LỜI NÓI ĐẦU

Trong những năm gần đây, cùng với sự phát triển của Công nghệ thông tin, mạng máy tính đang được phát triển rộng rãi, một trong những ứng dụng phổ biến trên mạng máy tính là Internet và các dịch vụ trở nên không thể thiếu trong cuộc sống hiện đại. Để phục vụ việc học tập nghiên cứu của sinh viên cao đẳng CNTT, chúng tôi biên soạn giáo trình ***Thiết kế và lập trình Web***. Giáo trình bao gồm 5 chương, mỗi chương đều có phần kiến thức lý thuyết và câu hỏi nghiên cứu cụ thể.

Chương 1: Giới thiệu chung

Giới thiệu các khái niệm cơ bản về mạng máy tính, Internet, địa chỉ IP, giao thức truyền thông và các khái niệm khác.

Chương 2: Thiết kế Web với ngôn ngữ đánh dấu siêu văn bản (HTML)

Trình bày khái niệm ngôn ngữ đánh dấu siêu văn bản, các thành phần cơ bản và cấu trúc của một tập tin HTML. Giới thiệu các vấn đề liên quan đến việc sử dụng cú pháp và hiệu ứng của ngôn ngữ đánh dấu siêu văn bản, để lập trình Web.

Chương 3: Ngôn ngữ kịch bản trong lập trình Web

Giới thiệu hai ngôn ngữ kịch bản phổ biến hiện nay là VBScript và JavaScript. Hướng dẫn các bước tiến hành khai báo, lập trình và sử dụng ngôn ngữ kịch bản trong HTML.

Chương 4: Lập trình Web động với công nghệ ASP

Giới thiệu lập trình Web động với công nghệ ASP (Active Server Page). Các khái niệm cơ bản, các đối tượng cơ bản trong ASP, ứng dụng vào lập trình một trang Web động cụ thể.

Chương 5: Kết nối cơ sở dữ liệu trong lập trình Web động với ASP

Giới thiệu ADO (ActiveX Data Object), các đối tượng của ADO, cách thức kết nối với cơ sở dữ liệu, xử lý lỗi trong khi lập trình các ứng dụng. Hướng dẫn sử dụng các lệnh SQL trong lập trình bằng ASP. Ứng dụng tổng hợp toàn bộ kiến thức để xây dựng một trang Web động hoàn chỉnh.

Nội dung trọng tâm được trình bày trong chương hai, chương ba và chương bốn, cuối mỗi chương này đều có bài tập hướng dẫn lập trình. Giáo trình ***Thiết kế và lập trình Web*** hướng dẫn cách xây dựng một ứng dụng Web từ cơ bản đến nâng cao bằng công nghệ HTML và ASP. Được biên soạn với phương châm đảm bảo tính logic, khoa học, thiết thực, dễ hiểu nhằm trang bị cho sinh viên những kiến thức cơ bản, phục vụ cho nghiên cứu, thiết kế, lập trình một ứng dụng Web hoàn chỉnh.

Tài liệu này được tham khảo từ một số tài liệu của các tác giả trong nước, tổng hợp và lược dịch từ một số tài liệu chuyên ngành của nước ngoài, nên một số thuật ngữ Tin học không thể thay thế bằng tiếng Việt. Để tiện cho việc trình bày ý tưởng xuyên suốt của tài liệu, chúng tôi để nguyên bản thuật ngữ tiếng Anh và giải thích bằng thuật ngữ tiếng Việt nếu có cụm từ tương đương.

Tuy có nhiều cố gắng trong công tác biên soạn, nhưng vẫn không tránh khỏi thiếu sót. Trong phạm vi hạn hẹp của cuốn tài liệu này, không thể đề cập được tất cả những vấn đề nóng hổi trong lĩnh vực thiết kế và lập trình Web đòi hỏi.

Rất mong sự đóng góp phê bình từ bạn đọc, để tài liệu được hoàn chỉnh hơn.

Mọi ý kiến góp ý, xin gửi về Trường Sĩ quan CH-KT Thông tin.

TÁC GIẢ

Chương 1

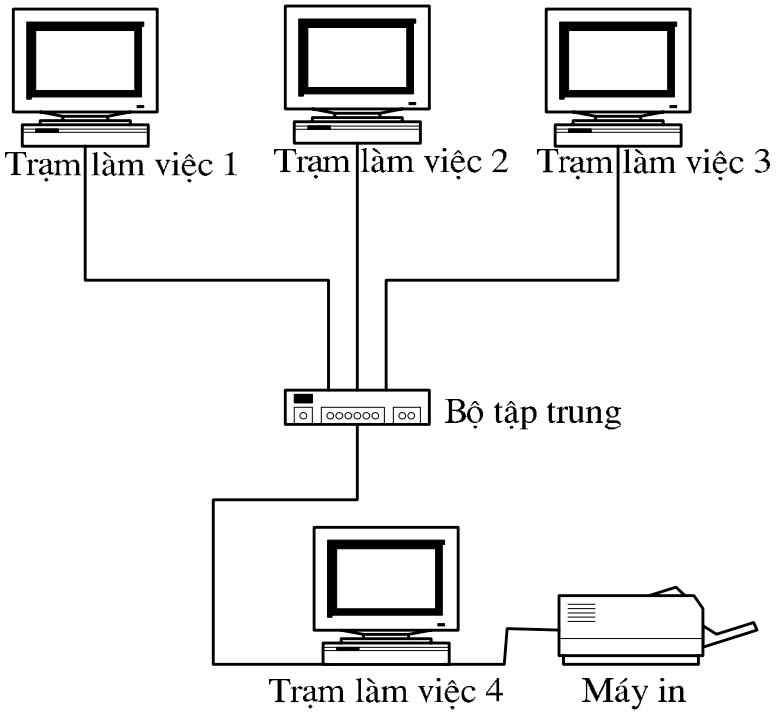
GIỚI THIỆU CHUNG

1.1 Mạng máy tính

1.1.1 *Định nghĩa*

Trong quá trình khai thác, sử dụng máy tính cá nhân (Personal Computer-PC), việc trao đổi, sử dụng thông tin của một xã hội phát triển có nhu cầu to lớn. Khi các máy vi tính chưa có sự liên kết với nhau, thì việc trao đổi thông tin mất rất nhiều thời gian để sao chép, gây nhiều phiền phức. Để giải quyết vấn đề trên với đà phát triển của nền công nghiệp máy tính, các thiết bị đặc biệt và mạng máy tính ra đời là một tất yếu.

Vì vậy, mạng (network) là một tập hợp các hệ thống máy tính và các thiết bị mạng, chia sẻ dữ liệu, chương trình, tài nguyên thông qua một đường truyền kết nối truyền thông dùng chung, trên cơ sở một hệ điều hành mạng.



Hình 1.1. Một mạng máy tính điển hình

Đường truyền là một hệ thống các thiết bị truyền dẫn vật lý để chuyển tải các tín hiệu sóng điện từ.

Đường truyền vật lý có thể phân làm 2 loại:

- Hữu tuyến: cáp đồng trục, cáp đôi dây xoắn, cáp quang, cáp điện thoại, và công nghệ mới nhất hiện nay là cáp điện năng thông thường.
- Vô tuyến: sóng cực ngắn (viba), tia hồng ngoại...

1.1.2 *Phân loại*

Hiện nay, thông thường mạng máy tính được phân loại như sau:

a. Mạng cục bộ - LAN (Local Area Network)

Các máy tính cá nhân và các máy tính khác trong phạm vi một khu vực hạn chế được nối với nhau bằng các dây cáp chất lượng tốt, sao cho những người sử

dụng có thể trao đổi thông tin, dùng chung các thiết bị ngoại vi, và sử dụng các chương trình cũng như các dữ liệu đã được lưu trữ trong một máy tính dành riêng gọi là máy dịch vụ tệp.

b. Mạng diện rộng - WAN (Wide Area Network)

Các mạng lớn hơn, gọi là mạng diện rộng (Wide Area Network), dùng các đường dây điện thoại hoặc các phương tiện liên lạc khác để liên kết lạc khác để liên kết các máy tính với nhau trong phạm vi từ vài chục đến vài ngàn dặm.

Sự khác nhau giữa LAN và WAN: khác nhiều về quy mô và mức độ phức tạp, mạng cục bộ có thể chỉ liên kết vài ba máy tính cá nhân và một thiết bị ngoại vi dùng chung đắt tiền, như máy in laser chẳng hạn. Các hệ thống phức tạp hơn thì có các máy tính trung tâm (máy dịch vụ tệp) và cho phép những người dùng tiến hành thông tin với nhau thông qua thư điện tử để phân phối các chương trình nhiều người sử dụng, và để thâm nhập vào các cơ sở dữ liệu dùng chung.

c. Mạng đô thị - MAN (Metropolitan Area Network)

Là một mạng trải dài trên một không gian địa lý lớn hơn LAN nhưng nhỏ hơn WAN. MAN thường được sử dụng như một mạng của một thành phố, một khu công nghiệp.

d. Mạng Intranet

Là một mạng sử dụng nội bộ như LAN hay WAN thực hiện được các ứng dụng, nói cách khác là các dịch vụ của INTERNET, chủ yếu là dịch vụ WEB với giao thức truyền tệp siêu văn bản - HTTP.

e. Mạng Internet

Một hệ thống gồm các mạng máy tính được liên kết với nhau trên phạm vi toàn thế giới, tạo điều kiện thuận lợi cho các dịch vụ truyền thông dữ liệu, như đăng nhập từ xa, truyền các tệp tin, thư tín điện tử, và các nhóm thông tin. Internet là một phương pháp ghép nối các mạng máy tính hiện hành, phát triển một cách rộng rãi tầm hoạt động của từng hệ thống thành viên

1.2 Internet

Mạng Internet ngày nay là một mạng toàn cầu, bao gồm hàng chục triệu người sử dụng, được hình thành từ cuối thập kỷ 60 từ một thí nghiệm của Bộ quốc phòng Mỹ. Tại thời điểm ban đầu đó là mạng ARPAnet của Ban quản lý dự án nghiên cứu Quốc phòng. ARPAnet là một mạng thử nghiệm phục vụ các nghiên cứu quốc phòng, một trong những mục đích của nó là xây dựng một mạng máy tính có khả năng chịu đựng các sự cố (ví dụ một số nút mạng bị tấn công và phá huỷ nh-ng mạng vẫn tiếp tục hoạt động).

Mạng cho phép một máy tính bất kỳ trên mạng liên lạc với mọi máy tính khác. Khả năng kết nối các hệ thống máy tính khác nhau đã hấp dẫn mọi người, vả lại đây cũng là phương pháp thực tế duy nhất để kết nối các máy tính của các hãng khác nhau.

Mạng Internet nguyên thuỷ được thiết kế nhằm mục đích phục vụ việc cung cấp thông tin cho giới khoa học, nên công nghệ của nó cho phép mọi hệ thống đều có thể liên kết với nó thông qua một cổng điện tử. Theo cách đó, có hàng ngàn hệ máy tính hợp tác, cũng như nhiều hệ thống dịch vụ thư điện tử có thu phí, như MCI và Compuserve chẳng hạn, đã trở nên thành viên của Internet. Với hơn hai triệu

máy chủ phục vụ chừng 20 triệu người dùng, mạng Internet đang phát triển với tốc độ bùng nổ, mỗi tháng có thêm khoảng một triệu người tham gia mới.

Ngày nay Internet cho phép hàng trăm triệu người trên khắp thế giới liên lạc và trao đổi thông tin với nhau thông qua tập các giao thức gọi chung là bộ giao thức TCP/IP (Transmission Control Protocol/Internet Protocol).

1.3 Các giao thức Internet

Ban đầu, bộ giao thức Internet (còn gọi là bộ giao thức TCP/IP) được phát triển bởi DoD (bộ quốc phòng Mỹ) và được đưa vào triển khai từ năm 1982 để cung cấp dịch vụ tăng cường tín hiệu trên các liên mạng lớn, kết hợp nhiều kiểu máy tính khác nhau. TCP/IP cho phép các loại máy tính với các kích cỡ khác nhau liên kết với Internet để giao tiếp với nhau. Hỗ trợ trên phần lớn các hệ thống, TCP/IP trở thành giao thức chuẩn của Internet. Phần TCP của giao thức này đảm bảo rằng rất cả lượng thông tin gửi đi đều được nhận đầy đủ và chính xác. Phần IP cung cấp kỹ thuật truyền dẫn các gói thông tin tới địa chỉ nhận một cách có hiệu quả. Trong những năm gần đây, các giao thức Internet ngày càng phổ biến và hình thành các giao thức mạng phổ dụng nhất hiện nay.

Có nhiều giao thức kết hợp với bộ giao thức Internet. Dưới đây là các mô tả một số giao thức này.

1.3.1 *Giao thức điều khiển phiên truyền*

Giao thức điều khiển phiên truyền (Transmission Control Protocol-TCP) là một giao thức Internet tương ứng với tầng giao vận của OSI. TCP cung cấp khả năng chuyển tải hướng kết nối, song công đầy đủ (full dupplex). Khi không cần phần điều hành chung của một tiến trình chuyển tải hướng kết nối thì giao thức gam dữ liệu người dùng (User Datagram Protocol-UDP) có thể được thay thế cho TCP ở cấp chuyển tải (giữa các máy chủ). TCP và UDP hoạt động tại cùng một tầng. TCP tương ứng với SPX trong môi trường Netware. TCP duy trì một tuyến kết nối logic giữa các máy tính gửi và nhận. Theo cách này, tính nguyên vẹn của phiên truyền được duy trì, TCP nhanh chóng phát hiện mọi sự cố trong phiên truyền để chỉnh lý, nhưng ngược lại, TCP không chạy nhanh bằng UDP.

TCP còn cung cấp tính năng phân chia và tập hợp các thông điệp, đồng thời có thể chấp nhận các thông điệp có kích thước bất kỳ từ các giao thức tầng phía trên. TCP phân chia các luồng thông điệp thành các phân đoạn mà IP có thể điều khiển và quản lý. Khi sử dụng kết hợp với IP, TCP bổ sung dịch vụ hướng kết nối và tiến hành đồng bộ hoá phân đoạn, bổ sung các số chuỗi tại mức byte.

Ngoài phân chia thông điệp, TCP còn có thể duy trì nhiều cuộc đối thoại (conversations) với các giao thức tầng phía trên và có thể cải thiện việc sử dụng băng thông mạng bằng cách tổ hợp nhiều thông điệp vào chung một phân đoạn. Mỗi tuyến kết nối mạch ảo được gán một ID kết nối có tên là *cổng* (port) để định danh các gam dữ liệu kết hợp với các tuyến kết nối đó.

1.3.2 *Giao thức Internet*

Giao thức Internet (Internet Protocol-IP) là một giao thức phi kết nối (connectionless) cung cấp dịch vụ gam dữ liệu và các gói tin IP thường được gọi là gam dữ liệu IP (IP datagram). IP là một giao thức chuyển gói tin thực hiện tiến trình định địa chỉ và chọn đường. Một phần đầu IP được nối vào các gói tin, được các giao thức cấp thấp hơn truyền theo dạng các khung (frame).

IP định đường các gói tin thông qua các liên mạng bằng cách vận dụng các bảng định tuyến động (dynamic routing table) được tham chiếu tại mỗi bước nhảy. Các phần xác định tuyến đường được tiến hành bằng cách tham khảo thông tin thiết bị mạng vật lý và logic, mà giao thức phân giải địa chỉ (Address Resolution Protocol-ARP) cung cấp.

IP thực hiện tách rời và lắp ghép lại các gói tin theo yêu cầu giới hạn kích thước các gói tin, được định nghĩa cho các tầng vật lý và liên kết dữ liệu thực thi. IP cũng thực hiện tính năng kiểm tra lỗi trên dữ liệu phần đầu bằng cách tổng kiểm tra (checksum), mặc dù dữ liệu của các tầng phía trên không được kiểm tra lỗi.

1.3.3 *Giao thức gam dữ liệu người dùng*

Giao thức gam dữ liệu người dùng (User Datagram Protocol-UDP) là một giao thức tầng giao vận phi kết nối (giữa các máy chủ). UDP không cung cấp các tín hiệu báo nhận thông điệp, thay vào đó, đơn giản là nó chỉ làm công việc chuyển tải các gam dữ liệu.

Cũng như TCP, UDP vận dụng các địa chỉ cổng để bàn giao các gam dữ liệu. Tuy nhiên, các địa chỉ cổng này không kết hợp với các mạch ảo mà chỉ đơn thuần là định danh các tiến trình xử lý của máy chủ cục bộ.

UDP được sử dụng nhiều hơn TCP khi khả năng bàn giao đáng tin cậy không quan trọng bằng khả năng thực hiện cao hoặc phải giữ phần điều hành chung của mạng ở mức thấp. Do UDP không cần thiết lập, bảo trì và kết thúc các kết nối hoặc điều khiển luồng dữ liệu nên nói chung nó chạy nhanh hơn TCP.

UDP là giao thức tầng giao vận, được sử dụng với giao thức quản trị mạng đơn giản (Simple Network Management Protocol-SNMP), là giao thức quản trị mạng chuẩn, được dùng với các mạng TCP/IP. UDP cho phép SNMP cung cấp tính năng quản trị mạng với phần điều hành chung ở mức tối thiểu.

1.3.4 *Giao thức phân giải địa chỉ*

Ba kiểu thông tin địa chỉ dưới đây được sử dụng trên các mạng TCP/IP:

- Địa chỉ vật lý: được tầng vật lý và tầng liên kết dữ liệu sử dụng.
- Các địa chỉ IP: Cung cấp các ID máy chủ và mạng logic. Các địa chỉ IP bao gồm bốn con số được biểu diễn dưới dạng thập phân có chấm. Ví dụ, 192.123.1.1 là một địa chỉ IP.
- Các tên nút logic: Định danh các máy chủ cụ thể bằng các ID ký tự-số. Chúng giúp ta dễ nhớ hơn so với các địa chỉ ID toàn số. Ví dụ, tsqtt.edu.vn là một tên nút logic (logical node name).

Căn cứ vào tên nút (node) logic mà giao thức phân giải địa chỉ (Address Resolution Protocol-ARP) có thể xác định địa chỉ IP kết hợp với tên đó. ARP duy trì các bảng dữ liệu phân giải địa chỉ và có thể quảng bá các gói tin để phát hiện các địa chỉ trên liên mạng. Các địa chỉ IP do ARP phát hiện có thể được cung cấp cho các giao thức tầng liên kết dữ liệu.

1.3.5 *Giao thức hệ thống tên miền*

Giao thức hệ thống tên miền (Domain Name System-DNS) cung cấp tính năng phân giải tên/địa chỉ như một dịch vụ cho các ứng dụng trên máy khách (client). Các hệ phục vụ DNS cho phép con người dùng tên các nút logic để truy cập các tài nguyên trên mạng.

1.3.6 *Giao thức chuyển thư đơn giản*

Giao thức chuyển thư đơn giản (Simple Mail Transfer Protocol-SMTP) và giao thức thư tín phiên bản 3 (Post Office Protocol version 3-POP3) là một giao thức để định đường thư tín thông qua các mạng. Nó sử dụng giao thức TCP/IP.

SMTP không cung cấp một hệ giao tiếp thư cho người dùng. Quy chuẩn, quản lý và trao đổi các thông điệp cho người dùng cuối (End user), tất cả đều phải tiến hành bởi một phần mềm trao đổi thư tín điện tử (như Outlook, Eudora,...).

1.3.7 *Giao thức truyền tập tin*

Giao thức truyền tập tin (File Transfer Protocol-FTP) là một giao thức để dùng chung các tập tin giữa các máy chủ nối mạng. FTP cho phép người dùng đăng nhập các máy chủ ở xa. Những người dùng đã đăng nhập có thể xem xét các thư mục, thao tác với các tập tin, thực thi các lệnh và chạy các chương trình trên máy chủ. FTP cũng có khả năng trao đổi các tập tin giữa các máy chủ không đồng bộ bằng cách hỗ trợ một cấu trúc yêu cầu tập tin độc lập với các hệ điều hành cụ thể.

1.3.8 *HTTP - HyperText Transfer Protocol*

Cách thức để trình duyệt WEB của người dùng nói chuyện với chương trình Web server khi người dùng sử dung WWW. *Hypertext*: cách thức liên kết tham chiếu đến những mẫu thông tin khác nhau.

1.4 Địa chỉ IP

Địa chỉ IP giúp chúng ta có thể nhận diện được các máy mà không cần quan tâm đến công nghệ mạng cơ sở. Ưu điểm của nó là có thể đơn giản hóa việc định đường đi trên mạng. Ngoài ra, địa chỉ IP còn mang tính toàn cục, nếu mạng cục bộ nào đó được nối vào Internet, thì toàn bộ các máy trong mạng đó sẽ được toàn Internet biết đến thông qua địa chỉ IP.

Địa chỉ IP đang được sử dụng hiện tại (IPv4) có 32 bit chia thành 4 octet (mỗi octet có 8 bit, tương đương 1 byte), cách đếm đều từ trái qua phải bit 0 cho đến bit 31, các octet tách biệt nhau bởi dấu chấm (.). Mỗi octet có thể biểu diễn bằng các số thập phân, nhị phân hoặc thập lục phân. Đia chỉ IP bao gồm có 3 thành phần chính :

- Bit nhận dạng lớp (Class bit)
- Định danh của mạng (Network Identifier – NET ID)
- Định danh máy chủ (Host Identifier – HOST ID)

Do tổ chức và độ lớn của các mạng cục bộ trong liên mạng khác nhau, để thuận tiện cho việc quản lý cấp phát địa chỉ IP người ta chia địa chỉ mạng thành 5 lớp. Ký hiệu là A, B, C, D, E

<table>
<tr><td>0</td><td colspan="4">NET ID (7 bits)</td><td colspan="4">HOST ID (24 bits)</td></tr>
<tr><td>1</td><td>0</td><td colspan="6">NET ID (14 bits)</td><td colspan="4">HOST ID (16 bits)</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td colspan="8">NET ID (21 bits)</td><td colspan="2">HOST ID(8 bits)</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td colspan="10">Multicast (18 bits)</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td colspan="9">Multicast (17 bits)</td></tr>
</table>

Hình 1.2. Cấu trúc các lớp địa chỉ lớp IP

- Lớp A: Sử dụng 7 bit định danh mạng và 24 bit để định danh các trạm. Lớp A cho phép sử dụng 27-2 mạng và 224 -2 trạm trong mỗi mạng. Lớp này thích hợp cho các mạng có số trạm cực lớn. Tổng số khoảng hơn 2 tỉ địa chỉ. Vùng địa chỉ lớp A có thể sử dụng gồm từ 1.0.0.1 đến 126.255.255.254.
- Lớp B: Sử dụng 14 bit định danh mạng và 16 bit để định danh các trạm. Lớp B cho phép sử dụng 214-2 mạng và 216 -2 trạm trong mỗi mạng. Tổng số khoảng hơn 1 tỉ địa chỉ. Vùng địa chỉ lớp B có thể sử dụng từ 128.0.0.1 đến 191.254.255.254.
- Lớp C: Sử dụng 21 bit định danh mạng và 8 bit để định danh các trạm. Lớp C cho phép sử dụng 221-2 mạng và 28 -2 trạm trong mỗi mạng. Lớp này thích hợp cho các mạng nhỏ, có số trạm trong mỗi mạng không quá 254. Tổng số khoảng hơn nửa tỉ địa chỉ. Vùng địa chỉ lớp B có thể sử dụng từ 192.0.0.1 đến 223.255.254.254.
- Lớp D: địa chỉ lớp này được sử dụng cho việc quảng bá (dùng để gửi IP datagram tới một nhóm trên mạng sử dụng cùng kiểu địa chỉ).
- Lớp E: địa chỉ dự phòng trong tương lai.

1.5 Các khái niệm khác

1.5.1 URL

URL (Uniform Resource Locator) là cách gọi khác của địa chỉ web. URL bao gồm tên của giao thức (thường là HTTP hoặc FTP), tiếp đến là dấu hai chấm (:), hai dấu gạch chéo (//), sau đó là tên miền muốn kết nối đến. Ví dụ về một URL là "http://www.vnn.vn/cntt" sẽ hướng dẫn trình duyệt web của chúng ta sử dụng giao thức giao thức http để kết nối đến máy tính www.vnn.vn, mở tệp web ngầm định có tên là default.htm (hay index.htm) trong thư mục cntt. Tên tệp tin ngầm định không cần gõ vào URL. Khi gõ URL cũng có thể bỏ qua tên giao thức http vì trình duyệt lấy giao thức http làm giao thức ngầm định.

URL có một cú pháp đặc biệt. Tất cả các URL phải chính xác, thậm chí có một ký tự sai hay thiếu một dấu chấm cũng không được Web Server chấp nhận, nhập sai một ký tự trong địa chỉ URL có thể dẫn chúng ta đến một Web site có nội dung khác hoặc nhận được thông báo Web site đó không tồn tại.

1.5.2 Hyperlink (siêu liên kết)

Hyperlink (siêu liên kết) là một thành phần cơ bản và rất cần thiết đối với một siêu văn bản World Wide Web. Siêu liên kết giúp chúng ta dễ dàng tìm kiếm các thông tin khác nhau về một chủ đề. Một siêu liên kết là một phần văn bản (hay hình ảnh) của trang Web, mà khi kích vào đó sẽ tự động thực hiện một trong các thao tác sau đây:

- Đưa đến phần khác của trang
- Đưa đến một trang web khác trong cùng một Web site
- Đưa đến một trang web khác trong Web site khác
- Cho phép download một file
- Chạy một ứng dụng, trình diễn một đoạn video hoặc âm thanh

Hình ảnh minh hoạ dưới đây là một phần của trang web. Những từ gạch dưới thể hiện các liên kết, chỉ cần nhấn chuột vào siêu liên kết, nội dung tài liệu mà nó trỏ tới sẽ được hiển thị.

Hình 1.3. Minh họa một Hyperlink (siêu liên kết)

1.5.3 *Web Browser (trình duyệt web)*

Web Browser là một công cụ hay chương trình cho phép truy xuất và xem thông tin trên Web. Có nhiều Web Browser để truy xuất Web, mỗi trình duyệt có những đặc điểm khác nhau và chúng hiển thị những trang Web không hoàn toàn giống nhau.

Các trình duyệt web bao gồm có Internet Explorer, Netscape Navigator Communicator, Opera, Mozilla Firefox, ... Tất cả các loại trình duyệt này đều có các phiên bản khác nhau, và các phiên bản mới nhất sẽ có nhiều tính năng hơn các phiên bản trước đó. Ngoài việc truy xuất Web, các trình duyệt còn cho phép chúng ta thực hiện các công việc khác như: gửi nhận email, tải các tập tin từ Web Server về, … thông qua các Add-on và Plugin của trình duyệt.

1.5.4 *Web Server (máy chủ Web)*

Web Server là máy chủ trong đó chứa thông tin dưới dạng trang Web (trang HTML có thể chứa âm thanh, hình ảnh, video, văn bản, …). Các Web Server được kết nối với nhau thông qua mạng Internet, mỗi Server có địa chỉ duy nhất trên Internet.

Thành phần chủ chốt của Web Server là phần mềm. Mỗi phần mềm Web Server chạy trên một nền tảng phần cứng và một hệ điều hành cụ thể. Một Web Server phải có cấu hình đủ mạnh để cung cấp các dịch vụ cho các client, đáp ứng đồng thời nhiều yêu cầu từ client và có khả năng lưu trữ lớn cho tài nguyên Web.

Nói về chức năng và hiệu năng, các Web Server phân thành 4 nhóm chính:

- Các máy chủ truyền thông thông thường.
- Máy chủ thương mại.
- Máy chủ mhóm làm việc.
- Máy chủ dùng cho mục đích đặc biệt.

Các tiêu chuẩn đánh giá một Web Server:

- Hiệu năng: nền tảng hệ điều hành và xử lý đa luồng.
- Bảo mật: Thông qua địa chỉ IP, tên máy chủ của mạng con, thư mục ... Web Oracle cung cấp phương án bảo mật thông tin theo tên người sử dụng và khoá mã được mã hoá hoàn toàn trong quá trình truyền thông trên mạng.
- Truy nhập và tích hợp CSDL: Hầu hết các Web Server đều sử dụng giao diện CGI, một số khác thì dùng giao diện lập trình ứng dụng (API) hoặc ngôn ngữ hỏi đáp có cấu trúc SQL.
- Quản lý và quản trị Web Server: Đặc tính quan trọng của tiêu chuẩn này là khả năng quản trị từ xa, giao diện đồ họa và điều khiển cấu hình của máy chủ.

1.5.5 *Web Site*

Web Site là một tập hợp các trang Web liên quan đến một công ty, một tập đoàn, một tổ chức, một cá nhân hay đơn giản chỉ là một chủ đề mà nhiều người cùng quan tâm. Ví dụ Web Site của Chính phủ (www.chinhphu.org.vn), của một cơ quan (Bộ GD&ĐT-www.moet.edu.vn), báo chí (www.thanhnien.com.vn), của một chủ đề (www.thuvientinhoc.vn)...

1.5.6 *World Wide Web*

World Wide Web (Web) là một dịch vụ hay còn gọi là một công cụ trên Internet ra đời gần đây nhất nhưng phát triển nhanh nhất hiện nay. Nó cung cấp một giao diện vô cùng thân thiện với người dùng, dễ sử dụng, thuận lợi và đơn giản để tìm kiếm thông tin.

Thực chất Web không phải là một hệ thống cụ thể với tên gọi như trên mà là một tập hợp các công cụ tiện ích và siêu giao diện (meta-Interface) giúp người sử dụng có thể tự tạo ra các "siêu văn bản" và cung cấp cho những người dùng khác trên Internet.

1.5.7 *Phân biệt Inetrnet và WWW*

WWW chỉ là một phần nhỏ của Internet. Internet bao hàm tất cả phần cứng và phần mềm, bao gồm HTTP, FTP (File Transfer Protocol, sẽ đề cập đến sau), Emails và Newgroups. WWW chủ yếu xây dựng trên các ký tự và hình ảnh mà chúng ta có thể xem bằng các trình duyệt Web.

1.5.8 *Web page*

Web page là trang Web, là một loại tập tin đặc biệt được viết bằng ngôn ngữ siêu văn bản HTML. Web page có thể hiển các thông tin văn bản, âm thanh, hình ảnh, video, … Trang Web này được đặt trên một máy chủ Web sao cho các máy khách có thể truy cập được nó, tập hợp nhiều trang Web có liên quan, ràng buộc đến nhau cho chúng ta một Web Site.

1.6 Cách thức tổ chức và xây dựng một Web Site

Việc xây dựng một trang Web để được nhiều người quan tâm là một công việc không đơn giản. Việc thiết kế không chỉ lưu ý đến vấn đề là mọi người có truy cập vào trang Web của mình hơn một lần hay không mà thông tin trên đó phải phụ thuộc hoàn toàn vào mục đích của việc tạo chúng.

Để tạo được một site hữu hiệu, ta phải chú ý đến những vấn đề sau:

- Có một mục đích rõ ràng: Đây là điểm quan trọng trong việc bắt đầu thiết kế Web.
- Luôn luôn nghĩ đến những client-người sẽ truy cập vào site: Chúng ta phải xét đến một số đặc điểm của người truy cập như là: lứa tuổi, nghề nghiệp, sở thích, thời gian rảnh rỗi ...
- Sử dụng những mục có khả năng dowload về thật nhanh. Một trong những lý do khiến những người truy cập vào trang Web của chúng ta cảm thấy chán nản là phải đợi lâu cho việc lấy tin và đó chính là lúc người ta sẽ nhấn vào nút Stop.
- Cố gắng làm cho Web Site của mình xuất hiện một cách trực quan: không nên cho quá nhiều màu sắc hoặc không có màu sắc trong trang.
- Đừng có cố gắng cho mọi thứ vào trong một trang: Một trang Web bừa

bộn sẽ gây ra cảm giác chán nản và nhức mắt.

- Tổ chức nội dung một cách thông minh: Nên nhớ rằng site của mình tạo ra không chỉ có "độ sâu" một bậc, do vậy chỉ có những thông tin thật cần thiết mới cho vào trang chủ. Ví dụ: giới thiệu tên công ty, mục đích, một số sản phẩm ...
- Kiểm tra, chạy thử site vừa thiết kế một cách kỹ trước khi đưa lên Web Server: Thử kiểm tra site bởi các trình duyệt Web, trên các hệ điều hành khác nhau hay là các chế độ kích thước cửa sổ khác nhau để đảm bảo rằng site của chúng ta thông suốt.

1.7 Phân loại Web

Dựa vào đặc trưng, kết nối dữ liệu và công cụ phát triển người ta có thể chia ra làm 3 loại Web sau đây:

1.7.1 Static pages (Web tĩnh):

Tính chất của các trang Web này là chỉ bao gồm các nội dung hiển thị cho người dùng xem. Ví dụ: hiển thị các trang dạng text, hình ảnh đơn giản chẳng hạn như một cốc cà phê đang bốc khói ...

1.7.2 Form pages (Mẫu biểu):

Ngoài nội dung như ở trang Web tĩnh, nó còn chứa các mẫu biểu (form) cho phép nhập các yêu cầu từ phía người sử dụng. Khi người dùng điền xong các form, ấn nút "Submit" và tất cảc các thông tin (yêu cầu) sẽ đưa đến đầu vào của một chương trình CGI (Common Gateway Interface) chạy trên Server. (thường thì các CGI xử lý và cất giữ thông tin vào các file dữ liệu trên Server rồi thông báo trả lại cho khách hàng).

Loại Web này thường được dùng để làm các phiếu điều tra, trưng cầu ý kiến, mua hàng ..v..v..

1.7.3 Dynamic Web (Web động)

Nội dung của trang Web động như trong 1 trang Web tĩnh, ngoài ra còn có nhúng các đoạn mã lệnh cho phép truy nhập cơ sở dữ liệu trên mạng.

Tuỳ theo nhu cầu, ứng dụng có thể cung cấp khả năng truy cập dữ liệu, tìm kiếm thông tin, …

1.8 Câu hỏi và bài tập chương 1

Câu 1: Phân biệt mạng Intranet và mạng Internet.

Câu 2: Mạng Internet sử dụng những giao thức nào? Chức năng của chúng?

Câu 3: Cấu trúc các lớp của địa chỉ IP.

Câu 4: Các khái niệm URL, hyperlinks, web page, web browser, web server.

Câu 5: Phân biệt Internet và World Wide Web.

Câu 6: Cách thức xây dựng một website.

Câu 7: Phân biệt các loại web

Chương 2

LẬP TRÌNH WEB VỚI NGÔN NGỮ ĐÁNH DẤU SIÊU VĂN BẢN (HTML)

2.1 Khái niệm ngôn ngữ HTML

HTML viết tắt của HyperText Mark-up Language (ngôn ngữ đánh dấu siêu văn bản). Có thể định nghĩa HTML:

Là một tập hợp các quy tắc và các thẻ (tag) được sử dụng để quy định các thức trình bày, hiển thị nội dung của các trang Web, tập hợp các quy tắc và thẻ này phải tuân theo một chuẩn quốc tế, đảm bảo cho các trình duyệt Web khác nhau,trên các nền phần cứng và hệ điều hành khác nhau đều hiểu được và hiển thị đúng nội dung của các trang Web.

HTML không phải là một ngôn ngữ lập trình, nó là một ngôn ngữ đánh dấu.

HTML dễ hiểu hơn nhiều so với hầu hết các ngôn ngữ lập trình.

Một tài liệu HTML là một tệp tin văn bản trong đó có sử dụng các thẻ HTML để quy định cách thức hiển thị văn bản khi nó được mở bởi một trình duyệt Web.

Cơ bản các thẻ định dạng trong HTML thường có từng cặp gồm: thẻ mở **<tag>** và thẻ đóng **</tag>**.

Các văn bản nằm giữa hai thẻ này sẽ được chịu tác động định dạng bởi thẻ. Ví dụ, thẻ **<B>** dùng để định dạng chữ in đậm, khi đó văn bản "**<B>**Hello**</B>**" sẽ được hiển thị là "**Hello**".

2.2 Lập trình web với ngôn ngữ HTML

2.2.1 Các thẻ định dạng cấu trúc của HTML

Các thẻ xác định cấu trúc tài liệu là bắt buộc phải có trong một tài liệu HTML. Sau đây chúng ta sẽ lần lượt học cách sử dụng các thẻ định dạng cấu trúc của một tài liệu HTML cơ bản.

a. HTML

Cặp thẻ này được sử dụng để xác nhận một tài liệu là tài liệu HTML, tức là nó có sử dụng các thẻ HTML để trình bày. Toàn bộ nội dung của tài liệu được đặt giữa cặp thẻ này. Tất cả các tập tin HTML đều bắt đầu bằng thẻ **<HTML>**, thẻ này thông báo cho trình duyệt Web biết rằng nó đang đọc một tài liệu có chứa các mã HTML và cuối các tập tin HTML sẽ là thẻ đóng tương ứng **</HTML>** thông báo kết thúc một tài liệu HTML.

Cú pháp:

```
<HTML>
... Toàn bộ nội dung của tài liệu được đặt ở đây
</HTML>
```

Trình duyệt sẽ xem các tài liệu không sử dụng thẻ **<HTML>** như những tệp văn bản bình thường.

b. HEAD

Một tài liệu HTML gồm có 2 phần: phần mở đầu và phần nội dung chính. Phần mở đầu giống như phần giới thiệu, các trình duyệt Web sử dụng phần mở đầu này để thu nhặt các thông tin khác nhau về tài liệu HTML này, chẳng hạn như tiêu

đề của tài liệu, các quan hệ được thiết lập giữa tài liệu và các thư mục. Thẻ **<HEAD>** được dùng để xác định phần mở đầu cho tài liệu.

Cú pháp:

```
<HTML>
... Phần mở đầu (header) của tài liệu được đặt ở đây
</HTML>
```

c. TITLE

Chúng ta có thể đặt tiêu đề cho tài liệu HTML của mình. Tiêu đề này sẽ được hiển thị trên thanh tiêu đề của trình duyệt. Cặp thẻ này chỉ có thể sử dụng trong phần mở đầu của tài liệu, tức là nó phải nằm trong thẻ phạm vi giới hạn bởi cặp thẻ **<HEAD>**.

Cú pháp:

```
<TITLE>Tiêu đề của tài liệu được đặt ở đây</TITLE>
```

d. BODY

Thẻ này được sử dụng để xác định phần nội dung chính của tài liệu. Cũng có thể sử dụng các tham số của thẻ để đặt ảnh nền cho tài liệu, màu nền, màu văn bản siêu liên kết, đặt lề cho trang tài liệu… Những thông tin này được đặt ở phần tham số của thẻ.

Cú pháp:

```
<BODY>
 .... phần nội dung của tài liệu được đặt ở đây
</BODY>
```

Trên đây là cú pháp cơ bản của thẻ **<BODY>**, tuy nhiên bắt đầu từ phiên bản HTML 3.2 thì có nhiều thuộc tính được sử dụng trong thẻ **<BODY>**.

Như vậy một tài liệu HTML cơ bản có cấu trúc như sau:

```
<HTML>
<HEAD>
<TITLE>Tiêu đề của tài liệu</TITLE>
</HEAD>
<BODY>
... Nội dung của tài liệu
</BODY>
</HTML>
```

Sau đây là các thuộc tính chính:

BACKGROUND=	Đặt một ảnh nào đó làm ảnh nền (background) cho văn bản. Giá trị của tham số này (phần sau dấu bằng) là URL của file ảnh. Nếu kích thước ảnh nhỏ hơn cửa sổ trình duyệt thì toàn bộ màn hình cửa sổ trình duyệt sẽ được lát kín bằng nhiều ảnh.

BGCOLOR=	Đặt mầu nền cho trang khi hiển thị. Nếu cả hai tham số **BACKGROUND** và **BGCOLOR** cùng có giá trị thì trình duyệt sẽ hiển thị mầu nền trước, sau đó mới tải ảnh lên phía trên.
TEXT=	Xác định màu chữ của văn bản, kể cả các đề mục.
ALINK= **VLINK=** **LINK=**	Xác định màu sắc cho các siêu liên kết trong văn bản. Tương ứng, alink (*active link*) là liên kết đang được kích hoạt - tức là khi đã được kích chuột lên; vlink (*visited link*) chỉ liên kết đã từng được kích hoạt;

e. Chú thích

Cặp thẻ này cho phép người biên soạn tài liệu HTML có thể thêm vào trong các tài liệu HTML những chú thích cần thiết, hoặc có thể sử dụng cặp thẻ này để thông báo cho trình duyệt bỏ qua một đoạn mã lệnh HTML. Các văn bản được đặt giữa hai thẻ này sẽ không được trình duyệt hiển thị.

Cú pháp:

```
<!-- Các chú thích được đặt ở đây -->
```

2.2.2 *Các thể định dạng khối*

a. Thể ***<P>***

Thẻ **<P>** được sử dụng để định dạng một đoạn văn bản.

Cú pháp:

```
<P>Nội dung đoạn văn bản</P>
```

b. Các thể định dạng đề mục H1/H2/H3/H4/H5/H6

HTML hỗ trợ 6 mức đề mục. Chú ý rằng đề mục chỉ là các chỉ dẫn định dạng về mặt logic, tức là mỗi trình duyệt sẽ thể hiện đề mục dưới một khuôn dạng thích hợp. Có thể ở trình duyệt này là font chữ 14 point nhưng sang trình duyệt khác là font chữ 20 point. Đề mục cấp 1 là cao nhất và giảm dần đến cấp 6. Thông thường văn bản ở đề mục cấp 5 hay cấp 6 thường có kích thước nhỏ hơn văn bản thông thường.

Dưới đây là các thẻ dùng để định dạng văn bản ở dạng đề mục:

<H1> ... </H1>	Định dạng đề mục cấp 1
<H2> ... </H2>	Định dạng đề mục cấp 2
<H3> ... </H3>	Định dạng đề mục cấp 3
<H4> ... </H4>	Định dạng đề mục cấp 4
<H5> ... </H5>	Định dạng đề mục cấp 5
<H6> ... </H6>	Định dạng đề mục cấp 6

c. Thể xuống dòng ***
***

Thẻ này không có thẻ kết thúc tương ứng (**</BR>**), nó có tác dụng chuyển sang dòng mới. Lưu ý, nội dung văn bản trong tài liệu HTML sẽ được trình duyệt Web thể hiện liên tục, các khoảng trắng liền nhau, các ký tự tab, ký tự xuống dòng đều được coi như một khoảng trắng. Để xuống dòng, ta phải sử dụng thẻ **
**.

d. 2.4 Thể ***<PRE>***

Để giới hạn đoạn văn bản đã được định dạng sẵn ta có thể sử dụng thẻ **<PRE>**. Văn bản ở giữa hai thẻ này sẽ được thể hiện giống hệt như khi chúng được đánh vào, ví dụ dấu xuống dòng trong đoạn văn bản giới hạn bởi thẻ **<PRE>** sẽ có ý nghĩa chuyển sang dòng mới (trình duyệt sẽ không coi chúng như dấu cách).

Cú pháp:

```
<PRE>Văn bản đã được định dạng</PRE>
```

2.2.3 *Các thể định dạng danh sách*

Cú pháp:

```
<UL>
<LI> Mục thứ nhất
<LI> Mục thứ hai
</UL>
```

Có 4 kiểu danh sách:

- Danh sách không sắp xếp (hay không đánh số) **<UL>**
- Danh sách có sắp xếp (hay có đánh số) **<OL>**, mỗi mục trong danh sách được sắp xếp thứ tự.
- Danh sách thực đơn **<MENU>**
- Danh sách phân cấp **<DIR>**

Với nhiều trình duyệt, danh sách phân cấp và danh sách thực đơn giống danh sách không đánh số, có thể dùng lẫn với nhau. Với thẻ OL ta có cú pháp sau:

```
<OL TYPE=1/a/A/i/I>
        <LI>Muc thu nhat
        <LI>Muc thu hai
        <LI>Muc thu ba
</OL>
```

Trong đó: TYPE =1 Các mục được sắp xếp theo thứ tự 1, 2, 3...
=a Các mục được sắp xếp theo thứ tự a, b, c...
=A Các mục được sắp xếp theo thứ tự A, B, C...
=i Các mục được sắp xếp theo thứ tự i, ii, iii...
=I Các mục được sắp xếp theo thứ tự I, II, III...

Ngoài ra còn thuộc tính **START**= xác định giá trị khởi đầu cho danh sách.

Thẻ **<LI>** có thuộc tính **TYPE=** xác định ký hiệu đầu dòng (bullet) đứng trước mỗi mục trong danh sách. Thuộc tính này có thể nhận các giá trị : ***disc*** (chấm tròn đậm); ***circle*** (vòng tròn); ***square*** (hình vuông).

2.2.4 *Các thẻ định dạng ký tự*

a. Các thẻ định dạng in ký tự

Sau đây là các thẻ được sử dụng để quy định các thuộc tính như in nghiêng, in đậm, gạch chân... cho các ký tự, văn bản khi được thể hiện trên trình duyệt.

Thẻ	Thuộc tính
<B> ... </B> **<STRONG> ... </STRONG>**	In chữ đậm
<I> ... </I> **<EM> ... </EM>**	In chữ nghiêng
<U> ... </U>	In chữ gạch chân
<DFN>	Đánh dấu đoạn văn bản giữa hai thẻ này là định nghĩa của một từ. Chúng thường được in nghiêng hoặc thể hiện qua một kiểu đặc biệt nào đó.
<S> ... </S> **<STRIKE> ... </STRIKE>**	In chữ bị gạch ngang.
<BIG> ... </BIG>	In chữ lớn hơn bình thường bằng cách tăng kích thước font hiện thời lên một. Việc sử dụng các thẻ **<BIG>** lồng nhau tạo ra hiệu ứng chữ tăng dần. Tuy nhiên đối với mỗi trình duyệt có giới hạn về kích thước đối với mỗi font chữ, vượt quá giới hạn này, các thẻ **<BIG>** sẽ không có ý nghĩa.
<SMALL> ... </SMALL>	In chữ nhỏ hơn bình thường bằng cách giảm kích thước font hiện thời đi một. Việc sử dụng các thẻ **<SMALL>**lồng nhau tạo ra hiệu ứng chữ giảm dần. Tuy nhiên đối với mỗi trình duyệt có giới hạn về kích thước đối với mỗi font chữ, vượt quá giới hạn này, các thẻ **<SMALL>** sẽ không có ý nghĩa.
^{...}	Định dạng chỉ số trên (SuperScript)
_{...}	Định dạng chỉ số dưới (SubScript)
<BASEFONT>	Định nghĩa kích thước font chữ được sử dụng cho đến hết văn bản. Thẻ này chỉ có một tham số size= xác định cỡ chữ. Thẻ **<BASEFONT>** không có thẻ kết thúc.
<FONT> ... </FONT>	Chọn kiểu chữ hiển thị. Trong thẻ này có thể đặt hai tham số size= hoặc color= xác định cỡ chữ và màu sắc đoạn văn bản nằm giữa hai thẻ. Kích thước có thể là tuyệt đối (nhận giá trị từ 1 đến 7) hoặc tương đối (+2,-4...) so với font chữ hiện tại.

b. Căn lề văn bản trong trang Web

Trong trình bày trang Web của mình chúng ta luôn phải chú ý đến việc căn lề các văn bản để trang Web có được một bố cục đẹp. Một số các thẻ định dạng như **<P>**, **<H**n**>**, **<IMG>**... đều có tham số **ALIGN** cho phép căn lề các văn bản nằm trong phạm vi giới hạn bởi của các thẻ đó.

Các giá trị cho tham số **ALIGN**:

LEFT	Căn lề trái
CENTER	Căn giữa trang
RIGHT	Căn lề phải

Ngoài ra, chúng ta có thể sử dụng thẻ **CENTER** để căn giữa trang một khối văn bản.

Cú pháp:

```
<CENTER>Văn bản sẽ được căn giữa trang</CENTER>
```

c. Các ký tự đặc biệt

Ký tự & được sử dụng để chỉ chuỗi ký tự đi sau được xem là một thực thể duy nhất. Ký tự ; được sử dụng để tách các ký tự trong một từ.

Ký tự	Mã ASCII	Tên chuỗi
<	&#060	&lt
>	&#062	&gt
&	&#038	&amp

d. Sử dụng màu sắc trong thiết kế các trang Web

Một màu được tổng hợp từ ba thành phần màu chính, đó là: Đỏ (Red), Xanh lá cây (Green), Xanh nước biển (Blue). Trong HTML một giá trị màu là một số nguyên dạng hexa (hệ đếm cơ số 16) có định dạng như sau:

#RRGGBB

Trong đó:

RR - là giá trị màu Đỏ.

GG - là giá trị màu Xanh lá cây.

BB - là giá trị màu Xanh nước biển.

Màu sắc có thể được xác định qua thuộc tính bgcolor= hay color=. Sau dấu bằng có thể là giá trị RGB hay tên tiếng Anh của màu. Với tên tiếng Anh, ta chỉ có thể chỉ ra 16 màu trong khi với giá trị RGB ta có thể chỉ tới 256 màu.

Sau đây là một số giá trị màu cơ bản:

Màu sắc	Giá trị	Tên tiếng Anh
Đỏ	#FF0000	RED
Đỏ sẫm	#8B0000	DARKRED
Xanh lá cây	#00FF00	GREEN
Xanh nhạt	#90EE90	LIGHTGREEN
Xanh nước biển	#0000FF	BLUE
Vàng	#FFFF00	YELLOW
Vàng nhạt	#FFFFE0	LIGHTYELLOW
Trắng	#FFFFFF	WHITE

Đen	#000000	BLACK
Xám	#808080	GRAY
Nâu	#A52A2A	BROWN
Tím	#FF00FF	MAGENTA
Tím nhạt	#EE82EE	VIOLET
Hồng	#FFC0CB	PINK
Da cam	#FFA500	ORANGE
Màu đồng phục hải quân	#000080	NAVY
	#4169E1	ROYALBLUE
	#7FFFD4	AQUAMARINE

Cú pháp:

```
<BODY
  LINK            = color
  ALINK           = color
  VLINK           = color
  BACKGROUND= url()
  BGCOLOR         = color
  TEXT            = color
  TOPMARGIN       = pixels
  RIGHTMARGIN = pixels
  LEFTMARGIN      = pixels
>
.... phần nội dung của tài liệu được đặt ở đây
</BODY>
```

Sau đây là ý nghĩa các tham số của thẻ **<BODY>**:

Các tham số	Ý nghĩa
LINK	Chỉ định màu của văn bản siêu liên kết
ALINK	Chỉ định màu của văn bản siêu liên kết đang đang chọn
VLINK	Chỉ định màu của văn bản siêu liên kết đã từng mở
BACKGROUND	Chỉ định địa chỉ của ảnh dùng làm nền
BGCOLOR	Chỉ định màu nền
TEXT	Chỉ định màu của văn bản trong tài liệu
SCROLL	YES/NO - Xác định có hay không thanh cuộn
TOPMARGIN	Lề trên
RIGHTMARGIN	Lề phải
LEFTMARGIN	Lề trái

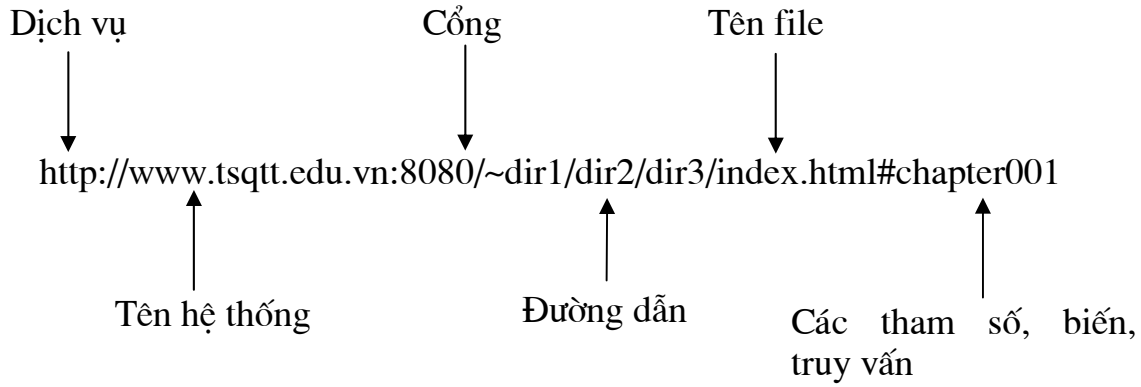
e. Chọn kiểu chữ cho văn bản

Cú pháp:

```
<FONT
FACE      = font-name
COLOR     = color
SIZE      = n >
... nội dung đoạn văn bản
</FONT>
```

f. Khái niệm văn bản siêu liên kết

Văn bản siêu liên kết hay còn gọi là siêu văn bản là một từ, một cụm từ hay một câu trên trang Web được dùng để liên kết tới một trang Web khác. Siêu văn bản là môi trường trong đó chứa các liên kết (link) của các thông tin. Do WWW cấu thành từ nhiều hệ thống khác nhau, cần phải có một quy tắc đặt tên thống nhất cho tất cả các văn bản trên Web. Quy tắc đặt tên đó là URL (Universal Resource Locator).



Hình 2.1 Các thành phần của URL được minh hoạ ở hình trên.

- Dịch vụ: Là thành phần bắt buộc của URL. Nó xác định cách thức trình duyệt của máy khách liên lạc với máy phục vụ như thế nào để nhận dữ liệu. Có nhiều dịch vụ như **http, wais, ftp, gopher, telnet**.
- Tên hệ thống: Là thành phần bắt buộc của URL. Có thể là tên miền đầy đủ của máy phục vụ hoặc chỉ là một phần tên đầy đủ – trường hợp này xảy ra khi văn bản được yêu cầu vẫn nằm trên miền của site. Tuy nhiên nên sử dụng đường dẫn đầy đủ.
- Cổng: Không là thành phần bắt buộc của URL. Cổng là địa chỉ socket của mạng dành cho một giao thức cụ thể. Giao thức http ngầm định nối với cổng 8080.
- Đường dẫn thư mục: Là thành phần bắt buộc của URL. Phải chỉ ra đường dẫn tới file yêu cầu khi kết nối với bất kỳ hệ thống nào. Có thể đường dẫn trong URL khác với đường dẫn thực sự trong hệ thống máy phục vụ. Tuy nhiên có thể rút gọn đường dẫn bằng cách đặt biệt danh (alias). Các thư mục trong đường dẫn cách nhau bởi dấu gạch chéo (/).
- Tên file: Không là thành phần bắt buộc của URL. Thông thường máy phục vụ được cấu hình sao cho nếu không chỉ ra tên file thì sẽ trả về file ngầm định trên thư mục được yêu cầu. File này thường có tên là

index.html, index.htm, default.html hay *default.htm* (với các Web site động thì file mặc định có thể có phần mở rộng là asp, aspx, jsp hay php…). Nếu cũng không có các file này thì thường kết quả trả về là danh sách liệt kê các file hay thư mục con trong thư mục được yêu cầu

- Các tham số: Không là thành phần bắt buộc của URL. Nếu URL là yêu cầu tìm kiếm trên một cơ sở dữ liệu thì truy vấn sẽ gắn vào URL, đó chính là đoạn mã đằng sau dấu chấm hỏi (?).URL cũng có thể trả lại thông tin được thu thập từ form. Trong trường hợp dấu thăng (#) xuất hiện đoạn mã đăng sau là tên của một vị trí (location) trong file được chỉ ra.

Để tạo ra một siêu văn bản chúng ta sử dụng thẻ **<A>**.

Cú pháp:

```
<A
        HREF        =  url
        NAME        = name
        TABINDEX    = n
        TITLE       = title
        TARGET      = _blank / _self
>
... siêu văn bản
</A>
```

Ý nghĩa các tham số:

Các tham số	Ý nghĩa
HREF	Địa chỉ của trang Web được liên kết, là một URL nào đó.
NAME	Đặt tên cho vị trí đặt thẻ.
TABLEINDEX	Thứ tự di chuyển khi ấn phím Tab
TITLE	Văn bản hiển thị khi di chuột trên siêu liên kết.
TARGET	Mở trang Web được liên trong một cửa sổ mới (*_blank*) hoặc trong cửa sổ hiện tại (*_self*), trong một frame (tên frame).

g. Địa chỉ tương đối

URL được trình bày ở trên là URL tuyệt đối. Ngoài ra còn có URL tương đối hay còn gọi là URL không đầy đủ. Địa chỉ tương đối sử dụng sự khác biệt tương đối giữa văn bản hiện thời và văn bản cần tham chiếu tới. Các thành phần trong URL được ngăn cách bằng ký tự ngăn cách (ký tự gạch chéo /). Để tạo ra URL tương đối, đầu tiên phải sử dụng ký tự ngăn cách. URL đầy đủ hiện tại sẽ được sử dụng để tạo nên URL đầy đủ mới. Nguyên tắc là các thành phần bên trái dấu ngăn cách của URL hiện tại được giữ nguyên, các thành phần bên phải được thay thế bằng thành phần URL tương đối. Chú ý rằng trình duyệt không gửi URL tương đối, nó bổ sung vào URL cơ sở đã xác định trước thành phần URL tương đối xác định sau thuộc tính **href=**. Ký tự đầu tiên sau dấu bằng sẽ xác định các thành phần nào của URL hiện tại sẽ tham gia để tạo nên URL mới.

Ví dụ, với địa chỉ URL: http://dit.tsqtt.edu.vn/HTML/ thì:

- Dấu hai chấm (:) chỉ dịch vụ giữ nguyên nhưng thay đổi phần còn lại.
- Dấu gạch chéo (/) chỉ dịch vụ và máy phục vụ giữ nguyên nhưng toàn bộ đường dẫn thay đổi. Ví dụ /JavaScript/index.htm sẽ tải file index.htm của thư mục JavaScript trên máy phục vụ dit.tsqtt.edu.vn.
- Không có dấu phân cách chỉ có tên file là thay đổi. Ví dụ index.htm sẽ tải file index.htm ở trong thư mục HTML của máy phục vụ dit.tsqtt.edu.vn.
- Dấu thăng (#): chỉ dịch vụ, máy phuc vụ, đường dẫn và cả tên file giữ nguyên, chỉ thay đổi vị trí trong file.

Do đường dẫn được xem là đơn vị độc lập nên có thể sử dụng phương pháp đường dẫn tương đối như trong UNIX hay MS-DOS (tức là dấu chấm (**.**) chỉ thư mục hiện tại còn hai dấu chấm (**..**) chỉ thư mục cha của thư mục hiện tại).

URL cơ sở có thể được xác định bằng thẻ <**BASE**>.

h. Kết nối mailto

Nếu đặt thuộc tính **href=** của thẻ <**A**> giá trị mailto:address@domain thì khi kích hoạt kết nối sẽ kích hoạt chức năng thư điện tử của trình duyệt.

```
<ADDRESS>
Trang WEB này được
<A href="mailto:webmaster@vnuh.edu.vn" >WEBMASTER
<\A> bảo trì
<\ADDRESS>
```

i. Vẽ một đường thẳng nằm ngang

Cú pháp:

```
<HR
        ALIGN      = LEFT / CENTER / RIGHT
        COLOR      = color
        NOSHADE
        SIZE       = n
        WIDTH      = width
>
```

Ý nghĩa các tham số:

Các tham số	Ý nghĩa
ALIGN	Căn lề (căn trái, căn phải, căn giữa)
COLOR	Đặt màu cho đường thẳng
NOSHADE	Không có bóng
SIZE	Độ dày của đường thẳng
WIDTH	Chiều dài (tính theo pixel hoặc % của bề rộng cửa sổ trình duyệt).

Thẻ này giống như thẻ <**BR**>, nó cũng không có thẻ kết thúc tương ứng.

2.2.5 *Các thẻ chèn âm thanh, hình ảnh*

a. Giới thiệu

Liên kết với file đa phương tiện cũng tương tự như liên kết bình thường. Tuy vậy phải đặt tên đúng cho file đa phương tiện. Phần mở rộng của file phải cho biết kiểu của file

Kiểu	Mở rộng	Mô tả
Image/GIF	**.gif**	Viết tắt của Graphics Interchange Format. Khuôn dạng này xuất hiện khi mọi người có nhu cầu trao đổi ảnh trên nhiều hệ thống khác nhau. Nó được sử dụng trên tất cả các hệ thống hỗ trợ giao diện đồ hoạ. Định dạng GIF là định dạng chuẩn cho mọi trình duyệt WEB. Nhược điểm của nó là chỉ thể hiện được 256 màu.
Image/JPEG	**.jpeg**	Viết tắt của Joint Photographic Expert Group. Là khuôn dạng ảnh khác nhưng có thêm khả năng nén.Ưu điểm nổi bật của khuôn dạng này là lưu trữ được hàng triệu màu và độ nén cao nên kích thước file ảnh nhỏ hơn và thời gian download nhanh hơn. Nó là cơ sở cho khuôn dạng MPEG. Tất cả các trình duyệt đều có khả năng xem ảnh JPEG.
Image/TIFF	**.tiff**	Viết tắt của Tagged Image File Format. Được Microsoft thiết kế để quét ảnh từ máy quét cũng như tạo các ấn phẩm.
Text/HTML	**.html, .htm**	
PostScript	**.eps, .ps**	Được tạo ra để hiển thị và in các văn bản có chất lượng cao.
Adobe Acrobat	**.pdf**	Viết tắt của Portable Document Format. Acrobat cũng sử dụng các siêu liên kết ngay trong văn bản cũng giống như HTML. Từ phiên bản 2.0, các sản phẩm của Acrobat cho phép liên kết giữa nhiều văn bản. Ưu điểm lớn nhất của nó là khả năng WYSISYG.
Video/MPEG	**.mpeg**	Viết tắt của Motion Picture Expert Group, là định dạng dành cho các loại phim (video). Đây là khuôn dạng thông dụng nhất dành cho phim trên WEB.
Video/AVI	**.avi**	Là khuôn dnạg phim do Microsoft đưa ra.
Video/Quick Time	**.mov**	Do Apple Computer đưa ra, chuẩn video này được cho là có nhiều ưu điểm hơn MPEG và AVI. Mặc dù đã được tích hợp vào nhiều trình duyệt nhưng vẫn chưa phổ biến bằng hai loại định dạng trên.
Sound/AU	**.au**	

Sound/MIDI	**.mid**	Là khuôn dạng dành cho âm nhạc điện tử hết sức thông dụng được nhiều trình duyệt trên các hệ thống khác nhau hỗ trợ. File Midi được tổng hợp số hoá trực tiếp từ máy tính.
Sound/Real Audio	**.ram**	Định dạng audio theo dòng. Một bất tiện khi sử dụng các định dạng khác là file âm thanh thường có kích thước lớn - do vậy thời gian tải xuống lâu, Trái lại audio dòng bắt đầu chơi ngay khi tải được một phần file trong khi vẫn tải về các phần khác. Mặc dù file theo định dạng này không nhỏ hơn so với các định dạng khác song chính khả năng dòng đã khiến định dạng này phù hợp với khả năng chơi ngay lập tức.
VRML	**.vrml**	Viết tắt của Virtual Reality Modeling Language. Các file theo định dạng này cũng giống như HTML. Tuy nhiên do trình duyệt có thể hiển thị được cửa sổ 3 chiều nên người xem có thể cảm nhận được cảm giác ba chiều.

b. Đưa âm thanh vào một tài liệu HTML

Cú pháp:

```
<BGSOUND
        SRC = url
        LOOP = n
>
```

Thẻ này không có thẻ kết thúc tương ứng (**</BGSOUND>**). Để chơi lặp lại vô hạn cần chỉ định **LOOP =** -1 hoặc **LOOP =** *INFINITE*. Thẻ **<BGSOUND>** phải được đặt trong phần mở đầu (tức là nằm trong cặp thẻ **<HEAD>**).

c. Chèn một hình ảnh, một đoạn video vào tài liệu HTML

Để chèn một file ảnh (.jpg, .gif, .bmp) hoặc video (.mpg, .avi) vào tài liệu HTML, ta có thể sử dụng thẻ **IMG**.

Cú pháp:

```
<IMG
ALIGN     = TOP/MIDDLE/BOTTOM
ALT       = text
BORDER    = n
SRC       = url
WIDTH     = width
HEIGHT    = height
HSPACE    = vspace
VSPACE    = hspace
TITLE     = title
```

```
DYNSRC    = url
START     =
FILEOPEN/MOUSEOVER
LOOP      = n>
```

Trong đó:

Các tham số	Ý nghĩa
ALIGN = TOP/ MIDDLE/ BOTTOM/ LEFT/ RIGHT	Căn hàng văn bản bao quanh ảnh
ALT = text	Chỉ định văn bản sẽ được hiển thị nếu chức năng show picture của browser bị tắt đi hay hiển thị thay thế cho ảnh trên những trình duyệt không có khả năng hiển thị đồ hoạ. Văn bản này còn được gọi là nhãn của ảnh. Đối với trình duyệt có khả năng hỗ trợ đồ hoạ, dòng văn bản này sẽ hiện lên khi di chuột qua ảnh hay được hiển thị trong vùng của ảnh nếu ảnh chưa được tải về hết. Chú ý phải đặt văn bản trong hai dấu nháy kép nếu triong văn bản chứa dấu cách hay các ký tự đặc biệt - trong trường hợp ngược lại có thể bỏ dấu nháy kép.
BORDER = n	Đặt kích thước đường viền được vẽ quanh ảnh (tính theo pixel).
SRC = url	Địa chỉ của file ảnh cần chèn vào tài liệu.
WIDTH/HEIGHT	Chỉ định kích thước của ảnh được hiển thị.
HSPACE/VSPACE	Chỉ định khoảng trống xung quanh hình ảnh (tính theo pixel) theo bốn phía trên, dưới, trái, phải.
TITLE = title	Văn bản sẽ hiển thị khi con chuột trỏ trên ảnh
DYNSRC = url	Địa chỉ của file video.
START = FILEOPEN/MOUSEOVER	Chỉ định file video sẽ được chơi khi tài liệu được mở hay khi trỏ con chuột vào nó. Có thể kết hợp cả hai giá trị này nhưng phải phân cách chúng bởi dấu phẩy.
LOOP = n/INFINITE	Chỉ định số lần chơi. Nếu LOOP = INFINITE thì file video sẽ được chơi vô hạn lần.

d. Image map

Image map cho phép click chuột lên một vùng nào đó trên một hình ảnh để mở một trang Web khác. Sử dụng thẻ MAP và AREA để thiết lập các thông tin cho một *image map*.

Cú pháp:

```
<MAP NAME="map-name">
<AREA SHAPE = RECT/CIRC/POLY
```

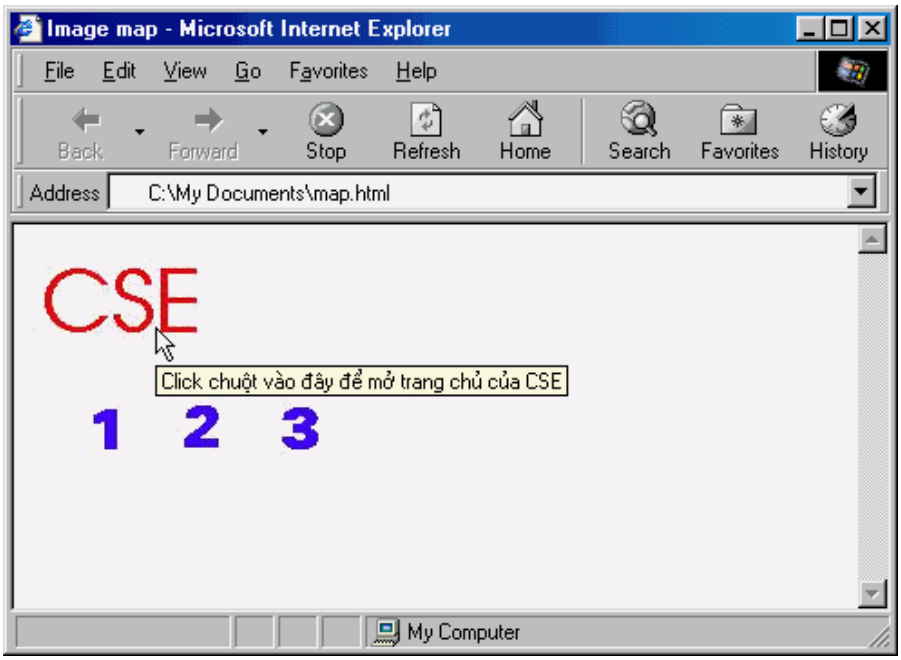
```
COORDS = coords
HREF = url
TITLE = text TARGET = _blank / _self >
<AREA ... >
<AREA ... >
...
</MAP>
```

Chèn ảnh:

```
<IMG SRC="image-name" USEMAP="#map-name">
```

Ví dụ minh họa:

```
<HTML>
<HEAD>
<TITLE>Image map</TITLE>
</HEAD>
<BODY>
<MAP NAME="map1">
<AREA SHAPE="RECT" COORDS="0, 0, 100, 50"
HREF="http:\\www.cse.com.vn" TITLE="Click chuột vào đây để mở
trang chủ của CSE">
<AREA SHAPE="CIRC" COORDS="42, 96, 25">
<AREA SHAPE="CIRC" COORDS="93, 96, 25">
<AREA SHAPE="CIRC" COORDS="147, 96, 25">
</MAP>
<IMG SRC="cse.jpg" USEMAP="#map1" BORDER=0>
</BODY>
</HTML>
```



Hình 2.1 Minh họa sử dụng Images map

2.2.6 *Chèn bảng*

Sau đây là các thẻ chính sử dụng để chèn bảng vào tài liệu HTML:

Thẻ	Thuộc tính
<TABLE> ... **</TABLE>**	Định nghĩa một bảng
<TR> ... </TR>	Định nghĩa một hàng trong bảng
<TD> ... </TD>	Định nghĩa một ô trong hàng
<TH> ... </TH>	Định nghĩa ô chứa tiêu đề của cột
<CAPTION> ... **</CAPTION>**	Tiêu đề của bảng

Cú pháp:

```
<TABLE
ALIGN                    = LEFT / CENTER / RIGHT
BORDER                   = n
BORDERCOLOR              = color
BORDERCOLORDARK          = color
BORDERCOLORLIGHT         = color
BACKGROUND               = url
BGCOLOR                  = color
CELLSPACING              = spacing
CELLPADDING              = pading>
<CAPTION>Tiêu đề của bảng biểu</CAPTION>
... Định nghĩa các dòng
<TR
ALIGN = LEFT/CENTER/RIGHT
VALIGN = TOP/MIDDLE/BOTTOM>
        ... Định nghĩa các ô trong dòng
<TD
ALIGN                = LEFT / CENTER / RIGHT
VALIGN               = TOP / MIDDLE / BOTTOM
BORDERCOLOR          = color
BORDERCOLORDARK      = color
BORDERCOLORLIGHT     = color
BACKBROUND           = url
BGCOLOR              = color
COLSPAN              = n
ROWSPAN              = n>
... Nội dung của ô
</TD>
</TR>
</TABLE>
```

Ý nghĩa các tham số:

Các tham số	Ý nghĩa
ALIGN / VALIGN	Căn lề cho bảng và nội dung trong mỗi ô.
BORDER	Kích thước đường kẻ chia ô trong bảng, được đo theo pixel. Giá trị 0 có nghĩa là không xác định lề, giữa các ô trong bảng chỉ có một khoảng trắng nhỏ để phân biệt. Nếu chỉ để border thì ngầm định border=1. Với những bảng có cấu trúc phức tạp, nên đặt lề để người xem có thể phân biệt rõ các dòng và cột.
BORDERCOLOR	Màu đường kẻ
BORDERCOLORDARK **BORDERCOLORLIGHT**	Màu phía tối và phía sáng cho đường kẻ nổi.
BACKGROUND	Địa chỉ tới tệp ảnh dùng làm nền cho bảng
BGCOLOR	Màu nền
CELLSPACING	Khoảng cách giữa các ô trong bảng
CELLPADDING	Khoảng cách giữa nội dung và đường kẻ trong mỗi ô của bảng.
COLSPAN	Chỉ định ô sẽ kéo dài trong bao nhiêu cột
ROWSPAN	Chỉ định ô sẽ kéo dài trong bao nhiêu hàng

2.2.7 *Sử dụng Khung – Frame*

Các *frame* cho phép bạn tổ chức cấu trúc nội dung các trang Web của mình bằng cách phức hợp nhiều tài liệu HTML để có thể xem chúng trong cùng một cửa sổ chính của trình duyệt Web. Để tạo một trang Web phức hợp bạn sử dụng các thẻ **FRAMESET** và **FRAME** để chia cửa sổ chính thành các khung chữ nhật (*frame*). Sau đó trong mỗi khung hình chữ nhật đó bạn chỉ định một tài liệu sẽ được hiển thị trong khung đó. Chú ý thẻ **FRAMESET** sẽ thay thế cho thẻ **BODY** trong một tài liệu HTML, điều đó có nghĩa là trong một tài liệu sử dụng thẻ **FRAMESET** sẽ không có thẻ **BODY** mà thay vào đó phần nội dung chính của tài liệu sẽ được định nghĩa bởi thẻ **FRAMESET**.

a. Thẻ FRAMESET

Cú pháp:

```
<FRAMESET
COLS            = col-widths
ROWS            = row-heights
BORDER          = pixels
BORDERCOLOR = color
FRAMEBORDER = 1/0 >
...
<FRAME ... >
```

```
<FRAME ... >
...
</FRAMESET>
```

Trong đó:

Các tham số	Ý nghĩa
COLS	Chia dọc cửa sổ thành các phần với kích thước chỉ định (theo pixel, % hoặc *).
ROWS	Chia ngang cửa sổ thành các phần với kích thước chỉ định (theo pixel, % hoặc *).
BORDER	Kích thước của đường kẻ viền khung
BORDERCOLOR	Chỉ định màu cho đường viền khung
FRAMEBORDER	Chỉ định có/không (1/0) hiển thị khung của các frame.

b. Thẻ FRAME

Cú pháp:

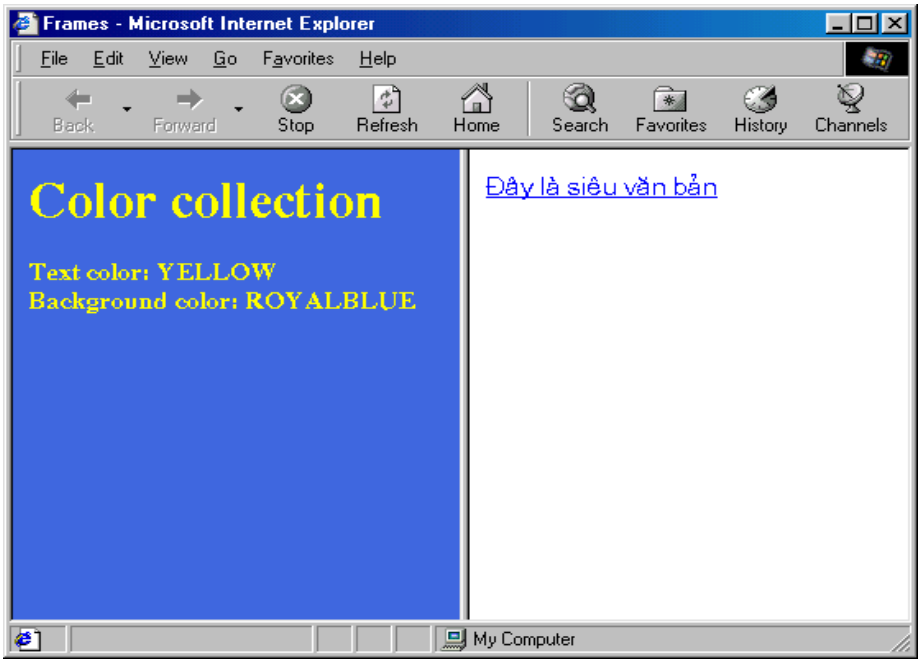
```
<FRAME
BORDERCOLOR = color
FRAMEBORDER = 0/1
MARGINHEIGHT = height
MARGINWIDTH = width
NAME = name
NORESIZE
SCROLLING = YES/NO
SRC = address
Target = Window_Name>
```

Trong đó:

Các tham số	Ý nghĩa
BORDERCOLOR	Màu đường viền khung.
FRAMEBORDER = 0 / 1	Chỉ định có/không viền khung.
MARGINHEIGHT	Khoảng cách giữa nội dung trong khung và đường viền ngang.
MARGINWIDTH	Khoảng cách giữa nội dung trong khung và đường viền dọc.
NAME	Đặt tên cho khung.
NORESIZE	Chỉ định không được thay đổi kích thước của khung.
SCROLLING = YES / NO	Chỉ định có hay không có thanh cuộn cho khung.
SRC	Địa chỉ của tài liệu sẽ được hiển thị trong khung.
Target	Chỉ ra cửa sổ nơi mà tài liệu được hiển thị

Ví dụ minh họa:

```
<HTML>
<HEAD><TITLE>Frames</TITLE></HEAD>
<FRAMESET
COLS = "30%, *">
<FRAME SRC = "vd45.htm">
<FRAME SRC = "vd48.htm">
</FRAMESET>
</HTML>
```



Hình 2.2 Sử dụng thẻ Frame

c. Thẻ IFRAME

Sử dụng thẻ **IFRAME** để đặt một frame vào trong một tài liệu HTML.

Cú pháp:

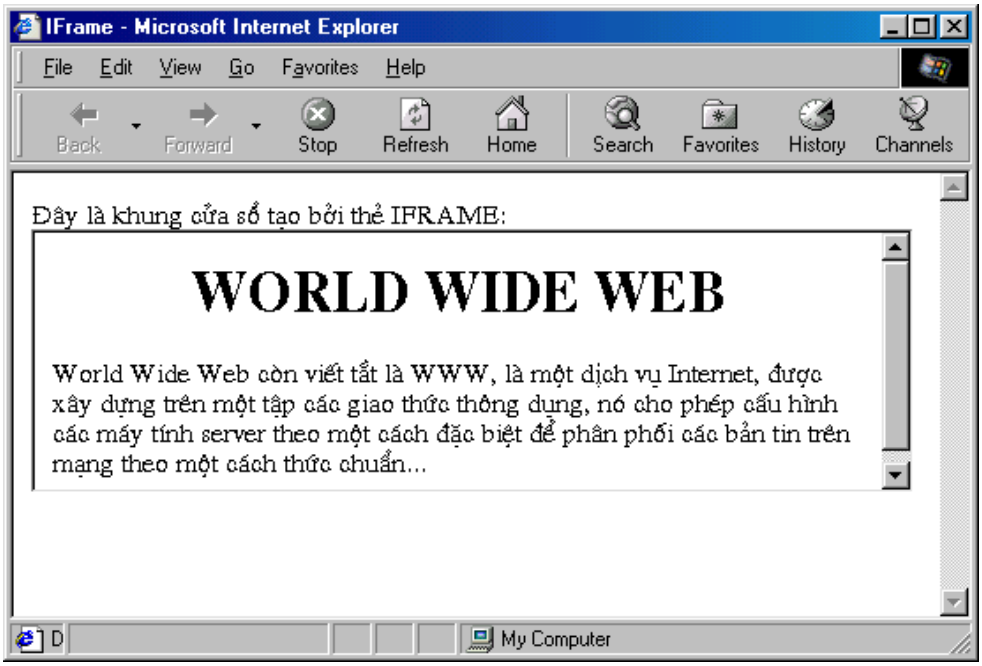
```
<IFRAME
ALIGN = LEFT / CENTER /RIGHT / TOP / BOTTOM
BORDER = pixels
BORDERCOLOR = color
FRAMEBORDER = 0/1
NORESIZE
SCROLLING = YES/NO
NAME = name
SRC = url
MARGINWIDTH = pixels
MARGINHEIGHT = pixels
WIDTH = n HEIGHT = n >
</IFRAME>
```

Trong đó:

Các tham số	Ý nghĩa
ALIGN	Căn lề cho khung
BORDER	Kích thước đường viền của khung
BORDERCOLOR	Màu đường viền của khung
FRAMEBORDER = 0 / 1	Khung có đường viền hay không
NORESIZE	Không được phép thay đổi kích thước của khung.
SCROLLING = YES / NO	Chỉ định khung có thanh cuộn hay không
NAME	Đặt tên cho khung
SRC	Địa chỉ tài liệu được hiển thị trong khung
MARGINWIDTH	Khoảng cách giữa văn bản nội dung của khung và các đường viền dọc.
MARGINHEIGHT	Khoảng cách giữa văn bản nội dung của khung và các đường viền ngang.
WIDTH	Đặt chiều rộng của khung
HEIGHT	Đặt chiều cao của khung

Ví dụ minh họa:

```
<BODY>
Đây là khung cửa sổ tạo bởi thẻ IFRAME:
<IFRAME
SRC = "vd43.htm"
WIDTH = 500>
</IFRAME>
</BODY>
```



Hình 2.3 Ví dụ về IFRAME

2.2.8 FORMS

Form là thành phần giao tiếp cơ bản giữa người duyệt Web với người tạo Web. Dữ liệu được nhập vào Form thông qua các hộp điều khiển (control).

a. HTML Forms

Người biên soạn HTML có thể tạo ra các HTML Form để tương tác với những người đọc tài liệu của họ chẳng hạn như cho phép người đọc nhập vào dữ liệu để chạy một chương trình CGI, ghi vào các nhận xét về trang Web đó. Các HTML Form có thể chứa các hộp văn bản (textbox), hộp danh sách lựa chọn (checkbox), nút bấm (push button), nút chọn (radio button)...

b. Tạo Form

Để tạo ra một form trong tài liệu HTML, chúng ta sử dụng thẻ FORM với cú pháp như sau:

Cú pháp:

```
<FORM
        ACTION     = ulr
        METHOD     = GET | POST
        NAME       = name
        TARGET     = frame_name  |  _blank  |  _self
        >
        <!-- Các phần tử điều khiển của form -->
        <INPUT ...>
        <INPUT ...>
</FORM>
```

Trong đó

Các tham số	Ý nghĩa
ACTION	Địa chỉ sẽ gửi dữ liệu tới khi form được submit (có thể là địa chỉ tới một chương trình CGI, một trang ASP...).
METHOD	Phương thức gửi dữ liệu.
NAME	Tên của form.
TARGET	Chỉ định cửa sổ sẽ hiển thị kết quả sau khi gửi dữ liệu từ form đến server.

Đặt các đối tượng điều khiển (như hộp văn bản, ô kiểm tra, nút bấm...) vào trang Web

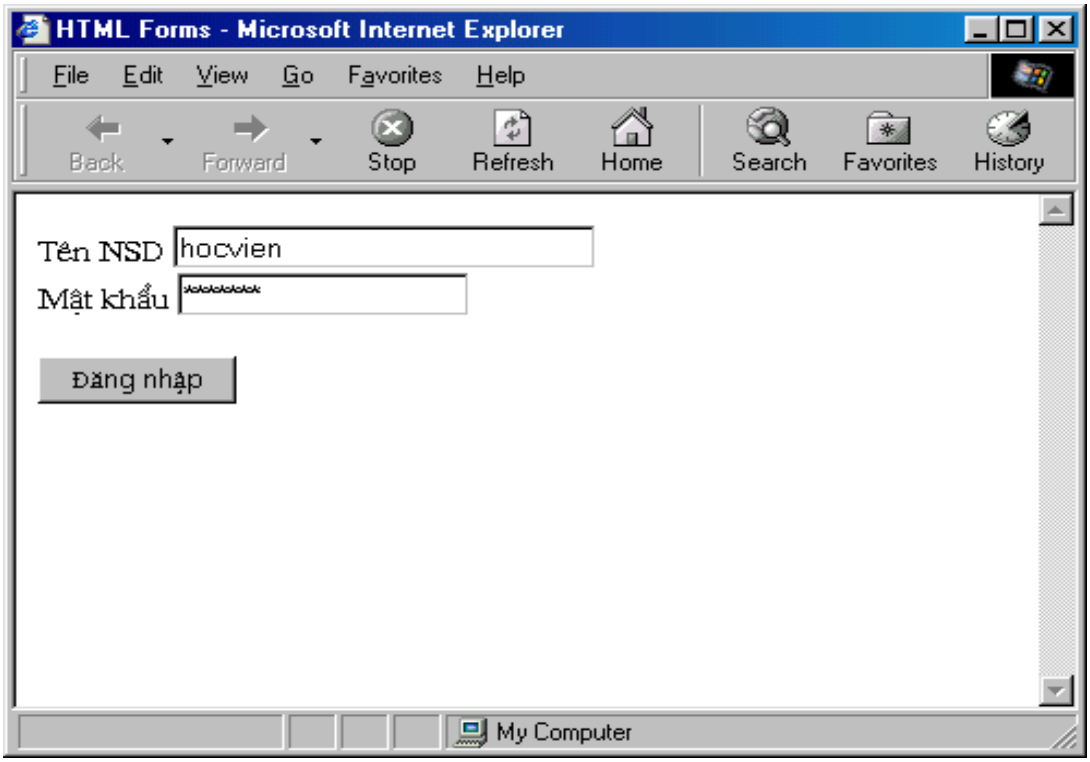
c. Thẻ INPUT

Cú pháp thẻ **INPUT**:

```
<INPUT
ALIGN = LEFT | CENTER | RIGHT
TYPE = BUTTON | CHECKBOX | FILE | IMAGE |
PASSWORD | RADIO | RESET | SUBMIT | TEXT
VALUE = value
>
```

Ví dụ minh họa:

```
<HTML>
<HEAD>
<TITLE>HTML Forms</TITLE>
<BODY>
<FORM METHOD="POST"
 ACTION="http://www.cse.com.vn/scripts/auth.asp">
<P>
Tên NSD <INPUT TYPE="TEXT" SIZE="30"><BR>
Mật khẩu <INPUT TYPE="PASSWORD" SIZE="20">
</P>
<INPUT TYPE="SUBMIT" VALUE="Đăng nhập">
</FORM>
</BODY>
</HTML>
```



Hình 2.4 Ví dụ về thẻ INPUT trong FORM

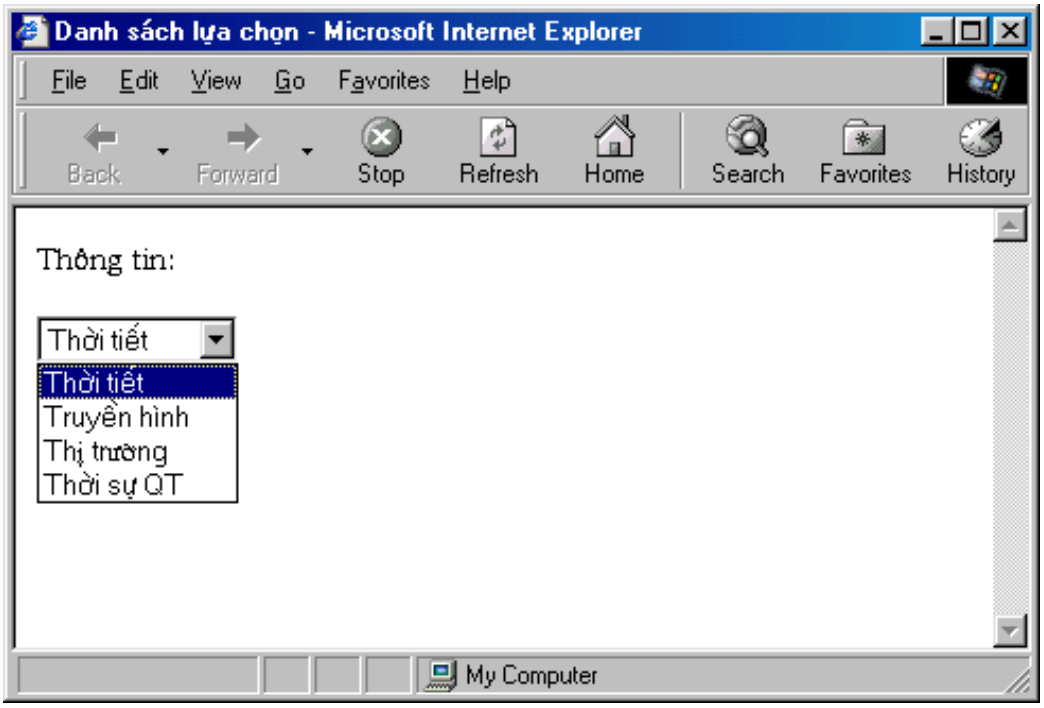
d. Tạo một danh sách lựa chọn bằng thẻ SELECT và OPTION

Cú pháp:

```
<SELECT NAME="tên danh sách"  SIZE="chiều cao">
<OPTION VALUE=value1 SELECTED> Tên mục chọn thứ nhất
<OPTION VALUE=value2> Tên mục chọn thứ hai
<!-- Danh sách các mục chọn -->
</SELECT>
```

Ví dụ minh họa:

```
<HTML>
<HEAD>
<TITLE>Danh sách lựa chọn</TITLE>
</HEAD>
<BODY>
<P>Thông tin:</P>
<SELECT>
<OPTION VALUE="1" SELECTED>Thời tiết
<OPTION VALUE="2">Truyền hình
<OPTION VALUE="3">Thị trường
<OPTION VALUE="4">Thời sự QT
</SELECT>
</BODY>
</HTML>
```



Hình 2.5 Ví dụ tạo một danh sách lựa chọn

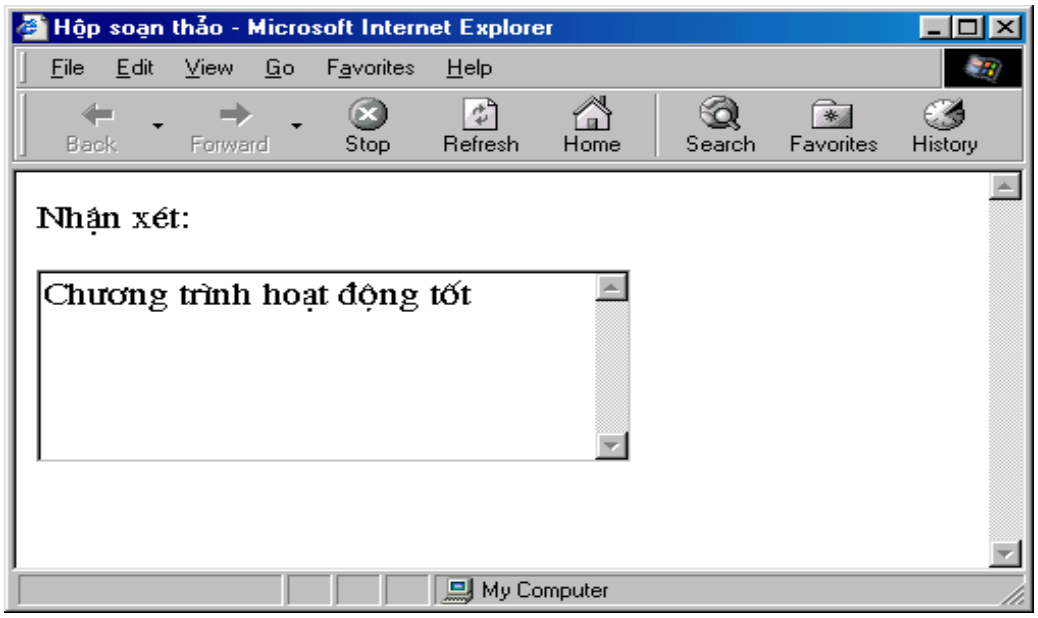
e. Tạo hộp soạn thảo văn bản bằng thẻ TEXTAREA

Cú pháp:

```
<TEXTAREA
        COLS=số cột
        ROWS=số hàng
        NAME=tên
>
        Văn bản ban đầu
</TEXTAREA>
```

Ví dụ minh họa:

```
<HTML>
<HEAD>
<TITLE>Hộp soạn thảo</TITLE>
</HEAD>
<BODY>
<P >Nhận xét:</P>
<TEXTAREA COLS="30" ROWS="5" NAME="nx">
</TEXTAREA>
</BODY>
</HTML>
```



Hình 2.6 Tạo hộp soạn thảo văn bản

2.3 DHTML (Dynamic HTML)

2.3.1 Định nghĩa:

Ngôn ngữ đánh dấu siêu văn bản động (Dynamic Hypertext Markup Language) là phiên bản mở rộng của HTML và JavaScript, ngôn ngữ này được dùng để tạo trang thông tin trên World Wide Web. Dynamic HTML có vị trí văn bản và đồ họa rất chính xác vì nó cho phép nội dung của trang Web thay đổi mỗi khi người dùng nhấn, kéo hay trỏ vào nút, hình ảnh hay các thành phần khác trên trang này.

2.3.2 Đặc điểm

Ngôn ngữ đánh dấu siêu văn bản động mang lại cho các nhà phát triển khả năng tạo những trang Web có hình thức và tính năng như một ứng dụng thực sự.

HTML động cho phép người dùng định vị chính xác văn bản và hình ảnh trên trang Web. Cả hai trình duyệt của Netscape và Microsoft và mới nhất là của Mozilla đều hỗ trợ hệ CSS để kiểm soát vẻ ngoài của trang Web. Ví dụ, các nhà phát triển có thể thay đổi kiểu chữ và kích cỡ của từng dòng tiêu đề trên Web site một cách đơn giản bằng cách thay đổi đặc tả trong trang đơn xác định hình thức

(Cascading Style Sheet) tương ứng. HTML động có cả khả năng liên kết cơ sở dữ liệu với trang Web để sửa đổi nội dung ngay trong khi thực thi. Trước khi trang HTML động được duyệt, trình duyệt phải bổ sung mã chương trình chạy ngoài trình duyệt như Java hay thành phần ActiveX.

2.3.3 *Một số hiệu ứng DHTML*

a. Tạo chuỗi ký tự chuyển động

Cú pháp:

```
<MARQUEE
BEHAVIOR        =type
DIRECTION       =LEFT | RIGHT
LOOP            =n
VSCROLLAMOUNT=n
SCOLLDELAY      =n
>
        Chuỗi ký tự muốn chuyển động
</MARQUEE>
```

Các thuộc tính:

Các tham số	Ý nghĩa
BEHAVIOR	Xác định cách thức chuyển động, với type=scroll thì chuỗi ký tự bắt đầu xuất hiện tại 1 cạnh của cửa sổ màn hình và biến mất ở cạnh bên kia, với type=slide thì chuỗi bắt đầu chuyển động từ 1 cạnh và dừng lại ở cạnh bên kia khi chuỗi chạm vào cạnh kia, với type=alternate thì chuỗi xuất hiện từ bên này sang bên kia và chuyển động ngược lại.
DIRECTION	Định hướng chuyển động cho chuỗi ký tự.
LOOP	Xác định số lần chuyển động của chuỗi. Nếu loop=infinite thì chuỗi sẽ xuất hiện liên tục.
VSCROLLAMOUNT	Xác định tốc độ chuyển động của chuỗi, tính bằng số pixel/giây.
SCOLLDELAY	Thời gian ngừng sau 1 lần chuyển động.

Nếu không có các thuộc tính trên thì chuyển động lặp đi lặp lại từ phải sang trái với tốc độ 6 pixel/1 giây và thời gian ngừng giữa các lần chạy là 90 giây.

Ví dụ minh họa:

```
<HTML>
<HEAD>
<TITLE>Chuỗi ký tự chuyển động</TITLE>
</HEAD>
<BODY>
<P><font color="red" face="tahoma" size="4"></font></P>
<MARQUEE BEHAVIOR=scroll DIRECTION=LEFT
```

```
LOOP=infinite SCROLLAMOUNT=60 SCOLLDELAY=5
>Ví dụ chuỗi ký tự chuyển động trong DHTML
</MARQUEE>
</BODY>
</HTML>
```

b. Thay đổi hình dạng chuột khi qua một vị trí

Cú pháp:

```
<TD STYLE= "cursor:thuộc tính của cursor>
```

Các thuộc tính và hình dạng tương ứng:

Auto	Hand	Move	Text	Wait	Help	Default	Crosshair
I			I				+

Ví dụ minh họa:

```
<HTML>
<HEAD>
<TITLE>Ví dụ về Cursor</TITLE>
</HEAD>
<BODY>
<TABLE BORDER="1">
<TR>
<TD STYLE="cursor:auto">auto</TD>
<TD STYLE="cursor:hand">hand</TD>
<TD STYLE="cursor:move">move</TD>
<TD STYLE="cursor:text">text</TD>
<TD STYLE="cursor:wait">wait</TD>
<TD STYLE="cursor:help">help</TD>
<TD STYLE="cursor:default">default</TD>
<TD STYLE="cursor:crosshair">crosshair</TD>
</TR>
</TABLE>
</BODY>
</HTML>
```

2.4 Câu hỏi và bài tập chương 2

Câu 1: Khái niệm ngôn ngữ HTML?

Câu 2: Nêu các thẻ xác định cấu trúc tài liệu HTML và ý nghĩa của chúng.

Câu 3: Ngôn ngữ đánh dấu siêu văn bản động (DHTML): khái niệm, đặc điểm.

Bài 1: Thiết kế website chương trình đào tạo cử nhân CNTT hệ cao đẳng trường Sĩ quan CH-KT Thông tin, trang gồm 3 frame như hình sau, toàn bộ website sử dụng font Time New Roman, yêu cầu:

- Top frame gồm có logo bên trái, chính giữa là banner của site, dòng slogan chạy bên dưới.

- Left frame là menu chính gồm các mục chính: tin tức, đào tạo, tổ chức, tài liệu, liên kết.
- Main frame dùng thể thể hiện nội dung các mục chọn trong main menu.
- Bottom frame (footer) là thông tin liêm hệ và bản quyền.

<table>
<tr><td>Logo</td><td>Banner</td></tr>
<tr><td>Tin tức</td><td>Slogan chuyển động</td></tr>
<tr><td>Đào tạo</td><td rowspan="4">Contents</td></tr>
<tr><td>Tổ chức</td></tr>
<tr><td>Tài liệu</td></tr>
<tr><td>Liên kết</td></tr>
<tr><td colspan="2">Footer</td></tr>
</table>

Bài 2: Thiết kế website quản lý sinh viên của Hệ Dân sự dưới dạng bảng (table), như sau:

<table>
<tr><td>Logo</td><td colspan="2">Banner</td></tr>
<tr><td>Date-Time</td><td colspan="2">Slogan chuyển động</td></tr>
<tr><td>Tin tức-Sự kiện</td><td rowspan="7">Contents</td><td rowspan="3">Quảng cáo</td></tr>
<tr><td>Thông báo-Hướng dẫn</td></tr>
<tr><td>Tuyển sinh-Đào tạo</td></tr>
<tr><td>Kết quả-Học bổng</td><td rowspan="4">Quảng cáo</td></tr>
<tr><td>Diễn đàn sinh viên</td></tr>
<tr><td>Tài liệu học tập</td></tr>
<tr><td>Liên kết website</td></tr>
</table>

Bài 3: Thiết kế một website theo chủ đề:

- Giới thiệu danh lam thắng cảnh của quê hương.
- Giới thiệu các món ẩm thực đặc sản của địa phương.
- Giới thiệu chức năng, lĩnh vực kinh doanh của một công ty.
- Giới thiệu các ấn phẩm của một nhà sách nhỏ.
- Giới thiệu sản phẩm may mặc của một công ty may xuất khẩu.
- Giới thiệu sản phẩm, hàng hóa cho một shop bán hàng.

Chương 3

NGÔN NGỮ KỊCH BẢN TRONG LẬP TRÌNH WEB

3.1 JavaScript

3.1.1 Tổng quan

Với HTML và Microsoft FrontPage chúng ta đã biết cách tạo ra trang Web - tuy nhiên chỉ mới ở mức biểu diễn thông tin chứ chưa phải là các trang Web động có khả năng đáp ứng các sự kiện từ phía người dùng. Hãng Netscape đã đưa ra ngôn ngữ kịch bản có tên là LiveScript để thực hiện chức năng này. Sau đó ngôn ngữ này được đổi tên thành JavaScript để tận dụng tính đại chúng của ngôn ngữ lập trình Java. Mặc dù có những điểm tương đồng giữa Java và JavaScript, nhưng chúng vẫn là hai ngôn ngữ riêng biệt.

JavaScript là ngôn ngữ dưới dạng script có thể gắn với các file HTML. Nó không được biên dịch mà được trình duyệt diễn dịch. Không giống Java phải chuyển thành các mã dễ biên dịch, trình duyệt đọc JavaScript dưới dạng mã nguồn. Chính vì vậy chúng ta có thể dễ dàng học JavaScript qua ví dụ bởi vì ta có thể thấy cách sử dụng JavaScript trên các trang Web.

a. Đặc tính của ngôn ngữ JavaScript

JavaScript là một ngôn ngữ thông dịch (interpreter), chương trình nguồn của nó được nhúng (embedded) hoặc tích hợp (integated) vào tập tin HTML chuẩn. Khi file được load trong Browser (có support cho JavaScript), Browser sẽ thông dịch các Script và thực hiện các công việc xác định. Chương trình nguồn JavaScript được thông dịch trong trang HTML sau khi toàn bộ trang được load nhưng trước khi trang được hiển thị.

JavaScript là một ngôn ngữ có đặc tính:

- Đơn giản.
- Động (Dynamic).
- Hướng đối tượng (Object Oriented).

b. Đối tượng trong JavaScript

Một trong những đặc tính quan trọng của ngôn ngữ JavaScript là khả năng tạo và sử dụng các đối tượng (Object). Các Object này cho phép người lập trình sử dụng để phát triển ứng dụng.

Trong JavaScript, các Object được nhìn theo 2 khía cạnh:

- Các Object đã tồn tại.
- Các Object do người lập trình xây dựng.

Trong các Object đã tồn tại được chia thành 2 kiểu:

- Các Object của chính JavaScript (JavaScript Built-in Object): JavaScript cung cấp 1 bộ các Built-in Object để cung cấp các thông tin về sự hiện hành của các đối tượng được load trong trang Web và nội dung của nó.Các đối tượng này bao gồm các phương pháp (Method) làm việc với các thuộc tính (Properties) của nó.
- Các đối tượng có sẵn được cung cấp bởi môi trường Netscape: Netscape Navigator cung cấp các đối tượng cho phép JavaScript tương

tác với file HTML, các đối tượng này cho phép chúng ta điều khiển việc hiển thị thông tin và đáp ứng các sự kiện trong môi trường Navigator.

c. Các đối tượng do người lập trình xây dựng

- Định nghĩa thuộc tính của đối tượng: (Object Properties)

Cú pháp

```
Object-name.Property-name (tên đối tượng.tên đặc tính)
```

Ví dụ: Một đối tượng airplane có các thuộc tính như sau:

Airplane.model Airplane.price
Airplane.seating Airplane.maxspeed
Airplane.fuel

- Thêm các phương pháp cho đối tượng: (Method to Object)

Sau khi đã có các thông tin về airplane ta tiếp tục xây dựng phương pháp để sử dụng thông tin này.Ví dụ muốn in ra mô tả của airplane hoặc tính toán khoảng cách tối đa của cuộc hành trình với nhiên liệu đã có:

Airplane.description()
Airplane.distance()

- Tạo một instance (thể hiện) của đối tượng:

Trước khi thao tác với một đối tượng của JavaScript ta phải tạo một instance cho đối tượng đó.

d. Nhúng JavaScript vào trong tập tin HTML

Đoạn mã JavaScript có thể được nhúng vào một file HTML theo một trong các cách sau đây:

- Sử dụng các câu lệnh và các hàm trong cặp thẻ **<SCRIPT>**
- Sử dụng các file nguồn JavaScript.
- Sử dụng một biểu thức JavaScript làm giá trị của một thuộc tính HTML.
- Sử dụng thẻ sự kiện (event handlers) trong một thẻ HTML nào đó.

Trong đó, sử dụng cặp thẻ **<SCRIPT>**...**</SCRIPT>** và nhúng một file nguồn JavaScript là được sử dụng phổ biến nhất.

<u>Sử dụng cặp thẻ <SCRIPT>...</SCRIPT></u>:

Cú pháp:

```
<SCRIPT LANGUAGE="JavaScript">
JavaScript Program
</SCRIPT>
```

<u>Sử dụng tập tin JavaScript bên ngoài</u>:

```
<SCRIPT LANGUAGE="JavaScript"
 SRC="http://www.tsqtt.edu.vn/scroll.js">
<!- - Dòng ẩn mã Script đối với các Browser không hỗ trợ
(support)
JavaScript Program
//Chú thích, tất cả những gì thuộc dòng này đều bị trình
```

```
biên dịch bỏ qua.  Chúng ta cũng có thể sử dụng cặp dấu
/* */ để chú thích cho một đoạn. Dòng kết thúc việc ẩn
mã Script - - >
</SCRIPT>
```

Thuộc tính của thẻ SCRIPT:

- LANGUAGE: Chỉ định ngôn ngữ được sử dụng trong Script và các phiên bản sử dụng (ví dụ như: JavaScript, JavaScript.1.2).
- SRC: Địa chỉ URL chỉ đến tập tin chương trình JavaScript (*.js)

Chú ý: Các file JavaScript bên ngoài không được chứa bất kỳ thẻ HTML nào. Chúng chỉ được chứa các câu lệnh JavaScript và định nghĩa hàm. Tên file của các hàm JavaScript bên ngoài cần có đuôi ***.js***, và server sẽ phải ánh xạ đuôi ***.js*** đó tới kiểu MIME application/x-javascript.

<u>Thêm chương trình vào tập tin HTML</u>:

```
<HTML>
<HEAD>
<TITLE>Listing</TITLE>
</HEAD>
<BODY>
Here is result:
<SCRIPT LANGUAGE="Javascript">
<!--
document.writeln("It work<BR>");
-->
</SCRIPT>
</BODY>
</HTML>
```

3.1.2 *Sử dụng JavaScript*

a. Cú pháp cơ bản của lệnh :

JavaScript xây dựng các hàm,các phát biểu,các toán tử và các biểu thức trên cùng một dòng và kết thúc bằng dấu chấm phẩy (;).

Ví dụ: document.writeln("It work
");

b. Các khối lệnh:

Nhiều dòng lệnh có thể được liên kết với nhau và được bao bởi cặp dấu ngoặc nhọn: { }

Ví dụ:

```
{
document.writeln("Does It work");
document.writeln("It work!");
}
```

c. Xuất dữ liệu ra cửa sổ trình duyệt:

Dùng 2 phương pháp document.write() và document.writeln()

Ví dụ:

document.write(“Test”);
document.writeln(“Test”);

Lệnh document.writeln(“Test”); cho phép xuất từ “Test” ra màn hình và sau đó đưa con trỏ xuống đầu dòng tiếp theo.

d. Xuất các thẻ HTML từ JavaScript

Ví dụ 1:

```
<HTML>
<HEAD>
<TITLE>Outputting Text</TITLE>
</HEAD>
<BODY>
This is text plain <BR>
<B>
<SCRIPT LANGUAGE==“Javascript”>
<!- -
document.write(“This is text bold </B>”);
- ->
</SCRIPT>
</BODY>
</HTML>
```

Ví dụ 2:

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE=“Javascript”>
<!- -
document.write(“<IMG SRC= ‘welcome.gif’>”);
document.write(“<BR><H1>WELCOME TO WEB
DESIGN</H1>”);
- ->
</SCRIPT>
</BODY>
</HTML>
```

e. Sử dụng phương pháp writeln() với thẻ PRE:

```
<HTML>
<HEAD>
<TITLE>Outputting Text</TITLE>
</HEAD>
<BODY>
```

```
<PRE>
<SCRIPT LANGUAGE="Javascript">
<!- -
document.writeln("One,");
document.writeln("Two,");
document.write("Three");
document.write("...");
- ->
</SCRIPT>
</PRE>
</BODY>
</HTML>
```

f. Các ký tự đặc biệt trong chuỗi:

\n : New line
\t : Tab
\r : carriage return
\f : form feed
\b: backspace
Ví dụ:
document.writeln("It work!\n");

g. Làm việc với các hộp hội thoại (dialog boxes):

Sử dụng hàm alert() để hiển thị thông báo trong một hộp.
Ví dụ:

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
<!- -
alert("Welcome to Web Design");
document.write('<IMG
SRC="welcome.gif">');
- ->
</SCRIPT>
</BODY>
</HTML>
```

h. Tương tác với người sử dụng:

Sử dụng phương pháp promt() để tương tác với người sử dụng.
Ví dụ 1:

```
<HTML>
```

```
<HEAD>
<TITLE>Listing</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
<!- -
document.write("Your favorite color is:");
document.writeln(prompt("enter your favorite color:","Blue"));
- ->
</SCRIPT>
</BODY>
</HTML>
```

Ví dụ 2:

```
<HTML>
<HEAD>
<TITLE>Listing</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
<!- -
document.write('<IMG SRC="welcome.gif">');
document.write("<H1>Greeting ,");
document.writeln(prompt("enter your name:","name"));
document.write("Welcome to netscape navigator 2.01 </H1>");
- ->
</SCRIPT>
</BODY>
</HTML>
```

Sử dụng toán tử + để cộng 2 chuỗi đơn lại:

Ví dụ 3:

```
<HTML>
<HEAD>
<TITLE>Listing</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
<!- -
document.write('<IMG
SRC="welcome.gif">');
```

```
document.write("<H1>Greeting ," + prompt("enter your name:","name") +  "Welcome to Web Design </H1>");
- ->
</SCRIPT>
</BODY>
</HTML>
```

3.1.3 *Các kiểu dữ liệu trong JavaScript:*

a. Dữ liệu kiểu số:

- Số nguyên: ví dụ: 720
- Số Octal: ví dụ: 056
- Số Hexa: ví dụ: 0x5F
- Số thập phân: ví dụ: 7.24, -34.2, 2E3

b. Dữ liệu kiểu chuổi:

Ví dụ: “Hello”
‘245’
“”

c. Dữ liệu kiểu Boolean:

Kết quả trả về là true hoặc false.

d. Dữ liệu kiểu null:

Trả về giá trị rỗng.

e. Dữ liệu kiểu văn bản (giống như kiểu chuỗi)

3.1.4 *Tạo biến trong JavaScript:*

Var example;
Var example=”Hello”;
Ta có thể dùng document.write(example); để xuất nội dung của một biến.
Ví dụ 1: Dùng từ khóa var để khai báo biến

```
<HTML>
<HEAD>
<TITLE>Listing</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
<!- -
var name=prompt("enter your name:","name");
- ->
</SCRIPT>
 </HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
<!--
```

```
document.write('<IMG SRC="Welcome.gif">');
document.write("<H1>Greeting ," + name + " Welcome to
Web Design </H1>");
- ->
</SCRIPT>
</BODY>
</HTML>
```

Ví du 2: Tạo lại một giá trị mới cho biến

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
var name=prompt("enter your name:","name");
alert ("greeting " + name + " , ");
name=prompt("enter your friend's name:","friend's  name");
 </SCRIPT>
 </HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
<!--
document.write('<IMG SRC="Welcome.gif">');
document.write("<H1>Greeting ," + name + " Welcome to
Web Design </H1>");
- ->
</SCRIPT>
</BODY>
</HTML>
```

3.1.5 *Làm việc với biến và biểu thức:*

a. Thiết lập biểu thức:

Cú pháp: <biến> <toán tử > <biểu thức>

<u>Toán tử</u>:

= Gán giá trị bên phải cho biến bên trái

Ví dụ: x=5

+= Cộng trái và phải, sau đó gán kết quả cho biến bên trái phép toán

Ví dụ: cho x=10, y=5

x+=y => x=15

-= Trừ bên trái cho bên phải, gán kết quả lại cho biến bên trái

x-=y => x=5

*= Nhân bên trái cho bên phải, gán kết quả cho biến bên trái

x*=y => x=50

/= Chia bên trái cho phải, gán kết quả lại cho biến bên trái
x/=y => x=2
%= Chia bên trái cho bên phải và lấy số dư gán lại cho biến bên trái
x%=y => x=0

Các toán tử khác:

Ví dụ:
x+=15+3 => x=18
8+5
32.5 * 72.3
12 % 5

Dấu ++ và dấu - - và dấu - :

Ví dụ:
x=5;
y=++x; (=> y=6 vì x tăng lên 6 sau đó gán cho y)
z=x++; (=> z=6 vì giá trị x=6 được gán cho z)
sau đó x tăng 1 => x=7
Do đó ta có kết quả cuối cùng là:
x=7; y=6; z=6;
Ví dụ: x=5;
x=-x => x=-5

b. Phép toán Logic

&&: và
||: hoặc
! not
Ví dụ:
x=5, y=2, c=3
(x>y) && (y>c) => false
(x>y) || (c<x) => true
!x

c. Toán tử so sánh trong JavaScript:

==: bằng
!=: khác
>: lớn hơn
<: nhỏ hơn
>=: lớn hơn hoặc bằng
<=: nhỏ hơn hoặc bằng
Ví dụ:
1==1 => true
3<1 => false
5 >=4 => true
“the” != “he” => true
4==“4” => true

d. Toán tử điều kiện:

Cú pháp: (điều kiện) ? giá trị 1 : giá trị 2

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

Ví dụ:

(day="Saturday") ? "Weekend" : "Not Saturday"

e. Toán tử chuỗi:

" Welcome to " + " Web Design"

Ví dụ:

Var welcome="Welcome to"

welcome += " Web Design"

=> welcome = "Welcome to Web Design"

Ví dụ : Sử dụng toán tử điều kiện để kiểm tra đầu vào

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
var question="What is 10+10 ?";
var answer=20;
var correct='<IMG SRC="correct.gif">';
var incorrect='<IMG SRC="incorect.gif">';
var response=prompt(question,"0");
var output = (response==answer) ? correct:incorrect;
</SCRIPT>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
<!--
document.write(output);
- ->
</SCRIPT>
</BODY>
</HTML>
```

3.1.6 *Cấu trúc điều kiện if – else*

Cú pháp:

```
if điều kiện
lệnh ;
hoặc
if điều kiện {
Mã JavaScript
}
```

Ví dụ:

```
If (day=="Saturday") {
document.writeln("It's the weekend");
}
If (day!="Saturday") {
document.writeln("It's not Saturday");
}
```

Sử dụng cấu trúc **else – if** cho ví dụ ở trên

```
If (day=="Saturday") {
document.writeln("It's the weekend");
}
else {
document.writeln("It's not Saturday");
}
```

Cấu trúc kết hợp các lệnh **if** lồng nhau:

```
if  điều kiện 1 {
  Các lệnh JavaScript
   if điều kiện 2 {
        Các lệnh JavaScript
   } else  {
        Các lệnh khác
   }
  Các lệnh JavaScript
} else  {
  Các lệnh khác
}
```

Ví dụ 1 : Sử dụng phương pháp **confirm()** với phát biểu **if**

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
 <SCRIPT LANGUAGE="Javascript">
var question="What is 10+10 ?";
var answer=20;
var correct='<IMG SRC="correct.gif">';
var incorrect='<IMG SRC="incorect.gif">';
 var response=prompt(question,"0");
 if (response != answer) {
   if (confirm("Wrong ! press OK for a second change"))
       response=prompt(question,"0");
}
```

```
 var output = (response ==answer ) ? correct:incorrect ;
</SCRIPT>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
<!--
document.write(output);
-->
</SCRIPT>
</BODY>
</HTML>
```

Ví dụ 2 : Sử dụng phương pháp **confirm()** với phát biểu **if - else**

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
 <SCRIPT LANGUAGE="Javascript">
var question="What is 10+10 ?";
var answer=20;
var correct='<IMG SRC="correct.gif">';
var incorrect='<IMG SRC="incorect.gif">';
var response=prompt(question,"0");
if (response != answer) {
   if (confirm("Wrong ! press OK for a second change"))
       response=prompt(question,"0");
}else {
     if (confirm("Correct ! press OK for a second question"))
        {
          question="What is 10*10";
          answer=100;
          response=prompt(question,"0");
        }
}
var output = (response ==answer ) ? correct:incorrect ;
</SCRIPT>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="Javascript">
<!--
document.write(output);
-->
</SCRIPT>
```

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

3.1.7 *Hàm và dối tượng*

Trong kỹ thuật lập trình các lập trình viên thường sử dụng hàm để thực hiện một đoạn chương trình thể hiện cho một module nào đó để thực hiện một công việc nào đó.

Trong Javascript có các hàm được xây dựng sẵn để giúp chúng ta thực hiện một chức năng nào đó ví dụ như hàm alert(), document.write(), parseInt() và ta cũng có thể định nghĩa ra các hàm khác của mình để thực hiện một công việc nào đó của mình, để định nghĩa hàm chúng ta theo cú pháp sau:

```
function function_name(parameters, arguments)
{
command block;
}
```

a. Truyền tham số:

```
 function printName(name) {
document.write("<HR>Your Name is <B><I>");
document.write(name);
document.write("</B></I><HR>");
}
```

Ví dụ:

Gọi hàm printName() với lệnh sau printName("Bob");

Khi hàm printName() được thi hành giá trị của name là "Bob", nếu gọi hàm printName() với đối số là một biến

```
var user = "John";
printName(user);
```

Khi đó name là "John". Nếu muốn thay đổi giá trị của name ta có thể làm như sau : name = "Mr. " + name;

b. Phạm vi của biến:

Biến toàn cục (Global variable): có giá trị ảnh hưởng trong toàn bộ chương trình.

Biến cục bộ (Local variable): chỉ có giá trị ảnh hưởng trong phạm vi hàm, đoạn mã chứa nó.

c. Trả về các giá trị:

Ví dụ: Dùng return để trả về giá trị của biến.

```
function cube(number) {
var cube = number * number * number;
return cube;
}
```

Ví dụ:

```
<HTML>
```

```
<HEAD>
<TITLE>Example</TITLE>
<SCRIPT LANGUAGE=”JavaScript”>
<!-- HIDE FROM OTHER BROWSERS
//DEFINE FUNCTION testQuestion()
function testQuestion(question) {
//DEFINE LOCAL VARIABLES FOR THE FUNCTION
var answer=eval(question);
var output= “What is ” + question + “?”;
var correct=‘<IMG SRC=“correct.gif”>’;
var incorrect=‘<IMG SRC=“incorrect.gif”>’;
//ASK THE QUESTION
var response=prompt(output,“0”);
//CHECK THE RESULT
return (response == answer) ? correct : incorrect;
}
// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</HEAD>
<BODY>
<SCRIPT LANGUAGE= “JavaScript”>
<!-- HIDE FROM OTHER BROWSERS
//ASK QUESTION AND OUTPUT RESULTS
var result=testQuestion(“10 + 10”);
document.write(result);
//STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</BODY>
</HTML>
```

Hàm eval dùng chuyển đổi giá trị chuỗi số thành giá trị số.
Ví dụ: eval(“10*10”) trả về giá trị là 100.

d. Hàm đệ qui

Ví dụ:

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
<SCRIPT LANGUAGE= “JavaScript”>
<!-- HIDE FROM OTHER BROWSERS
//DEFINE FUNCTION testQuestion()
function testQuestion(question) {
//DEFINE LOCAL VARIABLES FOR THE FUNCTION
```

```
var answer=eval(question);
var output= “What is ” + question + “?”;
var correct= ‘<IMG SRC=“correct.gif”>’;
var incorrect= ‘<IMG SRC=“incorrect.gif”>’;
//ASK THE QUESTION
var response=prompt(output,”0");
//CHECK THE RESULT
return (response == answer) ? correct : testQuestion(question);
}
// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</HEAD>
<BODY>
<SCRIPT LANGUAGE= “JavaScript”>
<!-- HIDE FROM OTHER BROWSERS
//ASK QUESTION AND OUTPUT RESULTS
var result=testQuestion(“10 + 10”);
document.write(result);
//STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</BODY>
</HTML>
```

Ví dụ 2:

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
<SCRIPT LANGUAGE= “JavaScript”>
<!-- HIDE FROM OTHER BROWSERS
//DEFINE FUNCTION testQuestion()
function testQuestion(question,chances) {
//DEFINE LOCAL VARIABLES FOR THE FUNCTION
var answer=eval(question);
var output= “What is ” + question + “?”;
var correct= ‘<IMG SRC=“correct.gif”>’;
var incorrect= ‘<IMG SRC=“incorrect.gif”>’;
//ASK THE QUESTION
var response=prompt(output,“0”);
//CHECK THE RESULT
if (chances > 1) {
return (response == answer) ? correct : testQuestion(question,chances-1);
} else {
```

```
return (response == answer) ? correct : incorrect;
}
}
// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
<!-- HIDE FROM OTHER BROWSERS
//ASK QUESTION AND OUTPUT RESULTS
var result=testQuestion("10 + 10",3);
document.write(result);
//STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</BODY>
</HTML>
```

3.1.8 *Tạo đối tượng trong JavaScript*

a. Định nghĩa thuộc tính của đối tượng:

```
function student(name,age, grade) {
    this.name = name;
    this.age = age;
    this.grade = grade;
}
```

Để tạo một Object ta sử dụng phát biểu ***new***.

Ví dụ để tạo đối tượng **student1**, ta sử dụng khai báo:

student1 = new student("Bob",10,75);

Ba thuộc tính của đối tượng **student1** là :

student1.name, student1.age, student1.grade

Ví dụ để tạo đối tượng **student2**:

student2 = new student("Jane",9,82);

Để thêm thuộc tính cho **student1**, ta có thể làm như sau:

student1.mother = "Susan";

Hoặc chúng ta có thể định nghĩa lại hàm **student**

```
function student(name,age, grade,mother) {
    this.name = name;
    this.age = age;
    this.grade = grade;
    this.mother = mother;
}
```

b. Đối tượng là thuộc tính của đối tượng khác

Ví dụ:

```
function grade (math, english, science) {
this.math = math;
this.english = english;
this.science = science;
}
bobGrade = new grade(75,80,77);
janeGrade = new grade(82,88,75);

student1 = new student("Bob",10,bobGrade);
student2 = new student("Jane",9,janeGrade);
```

student1.grade.math: dùng để lấy điểm Toán của **student1**
student2.grade.science: dùng lấy điểm môn Khoa học của **student2**

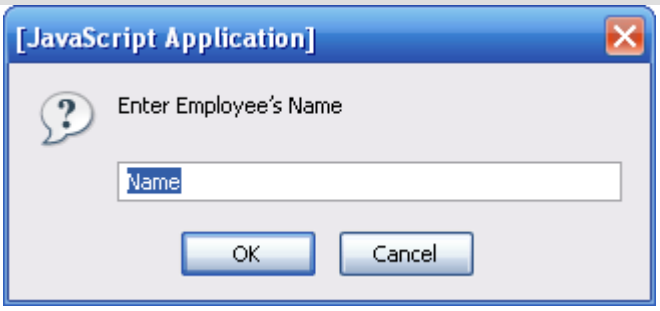
c. Thêm phương thức cho đối tượng:

```
function displayProfile() {
document.write("Name: " + this.name + "<BR>");
document.write("Age: " + this.age + "<BR>");
document.write("Mother's Name: " + this.mother + "<BR>");
document.write("Math Grade: " + this.grade.math + "<BR>");
document.write("English Grade: " + this.grade.english + "<BR>");
document.write("Science Grade: " + this.grade.science + "<BR>");
}
function student(name,age, mother,grade) {
this.name = name;
this.age = age;
this.grade = grade;
this.mother = mother;
this.displayProfile = displayProfile;
}
student1.displayProfile();
```

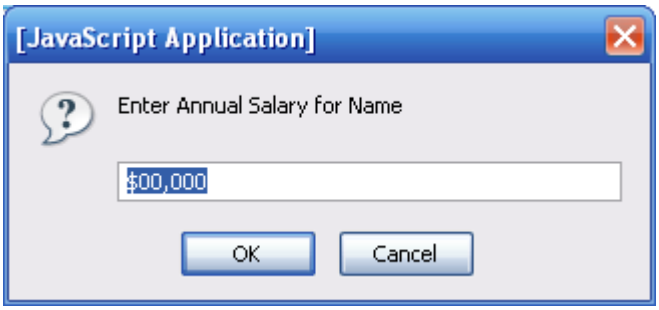
Ví dụ:

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!-- HIDE FROM OTHER BROWSERS
//DEFINE METHOD
function displayInfo() {
document.write("<H1>Employee Profile: " + this.name +
"</H1><HR><PRE>");
document.writeln("Employee Number: " + this.number);
document.writeln("Social Security Number: " + this.socsec);
```

```
document.writeln("Annual Salary: " + this.salary);
document.write("</PRE>");
}
//DEFINE OBJECT
function employee() {
this.name=prompt("Enter Employee's Name","Name");
this.number=prompt("Enter Employee Number for " +
this.name,"000-000");
this.socsec=prompt("Enter Social Security Number for " +
this.name,"000-00-0000");
this.salary=prompt("Enter Annual Salary for " +
this.name,"$00,000");
this.displayInfo=displayInfo;
}
newEmployee=new employee();
// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
<!-- HIDE FROM OTHER BROWSERS
newEmployee.displayInfo();
// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</BODY>
</HTML>
```



Hình 3.1 Form nhập tên nhân viên



Hình 3.2 Form nhập mức lương

Hình 3.3 Form hiển thị kết quả

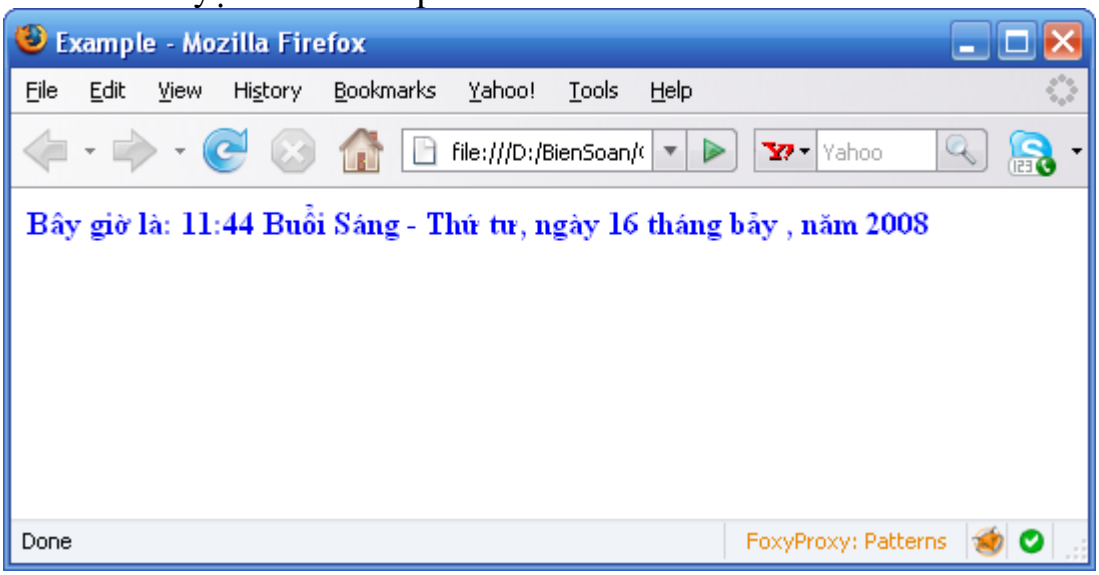
Ví dụ:

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
<script LANGUAGE="JavaScript">
<!-- Begin
var day="";
var month="";
var ampm="";
var ampmhour="";
var myweekday="";
var year="";
mydate = new Date();
myday = mydate.getDay();
mymonth = mydate.getMonth();
myweekday= mydate.getDate();
weekday= myweekday;
myyear= mydate.getYear();
year = myyear;
myhours = mydate.getHours();
ampmhour  =  (myhours > 12) ? myhours - 12 : myhours;
ampm =  (myhours >= 12) ? 'Buổ i Chiề u ' : ' Buổ i Sá ng ';
mytime = mydate.getMinutes();
myminutes =  ((mytime < 10) ? ':0' : ':') + mytime;
if(myday == 0)
    day = " Chủ Nhật , ";
else if(myday == 1)
    day = " Thứ hai, ";
```

```
else if(myday == 2)
    day = " Thứ ba, ";
else if(myday == 3)
    day = " Thứ tư, ";
else if(myday == 4)
    day = " Thứ năm, ";
else if(myday == 5)
    day = " Thứ sáu , ";
else if(myday == 6)
    day = " Thứ bảy , ";
if(mymonth == 0) {
    month = "tháng một ";}
else if(mymonth ==1)
    month = "tháng hai ";
else if(mymonth ==2)
    month = "tháng ba ";
else if(mymonth ==3)
    month = "tháng tư ";
else if(mymonth ==4)
    month = "tháng năm, ";
else if(mymonth ==5)
    month = "tháng sáu ";
else if(mymonth ==6)
    month = "tháng bảy ";
else if(mymonth ==7)
    month = "tháng tám ";
else if(mymonth ==8)
    month = "tháng chín ";
else if(mymonth ==9)
    month = "tháng mười ";
else if(mymonth ==10)
    month = "tháng mười một ";
else if(mymonth ==11)
    month = "tháng mười hai ";
// End -->
</script>
</HEAD>
<!--Trong phan co the xuat ra nhu sau: -->
<BODY>
<SCRIPT>
<!-- HIDE FROM OTHER BROWSERS
```

```
document.write("<b><font color=#0000ff face="tahoma, helvetica, arial'>" + ampmhour + "" + myminutes + ampm)
document.write("  -  " + day + " ngày " + myweekday +"  ");
document.write( month + "  , năm " +  year + "</font>");
// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</BODY>
</HTML>
```

Sau khi duyệt sẽ cho kết quả như sau:



Hình 3.4 Kết quả của đoạn script hiển thị ngày giờ hệ thống

3.1.9 *Sự kiện trong JavaScript*

Các sự kiện cung cấp các tương tác với cửa sổ trình duyệt và tài liệu hiện hành đang được load trong trang web, các hành động của user khi nhập dữ liệu vào form và khi click vào các button trong form.

Khi sử dụng bộ quản lý sự kiện bạn có thể viết các hàm để biểu diễn cho các hành động dựa vào các sự kiện đươc chọn.

a. Bảng sự kiện trong Javascript

Tên sự kiện	Mô tả
Blur	Xảy ra khi điểm tập trung của đầu vào được di chuyển ra khỏi một thành phần của Form (Khi click ra ngoài một trường)
Click	Khi user Click vào 1 link hoặc thành phần của Form.
Change	Xảy ra khi giá trị của Form Field bị thay đổi bởi user.
Focus	Xảy ra khi ngõ vào tập trung vào thành phần của Form
Load	Xảy ra khi một trang được Load vào trong bộ duyệt.
Mouseover	Xảy ra khi User di chuyển mouse qua một Hyperlink.
Select	Xảy ra khi User chọn 1 trường của thành phần Form.
Submit	Xảy ra khi User xác nhận đã nhập xong dữ liệu.
Unload	Xảy ra khi User rời khỏi trang Web.

b. Bộ quản lý sự kiện (Event Handler)

Để quản lý các sự kiện trong javascript ta dùng các bộ quản lý sự kiện.

Cú pháp của một bộ quản lý sự kiện:

<HTML_TAG OTHER_ATTRIBUTES eventHandler=”JavaScript Program”>

Ví dụ:

```
<INPUT TYPE=”text” onChange=”checkField(this)”>
```

Ví dụ:

```
<INPUT TYPE=”text” onChange=“if (parseInt(this.value) <= 5)
{
alert(‘Please enter a number greater than 5.’);
}
“>
```

Ví dụ:

```
<INPUT TYPE=”text” onChange=“
alert(‘Thanks for the entry.’);
confirm(‘Do you want to continue?’);
“>
```

Từ khóa **this**: quy cho đối tượng hiện hành, trong Javascript, Form là một đối tượng. Các thành phần của Form bao gồm text fields, checkboxes, radio buttons, buttons, và selection lists.

c. Các bộ quản lý sự kiện trong Javascript

Đối tượng	Bộ quản lý sự kiện tương ứng.
Selection list	onBlur, onChange, onFocus
Text element	onBlur, onChange, onFocus, onSelect
Textarea element	onBlur, onChange, onFocus, onSelect
Button element	onClick
Checkbox	onClick
Radio button	onClick
Hypertext link	onClick, onMouseOver
Reset button	onClick
Submit button	onClick
Document	onLoad, onUnload
Window	onLoad, onUnload
Form	onSubmit

d. Cách dùng bộ quản lý sự kiện onLoad & onUnload

```
<HTML>
<HEAD>
<TITLE>Example 5.1</TITLE>
</HEAD>
<BODY onLoad=”alert(‘Welcome to my page!’);”
onUnload=”alert(‘Goodbye! Sorry to see you go!’);”>
<IMG SRC=”title.gif”>
```

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

Ví dụ 1:

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
<SCRIPT LANGUAGE=”JavaScript”>
<!-- HIDE FROM OTHER BROWSERS
var name = “”;
// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</HEAD>
<BODY onLoad=”
name = prompt(‘Enter Your Name:’,’Name’);
alert(‘Greetings ‘ + name + ‘, welcome to my page!’);”
onUnload=” alert(Goodbye ‘ + name + ‘, sorry to see you go!’);”>
<IMG SRC=”title.gif”>
</BODY>
</HTML>
```

Ví dụ 2:

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
<SCRIPT LANGUAGE=”JavaScript”>
<!-- HIDE FROM OTHER BROWSERS
// DEFINE GLOBAL VARIABLE
var name = “”;
function hello() {
name = prompt(‘Enter Your Name:’,’Name’);
alert(‘Greetings ‘ + name + ‘, welcome to my page!’);
}
function goodbye() {
alert(Goodbye ‘ + name + ‘, sorry to see you go!’);
}
// STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</HEAD>
<BODY onLoad=”hello();” onUnload=”goodbye();”>
<IMG SRC=”title.gif”>
</BODY>
</HTML>
```

e. Các sự kiện và Form

Các sự kiện được sử dụng để truy xuất Form như: onClick, onSubmit, onFocus, onBlur, và onChange.

Ví dụ 1:

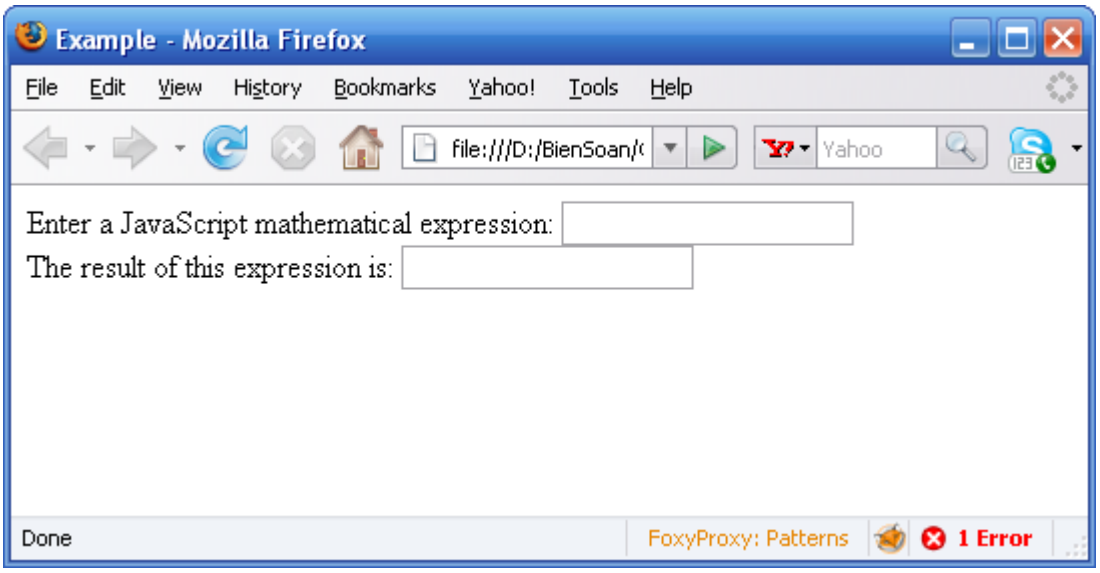
```
<INPUT TYPE=text NAME=”test” VALUE=”test”
onBlur=”alert(‘Thank You!’);”
onChange=”check(this);”>
```

Khi giá trị thay đổi function **check()** sẽ được gọi. Ta dùng từ khóa **this** để chuyển đối tượng của trường hiện hành đến hàm **check()**. Chúng ta cũng có thể dựa vào các phương pháp và các thuộc tính của đối tượng bằng phát biểu sau:

this.methodName() & this.propertyName.

Ví dụ 2:

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
 <SCRIPT LANGUAGE=”JavaScript”>
<!-- HIDE FROM OTHER BROWSERS
function calculate(form) {
form.results.value = eval(form.entry.value);
}
function getExpression(form) {
form.entry.blur();
form.entry.value = prompt(“Please enter a JavaScript
mathematical expression”,””);
calculate(form);
}
//STOP HIDING FROM OTHER BROWSERS -->
</SCRIPT>
</HEAD>
<BODY>
<FORM METHOD=POST>
Enter a JavaScript mathematical expression:
<INPUT TYPE=text NAME=”entry” VALUE=””
onFocus=”getExpression(this.form);”>
<BR>
The result of this expression is:
<INPUT TYPE=text NAME=”results” VALUE=””
onFocus=”this.blur();”>
</FORM>
</BODY>
</HTML>
```



Hình 3.5 Các sự kiện trên form

formObjectName.fieldname: Dùng để chỉ tên trường của hiện hành trong form.
formObjectName.fieldname.value: Dùng lấy giá trị của trường form hiện hành.

3.1.10 *Sử dụng vòng lặp trong JavaScript*

a. Vòng lặp for :

Cú pháp :

```
for ( init value ; condition ; update expression )
```

Ví dụ 1:

```
for  (i = 0 ; i < 5 ; i++)
  {
    lệnh ;
  }
```

Ví dụ 2:

```
<HTML>
<HEAD>
<TITLE> for loop Examle </TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
<!- -
var name=prompt("What is your name?" ,"name");
var query= " " ;
document.write("<H1>" + name + " 's 10 favorite foods </H1> ");
for (var i=1 ;i<=10;i++)
   {
       document.write(i  +  "  "  +  prompt('Enter  food  number  '  +  i,
'food' ) + '<BR>');
   }
```

```
- ->
</SCRIPT>
</BODY>
</HTML>
```

b. Vòng lặp while :

Cú pháp:

```
While ( điều kiện)
   {
      lệnh JavaScript ;
   }
```

Ví dụ 1:

```
var num=1;
while(num<=10)
  {
 document.writeln(num);
num++;
}
```

Ví dụ 2:

```
var  answer=" " ;
var correc=100;
var question=" what is 10*10 ?" ;
while(answer!=correct)
  {
     answer=prompt(question,"0");
  }
```

c. Tạo mảng với vòng lặp for:

```
function createArray(num)
   {
  this.length=num;
  for ( var j=0 ; j<num; j++)
 this[j]=0;
   }
```

Hàm sẽ tạo một mảng có giá trị index bắt đầu là 0 và gán tất cả các giá trị của mảng về 0 .

Để sử dụng đối tượng mảng ta có thể làm như sau:

newArray= new createArray(4);

Sẽ tạo ra một mảng gồm 4 thành phần newArray[0] … newArray[3]

3.1.11 *Sử dụng đối tượng Windows*

Window là đối tượng của môi trường Navigator, ngoài các thuộc tính Window đối tượng window còn giữ các đối tượng khác mà có thể được xem như là các thành phần (member) của window, các đối tượng đó là:

- Các frame đã được tạo
- Các đối tượng location và histtory
- Đối tượng document

Đối tượng document chứa (encompasses) tất cả các thành phần trong trang HTML. Đây là một đối tượng hoàn hảo có các đối tượng khác của JavaScript gán (attached) vào nó (như là anchor, form, history, link). Hầu như mọi chương trình JavaScript đều có sử dụng đối tượng này để tham khảo đến các thành phần trong trang HTML.

a. Các thuộc tính (properties) của đối tượng document

- alink
- anchor
- bgColor
- cookies
- fgColor
- form
- lastModified
- linkColor
- links
- location
- referrer
- title
- vlinkColor

b. Các hành vi (Methods) của đối tượng document

- clear()
- close()
- open()
- write()
- writeln()

c. Các thuộc tính của đối tượng Window

- defaultStatus : Giá trị mặt nhiên được hiển thị ở thanh trạng thái.
- frames: Mảng các đối tượng chứa đựng một mục cho mỗi frame con trong một frame tài liệu.
- parent: Được sử dụng trong FRAMSET
- self: Cửa sổ hiện hành , dùng để phân biệt giữa các cửa sổ hiện hành và các forms có cùng tên.
- status: Giá trị của chuỗi văn bản được hiển thị tại thanh status bar. Dùng để hiển thi các thông báo cho người sử dụng.
- top: Đỉnh cao nhất của cửa sổ cha.
- window

d. Các hành vi (Methods) của đối tượng window

- alert(): Hiện 1 thông báo trong hộp thoại với OK button.

- close(): Đóng cửa sổ hiện hành.
- open(): Mở một cửa sổ mới với 1 tài liệu được chỉ ra hoặc mở một tài liệu trong một tên cửa sổ được chỉ định.
- prompt(): Hiện một hộp thông báo.
- setTimeout():
- clearTimeout(): Hành vi này cung cấp cách gọi phát biểu JavaScript sau một khoảng thời gian trôi qua. Ngoài ra đối tượng window có thể thực hiện event handler: onLoad=statement.

3.1.12 *Làm việc với status bar*

Khi user di chuyển qua một hyperlink ta có thể hiện ra một thông báo tại thanh status bar của browser dựa vào event handler onMouseOver và bằng cách đặt self.status là một chuỗi (hoặc window.status).

Ví dụ:

```
<HTML>
<HEAD>
<TITLE>Status  Example</TITLE>
 <BODY>
<A HREF=”plc.htm” onMouseOver=”self.status=’Chuyen de PLC’;
return true ; “ >
Lop chuyen dề PLC </A>
<A    HREF=”tkweb.htm”    onMouseOver=”self.status=’Thiet    Ke
Trang Web’ ;
return true ; “ >
Thiet Ke Web</A>
</BODY>
</HTML>
```

3.1.13 *Mở và đóng các cửa sổ*

Sử dụng phương thức **open()** và **close()** ta có thể điều khiển việc mở và đóng cửa sổ chứa tài liệu:

open (“URL” , “WindowName” , “featureList”) ;

Các đặc điểm trong phương pháp open() gồm có:

- toolbar: tạo một toolbar chuẩn
- location: tạo một vùng location
- directories: tạo các button thư mục chuẩn
- status: tạo thanh trạng thái.
- menubar: tạo thanh menu tại đỉnh của cửa sổ
- scrollbars: tạo thanh scroll bar
- resizable: cho phép user thay đổi kích thước cửa sổ
- width: chỉ định chiều rộng cửa sổ theo đơn vị pixel
- height: chỉ định chiều cao cửa sổ theo đơn vị pixel

Ví dụ 1:

```
window.open( “plc.htm”, ”newWindow”, ”toolbar=yes, location=1,
```

```
directories=yes, status=yes, menubar=1, scroolbar=yes, resizable=0, copyhistory=1, width=200, height=200”);
```

Ví dụ 2:

```
<HTML>
<HEAD>
<TITLE>WINDOWS</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
function openWindow(url,name) {
popupWin = window.open(url, name, "scrollbars=yes,width=800, heigth=200 ");
}
-->
</SCRIPT>
</HEAD>
<BODY>
<a
href="javascript:openWindow('../chuyende/plc.htm','Win')">PLC</a>,
<a
href="javascript:openWindow('../chuyende/suachuaw.htm','stoogeWin')">Sua chua</a>,
<a
href="javascript:openWindow('../chuyende/tkweb.htm','stoogeWin')">Thiet ke web</a>
</BODY>
</HTML>
```

Để đóng cửa sổ ta có thể dùng phương thức **close()**

Ví dụ:

```
<HTML>
<HEAD>
<TITLE>Close Example</TITLE>
</HEAD>
<BODY >
<A HREF="#" ONCLICK="self.close();return false">
<IMG ALIGN="middle" SRC="../demo.gif" WIDTH="16" HEIGHT="16" BORDER="0"></A>
<A HREF="#" ONCLICK="self.close();return false">Close This Sample</A>
</BODY >
</HTML>
```

3.1.14 *Sử dụng đối tượng string*

String là một đối tượng của JavaScript, khi dùng đối tượng string chúng ta không cần các phát biểu để tạo một instance (thể hiện) của đối tượng, bất kỳ lúc nào ta đặt text giữa hai dấu ngoặc kép và gán nó đến một biến hoặc một thuộc tính thì ta đã tạo một đối tượng string.

a. Các thuộc tính của đối tượng ***string***

Thuộc tính **length** trả về số ký tự (chiều dài) của string.

b. Các phương thức (Methods) của đối tượng ***string***

- anchor (nameAttribute)
- big()
- blink()
- bold()
- charAt(index)
- fixed()
- fontcolor(color)
- fontsize(size)
- indexOf(character,[fromIndex])
- italics()
- lastIndexOf(character,[fromIndex])
- link(URL)
- small()
- strike()
- sub()
- substring(startIndex,endIndex)
- sup()
- toLowerCase()
- toUpperCase()

3.2 VBScript

VBScript là một công nghệ của Microsoft yêu cầu phải có Microsoft Internet Explorer. Trước khi bắt đầu học viết VBScript, chúng ta cần phải biết các khái niệm cơ bản về: WWW, HTML và các kiến thức căn bản để xây dựng một trang web.

3.2.1 *VBScript là gì?*

VBScript là một ngôn ngữ kịch bản. Một ngôn ngữ script là một ngôn ngữ lập trình nhẹ. VBScript là phiên bản nhẹ của ngôn ngữ lập trình Vusual Basic.

Khi VBScript được chèn vào trong văn bản HTML, trình duyệt Internet sẽ đọc văn bản HTML đó và dịch các đoạn mã VBScript. Các đoạn mã này được thực hiện hoặc là ngay lúc đó hoặc trong các sự kiện sau này.

3.2.2 *Biến và phạm vi biến*

Biến là một vùng chứa thông tin cần lưu trữ. Giá trị của biến có thể được thay đổi trong quá trình lập trình. Ta có thể làm việc với một biến thông qua tên của nó, cũng như có thể thay đổi giá trị của biến đó. Trong VBScript, tất cả các biến đều có kiểu là variant, và nó có thể lưu trữ bất kỳ dạng dữ liệu nào.

Quy tắc đặt tên biến: Bắt đầu bằng một chữ cái, không chứa dấu (.) và độ dài không quá 255 ký tự. Chúng ta có thể khai báo biến với các từ khoá Dim, Public hoặc Private.

Ví dụ dưới đây khai báo một biến tên name và gán cho nó một giá trị:

dim name

name = giá trị

Ta cũng có thể khai báo biến bằng cách sử dụng nó trong script của mình.

Ví dụ:

name = giá trị

Tuy vậy, cách khai báo này không được tường minh và không tốt cho ứng dụng của chúng ta, vì sau đó trong ứng dụng của mình, chúng ta có thể vô tình viết sai tên biến và có thể nhận được kết quả không chính xác khi chạy chương trình. Điều đó xảy ra là vì giả sử ta có một tên biến tên "name", sau đó ta gọi tới biến đó bằng một tên "nime" chẳng hạn, chương trình sẽ tự động sinh ra thêm 1 biến tên "nime". Để tránh xảy ra điều nhầm lẫn này, chúng ta nên sử dụng câu lệnh Option Explicit. Khi sử dụng câu lênh này, tất cả các biến đều phải khai báo trước khi sử dụng bởi các câu lệnh với từ khoá Dim, Public hoặc Private. Đặt câu lệnh Option Explicit trên đầu của chương trình, như ví dụ sau:

Option Explicit

dim name

name = giá trị

Cách gán giá trị cho biến:

Ta có thể gán giá trị cho cho một biến như sau:

name = "Nguyễn Minh Phượng"

i = 200 là thời gian sống của biến (Khoảng thời gian biến đó tồn tại được gọi là thời gian sống của nó). Khi khai báo một biến trong một thủ tục, biến đó chỉ được truy xuất tới trong phạm vi thủ tục đó. Khi thủ tục đó kết thúc, các biến đó cũng bị huỷ. Những biến này được gọi là biến cục bộ. Chúng ta có thể đặt các biến cục bộ trùng tên nhau trong các thủ tục khác nhau, bởi vì mỗi biến chỉ được nhận biết bởi chính thủ tục trong đó chúng được khai báo.

Nếu khai báo một biến bên ngoài một thủ tục, tất cả các thủ tục nằm trong cùng trang đó đều có thể truy nhập tới biến đó. Thời gian sống của biến này bắt đầu từ lúc nó được khai báo và kết thúc khi trang web được đóng lại.

Biến Array (mảng): Có những khi chúng ta muốn gán nhiều hơn 1 giá trị cho một biến, khi đó ta khai báo một biến có thể chứa một dãy dữ liệu. Biến này được gọi là biến mảng (array). Để khai báo một biến là biến array, chúng ta đặt dấu ngoặc đơn ngay sau tên biến.

Ví dụ sau chúng ta khai báo một biến array gồm có 3 giá trị:

dim names(2)

Giá trị số trong dấu ngoặc là 2. Chỉ số của biến array bắt đầu bởi 0 cho nên biến này sẽ bao gồm 3 giá trị. Đây là một array có độ dài cố định. Ta gán giá trị cho từng phần tử của array bằng cách sau:

names(0) = "Nguyễn Thanh Bình"

names(1)="Nguyễn Minh Phượng"

names(2)="Hoàng Khánh Hưng"

Tương tự như vậy chúng ta có thể lấy giá trị của bất kỳ phần tử nào trong array mà ta cần bằng cách sử dụng chỉ số tương ứng của phần tử:

eng = names(0)

Chúng ta chỉ có thể khai báo nhiều nhất tới 60 chiều cho một array. Các chiều được khai báo cách nhau bởi dấu phẩy. Ví dụ sau khai báo một array bao gồm 5 dòng và 7 cột:

dim table(4,6)

3.2.3 *Các kiểu dữ liệu*

a. Kiểu dữ liệu trong VBScript là gì?

VBScript chỉ có một kiểu dữ liệu tên là variant. Kiểu variant là một kiểu dữ liệu đặc biệt có thể chứa các loại thông tin khác nhau phụ thuộc vào cách sử dụng chúng. Cũng vì nó là kiểu dữ liệu duy nhất trong VBScript cho nên tất cả các hàm của VBScript đều trả về kiểu dữ liệu này.

Nói một cách đơn giản nhất, một biến variant có thể chứa thông tin là một số hoặc một xâu. Biến variant này xử sự như một số khi nó được sử dụng trong ngữ cảnh số và như một xâu khi sử dụng nó trong ngữ cảnh xâu. Điều đó có nghĩa là nếu ta làm việc với một dữ liệu trông giống kiểu số, VBScript sẽ cho rằng đó là một số và thực hiện tất cả các công việc phù hợp nhất với một số. Tương tự như vậy, nếu ta làm việc với dữ liệu là một xâu, VBScript coi đó là một xâu. Tất nhiên chúng ta hoàn toàn có thể coi dữ liệu số là một xâu bằng cách đặt số đó trong cặp ngoặc kép (“”).

b. Kiểu dữ liệu con của Variant – variant subtypes

Ngoài việc đơn giản là phân biệt số và xâu, một variant có thể phân biệt được thông tin số theo cách khác. Chẳng hạn chúng ta có thể có một dữ liệu số đại diện cho Date/Time. Khi sử dụng nó cùng với một dữ liệu kiểu Date/Time khác thì kết quả trả về luôn được biểu diễn dưới dạng Date/Time. Tất nhiên ta có thể còn có một loạt các dữ liệu dạng số với kích thước khác nhau từ kiểu Boolean cho tới kiểu floating – point. Các dạng thông tin khác nhau đó có thể được lưu trong biến variant gọi là các kiểu con (subtype). Phần lớn thời gian, chúng ta chỉ cần gán dữ liệu của mình vào biến variant và biến này sẽ hoạt động theo cách xử lý dữ liệu giống như chính dữ liệu mà nó chứa.

Bảng dưới đây mô tả các kiểu dữ liệu con của variant:

Subtype	Mô tả
Empty	Variant chưa được gán giá trị ban đầu. Có giá trị 0 đối với các biến kiểu số và xâu rỗng (“”) đối với biến xâu.
Null	Variant không chứa dữ liệu
Boolean	Có giá trị là True hoặc False
Byte	Chứa số nguyên từ 0 tới 255.
Integer	Chứa số nguyên từ -32,768 tới 32,767.
Currency	-922,337,203,685,477.5808 tới 922,337,203,685,477.5807.
Long	Chứa số nguyên từ -2,147,483,648 tới 2,147,483,647.
Single	Chứa số single-precision, floating-point từ -1.402823E38 tới -1.401298E-45 đối với giá trị âm, từ 1.401298E-45 tới 3.402823E38 đối với giá trị dương.

Double	Chứa số double-precision, floating-point - 1.79769313486232E308 to -4.94065645841247E-324 đối với giá trị âm, từ 4.94065645841247E-324 tới 1.79769313486232E308 đối với giá trị dương.
Date (Time)	Chứa một giá trị số đại diện cho ngày tính từ January 1, 100 tới December 31, 9999.
String	Chứa một xâu có độ dài bất kỳ dài nhất khoảng 2 tỷ ký tự
Object	Chứa một Object
Error	Chứa mã số lỗi

Chúng ta có thể dùng các hàm chuyển đổi kiểu dữ liệu để chuyển dữ liệu giữa các kiểu dữ liệu con với nhau. Thêm vào đó, hàm VarType cho ta biết thông tin về cách lưu trữ dữ liệu của mình trong biến Variant.

c. Các hàm trong VBScript:

Dưới đây liệt kê các hàm có sẵn trong VBScript. Các hàm này được chia ra thành các loại sau:

- Các hàm về thời gian
- Các hàm chuyển đổi kiểu dữ liệu
- Các hàm định dạng dữ liệu
- Các hàm toán học
- Các hàm về dãy
- Các hàm về xâu
- Các hàm khác

Các hàm về thời gian (Date/Time Functions)

Tên hàm	Mô tả
Cdate	Chuyển biểu thức có dạng date and time chuẩn sang dạng Date
Date	Trả về ngày giờ hệ thống
DateAdd	Trả về ngày được cộng thêm một khoảng thời gian
DateDiff	Trả về giá trị số là khoảng thời gian giữa hai giá trị ngày.
DatePart	Trả về phần xác định của ngày.
Day	Trả về ngày hiện tại. Giá trị từ 1 tới 31.
FormatDateTime	Trả về biểu thức đã được định dạng theo kiểu date or time
Hour	Trả về giá trị là một số chỉ giờ hiện hành trong ngày, có giá trị từ 0 tới 23.
IsDate	Trả về giá trị Boolean cho biết biểu thức có thể chuyển sang dạng ngày tháng hay không.
Minute	Trả về giá trị số là phút của giờ (có giá trị từ 0 tới 59)
Month	Cho biết tháng hiện hành (Có giá trị từ 1 tới 12)
MonthName	Trả về tên tháng
Now	Cho biết ngày giờ hiện hành của hệ thống
Second	Trả về số đại diện cho giây (Có giá trị từ 0 tới 59)
Time	Trả về giờ hệ thống
Timer	Trả về giá trị số giây tính từ 12:00 AM
Weekday	Trả về số đại diện cho ngày trong tuần (Có giá trị từ 1tới 7)

WeekdayName	Trả về tên ngày trong tuần
Year	Trả về năm hiện hành

Các hàm chuyển kiểu dữ liệu (Conversion Functions)

Tên hàm	Mô tả
Asc	Chuyển ký tự đầu tiên của xâu sang mã ANSI.
CBool	Chuyển dữ liệu kiểu variant sang kiểu subtype Boolean
CByte	Chuyển dữ liệu từ kiểu variant sang kiểu subtype Byte
CCur	Chuyển dữ liệu từ kiểu variant sang kiểu subtype Currency
CDate	Chuyển dữ liệu từ biểu thức dạng date/time sang kiểu subtype Date/Time
CDbl	Chuyển biểu thức từ kiểu variant sang kiểu subtype Double
Chr	Chuyển mã ANSI sang ký tự
CInt	Chuyển dữ liệu kiểu variant sang kiểu subtype Integer
CLng	Chuyển dữ liệu kiểu variant sang kiểu subtype Long
CSng	Chuyển dữ liệu kiểu variant sang kiểu subtype Single
CStr	Chuyển dữ liệu kiểu variant sang kiểu subtype String

Các hàm định dạng dữ liệu (Format Functions)

Tên hàm	Mô tả
FormatCurrency	Trả về biểu thức được định dạng kiểu như currency
FormatDateTime	Trả về biểu thức được định dạng kiểu date or time
FormatNumber	Trả về biểu thức được định dạng kiểu số.
FormatPercent	Trả về biểu thức được định dạng kiểu percentage

Các hàm toán học (Math Functions)

Tên hàm	Mô tả
Abs	Giá trị tuyệt đối của một số
Atn	Trả về cotan của một số
Cos	Giá trị cosine của một số (Góc)
Hex	Cho giá trị hexadecimal của một số
Int	Trả về phần nguyên của một số
Fix	Trả về phần nguyên của một số
Log	Logarit tự nhiên của một số
Oct	Cho giá trị octal của một số
Rnd	Cho một số ngẫu nhiên nhỏ hơn 1 và lớn hơn hoặc bằng 0
Sgn	Trả về một số đại diện cho dấu của số
Sin	Giá trị Sin của một số (Góc)
Sqr	Bình phương của một số
Tan	Giá trị Tang của một số (Góc)

Các hàm về array (Array Functions)

Tên hàm	Mô tả
Array	Trả về một variant chứa một array
IsArray	Trả về giá trị Boolean cho biết biến đó có phải là một array hay không.
Join	Trả về một xâu chứa số các xâu con trong dãy
LBound	Trả về cận dưới của chiếu được chỉ định của một array

Split	Trả về một array 1 chiều chứa một số lượng phần tử được chỉ định.
UBound	Trả về cận trên của chiều được chỉ định của array

Các hàm về xâu (String Functions)

Tên hàm	Mô tả
InStr	Trả về vị trí đầu tiên mà một xâu xuất hiện trong một xâu khác. Tìm kiếm được bắt đầu từ ký tự đầu tiên của xâu
InStrRev	Trả về vị trí đầu tiên mà một xâu xuất hiện trong một xâu khác. Tìm kiếm được bắt đầu từ ký tự cuối cùng của xâu
LCase	Chuyển tất cả các ký tự của một xâu thành chữ thường
Left	Trả về một xâu có độ dài được chỉ định tính từ ký tự đầu tiên
Len	Trả về độ dài của xâu
LTrim	Xoá các ký tự trắng bên trái của xâu
RTrim	Xoá các ký tự trắng bên phải của xâu
Trim	Xoá các ký tự trắng ở cả hai phía của xâu
Mid	Trả về một xâu có độ dài được chỉ định và bắt đầu từ một vị trí được chỉ định của xâu nguồn
Replace	Thay một phần của xâu bởi một xâu khác. Số các lần thay được chỉ định trước.
Right	Trả về một xâu có độ dài được chỉ định tính từ ký tự cuối cùng
Space	Trả về một xâu chỉ gồm toàn dấu cách. Số lượng dấu cách được chỉ định
StrComp	So sánh hai xâu và trả về một giá trị là kết quả của phép so sánh
String	Trả về một xâu có đọ dài được chỉ định và được tạo ra bằng cách lặp đi lặp lại một ký tự nào đó
StrReverse	Trả về một xâu bằng cách quay ngược một xâu có sẵn
UCase	Chuyển tất cả các ký tự của 1 xâu thành chữ hoa

Các hàm khác (Other Functions)

Tên hàm	Mô tả
CreateObject	Tạo một Object có kiểu được chỉ định
Eval	Đánh giá một biểu thức và trả về một giá trị là kết quả của sự đánh giá đó
InputBox	Hiển thị một hộp thoại cho phép người sử dụng có thể điền thông tin vào
IsEmpty	Trả về một giá trị Boolean cho biết một biến đã được gán giá trị hay chưa
IsNull	Kiểm tra xem một biến có là Null (Không chứa dữ liệu) không. Kết quả là một giá trị Boolean
IsNumeric	Trả về một giá trị Boolean cho biết biểu thức đó có thể chuyển thành dạng số không
MsgBox	Hiển thị một hộp tin nhắn và chờ người sử dụng click vào một nút lệnh, và trả về giá trị cho biết người sử dụng đã click nào nút lệnh nào
Round	Làm tròn một số
ScriptEngine	Trả về tên của script đang dùng

TypeName	Trả về tên kiểu dữ liệu con của biến
VarType	Trả về giá trị của kiểu dữ liệu con của biến

d. Các toán tử và biểu thức

VBScript có một tập hợp lớn các loại toán tử, chia ra thành ba loại là các toán tử số học, các toán tử so sánh và ghép nối (concatenation), và các toán tử logic.

Thứ tự ưu tiên của các toán tử

Khi có nhiều toán tử cùng xuất hiện trong một biểu thức, từng phần của biểu thức được đánh giá và xử lý theo một trình tự gọi là thứ tự ưu tiên. Ta có thể dùng dấu ngoặc đơn để thay đổi thứ tự ưu tiên và bắt một phần nào đó của biểu thức phải được thực hiện trước các phần khác. Các biểu thức bên trong dấu ngoặc đơn luôn được xử lý trước những biểu thức bên ngoài. Tất nhiên, nếu biểu thức trong ngoặc chứa nhiều toán tử thì chúng cũng phải tuân theo thứ tự ưu tiên chuẩn.

Khi các biểu thức chứa nhiều loại toán tử khác nhau, các toán tử số học được xử lý trước, sau đó đến các toán tử so sánh rồi cuối cùng là các toán tử logic. Các toán tử so sánh tất cả có cùng thứ tự ưu tiên, tức là chúng sẽ được xủa lý từ trái qua phải theo thứ tự xuất hiện. Các toán tử số học và logic được xử lý theo thứ tự sau:

Số học		So sánh		Logic	
Mô tả	Ký hiệu	Mô tả	Ký hiệu	Mô tả	Ký hiệu
Mũ hoá	^	So sánh bằng	=	Phủ nhận logic	Not
Phép nhân	*	So sánh khác nhau	<>	Và	And
Phép chia	/	Nhỏ hơn	<	Hoặc	Or
Chia lấy phần nguyên	\	Lớn hơn	>	Loại trừ	Xor
Chia lấy số dư	Mod	Nhỏ hơn hoặc bằng	<=	So sánh bằng	Eqv
Phép cộng	+	Lớn hơn hoặc bằng	>=		
Phép trừ	-	So sánh Object tương đương	Is		
Ghép xâu	&				

Khi phép nhân và chia cùng xuất hiện trong một biểu thức, chúng được xử lý từ phải qua trái theo thứ tự xuất hiện. Tương tự như vậy đối với phép cộng và trừ.

Phép ghép xâu không thuộc nhóm toán tử số học nhưng về thứ tự ưu tiên nó đứng sau các toán tử số học và trước các toán tử so sánh. Toán tử Is là một toán tử so sánh việc tham chiếu Object. Nó không dùng để so sánh object hay giá trị của chúng, nó chỉ cho biết xem hai tham chiếu object (object references) có loại hay không.

e. Các cấu trúc điều khiển

Khi viết chương trình, nhiều khi cần thực hiện một hành động nào đó tuỳ thuộc vào một số điều kiện, ta có thể dùng cấu trúc điều kiển để thực hiện điều này.

Trong VBScript có 3 dạng cấu trúc điều khiển:

Câu lệnh if ... then ... else: Sử dụng câu lệnh này khi cần lựa chọn một trong điều kiện để thực hiện một trong hai tập hợp lệnh. Dùng câu lệnh này ta có thể:

- Thực hiện một tập hợp lệnh nào đó nếu điều kiên thoả mãn.

```
if i = 10 then msgbox “Hello”
```

Nếu muốn thực hiện nhiều hơn một câu lệnh khi điều kiện được thoả mãn, chúng ta cần viết từng câu lệnh trên một dòng lệnh khác nhau và kết thúc bởi từ khoá “End If”.

```
if i = 10 then
msgbox “Hello”
i = i + 1
End if
```

- Lựa chọn một trong hai tập hợp lệnh để thực hiện: Nếu muốn thực hiện một tập hợp lệnh nào đó khi điều kiện được thoả mãn và thực hiện một tập hợp lệnh khác nếu điều kiện không thoả mãn, ta dùng như sau:

```
if i = 10 then
msgbox “Hello”
else
msgbox “Goodbye”
End if
```

Câu lệnh if ... then....elseif: Sử dụng câu lệnh này khi muốn lựa chọn một trong nhiều tập hợp lệnh để thực hiện.

```
if payment="Cash" then
  msgbox "You are going to pay cash!"
 elseif payment="Visa" then
  msgbox "You are going to pay with visa."
 elseif payment="AmEx" then
  msgbox "You are going to pay with American Express."
 else
  msgbox "Unknown method of payment."
end If
```

Câu lệnh Select case: Sử dụng câu lệnh này khi muốn lựa chọn một trong nhiều tập hợp lệnh để thực hiện.

```
select case payment
 case "Cash"
  msgbox "You are going to pay cash"
 case "Visa"
  msgbox "You are going to pay with visa"
 case "AmEx"
  msgbox "You are going to pay with American Express"
 case Else
  msgbox "Unknown method of payment"
end select
```

Câu lệnh này làm việc như sau: Đầu tiên chúng ta có một biểu thức, thường

là một biến, cần được đánh giá giá trị. Giá trị của biểu thức này được so sánh với từng giá trị trong cấu trúc Case. Nếu chúng bằng nhau, tập hợp các lệnh tương ứng với giá trị Case đó được thực hiện.

f. Các cấu trúc lặp

Câu lệnh For...Next: Lặp lại việc thực hiện một tập hợp các câu lệnh một số xác định lần. ta có thể sử dụng một biến đếm tăng dần hoặc giảm dần sau mỗi lần thực hiện vòng lặp.

Cú pháp:

```
For i = 1 to 10 step 2
Các lệnh ở đây
Next
```

Từ khoá step chỉ bước nhảy sau mỗi lần thực hiện các câu lệnh trong vòng lặp. Nếu dùng vòng lặp giảm dần thì giá trị của step cần đặt là số âm.

Giá trị ngầm định là 1.

Từ khoá Exit For dùng để nhảy ra khỏi vòng lặp.

Vòng lặp với For Each... Next: Vòng lặp này thực hiện một tập hợp lệnh đối với mỗi phần tử trong tập hợp, hoặc với mỗi phần tử trong một dãy. Câu lệnh này thực hiện không khác nguyên tắc của vòng For...Next, chỉ khác ở chỗ ta không cần chỉ ra số lượng lần muốn thực hiện vòng lặp.

```
dim names(2)
names(0)="Tove"
names(1)="Jani"
names(2)="Hege"
For Each x in names
  document.write(x & "<br />")
Next
```

Cấu trúc Do ... Loop:

- ***Từ khoá While***

Cấu trúc này dùng để thực hiện một tập hợp lệnh khi không biết trước số lần cần thực hiện. Vòng lặp sẽ thực hiện khi điều kiện While vẫn còn được thoã mãn. Sử dụng từ khoá While để kiểm tra điều kiện trong cấu trúc Do...Loop.

```
Do While i>10
  some code
Loop
```

Nếu i = 9 thì các câu lệnh trong cấu trúc này không được thực hiện lần nào. Nhưng nếu thay đổi đoạn mã trên như sau:

```
Do
  some code
Loop While i>10
```

Thì các câu lệnh trong Do...Loop được thực hiện ít nhất một lần ngay cả khi giá trị của i nhỏ hơn 10.

- ***Từ khoá Until***

Lặp lại việc thực hiện các lệnh cho tới khi điều kiện thoả mãn bằng việc sử

dụng từ khoá Until.

Cú pháp:

```
Do Until i=10
  some code
Loop
```

Nếu i = 10, các câu lệnh bên trong vòng lặp không được thực hiện lần nào.

```
Do
  some code
Loop Until i=10
```

Các câu lệnh bên trong vòng lặp được thực hiện ít nhất một lần trong trường hợp ta kiểm tra điều kiện sau.

- ***Từ khoá Exit Do***

Thoát ra khỏi Do...Loop: Dùng lệnh Exit Do để thoát ra khỏi vòng lặp Do...Loop:

```
Do Until i=10
i=i-1
  If i<10 Then Exit Do
Loop
```

Các câu lệnh trong vòng lặp được thực hiện khi khác 10, và khi i lớn hơn 10.

3.3 Câu hỏi và bài tập chương 3

3.3.1 Câu hỏi ôn tập

Câu 1: Khái niệm, đặc tính và cách thức nhúng JavaScript vào HTML.

Câu 2: Các kiểu dữ liệu trong JavaScript?

Câu 3: Các sự kiện trong JavaScript?

Câu 4: Khái niệm, cách thức nhúng VBScript vào tài liệu HTML.

Câu 5: Biến, cách thức đặt tên biến và phạm vi biến trong VBScript?

Câu 6: Các kiểu dữ liệu và các hàm trong VBScript?

3.3.2 Bài tập lập trình với các ngôn ngữ kịch bản

Bài 1: Tạo 2 nút bấm (OK và Cancel), thủ tục xử lý sự kiện khi nhấn nút OK được viết bằng VBScript và hàm sử lý sự kiện khi nhấn nút Cancel được viết bằng JavaScript.

Bài 2: Sử dụng JavaScript và VBScript lần lượt viết chương trình kiểm tra tính hợp lệ của dữ liệu nhập từ form. Nếu thiếu thì thông báo cho người dùng biết, ngược lại thì thông báo "thông tin đac nhập đầy đủ"

Bài 3: Thiết kế form nhập liệu bao gồm các textbox: Họ, tên, quốc tịch (listbox), điện thoại, địa chỉ, giới tính (radiobox), ngày tháng năm sinh (ngày, tháng là listbox), nghề nghiệp (listbox), tên đăng nhập, mật khẩu, nhập lại mật khẩu. Khi nhấn nút "Chấp nhận" thì kiểm tra tính đầy đủ và hợp lệ của dữ liệu, Nếu nhấn "Từ chối" thì reset lại tất cả các hộp nhập dữ liệu. Thực hiện bằng cả VBScript và JavaScript.

Bài 4: Thiết kế một form mô phỏng trang web đăng ký mail của Yahoo, sau khi nhấn Submit thì kiểm tra tính hợp lệ và đầy đủ của dữ liệu. Sử dụng cả VBScript và JavaScript để kiểm tra.

Chương 4

LẬP TRÌNH WEB ĐỘNG VỚI CÔNG NGHỆ ASP

4.1 Một số khái niệm cơ bản về ASP

4.1.1 Khái niệm Web động

Như đã biết ngôn ngữ đánh dấu siêu văn bản HTML là công cụ mô tả trang Web trên Internet. Khi trình duyệt yêu cầu một trang HTML, Web Server nhận yêu cầu và gửi trả lại file HTML được yêu cầu. Trình duyệt sẽ trình diễn trang HTML nhận được.

Nói chung các trang HTML là tĩnh về mặt nội dung. Mặc dù trình duyệt có thể xử lý các ngôn ngữ kịch bản như VBScript hay Jscript nếu như người ta cài đặt các máy ảo tại client để tạo ra một hiệu quả động nào đó với các tương tác hai chiều. Tuy nhiên tương tác này rất hạn chế nếu như dữ liệu cần sử dụng đặt tại server chứ không phải tại client.

Trên thực tế có nhu cầu tra cứu thông tin theo yêu cầu. Ví dụ một siêu thị điện tử, giới thiệu các mặt hàng trên trang Web, và thông tin về các mặt hàng đều được đưa lên đầy đủ. Nếu trang Web này là tĩnh được chuẩn bị trước thì ta không thể lọc ra những thông tin mà mình cần được mà phải duyệt cho tới khi gặp được mặt hàng mà mình quan tâm, nghĩa là phải đợi để thông tin được chuyển về đầy đủ. Vậy nhu cầu về một trang Web có thông tin được chọn lọc theo yêu cầu từ Browse ra đời. Các trang Web này được gọi là trang Web động. Nói một cách đơn giản là các trang Web động là các trang Web không tồn tại sẵn mà chỉ được tạo ra theo yêu cầu của người tra cứu. Trong trường hợp này CSDL Web không phải là tất cả mà còn các CSDL kiểu khác giúp tạo nên các trang Web. Chính vì thế cần đưa vào các trang HTML khả năng tạo Web động dưới dạng các dòng lệnh.

Microsoft quản lý các trang Web bởi IIS (Internet Information Server) trên WebServer. Nhưng IIS không tự tính toán được các dòng lệnh ở phía Server để tạo các trang Web động nên cần có thêm các thành phần khác.

Hiện nay có một số môi trường để tạo các trang Web động, có thể kể đến như: lập trình trên CGI, ASP, PHP, Java, JSP….

4.1.2 ASP là gì?

ASP (Active Server Page) là một thành phần mở rộng của IIS. Khi cài đặt, ASP sinh ra các bộ xử lý ảo đối với ngôn ngữ kịch bản (script engine) tại server để IIS có thể xử lý các mã script mà các mã này có thể viết đan xen trong các trang HTML. Khi Client gọi đến một file .asp trên Web Server, Web Server lập tức gọi đến Script engine để xử lý. Script engine sẽ thực hiện các lệnh script để biến trang ASP thành trang HTML rồi gửi lại Client. Chú ý rằng quá trình này thực hiện tại server chứ không phải tại Client. Vì vậy chúng ta không phải quan tâm tới việc browser xử lý các trang Web như thế nào. Như vậy thực sự quá trình này được thực hiện theo mô hình Client-Server.

ASP là công nghệ Web Server mới của Microsoft, nó được thiết kế để giúp người phát triển ứng dụng trên Web xây dựng các trang Web ứng dụng nhanh chóng và dễ dàng. ASP là một phần tích hợp của công nghệ cơ sở Active (Active Platform), là hạt nhân trong chiến lược internet của Microsoft. Active Platform là một tập hợp các ngôn ngữ, các chuẩn và các dịch vụ có thể được sử dụng để phát

triển cả ứng dụng Active Desktop(bản Client) và Active Server (bản Server) trong mô hình CSDL tính toán Client / Server. Mô hình Active Platform giúp cho người phát triển ứng dụng xây dựng ứng dụng hiệu quả về giá thành, mở rộng khả năng cảu các ứng dụng chạy trên Server cũng như chạy trên Client và nâng cao kỹ năng phát triển ứng dụng của họ. Đồng thời, nó cũng làm việc chuyển đổi từ ứng dụng Desktop sang ứng dụng Client/Server đầy đủ, dễ dàng.

4.1.3 *Scripting?*

Scripting là một đoạn chương trình mà chúng ta chèn vào các trang HTML để tạo tính "động" cho nó. Scripting dùng ngôn ngữ, cú pháp và cách thực hiện riêng. Tuy nhiên, có một vấn đề nảy sinh ở đây: Mỗi một hãng cung cấp lại định nghĩa một ngôn ngữ script khác nhau. Microsoft phát triển Visual Basic Script (VBScript), Sun Microsystem và Nescape phát triển JavaScript (JScript) và một số hãng khác hỗ trợ những ngôn ngữ như : Perl, Python, Awk ... .

a. Scripting trên Client:

Scripting trên Client có thể được chèn vào trang HTML bằng cặp tags <Script> ... </Script>.

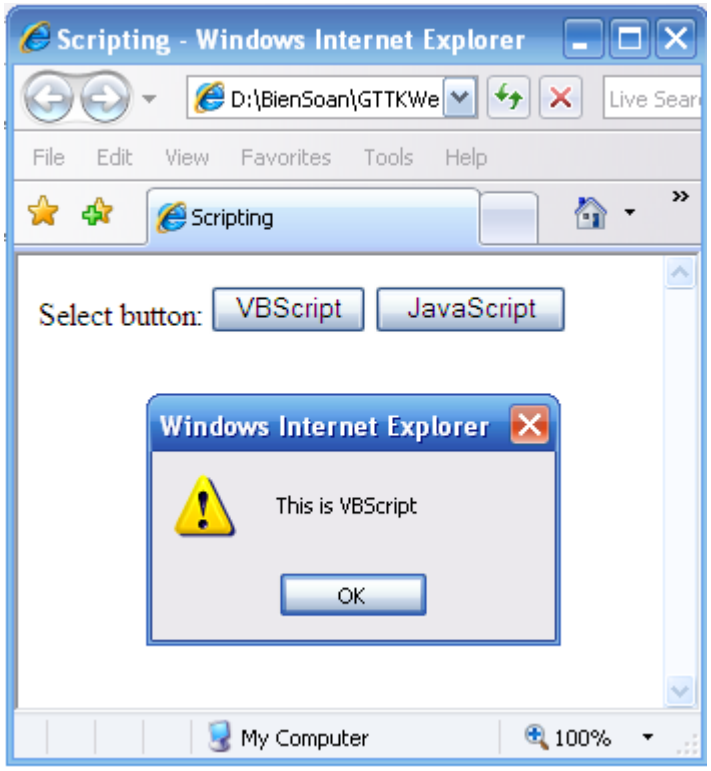
Để xác định ngôn ngữ Script ta dùng thuộc tính LANGUAGE.

Ví dụ sau sẽ minh họa sự kiện xảy ra khi người dùng nhấn vào một nút. Chức năng của nó được chỉ ra bằng thuộc tính ONCLICK. Scripting trên Client có thể làm việc trên bất kì máy chủ nào hỗ trợ ngôn ngữ của scripting .

Ví dụ: Dùng Scripting trên Client.

```
<HTML>
<HEAD>
<TITLE> Scripting </TITLE>
<SCRIPT LANGUAGE="VBSCRIPT">
sub vbs()
        alert("This is VBScript")
end sub
</SCRIPT>
<SCRIPT LANGUAGE="JAVASCRIPT">
function js(){
        alert("This is JavaScript")
}
</SCRIPT>
</HEAD>
<BODY>
Select button:
<INPUT TYPE="button" NAME="vbs" VALUE="VBScript"
ONCLICK="vbs()">
<INPUT TYPE= "button" NAME= "js" VALUE= "JavaScript" ONCLICK=
"js()">
</BODY>
</HTML>
```

Nhấn vào một nút, chương trình phù hợp sẽ được chạy :



Hình 4.1 Hoạt động của VBScript và JavaScript

b. Scripting trên Server:

ASP sử dụng Scripting trên Server để tự động tạo ra những trang trả lời . Nội dung sinh ra dựa trên những thông số của User được gửi tới cùng yêu cầu và sự t-ương tác giữa các đối tượng khác nhau . Ngoài ra chúng ta còn có thể sử dụng một số Object và Component do ASP cung cấp. Các Object làm đơn giản hoá một số công việc trên Server.

Scripting trên Server được chèn vào một file ASP bằng cách sử dụng cặp tags <SCRIPT> ... </SCRIPT> hoặc <% và %> . Để phân biệt Scripting được viết trên Client hay trên Server ta sử dụng thuộc tính RUNAT =“SERVER” .

Đối với hầu hết các browser thì ngôn ngữ Script mặc định là JavaScript. Bởi vậy, nếu chúng ta viết các mã lệnh bằng VBScript thì chúng ta phải khai báo với browse trớc khi sử dụng :

<SCRIPT LANGUAGE= “VBSCRIPT”>

Ngược lại với browser, ngôn ngữ script mặc định của ASP là VBScript. Nếu chúng ta dùng JavaScript thì khai báo lại nh sau (chú ý thuộc tính RUNAT):

<SCRIPT LANGUAGE=“JCRIPT” RUNAT=“SERVER”>

Tùy theo khả năng và sở thích, chúng ta có thể sử dụng một trong hai ngôn ngữ trên để lập trình. Tuy nhiên, có một chú ý quan trọng là hiện nay, JScript (ECMA Script) là ngôn ngữ Scripting chuẩn được Hiệp hội sản xuất máy tính Châu Âu (European Computer Manufactury Association) công nhận.

4.1.4 *Tạo và xem một file ASP*

Chúng ta sử dụng những công cụ của Microsoft như: Microsoft Frontpage, Microsoft’s Visual InterDev để tạo một file ASP .

Để xem một file ASP, chúng ta không thể gửi nó đến Browser như một trang

HTML bởi vì Browser không nhận biết được các file ASP. Lý do là các file này cần phải được thông dịch trên Server trước khi gửi ra Browser.

Chúng ta có thể sử dụng IE hoặc Nescape Navigator để xem kết quả của các file ASP nhưng chúng ta cần chắc chắn rằng Server mà chúng ta xử lý các mã có cài đặt ASP và đang chạy IIS hoặc Personal Web Server (PWS).

Trang ASP là một dạng text có kiểu là .asp, có cấu trúc gần giống như file HTML. Tất cả các thẻ có trong HTML thì đều dùng được trong ASP. Nhưng ngoài các thẻ thông thường của HTML, trong file asp còn có thể viết các thẻ khác nữa để thể hiện các dòng lệnh của Script để làm việc với dữ liệu có sẵn hay có thể tính toán ngay bên trong như là một ngôn ngữ lập trình thực sự.

Cơ chế hoạt động của ASP như sau:

Client sử dụng một Web Browser gửi yêu cầu HTTP tới một Server chạy Microsoft Internet Information Server (IIS). Sau khi nhận biết đó là yêu cầu cần xử lý của trang ASP, IIS chuyển yêu cầu này tới ASP engine, tại đây nội dung file .asp được xử lý, các mã HTML được để nguyên còn các Script được tính dựa theo các yêu cầu và được chuyển đổi thành các mã HTML. Nếu trong các Script có các câu lệnh gọi dữ liệu, nó sẽ liên kết tới Database Server và lấy các dữ liệu theo yêu cầu. Sau đó, các kết quả của việc thực hiện các Script (có thể là HTML tĩnh hoặc động) được gửi trả lại Client Browser.

Xét một ví dụ để làm rõ cơ chế này:

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
</HEAD>
<BODY>
<% For i=5 to 7%>
  <FONT SIZE=<%=i%>>Chào bạn!<BR></FONT>
<% Next %>
</BODY>
</HTML>
```

Kết quả là trình duyệt sẽ hiển thị 3 dòng Chào bạn! với kích thước tăng dần như sau

Chào bạn!

Chào bạn!

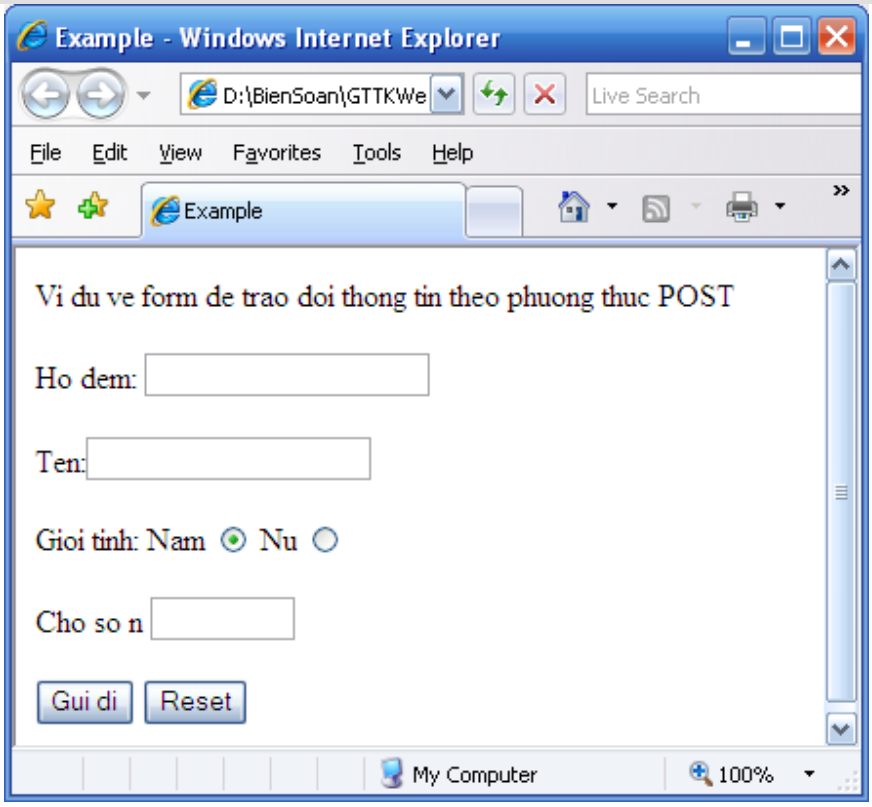
Chào bạn!

Trong ví dụ này chúng ta mới đề cập đến nguyên lý làm việc của ASP mà chưa nói đến cơ chế tạo Web động tức là cách tạo các trang Web thay đổi theo thông tin tương tác mà người sử dụng cung cấp. Trong ví dụ ta sẽ làm việc như sau:

Tạo một trang Web có một form để người sử dụng nhập vào họ, tên và giới tính qua Textbox và option. Người sử dụng cũng đánh vào một số tự nhiên. Khi nhận được, ASP sẽ gửi lại một trang Web với một lời chào phù hợp với giới tính và tên người đã được cung cấp đồng thời cho ra tất cả các ước số của số này.

Sau đây là đoạn mã của hai trang ASP thực hiện hiệu ứng này. Có một số câu lệnh script có thể còn lạ nhưng ý nghĩa hoàn toàn rõ ràng.

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
</HEAD>
<BODY>
<P>Vi du ve form de trao doi thong tin theo phuong thuc POST</P>
<FORM method="post" action="chao.asp">
<P> Ho dem: <input type ="text" name="Hodem" size=20></P>
<P>Ten:<input type ="text" name="Ten" size=20></P>
<P> Gioi tinh: Nam <input type ="radio" value="Nam" checked
name="Gioitinh" size=20>
 Nu <input type ="radio" value ="Nu" name="Gioitinh" size=20></P>
<P>Cho so n <input type ="Text" name="So" size=8></P>
<P><input type="submit" value ="Gui di" name="B1">
<input type="reset"  name="B2"></P>
</FORM>
</BODY>
</HTML>
```



Hình 4.2 Form sử dụng ASP

Đây là nội dung tệp Chao.asp mô tả ứng xử khi ta bấm nút Gửi đi tương ứng với hoạt động Submit của Form

```
<HTML>
<HEAD>
<TITLE>Example</TITLE>
```

```
</HEAD>
<BODY>
<% ho=request.Form("hodem")
ten=request.Form("ten")
so=request.Form("so")
gioitinh=request.form("gioitinh")
if gioitinh="Nam" then
 gioitinh="Ong"
else
 gioitinh="Ba"
end if
response.Write "Xin chao " & gioitinh & " " & ho &" " & ten &" <br>"
response.Write "Day la ket qua tinh " & "<br>"
for i=1 to so-1
 if so mod i =0 then
  response.Write i & "<br>"
 end if
next
%>
</BODY>
</HTML>
```

4.1.5 *Server-side Includes:*

Server-side Includes (SSI) là một thuật ngữ được sử dụng để mô tả cách thức các yếu tố khác nhau được chèn vào trang Web

a. Gắn những file text vào một trang với #include:

Chúng ta có thể gắn file text GetLastDay.txt (vốn là một file ASP, được save với tên trên, có function có chức năng lấy lại ngày cuối cùng trong một tháng) vào một trang Web bằng cách thêm câu lệnh trên vào trang và gọi chức năng:

```
<!-- #include file="GetLastDay.txt" -->
...
intLastDayAugust = GetLastDay(datAugust) ‘ chức năng chúng ta gắn vào
...
```

Nếu muốn gắn Script từ các file khác, file này phải chứa những phần Script hoàn chỉnh . Nói một cách khác, nó phải có đủ những tag <SCRIPT> ... </SCRIPT> hoặc <% ... %>.

b. Địa chỉ vật lý, địa chỉ ảo của file:

#include cho phép chúng ta chỉ đến một file bằng đờng dẫn vật lý hoặc đờng dẫn ảo.Ví dụ file Mytext.txt nằm trong th mục c:\TextFile và cũng có bí danh (alias) là /Text, ta có thể tham khảo tới nó bằng những cách sau:

```
<!-- #include file="C:\TextFile\MyFile.txt" --> ‘ đường dẫn vật lý
<!-- #include file="/Text/MyFile.txt" -->          ‘ đường dẫn ảo
```

4.2 Ưu điểm của việc sử dụng ASP tạo Web động

4.2.1 *Đơn giản, dễ học và hiệu quả:*

Học và phát triển ASP là rất dễ dàng. Ta có thể sử dụng ASP để xây dựng một Web site có khả năng tương tác cao. Vì các ngôn ngữ kịch bản như VBScript, Jscript được tích hợp trong ASP nên rất tiện cho người phát triển đã biết ngôn ngữ VB, Java hay C++, còn đối với người chưa biết thì việc học nó cũng dễ dàng.

Các ứng dụng ASP không cần có trình biên dịch. Trong một vài công nghệ khác như CGI, để phát triển các trang Web động cần phải có một trình biên dịch để dịch thành một chương trình có thể chạy được sử dụng các môi trường phát triển ứng dụng truyền thống như Visual C++. Sau khi ứng dụng được dịch, nó sẽ được copy vào thư mục CGI của Web Server. Chỉ cần có một chút sửa đổi chương trình thì ta phải dịch lại mã nguồn của chương trình và sau đó lại phải copy đè lên phiên bản trước của file chạy. ASP giải quyết vấn đề này bằng cách cung cấp các cách tạo lập trang Web một cách trực tiếp và dễ dàn hơn theo kiểu thông dịch(interpreter). Sau khi xây dựng xong một ứng dụng Web bằng ASP, ta không cần phải dịch chùng mà chỉ cần lưu giữ vào một file có kiểu là .asp và các ASP sẽ xử lý khi file này được gọi đến.

Ngoài các thành phần ASP sẵn có giúp chúng ta xây dựng rất nhiều ứng dụng động khác, ASP cũng cho phép ta tự tạo ra các thành phần ASP của riêng mình.

4.2.2 *Bảo mật được mã:*

Một điểm bất lợi trong việc sử dụng ngôn ngữ kịch bản Client là phơi bày tất cả các thông tin và thuật giải của bài toán. Nếu một bài toán sử dụng ngôn ngữ kịch bản tại Client như VBSCript thì bất kỳ ai nhìn vào mã nguồn của trang Web đều có thể thấy được thuật toán của nó.

Với ASP tất cả các Script được thực hiện trên Server và chỉ có kết quả ra dưới dạng HTML được gửi về Browser nên nếu người dùng muốn xem mã nguồn của trang Web thì họ chỉ xem được mã HTML chứ không xem được mã Script đã tạo nên trang Web đó. Như vậy nếu sử dụng ASP thì NSD không thể biết được thuật toán của nhà phát triển vì các mã ASP được thực hiện trên Server. ASP bảo vệ sự sở hữu về thông tin và thuật toán.

4.2.3 *Bảo trì dễ dàng:*

Môi trường phát triển ASP giúp nâng cao hiệu quả sử dụng của các thiết bị sẵn có. Môi trường phát triển ASP giúp cho người phát triển sử dụng một cách dễ dàng và có hiệu quả các kỹ năng sẵn có. ASP cung cấp một cơ cấu thiết lập các trang Web phức tạp sử dụng ngôn ngữ kịch bản quen thuộc như VBScript, Jscript/JavaScript hay Perl.

Đối với nhiều phần mềm Client/Server khác, ngoài phần được viết trên Server, còn cần phần viết trên Client. Như vậy độ phức tập và tốn kém về mặt lập trình sẽ tăng lên. Khi viết bằng ASP thì chỉ cần có trình duyệt Web tại máy Client, sau đó nối đến máy chủ, như thế việc trên Client không còn gì phả quan tâm. Mỗi khi cần sửa chữa hoặc nâng cấp không cần phải làm gì với bản Client.

Thông qua các câu lệnh Script, ta có thể kết nối đến với một CSDL tại một Database Server. Để làm việc trên CSDL này, ta có thể nhúng ngôn ngữ truy vấn SQL. Thông qua đó việc lọc dữ liệu đơn giản, công việc lọc dữ liệu được tiến hành trên Server nên tránh được ách tắc đường truyền.

4.3 Cài đặt IIS và tạo thư mục ảo cho ứng dụng

4.3.1 *1. Cài đặt IIS*

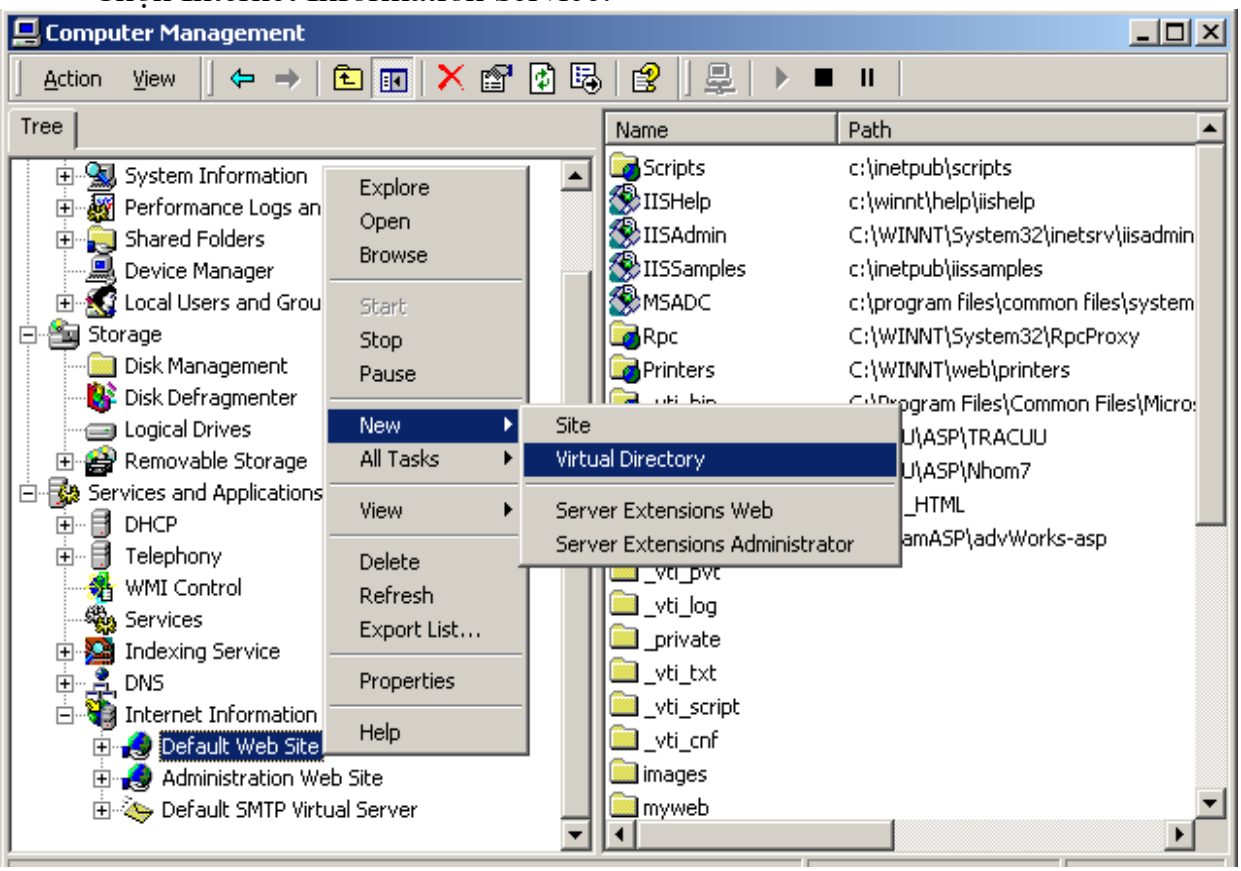
Internet Information Services mặc định không được cài đặt trên hệ điều hành Windows XP Professional. Ta có thể gỡ bỏ hoặc thêm các components bằng việc sử dụng chương trình ứng dụng Add/Remove Programs trong tiện ích Control Panel. Các bước cài đặt tuần tự như sau:

- Click **Start**, click **Control Panel**, và double-click **Add/Remove Programs**.
- Trong cột bên trái của hộp thoại **Add/Remove Programs** dialog box, click **Add/Remove Windows Components**.
- Khi cửa sổ Windows Components Wizard xuất hiện, click **Next**.
- Trong danh sách các thành phần của Windows(Windows Components), chọn **IIS**.
- Click **Next**, và làm theo các chỉ dẫn của Wizard.

4.3.2 *Tạo thư mục ảo:*

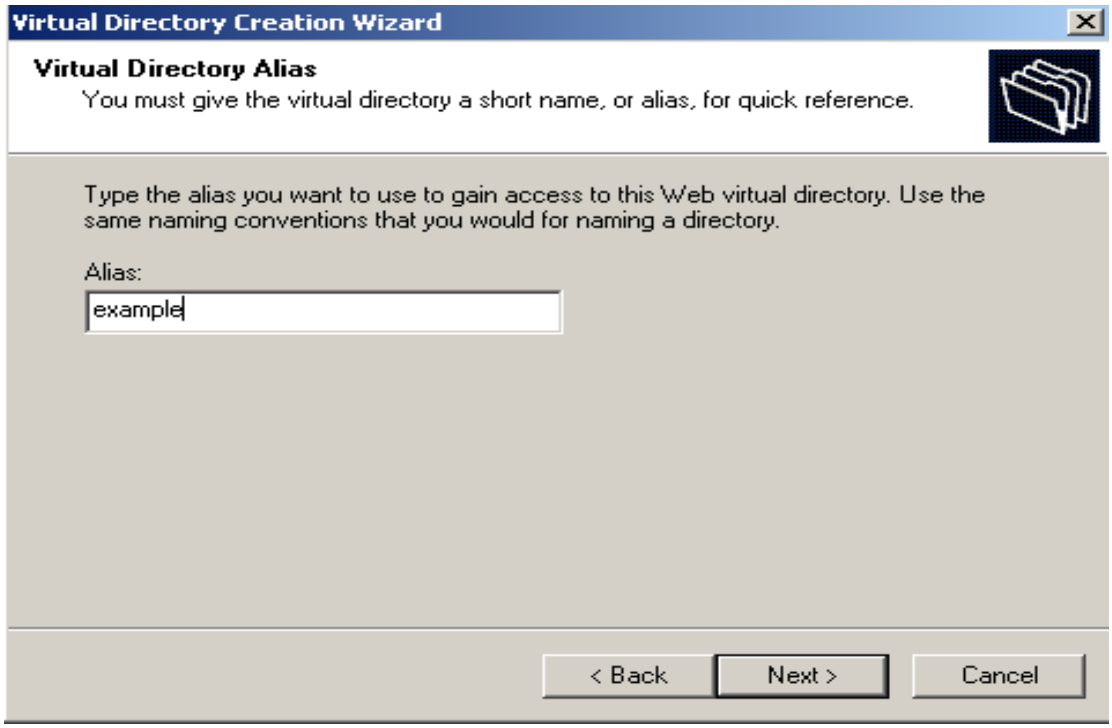
Chúng ta có thể tạo các thư mục ảo bên dưới ứng dụng Default Web site. Thông thường một ứng dụng Web được đặt trong một thư mục ảo và được tham chiếu đến thông qua địa chỉ URL.

Chọn Internet Information Service:



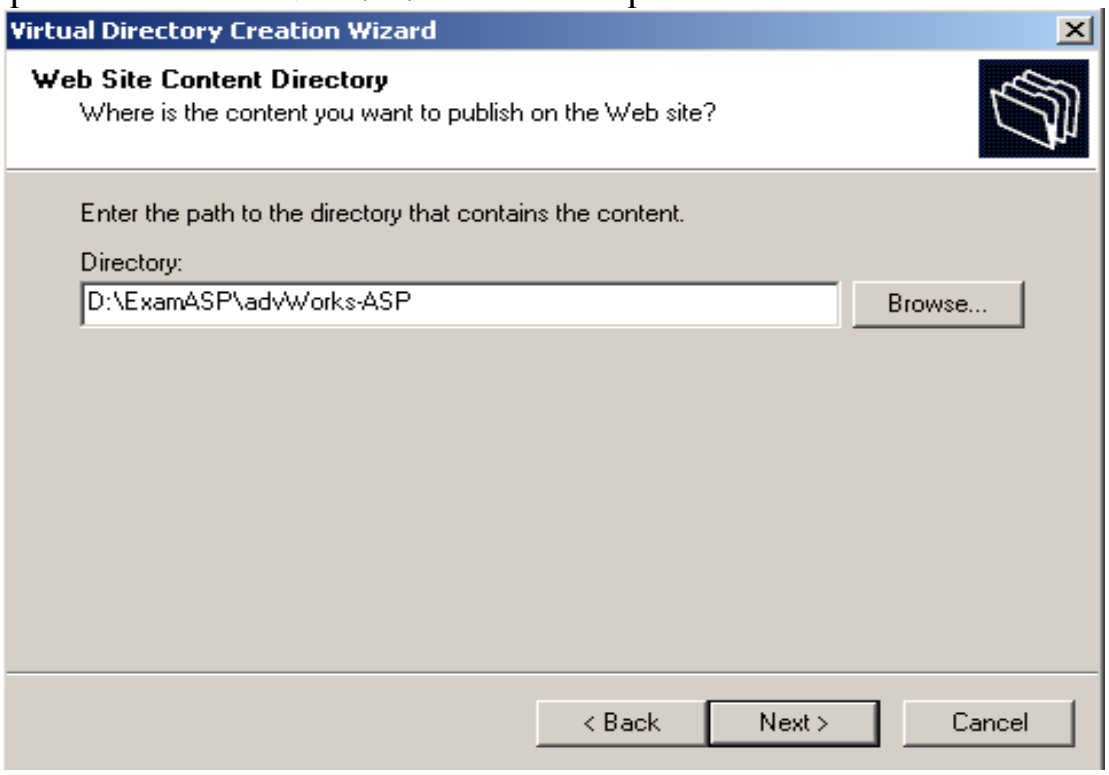
Hình 4.3 Tạo thư mục ảo trong IIS

Nhấn chuột phải vào Default Web Site, chọn New/Vitual Directory, cửa sổ trợ giúp Wizard sẽ hiện ra như sau:



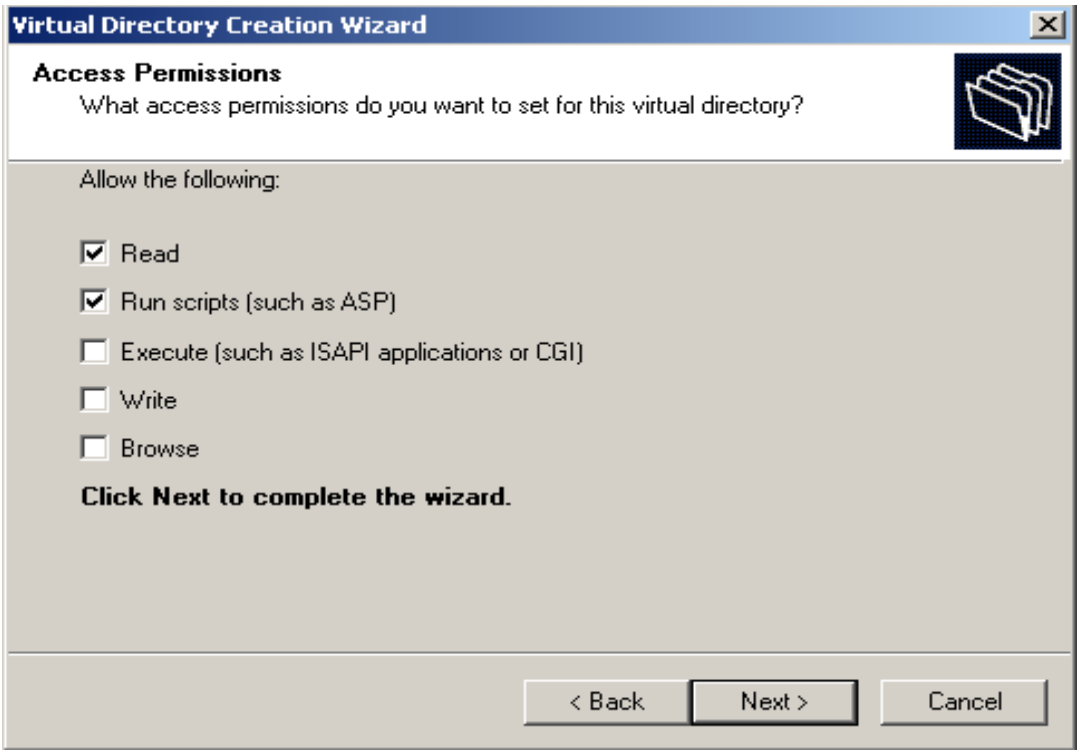
Hình 4.4 Nhập bí danh (Alias) cho the mục ảo

Trong ô nhập liệu Alias, nhập vào tên bí danh cho thư mục ảo, chẳng hạn example. Kích vào nút Next để đến bước kế tiếp.



Hình 4.5 Chọn đường dẫn vật lý cho thư mục ảo

Sau khi chọn đường dẫn vật lý cho thư mục ảo, ta tiến hành thiết lập quyền truy cập cho thư mục ảo. Hai quyền Read và Run Script là cần thiết để trang ASP có thể truy xuất được, ngoài ra ta có thể thiết lập các quyền khác cho người quản trị site (Administrator).



Hình 4.6 Thiết lập quyền truy cập cho thư mục ảo.

4.4 Cấu trúc và các dòng lệnh cơ bản của ASP

4.4.1 Các thành phần được dùng trong trang ASP

File ASP là một file dưới dạng Text, ta có thể sử dụng bất cứ trình soạn thảo văn bản dưới dạng text only để soạn thảo ra file ASP. File ASP có phần mở rộng là .asp . Trong file ASP có thể có:

- Các mã HTML.
- Các ký hiệu phấn cách Script.
- Các mã Script.
- Các thànhh phần ActiveX.
- Các đối tượng ASP.

Điểm khác biệt cơ bản giữa file ASP với file HTML là sự có mặt các dấu phân chách các mã Script với các mã HTML. Trong file ASP nếu ta viết hướng dẫn <%<lệnh>%> thì ASP hiểu rằng lệnh bên trong hai dấu <% và %> là một lệnh Script. Như đã nói ở trên, các lệnh Script có thể là VBScript hay Jscript.

4.4.2 Biến trong ASP

Biến được cho bằng một tên gọi nào đó (quy cách giống như biến dùng trong Visual Basic) – có thể tra cứu các tài liệu của VB để biết thêm nguyên tắc đặt tên biến. Khi sử dụng biến trong Script, ta không cần phải khai báo trước mà có thể sử dụng trực tiếp. Các biến trong ASP không có kiểu, kiểu của nó sẽ được xác định một cách tự động khi có lệnh gán giá trị vào biến. Nếu có khai báo biến thì cú pháp như sau: Dim tên_biến

4.4.3 Các lệnh cơ bản của ASP

a. Lệnh gán:

Cú pháp: <%<biến>=[giá trị]%>

Lệnh này sẽ nạp giá trị vào biến

b. Lệnh đưa ra màn hình giá trị của biến:

Cú pháp: <%=<biến>%>

Khi xử lý lệnh này, ASP chuyển đoạn mã ngữ trên thành một văn bản với nội dung chính là lệnh gán giá trị của biến. Khi trình duyệt xử lý nó sẽ hiển thị giá trị này ra màn hình.

c. Các cấu trúc điều khiển:

Câu lệnh If-then-else – end if

<% if <điều kiện> then <các câu lệnh> [else <các câu lệnh khác>]%>

Ví dụ kiểm tra thời gian để hiển thị câu "Bây giờ là buổi sáng" hay "Bây giờ là buổi chiều"

```
<% if time<=#12:00:00 AM# then
 x="Bây giờ là buổi sáng"
else
 x="Bây giờ là buổi chiều"
end if
%>
<%=x%>
```

Cấu trúc lựa chọn Select Case

Cú pháp:

```
Select case biến
Case <tậpgiátrị1>
 <dãy câu lệnh 1>
Case <tậpgiátrị2>
 <dãy câu lệnh 2>
…….
Case <tậpgiátrị n>
 <dãy câu lệnh n>
Case else
<dãy câu lệnh n+1>
end select
```

Ví dụ:

```
<%
bien=5
select case bien
 case 1,2,3
    Response.Write  ("chon 1")
  case 2,3,4
     Response.Write "chon 2"
  case else
   Response.write "chon 3"
```

```
  end select
%>
```

=> Kết quả: Chon 3

4.4.4 *Vòng lặp For:*

Cú pháp:

```
<%For <biến đếm>=<giá trị đầu> to <giá trị cuối>%>
[Các lệnh khác]
<%Next [biến đếm]%>
```

Ví dụ:

```
<%for i=5 to 7 %>
  <font size =<%=i%>>Chào bạn!<br></font>
<% Next %>
```

4.4.5 *Câu lệnh lặp không xác định:*

Cú pháp:

```
while <điều kiện>
 <thực hiện công việc>
Wend
```

```
Do while <điều kiện>
Loop
```

```
Do
<công việc>
Loop While <điều kiện>
```

Ví dụ:

```
<%
i=1
do
i=i+1
Response.Write i
loop while  i<=10
%>
```

4.5 Xây dựng các hàm và thủ tục trong ASP:

Ta có thể sử dụng các ngôn ngữ Script để xây dựng các hàm và thủ tục trong file ASP. Trước khi viết một hàm và thủ tục bằng ngôn ngữ gì ta phải thông báo cho ASP biết bằng thẻ Script như đã biết.

Cấu trúc một hàm trong ASP có dạng sau:

```
<SCRIPT RUNAT =SERVER LANGUAGE=”LANGUAGENAME”>
‘Hàm:
Function <FunctionName> (Biến)
```

```
Các dòng lệnh Script
End Function
'Thủ tục:
Sub <SubName> (Biến)
 Các dòng lệnh Script
End Sub
</SCRIPT>
```

Đối với hàm thì trong thân của hàm cần có một lệnh gán giá trị tính được cho một biến có thên trùng với tên hàm.

Cách gọi hàm hoàn toàn tương tự như cách lấy giá trị từ một biến.

Cách gọi thủ tục: Call SubName hoặc SubName

Ví dụ ta có hàm sau:

```
<%Function Calculate(A, B, Op)
Select Case Op
Case  "+"
Calculate = A+B
Case "-"
Calculate = A-B
Case "*"
Calculate = A*B
Case "/"
Calculate = A/B
End Select
End Function
        Response.write Calculate(2, 3, "+")
        Response.write Calculate(2, 3, "-)
%>
```

Chú ý: Có thể sử dụng <!--#include file|virtual="file_name"--> để sử dụng lại các hàm và thủ tục đã được xây dựng trong một file nào đó.

4.6 Sử dụng các đối tượng của ASP để trao đổi thông tin giữa Client và Server.

4.6.1 *Giới thiệu các đối tượng chính của ASP:*

a. Các đối tượng chính:

Tương tự như trong các ngôn ngữ lập trình hướng đối tượng, ASP cho phép người lập trình tạo ra các đối tượng, các lớp theo mục đích sử dụng riêng. ASP cũng cung cấp sẵn có một số đối tượng hay được sử dụng. Đó là 5 đối tượng sau:

Các đối tượng	Ý nghĩa
Application	Chia sẻ thông tin giữa các người dùng trong một ứng dụng
Session	Lưu giữ các thông tin duy nhất về phiên làm việc hiện thời của một người sử dụng cụ thể
Server	Cho phép truy cập tới máy chủ

Request	Lấy thông tin từ phía người dùng
Response	Gửi thông tin tới người dùng

Mỗi đối tượng đều có các phương thức đi kèm. Cú pháp chung để gửi thông điệp cho các đối tượng hoàn toàn bình thường:

Object.Method parameters.

Ở đây parameters có thể là biến, dữ liệu, chuỗi hoặc URL tuỳ thuộc vào phương thức Method.

Ngoài ra còn có đối tượng ObjectContext: để chấp nhận hoặc từ chối một giao tác. Đối tượng này được quản lý bởi MTS và có thể được khởi xướng nhờ một câu lệnh script chứa trong một trang ASP. Khi một trang ASP chứa @TRANSACTION thì trang đó sẽ được chạy trong giao tác đó và chỉ kết thúc khi giao tác đó đã thành công hoặc thất bại. Và đối tượng ASPError chứa thông tin về lỗi xuất hiện trong mã lệnh trong trang ASP.

b. File Global.asa

Là nơi khai báo các đối tượng, biến có phạm vi phiên làm việc hay toàn bộ ứng dụng. File Global được kích hoạt mỗi khi một phiên làm việc mới được thiết lập, tuy nhiên sự kiện Application_OnStart chỉ được kích hoạt một lần khi Webserver được khởi động. Mỗi một ứng dụng chỉ có thể có duy nhất một file Global.asa.

Các sự kiện của các đối tượng Application và Session được khai báo trong file Global.asa.

Cú pháp:

```
<Script Language=VBScipt  RUNAT=Server>
Application_OnStart
End Sub
Application_OnEnd
End Sub
Session_OnStart
End Sub
Session_OnEnd
End Sub
</Script>
```

Ngoài ra ta có thể viết các hàm và thủ tục đặt trong file Global.asa để phục vụ cho cả ứng dụng hay cho từng phiên làm việc cụ thể, các thủ tục và các hàm này phải nằm trong các sự kiện của hai đối tượng Application và Session.

4.6.2 *Đối tượng Request*

a. ý nghĩa:

Lấy yêu cầu từ máy khách nhờ phương thức HTTP . Là kiểu đối tượng quan trọng nhất trong ASP. Thông qua việc sử dụng đối tượng Request ta có thể lấy được cả dữ liệu và tham số trong một trang HTML được gửi qua đường địa chỉ.

Khi một Browser liên lạc với Server thông qua giao thức HTTP, Browser gửi yêu cầu tới Server, ngoài tên của trang được yêu cầu thì còn rất nhiều thông tin

khác đi kèm được gửi tới Server. Các thông tin này có thể là các biến môi trường, các thông tin do user cung cấp dưới dạng điền vào các bảng, Cookies, ... Tất cả các thông tin này được mã hoá và truyền đi cùng với HTTP headers. ASP cho phép lấy ra các thông tin này bằng cách sử dụng đối tượng Request.

b. Các thành phần của Request:

Tập hợp	Thuộc tính	Phương thức
QueryString Form ServerVariables Cookies ClientCertificate	TotalBytes	BinaryRead

Cú pháp:

Request.[Tập hợp](Biến)|thuộc tính|phương thức

Tập hợp:

Tập hợp	Ý nghĩa
ClientCertificate	Bao gồm các thông tin về certificate của Client.
Cookies	Đọc thông tin từ một Cookies đã có sẵn
Form	Giá trị các thành phần của form gửi đến từ Browser (Lấy thông tin do user gửi đến bằng phương thức POST)
QueryString	Lấy giá trị của các biến theo sau một URL (Lấy thông tin do user gửi đến bằng phương thức GET)
ServerVariables	Bao gồm các thông tin về Client Browser, Server và user

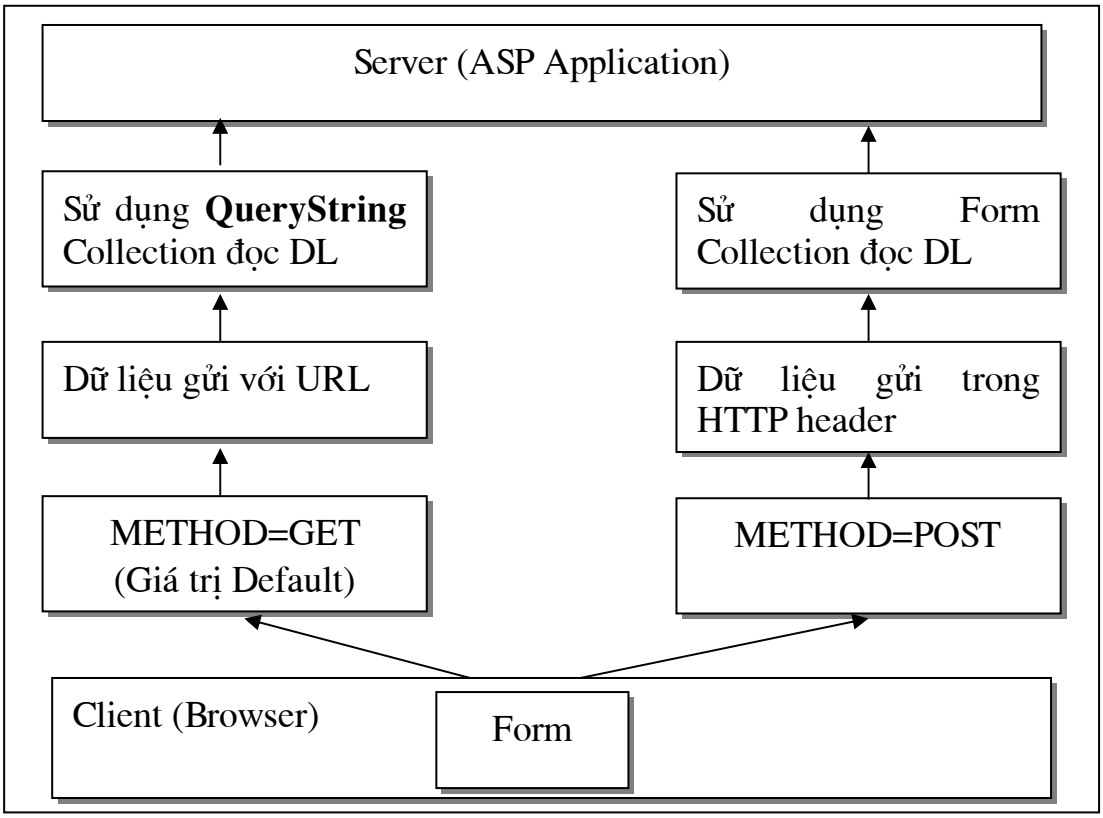
c. Tập hợp Form và QueryString:

Khi chúng ta sử dụng thẻ <FORM> trong một trang, ta có thể đặt thuộc tính METHOD của <FORM> là POST hay GET. Nếu chúng ta sử dụng GET (hay bỏ qua vì GET là giá trị mặc định của METHOD), trình duyệt sẽ lấy các giá trị trong tất cả các control để xây dựng thành QueryString và gắn vào URL của trnag được yêu cầu khi Submit trang hiện tại. Khi trang này đến Server các giá trị của nó nằm ở Collection Request.QueryString.

Ngược lại, nếu sử dụng phương thức POST, trình duyệt sẽ đưa tất cả các giá trị vào trong HTTP header gửi đến Server và các giá trị này có thể truy xuất qua Collection Request.Form

Nói chung, ta nên sử dụng phương thức Post trong tất cả các form HTML. Thứ nhất, chiều dài chuỗi của URL bị giới hạn nên nếu dùng QueryString sẽ có nguy cơ bị tràn và bị cắt bớt. Thứ hai, query string đưa các giá trị tường minh vào URL, và sẽ được ghi lại trong file log khi đi qua các Server, không bảo mật thông tin.

Sự khác nhau giữa hai phương thức gửi dữ liệu từ Client đến Server được chỉ ra trong sơ đồ sau:



Hình 4.7 Sự khác nhau giữa hai phương thức GET và POST.

Ví dụ sử dụng phương thức Post để gửi dữ liệu:

Nội dung file1.htm

```
<HTML>
<HEAD>
</HEAD>
<BODY>
<form action="file2.asp" method="POST">
Name:<input type="Text" name="Name" maxlength="20">
<br>Company:<input type="Text" name="Company" maxlength="20">
<br>Position:<input type="Text" name="Position" maxlength="20">
<br>Address:<textarea name="Address" rows="3"></textarea>
<br>Phone:<input type="Text" name="Phone" maxlength="20">
<br><input type="Submit" name="Submit" value="Submit">
</form>
</BODY>
</HTML>
```

Nội dung file2.asp

```
<HTML>
<HEAD>
</HEAD>
<BODY>
```

```
<% Response.Write Request.Form("Name")%>
works for<% Response.Write Request.Form("Company") %>
 at address<% Response.Write Request.Form("Address") %>
 as a<% Response.Write Request.Form("Position") %>.
</BODY>
</HTML>
```

Kết quả:

Jane Doe works for ISSI at address 5609 Kington Pike Knoville, TN as a Web Designer.

Ví dụ sử dụng phương thức Get để gửi dữ liệu:

Nội dung file1.asp:

```
<HTML>
<HEAD>
</HEAD>
<BODY>
<A HREF="File2.asp?language=ASP&type=general">
Query sample</A>
</BODY>
</HTML>
```

Nội dung file2.asp:

```
<%
For Each item In Request.QueryString
   Response.Write item & " = " & Request.QueryString(item) & "<BR>"
Next
%>
```

Output:

language = ASP

type = general

Hoặc file2.asp có thể viết:

```
<%
Response.Write "language =" & Request.QueryString("language")&
"<BR>"
Response.Write "type =" & Request.QueryString("type")& "<BR>"
%>
```

Chú ý: Khi lấy giá trị theo 2 tập hợp là Form hoặc QueryString ta chỉ cần chỉ ra Request("tên_biến")

Chẳng hạn: Request.Form("Name") có thể thay bằng Request("Name")

d. Sử dụng tập hợp Cookies:

Cookies là nhóm văn bản mà Website đặt vào một file trên đĩa cứng của web browser khi web browser truy cập Website đó. Cookie dùng để nhận diện khi Web browser này truy cập những lần sau.

Cookie được gửi đến Server cùng với mỗi yêu cầu. Dữ liệu trong Cookie được đặt trong tập hợp Cookies. Nó được truy cập tương tự QueryString và Form. Tuy nhiên, sử dụng đối tượng Request, ta chỉ đọc được giá trị của Cookie mà không thay đổi được nó.

Ví dụ:

```
<%
Response.Cookies("myProduct")("Main") = "DevSite"
Response.Cookies("myProduct")("Prd1") = "ASP"
Response.Cookies("myProduct")("Prd2") = "VBScript"
Response.Write Request.Cookies("myProduct")
%>
```

Kết quả:

PRD2=VBScript&PRD1=ASP&MAIN=DevSite

e. Sử dụng tập hợp ServerVariables:

Giá trị của các biến môi trường server. Điều này sẽ chép truy nhập tới các header HTTP.

Ví dụ:

```
<%IPAddress = Request.ServerVariables("REMOTE_ADDR")
Software = Request.ServerVariables("SERVER_SOFTWARE")
Protocol = Request.ServerVariables("SERVER_PROTOCOL")
Response.Write "Your IP Address is " & IPAddress & " and your server
is running " & Software & " under " & Protocol & "protocol."
%>
```

Kết quả

Your IP Address is 127.0.0.1 and your server is running Microsoft-IIS/4.0 under HTTP/1.1 protocol.

Hoặc ta có thể tham chiếu tới tất cả các tham số của tập hợp ServerVariables như sau:

```
<% For each item in request.servervariables
Response.write “<br>” & item & “=” & request.servervariables(item)&
“<br>”
Next
%>
```

f. Thuộc tính:

TotalBytes: Chỉ ra tổng số byte máy khách gửi tới máy chủ trong phần thân yêu cầu HTTP.

g. Phương thức:

BinaryRead: Đọc dữ liệu từ thân HTTP gửi đến server .

Ví dụ:

----------File1.html----------

```
<HTML>
<HEAD>
```

```
</HEAD>
<BODY>
<FORM ACTION="File2.asp" METHOD="POST">
Name:<INPUT TYPE="Text" NAME="name" MAXLENGTH="30">
<BR>
Age: <INPUT TYPE="Text" NAME="age" MAXLENGTH="10"><BR>
Sex: <INPUT TYPE="Text" NAME="sex" MAXLENGTH="10"><BR>
<INPUT TYPE="Submit" NAME="submit" VALUE="submit"><BR>
</FORM>
</BODY>
</HTML>
```

----------File2.asp----------

```
<%
Dim ByteCount,BinRead
ByteCount = Request.TotalBytes
BinRead=Resquest.BinaryRead(ByteCount)
Response.Write("ByteCount = " & ByteCount & " bytes")
%>
```

Kết quả:

Tùy thuộc vào dữ kiện được nhập tại form, ví dụ nếu Name=Bill, Age=56, and Sex=male. Thì ta sẽ có kết quả là:

ByteCount = 39 bytes

4.6.3 *Đối tượng Response*

Được sử dụng để truy xuất các đáp ứng mà server tạo ra để gửi về cho user

Tập hợp	Thuộc tính	Phương thức
Cookies	Buffer Charset ContentType Expires ExpiresAbsolute Status	AddHeader AppendToLog BinaryWrite Clear End Flush Redirect Write

Cú pháp:

Response. tập hợp|thuộc tính|phương thức

a. Tập hợp:

Cookies**:** Xác định giá trị của cookie gửi cho browser . Các thành phần của tập hợp này đều là giá trị chỉ ghi.

Request object cho phép chúng ta đọc thông tin của cookies khi có 1 yêu cầu được gửi đến. Còn Response object cung cấp khả năng đặt hoặc thay đổi các giá trị của cookies trước khi gửi trả lại cho client.

Ví dụ:

Dòng lệnh sau đây sẽ thêm 1 cookie vào cookie của client nếu nó chưa được đặt hoặc thay đổi giá trị của cookie này nếu nó đã tồn tại.

<% Response.cookies("independentcookie")="4th of July"%>

Chú ý: Việc thay đổi cookie với Response object cần được làm trước khi viết mã văn bản hoặc HTML để gửi tới client, nếu không sẽ gây ra lỗi.

b. Phương thức :

Phương thức	Ý nghĩa
AddHeader	Thêm phần đầu HTTP với một giá trị cụ thể để gửi lại cho browser
AppendToLog	Thêm một chuỗi văn bản vào mục nhập nhật ký máy chủ cho yêu cầu từ máy khách hiện thời
Binarywrite	Viết thông tin trực tiếp tới nội dung tín hiệu đáp mà không chuyển đổi ký tự nào
Clear	Xoá bộ đệm
End	Dừng việc xử lý 1 trang và trả lại kết quả hiện tại
Flush	Gửi những dữ liệu có trong bộ đệm tới browser ngay lập tức
Redirect	Chỉ cho browser kết nối đến một URL khác
Write	Gửi text trực tiếp tới trình duyệt phía user

❖ **AddHeader: Response.AddHeader Name, Value**

Đặt tên header HTML: Phải được gọi trước khi output được gửi cho client trừ khi response.buffer được đặt là true.

Ví dụ:

```
<%
Response.AddHeader "MyHeader", "ERROR"
%>
```

❖ **AppendToLog : Response.AppendToLog(String)**

Thêm xâu vào cuối mục log Web server cho yêu cầu này. String lớn nhất là 80 ký tự

Ví dụ:

```
<%
Response.AppendToLog("Error in Processing")
%>
```

Đầu ra được chèn vào cuối file IIS log:

10.78.176.37, - , 03/20/97, 7:55:20, W3SVC, SALES1, 10.78.176.37, Error in Processing

❖ **Clear: Response.Clear**

Xóa toàn bộ output HTML đang ở trong buffer, không xóa header, chỉ xóa phần thân (body). Nếu buffer không được thiết đặt sẽ gây ra lỗi. Phải đặt *Response.Buffer=True* (mặc định trong phiên bản 3.0)

Ví dụ:

```
<%Response.Clear %>
```

❖ **End: Response.End**

Kết thúc xử lý file .asp và trả lại kết quả hiện tại

Ví dụ:

```
<%
Response.Write "Hello World"
Response.End
Response.Write "Is this the End?"
%>
```

Kết quả:

Hello World

❖ **Flush**: **Response.Flush**

Gửi thông tin trong buffer ra ngay lập tức

Response.buffer=true

Ví dụ:

```
<%Response.Flush %>
```

❖ **Redirect: Response.Redirect(URL)**

Kết thúc xử lý file .asp hiện tại, chuyển trình duyệt tới một URL khác.

Ví dụ:

----------File1.asp----------

```
<%
Response.Write "This is File1.asp and switching to File2.asp"
Response.Redirect "File2.asp"
Response.Write "This line is not written to the file"
%>
```

----------File2.asp----------

```
<%
Response.Write "This is File2.asp"
%>
```

Kết quả:

File1 được thực hiện sau đó trình duyệt sẽ tải File2 lên:

----------File1.asp----------

This is File1.asp and switching to File2.asp

----------File2.asp----------

This is File2.asp

❖ **Write: Response.Write(Variant)**

Ghi thông tin dưới dạng xâu vào output HTTP hiện thời.

Ví dụ:

```
<%
Response.Write "Hello World"
Response.Write "<BR>"%>
```

Kết quả:
Hello World

c. Thuộc tính:

Thuộc tính	Ý nghĩa
Buffer	Xác định xem một trang có sử dụng bộ đệm để chứa các kết quả được tạo bởi Script hay được gửi ngay tới browser khi từng dòng được tạo ra và nhập vào dải HTML. Giá trị ngầm định là False
ContentType	Xác định kiểu nội dung HTTP được trả về. Kiểu ngầm định là Text/HTML
Expires	Khoảng thời gian 1 trang Web được lưu giữ trên browser
ExpiresAbsolute	Ngày và thời gian 1 trang Web được lưu giữ trên browser
Status	Giá trị của dòng trạng thái HTTP trả lại bởi server
Charset	Đặt dạng ngôn ngữ sử dụng cho client browser vào phần cuối của đoạn đầu ContentType

❖ **Buffer**

- Buffer: nơi lưu giữ tạm thời trước khi chuyển cho trình duyệt
- Chỉ ra output của trang có được lưu trong buffer hay không
- Sẽ không có sự gửi lại cho trình duyệt cho đến khi tất cả các script được xử lý xong, hoặc có lời gọi phương thức Flush hay End.

Thuộc tính này không thể thay đổi khi server đã gửi thông tin cho trình duyệt và phải được đặt ở dòng đầu tiên trong file .asp

Ví dụ:

```
<%
Response.Buffer = TRUE
x=0
Do
   x = x+1
   Response.Write x & "<BR>"
Loop
%>
```

❖ **Response.CacheControl**

Có cho phép máy chủ proxy được cache output của trang .asp hay không.

Máy chủ proxy dùng để đẩy nhanh tốc độ truy nhập tới trang Web nào đó bằng cách lưu lại một bản của trang Web trong cache.

- Nếu **CacheControl** được đặt là **"Public"**, thì cho phép cache.
- Nếu **CacheControl** đặt là **"Private"**, thì không cho phép cache.

Ví dụ:

```
<% Response.CacheControl = "Public" %>
<% Response.CacheControl = "Private" %>
```

❖ **Charset: Response.Charset**

Thêm tên của tập ký tự vào trong header content-type
Mặc định là ISO-LATIN-1
Ví dụ:
Trong ví dụ này, nếu header là: content-type: text/html
Thì header sẽ trở thành:
content-type: text/html; charset = MS_Kanji

```
<% Response.Charset("MS_Kanji") %>
```

❖ **ContentType** : **Response.ContentType**

Chỉ ra kiểu nội dung HTTP để trả lại. Mặc định là text/HTML
Ví dụ:

```
<% Response.ContentType = "application/vnd.ms-excel"  %>
```

❖ **Expires**

Xác định khoảng thời gian trước khi một trang được cache hết hạn
Ví dụ:

- Trường hợp 1: bất kỳ khi nào gọi tới trang đó, nó luôn luôn được refresh.
- Trường hợp 2: Nếu trở lại trang đó trước 15 phút, trang hiển thị sẽ là trang ở trong cache.

```
<% Response.Expires = 0 %>
<% Response.Expires = 15 %>
```

❖ **Response.ExpiresAbsolute**

Xác định ngày và thời gian chính xác một trang sẽ hết hạn.
Ví dụ:

```
<% Response.ExpiresAbsolute=#May 15, 1999 18:00:00# %>
```

❖ **Response.IsClientConnected**

- Xác định xem client đã ngừng kết nối với server từ **Response.Write** cuối cùng.
- Thuộc tính này đặc biệt có ý nghĩa để server không phải tiếp tục thực hiện chuyển những thông tin client không yêu cầu.

Ví dụ:

```
<%
  'Check to see if the client is connected.
  If Not Response.IsClientConnected Then
    'Get the sessionid to send to the shutdown function.
    Shutdownid = Session.SessionID
  'Perform shutdown processing.
    Shutdown(Shutdownid)
  End If
%>
```

❖ **Response.Status**

Dòng trạng thái do server trả lại

Ví dụ:

```
<%
IPAddress = Request.ServerVariables("REMOTE_ADDR")
If IPAddress <> "208.5.64.223" Then
   Response.Status = "403 Access Forbidden"
   Response.Write Response.Status
   Response.End
End If
%>
<BODY>
You have accessed this page through the IP Address of 208.5.64.223.
</BODY>
```

Kết quả sau sẽ được trả về nếu địa chỉ IP trên máy Client là 208.5.64.223:

You have accessed this page through the IP Address of 208.5.64.223.

Ngược lại nếu địa chỉ IP của Client không phải là 208.5.64.223 thì kết quả sau sẽ trả về:

403 Access Forbidden

4.6.4 *Đối tượng Server*

Cho phép truy nhập tới các phương thức và thuộc tính trên máy chủ.

Tập hợp	Thuộc tính	Phương thức
	ScripTimeOut	CreateObject HTMLEncode MapPath URLEncode

Cú pháp:

Server.Thuộc tính|Phương thức

a. Thuộc tính:

ScriptTimeout: Server.ScriptTimeout

thời gian tối đa để trang script chạy trên máy chủ. Nếu không đặt giá trị cho thuộc tính này thì giá trị mặc định của nó là 90 giây.

Nếu script nhập vào một vòng lặp vô hạn thì server sẽ kết thúc script đó để tránh bị overload bởi việc chạy liên tục các tiến trình sinh ra. Thời gian trước khi script bị kết thúc được định nghĩa bởi thuộc tính này.

```
<% Server.ScriptTimeout = 150 %>
```

Ta có thể lấy được giá trị của thuộc tính ScriptTimeout bằng cách:

```
<% timeout = Server.ScriptTimeout %>
```

b. Phương thức

Phương thức	Ý nghĩa
CreateObject	Tạo một thể hiện của đối tượng cụ thể trên Server
Execute	Cho phép gọi trang ASP khác trong một trang ASP
GetLastError	Mô tả đối tượng lỗi ASP, chỉ có ý nghĩa trước khi file asp gửi nội dung tới Client
HTMLEncode	Gắn một đoạn mã HTML vào một xâu đã được định dạng.
MapPath	Xác định đường dẫn vật lý trên máy chủ khi xét đết đường dẫn ảo.
Transfer	Chuyển tới trang ASP khác từ một trang ASP. Thông tin trạng thái hiện tại trong trang đầu tiên sẽ được chuyển tới trang thứ hai
URLEncode	Cho phép gắn một đoạn mã URL

❖ **Server.CreateObject(ObjectID)**

Tạo một thể hiện (instance) của đối tượng server (đối tượng activeX bất kỳ trên server), sau đó có thể sử dụng các phương thức và truy cập tới các thuộc tính của đối tượng đó.

ObjectID là đối tượng cần khởi tạo.

Đoạn mã sau dùng để kết nối với CSDL:

```
<% Set myconn = Server.CreateObject("ADODB.Connection") %>
```

❖ **Server.Execute (Path)**

- Cho phép gọi trang ASP khác trong một trang ASP. Khi trang được gọi tới hoàn thành các công việc của nó, sẽ trở lại tiếp tục thực hiện trang ASP gọi tới nó. Hiệu quả giống như các hàm, thủ tục (subroutines). Phương pháp có hiệu quả tương tự include.
- Phương thức Transfer chuyển tới trang ASP khác không quay lại trang đã gọi tới.
- Tham số Path chỉ đường dẫn tương đối hoặc vật lý, toàn bộ xâu này được đặt trong dấu nháy.

Ví dụ:

----------CallingAsp.asp----------

```
<HTML>
<BODY>
How now <%Server.Execute("CalledAsp.asp")%> cow?
</BODY>
</HTML>
```

----------CalledAsp.asp----------

```
<%
Response.Write "pink"
%>
```

Kết quả:

How now pink cow?

❖ **Server.GetLastError**

Trả lại đối tượng ASPError, đối tượng này có 9 thuộc tính chỉ đọc cung cấp thông tin chi tiết về lỗi.

Ví dụ:

```
<%
Dim objErrorInfo
Set objErrorInfo = Server.GetLastError
Response.Write("ASPCode = " & objErrorInfo.ASPCode)
Response.Write("ASPDescription = " & objErrorInfo.ASPDescription)
Response.Write("Category = " & objErrorInfo.Category)
Response.Write("Column = " & objErrorInfo.Column)
Response.Write("Description = " & objErrorInfo.Description)
Response.Write("File = " & objErrorInfo.File)
Response.Write("Line = " & objErrorInfo.Line)
Response.Write("Number = " & objErrorInfo.Number)
Response.Write("Source = " & objErrorInfo.Source)
%>
```

❖ **Server.HTMLEncode (String)**

Phương thức này cho phép mã hoá chuỗi thành mã HTML đối với xâu ASCII bất kỳ. Ví dụ, điều này cho phép hiển thị thẻ HTML mà không xử lý nó như những thẻ HTML thực sự.

Ví dụ:

```
<% Response.Write Server.HTMLEncode("The tag for a table is: <Table>") %>
```

Kết quả:

The tag for a table is: <Table>

Kết quả ở Browser:

The tag for a table is: <Table>

❖ **Server.MapPath (Path)**

Ánh xạ đường dẫn ảo, hay tương đối tới đường dẫn vật lý. Phương thức này không kiểm tra sự tồn tại thực sự của đường dẫn. Nếu bắt đầu bằng dấu / hoặc \ -> đường dẫn ảo. Còn không bắt đầu bằng ký tự đó -> đường dẫn tương đối.

Ví dụ:

```
<HTML>
<BODY>
The path of this file is <% Response.Write
Server.MapPath("test.asp")%>
The path of the file1 is <% Response.Write
Server.MapPath("\test.asp")%>
The path of the file2 is <% Response.Write
Server.MapPath("test\test.asp") %>
```

```
The path of the file3 is <% Response.Write Server.MapPath("\") %>
</BODY>
</HTML>
```

Kết quả:

The path of this file is C:\VANBANG2\ASP\Example\test.asp
The path of the file1 is d:\inetpub\wwwroot\test.asp
The path of the file2 is C:\VANBANG2\ASP\Example\test\test.asp
The path of the file3 is d:\inetpub\wwwroot

❖ **Server.Transfer (Path)**

Chuyển tới trang ASP khác từ một trang ASP. Thông tin trạng thái khởi tạo trong trang đầu tiên sẽ được chuyển tới trang thứ hai.

Ví dụ:

----------CallingAsp.asp----------

```
<%
Application("name") = "Application Maker"
Application("publishdate") = "05/15/01"
Application("author") = "DevGuru"
Set Application("Obj1") =
Server.CreateObject("ADODB.Connection")
Server.Transfer("CalledAsp.asp")
%>
```

----------CalledAsp.asp----------

```
<%;
Response.Write "Output from CalledAsp.asp"
For Each Item in Application.Contents
   If IsObject( Application.Contents(Item)) Then
   Response.Write Item & “is an object.<BR>”
   Else
   Response.Write Item & “=” & Application.Contents(Item) &“<BR>”
   End If
Next
%>
```

Kết quả:

Kết quả từ CalledAsp.asp
name=Application Maker
publishdate=05/15/01
author=DevGuru
OBJ1 is an object.

❖ **Server.URLEncode(String)**

Chuyển xâu thành dạng mã hoá URL, để đảm bảo hyperlink trong ASP đó được định dạng đúng đắn.

Ví dụ:

```
<% Response.Write Server.URLEncode("http://www.issi.net") %>
```

Kết quả:

http%3A%2F%2Fwww%2Eissi%2Fnet

***4.6.5** Đối tượng Application*

Một ứng dụng bao gồm các file có thể truy nhập thông qua một thư mục ảo xác định và các thư mục con của nó.

Đối tượng Application thể hiện toàn bộ một ứng dụng ASP. Chúng ta có thể sử dụng ứng dụng này để chia sẻ thông tin cho tất cả các người dùng trong một ứng dụng.

Đối tượng Application được bắt đầu khi có một yêu cầu đầu tiên một trang web bất kỳ từ thư mục ảo tại Web server và tồn tại cho đến khi Webserver ngừng hoạt động.

a. Tập hợp:

❖ **Application.Contents(Key)**

Chứa danh sách các mục vừa được khởi tạo và đưa vào đối tượng Application.

Ví dụ:

```
<% Application("name") = "Application Maker"
Application("publishdate") = "05/15/01"
Application("author") = "DevGuru"
Set Application("Obj1") = Server.CreateObject("ADODB.Connection")
For Each Item in Application.Contents
   If IsObject( Application.Contents(Item)) Then
     Response.Write Item & " is an object.<BR>"
   Else
     Response.Write Item & "=" & Application.Contents(Item) & "<BR>"
   End If
Next
%>
```

Kết quả:

name=Application Maker
publishdate=05/15/01
author=DevGuru
OBJ1 is an object

❖ **Phương thức của tập hợp Contents:**

- ***Application.Contents.Remove (Name|Integer)***

Loại bỏ mục nào đó trong collection **Application.Contents**

Name chỉ ra tên mục sẽ xóa, nằm trong cặp dấu nháy (“”). **Integer** chỉ ra vị trí mục trong collection sẽ được xóa. Giá trị này bắt đầu từ 1.

Ví dụ:

```
<%
Application("name") = "Application Maker"
Application("publishdate") = "05/15/01"
Application("author") = "DevGuru"
Set Application("Obj1") = Server.CreateObject("ADODB.Connection")
Application.Contents.Remove(1)
Application.Contents.Remove("publishdate")
For Each Item in Application.Contents
   If IsObject(Application.Contents(Item)) Then
     Response.Write Item & " is an object.<BR>"
   Else
     Response.Write Item & "=" &Application.Contents(Item) & "<BR>"
   End If
Next
%>
```

Kết quả:
author=DevGuru
Obj1 is an object.

- ***Application.Contents.RemoveAll***

Loại bỏ tất cả các mục trong collection **Application.Contents** .
Thêm cặp dấu ngoặc ()

```
<%Application.Contents.RemoveAll( )%>
```

❖ **Application.StaticObjects(Key)**

Chứa tất cả các mục đó được tạo trong ứng dụng bằng thẻ <OBJECT.

----------Global.asa----------

```
<OBJECT RUNAT=Server SCOPE=Application
ID=MyInfo PROGID="MSWC.MyInfo">
</OBJECT>
<OBJECT RUNAT=Server SCOPE=Application
ID=MyConnection PROGID="ADODB.Connection">
</OBJECT>
<OBJECT RUNAT=Server SCOPE=Application
ID=MyADRot PROGID="MSWC.ADRotator">
</OBJECT>
```

----------File.asp----------

```
<%
For Each Item In Application.StaticObjects
Response.Write Item & "<BR>"
Next
%>
```

Kết quả:
MyInfo
MyConnection
MyADRot

b. Sự kiện:

Ứng với hai hoạt động bắt đầu và kết thúc một đối tượng Application ta có hai sự kiện trong đối tượng Application, đó là:

Application_OnStart (khởi tạo các thông tin phục vụ cho một ứng dụng khi ứng dụng bắt đầu) và **Application_OnEnd** (được kích hoạt khi ứng dụng kết thúc)

Cú pháp của sự kiện Application_OnStart:

```
<SCRIPT LANGUAGE=VBScript  RUNAT=Server>
Sub Application_OnStart
  ' Nơi chèn đoạn Script cần thiết cho việc khởi tạo một ứng dụng
End Sub
Sub Application_OnEnd
  ' Nơi chèn đoạn Scipt cần thiết cho việc kết thúc một ứng dụng
End Sub
</SCRIPT>
```

Chú ý: Phải khai báo ngôn ngữ script được sử dụng trong các đoạn script sự kiện trong dòng đầu tiên của file global.asa

- Các đối tượng của ASP trong phần này chỉ có SERVER và APPLICATION

----------Global.asa----------

```
<script Language="VBScript" RUNAT=Server>
Sub Application_OnEnd()
End Sub
Sub Application_OnStart()
    Application("NumSession") = 0
    Application("NumVisited") = 0
    Session.Timeout = 10
End Sub
Sub Session_OnEnd()
  Application("NumSession") = Application("NumSession") – 1
End Sub
Sub Session_OnStart()
  Application("NumSession") = Application("NumSession") + 1
  Application("NumVisited") = Application("NumVisited") + 1
End Sub
</script>
```

----------File1.asp----------

```
Response.Write "You are " & Application("NumSession") & " of "
& Application("NumVisited") & " users."
```

Kết quả:
You are 1 of 1 users.

c. Phương thức:

Vì đối tượng Application có thể được chia sẻ thông tin giữa các người dùng do đó để đảm bảo những người dùng đó không thể cùng một lúc thay đổi nội dung của một biến trong đối tượng Application nó có hai phương thức là Lock và Unlock.

Phương thức	Ý nghĩa
Lock	Phương thức Lock ngăn cản các client khác cùng một lúc thay đổi giá trị của một biến do đối tượng Application lưu trữ
Unlock	Phương thức Unlock cho phép các client có thể sửa đổi các thuộc tính của đối tượng Application .

4.6.6 *Đối tượng Session*

Đối tượng Sesion được dùng để lưu trữ thông tin cần thiết cho một phiên làm việc của người dùng cụ thể. Các biến lưu trữ trong đối tượng Session không bị mất khi người dùng truy cập các trang Web khác trong ứng dụng. Thay vào đó các biến này tồn tại trong toàn bộ phiên làm việc của người dùng.

Khi một người dùng mới yêu cầu một trang Web từ ứng dụng, Web server tự động tạo một đối tượng Session và server sẽ phá huỷ đối tượng Session khi phiên làm việc kết thúc hoặc bị huỷ bỏ. Có thể đặt thời gian tồn tại cho một phiên làm việc tuy nhiên giá trị ngầm định cho một phiên làm việc tồn tại là 20 phút.

a. Tập hợp:

❖ **Session.Contents(Key)**

Chứa danh sách các mục đã khởi tạo và thêm vào bằng đối tượng **session.**
Không phải khởi tạo bằng thẻ <object>
Ví dụ:

```
<%
Session("name") = "Application Maker"
Session("publishdate") = "05/01/99"
Session("author") = "ISSI"
Set Session("Obj1") = Server.CreateObject("ADODB.Connection")
 For Each Item in Session.Contents
   If IsObject( Session.Contents(Item)) Then
     Response.Write Item & " is an object.<BR>"
   Else
     Response.Write Item & "=" & Session.Contents(Item) & "<BR>"
   End If
Next
%>
```

Kết quả:
NAME=Application Maker

PUBLISHDATE=05/01/99
AUTHOR=ISSI
OBJ1 is an object.

❖ **Session.StaticObjects(Key)**

Chứa danh sách các mục vừa được khởi tạo và đưa vào đối tượng Session nhờ sử dụng thẻ HTML <OBJECT>.

Ví dụ:

----------Global.asa----------

```
<OBJECT RUNAT=Server SCOPE=Session ID=MyInfo
PROGID="MSWC.MyInfo">
</OBJECT>
<OBJECT RUNAT=Server SCOPE=Session
ID=MyConnection PROGID="ADODB.Connection">
</OBJECT>
<OBJECT RUNAT=Server SCOPE=Session ID=MyADRot
PROGID="MSWC.ADRotator">
</OBJECT>
```

----------File.asp----------

```
<%
For Each Item In Session.StaticObjects
Response.Write Item & "<BR>"
Next
%>
```

Kết quả:
MyInfo
MyConnection
MyADRot

b. Sự kiện:

Cũng như đối tượng Application, đối tượng Session có hai sự kiện là Session_OnStart(được kích hoạt khi mỗi khi một phiên làm việc bắt đầu) và Session_OnEnd (được kích hoạt khi kết thúc một phiên làm việc)

Cú pháp

```
<SCRIPT LANGUAGE=VBScript RUNAT=Server>
Sub Session_OnStart
 ' Nơi chèn đoạn Script cần thiết cho việc khởi tạo một phiên
làm việc
End Sub
Sub Session_OnEnd
 ' Nơi chèn đoạn Script cần thiết khi một phiên làm việc kết thúc
End Sub
</SCRIPT>
```

c. Thuộc tính:

Thuộc tính	Ý nghĩa
SessionID	Trả về giá trị nhận biết duy nhất từng phiên làm việc của một người sử dụng
Timeout	Khoảng thời gian ngầm định là 20' cho sự tồn tại của một phiên làm việc trong một ứng dụng. Máy chủ Web sẽ duy trì thông tin phiên làm việc của người sử dụng mà không đưa ra yêu cầu hay làm mới lại một trang.

d. Phương thức:

- **Abandon: Huỷ bỏ một đối tượng Session và giải phóng các biến trong đối tượng Session.**
- **Session.Contents.Remove (Name|Integer)**

  Dùng để xóa tất cả các mục trong collection **Session.Contents**
  Có 2 lựa chọn tham số **Name** hoặc **Integer**
  - Name chỉ tên của mục cần xóa
  - Integer là số chỉ vị trí (bắt đầu từ 0) của mục cần xóa.
- **Session.Contents.RemoveAll**

  Xóa tất cả các mục trong session.contents
  Ví dụ:
  <%Session.Contents.RemoveAll( )%>

Chú ý: Có thể lưu trữ các giá trị trong đối tượng Session. Thông tin lưu trữ trong đối tượng Session có phạm vi phiên làm việc và có thể sử dụng được trong suốt một phiên làm việc.

4.7 Câu hỏi và bài tập chương 4

4.7.1 *Câu hỏi ôn tập*

1. ASP là viết tắt của cụm từ nào?

a. All Standard Pages
b. Active Server Pages
c. Active Standard Pages
d. A Server Page

2. Nội dung của đoạn script ASP được định nghĩa bởi cặp thẻ nào dưới đây?

a. <&>...</&>
b. <%>...</%>
c. <script>...</script>
d. <%...%>

3. IIS là gì? Nó hoạt động như thế nào?
4. Các bước thiết lập thư mục ảo trong IIS.
5. Các lệnh cơ bản của ASP.

4.7.2 *Bài tập về các cấu trúc điều khiển và vòng lặp.*

1. Viết đoạn mã chương trình ASP dùng cấu trúc Select...Case để hiển thị ra màn hình ngày hiện hành trong tuần bằng tiếng Việt.

2. Viết đoạn mã chương trình ASP dùng vòng lặp Do...Loop để viết ra màn hình 10 dòng chữ "Hello world" có kích thước tăng dần.

4.7.3 *Bài tập về các đối tượng.*

1. Viết ra màn hình câu "Hello world"

2. Viết một form lấy ý kiến của người sử dụng về trang web của mình, rồi hiển thị các thông tin người sử dụng điền vào ra màn hình.

3. Thiết kế và cài đặt trang Web hiển thị máy tính tay với các phép toán cộng, trừ, nhân, chia, lũy thừa, căn bậc hai, bình phương, nghịch đảo.

4. Viết một đoạn code dùng để đếm số lần truy cập vào một trang Web. Sau đó nhúng vào trang Web đã làm ở bài 2.

5. Thiết kế và cài đặt trang Web hiển thị lịch (calendar). Trang Web phải cho phép người dùng xem lịch tháng của một năm nào đó, xem lịch của các tháng trước và sau tháng hiện tại của năm hiện tại.

6. Thiết kế và cài đặt trang Web hiển thị hệ thống cây thư mục của máy chủ.

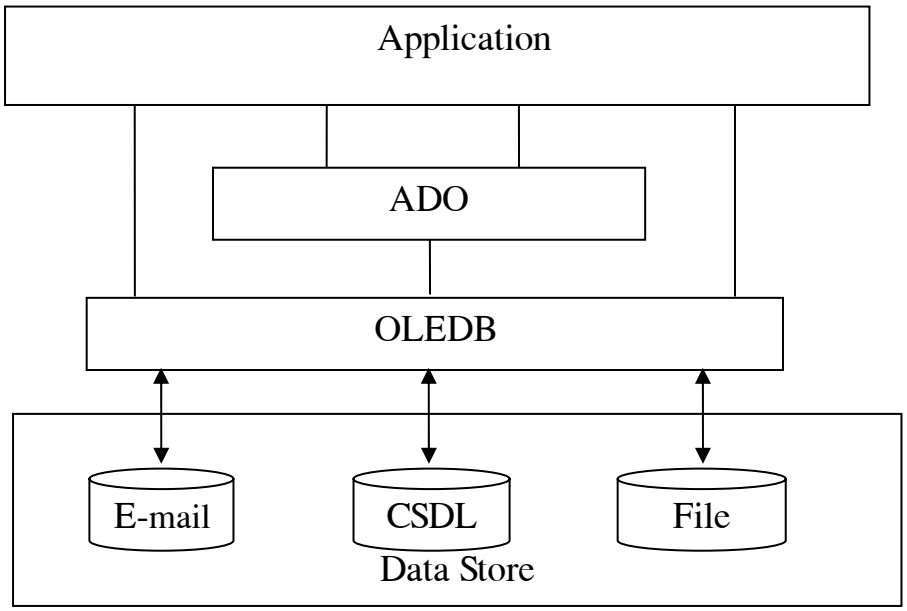
Chương 5

KẾT NỐI CƠ SỞ DỮ LIỆU TRONG LẬP TRÌNH WEB ĐỘNG VỚI ASP

5.1 Khái niệm về ADO

ActiveX Data Object là lớp đối tượng COM (Compenent Object Model) tập trung vào xử lý dữ liệu thông qua OLEDB của Windows. ADO thiết kế cho mục đích truy xuất dữ liệu tổng quát không chỉ dùng để truy xuất dữ liệu thuần tuý mà còn xử lý được cả dữ liệu file hay bất kỳ loại dữ liệu nào có hỗ trợ cơ chế cho phép giao tiếp thông qua OLEDB.

Mô hình kiến trúc của ADO tương tác giữa ứng dụng và nguồn dữ liệu:



Hình 5.1 Mô hình kiến trúc của ADO.

5.2 Trình tiêu thụ (consumer) và trình cung cấp (provider)

Trong bước lập trình, chương trình viết ra chính là trình tiêu thụ dữ liệu bởi nó cần truy xuất vào các nguồn dữ liệu để xử lý. Còn trình cung cấp là tập lệnh cho phép truy xuất vào nguồn dữ liệu theo cách đặc trưng của chúng. Provider cho phép giao tiếp giữa nguồn dữ liệu và tầng điều khiển OLEDB. Và ADO chỉ trao đổi với nguồn dữ liệu thông qua OLEDB mà không cần quan tâm tới cách thức làm việc của Provider. Để giúp OLEDB biết được Provider nào cần phải giao tiếp, khi mở kết nối ADO cần chỉ định trình cung cấp dữ liệu Provider tương ứng. Microsoft cung cấp sẵn một số Provider cho phép truy xuất dễ dàng vào các nguồn dữ liệu đang thông dụng như:

- Jet OLEDB 4.0 – Cơ sở dữ liệu Access
- DTS Packages – Dịch vụ chuyển đổi dữ liệu trong SQL Server
- ODBC Driver – Provider ch phép truy xuất nguồn dữ liệu thông qua ODBC
- SQL Server – Cơ sở dữ liệu SQL Server
- Oracle – Cơ sở dữ liệu Oracle
- Simple Provider – Cơ sở dữ liệu dạng Text

5.3 Mô hình đối tượng ADO

5.3.1 Đối tượng kết nối (Connection)

Cho phép thực hiện việc mở kết nối đến nguồn dữ liệu cần truy xuất. Thông qua Connection chỉ cần chỉ định trình cung cấp OLEDB Provider sẽ dùng để tiếp cận dữ liệu. Các thông tin kết nối bổ sung khác như username, password, server name,.. thường được lưu vào một chuỗi gọi là chuỗi kết nối (Connection String).

Chú ý: Có thể kết nối và truy xuất vào nguồn dữ liệu mà không bắt buộc phải dùng đối tượng Connection. Cá đối tượng khác như Command, RecordSet, Record,... cũng cho phép mở trực tiếp kết nối. Tuy nhiên sử dụng đối tượng Connection sẽ cho phép bạn tách biệt thao tác kết nối và thao tác truy cập cơ sở dữ liệu. Hơn nữa đối tượng Connection còn cung cấp thêm một số chức năng chuyên dụng khác như cho phép thực thi câu lệnh SQL tác động vào dữ liệu như Insert, Update, Delete, gọi thủ tục Procedure Store,... hoặc kiểm soát giao tác transaction như Rollback, commit.

5.3.2 Đối tượng Command:

Đối tượng này dùng cho mục đích thực thi câu lệnh tốt hơn Connection. Cho phép bạn chuyển tham số vào các lệnh thực thi SQL. Tham số có thể chỉ định kiểu hoặc giá trị tường minh. Các tham số có thể nhận trị trả về sau khi thực thi..Command có thể dùng cho cả 2 mục đích: thực thi câu lệnh SQL không cần nhận kết quả trả về như Insert, Update, Delete, Procedure Store, hoặc thực thi các lệnh trả về tập RecordSet như lệnh Select.

5.3.3 Đối tượng RecordSet:

Là đối tượng sử dụng thường xuyên trong ADO. Cung cấp kết quả trả về từ câu lệnh truy vấn một tập các bản ghi. Trang ASP có thể dùng vòng lặp để duyệt qua các bản ghi này và hiển thị dữ liệu kết xuất ra trang Web phía trình duyệt. Ngoài ra RecordSet còn cho phép thực hiện lọc dữ liệu từ tập các bản ghi, truy xuất đến tong trường cụ thể của bản ghi thông qua đối tượng Field hoặc danh sách các trường trong bản ghi thông qua đối tượng Fields

5.4 Kết nối với nguồn dữ liệu

Chuỗi kết nối được dùng để cung cấp thông tin cho đối tượng Connection biết đặc điểm của cơ sở hay nguồn dữ liệu mà ADO cần truy xuất.

5.4.1 Tạo một ODBC DSN

Trước khi tạo các Script truy xuất cơ sở dữ liệu (CSDL), ta cần chỉ dẫn cho ADO xác định nguồn dữ liệu cần truy xuất và cách thức liên kết CSDL.

Phổ biến và đơn giản nhất đó là sử dụng tên nguồn dữ liệu (Data Source Name-DSN) để định vị và cấu hình nguồn dữ liệu tương thích chuẩn ODBC. Với ODBC bạn có thể lựa chọn các kiểu DNS để tạo, đó là: User, System hoặc File. Các DNS User và System thường trú trong registry của hệ điều hành WindowsNT. System DNS cho phép tất cả người sử dụng truy nhập vào Server đó đều có thể truy xuất một CSDL, trong khi đó User DNS hạn chế đối với mỗi người sử dụng đăng nhập vào Server; File DSN sẽ lưu thông tin dưới dạng file cho phép nhiều người sử dụng truy xuất CSDL và dễ dàng chuyển từ Server này sang Server khác chỉ bằng việc copy các tệp DSN. Chúng ta có thể tạo ra DSN bằng cách:

- Vào trong Start\Control Panel, click chuột vào biểu tượng ODBC, chọn

một dạng DSN.

- Click Add, chọn một trình điều khiển dữ liệu(.MDB, SQL)
- Theo các chỉ dẫn trên màn hình để cấu hình DSN cho CSDL

Sau đây là một số kết nối đến các loại CSDL thông dụng:

5.4.2 *Cơ sở dữ liệu MS Access*

```
connStr="Provider=Microsoft.JetOLEDB.4.0; Data Source = C:\Program Files\Microsoft Visual Studio\VB98\BIBLIO.MDB; Persist Security Info=False"
set conn=Server.CreateObject("ADODB.Connection")
conn.open connStr
```

5.4.3 *Cơ sở dữ liệu MS Access thông qua trình điều khiển ODBC*

```
connStr="Driver=Microsoft Access Driver (*.mdb); DBQ=C:\ Program Files\Microsoft Visual Studio\VB98\BIBLIO.MDB;"
set conn=Server.CreateObject("ADODB.Connection")
conn.open connStr
```

5.4.4 *Cơ sở dữ liệu MS SQL Server*

```
connStr="Provider=SQLOLEDB.1;Persist Security Info=False; User ID=sa;Initial Catalog=AdvWorks;Data Source=localhost"
set conn=Server.CreateObject("ADODB.Connection")
conn.open connStr
```

Nếu nguồn dữ liệu hỗ trợ OLEDB ta có thể sử dụng ADO thông qua ODBC theo cách tổng quát:

```
connStr="Provider=MSDASQL.1;Persist Security Info=False; Data Source=Test"
set conn=Server.CreateObject("ADODB.Connection")
conn.open connStr
```

Trong đó Test là DataSource thiết lập trong System DNS của ODBC

5.5 Sử dụng đối tượng RecordSet

5.5.1 *Tạo RecordSet:*

```
set rs=Server.CreateObject("ADODB.RecordSet")
```

Để trích rút dữ liệu từ một hoặc nhiều bảng nào đó trong cơ sở dữ liệu, ta thường kết hợp đối tượng Connection vào câu lệnh SQL Select trong phương thức Open của RecordSet:

```
rs.open  SQLCommand, Conn
```

5.5.2 *Duyệt qua các bản ghi và truy xuất các trường của bản ghi:*

```
Do While not rs.eof
  Rs.fields]("field_name")
rs.movenext : di chuyển con trỏ của RecordSet đến bản ghi kế tiếp
loop
rs.close
```

```
Các cách truy cập hợp lệ và tương đương:
Rs("field_name")
Rs.fields("field_name")
Rs.fields("field_name").value
Rs(1).value
Rs.fields(1).Value
Nếu muốn duyệt qua tất cả các trường trong bản ghi có thể dùng lệnh:
For each Fld in rs.fields
 Response.write fld.name + ":" + fld.value + "<br>"
Next
```

5.5.3 *Lọc qua các bản ghi trong RecordSet*

Có thể sử dụng mệnh đề Where của câu lệnh Select hoặc có thể sử dụng thuộc tính Filter của RecordSet để chỉ định điều kiện lọc sau khi đã trích xuất dữ liệu.

Ví dụ:

```
sqlStr="Select * from Products"
rs.open sqlStr,conn
rs.filter= " productCode='" & SpecialCode & "'"
```

Thuộc tính Filter cho phép sử dụng mệnh đề lọc gần giống với mệnh đề Where

5.5.4 *Phân trang với đối tượng RecordSet:*

Đối tượng RecordSet cung cấp 3 thuộc tính quan trọng sau để sử dụng phân trang:

PageSize: Kích thước bản ghi trong một trang

PageCount: Tổng số trang RecordSet truy vấn được

AbsolutePage: Chỉ định trang hiện hành đang cần được đọc

Để RecordSet có khả năng phân trang, cần thiết lập thêm tham số cho RecordSet trước khi thực hiện truy vấn:

Rs.CursorLocation=3 ' Có thể sử dụng hằng adUseClient

Rs.PageSize=15 ' 15 bản ghi trong một trang

Tiết theo mở đối tượng RecordSet truy vấn dữ liệu với tuỳ chọn là các hằng adOpenForwardOnly(0), adLockReadOnly (1) truy cho phương thức Open như sau:

Rs.open sqlStr, Conn, 0,1

Công việc sau cùng là định vị trang thông qua thuộc tính AbsolutePage. Ta lưu lại vị trí hiện hành của trang dữ liệu thông qua giá trị chứa trong thẻ <input hidden>. Giá trị này sẽ được chuyển về trình chủ mỗi khi người dùng kích vào.

Xét ví dụ sau:

```
<%sqlStr="SELECT * FROM Products "
 ' page navigate session here ..........
      Dim lCurrentPage
      Dim lPageCount
      lCurrentPage = CLng(Request("page"))
      If lCurrentPage < 1 Then
```

```
                lCurrentPage = 1
        End If
        rs.CursorLocation = 3
        rs.PageSize = 15
rs.Open sqlStr, conn, 0, 1 'Const adOpenForwardOnly=0,
adLockReadOnly = 1
        lPageCount = rs.PageCount
        If lCurrentPage > lPageCount Then
                lCurrentPage = lPageCount
        End If
        if not rs.eof then
           rs.AbsolutePage = lCurrentPage
        end if
call ShowPageNavigation(lCurrentPage,lPageCount)
  Do While rs.AbsolutePage = lCurrentPage And Not rs.Eof
    Response.write rs("ProductName")
  rs.movenext
loop %>
<form name="viewFrm" >
  <input type=hidden name=page  >
</form>
<%
Sub ShowPageNavigation (lCurrentPage,lPageCount)
        If lCurrentPage <> 1 AND lCurrentPage <> 0 Then
%>
 <A HREF="javascript:setValue('<%= lCurrentPage - 1 %>');
">Previous
 <% Else %>
  Previous
  <% End If%>
  <%If lCurrentPage < lPageCount Then%>
<A HREF="javascript:setValue('<%= lCurrentPage + 1%>');
">Next
  <% Else %>
  Next
 <% End If%><BR>
  Page <B> <%= lCurrentPage%> </B>
 <%= lPageCount%>
<%End Sub %>
<script language=javascript>
function setValue(page){
```

```
        viewFrm.page.value= page;
        viewFrm.submit();
        }
</script>
```

5.6 Hiệu chỉnh các bản ghi

5.6.1 *Hiệu chỉnh các bản ghi dựa vào RecordSet:*

Thêm mới bản ghi: Để thêm mới vào bảng dữ liệu quản lý bởi phương thức RecordSet sử dụng phương thức AddNew

```
sqlStr=”Select * From Accounts”
rs.open sqlStr,Conn
‘ Thêm tài khoản mới vào bảng Accounts
rs.Addnew
‘gán giá trị cho bản ghi
with rs
 .fields(“username”)= ‘New User’
.fields(“password”)=’***’
end with
‘lưu lại
rs.update
‘Chỉnh sửa nội dung trong bản ghi hiện hành:
sqlStr=”Select * From Accounts”
rs.open sqlStr,Conn
with rs
.fields(“password”)=’newpassword’
end with
‘lưu lại
rs.update
‘Xoá bản ghi hiện hành:
sqlStr=”Select * From Accounts where username= ‘” &mkuser &”’”
rs.open sqlStr,Conn
rs.delete
```

5.6.2 *Hiệu chỉnh các bản ghi bằng câu lệnh SQL với đối tượng connection*

```
sqlStr=”delete * from Accounts where username= ‘” &mkuser &”’”
Conn.execute sqlStr
```

5.7 Sử dụng đối tượng Command

5.7.1 *Tạo đối tượng Command:*

```
set cmdUpdate=Server.CreateObject(“ADODB.Command”)
```

5.7.2 *Sử dụng đối tượng Command:*

```
sqlUpdate=“update accounts set password = ‘abc’ where
username=”‘” & username & “’”
```

```
cmdUpdate.ActiveConnection=strConn
cmdUpdate.CommandText=strUpdate
cmdUpdate.CommandType=adcmdText
cmdUpdate.Execute
```

Đối tượng Command cũng được dùng để nhận kết quả trả về từ câu lệnh Select hoặc từ một tên bảng dữ liệu, ví dụ để lấy toàn bộ nội dung bảng dữ liệu Accounts, ta chỉ cần chỉ ra tên bảng và mở RecordSet dựa vào đối tượng Command:

```
cmdTable.ActiveConnection=strConn
cmdTable.CommandText="Accounts"
cmdTable.CommandType=adCmdTable
rs.open cmdTable
```

5.8 Bài tập chương 5

Bài 1: Chọn chủ đề để thiết kế website

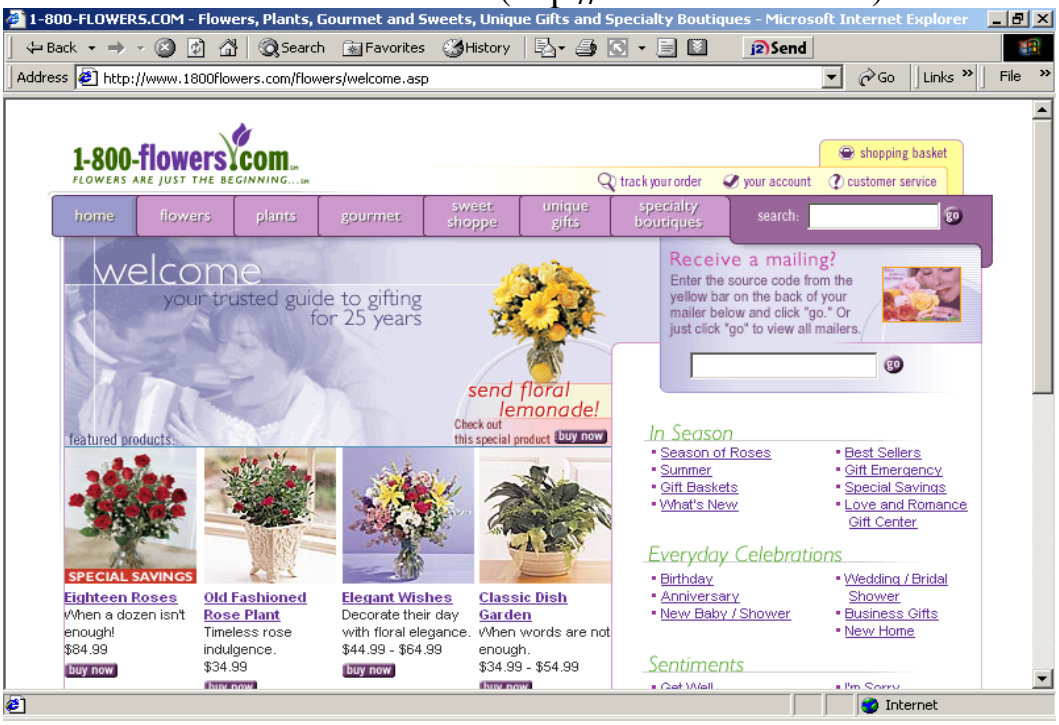
1. Website về dịch vụ việc làm.
2. Website về dịch vụ nhà đất (http://www.nhadat.com).
3. Website báo điện tử (http://vnexpress.net)
4. Website trường học.
5. Website dịch vụ giải trí như ECards, Điện hoa, ...

Bài 2: Xây dựng mô hình ứng dụng bán hàng qua mạng, dựa trên cơ sở dữ liệu, qua đó vận dụng các đối tượng của ADODB để có thể kết nối với cơ sở dự liệu sao cho có thể thể hiện và cung cấp được thông tin cho khách hàng. Chủ yếu giới thiệu cách thức thiết kế một ứng dụng Web có các chức năng sau:

- Xem thông tin chi tiết của một mặt hàng.
- Lựa chọn hàng, thêm vào, bớt ra khỏi giỏ hàng.
- Tính tiền.

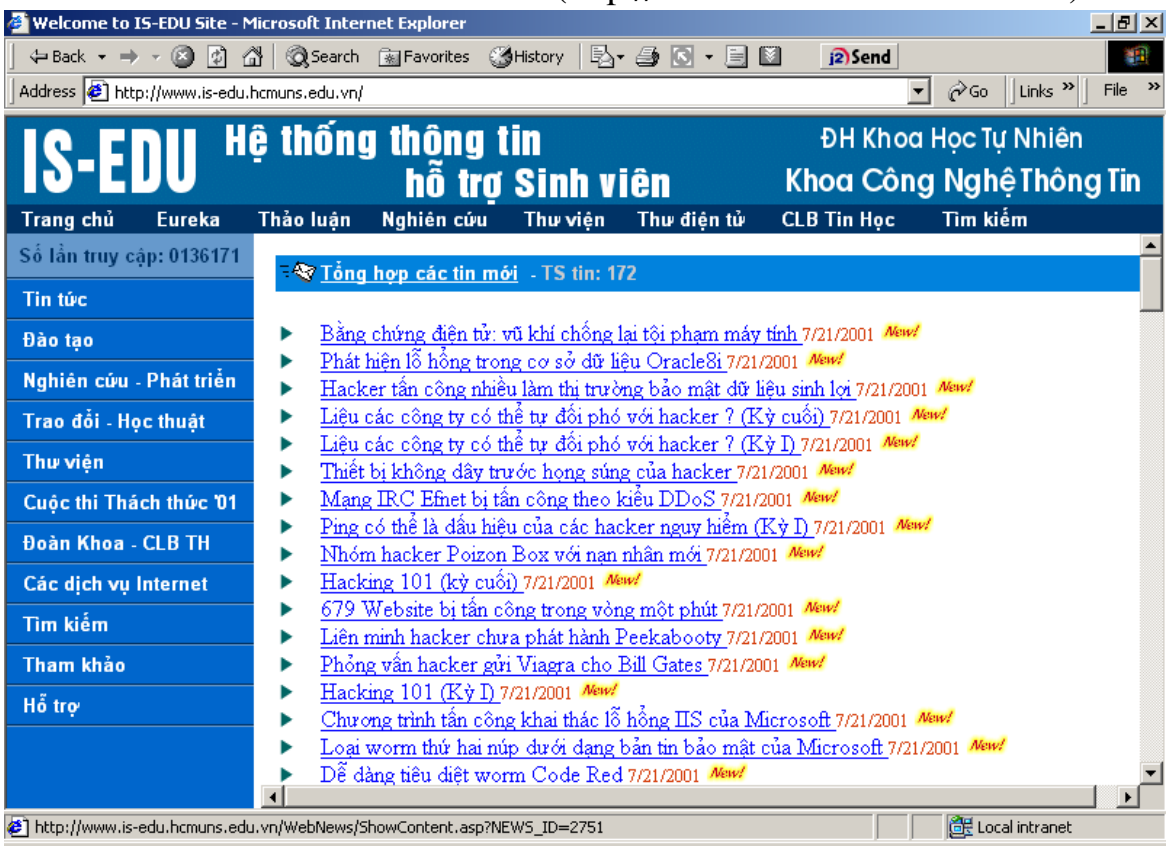
Bài 3: Bài tập tổng hợp, thiết kế một số Web site theo mẫu:

1. Thiết kế website theo mẫu sau (http://www.flowers.com)



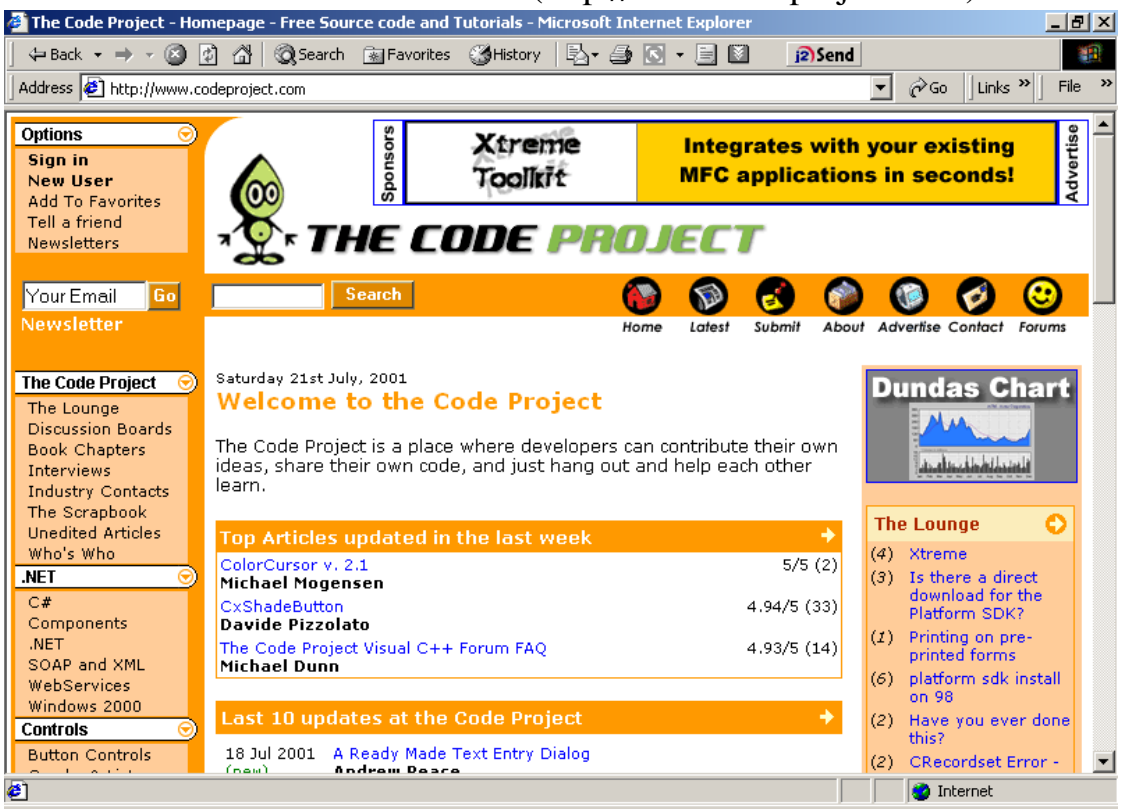
2. Thiết kế website theo mẫu sau (http://shopping.yahoo.com)

3. Thiết kế website theo mẫu sau (http://www.is-edu.hcmuns.edu.vn)



4. Thiết kế website theo mẫu sau (http://vnexpress.net)

5. Thiết kế website theo mẫu sau (http://www.codeproject.com)



TÀI LIỆU THAM KHẢO

1. ĐỖ MAI HƯƠNG: Xây dựng Web động dựa trên công nghệ ASP
   Học viện kỹ thuật quân sự - 2000.
2. NGUYỄN PHƯƠNG LAN: ASP 3.0/ASP.NET
   Nhà xuất bản Lao động xã hội - 2003.
3. PHẠM PHÚ TÀI: Javascript
   Mediaspace Club ấn hành nội bộ - 2003.
4. NGUYỄN THỊ THANH TRÚC: Giáo trình thiết kế và lập trình Web với ASP
   Đại học quốc gia Tp HCM - 2003.
5. NGUYỄN HỮU TUẤN: Giáo trình thiết kế Web
   Đại học quốc gia Hà Nội - 2001.
6. Giáo trình xây dựng và quản trị Website
   Trung tâm điện toán và truyền số liệu Khu vực I - 2003.
7. Nhóm tác giả ELICOM: Xây dựng Web động với ASP
   Nhà xuất bản Thống kê - 2000.
8. MICHAEL CORNING: Working with Active Server Page
   Que Corporation - 1997.
9. Learning VBScript
   XtraNet Communications INC, - 2000

MỤC LỤC

Lời nói đầu **1**

Chương 1 GIỚI THIỆU CHUNG **2**
1.1 Mạng máy tính **2**
1.1.1 Định nghĩa ***2***
1.1.2 Phân loại ***2***
1.2 Internet **3**
1.3 Các giao thức Internet **4**
1.3.1 Giao thức điều khiển phiên truyền ***4***
1.3.2 Giao thức Internet ***4***
1.3.3 Giao thức gam dữ liệu người dùng ***5***
1.3.4 Giao thức phân giải địa chỉ ***5***
1.3.5 Giao thức hệ thống tên miền ***5***
1.3.6 Giao thức chuyển thư đơn giản ***6***
1.3.7 Giao thức truyền tập tin ***6***
1.3.8 HTTP - HyperText Transfer Protocol ***6***
1.4 Địa chỉ IP **6**
1.5 Các khái niệm khác **7**
1.5.1 URL ***7***
1.5.2 Hyperlink (siêu liên kết) ***7***
1.5.3 Web Browser (trình duyệt web) ***8***
1.5.4 Web Server (máy chủ Web) ***8***
1.5.5 Web Site ***9***
1.5.6 World Wide Web ***9***
1.5.7 Phân biệt Inetrnet và WWW ***9***
1.5.8 Web page ***9***
1.6 Cách thức tổ chức và xây dựng một Web Site **9**
1.7 Phân loại Web **10**
1.7.1 Static pages (Web tĩnh): ***10***
1.7.2 Form pages (Mẫu biểu): ***10***
1.7.3 Dynamic Web (Web động) ***10***
1.8 Câu hỏi và bài tập chương 1 ..... **10**

Chương 2 LẬP TRÌNH WEB VỚI NGÔN NGỮ ĐÁNH DẤU SIÊU VĂN BẢN **11**
2.1 Khái niệm ngôn ngữ HTML **11**
2.2 Lập trình web với ngôn ngữ HTML **11**
2.2.1 Các thẻ định dạng cấu trúc của HTML ***11***
2.2.2 Các thẻ định dạng khối ***13***
2.2.3 Các thẻ định dạng danh sách ***14***
2.2.4 Các thẻ định dạng ký tự ***15***
2.2.5 Các thẻ chèn âm thanh, hình ảnh ***21***
2.2.6 Chèn bảng ***25***
2.2.7 Sử dụng Khung – Frame ***26***
2.2.8 FORMS ***30***
2.3 DHTML (Dynamic HTML) **33**
2.3.1 Định nghĩa: ***33***
2.3.2 Đặc điểm ***33***

2.3.3 Một số hiệu ứng DHTML *34*
2.4 Câu hỏi và bài tập chương 2 ..... **35**

Chương 3 NGÔN NGỮ KỊCH BẢN TRONG LẬP TRÌNH WEB **37**
3.1 JavaScript **37**
3.1.1 Tổng quan *37*
3.1.2 Sử dụng JavaScript *39*
3.1.3 Các kiểu dữ liệu trong JavaScript: *43*
3.1.4 Tạo biến trong JavaScript: *43*
3.1.5 Làm việc với biến và biểu thức: *44*
3.1.6 Cấu trúc điều kiện if – else *46*
3.1.7 Hàm và đối tượng *49*
3.1.8 Tạo đối tượng trong JavaScript *52*
3.1.9 Sự kiện trong JavaScript *57*
3.1.10 Sử dụng vòng lặp trong JavaScript *61*
3.1.11 Sử dụng đối tượng Windows *62*
3.1.12 Làm việc với status bar *64*
3.1.13 Mở và đóng các cửa sổ *64*
3.1.14 Sử dụng đối tượng string *66*
3.2 VBScript **66**
3.2.1 VBScript là gì? *66*
3.2.2 Biến và phạm vi biến *66*
3.2.3 Các kiểu dữ liệu *68*
3.3 Câu hỏi và bài tập chương 3 ..... **75**
3.3.1 Câu hỏi ôn tập *75*
3.3.2 Bài tập lập trình với các ngôn ngữ kịch bản *75*

Chương 4 LẬP TRÌNH WEB ĐỘNG VỚI CÔNG NGHỆ ASP **76**
4.1 Một số khái niệm cơ bản về ASP **76**
4.1.1 Khái niệm Web động *76*
4.1.2 ASP là gì? *76*
4.1.3 Scripting? *77*
4.1.4 Tạo và xem một file ASP *78*
4.1.5 Server-side Includes: *81*
4.2 Ưu điểm của việc sử dụng ASP tạo Web động **82**
4.2.1 Đơn giản, dễ học và hiệu quả: *82*
4.2.2 Bảo mật được mã: *82*
4.2.3 Bảo trì dễ dàng: *82*
4.3 Cài đặt IIS và tạo thư mục ảo cho ứng dụng **83**
4.3.1 1. Cài đặt IIS *83*
4.3.2 Tạo thư mục ảo: *83*
4.4 Cấu trúc và các dòng lệnh cơ bản của ASP **85**
4.4.1 Các thành phần được dùng trong trang ASP *85*
4.4.2 Biến trong ASP *85*
4.4.3 Các lệnh cơ bản của ASP *85*
4.4.4 Vòng lặp For: *87*
4.4.5 Câu lệnh lặp không xác định: *87*
4.5 Xây dựng các hàm và thủ tục trong ASP: **87**
4.6 Sử dụng các đối tượng của ASP để trao đổi thông tin giữa Client và Server. **88**

4.6.1 Giới thiệu các đối tượng chính của ASP: 88
4.6.2 Đối tượng Request....89
4.6.3 Đối tượng Response 94
4.6.4 Đối tượng Server 99
4.6.5 Đối tượng Application....103
4.6.6 Đối tượng Session 106
4.7 Câu hỏi và bài tập chương 4....108
4.7.1 Câu hỏi ôn tập....108
4.7.2 Bài tập về các cấu trúc điều khiển và vòng lặp. 108
4.7.3Bài tập về các đối tượng....109

Chương 5 KẾT NỐI CSDL TRONG LẬP TRÌNH WEB ĐỘNG VỚI ASP....110
5.1 Khái niệm về ADO 110
5.2 Trình tiêu thụ (consumer) và trình cung cấp (provider) 110
5.3 Mô hình đối tượng ADO 111
5.3.1 Đối tượng kết nối (Connection)....111
5.3.2 Đối tượng Command: 111
5.3.3 Đối tượng RecordSet:....111
5.4 Kết nối với nguồn dữ liệu 111
5.4.1 Tạo một ODBC DSN....111
5.4.2 Cơ sở dữ liệu MS Access....112
5.4.3 Cơ sở dữ liệu MS Access thông qua trình điều khiển ODBC 112
5.4.4 Cơ sở dữ liệu MS SQL Server....112
5.5 Sử dụng đối tượng RecordSet 112
5.5.1 Tạo RecordSet:....112
5.5.2 Duyệt qua các bản ghi và truy xuất các trường của bản ghi:....112
5.5.3 Lọc qua các bản ghi trong RecordSet....113
5.5.4 Phân trang với đối tượng RecordSet:....113
5.6 Hiệu chỉnh các bản ghi 115
5.6.1 Hiệu chỉnh các bản ghi dựa vào RecordSet: 115
5.6.2 Hiệu chỉnh các bản ghi bằng câu lệnh SQL với đối tượng connection 115
5.7 Sử dụng đối tượng Command 115
5.7.1 Tạo đối tượng Command:....115
5.7.2 Sử dụng đối tượng Command: 115
5.8 Câu hỏi và bài tập chương 5....116

Tài liệu tham khảo 119

Mục lục....120