Web design

CAO THANH SƠN

THIẾT KẾ VÀ LẬP TRÌNH WEB

NĂM 2007

CÁC KÝ HIỆU, VIẾT TẮT

Ký hiệu, viết tắt	Chú thích
ADO.NET	Microsoft's ActiveX Data Objects.Net
API	Application Programming Interface
ASP.NET	Active Server Pages.NET
C#	C-Sharp
CLR	Common Language Runtime
CTS	Common Type System
FTP	File Transfer Protocol
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IE	Internet Explorer
IIS	Internet Information Services
MSIL	Microsoft Intermediate Language
RAD	Rapid Application Development
SQL	Structure Query Language
URL	Uniform Resource Locator
XML	Extensible Markup Language

MỤC LỤC

PHẦN I. NHỮNG KHÁI NIỆM CƠ BẢN 5

CHƯƠNG 1. GIỚI THIỆU CHUNG 5

1.1. Một số khái niệm cơ bản 5

1.2. Giới thiệu các thẻ HTML 8

CHƯƠNG 2. THIẾT KẾ CÁC ỨNG DỤNG WEB 13

2.1. Xác định nhu cầu 13

2.2. Tổ chức thông tin 15

2.3. Thiết kế cấu trúc ứng dụng web 15

2.4. Phát triển các lược đồ duyệt các trang web 15

PHẦN II. MỘT SỐ CÔNG CỤ THIẾT KẾ WEB 18

CHƯƠNG 3. GIỚI THIỆU Microsoft FrontPage 2003 18

3.1. Giới thiệu 18

3.2. Khởi động Microsoft FrontPage 2003 18

3.3. Một số thao tác với tệp 19

3.4. Đặt thuộc tính cho trang 19

3.5. Định dạng font, định dạng đoạn 21

3.6. Tạo siêu liên kết 22

3.7. Tạo các điểm dừng (Bookmark) trong trang 23

3.8. Chèn các đối tượng vào trang web 23

3.9. Chèn bảng vào trang web 23

CHƯƠNG 4. GIỚI THIỆU ASP.NET 25

4.1. Giới thiệu ASP.NET 25

4.2. Sự khác biệt giữa ASP.NET và ASP 25

4.3. Sơ lược về .NET Framework 25

4.4. Khởi động Microsoft Visual Studio 2005 26

4.5. Tạo một Web site mới 27

4.6. Tạo Master Page 29

CHƯƠNG 5. GIỚI THIỆU NGÔN NGỮ LẬP TRÌNH C# 32

5.1. Giới thiệu 32

5.2. Biến và Hằng 32

5.3. Kiểu dữ liệu tiền định nghĩa 33

5.4. Câu lệnh điều kiện 36

5.5. Vòng lặp (Loops) 36

5.6. Mảng (Arrays) 38

5.7. Sử dụng các ghi chú 39

5.8. Từ định danh và từ khoá 40

CHƯƠNG 6. CÁC ĐIỀU KHIỂN, ĐỐI TƯỢNG TRONG ASP.NET 41
6.1. Một số điều khiển cơ bản 41
6.2. Điều khiển kiểm tra dữ liệu nhập vào 46
6.3. Các đối tượng trong ASP.NET..... 47

CHƯƠNG 7. TRUY CẬP CƠ SỞ DỮ LIỆU .NET..... 51
7.1. Tổng quan về ADO.NET 51
7.2. Sử dụng các Database Connection..... 54
7.3. Sử dụng hiệu quả các Connection..... 55
7.4. Các Transaction (giao dịch) 57
7.5. Commands..... 58
7.6. Executing Commands 59
7.7. Data Tables..... 60
7.8. Tạo một DataSet..... 64

PHỤ LỤC 67
I. CẤU HÌNH WEBSERVER..... 67
II. NHÚNG ĐOẠN JAVASCRIP VÀO TRANG WEB 70

PHẦN I. NHỮNG KHÁI NIỆM CƠ BẢN

CHƯƠNG 1. GIỚI THIỆU CHUNG

Nội dung:

- Một số khái niệm cơ bản
- Giới thiệu các thẻ HTML

1.1. Một số khái niệm cơ bản

1.1.1. Một trang Web

- Về khía cạnh nào đó trang Web giống một trang văn bản.
- Là bộ sưu tập gồm văn bản, hình ảnh, âm thanh,… được tổ chức một cách liên tục.
- Độ dài trang Web không giới hạn về mặt vật lý.
- Có khả năng liên kết trực tiếp với các trang Web khác.
- Thiết kế trên bất kỳ phần mềm soạn thảo văn bản nào.
- Ứng dụng Web tồn tại 2 dạng:
    - Web tĩnh: Có kịch bản ở trình khách
    - Web động: Có kịch bản ở trình chủ
- Cho dù Web tĩnh hay Web động, khi trình bày trên trình duyệt chỉ ở dạng các thẻ HTML.
- Website động và website tĩnh khác nhau như thế nào?

Website Tĩnh	Website động
Ưu điểm	
o Tốc độ truy cập nhanh. o Các máy chủ tìm kiếm dễ nhận diện website.	o Người quản trị dễ dàng thay đổi cập nhật thông tin bất cứ lúc nào một cách đơn giản, gần như tất cả những người dùng internet đều có thể làm được. o Có thực hiện những vấn đề phức tạp có thể là tính hóa đơn, quản lý đơn hàng, thanh toán online, so sánh, tìm kiếm sản phẩm theo yêu cầu cụ thể ... o Số lượng các trang phụ thuộc vào số lượng thông tin mà khách hàng cập nhật, các trang này sẽ tự động phát sinh theo các mục tương ứng và có liên kết với nhau.
Nhược điểm	
o Thay đổi thông tin khó khăn (mất nhiều thời gian và đòi hỏi có một số kỹ năng sử dụng html, phần mềm ftp). Do	o Tốc độ truy cập chậm hơn website tĩnh lý do là mã lệnh của website động cần webserver biên dịch mã lệnh lập trình thành các thẻ html (HyperText Make up

không có mã lệnh lập trình vì vậy việc cập nhật, thay đổi nội dung thông tin của website mang nặng tính thủ công nên cần nhiều thời gian. o Số lượng các trang thông tin theo lý thuyết là không giới hạn nhưng với số trang càng lớn càng tốn nhiều thời gian chẳng hạn cần thêm một trang thông tin thì phải sửa tất cả những trang còn lại.	Language_ngôn ngữ đánh dấu siêu văn bản) rồi mới chuyển đến máy của người lướt web.
Cách thức cập nhật thông tin	
o Xử lý trực tiếp vào các file html thông qua tài khoản ftp đưa lên internet.	o Thông qua tài khoản quản trị admin, khi đăng nhập sẽ xuất hiện chức năng công cụ quản trị tương ứng với quyền hạn của mỗi người quản trị. Điều này làm cho việc kiểm soát thông tin cũng như cập nhật, thay đổi rất đơn giản.

1.1.2. Trang chủ (Home Page)

- Là một trang Web đặc biệt, là điểm vào của một Website.
- Tạo ấn tượng đầu tiên với người duyệt Web
- Liên kết đến các trang khác.
- Chứa các nội dung chính

1.1.3. Website

- Website là một tập các trang Web được kết nối với nhau bằng các siêu liên kết.
- Để một website hoạt động được cần phải có 3 yếu tố cơ bản:
  - Cần phải có tên miền (domain).
  - Nơi lưu trữ website (hosting).
  - Nội dung các trang web hoặc cơ sở dữ liệu thông tin

1.1.4. Khái niệm HTML

- HTML (HyperText Markup Language)
- HTML không phải là một ngôn ngữ lập trình như Pascal, C,… Nó không thể tạo ra các chương trình ứng dụng dùng trực tiếp ngôn ngữ máy.
- HTML là ngôn ngữ đánh dấu siêu văn bản để tạo ra các liên kết giữa các trang văn bản đa dạng với nhau và liên kết với các Multimedia như phim, hình ảnh, âm thanh,…

1.1.5. Trình duyệt Web (Web Browser)

- Phần mềm phiên dịch đánh dấu của các file bằng HTML, định dạng chúng sang các trang Web, và thể hiện chúng cho người dùng.
- Để có thể thể hiện được một trang Web đúng ý nghĩa, cần phải có một trình duyệt Web.
- Các trình duyệt Web làm cho Internet trở nên thân thiện và dễ sử dụng hơn với người dùng.
- Một số trình duyệt thông dụng hiện nay: Internet Explorer, Netscape, Mozilla Firefox,…

1.1.6. Giao thức (Protocol)

- Giao thức là một phương thức truy cập Web của trình duyệt.
- Http: (HyperText Transfer Protocol: giao thức truyền tải siêu văn bản). Đây là giao thức cơ bản mà World Wide Web sử dụng. HTTP xác định cách các thông điệp (các file văn bản, hình ảnh đồ hoạ, âm thanh, video, và các file multimedia khác) được định dạng và truyền tải ra sao, và những hành động nào mà các Web server (máy chủ Web) và các trình duyệt Web (browser) phải làm để đáp ứng các lệnh rất đa dạng. Chẳng hạn, khi bạn gõ một địa chỉ Web URL vào trình duyệt Web, một lệnh HTTP sẽ được gửi tới Web server để ra lệnh và hướng dẫn nó tìm đúng trang Web được yêu cầu và kéo về mở trên trình duyệt Web. Hay nói cách khác, HTTP là giao thức truyền tải các file từ một Web server vào một trình duyệt Web để người dùng có thể xem một trang Web đang hiện diện trên Internet.
- File:// là giao thức truy cập trang Web ngay trên máy của người dùng.
- FTP: (File Transfer Protocol: giao thức truyền tệp) là một giao thức dùng để tải lên (upload) các file từ một trạm làm việc (workstation) hay máy tính cá nhân tới một FTP server hoặc tải xuống (download) các file từ một máy chủ FTP về một trạm làm việc (hay máy tính cá nhân). Đây là cách thức đơn giản nhất để truyền tải các file giữa các máy tính trên Internet. Khi tiếp đầu ngữ ftp xuất hiện trong một địa chỉ URL, có nghĩa rằng người dùng đang kết nối tới một file server chứ không phải một Web server, và một hình thức truyền tải file nào đó sẽ được tiến hành. Khác với Web server, hầu hết FTP server yêu cầu người dùng phải đăng nhập (log on) vào server đó để thực hiện việc truyền tải file. FTP hiện được dùng phổ biến để upload các trang Web từ nhà thiết kế Web lên một máy chủ host trên Internet, truyền tải các file dữ liệu qua lại giữa các máy tính trên Internet, cũng như để tải các chương trình, các file từ các máy chủ khác về máy tính cá nhân. Dùng giao thức FTP, bạn có thể cập nhật (xóa, đổi tên, di chuyển, copy) các file tại một máy chủ.

1.1.7. URL

- URL (Uniform Resource Locator) dùng để chỉ tài nguyên trên Internet. Sức mạnh của web là khả năng tạo ra những liên kết siêu văn bản đến các thông tin liên quan. Những thông tin này có thể là những trang web khác, những hình ảnh, âm thanh... Những liên kết này thường được biểu diễn bằng những chữ màu xanh có gạch dưới được gọi là anchor.
- Các URL có thể được truy xuất thông qua một trình duyệt (Browser) như IE hay Netscape.
- Ví dụ: Một URL có dạng http://www.vinhuni.edu.vn

  Nhờ địa chỉ URL mà ta có thể từ bất kỳ một máy nào trong mạng Internet truy nhập tới các trang web ở các website khác nhau.

1.2. Giới thiệu các thẻ HTML

1.2.1. Giới thiệu

Các lệnh của HTML đơn giản chỉ là các mã đánh dấu định dạng gọi là các thẻ (Tags). Bắt đầu thẻ bằng dấu nhỏ hơn "<", kết thúc thẻ bằng dấu lớn hơn ">", trong đó có tên thẻ và thuộc tính của thẻ nếu có. HTML không phân biệt chữ hoa hay chữ thường trong tên thẻ. Tên thẻ không chứa khoảng trống. Giữa các thuộc tính của tên thẻ cách nhau 1 dấu cách trống.

Thẻ HTML có hai loại

- Loại có thẻ mở kèm thẻ đóng

  Cú pháp: <TagName> Content </TagName>

  Ví dụ: <Title>My Website</Title>

  Dòng chữ này đậm
- Loại có thẻ mở, không có thẻ đóng

  Cú pháp: <TagName>

  Ví dụ:
 //xuống dòng, đưa con trỏ về đầu dòng

1.2.2. Các thẻ HTML cơ bản

1.2.2.1. Cấu trúc trang web

`<html>`
`<head>...</head>`
`<body>...</body>`
`</html>`
	Cấu trúc HTML
	Cấu trúc cơ bản cho mọi tài liệu HTML

`<title>...</title>`	**Tiêu đề trang web** Ở bên trong `<head>...</head>`
`<!-- comment -->`	**Chú thích** Các ghi chú và thông tin trong phần body nhưng không được browser hiển thị

`<body>...</body>`	**Thẻ body** Tất cả các thông tin được khai báo trong thẻ này đều có thể xuất hiện trên trang Web.
`<body bgcolor=#XXXXXX>`	**Màu Nền Cố định** Định màu nền cho trang web
`<body background= "filename.gif">`	**Thiết lập ảnh nền** Dùng một hình ảnh làm nền cho trang web

1.2.2.2. Các thẻ định dạng văn bản

`<Hn>...</Hn>`	**Thẻ các mục tiêu đề** Tiêu đề các mục có kích cỡ khác nhau, với n=1..6.

`<b>...</b>`
`<i>...</i>`
`<tt>...</tt>`
`<u>...</u>`
`<strike>...</strike>`
	Kiểu chữ
	Đặt kiểu chữ **đậm**, *nghiêng*, hay `typewriter` (chữ đánh máy). Kiểu chữ gach dưới và ~~gạch ngang~~

`<address>...</address>`	**Địa chỉ** Văn bản ở cuối được in nghiêng
`<blockquote>...` `  </blockquote>`	**Blockquote** Văn bản thụt vào trong cho các chú giải
`<pre>...</pre>`	**Văn bản Preformatted** Hiển thị dạng chữ đánh máy giữ nguyên các khoảng trắng và dấu xuống dòng.
`&xxxx;`	**Ký tự Đặc biệt** Mã cho ký tự đặc biệt và các dấu phụ
`<font size=X>...` `</font>`	**Kích thước Phông** Đổi kích thước của đoạn văn với X=1..7.
`<big>...</big>` `<small>...</small>`	**Kích thước Phông** Đổi kích thước phông lớn hơn hay nhỏ hơn kích thước thông thường.

`<font color=#RRGGBB>`
`...`
`</font>`
	Màu Phông
	Đổi màu đoạn văn được chọn với RRGGBB là mã màu hệ thập lục phân của trị màu RGB.

`<sup>...</sup>` `<sub>...</sub>`	**Superscript/Subscript** Tạo superscript (ví dụ $x^3 + 2xy + y^2 = 0$) hay subscript (ví dụ H_2SO_4)

1.2.2.3. Phân đoạn và ngắt quãng văn bản

`<p> ... </p>`	**Paragraph** Sang đoạn mới cùng với một dòng trống
` `	**Line Break** Sang dòng mới, không thêm dòng trống
`<hr>`	**Hard Rule** Sang dòng mới và tạo một đường phân cách.

`<center>...`
`  </center>`
`<p align=center>...`
`  </p>`
	Chỉnh lề ở giữa
	Chỉnh tất cả vào giữa trang

`<div align=`
`left|center|right>`
`...`
`</div>`
	Division
	Chỉnh mọi thứ về bên trái, vào giữa, hay về bên phải của trang

`<table>...</table>`
 	
`<table border=X`
`  cellpadding=Y`
`  cellspacing=Z>...`
`  </table>`
 	
`<tr>...</tr>`
`<td`
`align=left|center|right`
`valign=top|middle|bottom`
`rowspan=X colspan=Y`
`</td>`
	Tạo Bảng
	Bảng đơn giản vẽ đường viền có độ dày bằng một điểm đơn và các đường chia cách các phần tử. Bỏ qua thuộc tính `border`, hay thiết lập `border=0` tạo ra bảng không nhìn thấy được.
	Các thuộc tính cho tag `<td>` để chỉnh lề theo hàng và cột bên trong một ô. Các thuộc tính `rowspan` và `colspan` có thể sử dụng để tạo các ô được mở rộng ra hơn so với một ô bình thường.

1.2.2.4. Các thẻ danh sách

`<ul>` `<ol>`
`  <li>` `  <li>`
`  <li>` `  <li>`
`</ul>` `</ol>`
	Danh Sách
	Danh sách không có thứ tự `<ul>` tạo các mục có bullet, danh sách có thứ tự `<ol>` tạo các mục có đánh thứ tự

`<ul type=xxxx>` `<li type=xxxx>`	**Bullet cho Danh sách Không có thứ tự** Định lại dạng bullet mặc định: o `type=circle` ▪ `type=square` • `type=disc`
`<ol type=X>` `<li type=X>`	**Kiểu Đánh thứ tự cho Danh sách Có thứ tự** Sử dụng các ký tự khác nhau cho danh sách: • `type=1` (1, 2, 3...) • `type=A` (A, B, C,...) • `type=a` (a, b, c,...) • `type=I` (I, II, III,...) • `type=i` (i, ii, iii,...)

1.2.2.5. Chèn siêu liên kết

`<a href="file.htm">` `hypertext</a>`	**Liên kết Siêu văn bản Cục bộ** Liên kết đến tài liệu khác trong cùng thư mục

`<a`
`href="data/file.htm">`
`hypertext</a>`
	Liên kết Siêu văn bản Cục bộ
	Liên kết đến tài liệu khác thư mục có tên là "data" nằm trong thư mục có tài liệu HTML gọi

`<a href="../file.htm">` `hypertext</a>`	**Liên kết Siêu văn bản Cục bộ** Liên kết đến tài liệu khác trong thư mục cao hơn một cấp so với thư mục có tài liệu HTML gọi
`<a href="URL">` `hypertext</a>`	**Liên kết Siêu văn bản của Internet** Liên kết đến một Site Internet khác, được chỉ định bởi URL (Uniform Resource Locator)

`<a href="url">`
`<img src="file.gif"`
`border=0></a>`
	Liên kết Siêu Hình ảnh của Internet
	Tạo siêu liên kết cho ảnh. Thêm thuộc tính `border=0` để hủy bỏ hộp bao quanh hình ảnh

`<a href="mailto:` `"abc@xyz.edu">... </a>`	**Liên kết Mail của Internet** Tạo một thông báo email đến địa chỉ được chỉ định

1.2.2.6. Chèn các đối tượng khác

`<img src="file.gif">`
`<img src="file.gif"`
`alt="###"`
`align=top|middle|bottom|`
`left|right`
`height=x width=y>`
	Chèn hình ảnh
	Hiển thị một hình bên trong trang web.
	+ `src` (source) là tên đường dẫn, hay URL của tập tin hình ảnh;
	+ `alt` (alternative) là văn bản hiển thị cho các browser không đồ thị hay khi người sử dụng tắt việc nạp hình;
	+ `align` điều khiển vị trí của hình và văn bản quanh nó (`top/middle/bottom` chỉnh lề một dòng của văn bản theo sau; `left/right` đặt hình vào một bên trang còn bên kia là văn bản);
	+ `height` và `width` là chiều cao và chiều rộng của hình tính bằng điểm.

`<marquee`
`behavior=alternate|scrol`
`l|slide`
`direction=left|right|up|`
`down`
`height="102"`
`width="102">`
`marquee text`
`</marquee>`
	Chèn dòng chữ chạy
	+ behavior: các kiểu chạy
	+ direction: hướng chữ chạy
	+ height: chiều cao
	+ width: độ rộng

<u>Câu hỏi ôn tập</u>:

- Khái niệm một trang web, website, trang chủ, các thẻ HTML
- Sự khác biệt giữa HTTP và FTP?
- Website động và website tĩnh khác nhau như thế nào?

CHƯƠNG 2. THIẾT KẾ CÁC ỨNG DỤNG WEB

Nội dung:

- Xác định nhu cầu
- Tổ chức thông tin
- Thiết kế cấu trúc ứng dụng web
- Phát triển lược đồ duyệt các trang web

2.1. Xác định nhu cầu

- Mục đích của ứng dụng web là gì?

  Bước đầu tiên trong công đoạn thiết kế một Web site là chúng ta đã có những quyết định chắc chắn về việc chúng ta sẽ "xuất bản" cái gì với Web site của mình. Không có chủ định và mục tiêu rõ ràng thì cả web site đó sẽ trở nên lan man, sa lầy và cuối cùng đi đến một điểm khó có thể quay trở lại. Thiết kế cẩn thận và định hướng rõ là những chìa khoá dẫn đến thành công trong việc xây dựng một Web site.

  Trước khi xây dựng một Web site, chúng ta nên:

  - Xác định đối tượng độc giả của web site.
  - Web site có mục đích rõ ràng.
  - Thiết lập các chủ đề chính của web site.
  - Thiết kế các khối thông tin chủ yếu mà web site sẽ cung cấp.

  Chúng ta cũng nên bắt đầu với việc xác định nguồn tài nguyên về nội dung, hình ảnh thông tin mà chúng ta cần đến để tạo nền web site phù hợp với mục đích được đề ra - đó là nguồn thông tin sẽ duy trì cho web site hoạt động sau này nữa.

- Nó được dự định dùng để làm gì?

  + Đào tạo

  Các ứng dụng đào tạo trên cơ sở công nghệ web rất có trình tự trong mặt thiết kế, có rất ít cơ hội để đi lạc đề từ trang giới thiệu chính. Đừng làm độc giả và cả mục đích của chúng ta bị lộn xộn bởi các mối liên kết ra ngoài thông tin chủ chốt. Giới hạn các liên kết bằng nút "Tiếp tục", hay "Quay về trang trước" đảm bảo mọi độc giả sẽ nhìn thấy cùng một giáo trình, cho phép chúng ta dự đoán chính xác hơn thời gian truy nhập của người đọc. Đại đa số giáo trình giả thiết thời gian truy nhập dưới một giờ, hoặc sẽ được phân đoạn thành các phần ít hơn một giờ. Chúng ta cũng nên thông báo cho người đọc về lượng thời gian của bài giảng, hoặc cũng lưu ý họ đừng đi xa khỏi phần chính của bài giảng nếu bài giảng đó cần phải trả tiền để đọc.

Các ứng dụng đào tạo loại này thường yêu cầu sự đăng nhập (login) của độc giả, và cũng thường sử dụng câu hỏi dạng form có trả lời đúng/sai hoặc theo dạng lựa chọn câu trả lời từ một danh sách. Thông tin về đọc giả, bảng điểm được lưu trữ trong cơ sở dữ liệu được liên kết với web site.

+ Dạy học

Trong các ứng dụng dạy học dựa trên công nghệ web, thông tin được trình bày thường tinh tế và có chiều sâu hơn là trong các ứng dụng đào tạo. Các mối liên kết là mặt mạnh của web, tuy nhiên chúng có thể là một sự gây rối cho các học sinh từ trang trình bày chính. Nếu chúng ta cho phép người đọc liên kết đến các tài nguyên web khác ngoài web site của chúng ta, chúng ta nên nhóm các liên kết trong trang này cách biệt khỏi phần thông tin chính. Thông thường người đọc muốn in thông tin trên web và sẽ đọc chúng sau này. Chúng ta nên cung cấp cho họ một phiên bản "in" riêng, trong đó các trang riêng biệt, ngắn sẽ được gộp lại thành một trang dài.

+ Giáo dục

Các độc giả tự học hỏi, tự khám phá sẽ bực mình với phong cách thiết kế quá thu gọn, quá trình tự. Thông thường các độc giả nhóm này thường có trình độ cao. Thiết kế một cấu trúc uyển chuyển, có tương tác, không đơn điệu là lý tưởng đối với các độc giả này, do rất khó đoán định chính xác chủ đề nào sẽ được quan tâm nhất đối với một giáo sư hay với một sinh viên, kỹ sư. Thiết kế cũng phải cho phép truy nhập nhanh đến một phạm vi rộng các chủ đề, và thường cũng rất phong phú với các liên kết đến các thông tin có liên quan, trên web site của chúng ta hay trên các web site khác. Các danh sách dạng text của các liên kết cũng rất thích hợp cho các mục lục, bản chỉ số vì chúng được nạp xuống nhanh, đầy đủ thông tin, nhưng cho nhóm độc giả này lại dễ chán, và thế cần có hình ảnh đồ hoạ thiết kế đẹp, thay đổi cùng các minh hoạ đi kèm thông tin. Thời gian truy nhập không thể dự đoán được, nhưng thường ngắn hơn các site cho đào tạo, giáo dục vì độc giả thường khẩn cấp. Lựa chọn cho in ấn cũng là bắt buộc phải có cho các độc giả này.

+ Tham khảo

Các web site tham khảo được thiết kế tốt cho phép người đọc nhanh chóng đi thẳng vào vấn đề, tìm cái họ cần và sau đó dễ dàng in hoặc lưu giữ cái họ tìm thấy. Thông thường thông tin không phải là các "câu chuyện", do đó cấu trúc của nó hoàn toàn không có trình tự. Cấu trúc menu, nội dung nhất định phải được tổ chức cẩn thận để hỗ trợ tìm kiếm, thu nhận nhanh, dễ lưu giữ các file, in ấn khi cần. Cần giữ các hình ảnh đồ hoạ nhỏ để thời gian nạp xuống nhanh, và chúng ta cũng nên nghiên cứu, áp dụng phần mềm tìm kiếm thay vì chỉ cung cấp một danh sách các liên kết. Thời gian liên kết càng ngắn càng tốt.

- Các mục tiêu của ứng dụng web này là gì?

  Trước tiên cần có một tuyên bố khái quát ngắn và rõ ràng cho các mục tiêu của web site, điều sẽ giúp đỡ rất nhiều cho công việc thiết kế. Nó là điểm xuất phát để chúng ta mở rộng đến các mục tiêu chính, và cũng là một công cụ hữu hiệu để đánh giá sự thành công của một web site. Xây dựng một web site là cả một quá trình liên tục, nó không đơn thuần chỉ là một dự án duy nhất, một lần với các thông tin tĩnh. Việc biên tập, quản lý và duy trì kỹ thuật dài hạn nhất định phải bao trùm lên kế hoạch xây dựng web site. Thiếu điều này, tương lai của một web site sẽ cùng số phận giống như bao nhà văn, nhà báo, đầy lòng say mê buổi ban đầu, nhưng chẳng có kết quả cuối cùng nào cả.

2.2. Tổ chức thông tin

- Tổ chức theo một trật tự nội dung rõ ràng
- Tổ chức theo thứ tự từ điển
- Tổ chức theo thời gian
- Tổ chức theo không gian

2.3. Thiết kế cấu trúc ứng dụng web

- Việc tổ chức thông tin ra sao sẽ quyết định cấu trúc của ứng dụng web.
- Cấu trúc phân cấp:
  - Là cách tiếp cận truyền thống từ trên xuống.
  - Đầu tiên phải xây dựng các hạng mục ở mức cao, sau đó sẽ sắp xếp các tài liệu thuộc các hạng mục con.
- Cấu trúc siêu văn bản:
  - Các văn bản hoặc các bức ảnh được kết nối với các văn bản và đồ hoạ khác.
  - Các kết nối này có thể có ở các vị trí bất kỳ trong trang và tạo ra khả năng chuyển nhanh tới dữ liệu được kết nối.
- Cấu trúc kiểu cơ sở dữ liệu:
  - Xây dựng các trang thông tin từ một cơ sở dữ liệu khi các thông tin này được yêu cầu.

2.4. Phát triển các lược đồ duyệt các trang web

2.4.1. Phát triển các lược đồ

Lược đồ duyệt của ứng dụng web phụ thuộc rất nhiều vào cấu trúc mà ta đã xây dựng.

- Ảnh hưởng trực tiếp đến việc di chuyển của người sử dụng trong trạm Web.
- Việc truy cập vào các thông tin trình bầy.
- Tính đơn giản hay phức tạp khi truy cập các thông tin cũng quyết định rất nhiều đến thành công của trang Web.

2.4.2. Một số cấu trúc Web

a. Cấu trúc tuyến tính

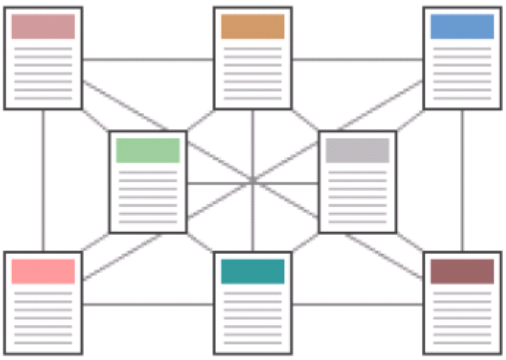
- Đơn giản, hiển thị thông tin một cách tuần tự
- Thông tin được sắp theo thứ tự logic hoặc thời gian
- Nếu nhiều thông tin thì sẽ trở nên phức tạp

b. Cấu trúc phân cấp

- Dễ dàng truy xuất thông tin
- Dễ dàng phân tích, dễ dàng xây dựng
- Cấu trúc rõ ràng

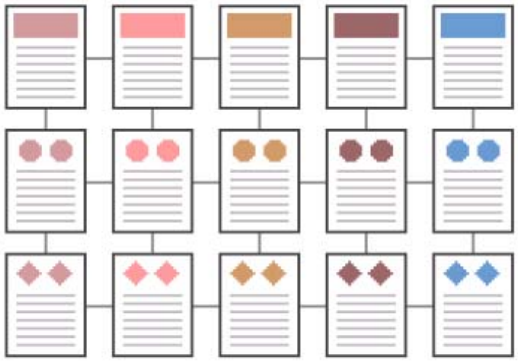
c. Cấu trúc mạng nhện

- Tự khám phá, tự do tưởng đối với độc giả
- Khai thác triệt để năng lực liên kết và kết hợp của Web.
- Khó hiểu, khó dự đoán đối với độc giả truy cập Web.

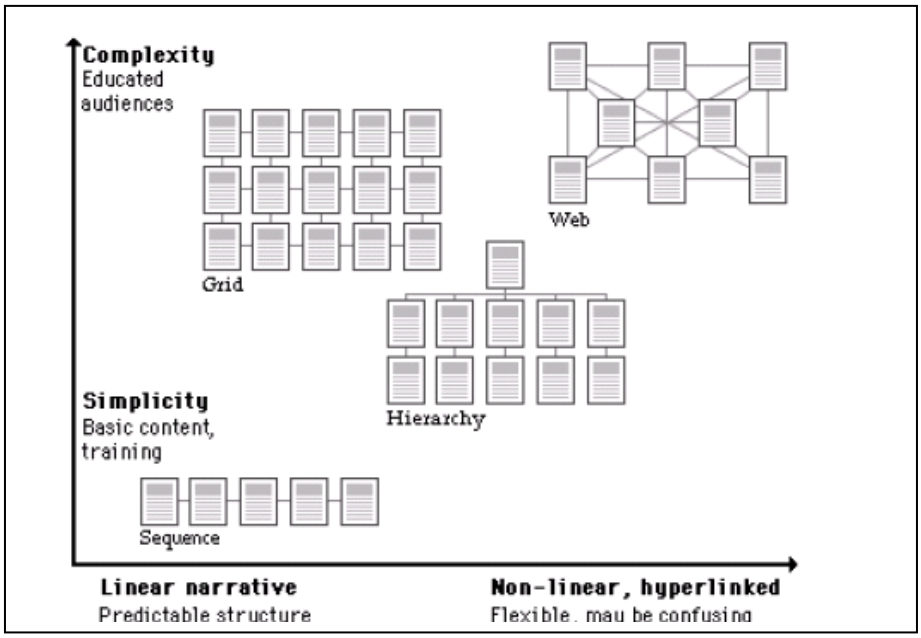


d. Cấu trúc ô lưới

- Tổ chức các thông tin liên quan với nhau.
- Khó hiểu với độc giả không xác định được mối quan hệ giữa những thông tin đó.



e. So sánh các cấu trúc



<u>Câu hỏi ôn tập</u>:

- Bắt đầu xây dựng một ứng dụng web chúng ta cần xác định những yêu cầu gì?
- Các cách tổ chức thông tin khi thiết kế ứng dụng web
- So sánh các cấu trúc web thường dùng
- Nêu một số cấu trúc web không nên sử dụng, vì sao?

PHẦN II. MỘT SỐ CÔNG CỤ THIẾT KẾ WEB

CHƯƠNG 3. GIỚI THIỆU Microsoft FrontPage 2003

Nội dung:

- Giới thiệu
- Khởi động Microsoft FrontPage 2003
- Một số thao tác với tệp
- Đặt thuộc tính cho trang
- Định dạng font, định dạng đoạn
- Tạo siêu liên kết
- Tạo các điểm dừng (Bookmark) trong trang
- Chèn các đối tượng vào trang
- Chèn bảng vào trang web

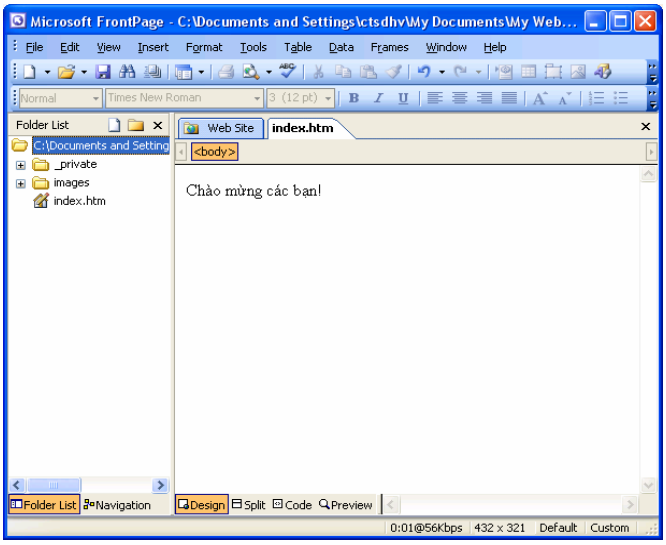
3.1. Giới thiệu

Ngôn ngữ HTML đã trình bày trong phần web tĩnh giúp bạn hiểu được cú pháp của ngôn ngữ tạo trang web và hỗ trợ cho bạn lập trình web động. Trong thực tế bạn không cần thiết phải làm những trang web tĩnh bằng cách công phu gõ vào từng thẻ của HTML vì đã có các công cụ tạo ra trang web một cách trực quan. Bạn chỉ sử dụng HTML chỉ khi nào thấy rằng công cụ của bạn dùng không thể hiện được những điều bạn mong muốn. Sau đây là một số công cụ phổ biến hiện nay: Microsoft FrontPage, Microsoft Word, Dreamweaver,...

Với sự ra đời của các công cụ soạn thảo trang web đã là cho việc tạo ra một trang web không còn khó khăn và mất nhiều thời gian nữa. Vấn đề đặt ra là trang web phải đẹp và trang nhã cùng với những thông tin phong phú. Vấn đề này phụ thuộc hoàn toàn vào sự tổ chức và năng khiếu thẩm mỹ của bạn.

3.2. Khởi động Microsoft FrontPage 2003

- Start/Programs/Microsoft Office/Microsoft Office FrontPage 2003

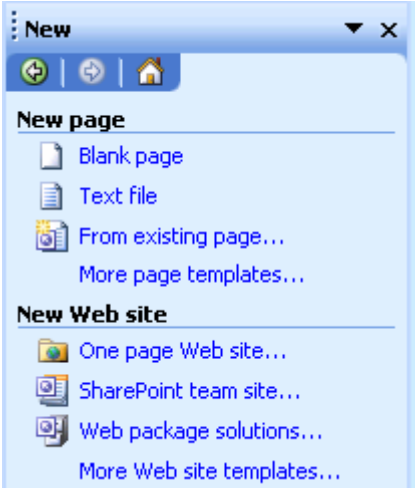


- Chọn cách thể hiện thư mục và trang web: click View và chọn Page (Folder).
- Để xem, sửa các thẻ HTML: Chọn tab Code
- Để soạn thảo trang web không dùng lệnh HTML: Chọn tab Design
- Để xem sơ lược kết quả trang web: Chọn tab Preview
- Để thể hiện chế độ vừa soạn thảo, vừa xem các lệnh HTML: Chọn tab Split

3.3. Một số thao tác với tệp

3.3.1. Tạo một tệp mới

- Vào File, chọn New
- Blank page: Tạo một trang mới
- Text file: tạo một tệp text
- From existing page: tạo một trang mới lấy nội dung từ tệp đã tồn tại
- More page templates.. tạo một trang mới từ các mẫu có sẵn.



3.3.2. Lưu lại tệp

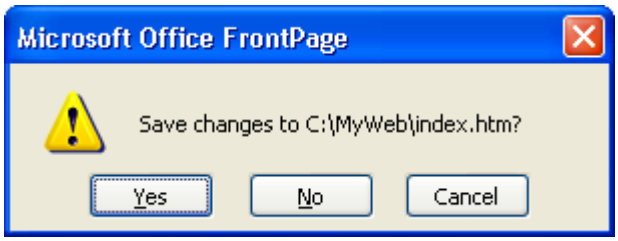
- Vào File, chọn Save (Ctrl + S)
- Gõ tên tệp vào ô File name, chọn Save

3.3.3. Mở tệp đã lưu

- Vào File, chọn Open (Ctrl + O)
- Chọn tệp cần mở, Open

3.3.4. Đóng tệp

- Vào File, chọn Close
- Nếu chưa lưu tệp, xuất hiện hộp thoại
  - Yes: lưu tệp và thoát
  - No: không lưu tệp
  - Cancel: không đóng tệp

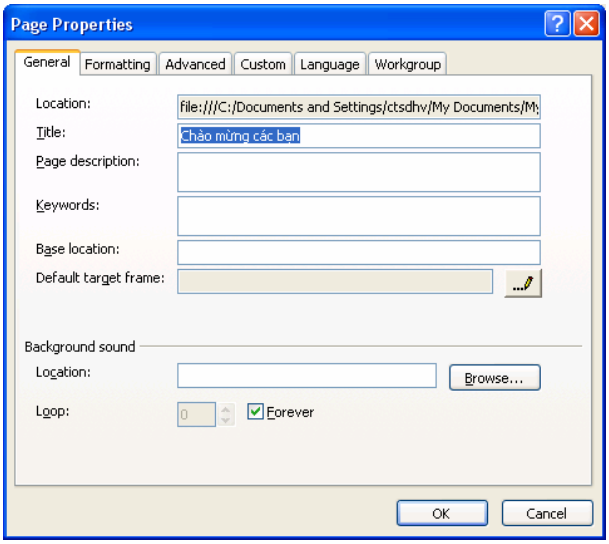


3.4. Đặt thuộc tính cho trang

- Vào File, chọn Properties

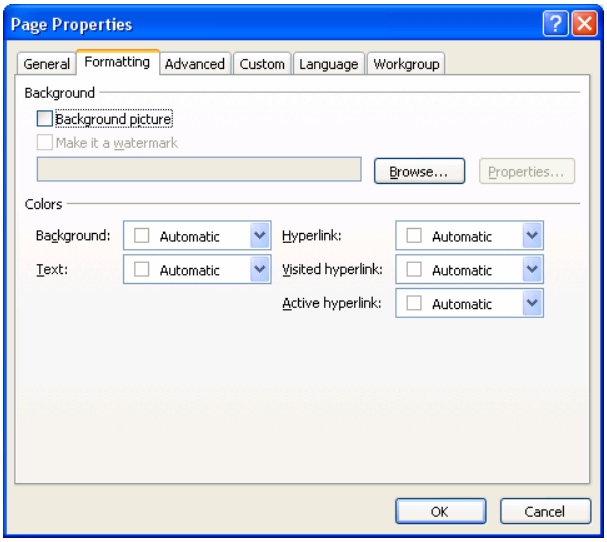
3.4.1. Tab General

- Title: Tiêu đề trang web
- Page description: mô tả trang
- Background sound: đặt nhạc nền cho trang (Chọn vào Forever nếu muốn lặp đi lặp lại nhạc nền)



3.4.2. Tab Formating

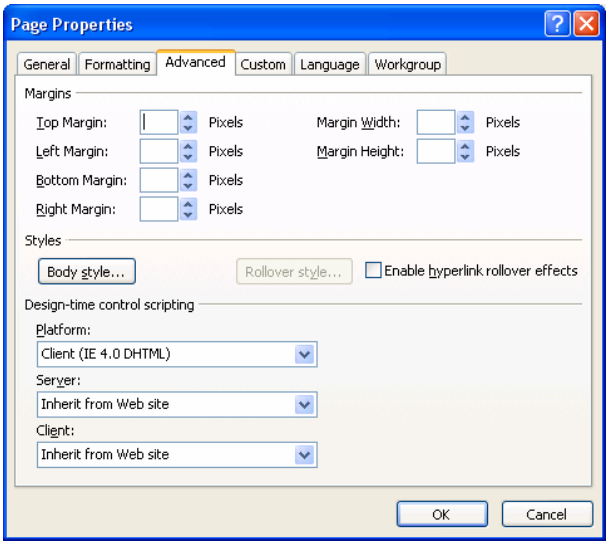
- Background picture: chọn ảnh nền cho trang (Browse: chọn ảnh)
- Color: định dạng màu nền, màu chữ, màu liên kết, ... mặc định



3.4.3. Tab Advanced

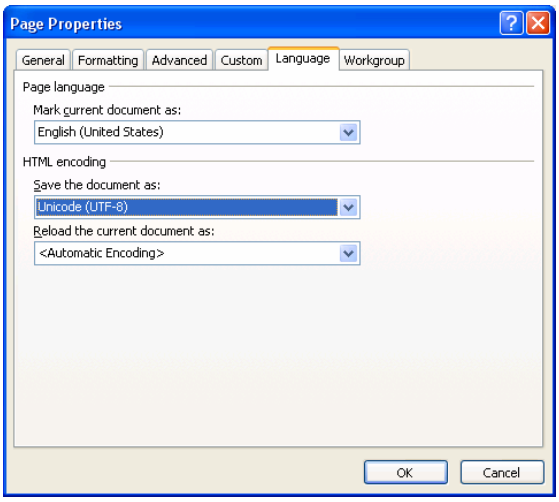
Margin: đặt khoảng cách lề

- Top Margin: khoảng cách lề trên
- Left Margin: Khoảng cách lề trái
- Bottom Margin: Khoảng cách lề dưới
- Right Margin: Khoảng cách lề phải



3.4.4. Tab Language

- Page language: ngôn ngữ mặc định cho trang web
- HTML encoding: chọn Unicode (UTF-8) để lưu định dạng trang web theo Unicode.

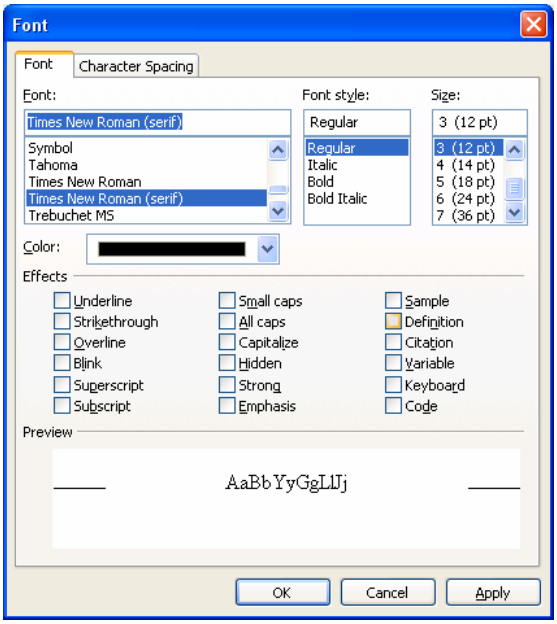


3.5. Định dạng font, định dạng đoạn

3.5.1. Định dạng font

Vào Format, chọn Font

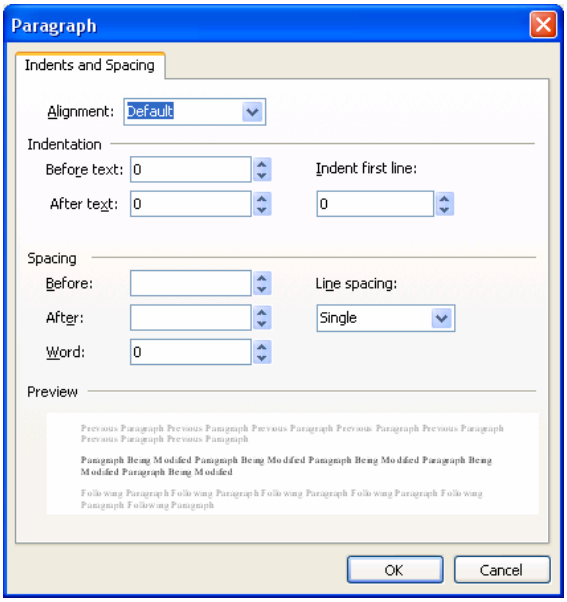
- Font: các font chữ Unicode
- Font Style: các kiểu chữ (Regular: bình thường; Italic: nghiêng; Bold: đậm)
- Size: kích cỡ chữ
- Color: màu chữ



3.5.2. Định dạng đoạn

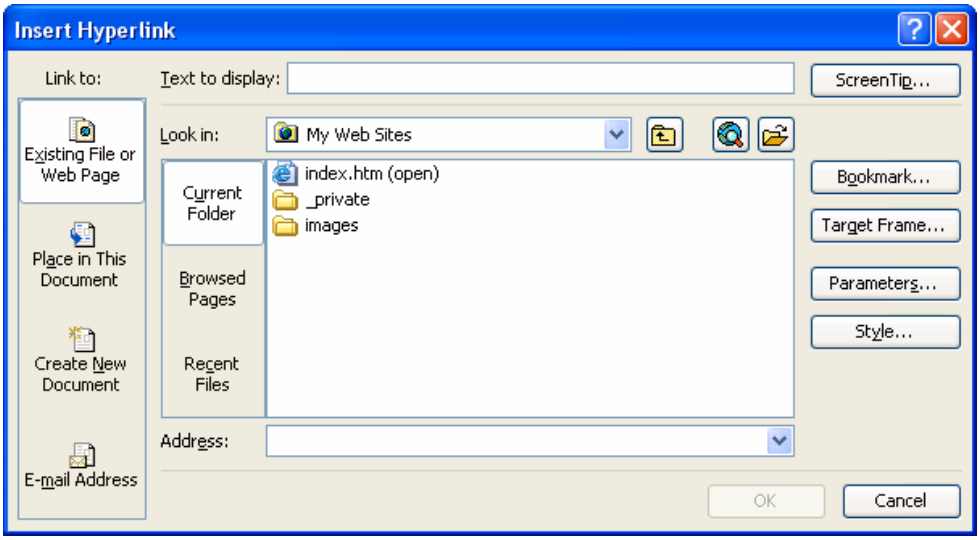
Vào Format, chọn Paragraph

- Alignment: căn lề
- Indentation: xê dịch lề
- Spacing: khoảng cách giữa các đoạn
- Line spacing: khoảng cách giữa các hàng trong đoạn



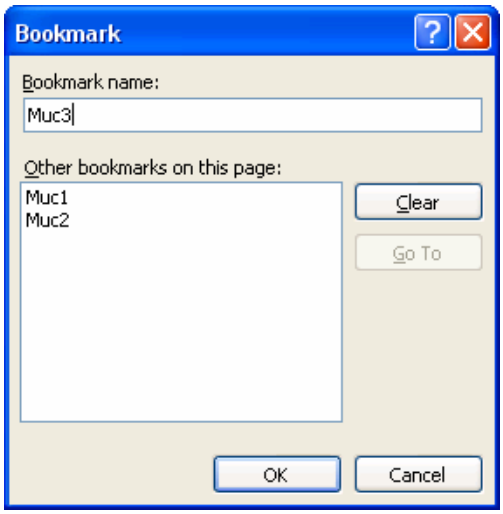
3.6. Tạo siêu liên kết

- Bôi đen vùng cần tạo siêu liên kết
- Vào Insert, chọn HyperLink
- Gõ trang web hoặc URL ở ô Address
- Target Frame:
  - Same Frame: Mở trang liên kết ngay trên trang hiện tại
  - New Windows: Mở trang liên kết trong cửa sổ mới.
- Bookmark: Tạo liên kết tới điểm dừng



3.7. Tạo các điểm dừng (Bookmark) trong trang

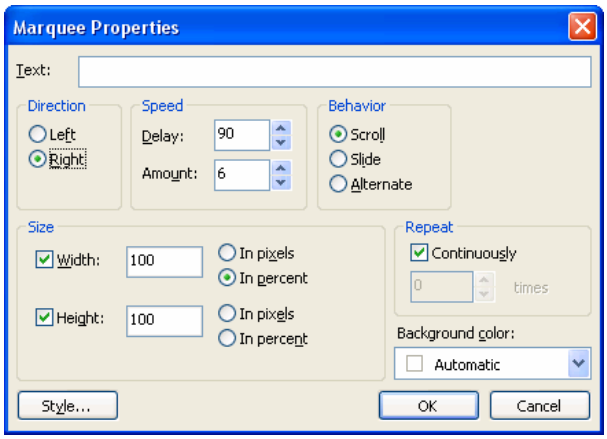
Để tạo liên kết đến các phần nội dung trong cùng 1 trang web ta phải tạo bookmark cho mỗi phần nội dung trong trang web để chỉ cần nhấp vào mục lục các phần nội dung là nhảy ngay đến phần nội dung tương ứng. Muốn tạo các liên kết nội tại trong trang ta phải tạo các điểm dừng (bookmark) trước. Các bước tạo bookmark như sau:



- Di chuyển con trỏ tới vị trí cần tạo điểm dừng
- Chọn chức năng Insert/Bookmark
- Đặt tên cho điểm dừng và click OK. Muốn xoá 1 bookmark nào thì chọn bookmark đã có trong danh sách và nhấp Clear. Muốn nhảy đến 1 bookmark đã định nghĩa thì chọn bookmark đó và click nút Goto.
- Nếu muốn liên kết tới điểm dừng này chỉ cần thực hiện thao tác tạo Hyperlink đến bookmark.

3.8. Chèn các đối tượng vào trang web

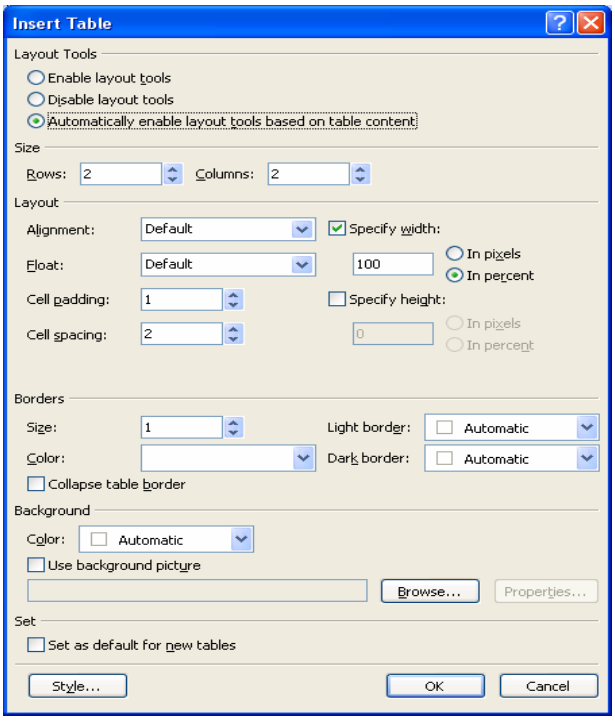
- Chèn ảnh: Insert\Picture\From file
- Chèn một đường ngang: Insert\Picture\Horizontal Line
- Chèn dòng chữ chạy: Insert\Web Component\ Dynamic Effects\Marquee
  - Direction: hướng
  - Speed: tốc độ
  - Behavior: kiểu chạy
  - Size: kích cỡ



3.9. Chèn bảng vào trang web

- Người ta thường dùng table để:
  - Hiển thị các thông tin có dạng dòng/cột, ví dụ như bảng thời khóa biểu, thông tin sản phẩm, ..
  - Trình bày (layout) các văn bản (text) và các ảnh đồ họa (graphics).

- Các bước thực hiện Table\Insert\Table
- Rows: số hàng
- Columns: Số cột
- Layout: định dạng bảng
  - Alignment: Căn lề
  - Specify width: Định độ rộng cho bảng
  - Specify height: Định chiều cao cho bảng
- Border: đường viền cho bảng
- Background
  - Color: màu nền cho bảng
  - User background picture: chọn ảnh nền cho bảng



<u>Bài tập</u>:

Tham khảo các trang web trên internet, xây dựng một số mẫu giao diện web sau:

- Trang web tin tức
- Trang web nghe nhạc
- Trang web bán hàng
- Trang web tìm kiếm

CHƯƠNG 4. GIỚI THIỆU ASP.NET

Nội dung:

- Giới thiệu ASP.NET
- Sự khác biệt giữa ASP.NET và ASP
- Sơ lược về .NET Framework
- Khởi động Microsoft Visual Studio 2005
- Tạo một Web site mới

4.1. Giới thiệu ASP.NET

ASP.NET được viết tắt từ Active Server Pages.NET. Nói đơn giản ASP.NET là một công nghệ có tính cách mạng dùng để phát triển các ứng dụng về mạng hiện nay cũng như trong tương lai. ASP.NET là một phương pháp tổ chức hay khung tổ chức (framework) để thiết lập các ứng dụng hết sức mạnh cho mạng dựa trên CLR (Common Language Runtime) chứ không phải là ngôn ngữ lập trình. Ngôn ngữ lập trình dùng để diễn đạt có thể là VB.NET, C#,...

4.2. Sự khác biệt giữa ASP.NET và ASP

- Tập tin của ASP.NET có phần mở rộng là .ASPX, còn tập tin của ASP là .ASP.
- Tập tin của ASP.NET được phân tích ngữ pháp (parsed) bởi XSPISAPI.DLL, còn tập tin của ASP được phân tích bởi ASP.DLL.
- ASP.NET là kiểu mẫu lập trình phát động bằng sự kiện (event driven), còn các trang ASP được thi hành theo thứ tự tuần tự từ trên xuống dưới.
- ASP.NET sử dụng trình biên dịch (compiled code) nên rất nhanh, còn ASP dùng trình thông dịch (interpreted code) do đó hiệu suất và tốc độ phát triển cũng kém hơn.
- ASP.NET hỗ trợ nhiều ngôn ngữ lập trình mới với .NET và chạy trong môi trường biên dịch (compiled environment), còn ASP chỉ chấp nhận VBScript và JavaScript nên ASP chỉ là một ngôn ngữ kịch bản (scripted language) trong môi trường thông dịch (interpreter environment). Không những vậy, ASP.NET còn kết hợp nhuần nhuyễn với XML (Extensible Markup Language) để trao đổi các thông tin qua mạng.
- ASP.NET hỗ trợ tất cả các trình duyệt (browser) và quan trọng hơn nữa là hỗ trợ các thiết bị di động (mobile devices). Chính các thiết bị di động, mà mỗi ngày càng phổ biến, đã khiến việc dùng ASP trong việc phát triển mạng nhằm vươn tới thị trường mới đó trở nên vô cùng khó khăn.

4.3. Sơ lược về .NET Framework

- Mọi chức năng ASP.NET có được hoàn toàn dựa vào .NET framework, do đó có chữ .NET trong ASP.NET. Ta cần phải hiểu rõ kiến trúc hạ tầng của

.NET framework để dùng ASP.NET một cách hiệu quả, trong đó quan trọng nhất là Common Language Runtime (CLR) và .NET Framework Class.

a) CLR (Common Laguage Runtime)

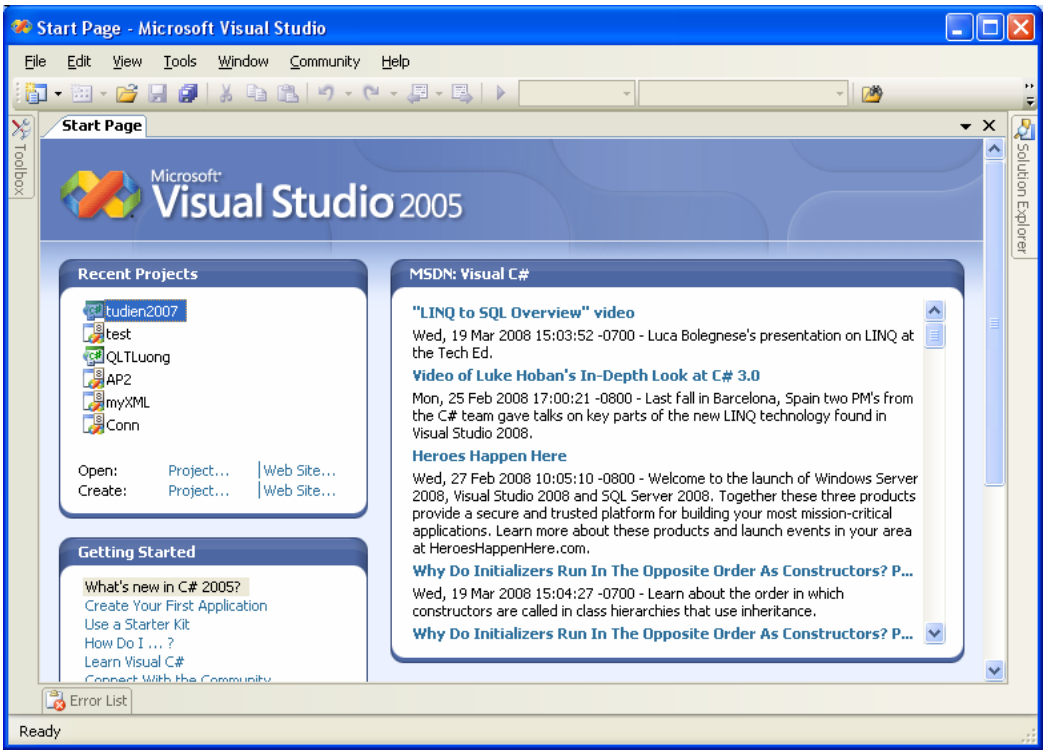
- Là môi trường được dùng để quản lý sự thi hành các mã nguồn mà ta đã soạn ra và biên dịch trong các ứng dụng. Tuy nhiên khi biên dịch mã nguồn, ta lại biên dịch chúng ra thành một ngôn ngữ trung gian gọi là Microsoft Intermediate Language (MSIL). Chính ngôn ngữ trung gian MSIL này là ngôn ngữ chung cho tất cả các ngôn ngữ .NET hiện có. Trong khi biên dịch, các ứng dụng cũng sản xuất ra những thông tin cần thiết, ta gọi những thông tin này là metadata. Đến khi ta chạy một ứng dụng, CLR sẽ tiếp quản (take-over) và lại biên dịch (compile) nguồn mã một lần nữa ra thành ngôn ngữ gốc (native language) của máy vi tính trước khi thi hành những công việc đã được bố trí trong nguồn mã đó.
- Thực hiện quản lý bộ nhớ, quản lý thực thi tiểu trình, thực thi mã nguồn, xác nhận mã nguồn an toàn, biên dịch và các dịch vụ hệ thống khác.
- Ngoài ra nó còn đảm bảo cho việc thực hiện cho việc bảo mật.

b) .NET Framework Classes

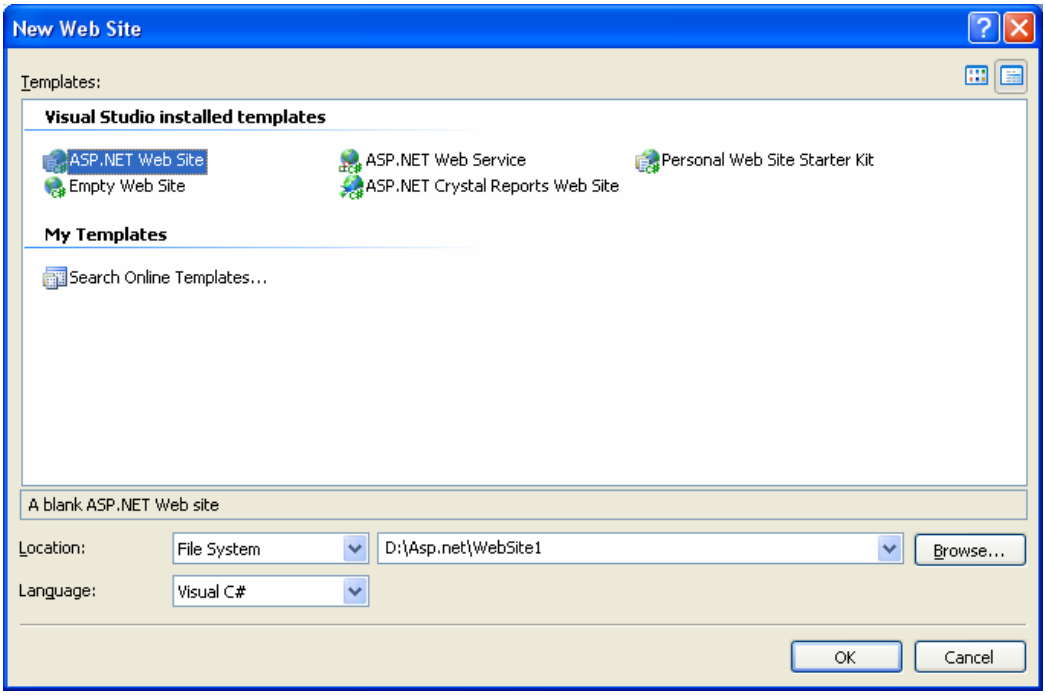
- Điều quan trọng nhất mà ta cần phải nhớ là mọi thứ trong .NET đều là đối tượng. Các đối tượng đó được tổ chức lại thành từng nhóm riêng biệt như trong một thư viện để ta dễ dàng sử dụng. Ta gọi các nhóm như vậy là không gian tên (namespaces), và ta sẽ dùng những không gian tên này để gọi hay nhập các lớp (classes) cần thiết cho ứng dụng của mình.
- Một namespace không chỉ là một nhóm các kiểu dữ liệu, mà nó làm cho tên của tất cả các kiểu dữ liệu trong cùng một không gian tên sẽ có tiếp đầu ngữ là tên của namespace đó. Nó cũng cho phép một không gian tên nằm trong một không gian tên khác. Ví dụ, hầu hết các hỗ trợ chung của các thư viện lớp cơ sở .NET đều nằm trong một không gian tên gọi là System. Lớp cơ sở Array nằm trong không gian tên này có tên đầy đủ là System.Array.

4.4. Khởi động Microsoft Visual Studio 2005

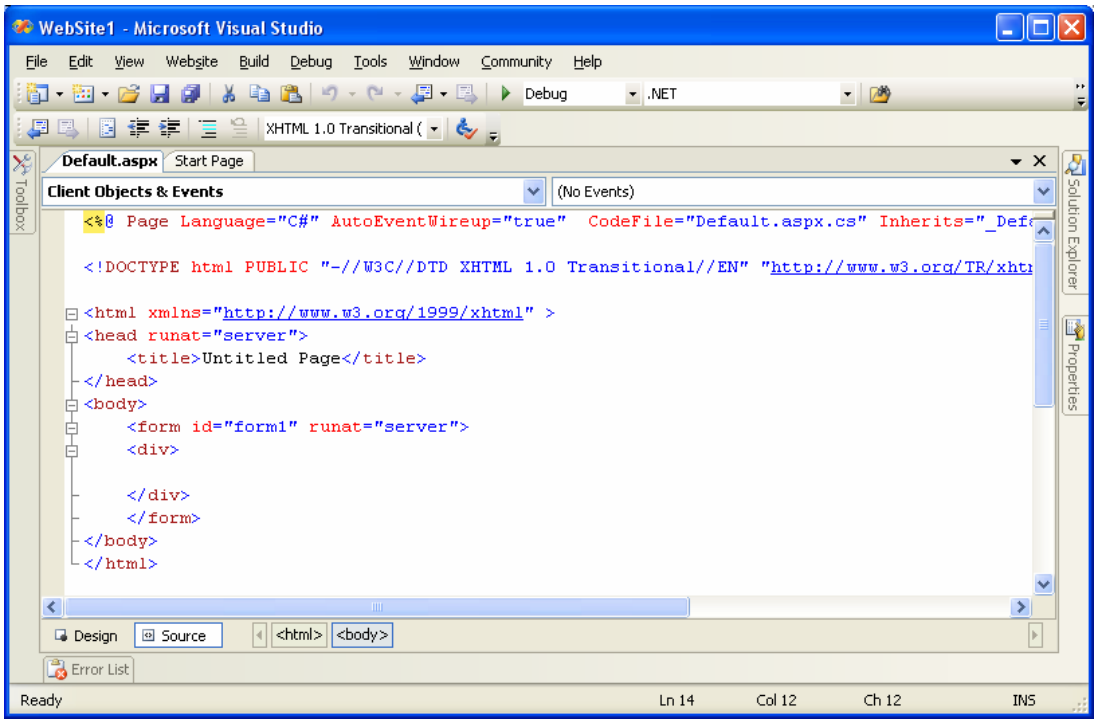
- Start/Program
- Microsoft Visual Studio 2005/Microsoft Visual Studio 2005



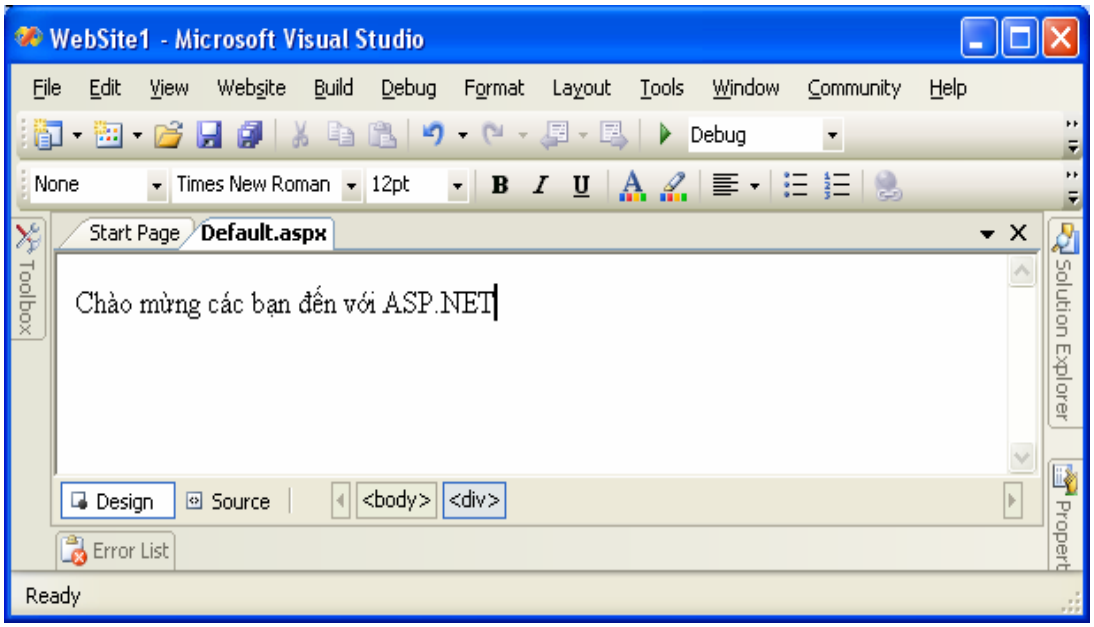
4.5. Tạo một Web site mới



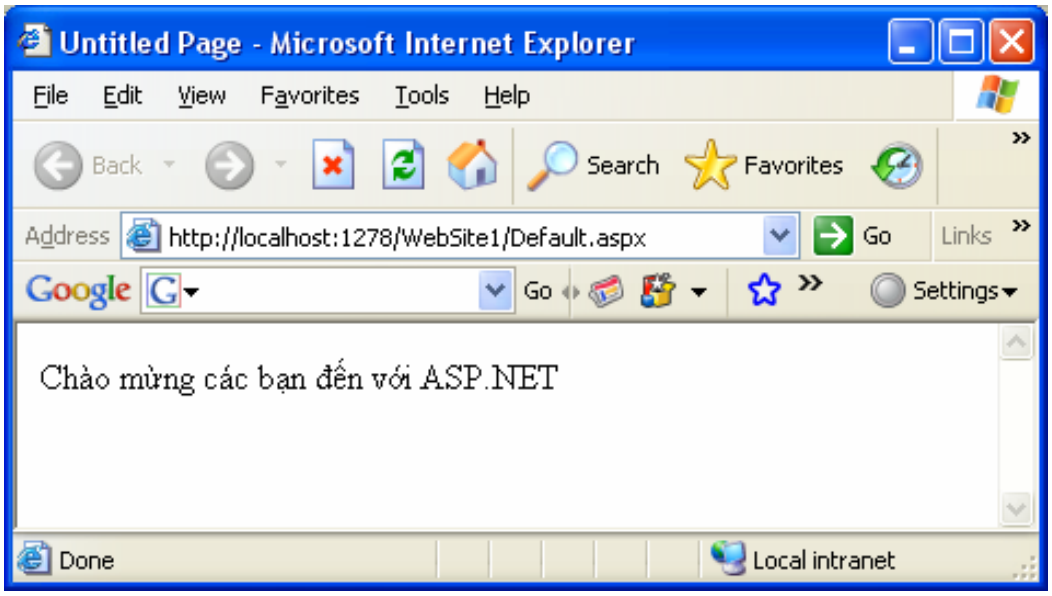
- Chọn ASP.NET Web site
- Language: Chọn ngôn ngữ lập trình (C#, VB.NET)



- Design: Chế độ thiết kế
- Source: Chế độ xem các thẻ ASP.NET
- Ví dụ: Xuất hiện dòng văn bản trên trang web.



- Thực hiện (F5), ta được nội dung trang web như sau:



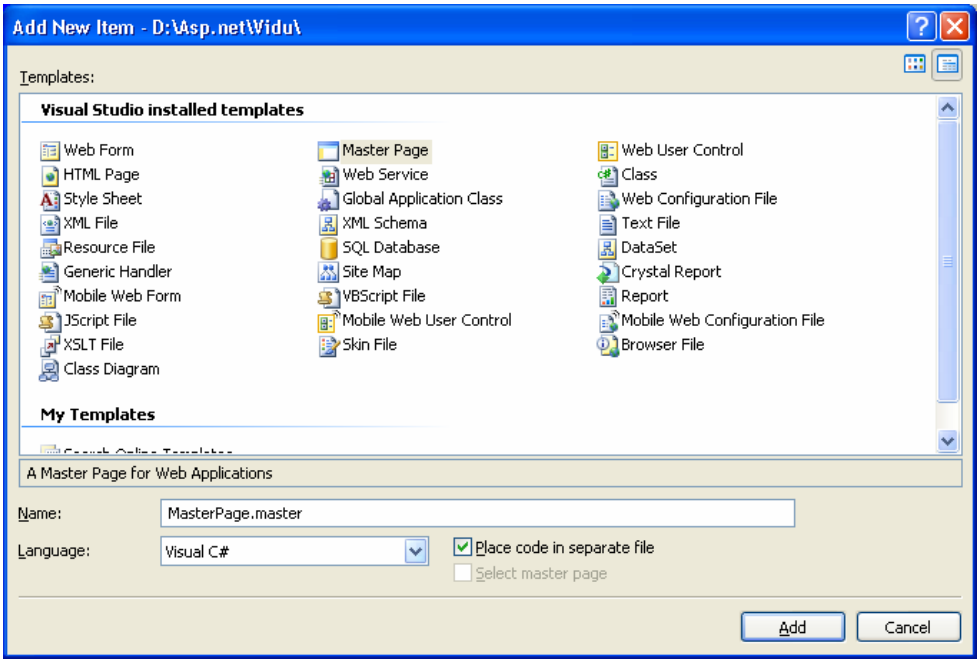
4.6. Tạo Master Page

4.6.1. Giới thiệu

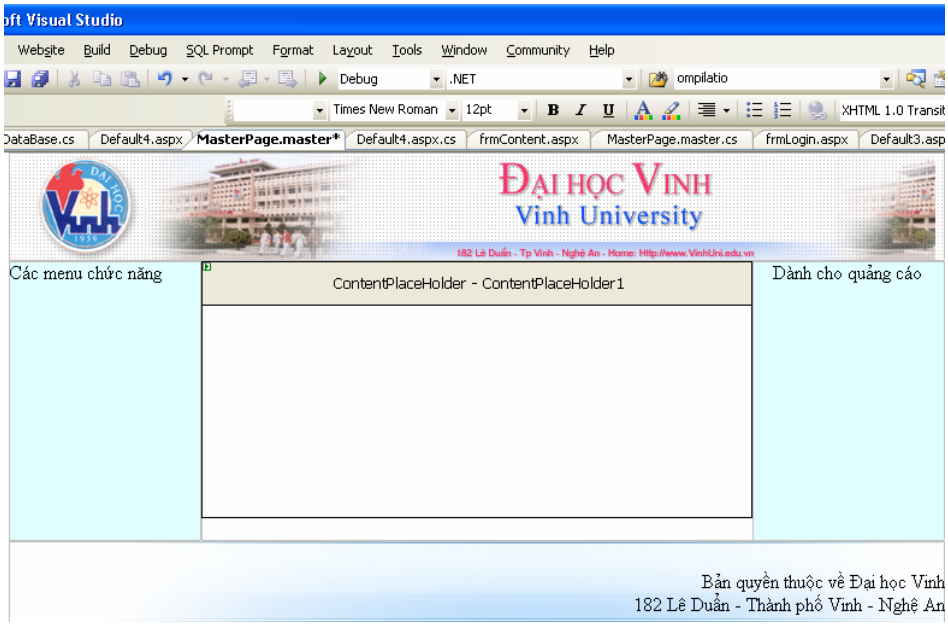
- Master pages yêu cầu 2 phần riêng biệt, phần master page và phần nội dung (content). Phần master page xác định giao diện và các điều hướng, ví dụ các thông tin chung (các thông tin được xuất hiện trên tất cả các trang của site). Trang nội dung là trang chứa các thông tin riêng biệt mà bạn muốn thể hiện. Khi một trang được hiển thị trên trình duyệt, master page sẽ trộn hai trang đó lại với nhau, kết hợp giữa trang giao diện từ master page và nội dung từ trang thông tin riêng biệt đó. Bằng cách sử dụng master page, bạn có thể tránh khỏi việc tạo lại các thông tin chung trên mỗi trang. Hơn nữa, nếu bất kỳ khi nào bạn quyết định thay đổi nội dung giao diện của toàn bộ các trang, bạn chỉ cần thay đổi trong trang master page.
- Một số đặc điểm nổi bật của master page
    - Giảm thiểu thời gian thiết kế và các tài nguyên bằng cách chỉ thiết kế những thông tin chung trong tệp master page.
    - Người thiết kế có thể tạo ra các trang có mẫu giống nhau bằng cách tham chiếu đến master page mặc định. Mỗi khi trang master thay đổi, giao diện của trang tham chiếu đến cũng thay đổi theo.
    - Sử dụng master page có thể cải thiện việc quản lý website, bởi vì bạn có thể thay đổi giao diện của website bằng cách thay đổi trong master page. Bạn không cần phải thay đổi trên toàn bộ các trang trong site của bạn.

4.6.2. Các bước thực hiện

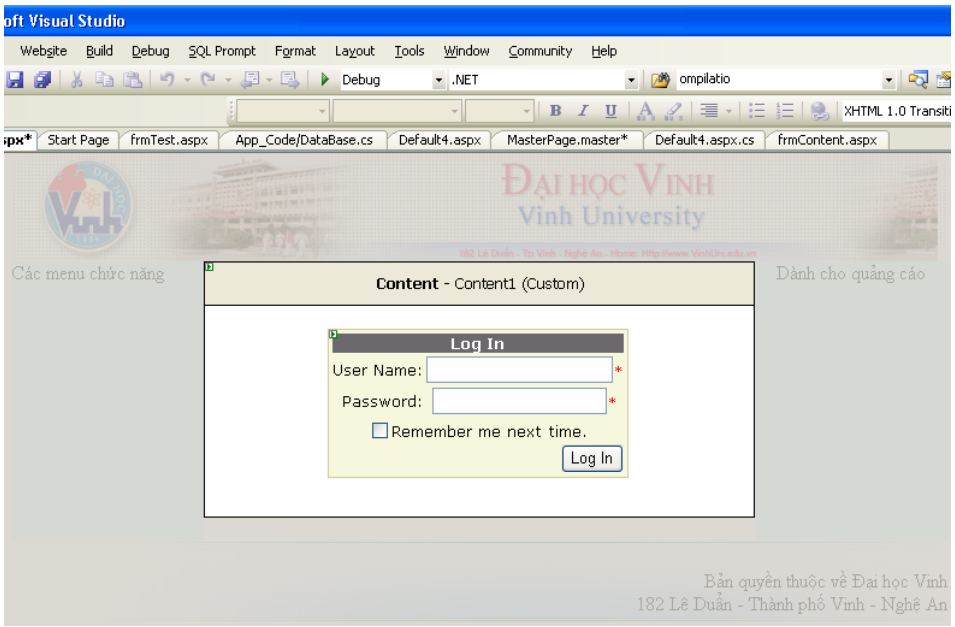
- Vào Website, chọn mục Add New Item
- Xuất hiện hộp thoại (xem hình)
  - Chọn Master Page, đặt tên ở mục Name
  - Chọn ngôn ngữ thể hiện ở mục Language
  - Chọn Add



- Xây dựng trang MasterPage.master với cấu trúc như sau



- Các phần ngoài vùng ContentPlaceHolder sẽ được giữ nguyên khi tạo một trang mới
- Để tạo một trang mới có cấu trúc tương tự trang MasterPage.master, chúng ta thực hiện các bước sau:
    - Mở trang MasterPage.master
    - Trong menu Website, chọn Add Content Page
    - Một trang mới được tạo ra có cấu trúc giống như trang MasterPage.master, chúng ta chỉ được phép chỉnh sửa trong phần Content của trang mới được tạo (xem hình).



<u>Câu hỏi ôn tập</u>:

1. Nêu sự khác biệt giữa ASP và ASP.NET
2. Các thành phần chính trong .NET framework
3. Cách thực hiện một ứng dụng ASP.NET
4. Ưu điểm khi tạo Master Page

CHƯƠNG 5. GIỚI THIỆU NGÔN NGỮ LẬP TRÌNH C#

Nội dung:

- Giới thiệu
- Biến và Hằng
- Kiểu dữ liệu tiền định nghĩa
- Câu lệnh điều kiện
- Vòng lặp
- Mảng
- Sử dụng các ghi chú
- Từ định danh và từ khóa

5.1. Giới thiệu

C# mô tả một ngôn ngữ hiện đại hướng đối tượng (object-oriented). Nó được thiết kế để chú ý đến việc diễn đạt C++ theo kiểu lập trình và phát triển nhanh ứng dụng RAD (Rapid Application Development) chẳng hạn như Microsoft Visual Basic, Delphi, C++ Builder. C# được kiến trúc bởi Anders Hejlsberg, người đã viết nên trình biên dịch Pascal và có nhiều đóng góp cho ngôn ngữ Delphi cũng như Java. Và do đó sự tiến triển của C# chịu ảnh hưởng bởi các ngôn ngữ như C++, SmallTalk, Java và các ngôn ngữ khác.

Trước hết, mọi thứ trong C# đều là đối tượng. C# không quan tâm đến dữ liệu toàn cục hay hàm toàn cục. Tất cả dữ liệu và phương thức trong C# được chứa trong khai báo: cấu trúc (struct) hoặc lớp (class). Tất cả dữ liệu và phương thức thao tác trên dữ liệu cần được đóng gói như một đơn vị chức năng, các đơn vị chức năng này là những đối tượng có thể được sử dụng lại, chúng độc lập và có thể tự hoạt động.

5.2. Biến và Hằng

5.2.1. Biến

Một biến dùng để lưu trữ giá trị mang một kiểu dữ liệu nào đó.

Cú pháp C# sau đây để khai báo một biến :

[*modifier*] *datatype identifer ;*

Với modifier là một trong những từ khoá: public, private, protected, . . . còn datatype là kiểu dữ liệu (int , long , float. . .) và identifier là tên biến.

Thí dụ dưới đây một biến mang tên i, kiểu số nguyên int và có thể được truy cập bởi bất cứ hàm nào.

```
public  int   i;
```

Ta có thể gán cho biến một giá trị bằng toán tử "=".

```
i  =  10 ;
```

Ta cũng có thể khai báo biến và khởi tạo cho biến một giá trị như sau :

```
int   i  =  10;
```

Nếu ta khai báo nhiều biến có cùng kiểu dữ liệu sẽ có dạng như sau:

```
int  x = 10, y = 20;
```

5.2.2. Hằng

Một hằng (constant) là một biến nhưng giá trị không thể thay đổi được suốt thời gian thi hành chương trình. Đôi lúc ta cũng cần có những giá trị bao giờ cũng bất biến.

Thí dụ

```
const int a = 100;   // giá trị này không thể bị thay đổi
```

Hằng có những đặc điểm sau :

- Hằng bắt buộc phải được gán giá trị lúc khai báo. Một khi đã được khởi gán thì không thể viết đè chồng lên.
- Giá trị của hằng được tính toán vào lúc biên dịch, do đó không thể gán một hằng từ một trị của một biến. Nếu muốn làm thế thì phải sử dụng đến một read-only field.
- Hằng bao giờ cũng static, tuy nhiên ta không thể đưa từ khoá static vào khi khai báo hằng.

Có ba thuận lợi khi sử dụng hằng trong chương trình của bạn :

- Hằng làm cho chương trình đọc dễ dàng hơn, bằng cách thay thế những con số vô cảm bởi những tên mang đầy ý nghĩa hơn.
- Hằng làm cho dễ sữa chương trình hơn.
- Hằng làm cho việc tránh lỗi dễ dàng hơn, nếu bạn gán một trị khác cho một hằng đâu đó trong chương trình sau khi bạn đã gán giá trị cho hằng, thì trình biên dịch sẽ thông báo lỗi.

5.3. Kiểu dữ liệu tiền định nghĩa

C# là một ngôn ngữ được kiểm soát chặt chẽ về mặt kiểu dữ liệu, ngoài ra C# còn chia các kiểu dữ liệu thành hai loại khác nhau: ***kiểu trị*** (value type) và ***kiểu qui chiếu*** (reference type). Nghĩa là trên một chương trình C# dữ liệu được lưu trữ một hoặc hai nơi tuỳ theo đặc thù của kiểu dữ liệu.

Thứ nhất là **stack** một vùng nhớ dành lưu trữ dữ liệu có chiều dài cố định, chẳng hạn int chiếm dụng 4 bytes. Mỗi chương trình khi đang thi hành đều được cấp phát riêng một stack riêng biệt mà các chương trình khác không được tác động tới. Khi một hàm được gọi hàm thi hành thì tất cả các biến cục bộ của hàm được đưa vào stack và khi hàm hoàn thành công tác thì những biến cục bộ của hàm đều bị lấy ra. Đây là cách thu hồi khi hàm hết hoạt động.

Thứ hai là **heap**, một vùng nhớ dùng lưu trữ dữ liệu có kích cỡ thay đổi, string chẳng hạn, hoặc dữ liệu có một cuộc sống dài hơn phương thức của một đối tượng chẳng hạn, thí dụ khi phương thức thể hiện (instantiate) một đối tượng, đối tượng được lưu trữ trên heap, và nó không bị đẩy ra khi hàm hoàn thành giống như

stack, mà ở nguyên tại chỗ và có thể trao cho các phương thức khác thông qua một qui chiếu.

C# cũng hỗ trợ kiểu con trỏ (pointer type) giống như C++ nhưng ít khi dùng đến và chỉ dùng khi làm việc với đoạn mã unmanaged. Đoạn mã unmanaged là đoạn mã được tạo ra ngoài môi trường .NET, chẳng hạn những đối tượng COM.

5.3.1. Kiểu giá trị được định nghĩa trước (Predefined Value Types)

Kiểu dữ liệu bẩm sinh (The built-in value types) trình bày ban đầu như integer và floating-point numbers, character, và boolean types.

Các kiểu Integer:

C# hỗ trợ 8 kiểu dữ liệu số nguyên sau:

Name	CTS Type	Description	Range (min:max)
sbyte	System.SByte	8-bit signed integer	-128:127 (-2^7:2^7-1)
short	System.Int16	16-bit signed integer	-32,768:32,767 (-2^{15}:2^{15}-1)
int	System.Int32	32-bit signed integer	-2,147,483,648:2,147,483,647 (-2^{31}:2^{31}-1)
long	System.Int64	64-bit signed integer	-9,223,372,036,854,775,808: 9,223,372,036,854,775,807 (-2^{63}:2^{63}-1)
byte	System.Byte	8-bit signed integer	0:255 (0:2^8-1)
ushort	System.UInt16	16-bit signed integer	0:65,535 (0:2^{16}-1)
uint	System.UInt32	32-bit signed integer	0:4,294,967,295 (0:2^{32}-1)
ulong	System.UInt64	64-bit signed integer	0:18,446,744,073,709,551,615(0:2^{64}-1)

Thí dụ :

```
long x = 0x12ab;// ghi theo hexa
uint ui = 1234U;
long l = 1234L;
ulong ul = 1234UL;
```

Kiểu dữ liệu số dấu chấm động (Floating Point Types)

Name	CTS Type	Description	Significant Figures	Range (approximate)
Float	System.Single	32-bit single-precision floating- point	7	$\pm 1.5 \times 10^{-45}$ to $\pm 3.4 \times 10^{38}$
Double	System.Double	64-bit double-precision floating- point	15/16	$\pm 5.0 \times 10^{-324}$ to $\pm 1.7 \times 10^{308}$

Thí dụ:

```
float   f  =  12.3F;
```

Kiểu dữ liệu số thập phân (Decimal Type):

Name	CTS Type	Description	Significant Figures	Range (approximate)
decimal	System.Decimal	128-bit high precision decimal notation	28	$\pm 1.0 \times 10^{-28}$ to $\pm 7.9 \times 10^{28}$

Thí dụ :

decimal d = 12.30M ; //có thể viết decimal d = 12.30m;

Kiểu Boolean :

Name	CTS Type	Value
Bool	System.Boolean	true or false

Kiểu Character Type:

Name	CTS Type	Value
char	System.Char	Biểu diễn ký tự Unicode 16-bit

5.3.2. Kiểu tham khảo tiền định nghĩa

C# hỗ trợ hai kiểu dữ liệu được định nghĩa trước:

Name	CTS Type	Description
object	System.Object	The root type, from which all other types in the CTS derive (including value types)
string	System.String	Các ký tự Unicode

Các ký tự escape thông dụng:

Escape Sequence	Character
\'	Single quote
\"	Double quote
\\	Backslash
\0	Null
\a	Alert
\b	Backspace
\f	Form feed
\n	Newline
\r	Carriage return
\t	Tab character
\v	Vertical tab

Kiểu chuỗi:

Đối tượng kiểu string chứa một chuỗi ký tự. Khi khai báo một biến chuỗi, chúng ta sử dụng từ khoá string như sau:

string myString;

Thường thì phải gán giá trị cho một biến kiểu string:

string myString = "Xin chao";

5.4. Câu lệnh điều kiện

5.4.1. Câu lệnh điều kiện if

Cú pháp như sau:

```
if (dieu_kien)
   Cau_lenh_1;
[else
   Cau_lenh_2;]
```

Nếu có nhiều hơn một câu lệnh để thi hành trong câu điều kiện chúng ta sẽ đưa tất cả các câu lệnh này vào trong dấu ngoặc móc ({ ...}).

Ý nghĩa:

- Nếu *dieu_kien* đúng thì thực hiện *Cau_lenh_1*;
- Ngược lại thì thực hiện các lệnh *Cau_lenh_2*;

5.4.2. Câu lệnh switch

Các câu lệnh if nằm lồng rất khó đọc, khó gỡ rối. Khi bạn có một loạt lựa chọn phức tạp thì nên sử dụng câu lệnh switch.

Cú pháp như sau:

switch (biểu thức)

{ case biểu thức ràng buộc:

câu lệnh

câu lệnh nhảy

[default: câu lệnh mặc định]

}

5.5. Vòng lặp (Loops)

C# cung cấp cho chúng ta 4 vòng lặp khác nhau (for, while, do...while, và foreach) cho phép chúng ta thực hiện một đoạn mã lặp lại khi đúng điều kiện lặp.

5.5.1. Vòng lặp for

Cú pháp:

for (*initializer; condition; iterator*)

statement(s)

Thí dụ:

Đoạn mã sau sẽ xuất ra tất cả số nguyên từ 0 đến 99:

```
for (int i = 0; i < 100; i++)
{
        Console.WriteLine(i);
}
```

5.5.2. Vòng lặp while

Cú pháp:

```
        while (condition)
                statement(s);
```

Thí dụ :

```
bool condition = true;
while (condition)
{
         //Vòng lặp thực hiện đến khi condition sai
        //Thực hiện các công việc
        condition = CheckCondition();
        //CheckCondition() trả về kiểu bool
}
```

5.5.3. Vòng lặp do . . . while (The do…while Loop)

```
bool condition;
do
{
        // Vòng lặp này sẽ thực hiện ít nhất một lần
        // thậm chí nếu câu điều kiện sai
} while (condition);
```

5.5.4. Vòng lặp foreach (The foreach Loop)

Cho phép bạn duyệt qua tất cả các phần tử trong dãy hoặc các tập hợp khác, và tuần tự xem xét từng phần tử một.Cú pháp như sau:

foreach (type identifier in expression) statement;

Thí dụ:

```
foreach (int temp in arrayOfInts)
{
        Console.WriteLine(temp);
}
foreach (int temp in arrayOfInts)
{
        temp++;
        Console.WriteLine(temp);
}
```

5.5.5. Câu lệnh goto

```
goto Label1;
        Console.WriteLine("This won't be executed");
Label1:
        Console.WriteLine("Continuing execution from here");
```

5.5.6. Câu lệnh break

Ta dùng câu lệnh break khi muốn ngưng việc thi hành và thoát khỏi vòng lặp.

5.5.7. Câu lệnh continue

Câu lệnh continue được dùng trong vòng lặp khi bạn muốn khởi động lại một vòng lặp nhưng lại không muốn thi hành phần lệnh còn lại trong vòng lặp, ở một điểm nào đó trong thân vòng lặp.

5.5.8. Câu lệnh return

Câu lệnh return dùng thoát khỏi một hàm hành sự của một lớp, trả quyền điều khiển về phía hàm gọi (caller). Nếu hàm có một kiểu dữ liệu trả về thì return phải trả về một kiểu dữ liệu này; bằng không thì câu lệnh được dùng không có biểu thức.

5.6. Mảng (Arrays)

Mảng là một cấu trúc dữ liệu cấu tạo bởi một số biến được gọi là những phần tử mảng. Tất cả các phần tử này đều thuộc một kiểu dữ liệu. Bạn có thể truy xuất phần tử thông qua chỉ số (index). Chỉ số bắt đầu bằng zero (0).

Có nhiều loại mảng: mảng một chiều, mảng nhiều chiều.

Cú pháp :

```
type[ ] array-name;
```

thí dụ:

```
int[] myIntegers; // mảng kiểu số nguyên
string[] myString ; // mảng kiểu chuỗi chữ
```

Bạn khai báo mảng có chiều dài xác định với từ khoá **new** như sau:

```
int[] myIntegers = new int[32];
integers[0] = 35;// phần tử đầu tiên có giá trị 35
integers[31] = 432;// phần tử 32 có giá trị 432
Bạn cũng có thể khai báo như sau:
int[] integers;
integers = new int[32];
string[] myArray = {"first element", "second element"};
```

5.6.1. Làm việc với mảng

Ta có thể tìm được chiều dài của mảng sau nhờ vào thuộc tính Length, xét ví dụ sau:

int arrayLength = integers.Length

Nếu các thành phần của mảng là kiểu định nghĩa trước (predefined types), ta có thể sắp xếp tăng dần bằng phương thức Array.Sort()

Array.Sort(myArray);

Chúng ta có thể đảo ngược mảng đã có nhờ vào phương thức Reverse():

Array.Reverse(myArray);

```
string[] artists = {"Leonardo", "Monet", "Van Gogh", "Klee"};
Array.Sort(artists);
Array.Reverse(artists);
foreach (string name in artists)
{
        Console.WriteLine(name);
}
```

5.6.2. Mảng nhiều chiều (Multidimensional Arrays)

Cú pháp :

type[,] array-name;

Thí dụ muốn khai báo một mảng hai chiều gồm hai hàng ba cột với phần tử kiểu nguyên :

int[,] myRectArray = new int[2,3];

Bạn có thể khởi gán mảng xem các ví dụ sau về mảng nhiều chiều:

```
//mảng 4 hàng 2 cột
int[,] myRectArray = new int[,]{{1,2},{3,4},{5,6},{7,8}};
string[,] beatleName =      { {"Lennon","John"},
                              {"McCartney","Paul"},
                              {"Harrison","George"},
                              {"Starkey","Richard"} };
```

chúng ta có thể sử dụng :

```
string[,,] my3DArray;
double [, ] matrix = new double[10, 10];
for (int i = 0; i < 10; i++)
{
        for (int j=0; j < 10; j++)
                matrix[i, j] = 4;
}
```

5.7. Sử dụng các ghi chú

Ở phần này chúng ta xem phần thêm các ghi chú vào đoạn mã. C# sử dụng kiểu truyền thống của C hàng đơn (// ...) và nhiều hàng (/* ... */).

Một chương trình C# cũng có thể chứa những dòng chú giải. Ví dụ:

// Ghi chú trên một dòng đơn

và

```
/*
        Ghi chú trên nhiều dòng
 */
```

5.8. Từ định danh và từ khoá

Từ định danh là tên chúng ta đặt cho biến, để định nghĩa kiểu sử dụng như các lớp, cấu trúc, và các thành phần của kiểu này. C# có một số quy tắc để định rõ các từ định danh như sau:

- Bắt đầu bằng ký tự
- Không được sử dụng từ khoá làm từ định danh

Trong C# có sẵn một số từ khoá (keyword).

abstract	do	implicit	params	switch
as	double	in	private	this
base	else	int	protected	throw
bool	enum	interface	public	true
break	event	internal	readonly	try
byte	explicit	is	ref	typeof
case	extern	lock	return	uint
catch	false	long	sbyte	ulong
char	finally	namespace	sealed	unchecked
checked	fixed	new	short	unsafe
class	float	null	sizeof	ushort
const	for	object	stackalloc	using
continue	foreach	operator	static	virtual
decimal	goto	out	string	volatile
default	if	override	struct	void
delegate				while

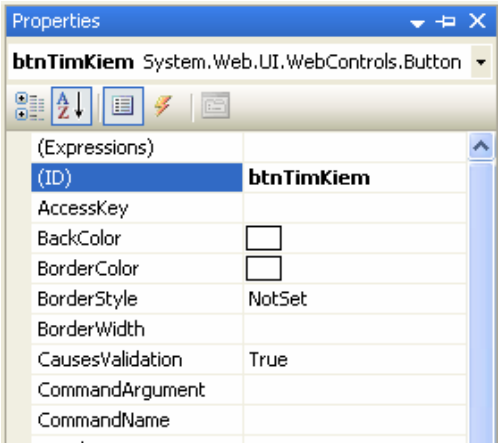
CHƯƠNG 6. CÁC ĐIỀU KHIỂN VÀ CÁC ĐỐI TƯỢNG TRONG ASP.NET

Các ứng dụng Web Forms bao gồm nhiều điều khiển (controls) khác nhau. Các control này có thể đơn giản như các control Button và TextBox, hay chúng có thể tinh vi và phức tạp hơn như các control TreeView và GridView. Trong .NET framework có nhiều control sẵn sàng kết hợp với các ứng dụng Web Forms, và rất nhiều control được dùng trong các phát triển ứng dụng .NET tuỳ biến.

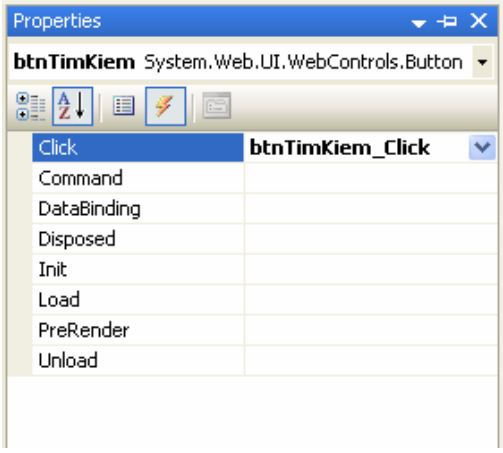
Visual Studio .NET có thể thêm các control này vào một Web form cho bạn. Mọi control thừa kế từ System.Web.UI.WebControls. Lớp này chứa các phương thức và các thuộc tính cơ bản được dùng bởi bất kỳ control nào cung cấp một giao diện cho người sử dụng.

- Các thao tác với các controls
- Hiện thanh Toolbox
- Kéo và thả controls vào web form

- Cửa sổ Properties: Nhấp chuột phải vào control, chọn Properties



Thuộc tính



Sự kiện

6.1. Một số điều khiển cơ bản

6.1.1. Label

- Sử dụng Label để hiển thị một đoạn văn bản trên trang web
- Một số thuộc tính:
  - BackColor: Tạo màu nền
  - BorderColor: Màu đường viền

- BorderStyle: Kiểu đường viền
- Enabled: Cho phép tác động/không tác động đến control
- Font: Tạo font, kích cỡ,... cho Label
- Text: Đoạn văn bản sẽ hiển thị
- ToolTip: Đoạn văn bản sẽ hiển thị khi rê chuột qua Label
- Visible: Ẩn/hiện Label

6.1.2. Button

- Sử dụng Button để tạo một nút và người dùng nhấp chuột vào nút để thực hiện một lệnh nào đó.
- Sự kiện:
  - Click: sự kiện được thực hiện khi người dùng nhấp chuột vào button

6.1.3. Checkbox

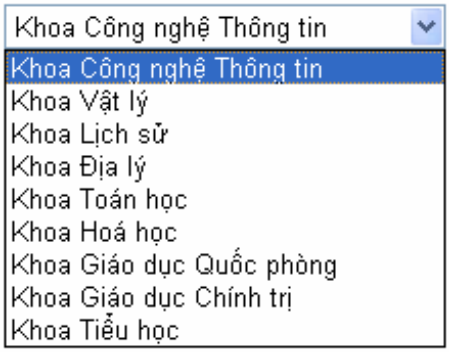
- Thường sử dụng trong các trường hợp nhập giá trị kiểu boolean (true, false), (yes, no).
- Một số sự kiện:
  - CheckedChange
  - Load

6.1.4. Radio button

- Thường sử dụng để người sử dụng chọn một trong các giá trị.
- Một số sự kiện:
  - CheckedChange
  - Load

6.1.5. DropDownList

- Tạo một danh sách trải xuống khi người dùng nhấp chuột vào, cho phép người dùng chọn một giá trị trong danh sách.
- Một số thuộc tính:
  - Id: định danh đối tượng
  - DataSource
  - DataTextField
  - DataValueField
  - Items



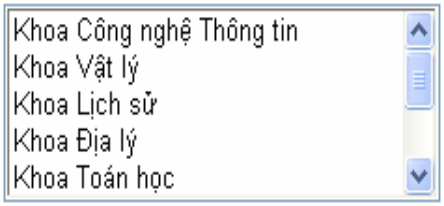
- Một số sự kiện:
  - SelectedIndexChange
  - TextChanged
  - Load
  - DataBind

Ví dụ: Thêm danh sách các tháng từ 1→ 12 vào DropDownList1

```
for (int i = 1;  i < 13; i++)
        DropDownList1.Items.Add("Tháng " + i.ToString());
```

6.1.6. Listbox

- Thường sử dụng để người sử dụng chọn một hoặc nhiều giá trị trong danh sách.



- Một số thuộc tính:
  - Id: định danh đối tượng
  - DataSource
  - DataTextField
  - DataValueField
  - Items
- Một số sự kiện:
  - SelectedIndexChange
  - TextChanged
  - Load
  - DataBind

Ví dụ: Thêm danh sách các tháng từ 1→ 12 vào ListBox1

```
for (int i = 1;  i < 13; i++)
        ListBox1.Items.Add("Tháng " + i.ToString());
```

6.1.7. TextBox

- Thường sử dụng để người dùng nhập thông tin, hiển thị thông tin trên trang web.
- Một số thuộc tính:
  - Id: định danh đối tượng
  - TextMode (SingleLine, MultiLine, Password)
- Một số sự kiện:
  - TextChange
  - Load

6.1.8. HyperLink

- Dùng để tạo siêu liên kết trên trang web
- Một số thuộc tính:
  - Id: định danh đối tượng

- ImageUrl
- NevigateUrl
- Target

- Một số sự kiện:
  - DataBinding
  - Load

6.1.9. Calendar

- Dùng để tạo lịch trên trang web
- Một số thuộc tính:
  - BackColor
  - BorderColor
  - DayHeaderStyle
  - DayStyle
  - Font
  - SelectedStyle
  - TitleStyle
  - TodayDayStyle
  - WeekendDayStyle

6.1.10. Image

- Dùng để đưa hình ảnh lên trang web.
- Một số thuộc tính:
  - Height
  - ImageAlign
  - ImageUrl
  - ToolTip
  - Visible
  - Width

6.1.11. GridView

- Hiển thị các thông tin theo dạng bảng, mỗi cột biểu diễn một trường (field) và mỗi hàng mô tả một bản ghi (record). GridView cho phép chúng ta lựa chọn, sắp xếp hay sửa mỗi bản ghi.
- Mỗi cột trong GridView mô tả bởi đối tượng DataControlField. Bảng dưới đây mô tả các kiểu cột có thể dùng:

Column field type	Description
BoundField	Hiển thị giá trị của một trường trong Data Source.

ButtonField	Hiển thị một nút (button) cho mỗi bản ghi trong GridView.
CheckBoxField	Hiển thị một CheckBox cho mỗi bản ghi trong GridView. Kiểu này thường áp dụng cho các trường nhận giá trị Boolean (true, false).
CommandField	Hiển thị các nút để thực hiện lệnh chọn, sửa, xóa
HyperLinkField	Hiển thị giá trị của một trường trong Data Source như là một siêu liên kết.
ImageField	Hiển thị ảnh cho mỗi bản ghi trong GridView
TemplateField	Hiển thị các nội dung do người lập trình định nghĩa cho mỗi bản ghi trong GridView.

- Một số thuộc tính
  - DataSource: dữ liệu nguồn, có thể lấy từ DataSet, DataTable
  - BackColor: màu nền
  - Caption: tiêu đề
  - GridLine: các kiểu đường lưới
- Một số sự kiện:

PageIndexChanged	Thực hiện khi một trang trong GridView được chọn, nhưng sau khi GridView chuyển trang.
PageIndexChanging	Thực hiện khi một trang trong GridView được chọn, trước khi GridView chuyển trang.
RowCreated	Thực hiện khi một hàng mới được tạo ra trong GridView. Sự kiện này thường dùng để thay đổi nội dung một hàng khi hàng được tạo ra.
RowDeleted	Thực hiện khi nút Delete trong hàng được nhấn, nhưng sau khi GridView xóa bản ghi trong Data Source. Sự kiện này thường để kiểm tra kết quả của thao tác xóa.
RowDeleting	Thực hiện khi nút Delete trong hàng được nhấn, nhưng trước khi GridView xóa bản ghi trong Data Source. Sự kiện này thường để hủy bỏ thao tác xóa (nếu cần).
RowEditing	Thực hiện khi nút Edit được trong hàng được chọn, nhưng trước khi GridView chuyển sang trang thái hiệu chỉnh. Sự kiện này thường để hủy bỏ thao tác hiệu chỉnh.

RowUpdated	Thực hiện khi nhấn vào nút Update, nhưng sau khi GridView cập nhật các hàng. Sự kiện này thường để kiểm tra kết quả của thao tác cập nhật.
RowUpdating	Thực hiện khi nhấn vào nút Update, nhưng trước khi GridView cập nhật các hàng. Sự kiện này thường dùng để hủy bỏ thao tác cập nhật.

6.2. Điều khiển kiểm tra dữ liệu nhập vào

6.2.1. RequiredFieldValidator

- Kiểm tra giá trị nhập vào của control (TextBox), nếu control chưa được nhập giá trị, đoạn chương trình sẽ ngừng và thông báo lỗi ErrorMessage.
- Một số thuộc tính
  - ControlToValidate: chọn control để kiểm tra giá trị nhập vào
  - ErrorMessage: thông báo lỗi nếu control chưa nhập thông tin
  - ForeColor: màu của thông báo lỗi

6.2.2. RangeValidator

- Kiểm tra giá giá trị nhập vào tại control, nếu nhập các giá trị ngoài giới hạn, đoạn chương trình sẽ ngừng và thông báo lỗi ErrorMessage.
- Một số thuộc tính
  - ControlToValidate: chọn control để kiểm tra các giá trị
  - ErrorMessage: thông báo lỗi
  - ForeColor: màu của thông báo lỗi
  - Type: kiểu dữ liệu sẽ kiểm tra ở control (string, integer, double, date, currency)
  - MinimumValue: giá trị bé nhất (để kiểm tra)
  - MaximumValue: giá trị lớn nhất (để kiểm tra)

```
//Kiểm tra giá trị nhập vào của TextBox1
//Giới hạn nhập cho phép 1 → 10
RangeValidator1.ControlToValidate = "TextBox1";
RangeValidator1.Type = ValidationDataType.Integer;
RangeValidator1.MaximumValue = "10";
RangeValidator1.MinimumValue = "1";
```

6.2.3. CompareValidator

- So sánh giá trị nhập vào ở 2 control, nếu không bằng nhau, chương trình ngừng và đưa ra thông báo ErrorMesssage.
- Một số thuộc tính
  - ControlToCompare: chọn control để so sánh

- ControlToValidate: chọn control để kiểm tra các giá trị với control so sánh.
- ErrorMessage: thông báo lỗi
- ForeColor: màu của thông báo lỗi

```
//Kiểm tra giá trị nhập vào của TextBox1 và TextBox2
CompareValidator1.ErrorMessage = "Values are different";
CompareValidator1.ControlToCompare = "TextBox1";
CompareValidator1.ControlToValidate = "TextBox2";
```

6.2.4. RegularExpressionValidator

- Kiểm tra giá trị nhập vào đúng theo một biểu thức
- Một số thuộc tính
  - ControlToValidate: chọn control để kiểm tra các giá trị với control so sánh
  - ErrorMessage: thông báo lỗi
  - ValidationExpression: biểu thức kiểm tra

```
//Kiểm tra giá trị nhập vào TextBox1,
//giá trị nhập vào phải 5 chữ số
RegularExpressionValidator1.ControlToValidate = "TextBox1";
RegularExpressionValidator1.ValidationExpression="\d{5}";
RegularExpressionValidator1.ErrorMessage="Must be 5 numeric digits";
```

6.3. Các đối tượng trong ASP.NET

Có rất nhiều đối tượng trong ASP.NET (ASP.NET Objects), trong chương này chúng ta chỉ xét 1 số đối tượng như sau:

- Response Object
- Request Object
- Page Object
- Session Object
- Applictation Object

6.3.1. Response Object

Response object cho phép Server đáp ứng, trả lời hay thông tin với Client.

6.3.1.1. Phương thức Write

Phương thức Write của Response object để hiện thị dòng chữ ở trình duyệt web của client.

Ví dụ:

```
Response.Write("Chào mừng các bạn đến với ASP.NET!");
Response.Write("<br>");
Response.Write("<hr>");
```

6.3.1.2. Phương thức Redirect

Phương thức Redirect dùng để chuyển sang một trang Web khác một cách gián tiếp.

Ví dụ:

Response.Redirect("http://vinhuni.edu.vn");

Response.Redirect("index.aspx");

6.3.2. Request Object

Request object dùng để thông tin giữa Server và Client browser. Browser dùng Request object để gửi thông tin cần thiết tới Server. Giống như Response, Request object là thể hiện của HttpRequest. Như vậy, Request object đại diện cho Client khi yêu cầu trang Web, còn Server sẽ dùng Response và Request để đáp ứng yêu cầu hay đòi hỏi thông tin từ Client.

Một ứng dụng quan trọng của Request object là thu thập thông tin của Client browser. Thường, thông tin của Client browser được gửi đi dưới dạng form hay querystring (querystring: thông tin gởi kèm vào phần đuôi của request URL).

Ví dụ dùng querystring như sau:

http://vinhuni.edu.vn?id=10&p=2

Dấu ? chỉ thị cho biết có thông tin đính kèm và & dùng phân biệt các cặp giá trị với nhau.

```
string id = Request.QueryString["id"].ToString();
string p = Request.QueryString["p"].ToString();
```

6.3.3. Page Object

Page object gồm tất cả thuộc tính (properties), phương thức (method) dùng cho các trang ASP.NET và xuất xứ từ Page class ở .NET framework.

Page object gồm một số thành phần như sau:

- Load
- IsPostBack
- Databind

Sự kiện Load dùng để khởi động khi trang Web bắt đầu hiển thị ở browser. IsPostBack cho ta biết form ở trang Web đã được gởi đi tới cùng trang Web hay không? Databind nối kết mọi dữ liệu (data) từ cơ sở dữ liệu (database) với công cụ (controls) ở trang Web.

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!Page.IsPostBack)
    {
        DateTime myDate = DateTime.Now;
        Label1.Text = "The date, time is: " + myDate.ToString();
    }
}
```

6.3.4. Session Object

Khi Client đã nhận được thông tin (information) từ Server, quá trình trao đổi qua lại đó kết thúc ngay tức khắc. Sau đó, Server và Client không còn kết nối

với nhau, vì thế muốn lưu lại một số thông tin để tạo kết nối lại khi cần thiết chúng ta dùng đối tượng Session. Session cho phép ta lưu giữ thông tin dưới các dạng như biến số (variables), objects, strings hay ... bất cứ loại thông tin nào ở Server, khi nào cần thiết thì lấy các thông tin đó kết nối lại. Thông tin tồn tại trong Session kết thúc khi người dùng thoát khỏi trình duyệt hoặc sang một trang khác.
Ví dụ:

```
//Tạo một số Session
Session["Name"]="VinhUni";
Session["Course"]="ASP.NET";
Session["AMessage"]="Chào mừng các bạn";
Session["ASPNET"]="Hello";
```

Ta dùng đối tượng Session khi ta cần lưu trữ một vài thông tin như username hay password. Trong trường hợp quá nhiều thông tin cần phải lưu trữ, chúng ta nên dùng cookies hay cơ sở dữ liệu (database) thích hợp và hiệu quả hơn.

Kiểm soát Session Object

Có nhiều cách kiểm soát đối tượng Session trong trang ASP.NET, cơ bản gồm có:

- Timeout
- Abandon

+ *Timeout* dùng bố trí khoảng thời gian để 1 Session có thể tồn tại trước khi ASP.NET hủy bỏ session đó, ví dụ như có người lướt mạng thăm trang Web rồi không có tác động gì nữa, session và các thông tin liên hệ hủy bỏ trong vòng 20 phút. 20 phút này là giá trị timeout mặc định (default) nếu dùng IIS Version 5.0, nếu muốn, ta có thể thay đổi giá trị timeout thành 60 phút chẳng hạn:

Session.Timeout = 60

Tuy vậy, ta nên cẩn thận khi thay đổi giá trị timeout này vì: Mỗi user lướt mạng đều được gắn hay đính kèm 1 đối tượng Session duy nhất lưu trữ trong bộ nhớ của Server. Vì vậy, nếu đặt timeout quá dài sẽ tốn bộ nhớ và không bảo mật. Mặt khác, nhất là đối với các mạng thương nghiệp (e-commerce site), nếu bố trí timeout quá ngắn, các sản phẩm được chọn mua và đặt trong shopping cart sẽ xoá sạch trước khi khách hàng tiến hành thủ tục trả tiền và như vậy gây trở ngại rất lớn.

Một cách tổng quát, timeout 20 phút là lý tưởng. Tuy nhiên, ta có thể thay đổi giá trị tùy theo tính chất của mạng, đối với ngân hàng (secure banking web site), timeout có thể rất ngắn nhưng với các mạng thương nghiệp, timeout sẽ lâu hơn.

+ *Abandon* dùng để kết thúc 1 session ngay tức khắc. Thí dụ, sau khi user kiểm tra email (Web Email) xong và logout để người khác không thể lợi dụng đọc mail, ta có thể kết thúc session đó bằng cách:

Session.Abandon();

Mệnh lệnh này sẽ xoá sạch các cookie tạm thời (temporary cookie) cũng như các thông tin liên hệ.

6.3.5. Application Object

Cũng tương tự như đối tượng Response, ASP.NET tạo ra HttpApplication Object gọi là Application chỉ khi nào ứng dụng của chúng ta khởi động - nghĩa là khi có user yêu cầu tham khảo trang Web lần đầu tiên.

Chỉ có duy nhất một Application object được tạo ra cho toàn bộ ứng dụng mà thôi, không như Session object được tạo ra riêng biệt cho từng user một.

Tuy vậy, HttpApplication Object giống Session Object ở chỗ: HttpApplication Object cũng được dùng để lưu trữ các biến số và các đối tượng. Các biến số và các đối tượng này có hiệu lực (available) cho toàn bộ ứng dụng (application) chứ không cá biệt cho một user như đối với Session.

Vì đối tượng Application chia sẻ thông tin giữa nhiều người sử dụng nên nó có các phương thức Lock và Unlock đi kèm.

Session

```
//Tạo một Session
Session["username"] = txtUserName.Text.Trim();
```

Application

```
//Dùng phương thức Lock và Unlock để tăng số người
//đang truy cập vào trang web, trong tệp Global.asax
Application.Lock();
num_Online ++;
Application["Number_Online"] = num_Online;
Application.UnLock();
```

CHƯƠNG 7. TRUY CẬP CƠ SỞ DỮ LIỆU .NET

Trong chương này, chúng ta sẽ bàn về cách làm sao để một chương trình C# sử dụng ADO.NET. Kết thúc chương này, chúng ta sẽ có được các kiến thức sau:

- Các kết nối cơ sở dữ liệu - sử dụng các lớp SqlConnection và OleDbConnection để kết nối và huỷ kết nối với cơ sở dữ liệu.
- Các lệnh thực thi - ADO.NET chứa một đối tượng command, thực thi SQL, hoặc có thể phát ra một stored procedure để trả về các giá trị. Các tùy chọn khác của đối tượng command sẽ được bàn kĩ, với các ví dụ cho từng tuỳ chọn được đưa ra trong các lớp Sql và OleDB.
- Tìm hiểu DataSet, DataTable, DataRow, và DataColumn,..

Nội dung:

- Tổng quan về ADO.NET
- Sử dụng các Database Connection
- Sử dụng hiệu quả các Connection
- Các giao dịch (Transaction)
- Commands
- Executing Commands
- Data Tables
- Tạo một DataSet

7.1. Tổng quan về ADO.NET

Giống như hầu hết các thành phần của .NET Framework, ADO.NET không chỉ là vỏ bọc của một vài API sẵn có. Nó chỉ giống ADO ở cái tên - các lớp và phương thức truy xuất dữ liệu đều khác hoàn toàn.

ADO (Microsoft's ActiveX Data Objects) là một thư viện của các thành phần COM đã từng được ca ngợi trong một vài năm trở lại đây. Các thành phần chủ yếu của ADO là Connection, Command, Recordset, và các Field object. Một connection có thể mở cơ sở dữ liệu, một vài dữ liệu được chọn vào một recordset, bao gồm các trường, dữ liệu này sau đó có thể thao tác, cập nhập lên server, và connection cần phải được đóng lại. ADO cũng giới thiệu một disconnected recordset, cái được dùng khi không muốn giữ kếp nối trong một thời gian dài.

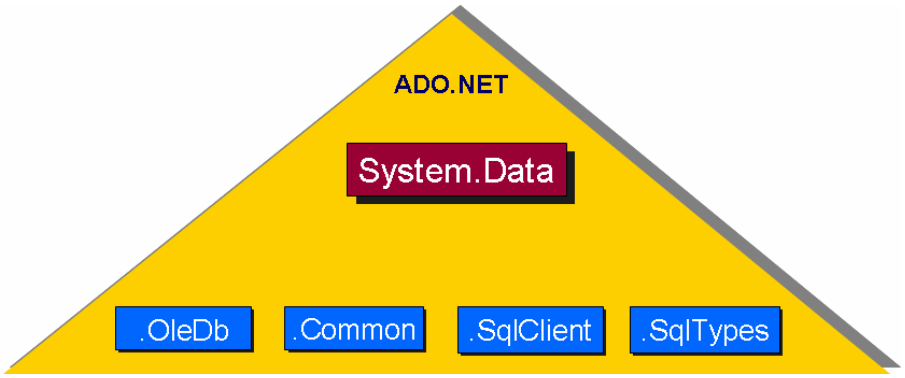
Có một vài vấn đề với ADO đó là sự không hài lòng về địa chỉ, sự cồng kềnh của một disconnected recordset. Hỗ trợ này không cần thiết với sự tiến hoá của tin học "web-centric", vì vậy nó cần được loại bỏ. Có một số giống nhau giữa lập trình ADO.NET và ADO (không phải ở cái tên), vì thế việc chuyển từ ADO sang ADO.NET không quá khó khăn.

ADO.NET chứa hai không gian tên cơ sơ dữ liệu - một cho SQL Server, và một cái khác cho các cơ sở dữ liệu được trình bày thông qua một giao diện OLEDB. Nếu cơ sở dữ liệu của bạn chọn là một bộ phận của OLEDB, bạn có thể dễ dàng kết nối với nó từ .NET - chỉ cần dùng các lớp OLE DB và kết nối thông qua các driver cơ sở dữ liệu hiện hành của bạn.

7.1.1. Các không gian tên (Namespace)

Các không gian tên sau chỉ ra các lớp và các giao diện được dùng cho việc truy xuất dữ liệu trong .NET:

- System.Data - Các lớp truy xuất dữ liệu chung
- System.Data.Common - Các lớp dùng chung bởi các data provider khác nhau
- System.Data.OleDb - Các lớp của OLE DB provider
- System.Data.SqlClient - Các lớp của SQL Server provider
- System.Data.SqlTypes - Các kiểu của SQL Server



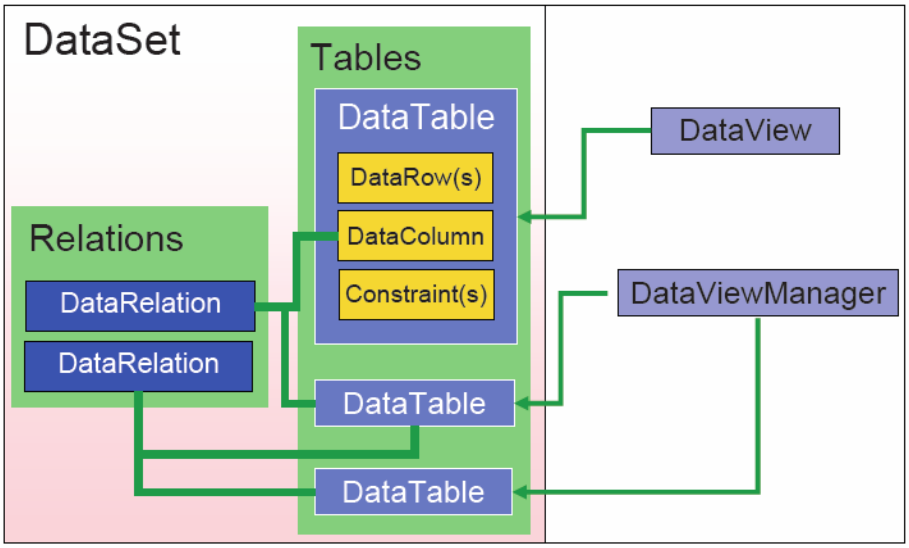
7.1.2. Các lớp dùng chung

ADO.NET chứa một số lớp được dùng không quan tâm là bạn đang dùng các lớp của SQL Server hay là các lớp của OLE DB.

Các lớp trong không gian tên System.Data:

- *DataSet* - Đối tượng này chứa một bộ các DataTable, có thể bao gồm quan hệ giữa các bảng, và nó được thiết kế cho truy xuất dữ liệu không kết nối.
- *DataTable* - Một kho chứa dữ liệu. Một DataTable bao gồm một hoặc nhiều DataColumns, và khi được tạo ra nó sẽ có một hoặc nhiều DataRows chứa dữ liệu.
- *DataRow* - Một bộ giá trị, tương đương một dòng trong bảng cơ sở dữ liệu, hoặc một dòng của bảng tính.
- *DataColumn* - Chứa các định nghĩa của một cột, chẳng hạn như tên và kiểu dữ liệu.
- *DataRelation* - Một liên kết giữa hai DataTable trong một DataSet. Sử dụng cho khóa ngoại và các mối quan hệ chủ tớ.

- *Constraint* - Định nghĩa một qui tắc cho một DataColumn (hoặc một bộ các cột dữ liệu), như các giá trị là độc nhất.



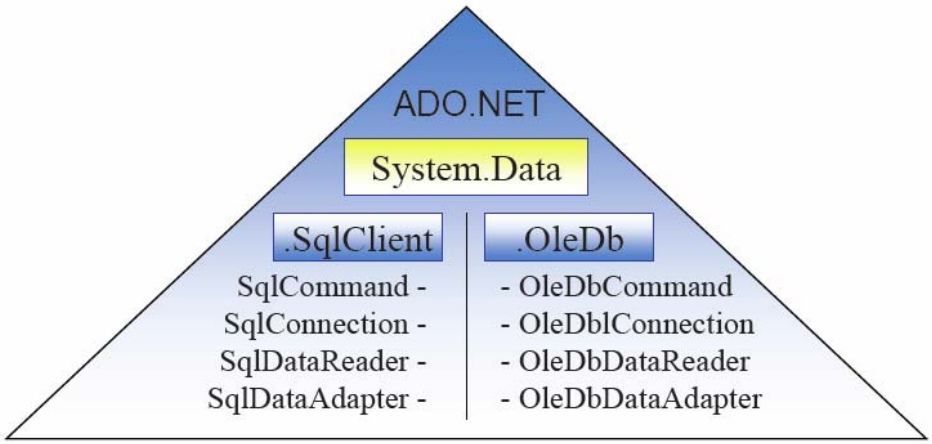
7.1.3. Các lớp cơ sở dữ liệu chuyên biệt

Bổ sung cho các lớp dùng chung ở trên, ADO.NET có một số các lớp dữ liệu chuyên biệt được đưa ra dưới đây. Các lớp này thực thi một bộ các giao diện chuẩn được định nghĩa trong không gian tên System.Data, cho phép sử dụng các lớp có cùng kiểu giao diện.

- *SqlCommand, OleDbCommand* - Một vỏ bọc của các câu lệnh SQL hoặc các lời gọi stored procedure.
- *SqlCommandBuilder, OleDbCommandBuilder* - Một lớp sử dụng các câu lệnh SQL (chẳng hạn như các câu lệnh INSERT, UPDATE, vàDELETE) từ một câu lệnh SELECT.
- *SqlConnection, OleDbConnection* - Kết nối với cơ sở dữ liệu. Giống như một ADO Connection.
- *SqlDataAdapter, OleDbDataAdapter* - Một lớp giữ các câu lệnh Select, Insert, Update, và Delete, chúng được sử dụng để tạo một DataSet và cập nhật Database.
- *SqlDataReader, OleDbDataReader* - Chỉ đọc, kết nối với data reader.
- *SqlParameter, OleDbParameter* - Định nghĩa một tham số cho một stored procedure.
- *SqlTransaction, OleDbTransaction* - Một giao tiếp cơ sở dữ liệu, được bọc trong một đối tượng.

Một đặc tính quan trọng của các lớp ADO.NET là chúng được thiết kế để làm việc trong môi trường không kết nối (connectionless), đóng một vai trò quan trọng trong thế giới "web-centric". Nó hiện được dùng để kiến trúc một server

(chẳng hạn như mua sách qua mạng) để kết nối một server, lấy một vài dữ liệu, và làm việc trên những dữ liệu này trên PC khách trước khi kết nối lại và truyền dữ liệu trở lại để xử lí.



.NET Data Providers Hierarchy

7.2. Sử dụng các Database Connection

Trong trình tự truy xuất cơ sở dữ liệu, chúng ta cần cung cấp các thông số kết nối, chẳng hạn như thiết bị mà cơ sở dữ liệu đang chạy, và khả năng đăng nhập. Bất kì ai đã từng làm việc với ADO sẽ dễ dàng quen với các lớp kết nối của .NET, OleDbConnection và SqlConnection. Đoạn mã sau đây mô tả cách để tạo, mở và đóng một kết nối đến cơ sở dữ liệu Northwind. Các ví dụ trong chương này được dùng cơ sở dữ liệu Northwind:

```
using System.Data.SqlClient;
string source = "Server=(local);" +
                "UID=QSUser;PWD=QSPassword;" +
                "DataBase=Northwind";
SqlConnection conn = new SqlConnection(source);
conn.Open();  //Mở kết nối
              // Thực hiện một số công việc ở đây
conn.Close();  //Đóng kết nối
```

Trong ví dụ chuỗi kết nối này, các tham số được dùng như sau (các tham số cách nhau bởi dấu chấm phẩy trong chuỗi kết nối).

- *Server=(local)* - Nó biểu diễn DataBase Server được kết nối. SQL Server cho phép một số các tiến trình Database Server Processes khác nhau chạy trên cùng một máy.
- *UID=QSUser* - Tham số này mô tả người dùng cơ sở dữ liệu. Bạn cũng có thể sử dụng User ID.

- *PWD=QSPassword* - và đây là Password cho người dùng đó. .NET SDK là một bộ các cơ sở dữ liệu giống nhau, và User/Password này được liên kết và được thêm vào trong quá trình cài đặt các ví dụ .NET. Bạn cũng có thể dùng Password.
- *Database=Northwind* - Cái này mô tả loại dữ liệu để kết nối - mỗi tiến trình SQL Server có thể đưa ra một vài loại dữ liệu khác nhau.

Ví dụ trên mở một kết nối cơ sở dữ liệu cùng chuỗi kết nối đã được định nghĩa, sau đó đóng kết nối lại. Khi kết nối đã được mở, bạn có thể thực hiện các lệnh để thao tác trên cơ sở dữ liệu, và khi hoàn tất, kết nối có thể được đóng lại.

SQL Server có một chế độ bảo mật khác - nó có thể dùng chế độ bảo mật của Windows, vì thế các khả năng truy cập của Windows có thể truyền cho SQL Server. Với lựa chọn này bạn có thể bỏ đi các vị trí UID và PWD trong chuỗi kết nối, và thêm vào Integrated Security=SSPI.

7.3. Sử dụng hiệu quả các Connection

Một cách tổng quát, khi sử dụng các tài nguyên "hiếm" trong .NET, chẳng hạn như các kết nối cơ sở dữ liệu, các cửa sổ, hoặc các đối tượng đồ họa, nên đảm bảo rằng các tài nguyên này luôn phải được đóng lại sau khi đã sử dụng xong. Dù vậy các nhà thiết kết của .NET có thể làm điều này nhờ trình thu gom rác, nó luôn làm sau một khoảng thời gian nào đó, tuy nhiên nó nên được giải phóng càng sớm càng tốt.

Rõ ràng là khi viết mã truy xuất một cơ sở dữ liệu, việc giữ một kết nối càng ít thời gian càng tốt để không làm ảnh hưởng đến các phần khác. Trong nhiều tình huống tiêu cực, nếu không đóng một kết nối có thể khoá không cho các người dùng khác truy nhập vào các bảng dữ liệu đó, một tác hại to lớn đối với khả năng thực thi của ứng dụng. Việc đóng một kết nối cơ sở dữ liệu có thể coi là bắt buộc, vì thế ứng dụng này chỉ ra cách cấu trúc mã của bạn để giảm thiểu các rủi ro cho một mã nguồn mở.

Có hai cách để đảm bảo rằng các kết nối cơ sở dữ liệu được giải phóng sau khi dùng.

7.3.1. Tùy chọn try/catch/finally

Tùy chọn thứ nhất để đảm bảo rằng các tài nguyên được dọn sạch là sử dụng các khối lệnh try…catch…finally, và đảm bảo rằng bạn đã đóng các kết nối trong khối lệnh finally. Đây là một ví dụ nhỏ:

```
try
{
        // Open the connection
        conn.Open();
        // Do something useful
}
catch ( Exception ex )
{
```

```
        // Do something about the exception
}
finally
{
        // Ensure that the connection is freed
        conn.Close ( ) ;
}
```

Với khối kết nối bạn có thể giải phóng bất kì tài nguyên nào mà bạn đã dùng. Vấn đề duy nhất trong phương thức này là bạn phải bảo đảm rằng bạn có đóng các kết nối - rất là dễ quên việc thêm vào khối finally, vì vậy một phong cách lập trình tốt rất quan trọng.

Ngoài ra, bạn có thể mở một số tài nguyên (chẳng hạn hai kết nối cơ sở dữ liệu và một file) trong một phương thức, vì vậy đôi khi các khối try…catch…finally trở nên khó đọc. Có một cách khác để đảm bảo rằng các tài nguyên được dọn dẹp - sử dụng câu lệnh.

7.3.2. Sử dụng khối câu lệnh

Trong lúc phát triển C#, phương thức .NET's dọn dẹp các đối tượng khi chúng không còn được tham chiếu nữa sử dụng các hàm hủy bỏ trở thành một vấn đề nóng hổi. Trong C++, ngay khi một đối tượng rời khỏi tầm vực, hàm hủy bỏ của nó sẽ tự động được gọi. Nó là một điều rất mới cho các nhà thiết có các lớp sử dụng tài nguyên, khi một hàm huỷ bỏ được sử dụng để đóng các tài nguyên nếu các người dùng quên làm điều đó. Một hàm hủy bỏ trong C++ được gọi bất kì khi nào một đối tượng vượt quá tầm vực của nó - vì vậy khi một ngoại lệ được phát ra mà không được chặn, tất cả các hàm hủy bỏ cần phải được gọi.

Với C#, tất cả đều tự động, các hàm hủy định trước được thay thế bởi trình thu gom rác, cái được dùng để hủy bỏ các tài nguyên tại một thời điểm trong tương lai. Chúng mang tính bất định, nghĩa là bạn sẽ không biết trước được khi nào thì việc đó sẽ xảy ra. Nếu quên không đóng một kết nối cơ sở dữ liệu có thể là nguyên nhân gây ra lỗi khi chạy trong .NET. Mã sau đây sẽ giải thích cách để sử dụng giao diện IDisposable để giải phóng tài nguyên khi thoát khỏi khối using.

```
string source =         "Server=(local);" +
                        "UID=QSUser;PWD=QSPassword;" +
                        "Database=Northwind";
using ( SqlConnection conn = new SqlConnection ( source ) )
{
        // Open the connection
        conn.Open ( ) ;
        // Do something useful
}
```

Đối tượng trong mệnh đề using phải thực thi giao diện IDisposable, nếu không sẽ tạo ra một lỗi biên dịch. Phương thức Dispose() sẽ tự động được gọi trong khi thoát khỏi khối using.

Khi xem mã phương thức Dispose() của SqlConnection (và OleDbConnection), cả hai đều kiểm tra trạng thái của đối tượng kết nối, và nếu nó đang mở phương thức Close() sẽ được gọi.

Khi lập trình bạn nên dùng cả hai tùy chọn trên. Ở những chỗ bạn cần các tài nguyên tốt nhất là sử dụng mệnh đề using(), dù vậy bạn cũng có thể sử dụng câu lệnh Close(), nếu quên không sử dụng thì khối lệnh using sẽ đóng lại giúp bạn. Không gì có thể thay thế được một bẫy ngoại lệ tốt, vì thế tốt nhất bạn dùng trộn lẫn hai phương thức như ví dụ sau:

```
try
{
  using (SqlConnection conn = new SqlConnection ( source ))
  {
        // Open the connection
        conn.Open ( ) ;
        // Do something useful
        // Close it myself
        conn.Close ( ) ;
  }
}
catch (Exception e)
{
        // Do something with the exception here...
}
```

Ở đoạn trên đã gọi tường minh phương thức Close() mặc dù điều đó là không bắt buộc vì khối lệnh using đã làm điều đó thay cho bạn; tuy nhiên, bạn luôn chắc rằng bất kì tài nguyên nào cũng được giải phóng sớm nhất có thể - bạn có thể có nhiều mã trong khối lệnh mã không khoá tài nguyên.

Thêm vào đó, nếu một ngoại lệ xảy ra bên trong khối using, thì phương thức IDisposable.Dispose sẽ được gọi để bảo đảm rằng tài nguyên được giải phóng, điều này đảm bảo rằng kết nối cơ sở dữ liệu luôn luôn được đóng lại. Điều này làm cho mã dễ đọc và luôn đảm bảo rằng kết nối luôn được đóng khi một ngoại lệ xảy ra.

Cuối cùng, nếu bạn viết các lớp bao bọc một tài nguyên có lẽ luôn thực hiện giao diện IDisposable để đóng tài nguyên. Bằng cách dùng câu lệnh using() nó luôn đảm bảo rằng tài nguyên đó sẽ được dọn dẹp.

7.4. Các Transaction (giao dịch)

Thường khi có nhiều hơn một cập nhật dữ cơ sở dữ liệu thì các thực thi này được thực hiện bên trong tầm vực của một transaction. Một transaction trong ADO.NET được khởi tạo bằng một lời gọi đến các phương thức

BeginTransaction() trên đối tượng kết nối cơ sở dữ liệu. Những phương thức này trả về một đối tượng có thể thực thi giao diện IDbTransaction, được định nghĩa trong System.Data.

Chuỗi mã lệnh dưới đây khởi tạo một transaction trên một kết nối SQL Server:

```
string source = "server=(local);" +
                "uid=QSUser;pwd=QSPassword;" +
                "database=Northwind";
SqlConnection conn = new SqlConnection(source);
conn.Open();
SqlTransaction tx = conn.BeginTransaction();
// Execute some commands, then commit the transaction
tx.Commit();
conn.Close();
```

Khi bạn khởi tạo một transaction, bạn có thể chọn bậc tự do cho các lệnh thực thi trong transaction đó. Bậc này chỉ rõ sự tự do của transaction này với các transaction khác xảy ra trên cơ sở dữ liệu.

7.5. Commands

Chúng ta lại nói lại về commands. Một command là một một kiểu đơn giản, một chuỗi lệnh SQL được dùng để truy xuất dữ liệu. Một command có thể là một stored procedure, hoặc là tên của một bảng sẽ trả về:

```
string source = "server=(local);" +
                "uid=QSUser;pwd=QSPassword;" +
                "database=Northwind";
string select = "SELECT  *  FROM Customers";
SqlConnection conn = new SqlConnection(source);
conn.Open();
SqlCommand cmd = new SqlCommand(select, conn);
```

Các mệnh đề SqlCommand và OleDbCommand thường được gọi là CommandType, chúng được dùng để định nghĩa các mệnh đề SQL, một stored procedure, hoặc một câu lệnh SQL. Sau đây là một bảng liệt kê đơn giản về CommandType:

CommandType	Example
Text (default)	String select = "SELECT ContactName FROM Customers"; SqlCommand cmd = new SqlCommand(select , conn);

StoredProcedure	SqlCommand cmd = new SqlCommand("CustOrderHist", conn);
	cmd.CommandType = CommandType.StoredProcedure;
	cmd.Parameters.Add("@CustomerID", "QUICK");

TableDirect	OleDbCommand cmd = new OleDbCommand("Categories", conn); cmd.CommandType = CommandType.TableDirect;

Khi thực thi một stored procedure, cần truyền các tham số cho procedure. Ví dụ trên cài đặt trực tiếp tham số @CustomerID, dù vậy có nhiều cách để cài giá trị tham số.

7.6. Executing Commands

Bạn đã định nghĩa các command, và bạn muốn thực thi chúng. Có một số cách để phát ra các statement, dựa vào kết quả mà bạn muốn command đó muốn trả về. Các mệnh đề SqlCommand và OleDbCommand cung cấp các phương thức thực thi sau:

- *ExecuteNonQuery()* – Thực thi các command không trả về kết quả gì cả
- *ExecuteReader()* – Thực thi các command và trả về kiểu IDataReader
- *ExecuteScalar()* – Thực thi các command và trả về một giá trị đơn

7.6.1. ExecuteNonQuery()

Phương thức này thường được dùng cho các câu lệnh UPDATE, INSERT, hoặc DELETE, để trả về số các mẫu tin bị tác động. Phương thức này có thể trả về các kết quả thông qua các tham số được truyền vào stored procedure.

```
using System;
using System.Data.SqlClient;
public class ExecuteNonQueryExample
{
  public static void Main(string[] args)
  {
    string source = "server=(local);" +
                    "uid=QSUser;pwd=QSPassword;" +
                    "database=Northwind";
    string select = "UPDATE Customers " +
                   "SET ContactName = 'Bob' " +
                   "WHERE ContactName = 'Bill'";
    SqlConnection  conn = new SqlConnection(source);
    conn.Open();
    SqlCommand cmd = new SqlCommand(select, conn);
    int rowsReturned = cmd.ExecuteNonQuery();
    Console.WriteLine("{0} rows returned.", rowsReturned);
    conn.Close();
  }
}
```

ExecuteNonQuery() trả về một số kiểu int cho biết số dòng bị tác động command.

7.6.2. ExecuteReader()

Phương thức này thực hiện các lệnh trả về một đối tượng SqlDataReader hoặc OleDbDataReader. Đối tượng này có thể dùng để tạo ra các mẫu tin như mã sau đây:

```
using System;
using System.Data.SqlClient;
public class ExecuteReaderExample
{
  public static void Main(string[] args)
  {
    string source = "server=(local);" +
                    "uid=QSUser;pwd=QSPassword;" +
                    "database=Northwind";
    string select = "SELECT * FROM Customers";
    SqlConnection conn = new SqlConnection(source);
    conn.Open();
    SqlCommand cmd = new SqlCommand(select, conn);
    SqlDataReader reader = cmd.ExecuteReader();
    while(reader.Read())
    {
        Console.WriteLine("Contact : {0,-20} Company : {1}",
                          reader[0] , reader[1]);
    }
  }
}
```

7.6.3. ExecuteScalar()

Trong nhiều trường hợp một câu lệnh SQL cần phải trả về một kết quả đơn, chẳng hạn như số các record của một bảng, hoặc ngày giờ hiện tại của server. Phương thức ExecuteScalar có thể dùng cho những trường hợp này:

```
using System;
using System.Data.SqlClient;
public class ExecuteScalarExample
{
  public static void Main(string[] args)
  {
    string source = "server=(local);" +
                    "uid=QSUser;pwd=QSPassword;" +
                    "database=Northwind";
    string select = "SELECT COUNT(*) FROM Customers";
    SqlConnection conn = new SqlConnection(source);
    conn.Open();
    SqlCommand cmd = new SqlCommand(select, conn);
    object o = cmd.ExecuteScalar();
  }
}
```

Phương thức trả về một đối tượng, Bạn có thể chuyển sang kiểu thích hợp.

7.7. Data Tables

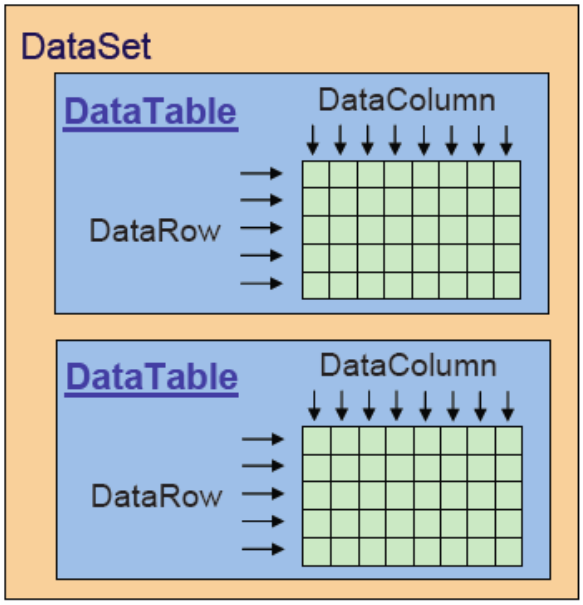
Một data table rất giống một bảng cơ sở dữ liệu vật lí – nó bao gồm một bộ các cột với các thuộc tính riêng, và có thể không chứa hoặc chứa nhiều dòng dữ liệu. Một data table có thể định nghĩa một khóa chính, bao gồm một hoặc nhiều

cột, và cũng có thể chứa các ràng buộc của các cột. Tất cả các thông tin đó được thể hiện trong **schema**.

Một đối tượng DataTable (cũng như một DataColumn) có thể có một số các mở rộng riêng liên quan đến thuộc tính của nó. Tập hợp này có thể nằm trong thông tin user-defined gắn liền với đối tượng. Ví dụ, một cột có thể đưa ra một mặt nạ nhập liệu dùng để giới hạn các giá trị hợp lệ cho cột đó. Các thuộc tính mở rộng đặc biệt quan trọng khi dữ liệu được cấu trúc ở một tầng giữa và trả về cho client trong một số tiến trình. Bạn có thể lưu một chuẩn hợp lệ (như min và max) cho các số của các cột.

Khi một bảng dữ liệu được tạo ra, có thể do việc chọn dữ liệu từ một cơ sở dữ liệu, đọc dữ liệu từ một file, hoặc truy xuất thủ công trong mã, tập hợp Rows được dùng để chứa giá trị trả về.

Tập hợp Columns chứa các thể hiện DataColumn có thể được thêm vào bảng này. Những định nghĩa schema của dữ liệu, ví dụ như kiểu dữ liệu, tính khả rỗng, giá trị mặc định,... Tập Constraints có thể được tạo ra bởi các ràng buộc khóa chính hoặc tính độc nhất.



7.7.1. Data Columns

Một đối tượng DataColumn định nghĩa các thuộc tính của một cột trong DataTable, chẳng hạn như kiểu dữ liệu của cột đó, chẳng hạn cột là chỉ đọc, và các sự kiện khác. Một cột có thể được tạo bằng mã, hoặc có thể được tạo tự động trong thời gian chạy.

Khi tạo một cột, tốt hơn hết là nên đặt cho nó một cái tên; nếu không thời gian chạy sẽ tự động sinh cho bạn một cái tên theo định dạng Column**n**, n là một số tự động tăng.

Kiểu dữ liệu của một cột có thể cài đặt bằng cách cung cấp trong cấu trúc của nó, hoặc bằng cách cài đặt thuộc tính DataType. Một khi bạn đã nạp (load) dữ

liệu vào một bảng dữ liệu bạn không thể sửa lại kiểu dữ liệu của một cột – nếu không bạn sẽ nhận một ngoại lệ.

Các cột dữ liệu có thể được tạo để giữ các kiểu dữ liệu của .NET Framework sau:

Boolean	Decimal	Int64	TimeSpan
Byte	Double	Sbyte	UInt16
Char	Int16	Single	UInt32
DateTime	Int32	String	UInt64

Một khi đã được tạo, bước tiếp theo là gán các thuộc tính khác cho đối tượng DataColumn, chẳng hạn như tính khả rỗng nullability, giá trị mặc định. Đoạn mã sau chỉ ra một số các tùy chọn được cài đặt trong một DataColumn:

```
DataColumn customerID = new DataColumn("CustomerID" , typeof(int));
customerID.AllowDBNull = false;
customerID.ReadOnly = false;
customerID.AutoIncrement = true;
customerID.AutoIncrementSeed = 1000;
DataColumn name = new DataColumn("Name" , typeof(string));
name.AllowDBNull = false;
name.Unique = true;
```

Các thuộc tính sau có thể được cài đặt trong một DataColumn:

Property	**Description**
AllowDBNull	Nếu là true, cho phép cột có thể chấp nhận DBNull.
AutoIncrement	Cho biết rằng dữ liệu của cột này là một số tự động tăng.
AutoIncrementSeed	Giá trị khởi đầu cho một cột AutoIncrement.
AutoIncrementStep	Cho biết bước tăng giữa các giá trị tự động, mặc định là 1.
Caption	Có thể dùng cho việc biểu diễn tên của cột trên màn hình.
ColumnMapping	Cho biết cách một cột ánh xạ sang XML khi một DataSet được lưu bằng cách gọi phương thức DataSet.WriteXml.
ColumnName	Tên của cột. Nó tự động tạo ra trong thời gian chạy nếu không được cài đặt trong cấu trúc.
DataType	Kiểu giá trị của cột.
DefaultValue	Dùng để định nghĩa giá trị mặc định cho một cột
Expression	Thuộc tính này định nghĩa một biểu thức dùng cho việc tính toán trên cột này

7.7.2. Data Rows

Lớp này cấu thành các phần khác của lớp DataTable. Các cột trong một data table được định nghĩa trong các thuộc tính của lớp DataColumn. Dữ liệu của bảng thật sự có thể truy xuất được nhờ vào đối tượng DataRow. Ví dụ sau trình bày cách truy cập các dòng trong một bảng dữ liệu.

```
string source = "server=(local);" +
                "uid=QSUser;pwd=QSPassword;" +
                "database=northwind";
string select = "SELECT ContactName,CompanyName FROM Customers";
SqlConnection  conn = new SqlConnection(source);
```

Mã sau đây giới thiệu lớp SqlDataAdapter, được dùng để điền dữ liệu cho một DataSet. SqlDataAdapter sẽ phát ra các SQL, và điền vào một bảng Customers trong DataSet.

```
SqlDataAdapter da = new SqlDataAdapter(select, conn);
DataSet ds = new DataSet();
da.Fill(ds , "Customers");
```

Trong mã dưới đây, bạn chú ý cách dùng chỉ mục của DataRow để truy xuất giá trị trong dòng đó. Giá trị của một cột có thể trả về bằng cách dụng một trong những chỉ mục được cài đè. Chúng cho phép bạn trả về một giá trị cho biết số, tên, hoặc DataColumn:

```
foreach(DataRow row in ds.Tables["Customers"].Rows)
        Console.WriteLine("'{0}' from {1}" , row[0] ,row[1]);
```

Mỗi dòng có một cờ trạng thái gọi là RowState, nó có thể dùng để xác định thực thi nào là cần thiết cho dòng đó khi nó cập nhật cơ sở dữ liệu. Thuộc tính RowState có thể được cài đặt để theo dõi tất cả các trạng thái thay đổi trên DataTable, như thêm vào các dòng mới, xóa các dòng hiện tại, và thay đổi các cột bên trong bảng. Khi dữ liệu được cập nhật vào cơ sở dữ liệu, cờ trạng thái được dùng để nhận biết thực thi SQL nào sẽ xảy ra. Những cờ này được định nghĩa bởi bảng liệt kê DataRowState:

DataRowState Value	**Description**
Added	Dòng được vừa mới được thêm vào tập hợp DataTable's Rows. Tất cả các dòng được tạo trên máy khách đều được cài đặt giá trị này, và cuối cùng là phát ra câu lệnh SQL INSERT khi cập nhật cho cơ sở dữ liệu.
Deleted	Giá trị này cho biết dòng đó có thể được đánh dấu xoá trong DataTable bởi phương thức DataRow.Delete(). Dòng này vẫn tồn tại trong DataTable, nhưng không thể trông thấy từ màn hình (trừ khi một DataView được cài

	đặt rõ ràng). Các dòng được đánh dấu trong DataTable sẽ bị xoá khỏi cơ sở dữ liệu khi nó được cập nhật.
Detached	Một dòng sẽ có trạng thái này ngay sau khi nó đươc tạo ra, và có thể cũng trả về trạng thái này bởi việc gọi phương thức DataRow.Remove(). Một dòng detached không được coi là một thành phần của bảng dữ liệu.
Modified	Một dòng sẽ được Modified nếu giá trị trong cột bất kì bị thay đổi.
Unchanged	Một dòng sẽ không thay đổi kể từ lần cuối cùng gọi AcceptChanges().

Trạng thái của một dòng phụ thuộc vào phương thức mà dòng đó đã gọi. Phương thức AcceptChanges() thường được gọi sau một cập nhật dữ liệu thành công (có nghĩa là sau khi thực hiện cập nhật cơ sở dữ liệu).

Cách phổ biến nhất để thay đổi dữ liệu trong một DataRow là sử dụng chỉ số, tuy vậy nếu bạn có một số thay đổi bạn cũng cần gọi các phương thức BeginEdit() và EndEdit() methods.

Khi một cập nhật được tạo ra trên một cột trong một DataRow, sự kiện ColumnChanging sẽ được phát ra trên các dòng của DataTable. Nó cho phép bạn ghi đè lên thuộc tính ProposedValue của các lớp DataColumnChangeEventArgs, và thay đổi nó nếu muốn. Cách này cho phép các giá tri trên cột có hiệu lực. Nếu bạn gọi BeginEdit() trước khi tạo thay đổi, sự kiện ColumnChanging vẫn xảy ra. Chúng cho phép bạn tạo một sự thay đổi kép khi cố gọi EndEdit(). Nếu bạn muốn phục hồi lại giá trị gốc, hãy gọi CancelEdit().

Một DataRow có thể liên kết với một vài dòng khác của dữ liệu. Điều này cho phép tạo các liên kết có thể điều khiển được giữa các dòng, đó là kiểu master/detail. DataRow chứa một phương thức GetChildRows() dùng để thay đổi một mảng các dòng liên quan đến các cột từ một bản khác trong cùng DataSet như là dòng hiện tại.

7.8. Tạo một DataSet

Trước tiên, chúng ta đã định nghĩa sơ đồ của bộ dữ liệu, với đầy đủ các DataTable, DataColumn, Constraint, và những gì cần thiết, nên tạo DataSet với một vài thông tin bổ sung. Có hai cách chính để đọc dữ liệu từ một nguồn bên ngoài và chèn nó vào DataSet:

- Dùng trình cung cấp dữ liệu
- Đọc XML vào trong DataSet

7.8.1. Tạo một DataSet dùng DataAdapter

Đoạn mã về dòng dữ liệu được giới thiệu trong lớp SqlDataAdapter, được trình bày như sau:

```
string select = "SELECT ContactName,CompanyName FROM Customers";
SqlConnection conn = new SqlConnection(source);
SqlDataAdapter da = new SqlDataAdapter(select , conn);
DataSet ds = new DataSet();
da.Fill(ds , "Customers");
```

Hai dòng in đậm chỉ ra cách dùng của SqlDataAdapter – OleDbDataAdapter cũng có nhưng tính năng ảo giống như Sql equivalent.

SqlDataAdapter và OleDbDataAdapter là hai lớp xuất phát từ một lớp cơ bản chứ không phải là một bộ các giao diện, và nhất là các lớp SqlClient, hoặc OleDb. Cây kế thừa được biểu diễn như sau:

```
System.Data.Common.DataAdapter
        System.Data.Common.DbDataAdapter
                System.Data.OleDb.OleDbDataAdapter
                System.Data.SqlClient.SqlDataAdapter
```

Trong quá trình lấy dữ liệu từ một DataSet, cần phải có một vài lệnh được dùng để chọn dữ liệu. Nó có thể là một câu lệnh SELECT, một stored procedure, hoặc OLEDB provider, một TableDirect command. Ví dụ trên sử dụng một trong những cấu trúc sẵn có trong SqlDataAdapter để truyền câu lệnh SELECT vào một SqlCommand, và phát nó khi gọi phương thức Fill() trên adapter.

7.8.2. Sử dụng một Stored Procedure trong một DataAdapter

Trước tiên chúng ta cần định nghĩa một stored procedure và cài nó vào cơ sở dữ liệu database. Stored procedure để SELECT dữ liệu như sau:

```
CREATE PROCEDURE RegionSelect AS
        SET NOCOUNT OFF
        SELECT * FROM Region
GO
```

Ví dụ này chỉ là một câu lệnh SQL đơn giản. Stored procedure này có thể đánh vào SQL Server Query Analyzer, hoặc bạn có thể chạy file StoredProc.sql để sử dụng ví dụ này.

Tiếp theo, chúng ta cần định nghĩa một SqlCommand để thực thi stored procedure này. Một lần nữa mã rất đơn giản, và hầu hết đã được đưa ra trong các phần trên:

```
private static SqlCommand GenerateSelectCommand(SqlConnection conn )
{
    SqlCommand  aCommand = new SqlCommand("RegionSelect" , conn);
    aCommand.CommandType = CommandType.StoredProcedure;
    aCommand.UpdatedRowSource = UpdateRowSource.None;
    return aCommand;
}
```

Phương thức này phát ra SqlCommand để gọi thủ tục RegionSelect khi thực thi. Và cuối cùng là móc nói nó với một SqlDataAdapter thông qua lời gọi phương thức Fill():

```
DataSet ds = new DataSet();
// Create a data adapter to fill the DataSet
SqlDataAdapter da = new SqlDataAdapter();
// Set the data adapter's select command
da.SelectCommand = GenerateSelectCommand (conn);
da.Fill(ds , "Region");
```

Ví dụ trên tạo ra một SqlDataAdapter mới, xem SqlCommand được phát ra thông qua thuộc tính SelectCommand của data adapter, và gọi Fill(), để thực thi stored procedure và chèn tất cả các dòng vào the Region DataTable.

7.8.3. Tạo một DataSet từ XML

Ngoài việc tạo sơ đồ cho một DataSet và các bảng tương ứng, một DataSet có thể đọc và ghi các dữ liệu của XML, giống như một file trên đĩa, một stream, hoặc một text reader.

Để load XML vào một DataSet, đơn giản gọi một trong những phương thức ReadXML(), chẳng hạn như đoạn mã sau, dùng để đọc từ một file trên đĩa:

```
DataSet ds = new DataSet();
ds.ReadXml("C:\\MyData.xml");
```

Câu hỏi ôn tập:

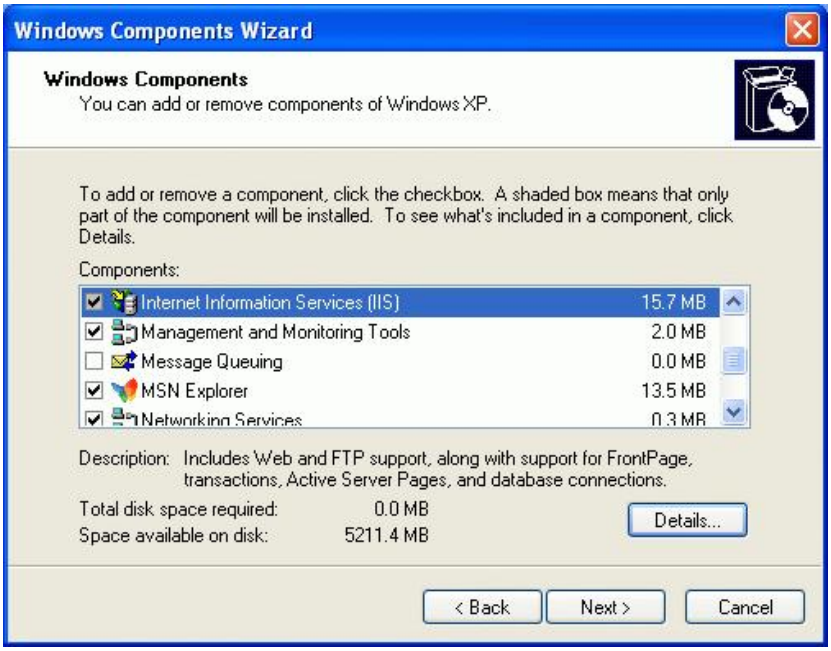
1. So sánh giữa ADO và ADO.NET
2. Các cách sử dụng connection có hiệu quả
3. Làm thế nào để gọi một store procedure
4. Thành phần nào không phải là thuộc tính của đối tượng DataAdapter
   a. SQLCommand b. DeleteCommand
   c. UpdateCommand d. InsertCommand
5. Lệnh nào dùng để hoàn tất một giao dịch (transaction)
   a. Finish b. Commit c. Rollback d. Update
6. Phương thức nào sau đây chắc chắn không làm thay đổi dữ liệu
   a. ExecuteNonQuery b. ExecuteReader
   c. ExecuteScalar d. ExecuteReadOnly
7. Đối tượng transaction được tạo ra bởi phương thức nào
   a. NewTransaction b. StartTransaction
   c. BeginTransaction d. CreateTransaction

PHỤ LỤC

I. CẤU HÌNH WEBSERVER

1.1. Cài đặt Internet Information Services (IIS)

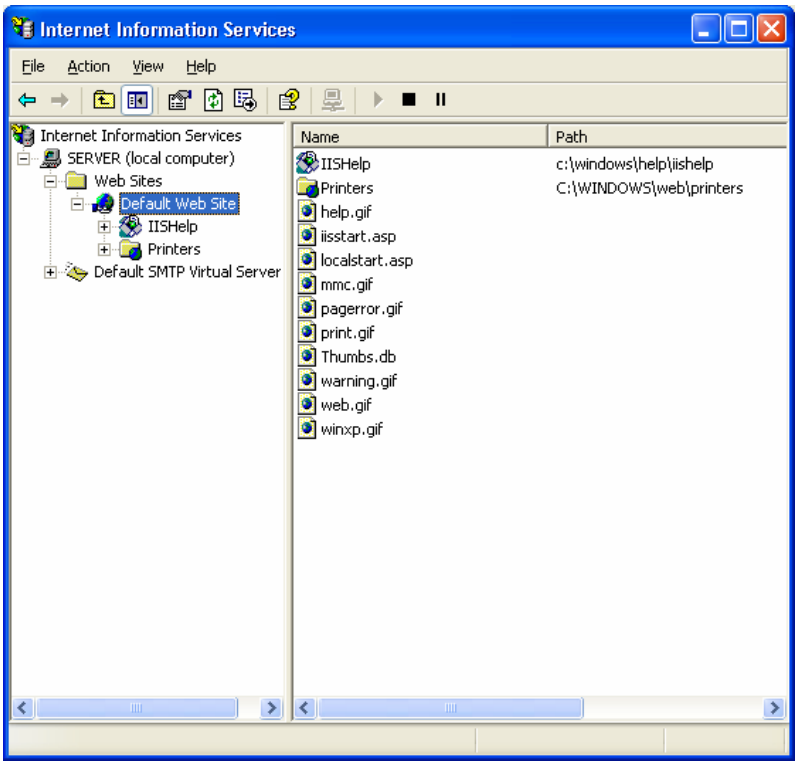
- Để một máy được xem như là một WebServer riêng, ta phải cài đặt:
  - Đối với hệ điều hành Windows 9x: Cài đặt phần mềm Personal Web Server
  - Đối với hệ điều hành Windows 2k, XP: Cài đặt component: Internet Information Services
- . Để cài Internet Information Services trong MS Windows XP Professional, ta bắt đầu chọn:
  - Start, Settings, Control Panel, Add/Remove Programs và nhấp đơn (click) Add/Remove Windows Components, chọn Internet Information Server như sau:



- Nếu ta nhấp đơn nút <Details>, ta có thể tự do lựa chọn thêm hay bớt các thành phần trong IIS, ví dụ như ta có thể bố trí thêm File Transfer Protocol Service (FTP Server) để quản lý một cách hiệu quả hơn việc tải lên (upload) hay tải xuống (download) các hồ sơ (documents) hay tập tin (files).
- Nhấp nút <Next>, Windows XP Professional sẽ bắt đầu tiến trình cài đặt IIS.
- Để xác định việc cài thành công Web Server, ta có thể thử như sau:
  - Mở Browser của bạn, ví dụ như Microsoft Internet Explorer và gõ hàng chữ như sau vào hộp địa chỉ http://localhost

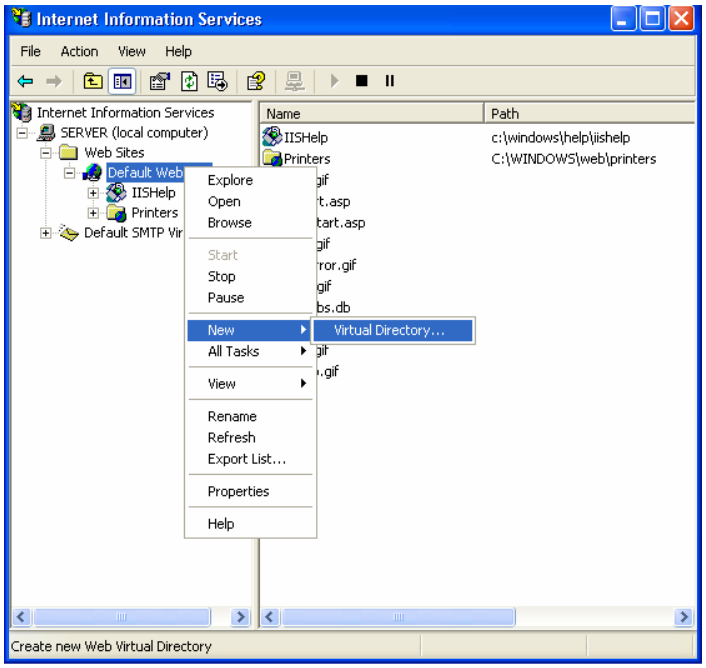
1.2. Khởi động Internet Information Services

Start\Setting\Control Panel\Administrative Tools\Internet Information Services

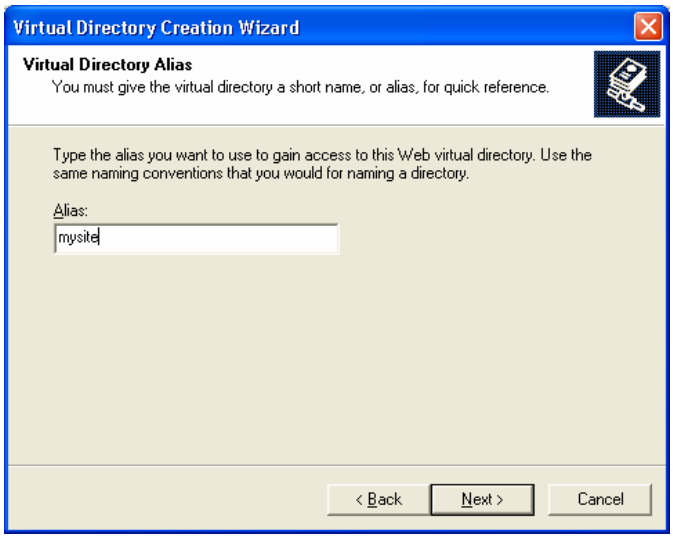


1.3. Tạo thư mục ảo

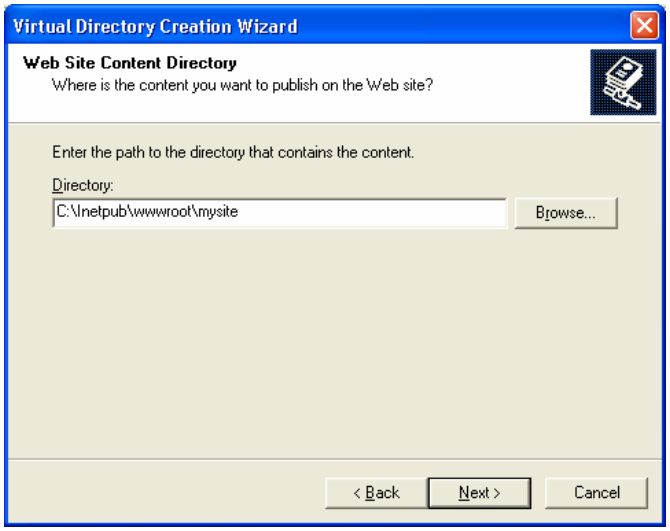
- Nhấp phải vào Default Web Site, chọn New, Virtual Directory:

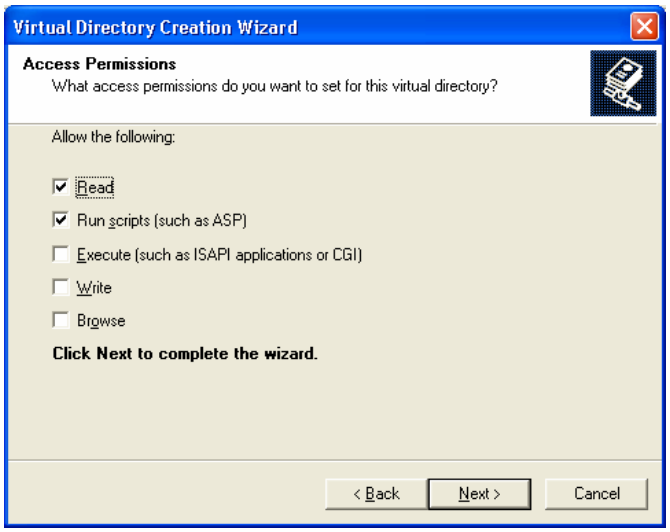


- Gõ bí danh (Alias)

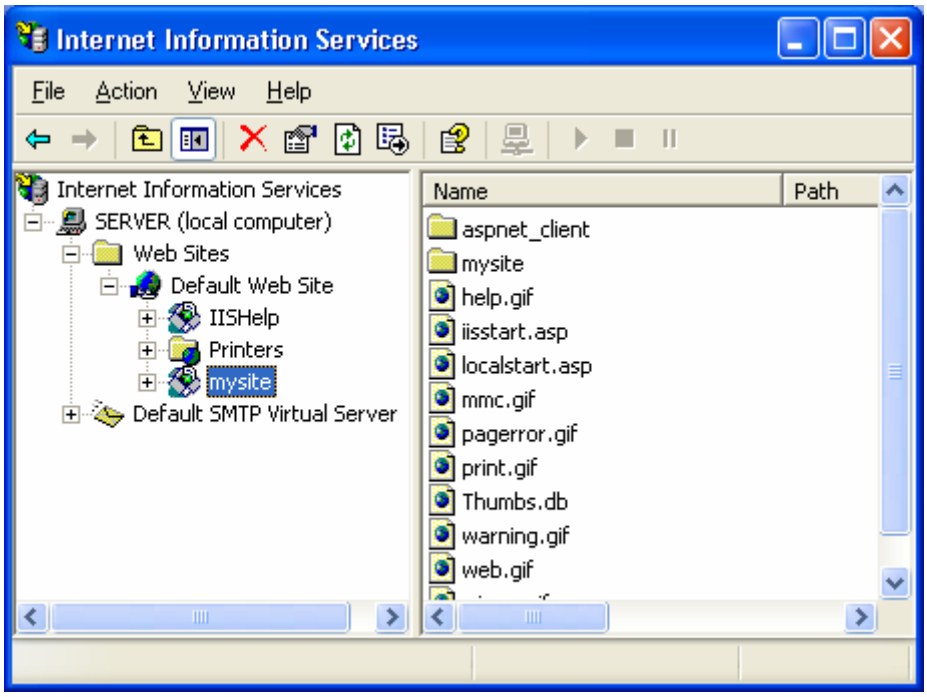


- Lựa chọn thư mục





- Chọn <Next>, chúng ta có kết quả như sau:



II. NHÚNG ĐOẠN JAVASCRIP VÀO TRANG WEB

2.1. Giới thiệu

- JavaScript là ngôn ngữ dưới dạng kịch bản (script) có thể gắn liền với các file HTML.
- JavaScript có kịch bản ở trình khách (client).
- Thường đáp ứng các sự kiện mà HTML không hỗ trợ.
- Trình duyệt nào hỗ trợ JavaScript thì có thể thực hiện được các lệnh của JavaScript.

2.2. Nhúng Javascript vào tệp HTML

- Bạn có thể nhúng JavaScript 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 nhiều hơn cả.

2.2.1. Sử dụng thẻ SCRIPT

- Script được đưa vào file HTML bằng cách sử dụng cặp thẻ

  <SCRIPT>

  //các đoạn lệnh của javascript

<\SCRIPT>.

- Các thẻ <SCRIPT> có thể xuất hiện trong phần <HEAD> hay <BODY> của file HTML. Nếu đặt trong phần <HEAD>, nó sẽ được tải và sẵn sàng trước khi phần còn lại của văn bản được tải.
- Thuộc tính được định nghĩa hiện thời cho thẻ <SCRIPT> là "LANGUAGE=" dùng để xác định ngôn ngữ script được sử dụng. Có hai giá trị được định nghĩa là "JavaScript" và "VBScript". Với Chương trình viết bằng JavaScript bạn sử dụng cú pháp sau :

```
<SCRIPT LANGUAGE="JavaScript">
    // INSERT ALL JavaScript HERE
</SCRIPT>
```

- Điểm khác nhau giữa cú pháp viết các ghi chú giữa HTML và JavaScript là cho phép bạn ẩn các mã JavaScript trong các ghi chú của file HTML, để các trình duyệt cũ không hỗ trợ cho JavaScript có thể đọc được nó như trong ví dụ sau đây:

```
<SCRIPT LANGUAGE="JavaScript">
    <!-- From here the JavaScript code hidden
    // INSERT ALL JavaScript HERE
    // This is where the hidden ends -->
</SCRIPT>
```

- Dòng cuối cùng của script cần có dấu // để trình duyệt không diễn dịch dòng này dưới dạng mã JavaScript.

2.2.2. Sử dụng một file nguồn JavaScript

- Thuộc tính SRC của thẻ <SCRIPT> cho phép bạn chỉ rõ file nguồn JavaScript được sử dụng (dùng phương pháp này hay hơn nhúng trực tiếp một đoạn lệnh JavaScript vào trang HTML).
- Cú pháp:

```
<SCRIPT SRC="file_name.js">
    ....
</SCRIPT>
```

- Thuộc tính này rấy hữu dụng cho việc chia sẻ các hàm dùng chung cho nhiều trang khác nhau. Các câu lệnh JavaScript nằm trong cặp thẻ <SCRIPT> và </SCRIPT> có chứa thuộc tinh SRC trừ khi nó có lỗi.
- 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.

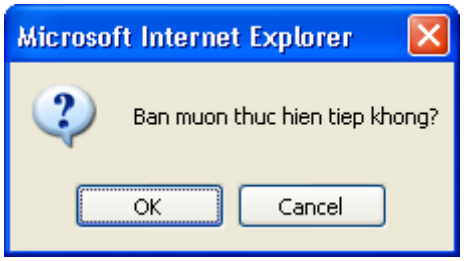
- Tên file của các hàm JavaScript bên ngoài cần có đuôi .js, và server sẽ phải ánh xạ đuôi .js đó tới kiểu MIME application/x-javascript.

2.2.3. Sử dụng JavaScript trong Asp.NET

- Như chúng ta đã biết trên Web form trong ASP.NET để xuất hiện một hộp thoại (khi muốn thông báo, muốn lựa chọn...) như Winform là rất khó. Nhưng chúng ta có thể cho hộp thoại xuất hiện bằng cách kết hợp với đoạn JavaScript.
- Ví dụ:
  - Tạo một button trên trang Web, đặt thuộc tính ID: btnThucHien
  - Trong sự kiện Page_Load thêm vào đoạn lệnh như sau

```
protected void Page_Load(object sender, EventArgs e)
{
  btnThucHien.Attributes.Add("onclick","javascript:if(confirm('Ban
  muon thuc hien tiep khong?')= = false) return false;");
}
```

  - Thực hiện trang, nhấp chuột vào nút sẽ có dòng thông báo



TÀI LIỆU THAM KHẢO

Tiếng Việt

[1]. Nguyễn Thiên Bằng, *Giáo trình SQL Server 2000*, NXB Lao động - Xã hội, 2005

[2]. Phan Hoàng, Anh Quang, *Giáo Trình Tự Học Lập Trình C#,* NXB Văn hóa Thông tin, 2007

[3]. Phạm Hữu Khang, *C# 2005 - Lập Trình Cơ Bản*, NXB Lao động - Xã hội, 2006

[4]. Phạm Hữu Khang, C# *2005 - Lập Trình Cơ Sở Dữ Liệu*, Tập 4, NXB Lao động - Xã hội, 2006

[5]. Nguyễn Văn Lân, *Kỹ Thuật Xây Dựng Ứng Dụng ASP.NET*, Tập 1, NXB Lao động - Xã hội, 2008

[6]. Nguyễn Văn Lân, Phương Lan, *Kỹ Thuật Xây Dựng Ứng Dụng ASP.NET*, T1, NXB Lao động - Xã hội.

[7]. Nguyễn Trường Sinh, *Học Thiết Kế Web Bằng Hình Minh Họa*, Tập2, NXB Phương Đông, 2007

[8]. Nguyễn Trường Sinh, *Thiết Kế Web Với JavaScript Và Dom*, NXB Phương Đông, 2007

[9]. Đậu Quang Tuấn, *Thiết Kế Trang Web Bằng FrontPage 2003*, NXB Giao thông vận tải, 2006

Tiếng Anh

[1]. Archor, Tom. *Inside C#*, Redmond, WA: Microsoft Press, 2001

[2]. Bill Evjen, Scott Hanselman, Farhan Muhammad, S. Srinivasa Sivakumar, Devin Rader, *Professional ASP.NET 2.0*, 2005

[3]. Liberty Jesse, Dan Hurwitz, *Programming ASP.NET*, Second Edition, 2003

[4]. Liberty Jesse. *Programming C#*, Sebastopol, CA: O'Reilly & Associates, 2001

Trang Web

- http://www.google.com.vn
- http://www.oreilly.com