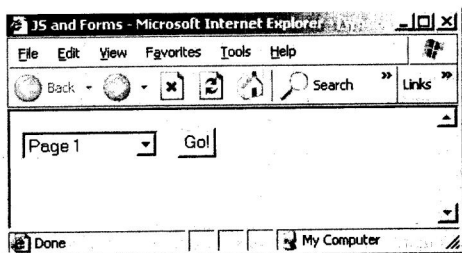
Cần bản thiết kế Web

```
<FORM name="coolform1">
<INPUT type="text" name="cooltext" value="I am Cool!!!"
size="20">
<INPUT type="button" name="change" value="Click to see
new text!"
onClick="document.coolform1.cooltext.value='You are cool
too!!'">
</FORM>
```



Sau khi nhấn nút:





Chú ý là ta có đặt tên cho form và select box. Chúng được dùng để truy xuất thuộc tính value của Select box. Chúng ta cũng cho các thuộc tính value= " " của các tag OPTION. Để ý là các value là các địa chỉ. Địa chỉ này là nơi bạn muốn trình duyệt đến khi người sử dụng chọn nó. Do đó, với mục chọn đầu tiên, <OPTION SELECTED value="jex6.htm">Page 1, chúng ta muốn trình duyệt đến trang "jex6.htm" khi người sử dụng chọn mục "Page 1". Bạn có thể đổi thành bất kỳ địa chỉ nào bạn muốn.

Phần rắc rối là sẽ truy xuất đến Select Box để người xem chuyển vị trí. Bạn thấy rằng có thể truy xuất đến Select Box bằng tên của nó:

```
document.guideform.guidelinks
```

Nhưng lệnh trên không cho chúng ta truy xuất đến các mục bên trong Select Box. Chỉ cho truy xuất đến hộp đó mà thôi. Để truy xuất đến các mục chọn lựa, chúng ta phải thêm thuộc tính "options":

```
document.guideform.guidelinks.options
```

Bây giờ thì chúng ta có thể đụng đến các mục trong danh sách, chúng ta truy xuất đến các mục riêng biệt dùng

một vài ký hiệu đặc biệt. (Các mục trong Select Box tạo thành một mảng). Mục đầu tiên là mục 0, được viết như sau `options[0]`. Mục thứ hai là `options[1]`. Nhớ rằng các mảng bắt đầu từ chỉ số 0, cho nên ta dùng `options[0]` để truy xuất trực tiếp đến mục đầu tiên.

Bây giờ bạn có thể lấy giá trị của mục đầu tiên như sau:

```
document.guideform.guidelinks.options[0].value
```

Giá trị sẽ là "jex6.htm". Nhưng vấn đề là bạn phải biết người sử dụng chọn địa chỉ nào để đến. Xác định được điều này bằng cách dùng một thuộc tính mới, đó là `selectedIndex`.

Thuộc tính `selectedIndex` cho biết mục mà người sử dụng đã chọn, cho nên nó không phải lúc nào cũng như nhau. Để lấy giá trị của mục mà người sử dụng chọn, cần phải dùng đến tên của form, và tên của Select Box, một lần nữa:

```
document.guideform.guidelinks.options[document.guideform.guidelinks.selectedIndex].value
```

Bạn cũng nên chú ý đến script trong tag button, nó gọi các lệnh JavaScript trong lệnh `onClick=""`. Button sẽ thực hiện thay đổi địa chỉ bằng cách thay đổi giá trị của `window.location`:

```
<INPUT type="button" name="go" value="Go!"  
onClick="window.location=document.guideform.guidelinks.o  
ptions[document.guideform.guidelinks.selectedIndex].valu  
e">
```

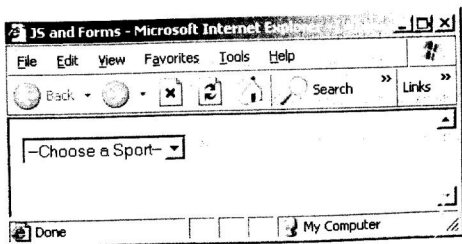
Dùng hộp danh sách không cần nhấn nút

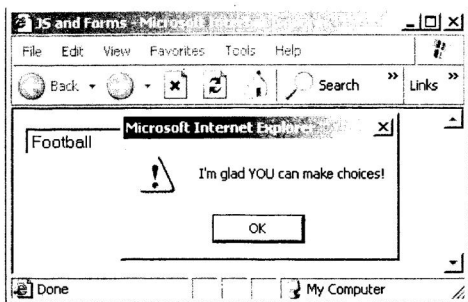
Bạn đã biết dùng hộp danh sách và một button ở phần trước, bây giờ sẽ dùng hộp danh sách tự động chuyển sang trang khác khi người sử dụng chọn mà không đòi hỏi phải

nhấn một button. Nghĩa là bạn muốn thay đổi ngay khi người sử dụng đã chọn.

Code cho script này thực sự ngắn hơn so với dùng một button. Trong phần này bạn học một cái mới là lệnh onChange. Lệnh onChange là một cách gọi JavaScript khác khi có gì đó xảy ra. Nó được viết code giống như viết code với onClick hay onMouseover. Sự khác nhau là onChange là thuộc tính dùng cho các phần tử của form, như select box. Thuộc tính onChange sẽ thực hiện các câu lệnh JavaScript khi có thay đổi gì đó trên phần tử của form, trong trường hợp này là Select box. Sau đây là một ví dụ về thuộc tính onChange, nó gửi cho người sử dụng một thông báo khi họ chọn một cái gì đó:

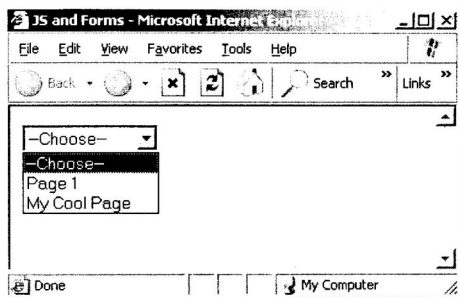
```
<FORM>
<SELECT name="sport" onChange="alert('I\'m glad YOU can
make choices!')">
<OPTION SELECTED>--Choose a Sport--
<OPTION>Football
<OPTION>Basketball
</SELECT>
</FORM>
```





Bạn có thể dùng thuộc tính `onChange` để làm các công việc giống như là `onClick` trong một button. Nói cách khác, dùng nó để thay đổi địa chỉ khi người sử dụng chọn. Sau đây là script làm điều này:

```
<FORM name="guideform">
<SELECT name="guidelinks"
onChange="window.location=document.guideform.guidelinks.options[
document.guideform.guidelinks.selectedIndex].value">
<OPTION SELECTED value="jdrop2.htm">--Choose--
<OPTION value="jex15.htm">Page 1
<OPTION value="jex16.htm">My Cool Page
</SELECT>
</FORM>
```



Sử dụng các hộp danh sách với các frame

Bạn muốn dùng hộp danh sách trong một frame để thay đổi địa chỉ của một frame khác. Dùng theo luật sau:

parent.frameName.location

Những gì bạn cần làm là thay đổi frameName thành tên của frame mà bạn muốn thay đổi.

Giả sử có một frameset như sau, dùng frame bên trái để thay đổi frame bên phải:

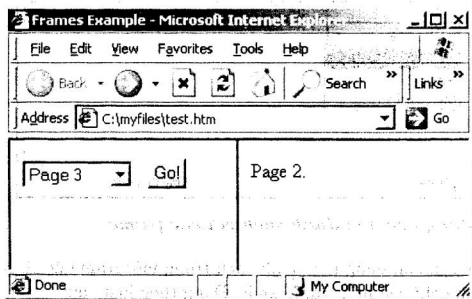
```
<HTML>
<HEAD>
<TITLE>Frames Example </TITLE>
</HEAD>
<FRAMESET cols="20%,80%">
<FRAME SRC="page1.htm" name="left_frame">
<FRAME SRC="page2.htm" name="right_frame">
<NOFRAMES>
Sorry, you can't see this page..
```

Căn bản thiết kế Web

</NOFRAMES>

</FRAMESET>

</HTML>

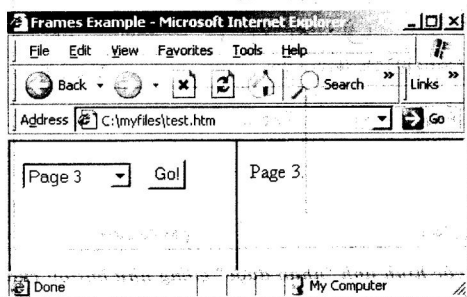


Soạn thảo trang HTML bên trái, cho vào một hộp danh sách và một nút. Code đầy đủ như sau:

```
<FORM name="guideform">
<SELECT name="guidelinks">
<OPTION SELECTED value="page3.htm">Page 3
<OPTION value="page4.htm">Cool Page
</SELECT>
 
<INPUT type="button" name="go" value="Go!"
onClick="parent.right_frame.location=document.guideform.
guidelinks.options[document.guideform.guidelinks.selecte
dIndex].value">
</FORM>
```

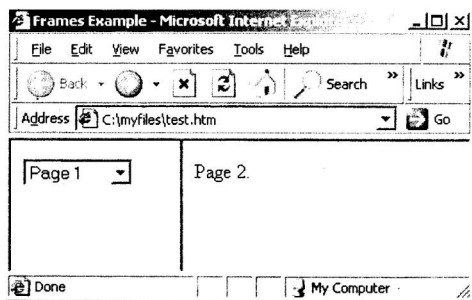

Căn bản thiết kế Web

Sau khi nhấn nút:



Nếu bạn dùng danh sách không cần button, code sẽ như sau:

```
<FORM name="guideform">  
<SELECT name="guidelinks"  
onChange="parent.right_frame.location=document.guideform  
.guidelinks.options[document.guideform.guidelinks.select  
edIndex].value">  
<OPTION SELECTED value="jex20.htm">Page 1  
<OPTION value="jex21.htm">Cool Page  
</SELECT>  
</FORM>
```



Tạo các hình ảnh “nhấn được” giống như button của form

Với các trình duyệt phiên bản 4 hoặc mới hơn cho ta thêm một số biến cố với con trỏ chuột. Button này dùng `onMouseover` và `onMouseout`, và hai cái mới: `onMouseDown` và `onMouseUp`.

onMouseDown Người sử dụng nhấn chuột.

onMouseUp Người sử dụng thả chuột.

Nó khác với `onClick` như thế nào? Với các biến cố này, chúng ta chia biến cố nhấn chuột thành hai phần riêng biệt: Nhấn chuột xuống, và thả chuột ra. Với cách này chúng ta có thể thay đổi hình ảnh theo các thao tác như trong một form. Xem đoạn HTML sau:

```
<A HREF="javascript:void(0)"
onMouseover="lightup('pic1')"
onMouseout="turnoff('pic1')"
onMouseDown="clickdown('pic1')"
onMouseUp="lightup('pic1')">
```

Căn bản thiết kế Web

```
<IMG SRC="upbutlav.gif" name="pic1" width="62"
height="28" border="0">
</A>
```

Chú ý là script này không làm gì khi chuột được nhấn trên hình ảnh. Hàm JavaScript trong HREF là void(), hàm này sẽ không làm gì nếu gửi vào nó giá trị 0. Cho nên, bạn có thể tạo một link mà không liên kết đến đâu cả:

```
<A HREF="javascript:void(0)">Click Here!</A>
```

Để hoàn tất công việc với hình ảnh này, cần phải nhờ các hàm. Đây là script trong phần HEAD:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
if (document.images)
{
    piclon= new Image(62,28);
    piclon.src="upbutblue.gif";
    picloff= new Image(62,28);
    picloff.src="upbutlav.gif";
    picldown=new Image(62,28);
    picldown.src="dbutblue.gif";
}

function lightup(imgName)
{
    if (document.images)
    {
        imgOn=eval(imgName + "on.src");
        document[imgName].src= imgOn;
    }
}

function turnoff(imgName)
{
    if (document.images)
    {
        imgOff=eval(imgName + "off.src");
        document[imgName].src= imgOff;
    }
}
```

Căn bản thiết kế Web

```
function clickdown(imgName)
{
    if (document.images)
    {
        imgDown=eval(imgName + "down.src");
        document[imgName].src=imgDown;
    }
}
//-->
</SCRIPT>
</HEAD>
```

Script này giống với script của *Hover button*, nhưng với hai sự thay đổi. Nâng cấp lên phần kiểm tra đối tượng, và thêm một hàm mới `clickdown()` để dùng trong biến cố `onMouseDown`.

Khi dùng script này, phải bảo đảm là đã có ba hình ảnh. Script này cần dùng ba hình ảnh như sau:



upbutlav.gif

Đây là hình ảnh dùng khi không có gì xảy ra.



upbutblue.gif

Hình này được dùng khi người sử dụng di chuyển chuột qua hình ảnh, và khi thả chuột ra (`onMouseUp`).



dbutblue.gif

Đây là hình ảnh sử dụng khi người sử dụng nhấn chuột (`onMouseDown`).

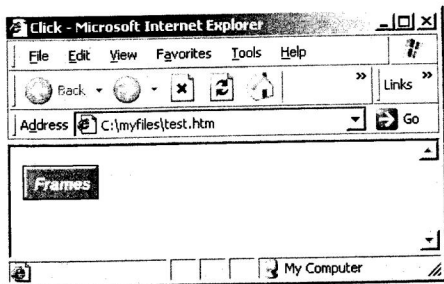
Hàm mới ở đây cũng làm công việc tương tự như các hàm cũ, nó thay đổi hình ảnh. Nếu xem lại HTML, bạn sẽ thấy hàm `clickdown()` được gọi với biến cố

onMouseDown. Nó làm cho hình ảnh thay đổi khi người sử dụng nhấn chuột vào button.

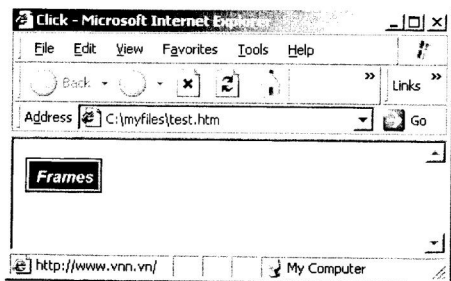
Bây giờ trong trang HTML bạn có thể dùng một địa chỉ thay vì dùng void(0):

```
<A HREF="http://www.vnn.vn"
onMouseover="lightup('pic1')"
onMouseout="turnoff('pic1')"
onMouseDown="clickdown('pic1')"
onMouseUp="lightup('pic1')">
<IMG SRC="upbutlav.gif" name="pic1" width="62"
height="28" border="0">
</A>
```

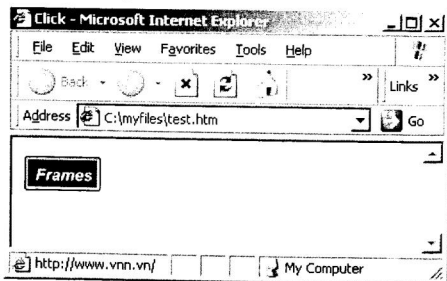
Sau đây là kết quả:



Khi di chuyển chuột qua hình ảnh:



Khi nhấn chuột vào hình ảnh:



Cách làm cho một file JavaScript làm việc trên nhiều trang html.

Sử dụng JavaScript bên ngoài cho phép bạn gọi một hàm JavaScript trên bất kỳ trang nào mà bạn có đặt tag `<SCRIPT></SCRIPT>` đặc biệt.

Tag SCRIPT này cũng giống như các tag SCRIPT bình thường, nhưng nó có thêm một thuộc tính SRC: SRC="http://someplace.com/script_name.js"

Sau đây là một ví dụ:

```
<SCRIPT language="JavaScript"  
SRC="http://www.mysite.com/jscript/jxt1.js">  
</SCRIPT>
```

Thuộc tính SRC cho biết địa chỉ của file JavaScript mà bạn muốn dùng. File script này phải được lưu dưới dạng file văn bản (plain text).

Mở một file văn bản mới trong trình soạn thảo văn bản, và viết các dòng code sau:

```
Function a_message()  
{  
alert('I came from an external script! Ha, Ha, Ha!!!!');  
}
```

Lưu nó lại với tên "jxt1.js"

Bây giờ bạn có thể dùng script này trên bất kỳ trang nào bạn muốn. Đặt tag SCRIPT trong tag <HEAD> </HEAD>. Bảo đảm là thuộc tính SRC chỉ tới file .js. Nếu file html và file .js trên cùng một thư mục, dùng tag như sau:

```
<SCRIPT language="JavaScript" SRC="jxt1.js"></SCRIPT>
```

Nếu các file này trên các thư mục khác nhau, dùng địa chỉ cụ thể. Ví dụ:

```
<SCRIPT language="JavaScript"  
SRC="http://someplace.com/jxt1.js"></SCRIPT>
```

Bây giờ bạn có thể gọi hàm thông báo từ trang html mà bạn đặt tag này. Chúng ta dùng một đầu link làm ví dụ:

```
<A HREF="javascript:a_message()">Click for a  
message..</A>
```

Script này không hữu dụng nhưng nó cho bạn biết cách sử dụng một hàm JavaScript từ bên ngoài. Bạn có thể viết các script khác và dùng nó với cách này.

Đây là cách dùng một script trên nhiều trang. Bằng cách thêm các tag SCRIPT vào các trang bạn cần dùng script, bạn có thể dùng lại các script mà không cần gõ chúng lại trên từng trang.

Đối tượng ngày tháng (Date Object)

Sử dụng date object và các phương thức (method) của nó để tạo một đồng hồ.

Để bắt đầu, chúng ta tìm hiểu về date object của JavaScript. Đối tượng này làm việc có một chút khác biệt so với các hàm đã định nghĩa trước mà chúng ta đã dùng. Để sử dụng các phương thức của date object, chúng ta phải tạo một cái gọi là *instance* của data object. Xem instance của date object như là một biến để cho phép chúng ta truy xuất đến các hàm thành phần của date object. Tại sao xem nó như là một biến? Bởi vì để tạo một instance của date object, bạn định nghĩa nó như là một biến mới:

```
var thetime=new Date();
```

Như bạn thấy, chúng ta không gán trực tiếp cho biến một giá trị. Thay vào đó, dòng code trên định nghĩa một biến như là một instance mới của date object. Chúng ta có thể dùng biến này để truy xuất các phương thức của date object. Sau đây là danh sách các phương thức:

Method	Mô tả
getHours()	Trả về số giờ hiện hành (0-23)
getMinutes()	Trả về số phút của giờ hiện hành (0-59)
getSeconds()	Trả về số giây của phút hiện hành (0-59)
getDay()	Trả về ngày hiện hành trong tuần (0-6)
getMonth()	Trả về tháng hiện hành trong năm (0-11)
getYear()	Trả về năm hiện hành (0-99)

Đây không phải là tất cả các phương thức, nhưng chúng đủ để tạo một đồng hồ. Ví dụ này chỉ cần dùng ba hàm đầu tiên, các hàm còn lại hữu dụng cho việc tùy biến đồng hồ.

Giả sử bạn muốn lấy giờ hiện hành trong ngày. Bạn có thể làm bằng cách định nghĩa một biến và gán cho nó giá trị của hàm `getHours()`. Tuy vậy dòng lệnh sau là sai:

```
var nhours=getHours();
```

Bạn phải nhớ rằng `getHours()` là một phương thức của `date object`. Để nhận được giá trị cho các biến, chúng ta phải dùng instance đã tạo lúc trước:

```
var thetime=new Date();  
var nhours=thetime.getHours();
```

Đây là một phần của lập trình hướng đối tượng. Ngay khi chúng ta tạo một đối tượng (object), chúng ta có thể truy xuất đến các hàm thành phần của nó với toán tử chấm (.). Bằng cách đặt dấu chấm giữa tên đối tượng (thetime) và hàm thành phần của nó, chúng ta đã thực thi hàm thành phần. Trong trường hợp này, hàm chỉ đơn

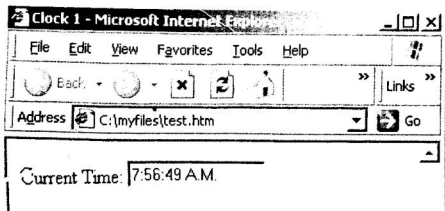
giản là trả về một giá trị (số giờ hiện hành trong ngày). Chúng ta dùng giá trị này như là giá trị của một biến mới, nhours.

Chúng ta đã sử dụng toán tử chấm trong các phần trước rồi, khi sử dụng hàm `document.write()`. Hàm `write()` là một hàm thành phần của đối tượng `document`, và hàm này ghi văn bản ra màn hình.

Một điểm thuận lợi của việc dùng các đối tượng là các đối tượng có thể chứa nhiều giá trị, trong khi các biến chỉ có thể giữ một giá trị tại một thời điểm. Một đối tượng cũng có thể dùng bất kỳ hàm thành phần nào mà nó có. Do đó, đối tượng `date` được đặt tên là `thetime` có thể dùng tất cả các hàm thành phần. Bây giờ, chúng ta lấy tất cả các giá trị còn lại của đồng hồ. Lấy giá trị giờ, phút và giây:

```
var thetime=new Date();  
var nhours=thetime.getHours();  
var nmins=thetime.getMinutes();  
var nsecn=thetime.getSeconds();
```

Với các giá trị này và thêm một ít code, chúng ta có thể tạo một đồng hồ như sau:



Chúng ta bắt đầu sử dụng thêm một số code để đồng hồ chạy đẹp hơn. Nhớ là `thetime.getHours()` trả về một số là giờ trong một ngày. Giá trị này là một số từ 0 đến 23. Nếu bạn không thích trình bày giờ theo 24 giờ, bạn có thể viết thêm một ít code để chuyển các giờ lớn hơn 12 thành 1,2,3.... Chúng ta có thể làm điều này bằng cách lấy giá trị của `nhours` trừ cho 12 khi nó lớn hơn 12.

```
if (nhours>=13) nhours-=12;
```

```
nhours-=12; ngắn hơn: nhours=nhours-12;
```

Chúng ta không có vấn đề như trên với phút và giây, nhưng chúng có vấn đề về thể hiện. Giả sử số phút là 32, script sẽ hiển thị là 32, nhưng số phút là 2, script sẽ hiển thị giống như sau:

11:2

Chúng ta cần số 0 đứng trước để dễ đọc hơn, như sau: 11:02. Do đó, chúng ta phải viết code để thêm số 0 vào trước số phút nếu nó nhỏ hơn 10.

```
if (nmins<10)  
nmins="0"+nmins;
```

Tương tự, chúng ta làm như thế với giây:

```
if (nsecn<10)  
nsecn="0"+nsecn;  
if (nmins<10)  
nmins="0"+nmins;
```

Như vậy vẫn chưa đủ. Đồng hồ hiển thị A.M hay P.M để biết là buổi sáng hay buổi chiều. Chúng ta có thể định nghĩa một biến khác, gọi là `AorP`. Chúng ta cần giá trị là "P.M" khi số giờ lớn hơn 12, và ngược lại là "A.M". Code như sau:

```
if (nhours>=12)
AorP="P.M.";
else
AorP="A.M.";
```

Đoạn code này phải được đặt trước đoạn code chuyển giờ sang hệ 12. Nếu không, bạn sẽ luôn nhận được là "A.M".

Sau đây là đoạn code đầy đủ:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
function startclock()
{
var thetime=new Date();

var nhours=thetime.getHours();
var nmins=thetime.getMinutes();
var nsecn=thetime.getSeconds();
var AorP=" ";

if (nhours>=12)
    AorP="P.M.";
else
    AorP="A.M.";

if (nhours>=13)
    nhours-=12;

if (nhours==0)
    nhours=12;

if (nsecn<10)
    nsecn="0"+nsecn;
if (nmins<10)
    nmins="0"+nmins;
```

Căn bản thiết kế Web

```
document.clockform.clockspot.value=nhours+":"+nmins+":"+nsecn+" "+AorP;
setTimeout('startclock()',1000);
}
//-->
</SCRIPT>
</HEAD>
<BODY onLoad="startclock()">
<FORM name="clockform">
Current Time: <INPUT TYPE="text" name="clockspot"
size="15">
</FORM>
</BODY>
```

Chú ý tới đoạn code xử lý giá trị của một thành phần của form:

```
document.clockform.clockspot.value=nhours+":"+nmins+":"+nsecn+" "+AorP;
```

Do đó, bạn cần có một form trong trang HTML (document), và form này chứa một textbox.

```
<FORM name="clockform">
Current Time: <INPUT TYPE="text" name="clockspot"
size="15">
</FORM>
```

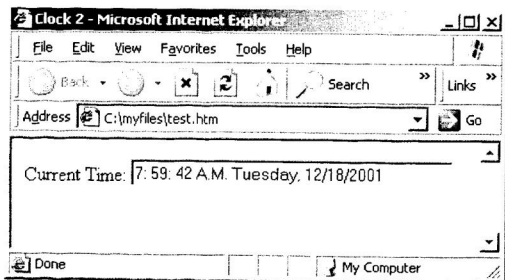
Bạn có thể thấy chúng ta sử dụng hàm `setTimeout()` để chạy hàm `startclock()` sau khoảng thời gian là một giây. Với cách này, đồng hồ sẽ chạy từng giây, và bạn có một đồng hồ chạy đúng theo từng giây.

Đĩ nhiên là trình duyệt không tự động chạy hàm `startclock()`. Nó cần một biến cố xảy ra. Biến cố xảy ra khi trang web được tải là thích hợp nhất, ta sẽ dùng biến cố `onLoad`. Biến cố `onLoad` là một thuộc tính của tag `BODY`, sau đây là cách dùng lệnh `onLoad`:

```
<BODY onLoad="startclock()">
```

Thêm một vài tính năng mới cho đồng hồ

Có thể bạn muốn đồng hồ có cả thứ và ngày. Có thể trông giống như sau:



Có thêm một số thông tin, đòi hỏi phải viết thêm script để nó làm việc theo những gì chúng ta muốn.

Bây giờ, chúng ta xét script. Chú ý là chúng ta gọi nhiều các hàm thành phần hơn của đối tượng date để nhận thêm thông tin. Sau đây là script:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--

function startclock()
{
var thetime=new Date();

var nhours=thetime.getHours();
var nmins=thetime.getMinutes();
```

```
var nsecn=thetime.getSeconds();  
var nday=thetime.getDay();  
var nmonth=thetime.getMonth();  
var ntoday=thetime.getDate();  
var nyear=thetime.getYear();  
var AorP=" ";
```

```
if (nhours>=12)  
    AorP="P.M.";
```

```
else  
    AorP="A.M.";
```

```
if (nhours>=13)  
    nhours-=12;
```

```
if (nhours==0)  
    nhours=12;
```

```
if (nsecn<10)  
    nsecn="0"+nsecn;
```

```
if (nmins<10)  
    nmins="0"+nmins;
```

```
if (nday==0)  
    nday="Sunday";
```

```
if (nday==1)  
    nday="Monday";
```

```
if (nday==2)  
    nday="Tuesday";
```

```
if (nday==3)  
    nday="Wednesday";
```

```
if (nday==4)  
    nday="Thursday";
```

```
if (nday==5)  
    nday="Friday";
```

Căn bản thiết kế Web

```
if (nday==6)
    nday="Saturday";

nmonth+=1;

if (nyear<=99)
    nyear= "19"+nyear;

if ((nyear>99) && (nyear<2000))
    nyear+=1900;

document.clockform.clockspot.value=nhours+":
"+nmins+": "+nsecn+" "+AorP+" "+nday+",
"+nmonth+"/"+ntoday+"/"+nyear;

setTimeout('startclock()',1000);

}


```


Đối tượng toán học (Math Object)

Các phương thức hữu dụng cho việc tính toán

Đối tượng toán học trong JavaScript cho phép bạn thực hiện các phép tính toán bằng cách dùng các hàm của Math object. Và nó cũng hỗ trợ một số hằng số, như pi. Cách dễ nhất để nhận được giá trị là khai báo một biến và gán giá trị của nó bằng thuộc tính hay hàm của đối tượng Math.

Nếu bạn muốn dùng một thuộc tính, nó trả về một giá trị hằng số (ví dụ: pi), bạn sẽ viết giống như sau:

```
var my_car=Math.property;
```

Bạn thay thế từ “property” với thuộc tính bạn cần lấy. Nếu bạn muốn lấy giá trị của pi, bạn phải dùng thuộc tính PI của đối tượng Math. Bạn có thể viết như sau:

```
var my_pie=Math.PI;
```

Dòng này trả về hằng số pi, một số khoảng 3.14....

Nếu bạn muốn sử dụng một hàm thành phần, để thực hiện một phép tính, bạn viết giống như sau:

```
var my_house=Math.function(x);
```

Đa số các hàm thành phần có một hoặc nhiều tham số, đó là những gì “x” đại diện. Bạn có thể thay thế x bằng một số hay một biến. Và thay thế “function” bằng tên hàm mà bạn muốn sử dụng. Ví dụ, bạn muốn lấy căn bậc hai của một số, bạn có thể gọi hàm thành phần tính căn bậc hai của đối tượng Math với một số làm tham số:

```
var my_money=Math.sqrt(2);
```

Dòng này cho ra căn bậc hai của 2, là một số khoảng 1.41..

Cần chú ý là “M” trong từ “Math” luôn luôn viết hoa. Từ Math được theo sau là một dấu chấm và tiếp theo là thuộc tính hay hàm bạn muốn sử dụng.

Sau đây là một số thuộc tính có thể sử dụng, các thuộc tính này trả về một hằng số:

Math.PI	Trả về hằng số pi
Math.SQRT2	Trả về căn bậc hai của 2
Math.SQRT1_2	Trả về căn bậc hai của $\frac{1}{2}$

Bên dưới là bảng các hàm thành phần thường sử dụng của Math object. Các hàm này tính toán các giá trị giữa trên các tham số gửi vào chúng (ngoại trừ hàm `random()`).

Method Function	Mô tả
<code>abs(x)</code>	Trả về giá trị tuyệt đối của biến x
<code>cos(x)</code>	Trả về cos của biến x.
<code>log(x)</code>	Trả về logarit tự nhiên của biến x.
<code>max(x, z)</code>	Trả về số lớn hơn của hai biến x và z.
<code>min(x, z)</code>	Trả về số nhỏ hơn của hai biến x và z.
<code>pow(x, z)</code>	Trả về giá trị x^z .
<code>random()</code>	Trả về một số ngẫu nhiên giữa 0 và 1.
<code>round(x)</code>	Trả về giá trị làm tròn của x đến số nguyên gần nhất.
<code>sin(x)</code>	Trả về giá trị sin của x.
<code>sqrt(x)</code>	Trả về căn bậc hai của biến x.
<code>tan(x)</code>	Trả về giá trị tan của biến x.

Lấy các số và các chuỗi ngẫu nhiên

Hàm `random()` của đối tượng `Math` cho phép bạn lấy các số phát sinh ngẫu nhiên để dùng trong script. Bạn có thể viết một script để phát sinh một chuỗi ngẫu nhiên hay các kiểu ngẫu nhiên khác.

Mặc nhiên, hàm `random` trả về một số ngẫu nhiên giữa 0 và 1. Với chừng này thì không có gì hữu dụng. Số mà nó phát sinh ra bạn chẳng có thể làm gì với nó. Bước đầu tiên để khắc phục điều này, ta dùng phép nhân. Nếu bạn muốn một số ngẫu nhiên từ 0 đến 4, nhân giá trị của `Math.random()` với 4:

```
var ran_number= Math.random()*4;
```

Số nhận được là số nằm giữa 0 và 4 nhưng nó vẫn chưa là một số nguyên. Để nhận được một số ngẫu nhiên là số nguyên, dùng hàm `Math.round()` với tham số là kết quả trên:

```
var ran_number=Math.round(Math.random()*4);
```

Bạn cũng có thể dùng hai bước để có một số ngẫu nhiên là số nguyên:

```
var ran_unrounded=Math.random()*4;  
var ran_number=Math.round(ran_unrounded);
```

Hãy xây dựng một đoạn script nhỏ với mục đích thông báo cho người xem một thông báo ngẫu nhiên khi họ nhấn vào button. Ta đã biết cách lấy số ngẫu nhiên, và chỉ cần một mảng để lưu trữ các câu thông báo – sau đó dùng nó với một số ngẫu nhiên. Chúng ta cần 5 chuỗi, do đó cần số ngẫu nhiên từ 0 đến 4. Bắt đầu bằng cách tạo một hàm, và đặt một biến để nhận được một số ngẫu nhiên:

```
function get_random()
{
    var ranNum= Math.round(Math.random()*4);
    return ranNum;
}
```

Hàm này chỉ đơn giản trả về một số ngẫu nhiên từ 0 đến 4. Hàm kế tiếp sẽ gọi hàm này và gán giá trị trả về cho một biến tên `whichQuote`, biến này làm chỉ số cho một mảng chứa các chuỗi. Do đó cần định nghĩa một mảng, và khởi tạo nó với năm chuỗi:

```
function getaQuote()
{
    var whichQuote=get_random();
    var quote=new Array(5)
    quote[0]="I love JavaScript..sometimes.";
    quote[1]="Why are you pushing my button?";
    quote[2]="The button you pushed can\'t push you
back. You bully, you!";
    quote[3]="This alert is here to inform you that
alerts are annoying.";
    quote[4]="Which came first, the button or the
alert?";

    alert(quote[whichQuote]);
}
```

Như bạn thấy, dòng cuối cùng của hàm cho người sử dụng một thông báo. Chuỗi thông báo là chuỗi được lưu trong mảng các chuỗi:

```
quote[whichQuote]
```

Biến `whichQuote` đã được gán một số ngẫu nhiên nằm giữa 0 và 4 từ hàm `get_random()`. Do đó, nếu giá trị của `wichQuote` là 3, bạn sẽ nhận một thông báo có nội dung là chuỗi trong `quote[3]`. Mỗi khi button được nhấn, nó chạy lại và gán một số ngẫu nhiên khác.

Đối với button, code trong phần BODY như sau:

```
<FORM name="form1">
<INPUT TYPE="button" value="Get a Quote!"
onClick="getaQuote()">
</FORM>
```

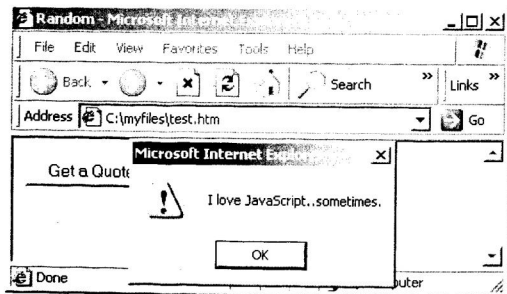
Đây là toàn bộ script:

```
<HEAD>
<SCRIPT language="JavaScript">
<!--
function get_random()
{
    var ranNum= Math.round(Math.random()*4);
    return ranNum;
}
function getaQuote()
{
    var whichQuote=get_random();
    var quote=new Array(5)
    quote[0]="I love JavaScript..sometimes.";
    quote[1]="Why are you pushing my button?";
    quote[2]="The button you pushed can\'t push you
back. You bully, you!";
    quote[3]="This alert is here to inform you that
alerts are annoying.";
    quote[4]="Which came first, the button or the
alert?";

    alert(quote[whichQuote]);
}
```

```
//-->
</SCRIPT>
</HEAD>

<BODY>
<FORM name="form1">
<INPUT TYPE="button" value="Get a Quote!"
onClick="getaQuote()">
</FORM>
</BODY>
```



Bạn có thể dùng nhiều chuỗi hơn bằng cách tăng kích thước mảng và thay đổi số bạn nhân với hàm `Math.random()`.

Xử lý chuỗi

Sử dụng `charAt` và `indexOf`

Hai phương thức này, `charAt` và `indexOf`, được sử dụng để tìm ra ký tự nào tại một vị trí trong chuỗi và vị trí của ký tự ở trong chuỗi hay tìm một chuỗi nhỏ hơn

trong một chuỗi. Trước khi tìm hiểu hai hàm này, ta cũng cần tìm hiểu hàm lấy chiều dài (length). Hàm này cho độ dài của một chuỗi, giá trị này rất hữu dụng cho các hàm khác.

length

Phương thức length trả về một số là độ dài của một chuỗi. Ví dụ nếu chuỗi dài 5 ký tự, hàm length sẽ trả về giá trị 5. Để sử dụng các hàm, bạn cần đặt các chuỗi trong các biến. Ví dụ, chỉ đơn giản gán một chuỗi vào một biến:

```
var my_car="Ferrari";
```

Để dùng phương thức length, chúng ta dùng toán tử chấm (.). Trước tiên dùng tên biến, sau đó dùng toán tử chấm và theo sau là từ "length", như vậy sẽ trả về độ dài của chuỗi:

```
my_car.length
```

Nếu bạn muốn dùng lại giá trị độ dài, bạn có thể gán nó vào một biến:

```
var how_long=my_car.length;
```

Trong trường hợp này, biến how_long sẽ có giá trị 7, bởi vì từ "Ferrari" có bảy ký tự. Phương thức length hữu dụng hơn khi dùng các giá trị được nhập vào từ người sử dụng và bạn không biết chuỗi đó là gì. Dùng phương thức này, bạn có thể lấy được chiều dài và dùng nó cho mục đích khác, như là thông báo lỗi khi họ nhập chuỗi dài hơn mức bạn quy định:

```
<SCRIPT language="JavaScript">
```

```
<!--
```

```
function getname()
```

```
{
```

```
var yourname=prompt('Enter your name, NOW!','');
if (yourname.length > 10)
{
    alert('That name is just too long, give me a shorter
one!');
    getname();
}
else
    alert('Hi '+yourname+'!');
}
//-->
</SCRIPT>
```

charAt

Sử dụng hàm `charAt`, bạn có thể tìm ra ký tự nào đặt tại vị trí được chỉ định trong chuỗi. Về cơ bản, nó cho phép bạn biết được ký tự thứ nhất, thứ hai, thứ ba, Điều quan trọng cần nhớ là khi tính vị trí của ký tự trong chuỗi là tính từ 0. Do đó, ký tự đầu tiên chính xác là tại vị trí 0.

Cũng như phương thức `length`, chúng ta gán chuỗi vào một biến, và dùng toán tử chấm để gọi phương thức `charAt`. Phương thức `charAt` lấy một đối số, đó là số chỉ vị trí của ký tự nó trả về. Do đó, nếu bạn muốn tìm ký tự đầu tiên trong chuỗi "Ferrari", có thể làm như sau:

```
var my_car="Ferrari";
var the_char=my_car.charAt(0);
alert('The 1st character is '+the_char+'.');
```

Đoạn code này cho chúng ta một thông báo "The first character is F." Có thể làm tương tự bằng cách gửi vào 1 làm tham số để nhận ký tự thứ hai....

Cũng tương tự với ký tự cuối cùng, chúng ta có thể dùng phương thức `length` để tìm ra ký tự nào ở vị trí cuối cùng trong chuỗi. Bởi vì độ dài tính từ 1, và vị trí tính từ 0, do đó vị trí cuối cùng của chuỗi là độ dài trừ đi 1. "Ferrari" có độ dài là 7, nhưng vị trí cuối cùng là 6 (đếm từ 0).

```
var my_car="Ferrari";  
var the_length=my_car.length;  
var last_char=my_car.charAt(the_length-1);  
alert('The last character is '+the_char+'.');
```

Chúng ta sẽ nhận được thông báo "The last character is i."

indexOf

Phương thức này cho phép kiểm tra một ký tự hay một chuỗi nhỏ với chuỗi hiện hành, và trả về vị trí của ký tự hay vị trí bắt đầu của chuỗi. Phải nhớ là vị trí tính từ 0, do đó nếu `indexOf` trả về 0, có nghĩa là ký tự hay chuỗi bắt đầu tại ký tự đầu tiên. Hàm `indexOf` trả về -1 nếu không tìm thấy ký tự hay chuỗi cần tìm trong chuỗi gốc. Nếu ký tự hay chuỗi cần tìm xuất hiện nhiều lần trong chuỗi gốc, `indexOf` chỉ trả về vị trí xuất hiện đầu tiên của ký tự trong chuỗi.

Trước tiên, hãy tìm ký tự 'a'. Hàm `indexOf` cũng cần một đối số giống như `charAt`, nhưng đối số này là một ký tự hay một chuỗi mà bạn cần tìm:

```
var my_car="Ferrari";  
var where_is_a=my_car.indexOf('a');  
alert('The a is at position '+where_is_a+'.');
```

Chú ý dấu móc bọc quanh đối số 'a'. Đoạn code này sẽ trả về vị trí của chữ a. Nội dung của thông báo là: "The a is at position 4."

Bây giờ tìm ký tự 'r' trong "Ferrari". Như bạn thấy, 'r' xuất hiện ba lần trong chuỗi. Nhưng chỉ trả về vị trí xuất hiện đầu tiên, tại vị trí 2:

```
var my_car="Ferrari";  
var where_is_r=my_car.indexOf('r');  
alert('The r is at position '+where_is_r+'.');
```

Câu thông báo bây giờ là "The r is at position 2."

Nếu không tìm thấy chuỗi cần tìm, `indexOf` sẽ trả về giá trị -1. Giá trị này có thể hữu dụng khi bạn hỏi thông tin từ người sử dụng, nhưng người sử dụng không nhập vào các thông tin có các ký tự bắt buộc, ví dụ:

```
function get_url()  
{  
var your_url=prompt('Enter your web page URL!','');  
var is_protocol_ok=your_url.indexOf('http://');  
if (is_protocol_ok==-1)  
{  
    alert('Error: Your url should begin with http://');  
    get_url();  
}  
else  
    alert('Thanks!');  
}
```

Đoạn code này kiểm tra để chắc chắn rằng người sử dụng có nhập vào `http://` như là một phần của địa chỉ. Nếu không, nó hiển thị một thông báo lỗi. Và bạn có thể dùng `http://` như là giá trị mặc nhiên của `prompt`:

```
var your_url=prompt('Enter your web page  
URL!','http://');
```

Bạn cũng có thể kiểm tra thêm là phải có ít nhất một dấu chấm (.) trong địa chỉ:

```
function get_url()
{
var your_url=prompt('Enter your web page
URL!', 'http://');
var is_protocol_ok=your_url.indexOf('http://');
var is_dot_ok=your_url.indexOf('.');
if ((is_protocol_ok==-1) || (is_dot_ok==-1))
{
    alert('Error: Your url should begin with http:// and
have at least one dot (.)!');
    get_url();
}
else
    alert('Thanks!');
}
```

Sử dụng phương thức split

Phương thức split dùng để chia một chuỗi thành hai hay nhiều phần dựa trên một ký tự để chia các phần. Ký tự để chia có thể là bất kỳ ký tự nào. Ví dụ, xem chuỗi sau:

```
var where_is_mytool="home/mytool/mytool.cgi";
```

Chú ý là dấu số có thể được dùng để chia chuỗi trên thành ba phần. Có thể lấy được thư mục home, thư mục chứa file (mytool), và tên file. Để làm điều này, chúng ta có thể dùng phương thức split.

```
var where_is_mytool="home/mytool/mytool.cgi";
var mytool_array=where_is_mytool.split("/");
```

Phương thức `split` lấy một tham số, đó là ký tự mà bạn muốn dùng để chia chuỗi gốc ra. Trong trường hợp này, nó là ký tự `"/"`. Bạn cũng chú ý là cái mà chúng ta nhận được đó là một mảng (biến để ta nhận kết quả có tên là `mytool_array`). Mảng này chứa ba phần, với phần đầu tiên chứa trong `mytool_array[0]`. Bây giờ chúng ta có thể dùng các phần của chuỗi này cho các công việc khác:

```
<SCRIPT language="JavaScript">
<!--
function divide_string()
{
var where_is_mytool="home/mytool/mytool.cgi";
var mytool_array=where_is_mytool.split("/");
alert(mytool_array[0]+" "+mytool_array[1]+" " +
mytool_array[2]);
}
//-->
</SCRIPT>
<FORM>
<INPUT TYPE="button" onClick="divide_string()"
value="Go!">
</FORM>
```

Đoạn code này sẽ xuất một thông báo khi bạn nhấn button.

Một vấn đề khác cần nhớ là có những lúc bạn không biết có bao nhiêu phần trong mảng sau khi dùng `split` (ví dụ, dữ liệu do người dùng nhập vào). Để bảo đảm là bạn lấy được tất cả chúng, bạn có thể dùng một vòng lặp và phương thức `length` để lấy tất cả chúng. Ví dụ:

```
<SCRIPT language="JavaScript">
<!--
```

```
function divide_string2()
{
var my_colors=prompt("What are your favorite colors?
(separate with a space)","");
if ((my_colors=="") || (my_colors==null))
{
    alert("You have no favorite colors? Bummer.");
}
else if (my_colors.indexOf(" ")== -1)
{
    alert("Either you only like one color or you didn't
separate with a space.");
    alert("You entered "+my_colors);
}
else
{
var col_array=my_colors.split(" ");
var part_num=0;
while (part_num < col_array.length)
{
    alert(col_array[part_num]);
    part_num+=1;
}
}
}
//-->
</SCRIPT>
<FORM>
<INPUT TYPE="button" onClick="divide_string2()"
value="Go!">
</FORM>
```

Chú ý là phương thức `length` trả về số các phần tử trong mảng, nó lớn hơn chỉ số của phần tử cuối cùng 1 đơn vị. Đó là lý do tại sao ta dùng `<` thay vì `<=`.

Đến đây bạn có thể dùng `split`, và các hàm xử lý chuỗi để kiểm tra tính đúng đắn của các địa chỉ web, địa chỉ email,...

Phần III ASP

Active Server Pages

Active Server Pages là gì?

Active Server Pages (ASPs) là các trang Web có chứa thêm các script chạy trên server (*server-side scripts*) để kết hợp văn bản và các tag HTML (Hypertext Markup Language). Server-side scripts là các lệnh đặc biệt đặt trong trang web, chúng sẽ được xử lý trước khi trang web được gửi từ Web Server đến trình duyệt của người sử dụng. Khi bạn gõ địa chỉ vào hộp địa chỉ hay nhấn vào một link trong trang Web, nghĩa là bạn yêu cầu Web Server trên một máy tính ở đâu đó gửi một file đến trình duyệt Web (thường gọi là client) trên máy tính của bạn. Nếu file đó là một file HTML bình thường, nó không có gì khác nhau khi trình duyệt nhận nó và trước khi Web server gửi nó đi.

Đối với ASP, quá trình cũng tương tự như trên, ngoại trừ một bước xử lý phụ thực hiện trước khi Web server gửi file. Trước khi Web Server gửi các file là các trang ASP cho trình duyệt Web, nó chạy tất cả các server-side scripts có chứa trong trang web. Một số script này hiển thị ngày tháng, thời gian và các thông tin khác. Các script khác xử lý các thông tin mà người sử dụng nhập vào một form....

Để phân biệt với các trang HTML bình thường, các trang ASP có phần mở rộng là .asp.

Bạn có thể làm gì với Active Server Pages?

Bạn có thể làm rất nhiều thứ với Active Server Pages.

- Hiển thị ngày tháng, thời gian, và các thông tin khác với nhiều cách khác nhau.
- Bạn có thể tạo một form yêu cầu thông tin từ người sử dụng và lưu trữ thông tin, hay gửi email...
- Tạo các ứng dụng Web tương tác với CSDL.

Trang ASP trông như thế nào?

Diện mạo của trang ASP phụ thuộc vào bạn xem nó bằng cái gì. Trên trình duyệt Web nhận nó, một trang ASP trông như một trang HTML bình thường. Nếu người sử dụng xem source (View ->Source) của trang ASP, họ sẽ thấy một trang HTML bình thường. Tuy nhiên, file ASP được đặt trên server trông rất khác. Ngoài các tag HTML và các văn bản bình thường, bạn còn thấy các server-side script. Đó là những gì bạn thấy trang ASP trên server trước khi nó được gửi đáp lại một yêu cầu đến nó.

Các server-side script trông như thế nào?

Server-side script trông khá giống các tag HTML. Tuy nhiên, thay vì bắt đầu và kết thúc bằng dấu nhỏ hơn (<) và dấu lớn hơn (>), các server-side script bắt đầu bằng <% và kết thúc với %>. Ký hiệu <% gọi là tag mở, và %> gọi là tag đóng. Giữa hai tag này là các server-side script. Bạn có thể chèn server-side script ở bất kỳ đâu trong trang Web – ngay cả trong các tag HTML.

Hiển thị ngày giờ hiện hành

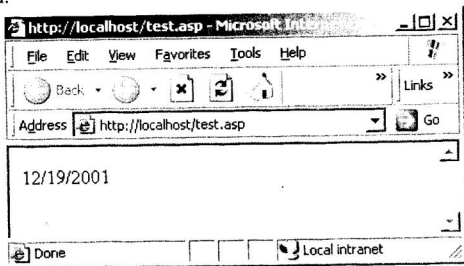
Ngày và giờ trong phần này là ngày giờ trên máy tính server.

Date

Để hiển thị ngày hiện hành trong trang Web, nhập dòng code sau:

```
<% =date %>
```

vào vị trí bạn muốn nó xuất hiện. Khi bạn xem trang web trong trình duyệt, bạn sẽ thấy nội dung tương tự như sau:



Chú ý: Mặc dù "=date" là một script ngắn, nhưng nó thực sự có hai phần. Phần "date" báo cho server lấy ngày tháng hiện hành. Dấu bằng (=) báo cho server hiển thị ngày tháng trên trang web. Nếu bạn chỉ nhập vào:

```
<% date %>
```

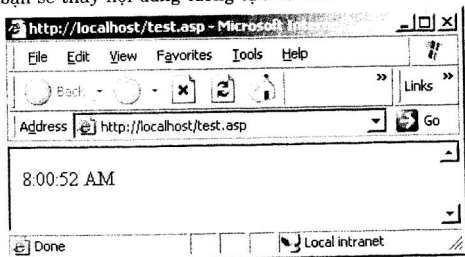
server lấy ngày tháng hiện hành của hệ thống, nhưng chỉ có vậy. Nó sẽ không được hiển thị.

Time

Để hiển thị giờ hiện hành, gõ:

```
<% =time %>
```

vào vị trí bạn muốn nó xuất hiện. Khi bạn xem trang web, bạn sẽ thấy nội dung tương tự như sau:

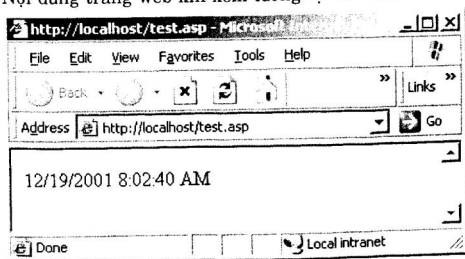


Now (Date và Time)

Để hiển thị ngày tháng và giờ hiện hành:

```
<% =now %>
```

Nội dung trang web khi xem tương tự như sau:



Thay đổi cách hiển thị Date và Time

Bạn cũng có thể dùng các hàm của ASP để tùy biến cách hiển thị ngày tháng trên trang web. Để làm điều này bạn kết hợp hàm `now` và một số hàm sau:

Month và Monthname

Để hiển thị tháng hiện hành trong trang Web, gõ:

```
<% =month(now) %>
```

vào vị trí bạn muốn nó xuất hiện. Khi bạn xem trang web, bạn sẽ thấy 1 nếu tháng hiện hành là tháng một, 2 nếu tháng hiện hành là tháng hai....

Để hiển thị tên của tháng hiện hành:

```
<% =monthname(month(now)) %>
```

Day

Để hiển thị ngày của tháng hiện hành:

```
<% =day(now) %>
```

Khi xem trang web, bạn sẽ thấy một số nằm giữa 1 và 31.

Year

Hiển thị năm hiện hành:

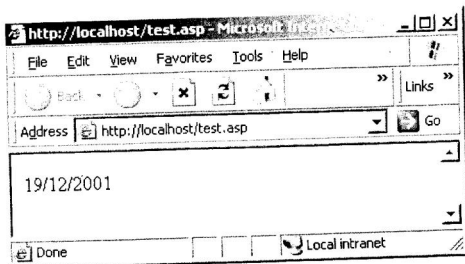
```
<% =year(now) %>
```

Ví dụ

Giả sử bạn muốn hiển thị ngày theo dạng *ngày/tháng/năm* thay vì *năm/tháng/ngày*. Để làm như trên, bạn dùng các hàm `day`, `month`, và `year` của ASP:

```
<% =day(now) %>/<% =month(now) %>/<% =year(now) %>
```

Kết quả khi xem trang Web là:



Weekday và Weekdayname

Để hiển thị ngày trong tuần như là số từ 1 đến 7 trong trang Web:

```
<% =weekday(now) %>
```

Khi xem trong trình duyệt IE, bạn sẽ thấy 1 nếu là chủ Nhật, 2 nếu là thứ Hai...

Hiển thị ngày trong tuần theo tên:

```
<% =weekdayname(weekday(now)) %>
```

Hour, Minute, và Second

Hiển thị giờ hiện hành:

```
<% =hour(now) %>
```

Hàm hour tính theo 24 giờ. Khi xem trang web, bạn sẽ thấy một số giữa 0 và 23.

Hiển thị phút hiện hành:

```
<% =minute(now) %>
```

Kết quả là một số từ 0 đến 59.

Hiển thị giây hiện hành:

```
<% =second(now) %>
```

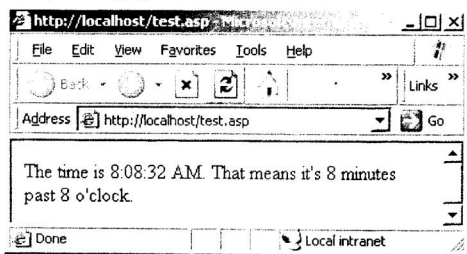
Khi hiển thị trên trang web, bạn sẽ thấy một số giữa 0 và 59.

Ví dụ

Thử cho đoạn code sau vào một trang web:

```
The time is <% =time %>. That means it's <% =minute(now) %> minutes past <% =hour(now) %> o'clock.
```

Kết quả tương tự như sau:



Timevalue

Từ đầu đến giờ bạn chưa dùng hàm timevalue. Nó cho một cách viết thời gian khác, như "2:24PM" và "14:24" sẽ viết chúng thành "2:24:00 PM". Hàm này hữu dụng nếu bạn muốn in thời gian theo một định dạng nào đó,

Ví dụ:

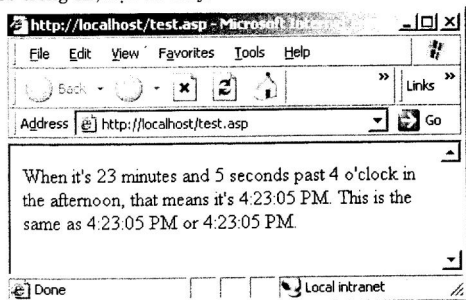
Trong phần trước chúng ta dùng các hàm hour, minute, và second để chia nhỏ thời gian thành các phần giờ, phút, giây. Với hàm timevalue, bạn có thể gán chúng lại với nhau. Nhập vào đoạn code sau vào trang web:

```
When it's 23 minutes and 5 seconds past 4 o'clock in the afternoon,  
that means it's <% =timevalue("16:23:05") %>.
```

Căn bản thiết kế Web

This is the same as `<% =timevalue("4:23:05PM") %>`
or `<% =timevalue("16:23:05PM") %>`.

Bảo đảm là bạn nhập "16:23:05PM", không phải là "16:23:05 PM". Số "05" và "PM" phải đi liền nhau, không cách nhau bởi một khoảng trắng. Khi xem trang Web trong IE, bạn sẽ thấy:



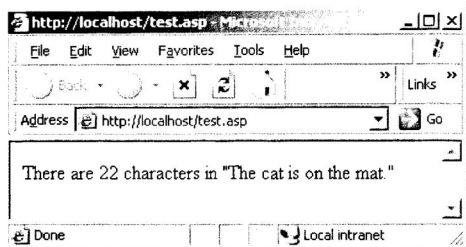
Hiển thị văn bản

len

Hàm len báo cho bạn biết có bao nhiêu ký tự trong một chuỗi ký tự. Tất cả các ký tự đều được đếm, kể cả khoảng trống. Ví dụ, để tìm chiều dài của câu "The cat is on the mat.", nhập vào trang web:

There are `<% =len("The cat is on the mat.") %>`
characters in "The cat is on the mat."

Khi xem trang web trên trình duyệt, sẽ thấy:

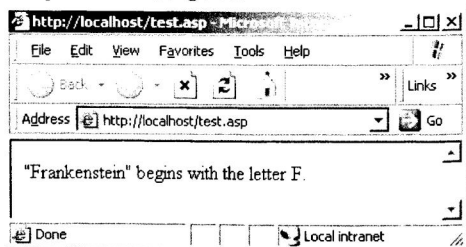


left

Bạn có thể dùng hàm `left` để lấy một số ký tự đầu tiên của một chuỗi. Ví dụ, để tìm ký tự đầu tiên của "Frankenstein", nhập vào trang web đoạn code sau:

```
"Frankenstein" begins with the letter <%  
=left("Frankenstein", 1) %>.
```

Kết quả khi xem trang web:

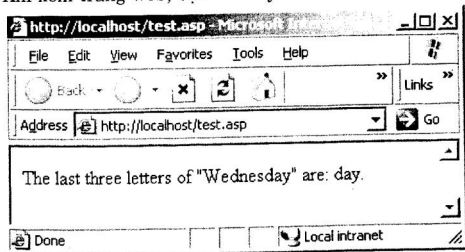


right

Để lấy một số ký tự cuối cùng của một chuỗi, sử dụng hàm `right`. Ví dụ, để tìm ba ký tự cuối cùng của "Wednesday", nhập vào trang Web:

```
The last three letters of "Wednesday" are: <%  
=right("Wednesday", 3) %>.
```

Khi xem trang web, bạn sẽ thấy:



Ví dụ

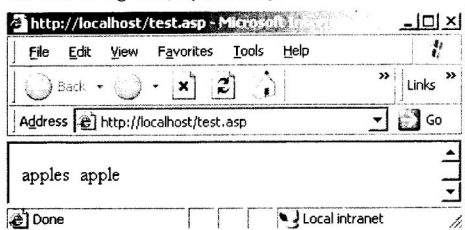
Bạn sẽ làm gì để lấy một vài ký tự nằm giữa một chuỗi. Ví dụ làm cách nào bạn lấy "apple" trong chuỗi "pineapples"?

Bạn có thể bắt đầu với ký tự thứ 5 từ bên trái và kết thúc tại ký tự thứ hai từ bên phải. Hoặc bạn có thể làm như cách sau:

```
<% =right("pineapples", 6) %> <%  
=left(right("pineapples", 6), 5) %>
```

Dòng code trên lấy 6 ký tự cuối của "pineapples" cho ra "apples". Sau đó lấy 5 ký tự đầu tiên của "apples", ta có được "apple".

Khi xem trang web, bạn sẽ thấy:

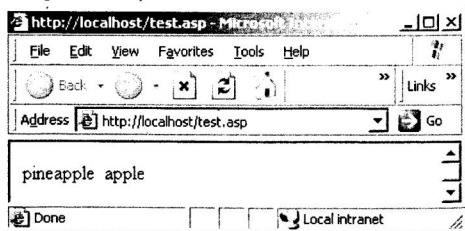


Thử nhập dòng code sau vào trang Web:

```
<% =left("pineapples", 9) %> <%  
=right(left("pineapples", 9), 5) %>
```

Dòng trên lấy 9 ký tự đầu tiên của "pineapples", ta được "pineapple". Sau đó lấy 5 ký tự cuối của "pineapple", cuối cùng được "apple".

Kết quả hiển thị là:



Sử dụng các hàm ngày tháng, thời gian và văn bản

Sau đây là một số ví dụ thú vị mà bạn có thể làm với các hàm ngày tháng, thời gian và văn bản.

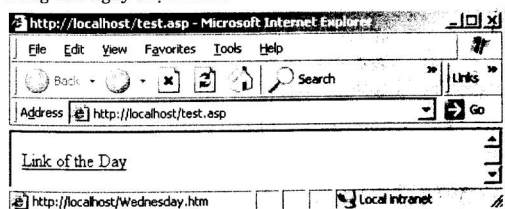
Link theo ngày (Day)

Giả sử bạn muốn tạo một link trở đến các trang khác nhau theo ngày trong tuần. Trước tiên, chọn các file HTML trên Web site của bạn mà bạn muốn link trở đến. Đặt tên chúng là "Sunday.htm", "Monday.htm", Bạn phải tạo đủ các file tương ứng với các ngày trong tuần.

Để tạo link như trên, nhập dòng sau:

```
<a href= <% =weekdayname(weekday(now)) %>.htm>Link of  
the Day</a>
```

vào vị trí bạn muốn link xuất hiện trên trang web. Khi bạn nhấn vào link trên trình duyệt, nó sẽ đưa bạn đến trang của ngày hiện hành.

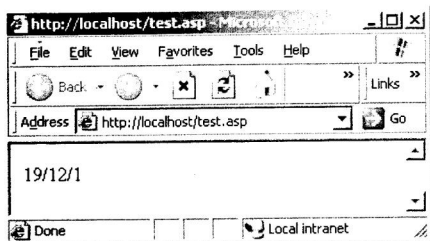


Hiển thị ngày tháng theo một cách khác

Phần trước ta đã biết cách đổi ngày từ tháng/ngày/năm sang ngày/tháng/năm. Bạn cũng có thể thay đổi ngày hiển thị chỉ với hai số cuối của năm, như sau:

```
<% =day(now) %>/  
<% =month(now) %>/  
<% =(year(now) - 1900) mod 100 %>
```

Khi xem trang web, sẽ thấy như sau:



Hiển thị thời gian theo cách khác

Trong ví dụ của phần trước, chúng ta viết một server-side script hiển thị thời gian hiện hành giữa các từ như: “The time is 36 minutes and 5 seconds past 13 o'clock.”. Script này sử dụng hàm hour của ASP, nó chỉ trả về giờ hiện hành, tính theo 24 giờ.

Trong ví dụ này, sẽ chỉ bạn cách chuyển thời gian 24-giờ sang thời gian 12-giờ. Để làm điều này, chúng ta cần tạo một server-side script sử dụng hàm hour và một số code phụ. Thay vì

```
<% =hour(now) %> o'clock
```

chúng ta sẽ viết một script lấy thông tin giờ và làm theo các điều kiện sau:

- Nếu giờ là 0, script hiển thị “midnight”.
- Nếu giờ là 12, script hiển thị “noon”.
- Nếu giờ là giá trị giữa 1 và 11, script không thay đổi gì, nhưng hiển thị “AM” sau “o'clock”.
- Nếu giờ là giá trị giữa 13 và 23, script trừ nó đi 12 và hiển thị “PM” sau “o'clock”.

Đoạn code như bên dưới. Nó không giống cách các chuyên gia làm, nhưng nó làm việc đúng, và đoạn code này khá dễ hiểu, bởi vì nó làm chính xác như các điều kiện mô tả ở trên.

The hour is

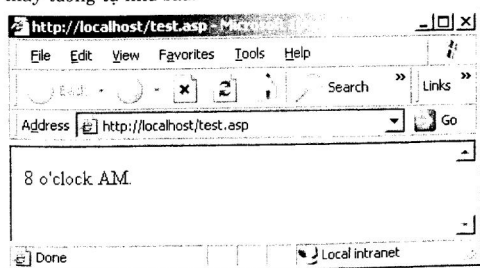
```
<% if hour(now) = 0 then %>
    midnight.
<% end if

    if hour(now) = 12 then %>
        noon.
<% end if

    if (hour(now) >= 1) and (hour(now) <= 11) then %>
<%     =hour(now) %> o'clock AM.
<% end if

    if (hour(now) >= 13) and (hour(now) <= 23) then %>
<%     =hour(now) - 12 %> o'clock PM.
<% end if %>
```

Nếu bạn hiển thị trang web có chứa đoạn code này, bạn sẽ thấy tương tự như sau:



Sử dụng biến, và form trong Active Server Pages

Form là một cách thuận tiện để trao đổi với người sử dụng trang Web. Sử dụng các form, bạn có thể tạo một form để người sử dụng nhập thông tin vào. Khi họ nhập đủ thông tin, bạn có thể xử lý kết quả một cách tự động.

Làm việc với form, có hai bước: trước tiên là tạo form, và sau đó xử lý nó. Để tạo một form cho một trang ASP, chỉ cần tạo một form HTML bình thường.

Để thử ví dụ này, tạo một file HTML ("form_response.html") có nội dung như sau:

form_response.html

```
<html>
<head><title>Asking for information</title></head>
<body>
<form method="post" action="form_response.asp">
Your name: <input type="text" name="name"
size="20"><BR>
Your email: <input type="password" name="email"
size="15"><BR>
<input type="Submit" value="Submit">
</form>
</body>
</html>
```

Active Server Page cung cấp một kỹ thuật để xử lý form đó: Request.Form

Tương ứng với form ở trên, chúng ta tạo một file asp để đưa ra trả lời:

form_response.asp

```
<html>
<head><title>Responding to a form</title></head>
<body>
Your name is <%=Request.Form("name")%> <BR>
Your email is <%=Request.Form("email")%>
```

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

Để hiển thị nội dung của một trường (field) trong form, nhập:

```
<% =Request.Form(fieldname) %>
```

với fieldname là tên của một *field* trong form.

Tạo một biến

Bạn muốn làm nhiều hơn với form, không chỉ là hiển thị lại nội dung của chúng lên trang web. Ví dụ, dựa trên nội dung của form, bạn tạo một biến và chèn biến đó vào một số vị trí khác trong trang trả lời (*response page*). Bạn cần tạo một biến. Để làm điều đó, chỉ cần tạo một tên và gán nó bằng nội dung của field.

Ví dụ, nếu bạn có một field "CatName" trong form, bạn có thể lưu nó trong một biến "TheName" bằng cách sau:

```
<% TheName = Request.Form("CatName") %>
```

Nếu bạn muốn thể hiện tên nhiều lần trong phần thân của trang web, chỉ cần cho biến vào trong văn bản. Ví dụ:

```
My cat's name is <% =TheName %>. Do you want to see
<% =TheName %>?.
```

Form trong ví dụ này hỏi người sử dụng tên của họ và màu ưa thích. Khi form được nhận trên server, server trả lời hiển thị các thông tin này:

```
nameandcolor.html
```

```
<html>
<head><title>Name and Color</title></head>
<body>
<FORM ACTION="nameandcolor.asp" METHOD=POST>
Let me know your Name and Favorite Color:
<P>YOUR NAME:
<INPUT TYPE="TEXT" NAME="YOURNAME" SIZE=20>
```

```
<P>COLOR:
<INPUT TYPE="RADIO" NAME="COLOR" VALUE="1" CHECKED>Red
<INPUT TYPE="RADIO" NAME="COLOR" VALUE="2">Green
<INPUT TYPE="RADIO" NAME="COLOR" VALUE="3">Blue
<P>
<INPUT TYPE="SUBMIT" VALUE="OK">
</FORM>
</body>
</html>
```

Tạo một file ASP ("nameandcolor.asp") có nội dung như sau:

nameandcolor.asp

```
<html>
<head><title>Name and Color</title></head>
<body>
<% TheName = Request.Form("YOURNAME") %>
<% colornumber = Request.Form("COLOR") %>
Hi, <% =TheName %>.<BR>
I know your favorite color is
<% if colornumber = "1" then %>
red
<% end if %>
<% if colornumber = "2" then %>
green
<% end if %>
<% if colornumber = "3" then %>
blue
<% end if %>.
</body>
</html>
```

If...Then...Else

Chỉ thị If....Then....Else cũng tương tự như trong các ngôn ngữ script khác. Xét ví dụ sau:

```
<%  
AA="water"  
  
If AA="water" Then  
  response.write ("I want to drink water")  
Else  
  response.write ("I want to drink milk")  
End If  
%>
```

Bạn cũng có thể viết nó theo cách sau:

```
<% AA="water"  
If AA="water" Then %>  
    I want to drink water  
  
<% Else %>  
    I want to drink milk  
  
<% End If %>
```

Trong cả hai trường hợp chúng ta kiểm tra điều kiện (AA= "water"), và nhờ một lệnh ghi ra lời khẳng định. Chúng ta có thể cho thực hiện bất kỳ lệnh nào trong chỉ thị này.

For....Next

Tương tự, chỉ thị này cũng tương tự như các chỉ thị trong các ngôn ngữ script khác. Xét ví dụ đơn giản sau:

```
I want to say "Hello" 10 times<BR>  
<% For mynumber = 1 to 10 %>  
<% =mynumber %> Hello<BR>  
<% Next %>  
END
```

Trong trường hợp này chúng ta đã định nghĩa một biến ("mynumber") và dùng chỉ thị For....Next để lặp lại 10 lần "Hello".

Chỉ thị For...Next cho phép định nghĩa giá trị bước tăng:

```
<% For mynumber = 1 to 20 STEP 2  
response.write("Hello<BR>")  
Next %>
```

```
<% For mynumber = 20 to 1 STEP -2  
response.write("Hello<BR>")  
Next %>
```

Trong cả hai trường hợp chúng ta đều nhận được cùng một kết quả ("Hello" 10 lần). Bước tăng có thể là số dương hay số âm như bạn thấy trong ví dụ.

Do While...Loop

Tiếp theo chúng ta sẽ định nghĩa vòng lặp với một điều kiện và một hay nhiều câu lệnh:

```
<%  
mynumber=0  
Do While mynumber<10  
    response.write("Hello<HR>")  
    mynumber=mynumber+1  
Loop  
%>
```

Trong ví dụ này, điều kiện là "mynumber<10" và các lệnh là in ra văn bản và tăng giá trị của biến "mynumber". Trong ví dụ này, mynumber sẽ tăng cho đến khi nó đạt giá trị 10. Sau đó vòng lặp sẽ kết thúc.

Do Until....Loop

Khá giống với vòng lặp trước, nó bao gồm một điều kiện và một hay nhiều câu lệnh:

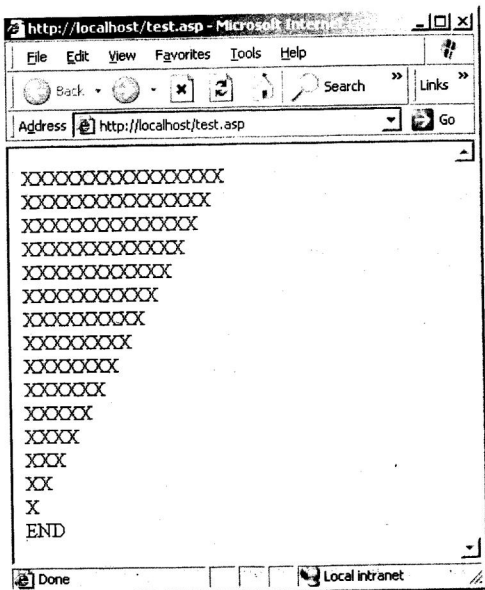

```
<%  
mynumber=0  
Do Until mynumber=10  
    response.write("Hello<HR>")  
    mynumber=mynumber+1  
Loop  
%>
```

Trong ví dụ này, điều kiện là “mynumber=10”, do đó mynumber sẽ tăng cho đến khi nó bằng 10, và sau đó kết thúc vòng lặp.

Xét một ví dụ dùng vòng lặp Do Until...Loop:

```
<%  
myfirstnumber=0  
mysecondnumber=0  
Do Until myfirstnumber=15  
    Do Until mysecondnumber=15  
        response.write("X")  
        mysecondnumber=mysecondnumber+1  
    Loop  
    Response.write("<BR>")  
    myfirstnumber=myfirstnumber+1  
    mysecondnumber=myfirstnumber  
Loop  
Response.write("END")  
%>
```

Kết quả của script trên là:



Select Case....End Select

Đây là một chỉ thị rất hữu dụng trong trường hợp chúng ta muốn kiểm tra các giá trị khác nhau của một biến. Xét ví dụ sau:

```
1  <%
2  mynumber=3
3  Select Case mynumber
4      Case 1
5          Response.write ("Number 1")
6      Case 2
7          Response.write ("Number 2")
8      Case 3
9          Response.write ("Number 3")
10     Case 4
11         Response.write ("Number 4")
12     Case 5
13         Response.write ("Number 5")
14     Case Else
15         Response write ("Mynumber is higher than 5")
16 End Select
17 %>
```

Trong ví dụ trên, chúng ta định nghĩa mynumber là 3, do đó nó sẽ thực hiện các lệnh bên dưới dòng 8 (trong trường hợp này chỉ có một lệnh được thực hiện, nhưng có thể đặt nhiều lệnh ở đây). Case Else không nhất thiết phải có.

Xét ví dụ thứ hai:

```
1  <%
2  username=request.form("username")
3  Select Case username
4      Case "Peter"
5          Response.write ("Hello, Peter")
6      Case "John"
7          Response.write ("Hello, John")
8      Case "Joe"
9          Response.write ("Hi, Joe")
10     Case Else
11         Response write ("I do not know you")
12 End Select
13 %>
```

Xét một ví dụ khác:

Backgroundform.html

```
<html>
<head><title>Chose background color</title></head>
<form action="backgroundresponse.asp" method="post">
Which color do you prefer to use as your background?
<BR>
<input type="radio" name="kindofcolor" value="defined"
checked>
Defined color
<select name="definedcolor">
<option value="#FFFFFF">White</option>
<option value="#FF0000">Red</option>
<option value="#00FF00">Green</option>
<option value="#0000FF">Blue</option>
</select>
<BR>
<input type="radio" name="kindofcolor" value="custom">
Custom color
<input type="text" size="8" name="mycolor"></input>
<BR><input type="Submit" value="Submit"></input>
</form>
</body>
</html>
```

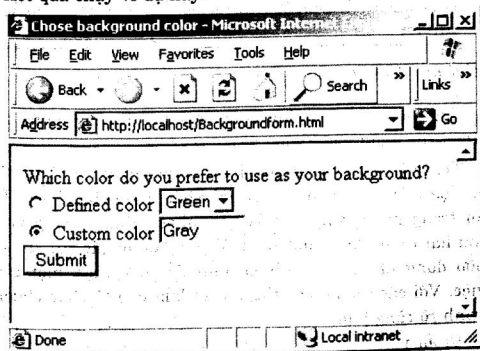
backgroundresponse.asp

```
<%
kindofcolor=Request.form("kindofcolor")
Select Case kindofcolor
  case "defined"
    colorofbackground=Request.form("definedcolor")
    Select Case colorofbackground
      case "#FFFFFF"
        texttoshow="White"
      case "#FF0000"
        texttoshow="Red"
      case "#00FF00"
        texttoshow="Green"
      case "#0000FF"
        texttoshow="Blue"
```

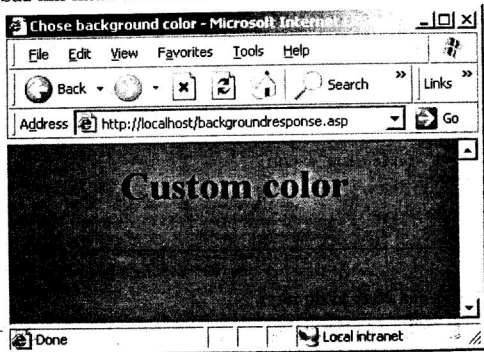
```
End select
case "custom"
  colorofbackground=Request.form("mycolor")
  texttoshow="Custom color"
End select
%>

<html>
<head><title>Chose background color</title></head>
<body bgcolor="<% =colorofbackground %>">
<center>
<H1><% =texttoshow %></H1>
</center>
</form>
</body>
</html>
```

Kết quả chạy ví dụ này:



Sau khi nhấn nút Submit:



Viết các thủ tục

Các thủ tục (hàm) ứng dụng trong ASP cũng giống như trong các ngôn ngữ khác.

Trong hai ví dụ kế tiếp, chúng ta hỏi người sử dụng tên, và dựa trên đó để có các trả lời khác nhau cho client. Trả lời trong hai trường hợp là như nhau, nhưng trong ví dụ thứ hai ta sử dụng một hàm. Việc sử dụng các thủ tục rất hữu dụng khi có quá nhiều lệnh để thực hiện một công việc. Với cách dùng các thủ tục sẽ làm cho tổ chức chương trình rõ ràng hơn.

Ví dụ 1

```
<%  
TheName=request.form("name")  
  
if TheName="John" then  
response.write ("Hi, John. How are you?")  
response.write ("<br>Did you know I got married last
```

```
month?")
else
response.write ("Hi. How are you?")
end if
%>
```

Ví dụ 2

```
<%
TheName=request.form("name")

if TheName="John" then
ResponseToJohn()
else
ResponseToUnknown()
end if

Sub ResponseToJohn()
response.write ("Hi, John. How are you?")
response.write ("<br>Did you know I got married last
month?")
End Sub

Sub ResponseToUnknown()
response.write ("Hi. How are you?")
End Sub
%>
```

Để sử dụng một thủ tục, bạn dùng theo kiểu sau:

Whatever()

Với Whatever là tên của thủ tục (nên đặt tên các thủ tục có ý nghĩa mô tả công việc cần làm để script dễ hiểu hơn). Chúng ta cũng cần cung cấp thông tin cho các thủ tục để chúng thực hiện một công việc cụ thể. Cung cấp dữ liệu theo cách sau:

Whatever(data1, data2 dataN)

Trong ví dụ sau sẽ cung cấp các dữ liệu khác nhau cho một thủ tục duy nhất dựa trên tên (Name) của một người nhập vào từ một form:

Ví dụ 3

<%	1
TheName=request.form("name")	2
	3
if TheName="John" then	4
ResponseToVisitor(35,Sue,New York)	5
else	6
if TheName="Peter" then	7
ResponseToVisitor(33,Sally,Los Angeles)	8
else	9
response.write("Who are you?")	10
end if	11
end if	12
	13
Sub ResponseToVisitor(AA,BB,CC)	14
response.write ("I know your are" & AA & "years old, ")	15
response.write ("you are married to" & BB & ", and")	16
response.write ("you are living in " & CC)	17
End Sub	
%>	

Trong dòng 14, nó chỉ định AA là biến nó nhận đầu tiên, BB là biến thứ hai, và CC là biến thứ ba. Giá trị của các biến này được cung cấp theo cùng thứ tự như trong dòng 5 hay dòng 8.

Ví dụ trên cũng cho thấy các thủ tục rất hữu dụng, tránh được việc lặp lại cùng một công việc với nhiều dòng script, do đó script sẽ ngắn gọn và dễ hiểu hơn.

Include/virtual

Server Site include hay SSI là một ngôn ngữ rất đơn giản và cũng rất ít lệnh. Chúng ta chỉ xét một lệnh SSI cho phép chúng ta dùng với các script ASP: include/virtual.

Trong ví dụ kế tiếp, chúng ta sẽ dùng include của SSI trong script asp của chúng ta (*response.asp*). Lệnh này cho

Căn bản thiết kế Web

phép thêm một tập các lệnh từ các file khác (trong ví dụ nên dưới là *file1.html* và *file2.asp*) và thực thi chúng.

Ví dụ 4

response.asp

```
<%  
TheName=request.form("name")  
if TheName="John" then  
  %>  
  <!--#include virtual="/file1.html" -->  
  <% else %>  
  <!--#include virtual="/file2.asp" -->  
  <%  
end if %>
```

File1.html

```
Hi, John.<br>  
I know your are 31 years old, you are married to  
Sue, and you are living in New York.
```

File2.asp

```
<%  
for i=1 to 3  
  response.write(TheName & "...<BR>")  
next  
response.write("Who are you?")  
%>
```

Trong trường hợp này, nếu tên của người sử dụng là John, thì chúng ta sẽ trả lời với *file1.html*. Nếu khác, sẽ thực hiện một số lệnh trong *file2.asp*

File dùng để include phải là một file văn bản (.txt, .html, .htm, .shtml, .asp ...). Mặc dù chúng ta đã sử dụng

Căn bản thiết kế Web

file1.html và *file2.asp*, script vẫn làm việc chính xác với *file1.txt* và *file2.txt* (thay đổi tên file không gây ảnh hưởng gì).

Bằng cách dùng SSI và ASP chúng ta có thể tạo ra các trang web an toàn:

secret_page.asp

```
<%
UserName=request.form ("username")
Password=request.form("password")
if UserName="myusername" and Password="mypassword" then
%>
<!--#include virtual="/cgi-bin/secret_info.txt" -->
<% else %>
<Form Action=secretpage.asp method=post>
Username: <input type=text name=username size=15><BR>
Password: <input type=text name=password size=15><BR>
<input type=Submit Value=Send>
</form>
<% end if %>
```

secret_info.txt

```
This is my secret information:<BR>
My name is John.<BR>
My surname is Smith.<BR>
<BR>End of secret information.
```

Trong trường hợp này thật thuận lợi để lưu file *secret_info.txt* trong thư mục *cgi-bin* (các file .txt trong thư mục này không thể truy xuất được bởi người sử dụng).

Trong phần kế tiếp chúng ta sẽ học cách giữ thông tin từ người sử dụng trong server (phương pháp **Session**) và cách chia sẻ thông tin giữa những người sử dụng (phương pháp **Application**).

Phương pháp Session

Lần đầu tiên người sử dụng truy cập đến trang web của bạn với một kết nối. Trong quá trình xử lý với server này, server và client sẽ trao đổi thông tin cho nhau để định danh mỗi bên. Do sự trao đổi thông tin này, server của chúng ta có thể xác định một người sử dụng cụ thể và các thông tin này có thể được dùng để gán cho từng client cụ thể. Mỗi quan hệ giữa các máy tính này gọi là một session (phiên dao dịch). Trong suốt thời gian một session còn hoạt động, có thể gán các thông tin cho một client cụ thể bằng cách dùng phương pháp **Session**. Chúng ta sẽ dùng một ví dụ để giải thích phương pháp này:

Giả sử chúng ta muốn cho phép một người dùng cụ thể truy xuất đến thông tin trên web site của chúng ta và chúng ta hiển thị tên của người sử dụng (username) trong tất cả các trang mà người này vào. Trong trường hợp này, ta dùng phương pháp Session.

Trong ví dụ này, chúng ta hỏi username của một người trong trang `responsetoform.asp`

<code>responsetoforms.asp</code>	
<code><% IF Request.form="" THEN %></code>	1
	2
<code><html></code>	3
<code><title>Our private pages</title></code>	4
<code><body></code>	5
<code>In order to access this pages fill the form</code>	6
<code>below:
</code>	7
<code><form method="post" action="index.asp"></code>	
<code>Username: <input type="text" name="username"</code>	8
<code>size="20">
</code>	
<code>Password: <input type="password" name="password"</code>	9
<code>size="15">
</code>	10
<code><input type="Submit" value="Submit"></code>	11
<code></form></code>	

</body>	12
</html>	13
<% ELSE %>	14
<%	15
IF Request.form("username")="Joe" AND	16
Request.form("password")="please" THEN	17
%>	18
<%	19
Session("permission")="YES"	20
Session("username")="Joe"	21
%>	22
<html>	23
<title>Our private pages</title>	24
<body>	25
Hi <% =Session("username") %>, you are allow to see	26
these pages: 	27
Page 1 	28
Page 2	29
</body>	30
</html>	31
<% ELSE %>	32
Error in username or password	33
<% END IF %>	34
<% END IF %>	35
	36
	37
	38

Giải thích hoạt động của trang này:

Trong dòng 1, kiểm tra xem có thông tin được gửi từ một form hay không. Nếu không có (Request.form= ""), một form được hiển thị để hỏi username và password.

Sau khi nhập vào form và gửi nó (submit), Request.form sẽ khác "" và script sẽ nhảy đến dòng 15. Trong dòng 17, kiểm tra username và password. Nếu

username là "Joe" và password là "please", thì hai biến này được đặt cho client (dòng 21-22).

```
Session("permission")="YES"  
Session("username")="Joe"
```

Các biến này vẫn được giữ trên server trong suốt thời gian tồn tại session (thông thường nó sẽ mất sau 20 phút nếu không có liên lạc với server).

Cuối cùng, nếu username và password đúng, một trang trả lời chứa các link sẽ được gửi cho client với tên của người sử dụng ở đầu trang. Trong ví dụ này, nếu username hay password sai, trang trả lời sẽ chứa văn bản trong dòng 34.

Giả sử người sử dụng nhấn vào link "Page 1" (pagel.asp). Code của pagel.asp như sau:

pagel.asp	
<% IF Session("permission")="YES" THEN %>	1
	2
<html>	3
<title>Page 1</title>	4
<body>	5
	6
Hi <% =Session("username") %>, welcome to Page 1	7
 	8
This page is empty at the moment, but it will be	9
very interesting in the next future	10
</body>	11
</html>	12
	13
<% ELSE %>	14
You are not allowed to access this page	15
	16
<% end IF %>	17

Trong dòng 1, nó kiểm tra giá trị của Session("permission") có phải là "YES" hay không. Nếu câu trả lời là đúng, một trang có các thông tin sẽ được gửi cho người sử dụng. Nếu không đúng là "YES", dòng văn bản trên dòng 15 sẽ được in ra.

Phương pháp Application

Với phương pháp Session ta có thể định nghĩa Session("whatever")="Joe", nhưng thông tin này không thể chia sẻ giữa nhiều người sử dụng (Session("whatever") là một giá trị duy nhất cho mỗi người sử dụng). Để thông tin được chia sẻ, ta dùng phương pháp Application.

Để hiểu tốt hơn về phương pháp này, chúng ta sẽ tạo một counter (đếm số người vào trang web) hiển thị trên tất cả các trang. Tạo một file *counter.asp* có nội dung như sau:

counter.asp	
<%	1
Application.Lock	2
Application("pagevisits")=Application("pagevisits")+1	3
Application.Unlock	4
%>	5
	6
<html>	7
<title>Page under construction</title>	8
<body>	9
	10
Under construction 	11
Page views: <% =Application("pagevisits") %>	12
	13
</body>	14
</html>	15

Căn bản thiết kế Web

Trong phần code đầu tiên, phương pháp Application chia sẻ thông tin cho các client khác nhau, cần phải tránh trường hợp các client khác thay đổi thông tin trong Application("pagavisits"). Application.Lock sẽ ngừng chia sẻ thông tin, và Application.Unlock sẽ cho phép tiếp tục chia sẻ. Dòng 3 tăng giá trị của counter.

Cuối cùng là một trang html được gửi cho client, có chứa giá trị của counter.

Chú ý:

- Thông tin lưu giữ trong Application("whatever") như trong ví dụ này sẽ mất khi server khởi động lại.

Session và Application đã được sử dụng để tạo một script để tán gẫu (chat).

chat.html

```
<html>
<head><title>Chat</title></head>
<frameset rows="*,100">
<frame src="chatopinions.asp">
<frame src="chatform.asp">
</frameset>
</html>
```

chatopinions.asp

```
<html>
<head>
<META HTTP-EQUIV=refresh
content="5;URL=chatopinions.asp"></head>
<body bgcolor=FFFFFF>
<center>
<table border=0 width=350>
<tr><td width=50>
<% =Application("A9") %></td><td width=300><% =Application("B9")
%>
```

```

</td></tr><tr><td>
<% =Application("A8") %></td><td><% =Application("B8") %>
</td></tr><tr><td>
<% =Application("A7") %></td><td><% =Application("B7") %>
</td></tr><tr><td>
<% =Application("A6") %></td><td><% =Application("B6") %>
</td></tr><tr><td>
<% =Application("A5") %></td><td><% =Application("B5") %>
</td></tr><tr><td width=50>
<% =Application("A4") %></td><td><% =Application("B4") %>
</td></tr><tr><td>
<% =Application("A3") %></td><td><% =Application("B3") %>
</td></tr><tr><td>
<% =Application("A2") %></td><td><% =Application("B2") %>
</td></tr><tr><td>
<% =Application("A1") %></td><td><% =Application("B1") %>
</td></tr></table>
<table border=0 width=350>
<tr><td align=right>
</td></tr></table>
</center>
</body>
</html>

```

chatform.asp

```

<%
If Request.Form ("Opinion")="" then
ShowForm()
else
    if Session("AA")="" then
        NewUser()
        GoAhead()
    else
        GoAhead()
    end if
end if %>

<% Sub ShowForm() %>

<html><body bgcolor=FFFFFF>
<center>
<form method=post action=chatform.asp>
<table border=0><tr><td>
<% if Session("AA")="" then %>
Chose a name or nickname
<input type=text name=Name size=20><BR>
Type your opinion
<% else %>
<% =Session("AA") %>

```



```
<% end if %>
<input type=text size=60 name=Opinion value="<%
=request.form("Opinion") %>"><BR>
<input type=submit value="Submit">
</td><tr></table>
</form>
</center>
</body>
</html>

<% End Sub %>

<% Sub GoAhead() %>

<%
BB=Request.form("Opinion")
BB=server.htmlencode(BB)

Application.Lock
Application("B1")=Application("B2")
Application("B2")=Application("B3")
Application("B3")=Application("B4")
Application("B4")=Application("B5")
Application("B5")=Application("B6")
Application("B6")=Application("B7")
Application("B7")=Application("B8")
Application("B8")=Application("B9")
Application("B9")=BB

Application("A1")=Application("A2")
Application("A2")=Application("A3")
Application("A3")=Application("A4")
Application("A4")=Application("A5")
Application("A5")=Application("A6")
Application("A6")=Application("A7")
Application("A7")=Application("A8")
Application("A8")=Application("A9")
Application("A9")=Session("AA")
Application.Unlock
%>

<html>
<head></head><body bgcolor=FFFFFF>
<center>
<form method=post action=chatform.asp>
<table border=0><tr><td>
<% =Session("AA") %> <input type=text size=60 name=Opinion>
<BR><input type=submit value="Submit">
</td></tr></table>
</form>
</center>
</body>
</html>
```

```

<% End Sub %>

<% Sub NewUser() %>
    <%
    AA=Request.form("Name")
    AA=server.htmlencode(AA)
    Session("AA")=AA
    %>

<% End Sub %>

```

Đối tượng Dictionary

Để hiểu cách làm việc của đối tượng Dictionary, chúng ta tạo một script nhỏ với mục đích dịch các số từ 1 đến 10 từ tiếng Anh sang tiếng Tây Ban Nha.

translate.asp

```

<%
SET
MyDictionary=CreateObject("Scripting.Dictionary")
MyDictionary.Add "one","uno"
MyDictionary.Add "two","dos"
MyDictionary.Add "three","tres"
MyDictionary.Add "four","cuatro"
MyDictionary.Add "five","cinco"
MyDictionary.Add "six","seis"
MyDictionary.Add "seven","siete"
MyDictionary.Add "eight","ocho"
MyDictionary.Add "nine","nueve"
MyDictionary.Add "ten","diez"

EnglishNumber="four"
SpanishNumber=MyDictionary.Item (EnglishNumber)
Response.Write(SpanishNumber)
%>

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

Script làm việc như thế nào:

Trước tiên ta định nghĩa một Dictionary tên "Mydictionary" (dòng 2)

- Chúng ta cho dữ liệu vào dictionary, tương ứng các số trong tiếng Anh và tiếng Tây Ban Nha (Dòng 4 đến dòng 13).

Khi thêm một cặp các số tiếng Anh và Tây Ban Nha vào đối tượng Dictionary, số được viết bằng tiếng Anh gọi là Key (khóa), và số viết bằng tiếng Tây Ban Nha là một Item (mục dữ liệu).

- Trong dòng 15, chúng ta dùng một biến tên `EnglishNumber` và cho nó một giá trị vào biến. ("four")
- Trong dòng 16, chúng ta định nghĩa một biến mới `SpanishNumber` và chúng ta sẽ nhận giá trị của nó từ Dictionary bằng cách chỉ định là ta muốn nhận Item tương ứng với một Key cụ thể (`EnglishNumber`).
- Trong dòng 17, số được dịch ra được gửi cho người sử dụng. Kết quả sẽ là "cuatro".

Bạn có thể thay đổi các giá trị trong dictionary theo các cách sau:

- `MyDictionary.Key ("one")="1"`

Dòng code trên thay thế Key "one" bằng một giá trị Key mới là "1". Mục dữ liệu Item không bị thay đổi.

- `MyDictionary.Item ("two")="2"`

Mục dữ liệu (Item) tương ứng với key "two" sẽ được thay thế bằng một giá trị mới là "2". Key "two" không bị thay đổi.

Chúng ta có thể hiển thị số cặp phần tử có trong Dictionary bằng dòng code sau:

MyDictionary.Count

Nếu bạn muốn kiểm tra có tồn tại một Key trong Dictionary hay không trước khi trả lời cho người sử dụng, bạn dùng so sánh như sau:

```
if MyDictionary.Exists ("ten")=True then
Response.Write("this key is included in the dictionary")
else
Response.Write("Error: no such a key in the dictionary")
end if
```

Ví dụ: Dịch một số từ tiếng Anh sang tiếng Tây Ban Nha

- Nếu không có thông tin nào gửi cho script (dòng 6), script sẽ gửi cho người sử dụng một form để nhập thông tin, dùng thủ tục Sendform() (dòng 34-40).
- Khi có yêu cầu dịch một số, script sẽ kiểm tra có tồn tại key tương ứng trong dictionary hay không (dòng 25). Nếu có, sẽ in ra số bằng tiếng Tây Ban Nha. Nếu không có, thông báo "No translation available" sẽ được gửi cho người sử dụng.

translation.asp	
<html>	1
<title>Page under construction</title>	2
<body>	3
<%	4
if request.form="" then	5
Sendform()	6
else	7
SET MyDictionary=CreateObject("Scripting.Dictionary")	8
	9
MyDictionary.Add "one","uno"	10
MyDictionary.Add "two","dos"	11
MyDictionary.Add "three","tres"	12
MyDictionary.Add "four","cuatro"	13
MyDictionary.Add "five","cinco"	14
MyDictionary.Add "six","seis"	15
	16

Cần bản thiết kế Web

MyDictionary.Add "seven","siete"	17
MyDictionary.Add "eight","ocho"	18
MyDictionary.Add "nine","nueve"	19
MyDictionary.Add "ten","diez"	20
	21
EnglishNumber=request.form("EnglishNumber")	22
Response.Write("English number: " & EnglishNumber)	23
	24
if MyDictionary.Exists (EnglishNumber)=True then	25
SpanishNumber=MyDictionary.Item (EnglishNumber)	26
Response.Write(" Spanish number: " &	
SpanishNumber)	27
else	28
Response.Write(" Spanish number: " & "No	
translation available")	29
end if	30
end if	31
%>	32
	33
<% Sub Sendform() %>	34
<form action=translation.asp method=post>	35
Write a number in English 	36
<input type=text size=30 name=EnglishNumber> 	37
<input type=submit Value="Enter to my Secret Page">	38
</form>	39
<% End Sub %>	40
	41
</body>	42
</html>	43

Ví dụ: Password bảo vệ thông tin

Trong ví dụ này các key và các item được sử dụng là các username và các password.

secretpage.asp
<%
if request.form="" then
Sendform()
else
SET MyDictionary=CreateObject("Scripting.Dictionary")
MyDictionary.Add "John","123"
MyDictionary.Add "Peter","456"
MyDictionary.Add "Anna","789"
Username=request.form("Username")
Password=request.form("password")
if MyDictionary.Exists (Username)=True AND

Căn bản thiết kế Web

```
Password=MyDictionary.Item (Username) then
    SecretInfo()
else
    Response.Write("Error: incorrect username or password")
end if
end if
%>

<% Sub Sendform() %>
<form action=secretpage.asp method=post>
Username: <input type=text size=30 name=Username><BR>
Password: <input type=password size=30 name=Password><BR>
<input type=submit Value="Translate to Spanish">
</form>
<% End Sub %>

<% Sub SecretInfo() %>
<html>
<head>
<title>My Secret Page</title>
</head>
<body bgcolor=FFFFFF>
<center>
<h1>This is my secret info</h1>
Hello !<BR>
Do you want to be my friend?
</center>
</body>
</html>
<% End Sub %>
```

Ví dụ: thay đổi một Key

Trong ví dụ này ta thay đổi key "one" thành "1". Kết quả vẫn là "uno".

translate.asp

```
<%
SET MyDictionary=CreateObject("Scripting.Dictionary")
MyDictionary.Add "one","uno"
MyDictionary.Add "two","dos"
MyDictionary.Add "three","tres"
MyDictionary.Add "four","cuatro"
MyDictionary.Add "five","cinco"
MyDictionary.Add "six","seis"
```

Căn bản thiết kế Web

```
MyDictionary.Add "seven", "siete"
MyDictionary.Add "eight", "ocho"
MyDictionary.Add "nine", "nueve"
MyDictionary.Add "ten", "diez"

MyDictionary.Key ("one")="1"

EnglishNumber="1"
SpanishNumber=MyDictionary.Item (EnglishNumber)
Response.Write(SpanishNumber)
%>
```

Ví dụ: thay đổi một item

Trong ví dụ này ta sẽ thay giá trị “dos” thành “2”. Kết quả sẽ hiển thị là “2”.

Translate.asp

```
<%
SET MyDictionary=CreateObject("Scripting.Dictionary")

MyDictionary.Add "one", "uno"
MyDictionary.Add "two", "dos"
MyDictionary.Add "three", "tres"
MyDictionary.Add "four", "cuatro"
MyDictionary.Add "five", "cinco"
MyDictionary.Add "six", "seis"
MyDictionary.Add "seven", "siete"
MyDictionary.Add "eight", "ocho"
MyDictionary.Add "nine", "nueve"
MyDictionary.Add "ten", "diez"

MyDictionary.Item ("two")="2"

EnglishNumber="two"
SpanishNumber=MyDictionary.Item (EnglishNumber)
Response.Write(SpanishNumber)
%>
```

Ví dụ: Đếm số phần tử trong Dictionary

Giá trị trong trang trả lời là “10”.

translate.asp

```
<%  
SET MyDictionary=CreateObject("Scripting.Dictionary")  
MyDictionary.Add "one","uno"  
MyDictionary.Add "two","dos"  
MyDictionary.Add "three","tres"  
MyDictionary.Add "four","cuatro"  
MyDictionary.Add "five","cinco"  
MyDictionary.Add "six","seis"  
MyDictionary.Add "seven","siete"  
MyDictionary.Add "eight","ocho"  
MyDictionary.Add "nine","nueve"  
MyDictionary.Add "ten","diez"  
Response.Write(MyDictionary.Count)  
%>
```

Cookies

Phương pháp cookies rất giống với phương pháp Session: sự khác nhau cơ bản là Cookies lưu trữ thông tin ở máy tính *client*, không lưu trên *server* như phương pháp Session, do đó nó thích hợp cho các web site có số người truy cập lớn. Phương pháp này gửi thông tin cho client và lấy thông tin này bất cứ lúc nào nó cần. Và chúng ta cũng cần phải biết cách xóa thông tin mà chúng ta đã lưu trữ trên client khi không cần đến nó nữa.

Khi người sử dụng vào trang asp của chúng ta, chúng ta có thể lưu trữ các thông tin liên quan lên máy tính của họ. Thứ tự như sau:

```
<% response.Cookies ("whatever")="information" %>
```

Khi dòng này được thực hiện, người sử dụng sẽ có thông tin trên máy của họ, và bất cứ khi nào ta cần thông tin đó, có thể lấy nó với dòng code sau:

```
<% =request.Cookies ("whatever") %>
```


hay

```
<% variable1=request.Cookies ("whatever") %>
```

Xét một ví dụ dùng Cookies: Giả sử có nhiều người vào web site của chúng ta và chúng ta muốn báo cho họ biết họ đã vào web site của chúng ta bao nhiêu lần.

```
<% If Request.Cookies ("NumberVisits")="" Then %>
    <% Response.Cookies ("NumberVisits")=1 %>
    This is your first visit to this page. Welcome.
<% Else %>
    <% VarNumberVisits=Request.Cookies ("NumberVisits")
    VarNumberVisits=NumberVisits+1
    Response.Cookies ("NumberVisits")=VarNumberVisits %>
    Welcome back to this page. You have visited this
page <% =VarNumberVisits %> times.
    <BR>Check my great links
    <BR>.....
    <BR>.....
<% End If %>
```

Phương pháp Cookies có thể được sử dụng để hiển thị cho người sử dụng các thông tin cụ thể mà chúng ta yêu cầu họ nhập vào từ một form, ví dụ như thông tin cho phép truy xuất nội dung của một trang web hay để tùy biến giao diện (màu nền, màu chữ, có frame hay không...), các thông tin để tự động nhập vào form.

Mở file và đọc nội dung từ một file văn bản.

Ví dụ 1: Đây là đoạn code cơ bản cần để mở một file văn bản:

1	<%
2	Set fs = CreateObject("Scripting.FileSystemObject")
3	
4	Set wfile = fs.OpenTextFile("c:\Mydir\myfile.txt")
5	filecontent = wfile.ReadAll
6	

7	wfile.close
8	Set wfile=nothing
9	Set fs=nothing
10	
11	response.write(filecontent)
12	%>

Dòng 2 tạo một môi trường thích hợp để cho phép thực hiện các thao tác xử lý file trên server. Chúng ta định nghĩa biến "fs" để làm điều đó (có thể lấy tên biến bất kỳ).

Dòng 4, tạo một biến mới "wfile" để áp dụng phương thức OpenTextFile cho biến "fs". Chúng ta cũng định nghĩa vị trí chính xác của file chúng ta muốn mở trong dòng này (cần một đường dẫn đầy đủ).

Trong dòng 5, đọc toàn bộ nội dung của file vào biến "filecontent" dùng lệnh "ReadAll".

Dòng 7 đến 9 dùng để báo server biết đã kết thúc tất cả các thao tác xử lý file.

Dòng 11, trả xuống cho client là nội dung của biến "filecontent".

Ví dụ 2: giả sử chúng ta có một file chứa các thông tin khác nhau trên mỗi dòng (tên ở dòng đầu tiên, họ ở dòng thứ hai, và tuổi ở dòng thứ ba) và chúng ta muốn dùng chúng riêng biệt. Sau đây là script làm điều này:

1	<%
2	Set fs = CreateObject("Scripting.FileSystemObject")
3	
4	Set wfile = fs.OpenTextFile("c:\Mydir\myfile.txt")
5	firstname = wfile.ReadLine
6	lastname = wfile.ReadLine
7	theage = wfile.ReadLine
8	
9	wfile.close

```
10 Set wfile=nothing
11 Set fs=nothing
12
13 %>
14
15 Your first name is <% =firstname %><BR>
16 Your last name is <% =firstname %><BR>
17 Your are <% =firstname %> years old<BR>
```

Ví dụ này cũng đơn giản như ví dụ trước, nhưng trong trường hợp này mỗi dòng đọc từ myfile.txt được lưu trong các biến khác nhau (dòng 5 đến 7), và chúng được sử dụng trong dòng 15 đến 17 để trả lời cho client.

Ví dụ 3: Ví dụ này đọc tất cả các dòng trong file, và trang xuất ra sẽ có số thứ tự và nội dung của mỗi dòng.

```
1 <%
2 Set fs = CreateObject("Scripting.FileSystemObject")
3
4 Set wfile = fs.OpenTextFile("c:\Mydir\myfile.txt")
5
6 counter=0
7 do while not wfile.AtEndOfStream
8     counter=counter+1
9     singleline=wfile.readline
10    response.write (counter & singleline & "<br>")
11 loop
12
13 wfile.close
14 Set wfile=nothing
15 Set fs=nothing
16
17 %>
```

Trong dòng 6, chúng ta định nghĩa biến “counter”, và trong dòng 7 đến 11 chúng ta lặp lại các lệnh bằng vòng lặp Do_While_Loop trong khi chưa hết file (điều kiện là “not wfile.AtEndOfStream”).

Ví dụ 4: Giả sử chúng ta có một file, dòng thứ nhất là một số và dòng thứ hai cũng là một số.

```
1  <%
2  Set fs = CreateObject("Scripting.FileSystemObject")
3
4  Set wfile = fs.OpenTextFile("c:\Mydir\myfile.txt")
5
6  number1 = CInt(wfile.ReadLine)
7  number2= CInt(wfile.ReadLine)
8
9  numberland2 = number1 + number2
10 response.write (numberland2)
11
12 wfile.close
13 Set wfile=nothing
14 Set fs=nothing
15
16 %>
```

Trong ví dụ trước, chúng ta lưu giữ giá trị trong mỗi dòng vào một biến, nhưng biến đó là một chuỗi. Trong ví dụ này, chúng ta lưu giữ nội dung của dòng 1 và dòng 2 vào các biến "number1" và "number2" bằng cách dùng hàm "CInt" (chuyển chuỗi thành số). Hàm này cho phép chúng ta cộng hai số này lại trong dòng 9 và gửi kết quả cho người sử dụng (dòng 10).

Tạo và ghi vào file văn bản

Ví dụ 1: Code để tạo một file cũng tương tự như code ta dùng để mở một file:

```
1  <%
2  thetext="Write this text in the file"
3
4  Set fs = CreateObject("Scripting.FileSystemObject")
5
6  Set wfile = fs.CreateTextFile("c:\Mydir\myfile.txt",
7  True)
8  wfile.Write (thetext)
9  wfile.close
```

```
10 Set wfile=nothing
11 Set fs=nothing
12
13 response.write("Text created")
14 %>
```

Sự khác nhau là ở dòng 6 và 7.

Phương thức dùng trong dòng 6 là "CreateTextFile"; cần phải chỉ rõ đường dẫn đầy đủ của file muốn tạo; ta dùng tham số True để cho phép ghi chồng lên file đã tồn tại. Dùng False nếu không muốn ghi chồng lên file đã có.

Dòng 7 sẽ ghi vào file chuỗi trong biến "thetext".

Chúng ta cũng có thể dùng lệnh sau để ghi vào file:

```
wfile.WriteLine (thetext1)
wfile.WriteLine (thetext2)
...
```

Trong trường hợp này, chúng ta ghi nội dung của biến "thetext1" vào dòng 1, nội dung trong biến "thetext2" vào dòng 2,...

Ví dụ: Giả sử bạn muốn ghi lại tất cả các địa chỉ IP của tất cả người sử dụng trang web của bạn vào một file có tên "mylog.txt".

```
1 <%
2 VisitorsIP=Request.ServerVariables ("REMOTE_ADDR")
3
4 Set fs = CreateObject("Scripting.FileSystemObject")
5
6 Set wfile = fs.OpenTextFile("c:\Mydir\mylog.txt",
7   forappending)
8   wfile.WriteLine (VisitorsIP)
9
10 wfile.close
11 Set wfile=nothing
12 Set fs=nothing
```

13	response.write("IP registered")
14	%>

Địa chỉ IP được lấy trong dòng 2. Trong trường hợp này chúng ta mở một file "mylog.txt" trong dòng 6 với tham số "forappending". Tham số này cho phép ta mở file và ghi địa chỉ IP của người sử dụng gần nhất vào cuối file,

Global.asa

Global.asa là một file văn bản (text file) đặt trong thư mục chính (/global.asa). Dưới đây là một ví dụ về cấu trúc file global.asa cơ bản:

```
global.asa

<SCRIPT LANGUAGE="VBScript" RUNAT="Server">
Sub Application_OnStart
.....
End Sub
Sub Application_OnEnd
.....
End Sub
Sub Session_OnStart
.....
End Sub
Sub Session_OnEnd
.....
End Sub
</SCRIPT>
```

File này được kích hoạt trong các trường hợp:

- Khi người sử dụng đầu tiên vào web site
- Khi một session (phiên dao dịch) mới bắt đầu.

Căn bản thiết kế Web

Trong cả hai trường hợp, chúng ta có thể xác định một loạt các sự kiện để thực hiện trong file trên:

1. `Application_OnStart`

Nó được thực hiện trước khi session đầu tiên bắt đầu.

2. `Application_OnEnd`

Nó được thực hiện khi ứng dụng kết thúc.

3. `Session_OnStart`

Thực hiện khi server tạo một session mới (khi có một client truy xuất vào server).

4. `Session_OnEnd`

Thực hiện khi một session bị hủy sau một khoảng thời gian nào đó mà không có liên lạc giữa client và server (thông thường là 20 phút).

Xét một ví dụ rất đơn giản:

Đếm số người đang sử dụng trang web:

Tạo file `global.asa` và lưu nó vào thư mục chính của web site, có nội dung như sau:

```
global.asa
```

```
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">
```

```
Sub Application_OnStart
```

```
application("activevisitors")=0
```

```
End Sub
```

```
Sub Application_OnEnd
```

```
End Sub
```

```
Sub Session_OnStart
```

```
application.lock
```

```
application("activevisitors")=application("activevisitors")+1
```

```
application.unlock
```

```
End Sub
```

```
Sub Session_OnEnd  
application.lock  
application("activevisitors")=application("activevisitors")-1  
application.unlock  
End Sub  
</SCRIPT>
```

Lần đầu tiên một người sử dụng vào trang web, global.asa sẽ được thực hiện, và vì vậy, application("activevisitors") trong dòng 4 nhận giá trị "0". Ngay sau đó (như là bắt đầu một session mới), trong dòng 12, application("activevisitors") sẽ tăng lên 1. Mỗi lần một người sử dụng mới vào trang web sẽ làm application("activevisitors") tăng lên 1, và tương tự, mỗi khi một session kết thúc, application("activevisitors") sẽ giảm đi 1 (dòng 18).

Để hiển thị số người đang vào trang web, dùng code như sau:

```
index.asp
```

```
There are <% =application("activevisitors") %> active  
visitors.
```

Các hàm và các thủ tục

abs (n: số)

Hàm. Trả về giá trị tuyệt đối của n.

chr (asciicharcode: số)

Trả về chuỗi có chứa ký tự ASCII được chỉ định bởi *asciicharcode*.

date

Trả về chuỗi chứa ngày tháng hiện hành của hệ thống.

day (*thedata*: chuỗi)

Trả về một số chứa ngày lấy ra từ *thedata*.

hour (*thetime*: chuỗi)

Lấy giá trị giờ từ *thetime* và trả về là một số.

left (*thestring*: chuỗi, *n*: số)

Trả về chuỗi có chứa *n* ký tự đầu tiên của *thestring*.

len (*thestring*: chuỗi)

Trả về số ký tự có trong *thestring*.

minute (*thetime*: chuỗi)

Trả về một số là số phút lấy ra từ *thetime*.

month (*thedata*: chuỗi)

Trả về một số là tháng lấy ra từ *thedata*.

monthname (*themonth*: số)

Trả về một chuỗi chứa tên của một tháng được chỉ định bởi *themonth*.

now

Trả về ngày và thời gian hiện hành của hệ thống.

Request.Form (*fieldname*: chuỗi)

Trả về nội dung của field có tên *fieldname* trong một form.

ServerVariables

Request.ServerVariables ("HTTP_User-Agent")

Trả về chuỗi chứa loại trình duyệt.

Request.ServerVariables ("REMOTE_ADDR")

Trả về chuỗi chứa địa chỉ IP của client.

Request.ServerVariables ("REMOTE_HOST")

Trả về chuỗi chứa tên domain của client.

Căn bản thiết kế Web

`Request.ServerVariables ("SERVER_NAME")`

Trả về chuỗi chứa tên domain của server. Nếu server không có domain, nó trả về chuỗi chứa địa chỉ IP của server.

Response.redirect (*anotherURL*: chuỗi)

Chuyển hướng trình duyệt đến một địa chỉ được xác định bởi *anotherURL*.

right (*thestring*: chuỗi , *n*: số)

Trả về chuỗi chứa *n* ký tự cuối cùng của *thestring*.

second (*thetime*: chuỗi)

Trả về một số chứa số giây lấy ra từ *thetime*.

Server.URLEncode (*thestring*: chuỗi)

Chuyển *thestring* từ một chuỗi sang một URL bằng cách chuyển đổi một vài ký tự nào đó. Ví dụ, các ký tự khoảng trống được thay thế bằng %20. Trả về một URL.

time

Trả về chuỗi chứa thời gian hiện hành của hệ thống.

timevalue (*thetime*: chuỗi)

Trả về chuỗi chứa thời gian lấy ra từ *thetime*.

weekday (*thedate*: chuỗi)

Trả về một số chứa trị ngày trong tuần lấy ra từ *thedate*.

weekdayname (*weekday*: số)

Trả về một chuỗi chứa tên của ngày trong tuần được chỉ định bởi *weekday*.

year (*thedate*: string)

Trả về một số chứa giá trị năm lấy ra từ *thedate*.

Các toán tử

Các toán tử sau được hỗ trợ trong ASP:

Addition (+)	Cộng
And	Và
Assignment (=)	Gán
Concatenation (&)	Ghép
Division (/)	Chia
Exponentiation (^)	Mũ
Is	Kiểm tra
Equals (=)	Bằng
Greater Than (>)	Lớn hơn
Greater Than or Equal to (>=)	Lớn hơn hoặc bằng
Less Than (<)	Nhỏ hơn
Less Than or Equal to (<=)	Nhỏ hơn hoặc bằng
Mod	Chia lấy phần dư
Multiplication (*)	Nhân
Negation (-)	Phủ định giá trị
Not	Phủ định
Not Equal To (<>)	Không bằng
Or	Hoặc
Subtraction (-)	Trừ
Xor	Xor

CĂN BẢN THIẾT KẾ WEB

Chịu trách nhiệm xuất bản : CÁT VĂN THÀNH

Biên tập : NGUYỄN HẠNH

Sửa bản in : HOÀNG TRIẾT

Thực hiện liên doanh : NT BOOKS

Email : hohung@hcm.vnn.vn

ĐT : 08.9950144 - 0903932938

In 1.000 cuốn, khổ 14 x 20 cm. Tại XN In Gia Định, số 9D Nơ Trang Long, Q. Bình Thạnh, TP. Hồ Chí Minh - ĐT: 8412644. Số đăng ký kế hoạch xuất bản: 206-104/XB-QLXB. Nhà Xuất bản Thống Kê cấp ngày 1.02.2002. In xong và nộp lưu chiểu tháng 3 năm 2002.

huy 3.0