

**TRUNG TÂM HƯNG YÊN – APTECH**

*Địa chỉ : Tầng 2, Nhà A – Đại học SPKT Hưng Yên*  
*Điện thoại : 0321-713.319; Fax: 0321-713.015*  
*E-mail : [aptech@utehy.edu.vn](mailto:aptech@utehy.edu.vn);*  
*Website : <http://www.aptech.utehy.vn>*



# **TÀI LIỆU**

## **KHÓA HỌC LẬP TRÌNH ASP.NET**

*Biên soạn:*

- *Nguyễn Minh Quý*
- *Phạm Ngọc Hưng*
- *Lê Quang Lợi*

**HƯNG YÊN 7/2008**

## MỤC LỤC

<b>BÀI SỐ 1: MỞ ĐẦU VỀ ASP.NET.....</b>	<b>6</b>
1.1. Giới thiệu tổng quan công nghệ .NET .....	6
1.1.1 Sự ra đời của .NET .....	6
1.1.2 .NET Framework là gì .....	7
1.1.3 Một số ưu điểm chính của .NET framework .....	9
1.2. Giới thiệu ASP.NET .....	10
1.3. Cài đặt Visual Studio.NET 2008 .....	10
1.3.1 Các phiên bản .NET .....	10
1.3.2 Cài đặt Visual Studio.NET 2008 .....	10
1.4. Giới thiệu môi trường tích hợp (IDE) của ASP.NET .....	11
1.5. Tạo/lưu/mở/chạy ứng dụng ASP.NET .....	13
1.5.1 Tạo mới .....	13
1.5.2 Lưu ứng dụng Web .....	14
1.5.3 Mở (Chạy) ứng dụng .....	14
1.6. Cơ bản về CSS và DHTML .....	15
1.6.1 CSS .....	15
1.6.2 DHTML .....	15
1.7. Định dạng các thẻ sử dụng CSS .....	16
1.7.1 Định dạng ở mức dòng (Inline) .....	16
1.7.2 Định dạng bởi bộ chọn ID .....	16
1.7.3 Định dạng bởi bộ chọn thẻ (tag) .....	16
1.7.4 Định dạng bởi lớp (Class) .....	17
1.7.5 Vấn đề tổ chức lưu trữ .....	19
1.8. Truy xuất thuộc tính các thẻ HTML và CSS bằng JavaScript .....	19
1.8.1 Truy xuất các thuộc tính của thẻ .....	19
1.8.2 Truy xuất các thuộc tính CSS .....	20
<b>BÀI SỐ 2: THỰC HÀNH .....</b>	<b>22</b>
<b>BÀI SỐ 3: ASP.NET và Web form.....</b>	<b>32</b>
3.1 Mô hình lập trình phía máy chủ .....	32
3.2 Cơ chế xử lý file ASP.NET phía máy chủ .....	34
3.3 Một số ví dụ minh họa .....	36
3.3.1 Yêu cầu xử lý tại phía server thông qua Runat="Server" .....	36
3.3.2 Yêu cầu xử lý bên phía server thông qua cặp thẻ <% %> .....	37
3.3.3 Yêu cầu xử lý bên server thông qua Script .....	38
3.3.4 Yêu cầu xử lý bên phía server bằng cách đặt trong Code file .....	38
3.4 Webform trong ASP.NET .....	39
3.5 Tìm hiểu cấu trúc trang ASP.NET .....	39
3.6 Code behind và viết code phía Server .....	42
3.7 HTML Server Controls và Web controls .....	43
3.7.1 Giới thiệu .....	43
3.7.2 Cách thức tạo phần tử HTML Server Control và ASP.NET control .....	43
<b>BÀI 4: THỰC HÀNH .....</b>	<b>45</b>
<b>BÀI 5: Tìm hiểu và sử dụng các Server/Ajax Controls .....</b>	<b>53</b>
5.1 HTML Server Controls .....	53
5.2 Web server Controls .....	53
5.2.1 Khai báo (tạo các phần tử web server control) .....	53
5.2.2 Cơ chế xử lý các phần tử web server control .....	54
5.2.2 Thực thi các câu lệnh tại phía server .....	59

5.2.3 Mô hình xử lý sự kiện trong ASP.NET .....	59
5.3 Ajax Control Toolkit.....	60
5.3.1 Giới thiệu .....	60
5.3.2 Hướng dẫn sử dụng một số Ajax Control cơ bản .....	61
5.4 Thảo luận công nghệ Ajax .....	62
<b>BÀI 6: THỰC HÀNH .....</b>	<b>63</b>
<b>BÀI 7: Tạo và sử dụng Custom Control.....</b>	<b>67</b>
7.1 Giới thiệu User Custom Control .....	67
7.2 Các bước tạo User Custom control .....	67
7.3 Thêm các thuộc tính, phương thức và sự kiện vào UCC .....	69
7.3.1 Thêm thuộc tính vào UCC.....	69
7.3.2 Thêm phương thức vào UCC .....	70
7.3.3 Thêm sự kiện vào UC .....	71
7.4 Truy cập thuộc tính, phương thức của các phần tử con trong UCC.....	71
7.5 Minh họa tạo một số điều khiển .....	73
<b>BÀI 8: THỰC HÀNH .....</b>	<b>76</b>
<b>BÀI 9: Các đối tượng trong ASP.NET.....</b>	<b>83</b>
9.1 Request Object.....	83
9.1.1 Đối tượng Request dùng để làm gì ? .....	83
9.1.2 Các thành phần (thuộc tính và phương thức) chính.....	83
9.1.3 Ví dụ sử dụng .....	83
9.2 Response Object .....	86
9.2.1 Đối tượng Response dùng để làm gì ? .....	86
9.2.2 Các thành phần (thuộc tính và phương thức) chính.....	86
9.2.3 Ví dụ sử dụng .....	86
9.3 Server Object .....	87
9.3.1 Đối tượng Server dùng để làm gì ? .....	87
9.3.2 Các thành phần (thuộc tính và phương thức) chính.....	87
9.3.3 Ví dụ sử dụng .....	87
9.4 Session Object .....	87
<b>9.4.1. Biến Session .....</b>	<b>87</b>
<b>9.4.2. Đối tượng Session .....</b>	<b>88</b>
9.5 Application Object.....	88
9.5.1 Đối tượng Application dùng để làm gì ? .....	88
9.5.2. Khái niệm biến toàn ứng dụng .....	88
9.5.3. Đối tượng Application.....	88
<b>Một số bài tập tổng hợp: .....</b>	<b>89</b>
<b>BÀI 10: THỰC HÀNH .....</b>	<b>94</b>
<b>BÀI 11. Truyền dữ liệu giữa các webpage,.....</b>	<b>94</b>
<b>MasterPage và gỡ rối (Debug) chương trình.....</b>	<b>94</b>
11.1 Truyền (Post) dữ liệu giữa các trang bằng mã lệnh C# .....	94
11.2 Truy xuất đến các phần tử bằng phương thức FindControl .....	94
11.3 Truy xuất đến trang gửi thông qua thuộc tính PreviousPage.....	94
11.4 MasterPage .....	94
11.5 Gỡ rối.....	97
11.5.1 Giới thiệu.....	97
11.5.2 Chạy ứng dụng ở chế độ gỡ rối .....	97
11.5.3 Khái niệm điểm dừng .....	97

11.5.4 Chạy từng dòng lệnh với chế độ Step Into (F8) .....	97
11.5.5 Chạy từng dòng lệnh với chế độ Step Over (Shift-F8).....	97
11.5.6 Chạy từng dòng lệnh với chế độ Step Out (Ctrl-Shift-F8) .....	97
11.2 Sử dụng Custom Error page.....	97
11.3 Ghi các vết gây lỗi (Trace errors) .....	97
11.4 Sử dụng công cụ gỡ rối/ Menu Debug .....	97
11.5 Tracing lỗi ở mức trang/ Mức toàn ứng dụng .....	97
<b>BÀI 12: THỰC HÀNH .....</b>	<b>97</b>
<b>BÀI 13: CÔNG NGHỆ ADO.NET .....</b>	<b>98</b>
13.1 Giới thiệu chung .....	98
13.2 Kiến trúc của ADO.NET .....	99
13.3 Các lớp thao tác với CSDL: Connection, Command, .....	100
13.3.1 Lớp Connection.....	100
13.3.2 Lớp Command.....	102
13.3.3 Lớp DataReader .....	104
13.3.7 Lớp DataColumn.....	106
13.3.8 Lớp DataTable.....	106
13.3.9 Lớp DataRow .....	107
13.3.10 Lớp DataSet.....	108
13.3.11 Lớp DataAdapter .....	108
<b>BÀI 14: THỰC HÀNH .....</b>	<b>111</b>
<b>BÀI 15: Tìm hiểu và ứng dụng cơ chế Data Binding.....</b>	<b>118</b>
15.1 Giới thiệu DataBinding.....	118
15.2 Data Binding .....	118
15.2.1 Dạng gắn kết dữ liệu đơn (Single DataBinding) .....	118
15.2.2 Dạng gắn kết dữ liệu có sự lặp lại (Repeated Data Binding) .....	119
15.3 Các điều khiển Data Source (Data source controls).....	121
15.3.1 Giới thiệu về DataSource controls .....	121
15.3.2 Sử dụng SqlDataSource để chọn (Select) dữ liệu .....	122
15.3.3 Sử dụng SqlDataSource để cập nhật dữ liệu .....	124
15.3.4 Xóa bản ghi trong CSDL bằng SqlDataSource .....	127
<b>BÀI 16: THỰC HÀNH .....</b>	<b>129</b>
<b>BÀI 17: Làm việc với GridView .....</b>	<b>133</b>
17.1 Giới thiệu tổng quan .....	133
17.2 Tìm hiểu lớp GridView .....	133
17.2.1 Các thuộc tính và cột thuộc tính.....	133
17.2.2 Các style áp dụng cho GridView.....	134
17.2.3 Các sự kiện .....	135
17.2.4 Các phương thức .....	136
17.3 Các tính năng hỗ trợ của GridView .....	137
17.3.1 Phân trang.....	137
17.3.2 Tính năng tự động sắp xếp .....	139
17.3.3 Các mẫu hiển thị - Template .....	140
17.4 Tạo các cột tùy biến HyperLink, BoundColumn.....	141
17.4.1 Tạo cột BoundField thủ công .....	141
17.4.2 Tạo một cột hyperlink .....	141
17.5 Tạo và xử lý các cột Select, Edit, Delete, Update...	144
17.5.1 Thêm cột Select, Edit - Update, Delete .....	144
17.5.2 Cập nhật dữ liệu .....	145

17.5.3 Xóa dữ liệu .....	146
<b>BÀI 18: THỰC HÀNH .....</b>	<b>148</b>
<b>BÀI 19: Sử dụng Templates .....</b>	<b>155</b>
19.1 Giới thiệu tổng quan .....	155
19.2 Các điều khiển hỗ trợ Templates.....	155
19.2.1 Một số điều khiển hỗ trợ Template thường dùng .....	155
19.2.2 Các loại Template.....	155
19.3 Repeater control, DataList control, GridView control.....	156
19.3.1 Tạo template với GridView.....	156
19.3.2 Tạo template với DataList .....	160
19.3.3 Tạo Template với Repeater (light-weight) .....	161
20. Đóng gói website .....	162
<b>BÀI 20: THỰC HÀNH .....</b>	<b>163</b>

**TRUNG TÂM HƯNG YÊN – APTECH**

*Địa chỉ : Tầng 2, Nhà A – Đại học SPKT Hưng Yên*  
*Điện thoại : 0321-713.319; Fax: 0321-713.015*  
*E-mail : [aptech@utehy.edu.vn](mailto:aptech@utehy.edu.vn);*  
*Website : <http://www.aptech.utehy.vn>*



# **TÀI LIỆU**

# **KHÓA HỌC LẬP TRÌNH ASP.NET**

*Biên soạn:*

- *Nguyễn Minh Quý*
- *Phạm Ngọc Hưng*
- *Lê Quang Lợi*

**HƯNG YÊN 7/2008**

## BÀI SỐ 1: MỞ ĐẦU VỀ ASP.NET

### Mục tiêu: Kết thúc bài học, sinh viên có thể

- Nêu được các đặc điểm chính của công nghệ .NET
- Mô tả được các thành phần cơ bản bên trong .NET Framework
- Cài đặt và cấu hình hệ thống để chạy các trang ASP/ ASP.NET
- Sử dụng hệ thống IDE của VS 2008 để tạo, lưu và chạy ứng dụng web
- Nêu được các ưu điểm của web động - DHTML
- Định dạng trang web sử dụng CSS
- Truy xuất các thuộc tính của phần tử web thông qua CSS và Javascript

### Nội dung

## 1.1. Giới thiệu tổng quan công nghệ .NET

### 1.1.1 Sự ra đời của .NET

Trước đây và cả ngày nay, trong lĩnh vực phát triển phần mềm có rất nhiều (hàng ngàn thậm chí hàng vạn) ngôn ngữ lập trình được sử dụng để phát triển phần mềm (như Delphi, Ada, Cobol, Fortran, Basic, LISP, Prolog, Foxpro, Java, Pascal, C/C++, Visual Basic, VC++, C#...). Mỗi ngôn ngữ đều có những ưu và nhược điểm riêng, chẳng hạn Fortran là lựa chọn số một cho các tính toán khoa học; Prolog là lựa chọn rất tốt để phát triển các phần mềm thông minh (AI, Expert Systems...); Java có lợi thế phát triển các ứng dụng mạng, ứng dụng Mobile và độc lập hệ điều hành (Write One – Run Everywhere); Visual Basic tỏ ra dễ học và dễ phát triển các ứng dụng Winform; C# vượt trội bởi sự kết hợp giữa sức mạnh của C++ và sự dễ dàng của Visual Basic...

Những ưu điểm có tính đặc thù của từng ngôn ngữ là điều đã được khẳng định. Tuy nhiên, điều mà ai cũng thấy rõ là rất khó để có thể tận dụng được sức mạnh của tất cả các ngôn ngữ lập trình trong một dự án phần mềm, chẳng hạn không thể hoặc rất khó khăn để viết một ứng dụng có sử dụng đồng thời cả ngôn ngữ Visual Basic và Java hay Foxpro với Delphi v.v... Nói cách khác, việc “liên thông” giữa các ngôn ngữ là gần như không thể.

Cũng do sự khác biệt giữa các ngôn ngữ lập trình mà việc tiếp cận hay chuyển đổi sang ngôn ngữ lập trình mới sẽ tốn rất nhiều thời gian (Tuy rằng về tư tưởng và nguyên lý có tương tự nhau). Vì vậy, khi các dự án sử dụng ngôn ngữ lập trình khác nhau thì chi phí cho chuyển đổi/ học hỏi sẽ là rất lớn, gây lãng phí thời gian không cần thiết và chất lượng phần mềm chắc chắn không cao.

Ngoài ra, cùng với sự phát triển như vũ bão của Internet thì mô hình phát triển ứng dụng cũng rất khác xưa. Các ứng dụng ngày nay không chỉ chạy riêng lẻ (stand-alone) trên máy tính PC mà còn có thể chạy trên môi trường mạng, cung cấp hay truy cập các dịch vụ từ xa (ứng dụng phân tán). Vai trò của phần mềm đã dần chuyển từ chỗ cung cấp các chức năng (Functional) cụ thể sang cung cấp các dịch vụ (Services).

Từ những hạn chế trong quá trình phát triển phần mềm như đã nêu, đòi hỏi phải có một cách tiếp cận sao cho tối ưu nhất, vừa đảm bảo tốn ít chi phí chuyển đổi vừa đảm bảo nhiều người có thể tham gia cùng một dự án mà không nhất thiết phải viết trên cùng một ngôn ngữ lập trình, đồng thời ứng dụng phải hoạt động tốt trong môi trường mạng Internet. Đó chính là lý do để Microsoft cho ra công nghệ phát triển phần mềm mới .NET!

Microsoft .NET là một nền tảng (Platform) phát triển ứng dụng mới và hoàn chỉnh nhất từ trước tới nay. Sự ra đời của Microsoft.NET có tính cách mạng, nó đem đến cho các nhà lập trình một phong cách phát triển phần mềm đột phá, khắc phục hầu hết các hạn chế trước

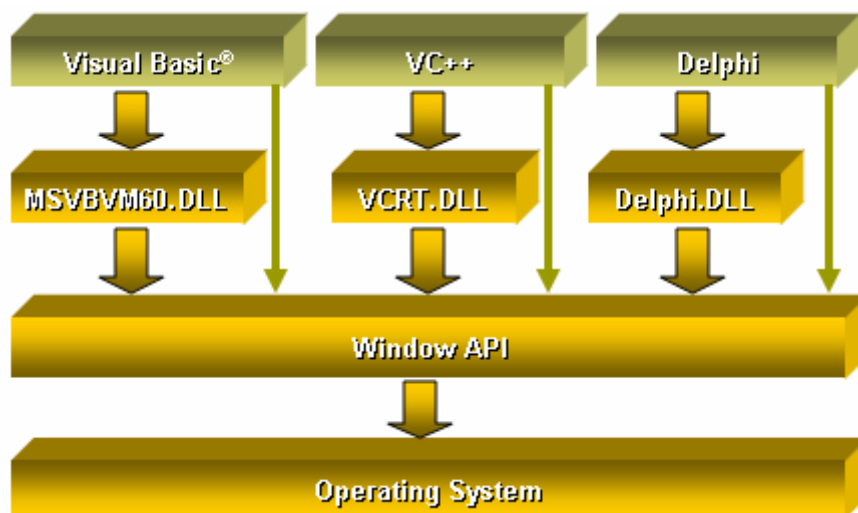
đây của các ngôn ngữ lập trình. Việc sử dụng .NET không chỉ giúp phát triển các ứng dụng đơn lẻ mà còn có thể phát triển các ứng dụng phân tán ở qui mô rất lớn; .NET làm giảm thiểu thời gian phát triển ứng dụng, nâng cao rõ rệt chất lượng sản phẩm phần mềm.

Phiên bản .NET đầu tiên (v 1.0) được Microsoft đưa ra thị trường vào năm 2001.

### 1.1.2 .NET Framework là gì .

Thông thường, mỗi ngôn ngữ lập trình đều có một tập các thư viện riêng, chẳng hạn: VC++ thì có thư viện chính là **msvcr.dll**; Visual Basic thì có **msvbvm60.dll** ... Các thư viện này chứa các hàm, thủ tục cơ bản của mỗi ngôn ngữ (ví dụ hàm, thủ tục xử lý xâu, xử lý toán học,...). Tất cả những thứ này có ý nghĩa logic giống nhau nhưng về cách sử dụng hay cú pháp thì hầu như là khác nhau. Điều này khiến cho một lập trình viên C++ không thể áp dụng những kiến thức họ biết sang VB hoặc ngược lại. Hơn nữa, việc phát triển bộ thư viện riêng cho mỗi ngôn ngữ như vậy là quá dư thừa.

Ý tưởng của Microsoft đó là **KHÔNG** xây dựng một tập thư viện riêng biệt cho từng ngôn ngữ lập trình mà sẽ xây dựng một bộ thư viện dùng **CHUNG**. Tập thư viện dùng chung này hình thành nên một bộ khung (Framework) để các lập trình viên viết ứng dụng trên bộ khung sẵn có đó. Bộ Khung này thực chất là một tập các thư viện được xây dựng sẵn, đáp ứng mọi nhu cầu phát triển các ứng dụng Desktop, Network, Mobile, web...



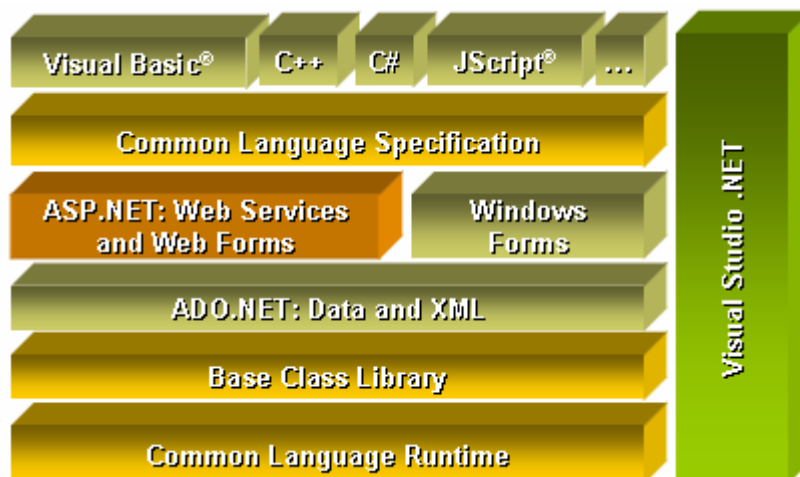
Mô hình xây dựng phần mềm bằng ngôn ngữ truyền thống

### Các thành phần và chức năng chính trong .NET Framework

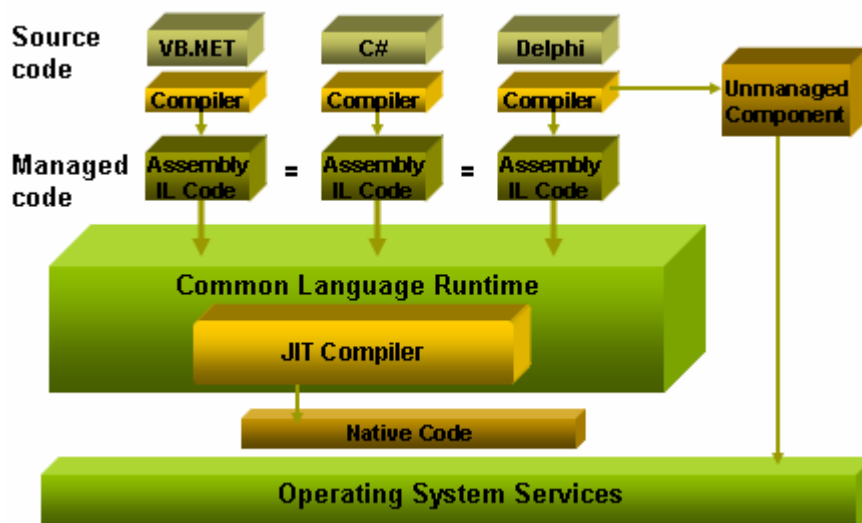
- *Common Language Runtime* (Trình thực thi ngôn ngữ chung): Sau khi ứng dụng được biên dịch ra file “Exe” (exe này khác với file exe thông thường. Nội dung của file exe này tuân theo một chuẩn/ngôn ngữ chung, dù là viết bằng C# hay VB.NET. Ngôn ngữ này gọi là ngôn ngữ chung), tiếp theo để file exe trung gian này có thể chạy được trên máy hiện hành thì cần phải được biên dịch ra mã máy tương ứng. Việc biên dịch và chạy được là nhờ Chương trình thực thi ngôn ngữ chung – CLR (Common Language Runtime).
- *Base Class Library*: Là tập các thư viện chứa các lớp cơ bản để sử dụng trong tất cả các ngôn ngữ .NET. Ví dụ các lớp xử lý xâu, xử lý toán học...
- *ADO.NET*: Là tập các thư viện chuyên dành cho thao tác với Cơ sở dữ liệu.
- *ASP.NET*: Các thư viện dành cho phát triển các ứng dụng Web (webform).



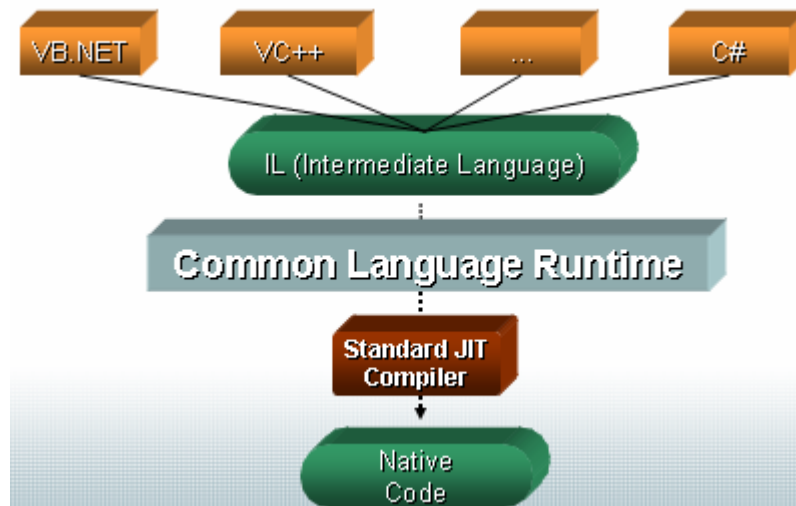
- *Windows Forms*: Các thư viện dành cho phát triển các ứng dụng Windows (winform).
- *Common Language Specification*: Phần này có nhiệm vụ đặc tả ngôn ngữ chung để các chương trình viết trên các ngôn ngữ lập trình khác nhau phải tuân theo. Nói cách khác, biên dịch các chương trình viết trên các ngôn ngữ lập trình khác nhau về một ngôn ngữ thống nhất chung (Common Language). Nhờ điều này mà
- *Các ngôn ngữ lập trình*.



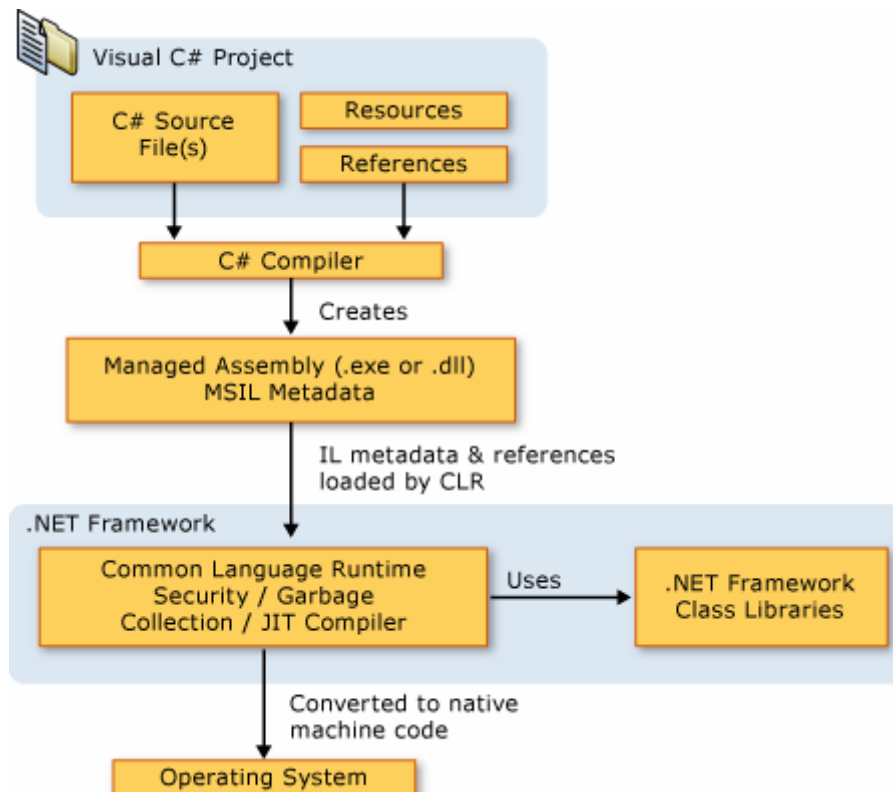
Kiến trúc của .NET Framework



Mô hình biên dịch và thực thi chương trình của ứng dụng .NET (1)



Mô hình biên dịch và thực thi chương trình của ứng dụng .NET (2)



Một cái nhìn khác về mô hình biên dịch và thực thi ứng dụng

### 1.1.3 Một số ưu điểm chính của .NET framework

- Tất cả các ngôn ngữ đều thừa hưởng một thư viện thống nhất. Khi sửa chữa hay nâng cấp thư viện này thì chỉ phải thực hiện một lần.
- Phong cách phát triển ứng dụng nhất quán và tương tự nhau giữa các ngôn ngữ lập trình. Có thể chuyển đổi sang ngôn ngữ lập trình .NET khác nhau một cách dễ dàng.
- Viết các ứng dụng webform không khác nhiều so với ứng dụng winform.
- Cung cấp một tập thư viện truy xuất CSDL thống nhất (ADO.NET) cho mọi ngôn ngữ .NET.

- Hỗ trợ cơ chế “Write one – Run everywhere” (Viết một lần chạy mọi nơi). Một ứng dụng viết bằng .NET có thể chạy trên bất cứ hệ điều hành nào mà không cần phải sửa lại code, miễn là máy đó có cài .NET framework.
- Cung cấp hệ thống kiểu chung (Common Type), do vậy đảm bảo tính thống nhất về kiểu dữ liệu giữa các ngôn ngữ lập trình.
- Cho phép sử dụng nhiều ngôn ngữ lập trình trong cùng một dự án.
- Kết thừa và sử dụng chéo giữa các ngôn ngữ lập trình dễ dàng như trên cùng một ngôn ngữ (Có thể viết một class trên C#, sau đó kế thừa trong VB.NET và ngược lại).
- Việc triển khai (Deploy) các ứng dụng dễ dàng. Chỉ cần Copy-and-run (copy là chạy). Không cần cài đặt và tránh được “địa ngục DLL” như trước đây.

## **1.2. Giới thiệu ASP.NET**

ASP.NET là công nghệ phát triển các ứng dụng trên nền web, thế hệ kế tiếp của ASP (Active Server Page – Trang web được xử lý bên phía máy chủ). ASP.NET là một thành phần nội tại (có sẵn) của .NET Framework. Vì vậy nó tận dụng được sức mạnh của .NET Framework. ASP.NET có một số ưu điểm chính:

- Có thể sử dụng để phát triển các ứng dụng web đủ mọi kích cỡ, từ ứng dụng nhỏ nhất cho đến ứng dụng toàn doanh nghiệp (Enterprise).
- Ứng dụng viết bằng ASP.NET dễ dàng tương thích với nhiều loại trình duyệt khác nhau. Nhà phát triển không cần phải quan tâm nhiều đến trình duyệt nào được sử dụng để duyệt website, điều này sẽ được framework tự render ra mã tương ứng.
- Khi sử dụng bộ IDE của Visual Studio, cách thức lập trình sẽ giống hệt như lập trình winform.
- Truy xuất dữ liệu bằng công nghệ ADO.NET có sẵn của .NET Framework.
- Chạy ứng dụng cực nhanh bởi cơ chế biên dịch và Cached.
- Có thể tăng tốc ứng dụng bằng cách Cache các điều khiển, các trang.
- Bảo mật vượt trội.
- Tốn ít dòng lệnh hơn so với ASP/PHP/Perl khi thực hiện cùng một công việc.
- Dễ dàng bảo trì và dễ đọc hơn bởi Code và Giao diện được tách biệt. Điều này cũng giúp cho tính chuyên biệt hóa cao hơn. (Một người chỉ lo code phần xử lý nghiệp vụ, người khác thì chỉ lo code phần giao diện v.v...).
- ASP sử dụng ngôn ngữ lập trình VB.NET hoặc C# hoặc cả hai để phát triển ứng dụng.

## **1.3. Cài đặt Visual Studio.NET 2008**

### **1.3.1 Các phiên bản .NET**

Cho đến thời điểm này (2008), Visual studio .NET đã có các phiên bản:

- Visual Studio 2003, .NET Framework 1.1
- Visual Studio 2005, .NET Framework 2.0
- Visual Studio 2008, .NET Framework 3.5

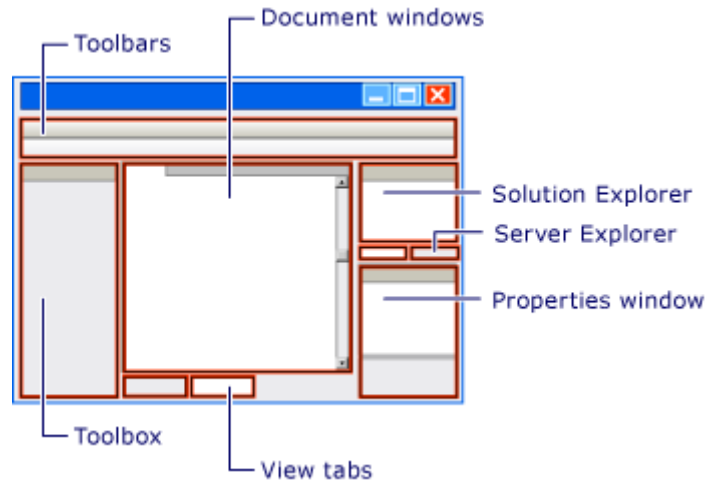
### **1.3.2 Cài đặt Visual Studio.NET 2008**

Bộ Visual Studio.NET 2008 được đóng gói trong một đĩa DVD (tương đương 8 đĩa CD). Trong đó bao gồm cả bộ MSDN. Kích thước khoảng 4.5 GB.

Việc cài đặt vô cùng dễ dàng, chỉ việc chạy file Setup sau đó chọn các mặc định khi được hỏi. Tuy nhiên, để tiết kiệm không gian đĩa thì chỉ nên chọn các sản phẩm cần thiết để cài đặt.

#### **1.4. Giới thiệu môi trường tích hợp (IDE) của ASP.NET.**

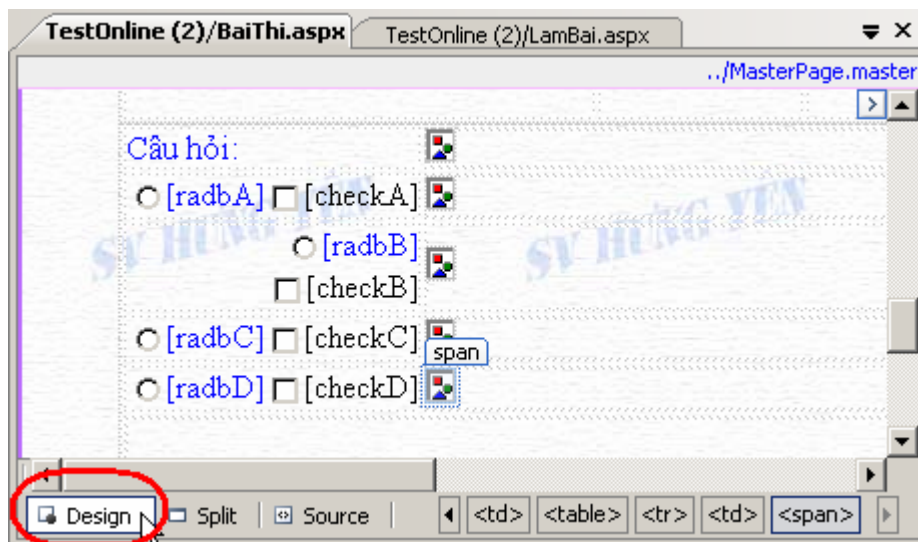
Một điều thật tuyệt vời là Visual Studio sử dụng một trình IDE chung cho toàn bộ ngôn ngữ lập trình (ASP.NET, VB.NET, C#,...). Điều này đảm bảo tính nhất quán cho các ngôn ngữ trên nền .NET, giúp bạn chỉ cần “Học một lần nhưng áp dụng mọi nơi”.



*Cửa sổ giao diện chính của môi trường phát triển tích hợp.*

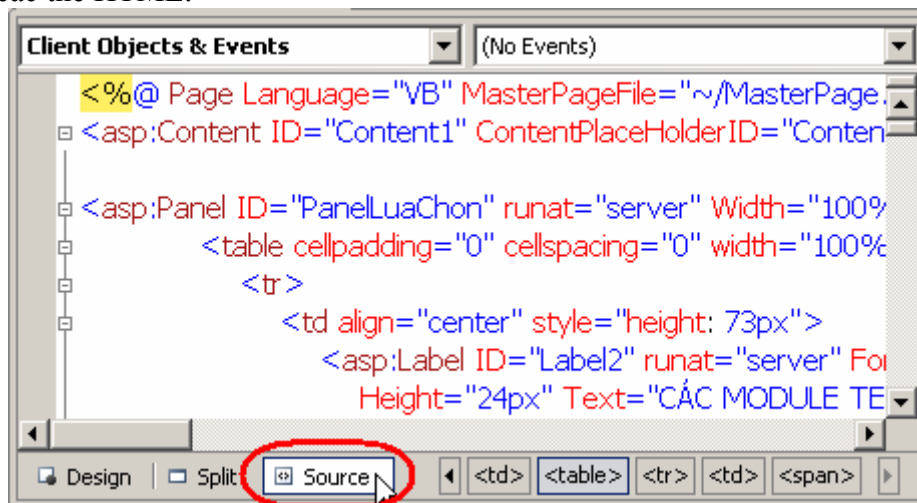
##### **Trong đó:**

- Tab Design để hiển thị trang web ở chế độ Design, tức là cho phép sửa chữa nội dung trang web trực quan.



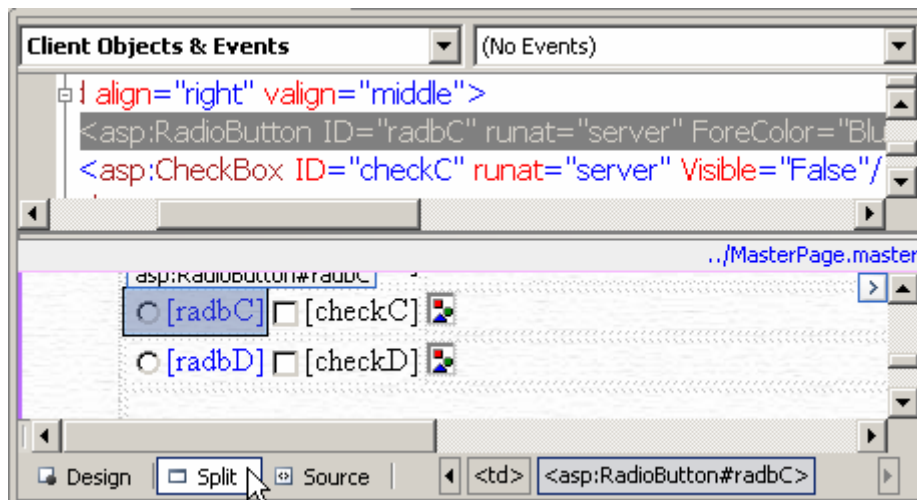
*Mở trang ở chế độ Design*

- Tab Source: Mở trang ở chế độ mã nguồn HTML. Tại đây người dùng có thể soạn thảo trực tiếp các thẻ HTML.

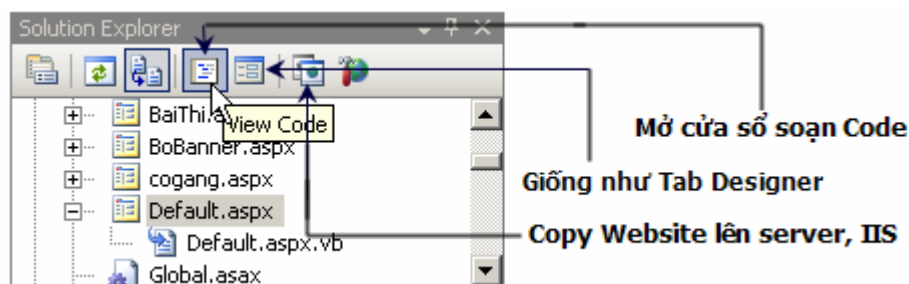


Mở trang ở chế độ Source

- Tab Split: Cho phép xem trang web đồng thời ở cả hai chế độ.

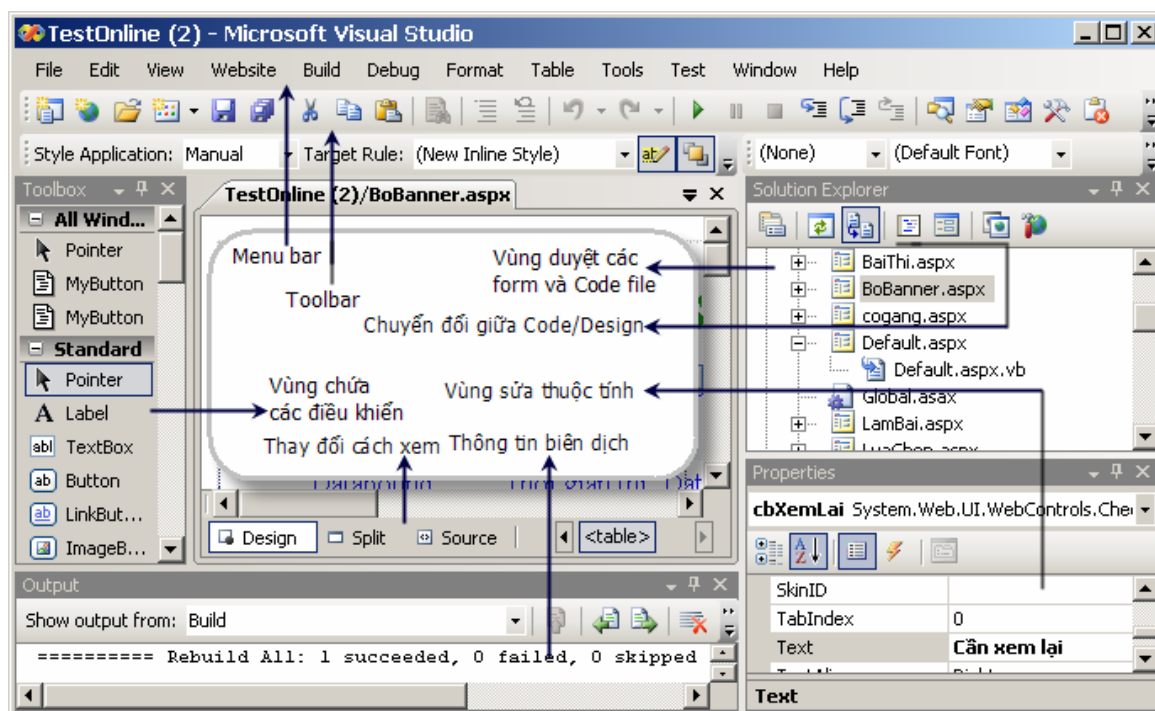


Mở trang ở chế độ kết hợp, vừa xem code HTML vừa xem Design.



Mở cửa sổ soạn Code (C#, VB.NET)


\*\*\* Ngoài thao tác trực tiếp thông qua hệ thống menu, nút lệnh, người dùng còn có thể sử dụng tổ hợp các phím tắt. (Mở menu bar và xem tổ hợp phím tắt bên cạnh). Ví dụ: Shift+F7 để xem ở chế độ Design, F7 xem ở chế độ Code, F4 Focus tới Properties....

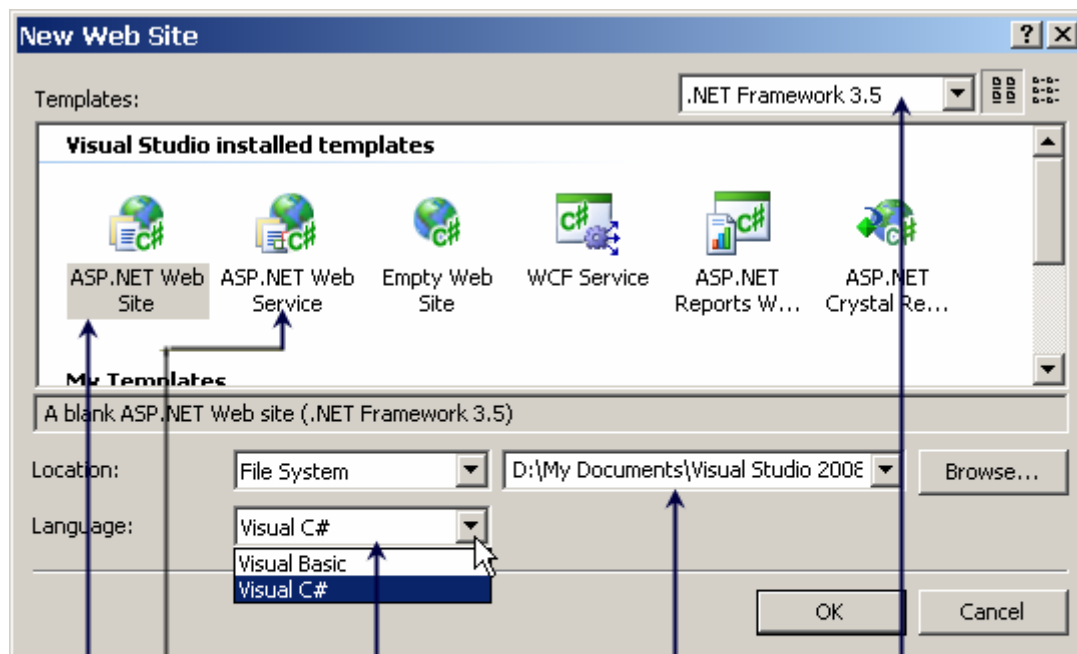


Giao diện của hệ thống IDE.

## 1.5. Tạo/lưu/mở/chạy ứng dụng ASP.NET

### 1.5.1 Tạo mới

Có thể vào menu **File** → **New Website** hoặc biểu tượng  trên thanh công cụ.



### 1.5.2 Lưu ứng dụng Web

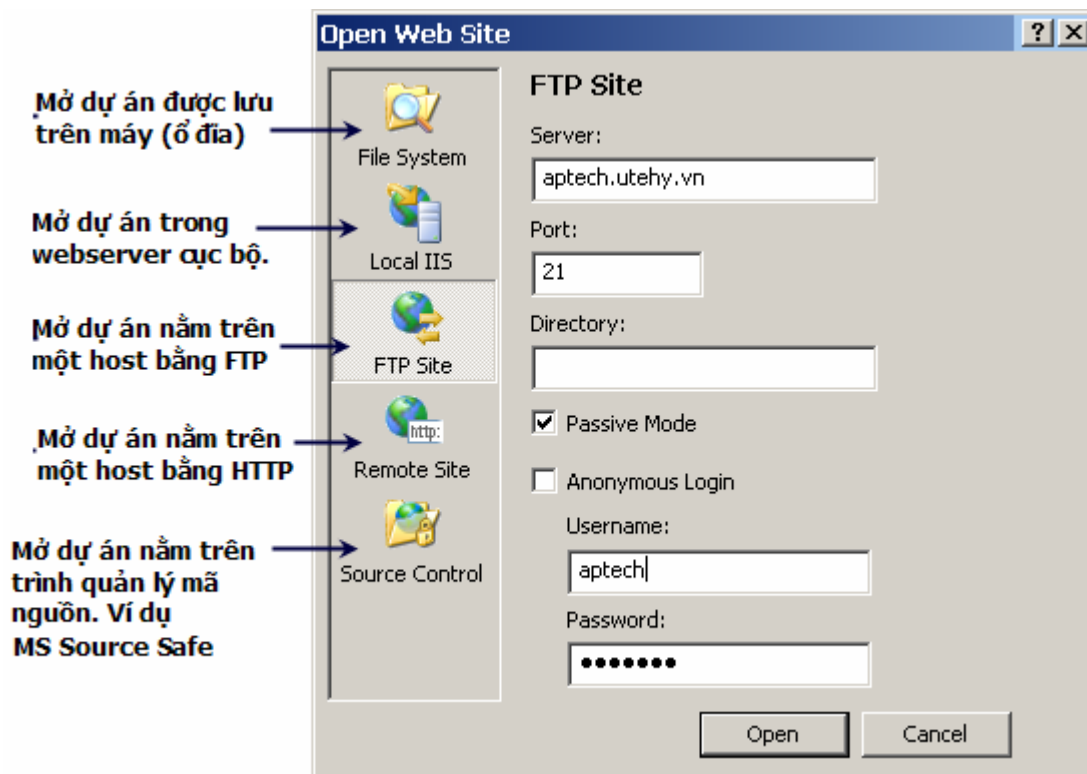
- Nhấn Ctrl-S để lưu trang hiện tại
- Nhấn Ctrl-Shift-S để lưu toàn bộ các trang.

### 1.5.3 Mở (Chạy) ứng dụng

a) Mở ứng dụng web.

- Nhấn tổ hợp phím Alt-Shift-O
- Vào Menu File, chọn : **Open Web Site**


Có thể mở ứng dụng web theo một trong các cách như sau:



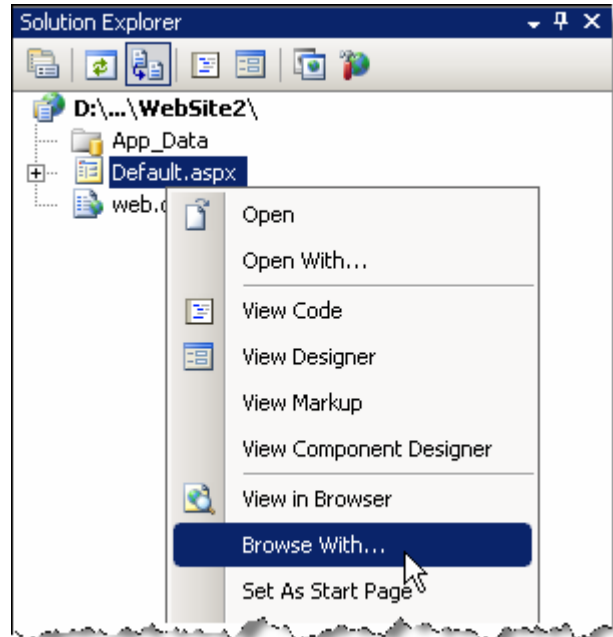
Mở ứng dụng web từ nhiều nguồn.

b) Chạy ứng dụng web

Đối với ASP.NET, toàn bộ ứng dụng web có thể được biên dịch thành file nhị phân để chạy nhanh hơn. Tuy nhiên ASP.NET cũng cho phép người dùng chạy từng trang riêng biệt.

- Nhấn F5 (Hoặc biểu tượng  trên thanh công cụ) để chạy ứng dụng và cho phép Debug trên trình duyệt.
- Nhấn Ctrl-F5 để chạy ứng dụng nhưng không cho Debug trên trình duyệt.
- Trong trường hợp muốn chạy chương trình và gỡ rối ở mức dòng lệnh/ thủ tục thì có thể nhấn F8, Shift-F8.

Người dùng có thể chạy (Browse) trang web bất kỳ bằng cách chọn, sau đó click chuột phải và chọn mục **View In Browser** (Hoặc nhấn tổ hợp phím Ctrl-Shift-W). Trong trường hợp có nhiều trình duyệt trong máy thì có thể chọn trình duyệt mặc định khi View In Browser bằng cách click chuột phải lên trang và chọn **Browse With** như hình bên.



Chọn trình duyệt mặc định

## 1.6. Cơ bản về CSS và DHTML.

### 1.6.1 CSS

Đối với các trang HTML trước đây, việc định dạng (format) các phần tử thường được đặt theo cú pháp dạng, <Loại\_phần\_tử Thuộc\_Tính1=Giá\_Trị1 Thuộc\_Tính2=Giá\_Trị2...>. Đây là cách định dạng có khá nhiều hạn chế, rất khó đọc code cũng như khó bảo trì. Đặc biệt khi xét đến góc độ lập trình.

Để khắc phục được những hạn chế này, hiện nay người ta đề xuất ra một qui tắc định dạng mới, đó là sử dụng CSS (Cascading Style Sheet – bảng định kiểu).

CSS thực chất là một tập các qui tắc để format/ định kiểu (style) cho các phần tử được hiển thị và định vị trên trang web. Nhờ vào CSS mà việc định dạng (kiểu) cho các phần tử trở nên dễ dàng và linh hoạt hơn rất nhiều.

Theo qui tắc định dạng của CSS thì các thuộc tính của một phần tử nào đó sẽ được thiết lập theo cách nhất quán, dạng: **Thuộc\_Tính: Giá\_Trị; Thuộc\_Tính:Giá\_Trị; .....**Danh sách đầy đủ các thuộc tính này có thể tra cứu dễ dàng trên Internet hoặc chính trình soạn thảo VS 2008 sẽ tự liệt kê trong khi chúng ta soạn code.

### 1.6.2 DHTML

Dynamic HTML (DHTML) là khả năng của các trang web có thể thay đổi nội dung hiển thị và định vị động của các phần tử.

Với các trang web tĩnh (Static web) thì khi nội dung trang web được hiển thị lên trên trình duyệt thì người dùng không có khả năng sửa đổi nội dung cũng như thay đổi vị trí của các phần tử HTML. Còn đối với những trang web có sử dụng JavaScript và CSS thì kể cả khi trang web đã hiển thị rồi thì vẫn có khả năng thay đổi nội dung (thêm, sửa, xóa, thay đổi định dạng, vị trí các phần tử...). Trang web như thế được gọi là trang web động (phía client). Chú ý rằng, trang web động này khác với trang web động (phía server) mà phần sau chúng ta sẽ đề cập ở các phần sau của tài liệu này.



## 1.7. Định dạng các thẻ sử dụng CSS

### 1.7.1 Định dạng ở mức dòng (Inline)

Định dạng ở mức dòng tức là việc định dạng các phần tử theo kiểu CSS ở ngay trong định nghĩa phần tử. Cú pháp chung như sau:

<Loại\_PT **Style** = “tt1:gt1; tt2:gt2; ...; tt<sub>n</sub>: gt<sub>n</sub>” ....> trong đó: tt = thuộc tính; gt = giá trị

Ví dụ: Định dạng cho textbox dưới đây có nền xanh, chữ trắng và viền đỏ.

```
<input style="border-color:Red; background-color:Blue; color:White" />
```

### 1.7.2 Định dạng bởi bộ chọn ID

Khi muốn cho một loạt các phần tử có cùng thuộc tính ID giống nhau được định dạng như sau thì người ta định nghĩa một bộ chọn ID. Cú pháp có dạng:

```
<style Type = “text/css”>
    #Tên {
        Tên_Thuộc_tính: Giá_Trị;
        Tên_Thuộc_tính: Giá_Trị;
        Tên_Thuộc_tính: Giá_Trị;
        .....
    }
</style>
```

*Ví dụ:*

- Định nghĩa bộ chọn tên là “Chuong” (Chương), có màu đỏ, cỡ chữ 20 và đậm.

```
#Chuong
{
    color:Red;
    font-size:20pt;
    font-weight:bold;
}
```

*- Áp dụng:*

< P id = “**Chuong**”> Đây là màu đỏ, cỡ chữ 20pt và **đậm** </P>

< H1 id = “**Chuong**”> Đây cũng là màu đỏ, cỡ chữ 20pt và **đậm** </H1>

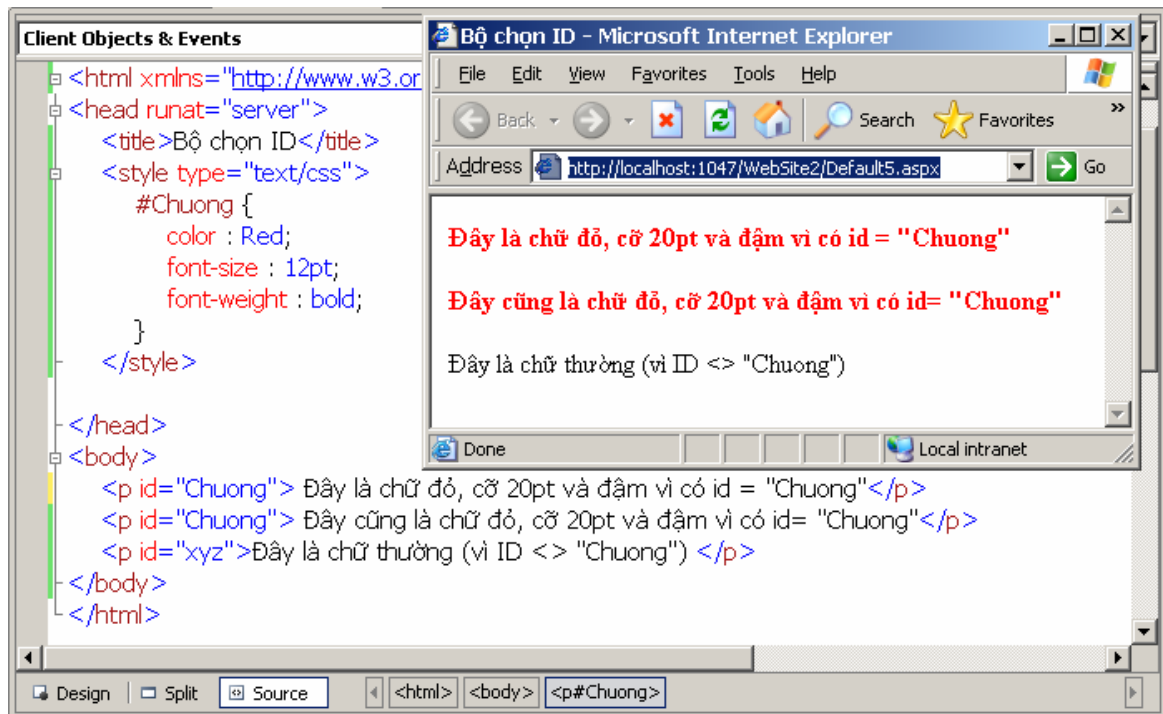
<H1 id=“xyz”> Đây thì không phải màu đỏ, vì có thuộc tính **ID** ≠ “**Chuong**”</H1>.

### 1.7.3 Định dạng bởi bộ chọn thẻ (tag)

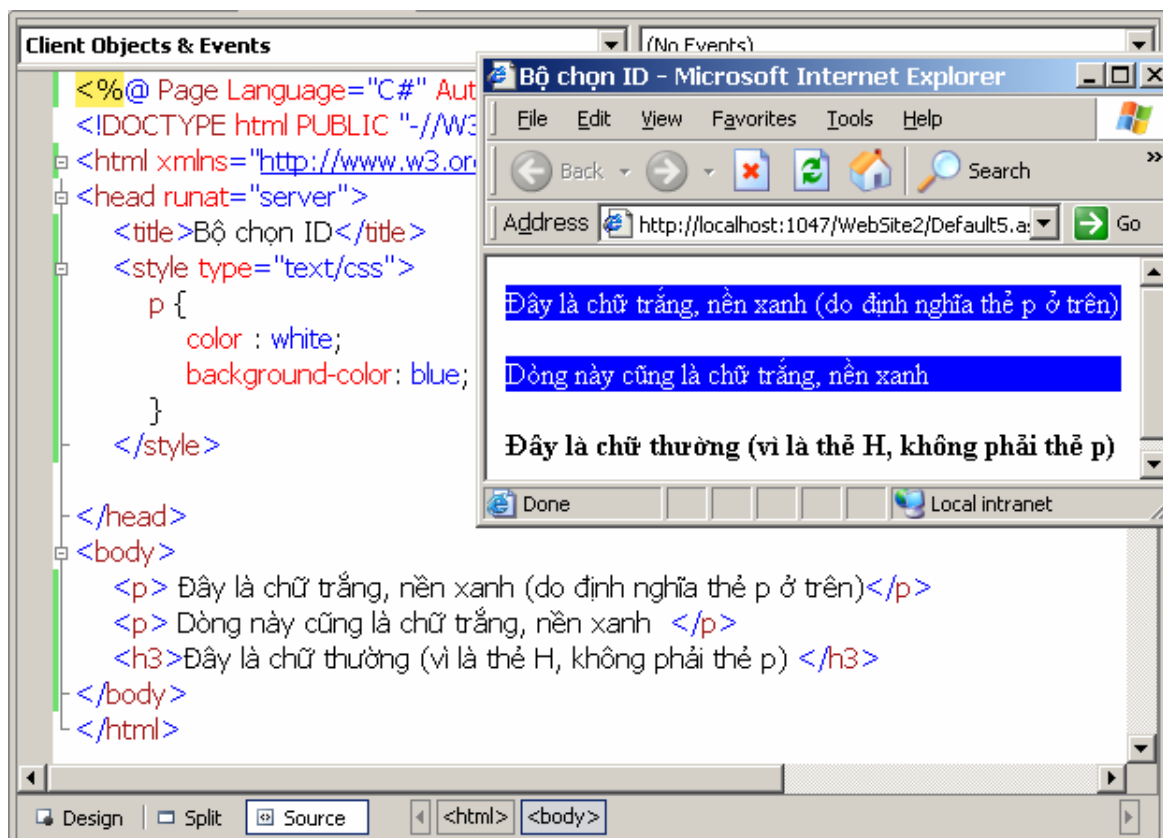
Khi muốn cho một loạt các phần tử cùng loại có định dạng giống nhau mà không cần ID giống nhau thì người định nghĩa CSS kiểu bộ chọn:

Cú pháp:

```
<style Type = “text/css”>
    Tên_Loại_Thẻ {
        Tên_Thuộc_tính: Giá_Trị;
        Tên_Thuộc_tính: Giá_Trị;
        Tên_Thuộc_tính: Giá_Trị;
        .....
    }
</style>
```



Ví dụ đầy đủ về Bộ chọn ID



Ví dụ về định nghĩa bộ chọn thẻ

#### 1.7.4 Định dạng bởi lớp (Class)

Còn một cách định nghĩa khác hay dùng nhất và linh hoạt nhất đó là cách định nghĩa lớp, ý tưởng chủ đạo là: Ta định nghĩa sẵn một lớp chứa các định dạng và khi muốn áp dụng định dạng đó cho phần tử nào thì chỉ việc gán lớp này cho phần tử.

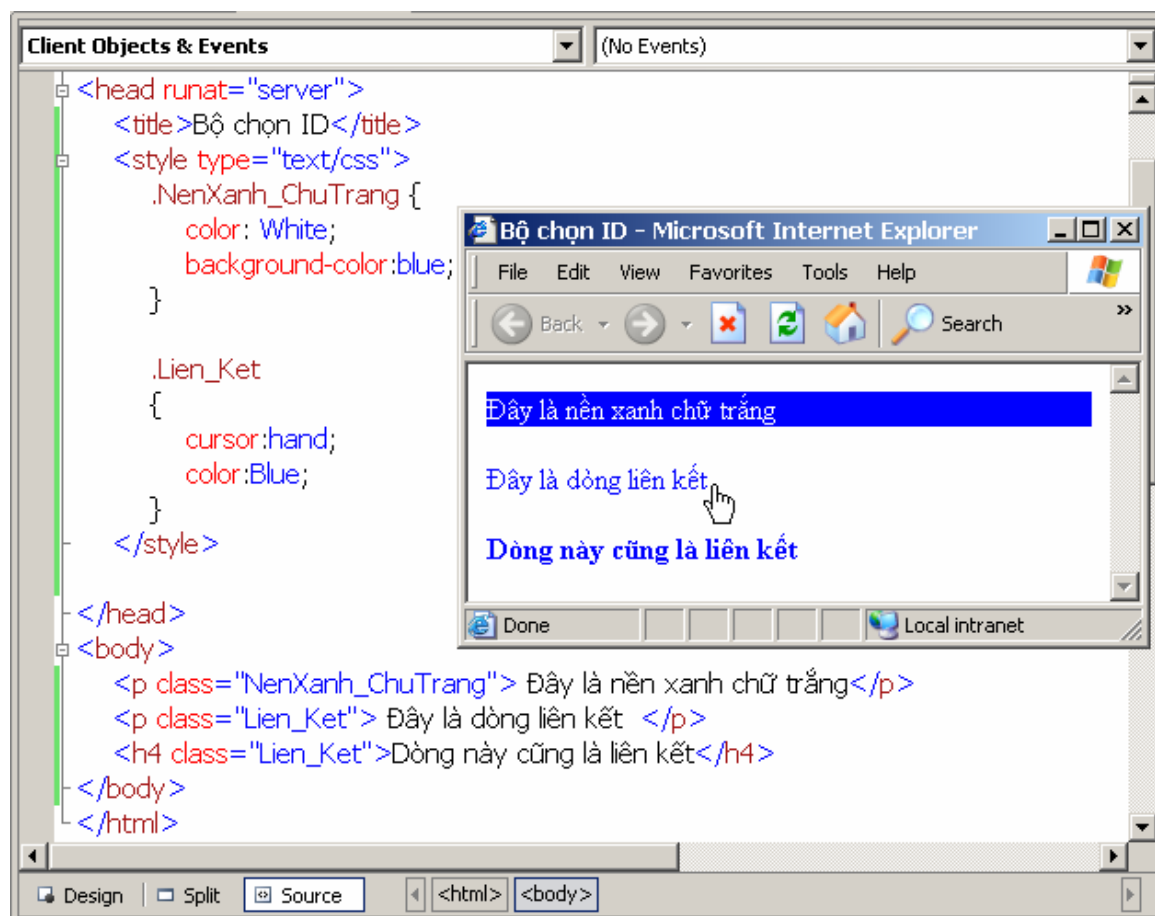
Cú pháp định nghĩa lớp như sau:

```
<style type="text/css">
  .<Tên_Lớp>
  {
    Tên_Thuộc_Tính: Giá_trị;
    Tên_Thuộc_Tính: Giá_trị;
    Tên_Thuộc_Tính: Giá_trị;
    .....
  }
</style>
```

**Ví dụ:** Định nghĩa 2 lớp là NenXanh\_ChuTrang và lớp Lien\_Ket.

```
<style type="text/css">
  .NenXanh_ChuTrang {
    color: White;
    background-color:blue;
  }
  .Lien_Ket
  {
    cursor:hand;
    color:Blue;
  }
</style>
```

**Ví dụ sử dụng:**

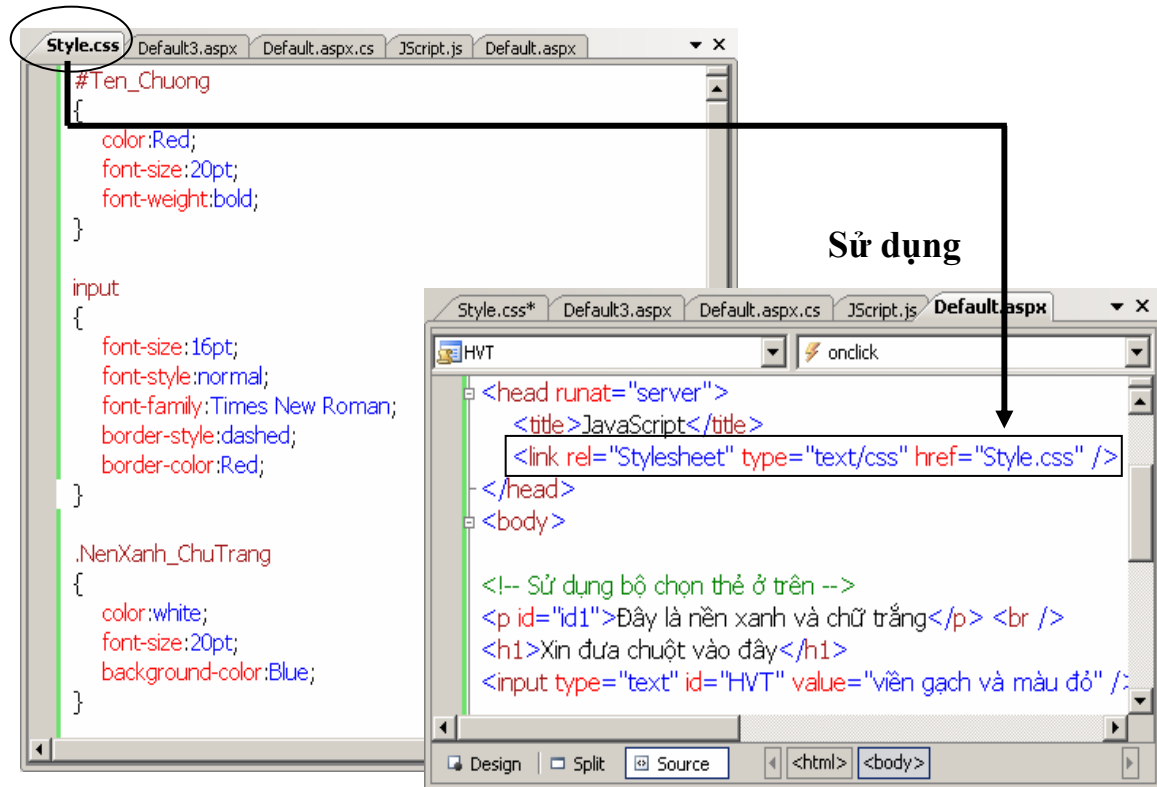


Ví dụ đầy đủ và kết quả.

### 1.7.5 Vấn đề tổ chức lưu trữ.

Các định nghĩa về CSS có thể được đặt ngay trong tệp nguồn nhưng cũng có thể được đặt riêng ra một tệp khác. Tệp này thường có đuôi mở rộng là style. Nội dung của tệp chỉ chứa các định nghĩa CSS (Gồm định nghĩa bộ chọn ID, bộ chọn thẻ và lớp).

Ví dụ về một tệp CSS và cách tham chiếu (sử dụng) tệp đó.



Nội dung tệp CSS và cách sử dụng tệp CSS trong file nguồn.

## 1.8. Truy xuất thuộc tính các thẻ HTML và CSS bằng JavaScript

### 1.8.1 Truy xuất các thuộc tính của thẻ

Nhìn chung, các trình duyệt đều tổ chức lưu trữ các đối tượng theo cấu trúc phân cấp, trong đó đối tượng window là đối tượng lớn nhất, nó bao gồm các đối tượng con là Location, history, screen, event.... Có thể thấy rõ hơn sự phân cấp này trong hình vẽ sau đây. Từ mô hình các đối tượng này, ta có thể dễ dàng biết cách truy xuất tới các phần tử mong muốn.

Một số cách khác dùng để truy xuất tới các phần tử trong trang web đó là sử dụng các phương thức document.getElementById("ID\_Của\_Phần\_Tử") (ID đặt trong cặp dấu ""), document.getElementsByName(Tên\_Phần\_tử) hay document.all.<ID của phần tử>

#### Ví dụ:

- Để truy xuất đến phần tử có ID="txtHoVaTen", có thể viết:

**document.getElementById("txtHoVaTen")** hoặc **document.all.txtHoVaTen**

- Để truy xuất đến thuộc tính value của phần tử có thuộc tính id = "txtHoVaTen", ta viết: **document.getElementById("txtHoVaTen").value** hoặc **document.all.txtHoVaTen.value**.

- Để lấy tất cả các phần tử có cùng giá trị name = "chkMatHang", ta viết:

document.getElementsByName("chkMatHang"), lệnh này sẽ trả về một mảng các phần tử có giá trị là chkMatHang.

- Để lấy tất cả các thẻ là input, ta viết:

document.getElementsByTagName("input"), lệnh này cũng trả về cho ta một mảng các phần tử.

\*\* Chú ý: Khi kết quả trả về là một mảng thì có thể duyệt bằng vòng lặp, ví dụ:

```
<html>
<body>
    <input type="text" value = "ASP.NET">
    <form id=form1 action="" method="post">
        <script language="javascript" type="text/javascript">

            var KetQua = document.getElementsByTagName("input");

            var i;
            for (int i=0; i<KetQua.length; i++)
            {
                alert("Giá trị của text box " + i + " là : " + KetQua[i].value);
            }
        </script>
    </form>
</body>
</html>
```

### 1.8.2 Truy xuất các thuộc tính CSS

Trong quá trình hoạt động của website, có thể có những lúc ta cần phải sửa đổi giá trị thuộc tính CSS nào đó của một phần tử, khi đó ta cần phải truy cập đến thuộc tính này.

Cú pháp truy cập như sau:

- ❖ window.<giá trị ID>.style.<thuộc\_Tính> hoặc
- ❖ <giá trị của thẻ>.style.<thuộc\_Tính> hoặc
- ❖ window.<Giá trị Name>.style.<thuộc\_Tính> hoặc
- ❖ <Giá trị Name của thẻ>.style.<thuộc\_Tính>

Ví dụ, có thể được đặt CSS như sau:

```
<html>
<body>

<input type="text" id="txtThongBao" name="txtTB"
    Style = "color:white; background-color:Blue"
    value = "Đây là một thông báo có chữ trắng và nền ...." />

<input type="button" value="Click here" onclick="ChangeColor()" />

    <script language="javascript" type="text/javascript">
    function ChangeColor()
    {
```

```
txtThongBao.style.color="yellow";  
// Hoặc txtTB.style.color="yellow";  
// Hoặc window.txtThongBao.style.color="yellow";  
// Hoặc window.txtTB.style.color="yellow";  
}  
</script>  
</body>  
</html>
```

\*\*\* Chú ý: Trong các ứng dụng web ngày nay, thuộc tính name ít được dùng và thuộc tính id được sử dụng phổ biến hơn. Vì vậy, để định danh cho các phần tử trong trang web, chúng ta nên sử dụng thuộc tính id thay vì name (trừ những ngoại lệ).

## BÀI SỐ 2: THỰC HÀNH

**Mục tiêu:** Kết thúc bài thực này, người học có thể

- Tạo và định dạng các thẻ HTML bằng CSS
- Truy xuất các đối tượng trình duyệt và các phần tử HTML bằng JavaScript.
- Tạo trang web đăng ký có xử lý tính hợp lệ của dữ liệu nhập vào.

**Nội dung:**

Định dạng các phần tử bằng CSS và sử dụng JavaScript để kiểm tra dữ liệu

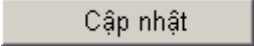
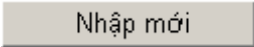
**Yêu cầu:**

- ❖ Tạo một trang web trong VS 2008 phục vụ việc nhập thông tin về cán bộ.
- ❖ Trang web này được tạo trên IIS Cục bộ.
- ❖ Sử dụng các style để định nghĩa cho các phần tử.
- ❖ Sử dụng JavaScript để kiểm tra tính hợp lệ của dữ liệu.

*Đặc tả giao diện, chức năng và các ràng buộc:*

### 1. Giao diện (Trang bên)

#### 2. Đặc tả xử lý

- Khi người dùng nhấn vào nút  thì thực hiện gửi toàn bộ nội dung đang nhập của trang hiện hành sang trang CapNhatCanBo.aspx.
- Khi người dùng nhấn vào nút  thì nội dung trong các ô nhập được reset trở về giá trị mặc định (như trong hình).

#### 3. Đặc tả ràng buộc

- Họ và tên không được rỗng và phải  $\leq 40$  ký tự.
- Ngày, tháng năm phải hợp lệ.
- Các trường đánh dấu \* là bắt buộc phải có.
- Các trường số (như ngày sinh, hệ số lương,...) phải là các số, không được là ký tự.
- Các hộp Textarea không được quá 1000 ký tự.
- Ở các hộp text, khi người dùng click chuột (focus) thì giá trị mặc định sẽ bị xóa để cho người dùng gõ giá trị mới. Nếu người dùng di chuyển sang phần tử khác mà không nhập giá trị nào thì đặt lại giá trị mặc định như ban đầu.
- Khi trường nào nhập sai thì sẽ đặt focus vào đúng trường sai đó.

#### 4. Một số kiến thức cần thiết và gợi ý:

- Định nghĩa style cho các mục giống nhau
- Đặt thuộc tính Action cho form để chuyển thông tin cho trang bất kỳ
- Dùng hàm isNaN(n) để kiểm tra xem n có phải là số hay không.
- Dùng phương thức focus của phần tử để đặt tiêu điểm.
- Tạo các phần tử kiểu submit và kiểu reset cho nút **Cập nhập** và **nhập mới**.
- Viết một hàm kiểm tra cho sự kiện Onclick của nút **Cập nhật**.
- Nếu không muốn cho một sự kiện nào đó (ví dụ onclick) kích hoạt thì viết trong sự kiện đó là “return false” hoặc return KQ; với KQ là một biểu thức, hàm có giá trị false

CHƯƠNG TRÌNH QUẢN LÝ CÁN BỘ VERSION 1.0

**NHẬP HỒ SƠ CÁN BỘ**

THÔNG TIN CÁ NHÂN	
*Họ và tên	<input type="text"/>
*Ngày sinh (ngày/tháng/năm)	--- / 1 / --- <input type="radio"/> Nam <input type="radio"/> Nữ
Chức vụ hiện tại (Đăng, chính quyền,...)	<input type="text"/>
*Quê quán	<input type="text"/>
*Nơi ở hiện nay	<input type="text"/>
TRÌNH ĐỘ HỌC VẤN	
Dân tộc :	<input type="text"/> Tôn giáo: <input type="text"/>
Thành phần gia đình:	<input type="text"/>
Nghề trước khi tuyển dụng	<input type="text"/>
Tham gia cách mạng:	Ngày <input type="text"/> Tổ chức <input type="text"/> Công tác <input type="text"/>
Ngày vào Đảng:	<input type="text"/> ngày vào chính thức <input type="text"/>
Ngày nhập ngũ:	<input type="text"/> ngày xuất ngũ <input type="text"/>
*Trình độ Văn hóa:	Học hàm: <input type="text"/> Học vị: <input type="text"/>
Lý luận chính trị	<input type="text"/>
Trình độ ngoại ngữ	Anh <input type="text"/> Nga <input type="text"/> Pháp <input type="text"/>
*Ngạch công chức, viên chức:	Mã số: <input type="text"/> *Hệ số lương: <input type="text"/>
Danh hiệu được phong (năm):	<input type="text"/>
Sở trường công tác:	<input type="text"/>
Khen thưởng (huân, huy chương cao nhất)	<input type="text"/>
Kỷ luật (đăng, chính quyền, năm, lý do, hình thức)	<input type="text"/>
ĐÀO TẠO, BỒI DƯỠNG CHUYÊN MÔN, NGHIỆP VỤ, LÝ LUẬN, NGOẠI NGỮ	
Ghi rõ Tên trường, ngành học, thời gian, loại hình, văn bằng chứng chỉ	<input type="text"/>
** Loại hình: Chính qui, tại chức, chuyên tu, bồi dưỡng, văn bằng: Tiến sĩ, thạc sĩ, cử nhân, kỹ sư.	
TÓM TẮT QUÁ TRÌNH CÔNG TÁC	
Ghi rõ thời gian bắt đầu và kết thúc; chức danh, chức vụ, đơn vị công tác tương ứng	<input type="text"/>
Đặc điểm lịch sử bản thân	
Đặc điểm lịch sử bản thân	<input type="text"/>
Quan hệ với người nước ngoài	<input type="text"/>
Quan hệ gia đình (Bố, mẹ, anh chị em ruột)	<input type="text"/>
Hoàn cảnh kinh tế gia đình	<input type="text"/>

Cập nhật

Nhập mới



## **Hướng dẫn:**

### **1. Định nghĩa style:**

Khi thiết kế giao diện cho trang web, trước hết cần xác định xem có những phần tử nào cùng một định dạng (style). Khi đó ta nên định ra một class chứa các định dạng mong muốn để áp dụng cho các phần tử cùng loại này.

#### **Lesson02.css**

```
.HeadTitle
{
    font-size: xx-large;
    font-weight: bold;
    text-align: center;
    color:Purple;
    margin-bottom:30px;
}

.CellSpace
{
    border-spacing:1px;
}

.Tiêu_Đề_Chính
{
    color:White;
    background-color:Purple;
    font-size:12pt;
    font-weight:bold;
    margin:5px 0px 5px 0px;
    height:25px;
}

.Cột1
{
    color:Gray;
    font-style:italic;
    text-align:right;
    width:30%;
}

.Cột2
{
    width:70%;
    text-align:left;
}

.TextboxDài
{
    width:99%;
    text-align:left;
}
```

```
.TTBatBuoc
{
    background-color:Yellow;
}

.Table
{
    table-layout:auto;
    border-style:solid;
    border-color:Purple;
    border-width:1px;
    border-collapse:collapse;
    background-color:White;
    width:800px;
}

td
{
    vertical-align:middle;
}

input
{
    margin:2px 0px 2px 2px;
}

input.NgayThang
{
    text-align:center;
    width:80px;
}

select
{
    text-align:center;
    width:100px;
}
```

## 2. Code trang giao diện

### NhapHSCB.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="NhapHSCB.aspx.cs"
Inherits="Lesson_02___LAB_YahooRegister" %>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head runat="server">
    <title>Nhập hồ sơ cán bộ</title>
    <link rel="Stylesheet" href="Lesson02.css" type="text/css" />
</head>
```

```
<body >
  <form id="form1" action="CapNhatCanBo.aspx" method="post" >
  <div style="text-align:center">
    <p style="border-bottom:solid; border-width:thin; font-size:20pt;
      margin:0; padding:0X; border-spacing:0px">
      CHƯƠNG TRÌNH QUẢN LÝ CÁN BỘ VERSION 1.0
    </p><br /> <br />

    <p class="HeadTitle">NHẬP HỒ SƠ CÁN BỘ</p>

    <table class="Table">
      <tr class="CellSpace">
        <td colspan="2" class="Tiêu_Đề_Chính">THÔNG TIN CÁ NHÂN</td>
      </tr>

      <tr>
        <td class="Cột1">*Họ và tên</td>
        <td class="Cột2"><input type="text" id="HoVaTen" class="TextboxDài" />
      </td>
      </tr>

      <tr>
        <td class="Cột1">*Ngày sinh (ngày/tháng/năm)</td>
        <td class="Cột2">
          <select id="NgaySinh">
            <option value="">1</option>
            <option value="2">2</option>
          </select> /
          <select id="cboThangSinh" >
            <option value="1">1</option>
            <option value="2">2</option>
            <option value="3">3</option>
            <option value="4">4</option>
            <option value="5">5</option>
            <option value="6">6</option>
            <option value="7">7</option>
            <option value="8">8</option>
            <option value="9">9</option>
            <option value="10">10</option>
            <option value="11">11</option>
            <option value="12">12</option>
          </select> /

          <select id="NamSinh">
            <option value="1950">1950</option>
            <option value="1951">1951</option>
            <option value="1952">1952</option>
          </select>

          Giới tính:
          <input type="radio" id="optNam" checked="checked" /> Nam
```

```
        <input type="radio" id="optNu"/>Nữ
    </td>
</tr>
<tr>
    <td class="Cột1">Chức vụ hiện tại (Đảng, chính quyền,...)</td>
    <td><input type="text" class="TextboxDài" /></td>
</tr>

<tr>
    <td class="Cột1">*Quê quán</td> <td class="Cột2">
        <input type="text" class="TextboxDài"/></td>
</tr>

<tr>
    <td class="Cột1">*Nơi ở hiện nay</td>
    <td class="Cột2"> <input type="text" class="TextboxDài"/></td>
</tr>

<tr>
    <td colspan="2" class="Tiêu_Đề_Chính">TRÌNH ĐỘ HỌC VẤN</td>
</tr>

<tr>
    <td class="Cột1">Dân tộc : </td>
    <td class="Cột2"><input type="text" /> Tôn giáo: <input type="text" />
</td>
</tr>
<tr>
    <td class="Cột1">Thành phần gia đình:</td>
    <td class="Cột2"> <input type="text" class="TextboxDài"/></td>
</tr>
<tr>
    <td class="Cột1">Nghề trước khi tuyển dụng</td>
    <td class="Cột2"><input type="text" class="TextboxDài" /></td>
</tr>

<tr>
    <td class="Cột1">Tham gia cách mạng: </td>
    <td class="Cột2">
        Ngày <input value="..../...../....." style="width:15%; text-align:center"
        onfocus="XuLyFocus(this);" onblur="XuLyLostFocus(this);" />

        Tổ chức <input style="width:20%" />
        Công tác <input style="width:20%" />
    </td>
</tr>

<tr>
    <td class="Cột1">Ngày vào Đảng: </td>
    <td class="Cột2"><input type="text" value="...../...../....." id="NgayVaoDang"
class="NgayThang"
        onfocus="XuLyFocus(this);" onblur="XuLyLostFocus(this);"/>
</td>
</tr>
```

```
        ngày vào chính thức <input type="text" class="NgàyThang"
        value="...../...../....."
        onfocus="XuLyFocus(this);" onblur="XuLyLostFocus(this);"/>
    </td>
</tr>

<tr>
    <td class="Cột1">Ngày nhập ngũ:</td>
    <td class="Cột2"><input type="text" value="..../...../...." class="NgàyThang"
        onfocus="XuLyFocus(this);" onblur="XuLyLostFocus(this);"/>
        ngày xuất ngũ <input type="text" class="NgàyThang" value="..../...../....."
        onfocus="XuLyFocus(this);" onblur="XuLyLostFocus(this);"/>
    </td>
</tr>

<tr>
    <td class="Cột1">*Trình độ Văn hóa: </td>
    <td class="Cột2"><input style="width:15%" />
        Học hàm:
        <select>
            <option value="">-----</option>
            <option value="Thạc Sĩ">Thạc sĩ </option>
            <option value="Tiến Sĩ">Tiến sĩ</option>
        </select>
        Học vị :
        <select>
            <option value="">-----</option>
            <option value="Giáo sư">Giáo sư</option>
            <option value="Phó giáo sư">Phó giáo sư</option>
        </select>
    </td>
</tr>

<tr>
    <td class="Cột1">Lý luận chính trị </td>
    <td class="Cột2">
        <select> <option>-----</option>
            <option value = "Sơ cấp">Sơ cấp</option>
            <option value="Trung cấp">Trung cấp</option>
            <option value="Cao cấp">Cao cấp</option>
            <option value="Cử nhân">Cử nhân</option>
        </select>
    </td>
</tr>

<tr>
    <td class="Cột1">Trình độ ngoại ngữ</td>
    <td class="Cột2">
        Anh <select><option>-----</option>
            <option value="A">A</option>
            <option value="B">B</option>
            <option value="C">C</option>
        </select>
    </td>
</tr>
```

```

        </select>
        Nga <select>
            <option>-----</option>
            <option value="A">A</option>
            <option value="B">B</option>
            <option value="C">C</option>
        </select>
        Pháp <select>
            <option>-----</option>
            <option value="A">A</option>
            <option value="B">B</option>
            <option value="C">C</option>
        </select>
    </td>
</tr>

<tr>
    <td class="Cột1">*Ngạch công chức, viên chức:</td>
    <td class="Cột2">
        <input style="width:20%" />
        Mã số: <input style="width:15%" />
        *Hệ số lương: <input style="width:15%" />
    </td>
</tr>

<tr>
    <td class="Cột1">Danh hiệu được phong (năm): </td>
    <td class="Cột2"><input class="TextboxDài" /></td>
</tr>

<tr>
    <td class="Cột1">Sở trường công tác:</td>
    <td class="Cột2"><input class="TextboxDài" /></td>
</tr>

<tr>
    <td class="Cột1">Khen thưởng (huân,huy chương cao nhất)</td>
    <td class="Cột2"><input class="TextboxDài" /></td>
</tr>

<tr>
    <td class="Cột1">Kỷ luật (đảng, chính quyền, năm, lý do, hình thức)</td>
    <td class="Cột2">
        <textarea class="TextboxDài" cols="50" rows="3"></textarea>
    </td>
</tr>

<tr class="Tiêu_Đề_Chính">
    <td colspan="2">
        ĐÀO TẠO, BỒI DƯỠNG CHUYÊN MÔN, NGHIỆP VỤ, LÝ LUẬN, NGOẠI NGỮ
    </td>
</tr>
</tr>
```

```
<tr>
  <td class="Cột1">Ghi rõ Tên trường, ngành học, thời gian, loại hình, văn bằng,
    chứng chỉ</td>
  <td><textarea class="TextboxDài" cols="100" rows="5"></textarea>
</td>
</tr>
<tr>
  <td colspan="2" style="color:Blue">
    ** Loại hình: Chính qui, tại chức, chuyên tu, bồi dưỡng; văn bằng: Tiến sĩ,
    thạc sĩ, cử nhân, kỹ sư.
  </td>
</tr>

<tr class="Tiêu_Đề_Chính">
  <td colspan="2">TÓM TẮT QUÁ TRÌNH CÔNG TÁC</td>
</tr>
<tr>
  <td class="Cột1">Ghi rõ thời gian bắt đầu và kết thúc; chức danh, chức vụ,
    đơn vị công tác tương ứng</td>
  <td><textarea class="TextboxDài" cols="100" rows="5"></textarea></td>
</tr>

<tr class="Tiêu_Đề_Chính">
  <td colspan="2">Đặc điểm lịch sử bản thân</td>
</tr>

<tr>
  <td class="Cột1">Đặc điểm lịch sử bản thân</td>
  <td class="Cột2">
    <textarea class="TextboxDài" cols="100" rows="5"></textarea>
  </td>
</tr>

<tr>
  <td class="Cột1">Quan hệ với người nước ngoài</td>
  <td class="Cột2">
    <textarea class="TextboxDài" cols="100" rows="5"></textarea>
  </td>
</tr>

<tr>
  <td class="Cột1">Quan hệ gia đình (Bố, mẹ, anh chị em ruột)</td>
  <td class="Cột2">
    <textarea class="TextboxDài" cols="100" rows="5"></textarea>
  </td>
</tr>

<tr>
  <td class="Cột1">Hoàn cảnh kinh tế gia đình</td>
  <td class="Cột2">
    <textarea class="TextboxDài" cols="100" rows="5"></textarea>
  </td>
</tr>
```

```
        </td>
    </tr>
</table>
<br />
<table class="Table" style="border:0">
    <tr>
        <td style="text-align:right"><input type="submit" value="    Cập nhật    "
            onclick="return KiemTra();" /></td>
        <td style="text-align:left"><input type="reset" value=" Nhập mới " /></td>
    </tr>
</table>
</div>
</form>

<script language="javascript" type="text/javascript">
    var Giá_Trị_Cũ;

    /// Hàm xử lý khi người dùng bấm vào nút Nhập
    function KiemTra()
    {
        if (form1.HoVaTen.value.length==0)
        {
            alert("Họ tên phải khác rỗng !");
            form1.HoVaTen.focus();
            return false;
        }

        if( isNaN(form1.NgaySinh.value)==false)
        {
            alert("Ngày sinh phải là số ");
            form1.NgaySinh.focus();
            return false;
        }
        return true;
    }

    /// Hàm xử lý khi phần tử nhận được focus
    function XuLyFocus(txt)
    {
        Giá_Trị_Cũ=txt.value;
        txt.value="";
    }

    /// Hàm xử lý khi phần tử mất focus
    function XuLyLostFocus(txt)
    {
        if (txt.value=="") txt.value=Giá_Trị_Cũ;
    }
</script>
</body>
</html>
```



## BÀI SỐ 3: ASP.NET và Web form

### 3.1 Mô hình lập trình phía máy chủ

Trong thế giới web, tất cả các giao tiếp giữa Client (trình duyệt) và Server (web server) đều được thực hiện theo cơ chế “**Request and Response**”. Tức là, trước tiên phía máy khách cần phải “request” (gửi yêu cầu) tới Server, sau đó phía server sẽ “response” (hồi đáp) lại yêu cầu.

Cùng một cơ chế này, người ta có 2 cách tiếp cận để xử lý “request trang web” từ máy khách:

**Cách 1:** Khi máy khách yêu cầu một trang – ví dụ trang **abc.** – thì máy chủ sẽ đọc toàn bộ nội dung của trang và gửi về cho phía máy khách mà không thực hiện bất kỳ xử lý nào. Nó hoàn toàn không qua tâm đến ý nghĩa bên trong của trang **abc.** Nội dung trang này sau đó sẽ được phía trình duyệt xử lý.

**Cách 2:** Khi máy khách yêu cầu một trang – ví dụ trang **xyz.** – thì máy chủ sẽ đọc toàn bộ nội dung của trang đó và **xử lý tại Server** (trước khi gửi về cho client) **để được kết quả**, tiếp theo lấy kết quả xử lý được gửi về cho phía máy khách. Kết quả trả về cho máy khách có thể chứa các phần tử HTML, các câu lệnh JavaScript, các định nghĩa kiểu CSS....và tiếp tục được phía client (trình duyệt) xử lý như cách 1.

Với cách 1, do việc xử lý không diễn ra bên phía server nên trang web không thể đọc/ ghi các dữ liệu trên Server được (ví dụ Danh sách khách hàng, danh mục sản phẩm,...). Vì vậy nó chỉ phù hợp với các trang web đơn giản, không đòi hỏi xử lý chi tiết.

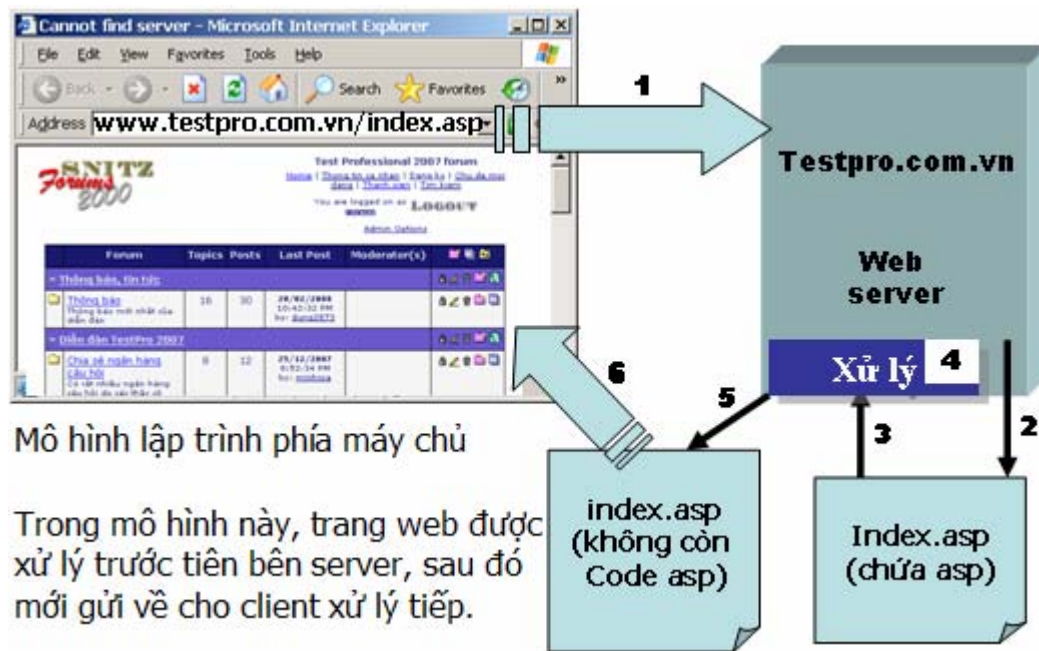
Với cách 2, do việc xử lý thông tin ở tại server nên hoàn toàn có thể đọc/ ghi dữ liệu trên chính server đó. Vì vậy, nó phù hợp với các dự án lớn và tính bảo mật cao. Mô hình theo cách này gọi là mô hình lập trình phía máy chủ.

Dưới đây là hình ảnh minh họa cho 2 mô hình này:

#### ❖ Mô hình lập trình phía máy khách (Client side)



❖ Mô hình lập trình phía máy chủ



**Câu hỏi:** Khi nào thì một trang sẽ được xử lý ở bên Server trước ? hay nói cách khác là khi nào thì được gọi là xử lý theo mô hình phía server?

**Trả lời:** Các trang (file) có đuôi mở rộng mà server có thể xử lý, ví dụ: asp, php, jsp, aspx...

**Câu hỏi:** Có thể lấy một ví dụ về một trang sẽ được xử lý phía server và trang sẽ không được xử lý phía server ?

Trang Trang1.htm Trang2.aspx

```
<html>
<body>
    Hello world
</body>
</html>
```

2 dòng này  
sẽ được xử lý  
bên phía  
server trước

```
<%@ Page Language="C#" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
        <div>

<% Response.Write (DateTime.Today.Date.ToString()); %>

        </div>
    </form>
</body>
</html>
```

**Câu hỏi:** Chương trình Client và server có nhất thiết phải nằm trên hai máy tính riêng biệt không ? và Client là các trình duyệt rồi (IE, FireFox...), còn server là chương trình nào ?

**Trả lời:** Hai chương trình này hoàn toàn có thể nằm trên cùng một máy tính. Chương trình server thực chất là một chương trình có tên là IIS (Internet Information Service).

Câu hỏi: Phải viết như thế nào để server hiểu là cần phải xử lý bên phía server trước khi gửi về cho phía Client ?

Trả lời: Trước tiên phải đặt phần mở rộng cho file (ví dụ .aspx), sau đó trong trình duyệt cần phải đặt những nội dung muốn xử lý bên phía server trong cặp thẻ đặc biệt, ví dụ:

```
<% Response.Write (DateTime.Today.Date.ToString ()); %>
```

*Hoặc:*

```
<form id="form1" runat="server">  
    <asp:Calendar runat="server" ID="Lịch"> </asp:Calendar>  
</form>
```

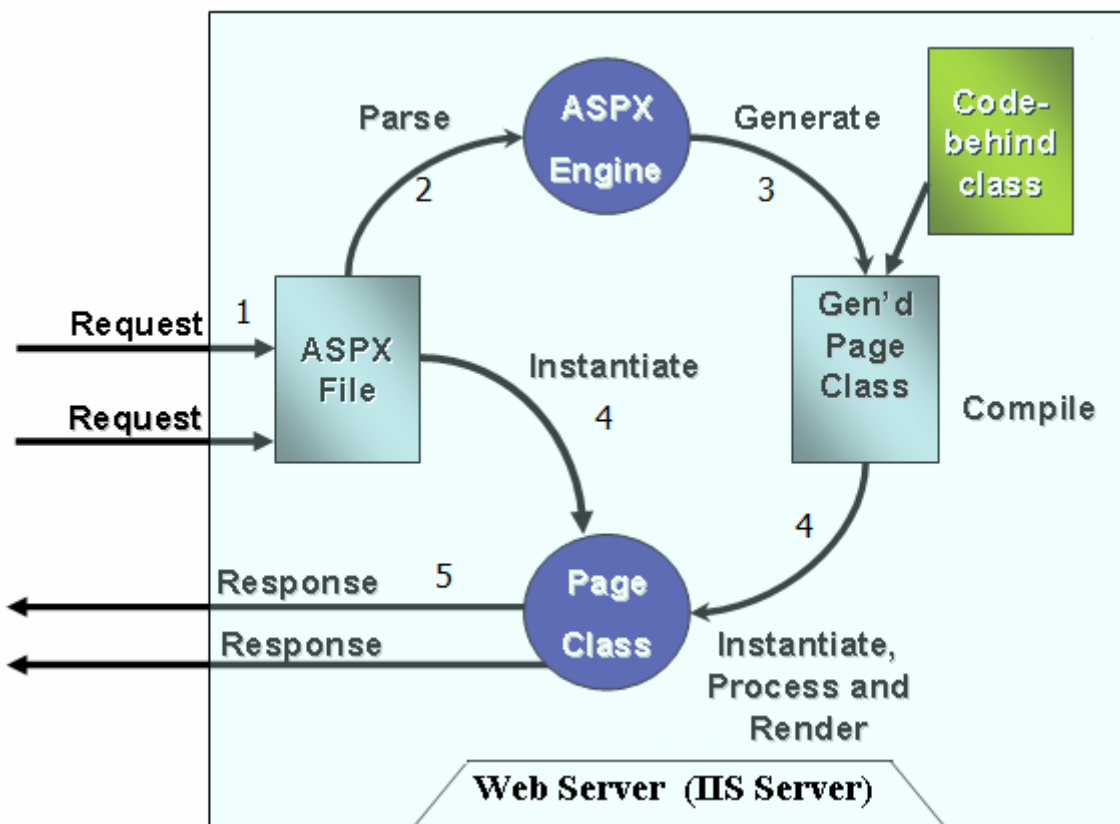
\*\*\* Chính các ký hiệu <% %> và **Runat = "Server"** đã “mách bảo” Server là : “Hãy xử lý nội dung đó bên phía server đi”!. Nếu không có những ký hiệu này thì mặc nhiên server làm mỗi việc là gửi trả lại cho trình duyệt xử lý.

Câu hỏi: Sao không gửi ngay cho trình duyệt xử lý như trước đây mà cứ phải để server xử lý ...!. Để Client xử lý sẽ giảm tải cho server, điều này chẳng tốt hơn sao ?

Trả lời: Vì trình duyệt chỉ có thể hiểu và xử lý được các thẻ HTML và Javascript thôi, còn nó không thể xử lý được các nội dung phức tạp. Ví dụ nó không hiểu **asp:Calendar** là gì ?

### 3.2 Cơ chế xử lý file ASP.NET phía máy chủ.

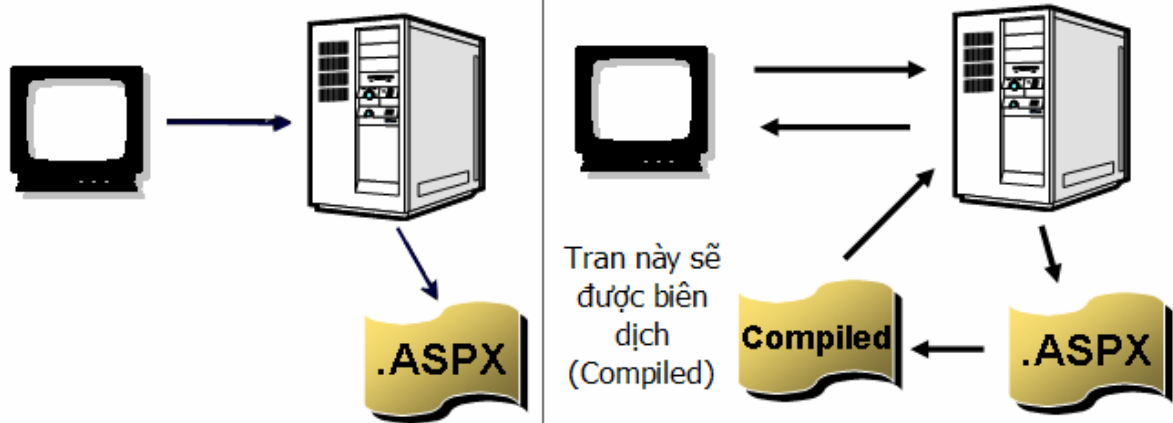
Đối với các trang ASP.NET, thì cơ chế xử lý giống như đã mô tả ở trên, tức là theo mô hình xử lý bên phía server. Nhưng có bổ sung thêm tính năng Compile and Cache:



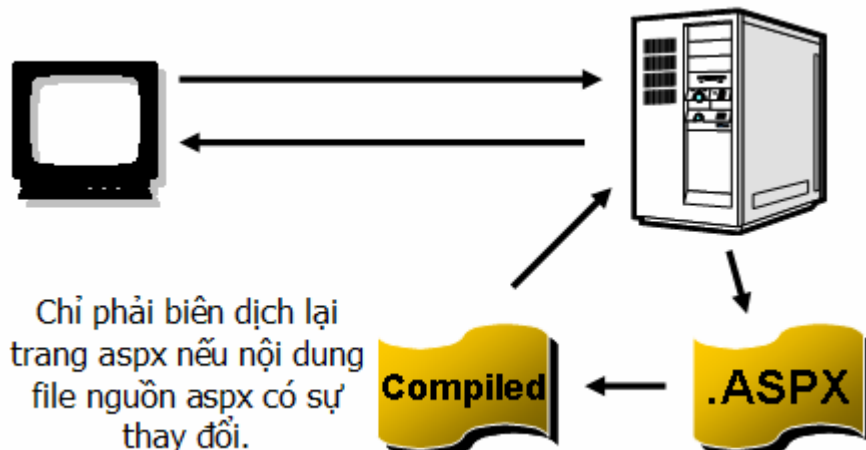
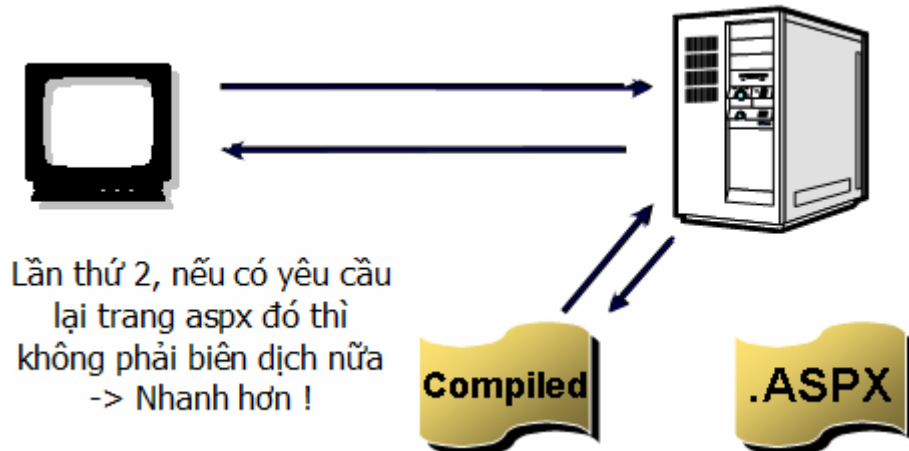
Giải thích cơ chế xử lý ở trên:

- Bước 0: Người lập trình phải tạo các trang ASPX (giả sử tên trang đó là **abc.aspx**) và đặt nó vào trong thư mục web của web server (có tên là www.server.com). Trên thanh địa chỉ của trình duyệt, người dùng nhập trang www.server.com/abc.aspx.
- Bước 2: Trình duyệt gửi yêu cầu tới server với nội dung: **"Làm ơn gửi cho tôi trang abc.aspx thì tốt !"**.
- Bước 3: web server sẽ biên dịch code của trang aspx (bao gồm cả các mã code vb.net/ c# - gọi là code behind hay code file) thành class.
- Bước 4: Lớp sau khi được biên dịch sẽ thực thi.
- Bước 5: trả kết quả về cho trình duyệt

Riêng với ASP.NET thì việc biên dịch sẽ được thực hiện "thông minh hơn", như sau:



**Lần đầu tiên trang aspx được yêu cầu.**



### 3.3 Một số ví dụ minh họa.

#### 3.3.1 Yêu cầu xử lý tại phía server thông qua Runat="Server"

The screenshot displays the Visual Studio IDE with the 'Source' view selected. The code editor shows the following ASP.NET markup:

```
<%@ Page Language="C#" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Server side - example 1</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:Calendar runat="server" ID="Lịch"> </asp:Calendar>
  </form>
</body>
</html>
```

Below the code editor, there is a blue box with the following text:

**Dòng :**  
`<asp:Calendar runat="server" ID="Lịch"></asp:Calendar>`  
 Sẽ được xử lý bên phía server vì bên trong nó có ghi **Runat = "Server"**.  
 Vì Server rất hiểu câu lệnh này nên nó sẽ sinh ra các thẻ HTML và các câu lệnh JavaScript gửi trả về cho trình duyệt để tạo thành một Calendar như hình bên cạnh --->

To the right of the code editor, a preview window titled 'Server side - example 1 - Microsoft...' shows the rendered output. It features a calendar for July 2008. The calendar is a table with days of the week as headers and dates as rows. The date 18 is highlighted.

Sun	Mon	Tue	Wed	Thu	Fri	Sat
29	30	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2
3	4	5	6	7	8	9

### 3.3.2 Yêu cầu xử lý bên phía server thông qua cặp thẻ <% %>

The screenshot shows the Visual Studio IDE with the 'Client Objects & Events' window open. The code in the editor is as follows:

```
<%@ Page Language="C#" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title>Server side - example 2</title>
</head>
<body>
<form id="form1" runat="server">
<% int i;
for (i = 1; i <= 10; i++)
Response.Write (i + "<br>");
%>
</form>
</body>
</html>
```

An arrow points from the code block to a preview window titled 'Server side - example 2 - Microsoft...'. The preview window displays the output of the code, which is a list of numbers from 1 to 10, each on a new line.

Below the code, a text box contains the following text:

Khởi lệnh ở trên sẽ được phía server xử lý trước, sau đó mới lấy kết quả xử lý được gửi trả lại cho phía Client (trình duyệt):

**Câu hỏi: Kết quả trả về cho client trong trường hợp này cụ thể là gì ???**

Ngoài 2 cách trên, còn 2 cách để yêu cầu xử lý trang web trực tiếp trên server, đó là: Đặt các câu lệnh ngay trong cặp thẻ Script, nhưng có thuộc tính Runat = "Server":

```
.....
<script language="C#" type="text/C#" runat="server">
    /// <summary>
    /// Các câu lệnh/ khai báo biến/ khai báo hàm/ định nghĩa lớp v.v...
    /// cần xử lý bên phía server thì đặt vào đây ! Ví dụ:
    /// </summary>
    string HoVaTen = "Aptech Center";
    public int Tong (int a, int b)
    {
        return a + b;
    }

    // Hoặc định nghĩa lớp
    public class Example
    {
        public int Tich (int a, int b)
        {
            return a * b;
        }
    }
</script>
.....
```



### 3.3.3 Yêu cầu xử lý bên server thông qua Script

The screenshot shows the 'Client Objects & Events' window in Visual Studio. The code is for a web page titled 'Server side - example 3'. It includes a C# script block with a method 'Tong' that calculates the sum of two text box values. A callout bubble points to the script block with the text: 'Một ví dụ đầy đủ hơn, trong đó có chứa cả phần tử server và mã lệnh'. Below the code, a preview window shows the rendered page with two text boxes containing '100' and '50', a 'Tinh' button, and a result box containing '150'.

```
<%@ Page Language="C#" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title>Server side - example 3</title>
</head>
<body>
<form id="form1" runat="server">
<asp:TextBox id="txtA" runat="server" width="50px" />
<asp:TextBox id="txtB" runat="server" width="50px"/>
<asp:Button id="cmdTinhTong" Text="Tinh" runat="server" OnClick="Tong" />
<asp:TextBox id="txtKetQua" runat="server" width="50px"/>

<script language="c#" type="text/C#" runat="server" width="20px">
public void Tong (object sender, EventArgs e)
{
    txtKetQua.Text = (int.Parse (txtA.Text) + int.Parse (txtB.Text)).ToString();
}
</script>
</form>
</body>
</html>
```

### 3.3.4 Yêu cầu xử lý bên phía server bằng cách đặt trong Code file

The screenshot shows the 'Client Objects & Events' window in Visual Studio. The code is for a web page titled 'Lesson\_03\_Default'. It includes a C# script block with a method 'Tong' that calculates the sum of two text box values. A callout bubble points to the script block with the text: 'Codefile : default.aspx' and 'Class: Lesson\_03\_Default'. Below the code, a preview window shows the rendered page with two text boxes containing '100' and '50', a 'Tinh' button, and a result box containing '150'.

```
<%@ Page Language="C#" CodeFile="~/Lesson_03/Default.aspx.cs"
Inherits="Lesson_03_Default"%>

using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;

public partial class Lesson_03_Default : System.Web.UI.Page
{
    protected void Tong (object sender, EventArgs e)
    {
        txtKetQua.Text = (int.Parse (txtA.Text) + int.Parse (txtB.Text)).ToString ();
    }
}
```

### 3.4 Webform trong ASP.NET

Để xây dựng ứng dụng web, ASP.NET cung cấp sẵn cho các nhà lập trình rất nhiều lớp ngay khi cài đặt .NET framework. Trong số này có một lớp đặc biệt quan trọng là **Page**. Mỗi lớp Page sẽ trình bày một trang tài liệu – tương ứng với một window – và được gọi là một web form.

Web form là một công nghệ cho phép xây dựng các trang web trong đó có thể lập trình được. Các trang này gọi là ASP.NET web form pages hay ngắn gọn là web form.

Các trang web xây dựng bằng ASP.NET sẽ không phụ thuộc vào trình duyệt (tức là trình duyệt nào cũng cho kết quả như nhau và hiển thị giống nhau).

Một số ưu điểm của web forms:

- ❖ Web forms có thể được thiết kế và lập trình thông qua các công cụ phát triển ứng dụng nhanh (RAD).
- ❖ Web form hỗ trợ một tập các điều khiển (controls) có thể mở rộng.
- ❖ Bất kỳ một ngôn ngữ .NET nào cũng có thể được dùng để lập trình với web forms.
- ❖ Asp sử dụng trình thực thi ngôn ngữ chung (CLR) của .NET framework do đó thừa hưởng mọi ưu thế của .NET Framework. Ví dụ : Khả năng thừa kế.

### 3.5 Tìm hiểu cấu trúc trang ASP.NET

Một trang ASP.NET bao gồm cả phần giao diện người dùng và phần xử lý logic bên trong. Giao diện người dùng chịu trách nhiệm hiển thị các thông tin và tiếp nhận dữ liệu từ người dùng, trong khi đó phần xử lý (lập trình) đảm nhiệm việc điều khiển sự tương tác của người dùng với trang web. Phần giao diện người dùng bao gồm một file chứa ngôn ngữ đánh dấu – như HTML hoặc XML và server controls chẳng hạn. File này được gọi là một **Trang (Page)** và có đuôi mở rộng là **aspx**.

Phản đáp ứng các tương tác của người dùng với trang web được thực hiện bởi một ngôn ngữ lập trình chẳng hạn như Visual Basic.NET và C#. Chúng ta có thể thực hiện việc viết code bằng bất kỳ ngôn ngữ lập trình nào được hỗ trợ bởi CLR ở ngay trong trang ASPX hoặc tách ra một file riêng. File tách riêng này được gọi là file Code Behind hay mới đây gọi là Code file. Đuôi mở rộng của Code file là **.VB** (Nếu dùng ngôn ngữ Visual Basic) hoặc **.CS** (nếu dùng ngôn ngữ C#).


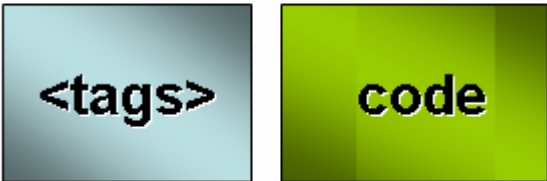
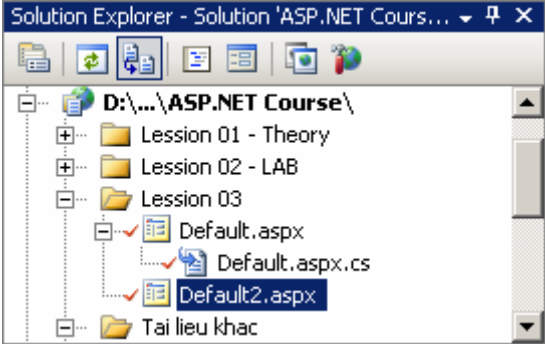
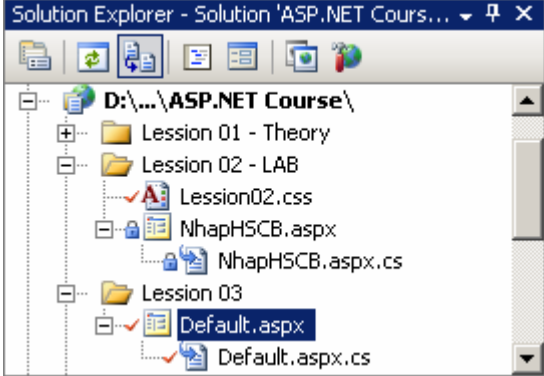


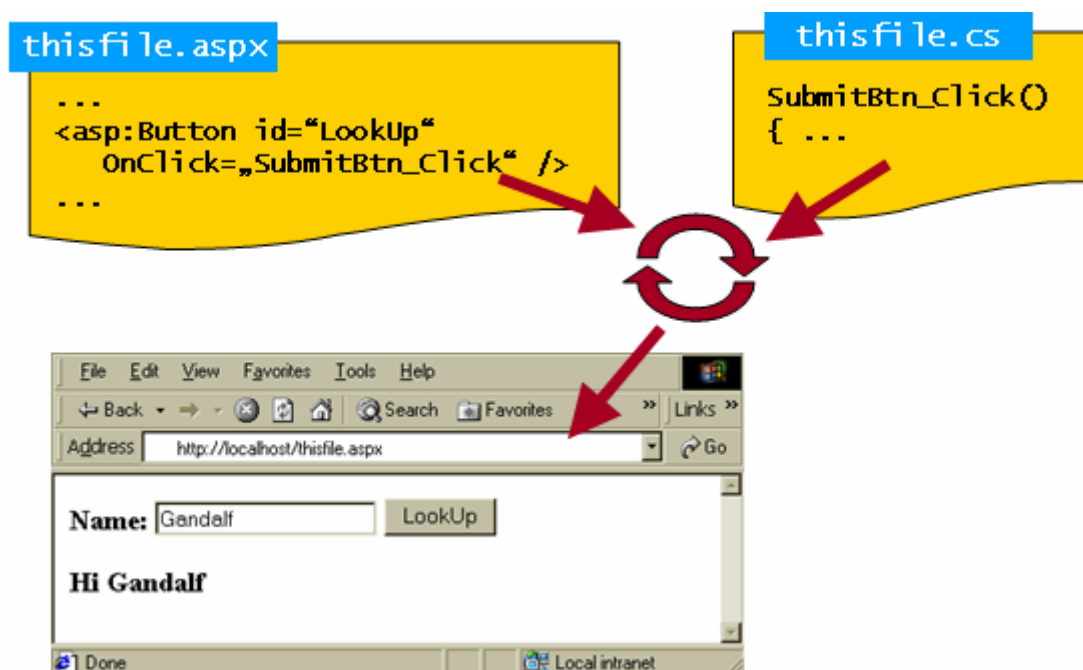
Cách lưu trữ này được minh họa qua một ứng dụng cụ thể dưới đây.

Trong đó, trang web thứ nhất **Default2.aspx** chứa cả code (C#) và giao diện (HTML) còn trang web thứ hai đặt code và giao diện ra 2 file riêng biệt. default.aspx và default.cs.

\*\*\* Chú ý: Có thể kết hợp để vừa đặt code trong file **aspx** vừa đặt code trong file **cs**.



Trong 1 file	Đặt riêng (“code-behind”)
	
	
Trang <b>Default2.aspx</b> chứa code ở bên trong nó.	Code được đặt trong <b>default.cs</b> Còn phần giao diện chứa trong <b>default.aspx</b>



Một webform bao gồm 2 thành phần:

- Thành phần giao diện (trang *thisfile.aspx*)
- Thành phần xử lý (lập trình) *thisfile.cs*

- Phân tích một trang ASP.NET thực tế (Trang này lưu code và giao diện trên 2 file):

**File Default.aspx**

```
<%@ Page Language="C#" CodeFile="~/Lesson 03/Default.aspx.cs"
Inherits = "Lesson03_default"%>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Server side - example 3</title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:TextBox id="txtA" runat="server" width="50px" />
        <asp:TextBox id="txtB" runat="server" width="50px"/>
        <asp:Button id="cmdTinhTong" Text="Tính" runat="server" OnClick="Tong" />
        <asp:TextBox id="txtKetQua" runat="server" width="50px"/>
    </form>
</body>
</html>
```

Nội dung file code (default.cs) như sau:

**File Default.cs**

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;

public partial class Lesson03_default : System.Web.UI.Page
{
    protected void Tong (object sender, EventArgs e)
    {
        txtKetQua.Text = (int.Parse (txtA.Text) + int.Parse (txtB.Text)).ToString ();
    }
}
```

**Trong file default.aspx:**

- **Page Language="C#"** : chỉ ra rằng ngôn ngữ được sử dụng để lập trình là C#
- **CodeFile="~/Lesson 03/Default.aspx.cs"**: Cho biết nội dung file chứa code xử lý là file ~/Lesson 03/Default.aspx.cs.
- **Inherits = "Lesson03\_default"**: Cho biết là trang giao diện thừa kế từ lớp nào trong file ~/Lesson 03/Default.aspx.cs (Bởi vì một file có thể có chứa nhiều lớp).
- **<head runat="server">**  
    <title>Server side - example 3</title>

</head>

Cho biết là thẻ này cần được xử lý bên phía server. Tuy nhiên nội dung trong thẻ này không có gì đặc biệt để xử lý và kết quả sau xử lý sẽ là (không có `runat="server"`):

<head>

<title>Server side - example 3</title>

</head>

- `<form id="form1" runat="server">` : Cho biết là nội dung trong cặp thẻ form cần được xử lý bên phía server.
- `<asp:TextBox id="txtA" runat="server" width="50px"/>` : là thẻ tạo ra phần tử textbox, tuy nhiên do có thuộc tính `runat = "server"` nên việc tạo này sẽ được thực hiện ở bên server, được kết quả trả về (là `<input type="TextBox" id="txtA" style = "width:50px">`)

<script language="C#" type="text/C#" runat="server">

```
public int Hieu (int a, int b)
{ return a - b; }
```

</script>

Đoạn script này có thuộc tính `runat="Server"`, vì vậy nó sẽ được xử lý phía server. Thuộc tính `language = "C#"` cho biết ngôn ngữ sử dụng để viết là C Sharp.

### **Trong file default.cs**

Nội dung file này hoàn toàn chứa các câu lệnh của ngôn ngữ lập trình VB.NET hoặc C#. Việc viết code cho file đó hoàn toàn giống như viết các chương trình trên window form hay chương trình Console.

Chú ý: Trong file này không được chứa trực tiếp các thẻ HTML.

Các câu lệnh trong file này HOÀN TOÀN ĐƯỢC PHÉP TRUY CẬP TỚI CÁC PHẦN TỬ ở trong file `default.aspx` có thuộc tính `runat = "server"`.

Câu hỏi: Nếu trong file `default.cs` có dòng lệnh sau:

**`cmdTinhTong.Text = "Tính tổng";`**

thì chương trình có báo lỗi không ? Vì sao ?

### **3.6 Code behind và viết code phía Server.**

Các file chứa mã code (VB.NET hoặc C#) được gọi là Code file (cách gọi mới) hay Code behind (cách gọi cũ). Mã lệnh tại đây thường xử lý các tác vụ liên quan đến nghiệp vụ, trong đó cũng có các câu lệnh cho phép gửi kết quả về cho phía trình duyệt. Cụ thể là phương thức `write` của đối tượng `Response`.

Ví dụ muốn trả một chuỗi S về cho trình duyệt hiển thị, ta viết: `Response.write(S)`.

Việc sử dụng phương thức `write` này như thế nào để “sinh ra” các phần tử cho trình duyệt hiểu là một kỹ năng quan trọng.

Nhìn chung, người ta thường chia các web form thành 2 phần là trang chứa giao diện (`.aspx`) và trang chứa mã code (`.vb`; `.cs`) để đảm bảo tính chuyên môn hóa và dễ bảo trì hơn.

### 3.7 HTML Server Controls và Web controls

#### 3.7.1 Giới thiệu

Để giúp cho việc phát triển các ứng dụng web nhanh chóng và thuận tiện, ASP.NET cung cấp cho chúng ta một tập hợp các điều khiển sẵn có để thực hiện hầu hết các công việc phổ biến hàng ngày. Các điều khiển này chia làm 2 loại: HTML Server Control và ASP.NET server control.

Các điều khiển (phần tử) này đều được xử lý bên phía server (có thuộc tính `runat=server`) vì vậy chúng ta đều có thể truy cập đến các phần tử này bằng các câu lệnh C# (các câu lệnh nằm bên trong Code file).

Điểm khác biệt giữa HTML Server control và ASP.NET server control ở chỗ:

- Điều khiển HTML Server control thì có số lượng và cách thức tạo giống hệt các phần tử HTML mà ta vẫn tạo trong trang HTML, chỉ khác một điều là có thêm `runat = "server"`;
- Điều khiển ASP.NET control thì có nhiều thuộc tính hơn, thực hiện được chức năng phức tạp hơn HTMLServer controls.

#### 3.7.2 Cách thức tạo phần tử HTML Server Control và ASP.NET control

##### a) HTML Server control

❖ Cú pháp tạo phần tử HTML Server control:

- `<Tên_Loại_Thẻ runat="server" thuộc_Tính = "giá trị" ....>`
- Trong đó: Tên loại thẻ là input, select, p, h1, ....

❖ Ví dụ: `<input type = "text" id="txtHoTen" runat = "server">`

##### b) ASP.NET server control

❖ Cú pháp tạo phần tử ASP.NET server control

- `<asp: Loại_PT thuộc_tính = "giá trị" .... runat = "server">`
- Trong đó **asp:** là bắt buộc, Loại\_PT có thể là button, textbox, calendar, select, treeview, adRotator, listview, gridview, image,....

❖ Ví dụ:

- `<asp:TextBox ID="txtHVT" runat="server"></asp:TextBox>`
- `<asp:Calendar ID="cal" runat="server" BorderColor="Blue"></asp:Calendar>`
- `<asp:Table> <asp:TableRow>  
    <asp:TableCell>cell</asp:TableCell></asp:TableRow>  
</asp:Table>`

**\*\*\*\* Chú ý \*\*\*\***

- ❖ Để có thể truy xuất tới các phần tử này trong Code file (hay server script phía server) thì mỗi phần tử cần phải đặt cho nó một id duy nhất.
- ❖ Trong tất cả các ứng dụng, nếu có thể được thì nên dùng các ASP.NET server control để đảm bảo tính tương thích với trình duyệt.
- ❖ Các điều khiển ASP.NET server control hoàn toàn có thể do người dùng tạo ra. (phần này sẽ được đề cập trong phần Lập trình ASP.NET nâng cao)

## BÀI 4: THỰC HÀNH

### Mục tiêu: Kết thúc bài thực hành này, người học có thể

- ❖ Tạo các phần tử web server control
- ❖ Sử dụng các câu lệnh phía server để truy cập các phần tử trang web.
- ❖ Tạo một trang web cho phép nhập thông tin về hồ sơ cán bộ.

### Nội dung thực hành

#### 1. Yêu cầu

Tạo một trang như bài thực hành số 2 nhưng sử dụng các phần tử ASP Server control thay vì sử dụng các phần tử HTML.

#### 2. Hướng dẫn:

##### a) Các kiến thức cơ bản:

##### - Tạo textbox:

```
<asp:TextBox runat="server" id="HoVaTen" class="TextboxDài" />
```

##### - Tạo textbox có nhiều dòng:

```
<asp:TextBox runat="server" ID="txtKyLuot" TextMode="MultiLine"
    CssClass="TextboxDài" Rows="5" Columns="80">
</asp:TextBox>
```

##### - Tạo hộp comboBox (HTML Server control)

```
<select runat="server" id="cboTiengAnh">
    <option>-----</option>
    <option value="A">A</option>
    <option value="B">B</option>
    <option value="C">C</option>
</select>
```

##### - Tạo hộp ComboBox (ASP server control)

```
<asp:ListBox ID="cboTiengAnh" runat="server">
    <asp:ListItem Text="A" Value="A"></asp:ListItem>
    <asp:ListItem Text="B" Value="B"></asp:ListItem>
    <asp:ListItem Text="C" Value="C"></asp:ListItem>
</asp:ListBox>
```

*Chú ý: Muốn tạo danh sách dạng ListBox, chỉ cần thêm thuộc tính Rows với giá trị > 1*

##### - Tạo nút Radio option (Những Radio có GroupName giống nhau sẽ cùng một nhóm)

```
<asp:RadioButton runat="server" type="radio" id="optNam"
    GroupName="GT" checked="true"/>
```

- Để truy xuất tới các phần tử server Control bằng mã lệnh C#, có thể thực hiện như lập trình winform (console) thông thường.
- Để truy xuất tới các phần tử trong một danh sách – ví dụ ComboBox – sử dụng thuộc tính Items. Ví dụ: cboNgaySinh.Items[1], ...

##### b) Minh họa mẫu

#### NhapHSCB.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="NhapHSCB.aspx.cs" Inherits="Lesson_04" %>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head id="Head1" runat="server">
  <title>Nhập hồ sơ cán bộ</title>
  <link rel="Stylesheet" href="Lesson04.css" type="text/css" />
</head>

<body >
  <form id="form1" action="CapNhatCanBo.aspx" method="post" runat="server" >
  <div style="text-align:center">
    <p style="border-bottom:solid; border-width:thin; font-size:20pt;
      margin:0; padding:0X; border-spacing:0px">
      CHƯƠNG TRÌNH QUẢN LÝ CÁN BỘ VERSION 1.0
    </p><br /> <br />

    <p class="HeadTitle">NHẬP HỒ SƠ CÁN BỘ</p>

    <table class="Table">
      <tr class="CellSpace">
        <td colspan="2" class="Tiêu_Đề_Chính">THÔNG TIN CÁ NHÂN</td>
      </tr>

      <tr>
        <td class="Cột1">*Họ và tên</td>
        <td class="Cột2">
          <asp:TextBox runat="server" id="HoVaTen" class="TextboxDài" />
        </td>
      </tr>

      <tr>
        <td class="Cột1">*Ngày sinh (ngày/tháng/năm)</td>
        <td class="Cột2">
          <asp:ListBox Rows="1" id="cboNgaySinh" runat="server">
          </asp:ListBox> /

          <asp:ListBox Rows="1" id="cboThangSinh" runat="server" >
          </asp:ListBox> /

          <asp:ListBox Rows="1" id="cboNamSinh" runat="server" >
          </asp:ListBox>

          Giới tính:
          <asp:RadioButton runat="server" type="radio" id="optNam"
            GroupName="GT" checked="true"/> Nam
          <asp:RadioButton runat="server" id="optNu" GroupName="GT"/> Nữ
        </td>
      </tr>

      <tr>
        <td class="Cột1">Chức vụ hiện tại (Đăng, chính quyền,...)</td>
        <td><asp:TextBox runat="server" ID="txtChucVu" CssClass="TextboxDài" /></td>
      </tr>

      <tr>
```

```
<td class="Cột1">*Quê quán</td> <td class="Cột2">
<asp:TextBox runat="server" ID="txtQueQuan" CssClass="TextboxDài"/></td>
</tr>

<tr>
<td class="Cột1">*Nơi ở hiện nay</td>
<td class="Cột2"> <asp:TextBox runat="server" ID="txtNoiOHienNay" CssClass="TextboxDài"/>
</td>
</tr>

<tr>
<td colspan="2" class="Tiêu_Đề_Chính">TRÌNH ĐỘ HỌC VẤN</td>
</tr>

<tr>
<td class="Cột1">Dân tộc : </td>
<td class="Cột2">
<asp:TextBox runat="server" ID="txtDanToc" Text="Kinh"/>
Tôn giáo: <asp:TextBox runat="server" ID="txtTonGiao" Text="Không" />
</td>
</tr>

<tr>
<td class="Cột1">Thành phần gia đình:</td>
<td class="Cột2"> <asp:TextBox runat="server" ID="txtThanhPhan" cssclass="TextboxDài"/></td>
</tr>

<tr>
<td class="Cột1">Nghề trước khi tuyển dụng</td>
<td class="Cột2">
<asp:TextBox runat="server" ID="txtNgheTruocKhiTuyen" CssClass="TextboxDài" />
</td>
</tr>

<tr>
<td class="Cột1">Tham gia cách mạng: </td>
<td class="Cột2">
Ngày <asp:TextBox runat="server" ID="txtNgayThamGiaCM" Text="....../.../...."
style="width:15%; text-align:center" onfocus="XuLyFocus(this);" onblur="XuLyLostFocus(this);" />

Tổ chức <asp:TextBox runat="server" ID="txtToChuc" style="width:20%" />
Công tác <asp:TextBox runat="server" ID="txtCongTac" style="width:20%" />
</td>
</tr>

<tr>
<td class="Cột1">Ngày vào Đảng: </td>
<td class="Cột2">
<asp:TextBox runat="server" ID="txtNgayVaoDang" Text=".../.../..." CssClass="NgayThang"
onfocus="XuLyFocus(this);" onblur="XuLyLostFocus(this);"/>
ngày vào chính thức <asp:TextBox runat="server" ID="txtThangVaoDang"
cssclass="NgayThang" Text=".../.../..." onfocus="XuLyFocus(this)" onblur="XuLyLostFocus(this)"/>
</td>
</tr>

<tr>
<td class="Cột1">Ngày nhập ngũ:</td>
```



```
<td class="Cột2"><input type="text" value=".../.../..." class="NgayThang"
onfocus="XuLyFocus(this);" onblur="XuLyLostFocus(this);"/>
ngày xuất ngũ <input type="text" class="NgayThang" value=".../.../..."
onfocus="XuLyFocus(this);" onblur="XuLyLostFocus(this);"/>
</td>
</tr>

<tr>
<td class="Cột1">*Trình độ Văn hóa: </td>
<td class="Cột2"><input style="width:15%" />
Học hàm:
<select runat="server" id="cboHocHam">
<option value="">-----</option>
<option value="Thạc Sĩ">Thạc sĩ </option>
<option value="Tiến Sĩ">Tiến sĩ</option>
</select>
Học vị :
<select runat="server" id="cboHocVi">
<option value="">-----</option>
<option value="Giáo sư">Giáo sư</option>
<option value="Phó giáo sư">Phó giáo sư</option>
</select>
</td>
</tr>

<tr>
<td class="Cột1">Lý luận chính trị </td>
<td class="Cột2">
<select runat="server" id="cboTrinhDoLyLuan">
<option>-----</option>
<option value="Sơ cấp">Sơ cấp</option>
<option value="Trung cấp">Trung cấp</option>
<option value="Cao cấp">Cao cấp</option>
<option value="Cử nhân">Cử nhân</option>
</select>
</td>
</tr>

<tr>
<td class="Cột1">Trình độ ngoại ngữ</td>
<td class="Cột2">
Anh <select runat="server" id="cboTiengAnh">
<option>-----</option>
<option value="A">A</option>
<option value="B">B</option>
<option value="C">C</option>
</select>
Nga <select runat="server" id="cboTiengNga">
<option>-----</option>
<option value="A">A</option>
<option value="B">B</option>
<option value="C">C</option>
</select>
Pháp <select runat="server" id="cboTiengPhap">
<option>-----</option>
```

```

        <option value="A">A</option>
        <option value="B">B</option>
        <option value="C">C</option>
    </select>
</td>
</tr>

<tr>
    <td class="Cột1">*Ngạch công chức, viên chức:</td>
    <td class="Cột2">
        <asp:TextBox runat="server" ID="txtNgach" style="width:20%" />
        Mã số: <asp:TextBox runat="server" ID="txtMaNgach" style="width:15%" />
        *Hệ số lương: <asp:TextBox runat="server" ID="txtHeSoLuong" style="width:15%" />
    </td>
</tr>

<tr>
    <td class="Cột1">Danh hiệu được phong (năm): </td>
    <td class="Cột2"><asp:TextBox runat="server" ID="txtDanhHieu" CssClass="TextboxDài" />
</td>
</tr>

<tr>
    <td class="Cột1">Sở trường công tác:</td>
    <td class="Cột2"><asp:TextBox runat="server" ID="txtSoTruongCT" CssClass="TextboxDài" />
</td>
</tr>

<tr>
    <td class="Cột1">Khen thưởng (huân,huy chương cao nhất)</td>
    <td class="Cột2"><asp:TextBox runat="server" ID="txtKhenThuong" CssClass="TextboxDài" />
</td>
</tr>

<tr>
    <td class="Cột1">Kỷ luật (đăng, chính quyền, năm, lý do, hình thức)</td>
    <td class="Cột2">
        <asp:TextBox runat="server" ID="txtKyLuat" TextMode="MultiLine"
        CssClass="TextboxDài" Rows="5" Columns="80"></asp:TextBox>
    </td>
</tr>

<tr class="Tiêu_Đề_Chính">
    <td colspan="2">
        ĐÀO TẠO, BỒI DƯỠNG CHUYÊN MÔN, NGHIỆP VỤ, LÝ LUẬN, NGOẠI NGỮ
    </td>
</tr>

<tr>
    <td class="Cột1">
        Ghi rõ Tên trường, ngành học, thời gian, loại hình,văn bằng, chứng chỉ
    </td>
    <td><asp:TextBox runat="server" ID="txtQuaTrinhDaoTao"
    CssClass="TextboxDài" TextMode="MultiLine" Columns="100" rows="5"></asp:TextBox>
    </td>
</tr>
```

```
</tr>
<tr>
  <td colspan="2" style="color:Blue">
    ** Loại hình: Chính qui, tại chức, chuyên tu, bồi dưỡng; văn bằng: Tiến sĩ,
    thạc sĩ, cử nhân, kỹ sư.
  </td>
</tr>

<tr class="Tiêu_Đề_Chính">
  <td colspan="2">TÓM TẮT QUÁ TRÌNH CÔNG TÁC</td>
</tr>
<tr>
  <td class="Cột1">Ghi rõ thời gian bắt đầu và kết thúc; chức danh, chức vụ,
    đơn vị công tác tương ứng</td>
  <td><asp:TextBox runat="server" ID="txtQuaTrinhCongTac"
    CssClass="TextboxDài" TextMode="MultiLine" Columns="100" rows="5">
    </asp:TextBox>
  </td>
</tr>

<tr class="Tiêu_Đề_Chính">
  <td colspan="2">Đặc điểm lịch sử bản thân</td>
</tr>
<tr>
  <td class="Cột1">Đặc điểm lịch sử bản thân</td>
  <td class="Cột2">
    <asp:TextBox runat="server" ID="txtDDBanThan"
    CssClass="TextboxDài" TextMode="MultiLine" Columns="100" rows="5">
    </asp:TextBox>
  </td>
</tr>

<tr>
  <td class="Cột1">Quan hệ với người nước ngoài</td>
  <td class="Cột2">
    <asp:TextBox runat="server" ID="txtQHNGuoiNN"
    CssClass="TextboxDài" TextMode="MultiLine" Columns="100" rows="5">
    </asp:TextBox>
  </td>
</tr>

<tr>
  <td class="Cột1">Quan hệ gia đình (Bố, mẹ, anh chị em ruột)</td>
  <td class="Cột2">
    <asp:TextBox runat="server" ID="txtQHGD" CssClass="TextboxDài"
    TextMode="MultiLine" Columns="100" rows="5"></asp:TextBox>
  </td>
</tr>

<tr>
  <td class="Cột1">Hoàn cảnh kinh tế gia đình</td>
  <td class="Cột2">
    <asp:TextBox runat="server" ID="txtHoanCanhKT" CssClass="TextboxDài"
    TextMode="MultiLine" Columns="100" rows="5"></asp:TextBox>
  </td>
</tr>
```

```
</td>
</tr>
</table>

<asp:Label runat="server" id="lblTrangThai" visible="false" style="text-align:center" >
</asp:Label>

<br />
<table class="Table" style="border:0">
  <tr>
    <td style="text-align:right">
      <asp:Button runat="server" id="cmdSubmit" Text="    Cập nhật    " />
    </td>

    <td style="text-align:left">
      <input type="reset" value=" Nhập mới " />
    </td>
  </tr>
</table>
</div>
</form>

<script language="javascript" type="text/javascript">
  var Giá_Trị_Cũ;

  /// Hàm xử lý khi phần tử nhận được focus
  function XuLyFocus(txt)
  {
    Giá_Trị_Cũ=txt.value;
    txt.value="";
  }

  /// Hàm xử lý khi phần tử mất focus
  function XuLyLostFocus(txt)
  {
    if (txt.value=="") txt.value=Giá_Trị_Cũ;
  }
</script>
</body>
</html>
```

#### **NhapHSCB.aspx.cs**

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
```

```
public partial class Lesson_04 : System.Web.UI.Page
{
    protected void Khởi_Tạo_Các_Controls ()
    {
        int i;
        ListItem L;

        //Ngày sinh
        for (i = 1; i <= 31; i++)
        {
            L = new ListItem (i.ToString (), i.ToString ());
            cboNgaySinh.Items.Add (L);
        }

        // Tháng sinh
        for (i = 1; i <= 12; i++)
        {
            L = new ListItem (i.ToString (), i.ToString ());
            cboThangSinh.Items.Add (L);
        }

        //Năm sinh
        for (i = 1950; i <= 1990; i++)
        {
            L = new ListItem (i.ToString (), i.ToString ());
            cboNamSinh.Items.Add (L);
        }
    }

    public void KiemTra ()
    {
        if (txtHoVaTen.Text.Trim ().ToString () == "")
        {
            lblTrangThai.Visible = true;
            lblTrangThai.Text = "Họ tên không được rỗng !";
        }
        else
        {
            lblTrangThai.Visible = false;
        }
    }

    protected void Page_Load (object sender, EventArgs e)
    {
        Khởi_Tạo_Các_Controls ();
    }

    protected void cmdSubmit_Click (object sender, EventArgs e)
    {
        KiemTra ();
    }
}
```

## BÀI 5: Tìm hiểu và sử dụng các Server/Ajax Controls

### Mục tiêu của bài: Kết thúc bài học này học viên có thể:

- ❖ Nêu được chức năng của một số phần tử HTML Server control và ASP Server Control thường dùng.
- ❖ Khai báo sự kiện và viết được trình xử lý sự kiện gắn vào các Controls.
- ❖ Phân tích được cơ chế xử lý sự kiện trong một trang ASP.NET.
- ❖ Sử dụng các điều khiển HTML, ASP Server control để xây dựng một số ứng dụng đơn giản.
- ❖ Trình bày được tổng quan về công nghệ AJAX.

### 5.1 HTML Server Controls

Đây là các điều khiển được tạo bằng cách thêm thuộc tính ID và RUNAT = "Server" bên trong mỗi thẻ HTML thông thường. Cách này thường được dùng khi muốn chuyển đổi một trang ASP trước đây sang ASP.NET. Tuy nhiên, các điều khiển loại này không có nhiều thuộc tính phục vụ công việc lập trình. So với ASP Server Control (hay web server control) thì được dùng ít hơn, do web server control tương thích với nhiều trình duyệt hơn và tập thuộc tính, phương thức cũng phong phú hơn rất nhiều.

Vì lý do đó và cũng để tránh sự lẫn lộn giữa HTML Server control và web server control, kể từ nay về sau chúng ta thống nhất chỉ sử dụng web server control mà không sử dụng đến HTML Server control (trừ những ngoại lệ).

### 5.2 Web server Controls

Để có thể sử dụng code bên phía server truy xuất tới các phần tử trong webform, ASP.NET cung cấp cho chúng ta một số phần tử đặc biệt, gọi là web server control hay ASP Server controls. Với các phần tử loại này, chúng ta có thể tạo ra các phần tử rất phức tạp chỉ với một hoặc vài dòng code. Ví dụ phần tử Calendar, phần tử GridView, Container,... Các phần tử này hỗ trợ phong cách lập trình theo mô hình hướng đối tượng.

#### 5.2.1 Khai báo (tạo các phần tử web server control)

Cú pháp khai báo thường có dạng:

Dạng 1 (Không có thẻ đóng)	Dạng 2 (có thẻ đóng tường minh)
<code>&lt;asp: Tên_Điều_Khiển ID = "Định danh duy nhất" runat = "Server" tt1="gt1" tt2="gt2" ... &gt;</code>	<code>&lt;asp: Tên_Điều_Khiển ID = "Định danh duy nhất" runat = "Server" tt1="gt1" tt2="gt2" ... &gt; &lt;/ asp: Tên_Điều_Khiển&gt;</code>

Trong đó: 2 thuộc tính thường bắt buộc phải có là **ID** và **runat**. Thuộc tính ID là tên duy nhất được dùng để tham chiếu khi viết code phía server. Thuộc tính runat="Server" chỉ ra rằng phần tử này cần phải được xử lý phía server trước khi gửi về cho Client.

Có thể đặt giá trị cho các thuộc tính ở ngay bên trong các thẻ này, thậm chí có thể đưa các mã JavaScript ! ...

### 5.2.2 Cơ chế xử lý các phần tử web server control

Cơ chế xử lý các phần tử này như sau: Phía máy chủ sẽ đọc những thẻ nào có thuộc tính `runat = "Server"` và đem xử lý, kết quả sau khi xử lý sẽ được gửi trả về cho phía trình duyệt.

Ta xét một ví dụ cụ thể dưới đây để hiểu rõ cơ chế xử lý các phần tử webserver control bên phía máy chủ:

#### **Ví dụ 1:** Xử lý thẻ **form** có thuộc tính **runat="Server"**

```
<form runat="server">
```

```
</form>
```

Server sẽ đọc thẻ form này và xử lý (vì có thuộc tính `runat = "server"`), và cho kết quả là:

```
<form name="ctl01" method="post" action="Default.aspx" id="ctl01">
  <div>
    <input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
      value="/wEPDwUJNzgZNDMRYrsdybepETo3ZpHQh9iJeRBA=="
    />
  </div>
</form>
```

Như vậy, rõ ràng là server đã "chế biến" thêm một chút trước khi trả về cho trình duyệt, đó là bỏ thuộc tính `runat="Server"` nhưng thêm thuộc tính `name="ctl01"`, `method`, `action`, `id`... ngoài ra còn thêm các thẻ `<div>`, `<input>`. Nói cách khác là server đã biến cái ban đầu mà trình duyệt không thể hiểu được (phần có `runat="server"` ở trên) thành cái mà trình duyệt có thể xử lý được (phần kết quả ở dưới).

#### **Ví dụ 2:** Xử lý phần tử **TextBox**:

```
<asp:TextBox ID="HVT" runat="server" Text="Hello" BackColor="blue"></asp:TextBox>
```

Server sẽ chế biến (gender) thành:

```
<input name="HVT" type="text" value="Hello" id="HVT" style="background-color:blue;" />
```

Ở ví dụ này, sau khi xử lý server đã bỏ thuộc tính **runat**, **asp:TextBox**, **BackColor**... và tạo ra thẻ tương đương mà trình duyệt có thể hiểu là `input`, `type="text"`, `background-color`, `name`...

#### **Ví dụ 3:** Xử lý phần tử **Calendar**

Với các phần tử phức tạp hơn thì Server sẽ phải mất nhiều công chế biến hơn. Ví dụ đối với thẻ **Calendar** như sau (chỉ có 1 dòng):

```
<asp:Calendar ID="Cal" runat="server">    </asp:Calendar>
```



**Thì server sẽ tạo ra đoạn code HTML rất "khủng" !!!:**

```
<table id="Table1" cellspacing="0" cellpadding="2" title="Calendar" border="0"
  style="border-width:1px;border-style:solid;border-collapse:collapse;">
  <tr>
    <td colspan="7" style="background-color:Silver;">
      <table cellspacing="0" border="0" style="width:100%;">
        <tr>
          <td style="width:15%;">
            <a href="javascript:__doPostBack('cal','V3074')"
              style="color:Black" title="the previous month">&lt;
            </a>
          </td>
          <td align="center" style="width:70%;">July 2008</td>
          <td align="right" style="width:15%;">
            <a href="javascript:__doPostBack('cal','V3135')"
              style="color:Black" title="Go to the next month">&gt;
            </a>
          </td>
        </tr>
      </table>
    </td>
  </tr>
  <tr>
    <th align="center" abbr="Sunday" scope="col">Sun</th>
    <th align="center" abbr="Monday" scope="col">Mon</th>
    <th align="center" abbr="Tuesday" scope="col">Tue</th>
    <th align="center" abbr="Wednesday" scope="col">Wed</th>
    <th align="center" abbr="Thursday" scope="col">Thu</th>
    <th align="center" abbr="Friday" scope="col">Fri</th>
    <th align="center" abbr="Saturday" scope="col">Sat</th>
  </tr>
  <tr>
    <td align="center" style="width:14%;">
      <a href="javascript:__doPostBack('cal','3102')"
        style="color:Black" title="June 29">29</a>
    </td>
    <td align="center" style="width:14%;">
      <a href="javascript:__doPostBack('cal','3104')"
        style="color:Black" title="July 01">1</a>
    </td>
    <td align="center" style="width:14%;">
      <a href="javascript:__doPostBack('cal','3105')"
        style="color:Black" title="July 02">2</a>
    </td>
    <td align="center" style="width:14%;">
      <a href="javascript:__doPostBack('cal','3106')"
        style="color:Black" title="July 03">3</a>
    </td>
    <td align="center" style="width:14%;">
      <a href="javascript:__doPostBack('cal','3107")
```

```
        style="color:Black" title="July 04">4</a>
    </td>
    <td align="center" style="width:14%;">
        <a href="javascript:__doPostBack('cal','3108')"
        style="color:Black" title="July 05">5</a>
    </td>
</tr>
```

>>>>> Còn gấp khoảng 5 lần đoạn code ở trên nữa >>>>>

Vì trang HTML truyền thống không có một phần tử nào có thể hiển thị đầy đủ một lịch biểu (Calendar) nên ASP.NET đã phải tạo ra phần tử đó bằng cách kết hợp từ rất nhiều thẻ HTML đơn giản (table, th, tr, td, a,... ) như ở trên.

Ví dụ này là một minh chứng cho thấy ASP.NET đã làm đơn giản hóa quá trình phát triển ứng dụng. Giảm thiểu việc phải viết code và bảo trì chương trình dễ dàng hơn rất nhiều.

**Ví dụ 4:** Xử lý phần tử web server control nhưng có thêm thuộc tính phía client là một đoạn script (Lưu ý: Thuộc tính onClick không có trong danh sách thuộc tính của asp:label):

```
<asp:Label  
    runat="server" ID="lblHello"  
    Text="Hello world"  
    onClick="alert('Hello world');">  
</asp:Label>
```

Kết quả server xử lý phần tử Label ở trên là:

```
<span id="Span1" onClick="alert('Hello world');">Hello world</span>
```

Ta biết rằng phần tử asp:Label không có thuộc tính onClick (bằng chứng là khi gõ không thấy xuất hiện trong danh sách). Nhưng khi chạy trang web không thấy có báo lỗi. Vậy cơ chế xử lý của ASP.NET là như thế nào đối với những phần tử như thế này ?

Cách thức như sau: Khi các thẻ có runat="server" thì sẽ được web server đọc và xử lý tất cả các nội dung nằm bên trong thẻ đó. Tuy nhiên, nếu gặp thẻ nào đó mà có thẻ nó chưa hiểu (ví dụ chỉ client hiểu được) thì nó trả nguyên phần đó cho phía client.

Chính vì vậy mà ta thấy kết quả trả về cho phía client vẫn chứa nguyên phần onClick="alert('Hello world');".

Đây cũng là cách mà người ta hay sử dụng để đan xen các đoạn code vừa được xử lý phía server, vừa xử lý phía client.

**Ví dụ 5:** Xử lý các phần tử server mà trong đó có chứa mã của Client.

```
<asp:TextBox runat="server" ID="ht" Text="Hello"  
    onmousemove="document.bgColor='blue';"  
    onmouseout="document.bgColor='yellow';"  
>
```

Kết quả mà server gửi trả về trình duyệt sau khi xử lý là:

```
<input name="ht" type="text" value="Hello" id="Text1"  
    onmousemove="document.bgColor='blue';"  
    onmouseout="document.bgColor='yellow';"  
>
```

Vì hai sự kiện onMouseMove và onMouseOut không có trên server nên nó sẽ gửi nguyên giá trị đó cho phía trình duyệt. Đối với trình duyệt thì 2 sự kiện này đã quá quen thuộc: onMouseMove xuất hiện khi người dùng di chuột trên phần tử textbox và sự kiện onMouseOut xuất hiện khi người dùng di chuyển chuột ra ngoài textbox.

### 5.2.2 Thực thi các câu lệnh tại phía server

Trong rất nhiều trường hợp chúng ta muốn thực thi một số câu lệnh ngay tại phía server khi người dùng thực hiện một thao tác nào đó – ví dụ click chuột – thì có thể viết sẵn các thủ tục sự kiện để thực thi ứng với các sự kiện này.

Để gọi một thủ tục (phương thức) phía server khi một sự kiện xảy ra đối với phần tử web server control, thì viết như sau:

```
<asp:Tên_Phần_Tử runat = "Server" onClick = "Tên_Phương_Thức" ....>
```

Ví dụ: Gọi phương thức KiemTra khi người dùng nhấn vào nút "cmdKiemTra":

```
<asp:Button ID="cmdKiemTra" runat="server" OnClick="KiemTra" />
```

**Lưu ý:** Tên của phương thức trong phần OnClick không chứa tham số và dấu ngoặc. Nhưng khi định nghĩa phương thức này thì thường phải có 2 tham số như ví dụ sau:

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;

public partial class _default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void KiemTra (object sender, EventArgs e)
    {
        /// Đặt câu lệnh ở đây !
    }
}
```

### 5.2.3 Mô hình xử lý sự kiện trong ASP.NET

1. Lần đầu tiên khi trang web được chạy. ASP.NET tạo trang và các đối tượng và thực hiện khởi tạo, sau đó trang sẽ được chuyển đổi (rendered) thành trang HTML để trả về cho phía client. Các đối tượng của trang sau đó cũng được giải phóng khỏi bộ nhớ của server.
2. Tại một thời điểm nào đó, nếu người dùng thực hiện một số công việc (ví dụ click chuột lên button có runat = "Server") khi đó hệ thống sẽ tự động thực hiện hành động gọi là "Postback" (Dịch là gửi/gọi trở lại Server) và lúc đó toàn bộ dữ liệu trong phần tử form sẽ được gửi về Server.
3. ASP.NET tạo lại các đối tượng của trang và trả về client trạng thái cuối cùng khi chúng được gửi đi.
4. Tiếp theo, ASP.NET sẽ kiểm tra xem thao tác nào đã gây ra sự kiện postback đó và kích hoạt (gọi) sự kiện tương ứng (ví dụ Button.Click). Thông thường, trong phần

xử lý sự kiện này chúng ta thường thực hiện một số xử lý như cập nhật CSDL, kiểm tra,....

5. Trang này sau đó được biến đổi (chuyển đổi) thành trang HTML và gửi về cho client (trình duyệt). Tiếp theo các đối tượng của trang (như asp:Button; asp:ListBox,...) sẽ được giải phóng khỏi bộ nhớ. Nếu có sự kiện Postback khác xuất hiện thì ASP.NET sẽ lặp lại xử lý ở bước 2 cho đến bước 4.

Chú ý: Khi tạo một số phần tử như Button thì mặc định hệ thống sẽ tự động Postback mỗi khi người dùng click chuột. Đối với một số phần tử khác như TextBox thì mặc định là không Postback. Nếu muốn Postback thì đặt thuộc tính AutoPostBack = "true", ví dụ:

```
<asp:DropDownList runat="server" ID="ds" AutoPostBack="true"></asp:DropDownList>
```

Hay : 

```
<asp:TextBox runat="server" ID="txtHVT" AutoPostBack="true"></asp:TextBox>
```

Khi đã đặt AutoPostBack = "true" thì mỗi khi người dùng chọn một mục khác (đối với DropDownList) và thay đổi nội dung (đối với TextBox) thì hệ thống sẽPostBack trang web về bên Server. Và sự kiện SelectedIndexChanged và TextChanged tương ứng sẽ được gọi:

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;

public partial class _default : System.Web.UI.Page
{
    protected void ds_SelectedIndexChanged (object sender, EventArgs e)
    {
    }

    protected void txtHVT_TextChanged (object sender, EventArgs e)
    {
    }
}
```

## 5.3 Ajax Control Toolkit

### 5.3.1 Giới thiệu

Rõ ràng là ASP.NET đã cung cấp cho chúng ta rất nhiều các điều khiển (asp server control), giúp giảm đáng kể công sức phát triển ứng dụng. Tuy nhiên, với chừng đó thì vẫn chưa đủ và vẫn còn vô số những hạn chế cần phải khắc phục. Đặc biệt là các điều khiển hiển thị trên màn hình, các điều khiển kiểm tra dữ liệu nhập (Data Validation control),...

Bộ Ajax Toolkit Control là một tập các điều khiển nhằm đáp ứng các yêu cầu đó.

Bộ Ajax Toolkit control không phải là thành phần nội tại của ASP.NET. Nó chỉ là một thành phần bổ sung cho ASP.NET. Có thể download tại địa chỉ <http://asp.net>

Sau khi cài đặt file ASPAJAXExtSetup.msi, hệ thống sẽ có một file là AjaxControlToolkit.dll, chúng ta sẽ vào menu website → Project Reference

### 5.3.2 Hướng dẫn sử dụng một số Ajax Control cơ bản

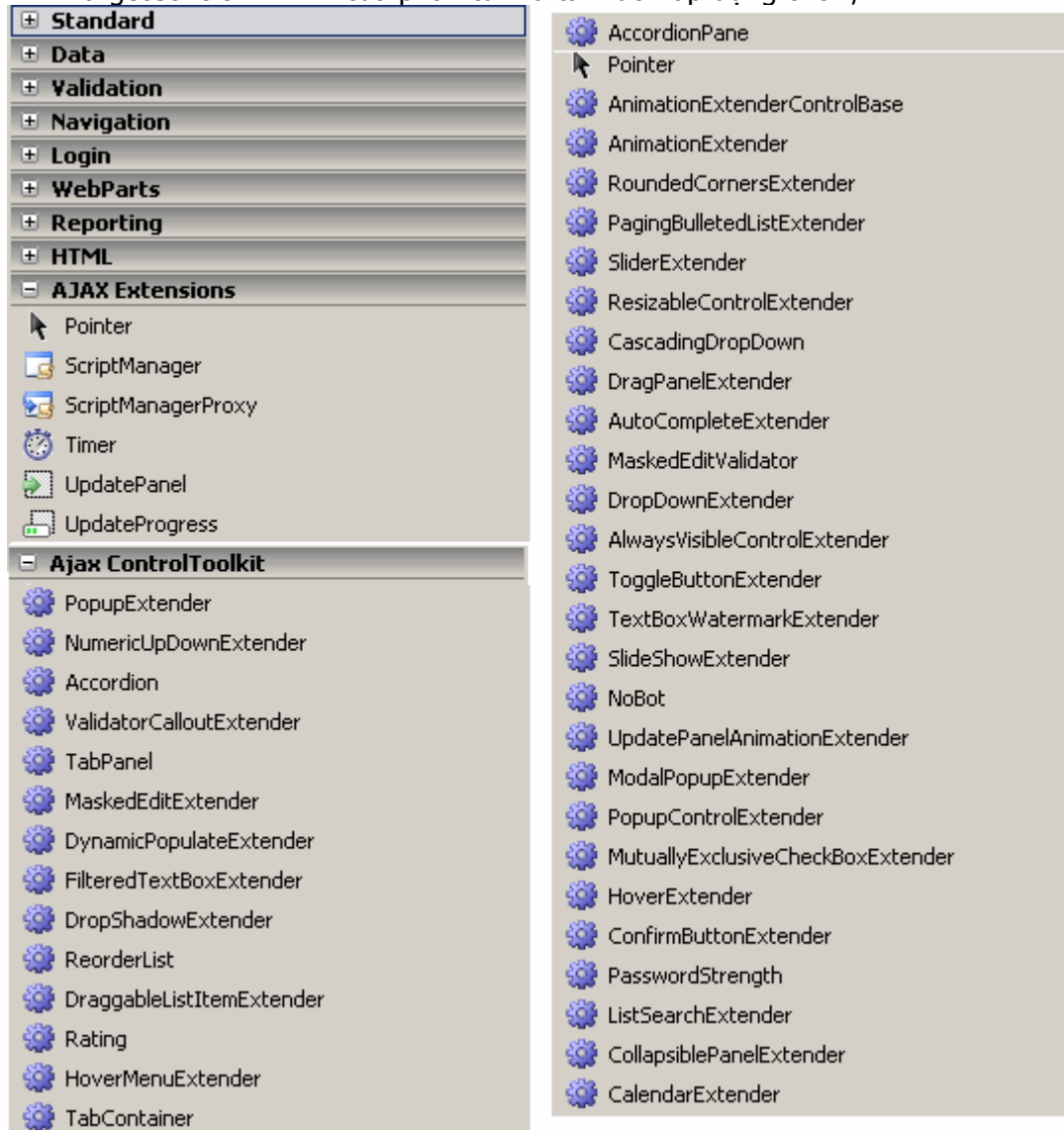
Các bước sử dụng:

**Bước 1:** Thêm tham chiếu Ajax control toolkit vào Project

**Bước 2:** Chèn một ScriptManager vào trang và Tạo các phần tử theo cú pháp

<ajaxToolkit:Tên\_Phần\_Tử ID="Giá\_Trị\_ID" runat="server"

TargetControlID="ID của phần tử mà ta muốn áp dụng cho" />



#### Một số phần tử Ajax

**Ví dụ:** Cho phép người dùng chọn ngày tháng và năm khi người dùng focus vào ô textbox:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default2.aspx.cs" Inherits="Default2" %>
```

```
<%@ Register  
    Assembly="AjaxControlToolkit"  
    Namespace="AjaxControlToolkit"  
    TagPrefix="ajaxToolkit" %>
```

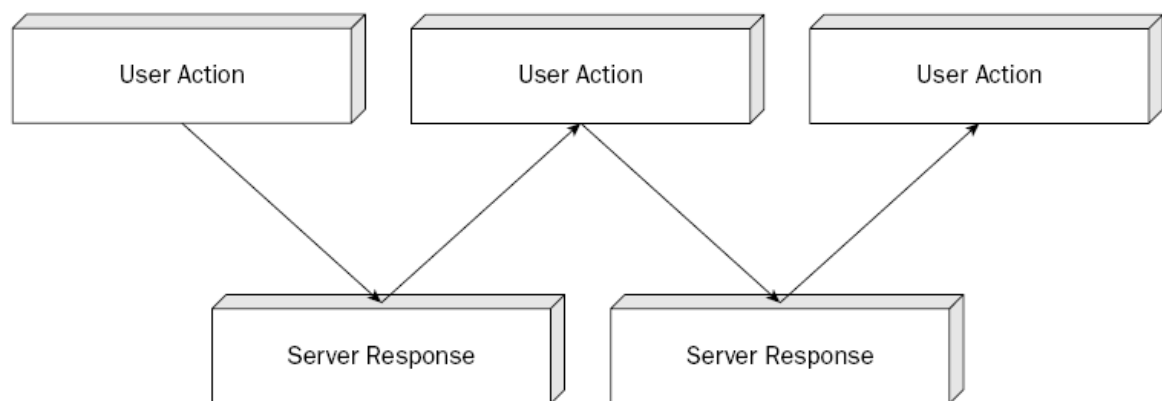
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Untitled Page</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:TextBox runat="server" ID="txtNgaySinh" autocomplete="off" />

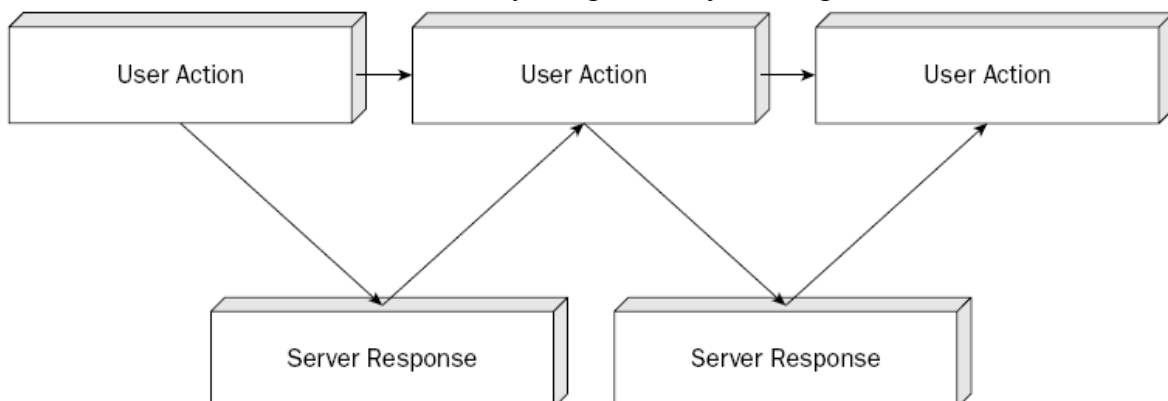
    <asp:ScriptManager ID="ScriptManager1" runat="server">
    </asp:ScriptManager>

    <ajaxtoolkit:calendarextender ID="defaultCalendarExtender" runat="server"
      TargetControlID="txtNgaySinh" />
  </form>
</body>
</html>
```

## 5.4 Thảo luận công nghệ Ajax



Mô hình xử lý trang web truyền thống



Mô hình xử lý của Ajax

## BÀI 6: THỰC HÀNH

**Mục tiêu:** Sau khi kết thúc bài thực hành, người học có thể:

- ❖ Sử dụng ngôn ngữ lập trình C# để lập trình xử lý phía server.
- ❖ Sử dụng được một số điều khiển Ajax để trình bày giao diện và kiểm tra dữ liệu nhập.

### **Nội dung:**

#### **1. Viết lệnh phía server**

**Yêu cầu:** Kiểm tra dữ liệu nhập ở bài trước (Nhập hồ sơ cán bộ), đảm bảo các yêu cầu sau:

- ❖ Họ tên không được rỗng, chỉ chứa chữ cái và dấu cách. Độ dài  $\leq 30$ ;
- ❖ Ngày tháng năm sinh phải hợp lệ. Ví dụ tháng 2 của năm nhuận có 29 ngày, các năm khác là 28 ngày.
- ❖ Ngày sau phải "lớn hơn" ngày trước. Ví dụ ngày xuất ngũ phải trước ngày nhập ngũ.
- ❖ Các trường số chỉ chứa giá trị là số dương, không chứa ký tự thông thường.
- ❖ Các ô textbox nhiều dòng, chứa tối đa là 1000 ký tự.
- ❖ Chừng nào dữ liệu còn nhập sai thì cần phải focus đến đó để yêu cầu người dùng sửa lại.
- ❖ Nếu mọi dữ liệu nhập đều hợp lệ thì gửi ẩn các điều khiển và chỉ một dòng thông báo là : **"Hồ sơ đã được cập nhật"** ra giữa màn hình !.

**Hướng dẫn:**

- ❖ Để đọc giá trị value của mục đang được chọn trong DropDownList thì viết: `DDL_Name.Text` . Trong đó: `DDL_Name` là id của dropdownlist.
- ❖ Để truy xuất tới một phần tử có chỉ số `i` trong Dropdownlist, viết: `DDL_Name.Items[i]`;
- ❖ Để xóa các mục: `DDL_Name.Clear()`; để đếm số mục : `DDL_Name.Count`;
- ❖ Để thêm một mục vào DDL (Dropdownlist)
  - `DS.Items.Add ("Mục mới");` Hoặc
  - `DS.Items.Add (new ListItem ("text", "value"));`
- ❖ Hàm chuyển ngày tháng dạng xâu sang kiểu Date: `DateTime d`;
  - `DateTime D = DateTime.Parse (s);`
  - `int D.Day; D.Month; D.Year` → trả về ngày, tháng năm của D.
- ❖ Để focus tới một phần tử: Viết `<Giá trị ID>.Focus()`; VD: `DS.Focus()`;
- ❖ Phép toán lấy phần nguyên, phần dư:
  - Lấy nguyên: `20 / 6 → 3`
  - Lấy phần dư: `20 % 6 → 2`

#### **2. Sử dụng một số điều khiển Ajax Control Toolkit**

Cách đăng ký và đưa Ajax Control Toolkit vào dự án:



- Cần có 3 file sau:



- **Bước 1**, vào menu website → Add Reference → Chọn Tab Browse. Sau đó chọn 2 tệp (1) và (2) ở trên. Click OK.
- **Bước 2**: Click chuột phải lên hộp toolbox và chọn "**Choose Items**". Khi hộp thoại mở ra, Browse tới file (3) ở trên. Click OK.

### 2.1 Sử dụng Calendar

Yêu cầu: Viết một trang cho phép người dùng nhập vào ngày sinh của họ. Ngày sinh này được nhập bằng cách chọn trực tiếp trên lịch (Calendar).

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default3.aspx.cs" Inherits="Default3" %>
```

```
<%@ Register  
    Assembly="AjaxControlToolkit"  
    Namespace="AjaxControlToolkit"  
    TagPrefix="ajaxToolkit" %>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">  
<head runat="server">  
    <title>Calendar</title>  
</head>  
<body>
```

```
    <form id="form1" runat="server">  
        <asp:ScriptManager ID="ScriptManager1"  
            runat="Server" EnableScriptGlobalization="true"  
            EnableScriptLocalization="true" />
```

```
        <b>Nhập ngày sinh:</b>  
        <asp:TextBox runat="server" ID="txtNS"></asp:TextBox>
```

```
        <ajaxToolkit:CalendarExtender ID="cal"  
            TargetControlID="txtNS" runat="server" >
```

```
            </ajaxToolkit:CalendarExtender>  
        </form>  
</body>  
</html>
```

#### Lưu ý:

- Trong thẻ <ajaxToolkit:CalendarExtender ID="cal" TargetControlID="txtNS" runat="server" >. Thuộc tính TargetControlID cho biết là sẽ hiển thị Calendar khi người dùng focus điều khiển có id là txtNS.
- o Trong trang cần phải có thẻ <asp:ScriptManager và đặt trong thẻ Form.

## 2.2 Nhập ngày tháng theo định dạng (sử dụng điều khiển MaskedEdit extender)

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="MaskedEdit.aspx.cs"
Inherits="MaskedEdit" %>
```

```
<%@ Register
    Assembly="AjaxControlToolkit"
    Namespace="AjaxControlToolkit"
    TagPrefix="ajaxToolkit" %>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Maskedit</title>
</head>
<body>
    <form id="form1" runat="server">
```

```
        <asp:ScriptManager ID="SM1" runat="server" EnablePartialRendering="True" />
```

Nhập ngày tháng theo dạng: MM/dd/yyyy (culture sensitive)<br />

```
        <asp:TextBox ID="TextBox1" runat="server" Width="130px"
ValidationGroup="Demo1" MaxLength="1" style="text-align:justify" />
```

```
        <ajaxToolkit:MaskedEditExtender ID="MaskedEditExtender1" runat="server"
            TargetControlID="TextBox1"
            Mask="99/99/9999"
            MessageValidatorTip="true"
            MaskType="Date"
            DisplayMoney="Left"
            AcceptNegative="Left" />
```

```
        <ajaxToolkit:MaskedEditValidator ID="MaskedEditValidator1" runat="server"
            ControlExtender="MaskedEditExtender1"
            ControlToValidate="TextBox1"
            IsValidEmpty="False"
            EmptyValueMessage="Phải nhập ngày tháng"
            InvalidValueMessage="Ngày tháng nhập sai"
            ValidationGroup="Demo1"
            Display="Dynamic"
            TooltipMessage="Nhập vào một ngày" />
```

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

**\*\* Chú ý:** Trường hợp này cần phải đặt 2 điều khiển Ajax, một cái là ajaxToolkit:MaskedEditExtender và một cái là ajaxToolkit:MaskedEditValidator. Cái thứ hai cần phải gắn với cái thứ nhất bằng cách đặt thuộc tính ControlExtender.

### 2.3 Sử dụng Dialog modal

```
%@ Page Language="C#" AutoEventWireup="true" CodeFile="PopupDialog.aspx.cs"
Inherits="PopupDialog" %>
```

```
<%@ Register
    Assembly="AjaxControlToolkit"
    Namespace="AjaxControlToolkit"
    TagPrefix="ajaxToolkit" %>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Modal Dialog</title>
</head>
<body>
```

Phần tiêu đề của  
hộp thoại. Đặt  
trong một Panel.

```
<form id="form1" runat="server">
<asp:Button ID="cmdOpenPopup" runat="server" Text="Cửa sổ đăng nhập" />

<asp:ScriptManager ID="sm1" runat="server"></asp:ScriptManager>
```

```
User name: <asp:TextBox ID="txtUserID" runat="server"></asp:TextBox> <br />
Password : <asp:TextBox ID="txtPassword" runat="server"></asp:TextBox><br />
<asp:Button ID="cmdCancel" runat="server" Text="Hủy bỏ" />
<asp:Button ID="cmdLogin" runat="server" Text="Đăng nhập" />
</asp:Panel>
```

Phần nội dung  
hộp thoại. Đặt  
trong một Panel.

```
<ajaxToolkit:ModalPopupExtender
    ID="Popup01" runat="server"
    TargetControlID="cmdOpenPopup"
    PopupControlID="NoiDung"
    OkControlID="cmdLogin"
    DropShadow="true"
    PopupDragHandleControlID="TieuDe">
</ajaxToolkit:ModalPopupExtender>
```

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

## BÀI 7: Tạo và sử dụng Custom Control

### 7.1 Giới thiệu User Custom Control

Visual studio cung cấp cho chúng ta rất nhiều các điều khiển để phát triển ứng dụng gọi là điều khiển nội tại (Instrict control). Ngoài ra, nó còn cung cấp cho chúng ta khả năng tự xây dựng các điều khiển tùy biến nếu các điều khiển hiện hành không đáp ứng được yêu cầu. Ví dụ: Nếu ứng dụng của bạn cần chiếc máy tính (Calculator) ở rất nhiều trang thì giải pháp tốt nhất là tạo một điều khiển Calculator riêng thay việc kết hợp các điều khiển truyền thống, khi đó ta có thể sử dụng điều khiển này trong toàn bộ ứng dụng.

Bài học này sẽ hướng dẫn cách tạo và sử dụng điều khiển do lập trình viên tự xây dựng – hay còn gọi là điều khiển tùy biến (Custom Control). Tiếp theo sẽ minh họa thêm một số ví dụ về tạo điều khiển tùy biến để người đọc hiểu rõ hơn.

Thực chất của User Custom Control (UCC) chính là một "trang con", trong đó có thể chứa **bất kỳ nội dung nào** (trừ các thẻ <HTML> <BODY>, <FORM>, vì một trang chỉ có duy nhất một lần xuất hiện các thẻ này). "Trang con" này sau đó có thể được chèn (Include) vào các trang khác để sử dụng. Khi muốn cập nhật nội dung ở tất cả các trang, ta chỉ việc sửa đổi duy nhất UCC ban đầu. Khả năng này của ASP.NET giúp chúng ta xây dựng ứng dụng nhanh hơn, dễ bảo trì hơn.

Mỗi một UCC được đặt trong một trang có phần mở rộng là \*.ascx. File này có đặc điểm là không truy cập trực tiếp từ trình duyệt mà chỉ được chèn vào các trang aspx.

### 7.2 Các bước tạo User Custom control

Việc thực hiện tạo User custom control trải qua 3 bước chính như sau:

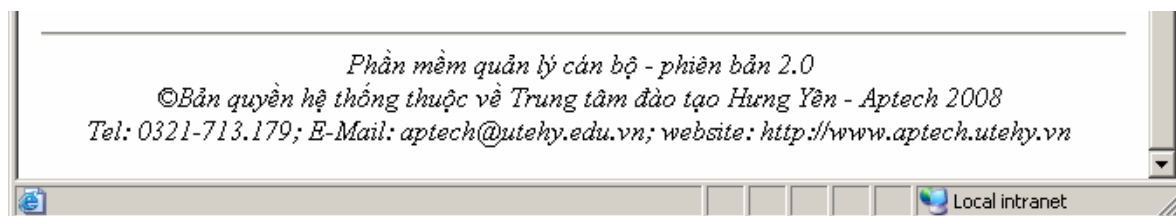
**Bước 1:** Thêm một web form vào Project hiện hành

- ❖ Vào menu website, chọn Add new item. Chọn loại Web user control
- ❖ Đặt tên cho web user control.

**Bước 2:** Soạn nội dung của trang.

**Bước 3:** Lưu lại nội dung của trang.

**Ví dụ:** Tạo một UCC chứa thông tin liên hệ của Trung tâm Hưng Yên Aptech, thông tin này sẽ được dùng làm Footer (chân trang) của tất cả các trang trong hệ thống phần mềm quản lý cán bộ.



Kết quả sau khi đưa UCC vào trang web.

ở đây ta cần tạo 2 trang, trang Footer.ascx chứa nội dung của UCC và trang Default.aspx, để sử dụng UCC Footer.ascx.

Nội dung trang Footer.ascx và Default.aspx như sau:

#### Footer.ascx

```
<%@ Control Language="C#"
    AutoEventWireup="true"
    CodeFile="Footer.ascx.cs"
    Inherits="Footer"
%>

<table border="0px" width="100%">
  <tr>
    <td align="center" style="font-style:italic" runat="server" id="NoiDung">
      <hr />
      Phần mềm quản lý cán bộ - phiên bản 2.0 <br />
      &copy;Bản quyền hệ thống thuộc về Trung tâm đào tạo Hưng Yên - Aptech 2008
      <br />
      Tel: 0321-713.179; E-Mail: aptech@utehy.edu.vn; website:

      <asp:LinkButton runat="server" ID="AptechLink"
        PostBackUrl="http://aptech.utehy.vn" Text="www.aptech.utehy.vn">
      </asp:LinkButton>

    </td>
  </tr>
</table>
```

#### Default.aspx

```
<%@ Page Language="C#"
    AutoEventWireup="true"
    CodeFile="Default.aspx.cs"
    Inherits="_Default"
%>

<%@ Register
    TagPrefix="Aptech"
    TagName="Footer"
    Src="~/Footer.ascx"
%>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Phần mềm quản lý cán bộ - Version 2.0</title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <Aptech:Footer ID="Footer1" runat="server" />
    </div>
  </form>
</body>
</html>
```



\*\*\* Một số điểm cần lưu ý khi tạo UCC:

- ❖ Trang UCC cũng có trang Code C# tương ứng là : **.ascx.cs**
- ❖ Đầu trang ascx, thay vì viết chỉ dẫn **<%@ Page...** ta thay bằng: **<%@ Control...**
- ❖ Trong UCC không có cặp thẻ HTML, BODY, FORM.
- ❖ Khi "chèn" UCC vào trang aspx, cần phải thêm thẻ **<%@ Register...** ngay sau thẻ **<%@ Page...** Tuy nhiên có thể dùng phương pháp kéo-thả bằng cách click vào trang ascx và "kéo-thả" vào form .aspx (ở chế độ Design view).
- ❖ Một UCC có thể xuất hiện nhiều lần trong một trang.
- ❖ Muốn thay đổi UCC trên các trang thì phải trở về trang ascx ban đầu để chỉnh sửa.

### 7.3 Thêm các thuộc tính, phương thức và sự kiện vào UCC

Mỗi UCC thực chất là một Class, do vậy hoàn toàn có thể thêm các thuộc tính, phương thức, sự kiện ..

#### 7.3.1 Thêm thuộc tính vào UCC

Để thêm thuộc tính, mở file chứa code (file có phần mở rộng là ascx.cs) và khai báo thuộc tính cần thiết.

**Ví dụ:** Thêm thuộc tính Mau\_Nen cho UCC Footer ở trên để đặt Màu nền cho dòng Footer.

##### Footer.ascx.cs

```
using System;
using System.Web.UI;

public partial class Footer : UserControl
{
    private string mau_nen;
    public string Mau_Nen
    {
        set
        {
            mau_nen=value;
            NoiDung.BgColor = mau_nen;
        }
        get
        {
            return mau_nen;
        }
    }

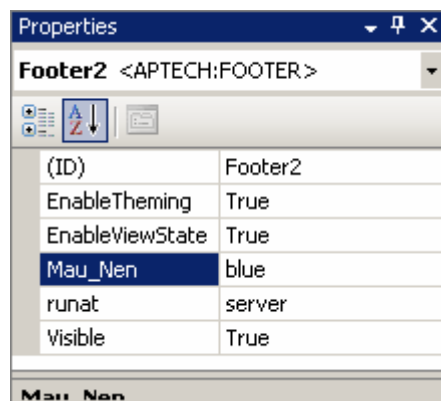
    protected void Page_Load (object sender, EventArgs e)
    {
    }
}
```

**Chú ý:** - **NoiDung** là ID của phần tử <TD> trong trang Footer.ascx. Do vậy viết **NoiDung.BgColor = mau\_nen**; trong đoạn chương trình trên là đặt màu nền cho phần tử <td>.

- Từ khóa **value** ở trên là giá trị được gán cho thuộc tính. Ví dụ nếu ta viết **NoiDung.Mau\_Nen="blue"** thì **value** khi này có giá trị là "blue".
- Mỗi khi ta gán giá trị cho thuộc tính thì phần bên trong set { ... } sẽ được gọi và khi ta đọc giá trị của thuộc tính thì phần get { } sẽ được gọi. Trong mỗi phần set và get này hoàn toàn có thể đặt các câu lệnh xử lý.

Sau khi thêm thuộc tính, bạn có thể đọc/ghi giá trị này thông qua câu lệnh hoặc gán giá trị trực tiếp trong chế độ code và design. Ví dụ đặt lại màu nền:

.....  
<Aptech:Footer ID="Footer1" runat="server" **Mau\_Nen="purple"**/>  
.....



Đặt trực tiếp trong cửa sổ Properties

### 7.3.2 Thêm phương thức vào UCC

Tương tự như các Class, chúng ta có thể thêm các phương thức vào cho lớp như khi xây dựng các lớp thông thường.

**Ví dụ:** Thêm một phương thức làm ẩn/ hiện UCC ở trên.

#### Footer.aspx.cs

```
using System;
using System.Web.UI;

public partial class Footer : UserControl
{
    private string mau_nen;
    public string Mau_Nen
    {
        set
        {
            mau_nen=value;
            NoiDung.BgColor = mau_nen;
        }
    }
}
```

```
        get
        {
            return mau_nen;
        }
    }

    /// Value= true ==> Hiện.
    /// Value=False ==> Ẩn
    public void ShowHideUCC (Boolean Value)
    {
        if (Value == true)
            this.Visible = true;
        else
            this.Visible = false;
    }

    protected void Page_Load (object sender, EventArgs e)
    {
    }
}
```

Việc gọi các phương thức này cũng giống như những phương thức thông thường trước đây.

### 7.3.3 Thêm sự kiện vào UC

Phần này sẽ được giới thiệu trong các chuyên đề cao hơn.

## 7.4 Truy cập thuộc tính, phương thức của các phần tử con trong UCC.

Vì một UCC có thể chứa nhiều điều khiển bên trong, vì vậy có những lúc ta cần truy cập đến một số thuộc tính, phương thức của điều khiển con này. Tuy nhiên điều này là không được phép, giải pháp cho vấn đề này đó là: Tạo thêm phương thức dạng Public hoặc Protected của UCC để có thể truy cập đến các phần tử con bên trong.

**Ví dụ:** Thêm một phương thức SetAptechLinkText cho phép thay đổi lại nhãn liên kết đến trang Aptech. Code của trang sẽ như sau:

#### Footer.ascx.cs

```
using System;
using System.Web.UI;

public partial class Footer : UserControl
{
    private string mau_nen;
    public string Mau_Nen
    {
        set
        {
            mau_nen=value;
            NoiDung.BgColor = mau_nen;
        }
        get
    }
}
```



```
    {
        return mau_nen;
    }
}

/// Value= true ==> Hiện.
/// Value=False ==> Ẩn
public void ShowHideUCC (Boolean Value)
{
    if (Value == true)
        this.Visible = true;
    else
        this.Visible = false;
}
```

```
/// Phương thức cho phép thay đổi nhãn của liên kết đến trang Aptech
public void SetAptechSiteText (string newText)
{
    AptechLink.Text = newText;
}
```

```
protected void Page_Load (object sender, EventArgs e)
{
}
}
```

Khi sử dụng:

**Trang Default.aspx.cs**

```
using System;
using System.Web.UI;

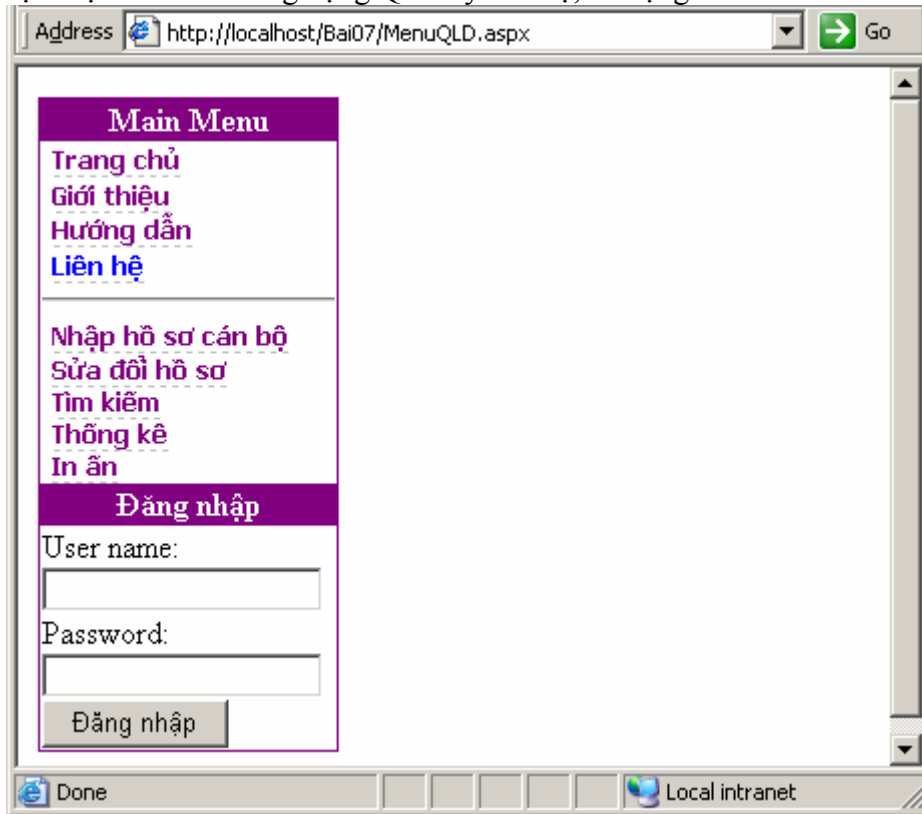
public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Footer1.SetAptechSiteText ("Trang chủ của Aptech");
    }
}
```

Nếu điều khiển con bên trong có rất nhiều thuộc tính (ví dụ Table) thì rõ ràng để truy cập tới các thuộc tính của nó (rất nhiều) ta sẽ phải tạo vô số thuộc tính cho UCC. Trong trường hợp như vậy, để tránh phải khai báo quá nhiều thuộc tính, ta tạo một thuộc tính và trả về là đối tượng cần truy xuất, ví dụ để bên ngoài có thể truy xuất đến toàn bộ các thuộc tính và phương thức của Textbox1 nằm trong UCC thì ta tạo một thuộc tính kiểu như sau cho UCC:

```
.....
public TextBox txtUserName {
    get {
        return TextBox1;
    }
}
```

## 7.5 Minh họa tạo một số điều khiển

**Ví dụ 1:** Tạo một menu cho ứng dụng Quản lý cán bộ, có dạng:



Kết quả

### File Menu.ascx

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="MainMenu.ascx.cs" Inherits="MainMenu" %>
```

```
<style type="text/css">
    .HLink
    {
        font-family:Tahoma;
        font-weight:bold;
        font-size:10pt;
        text-decoration:none;
        border-bottom-style:dotted;
        border-bottom-width:1px;
        border-bottom-color:Silver;
        margin:5px 2px 2px 5px;
        width:100%;
    }
</style>

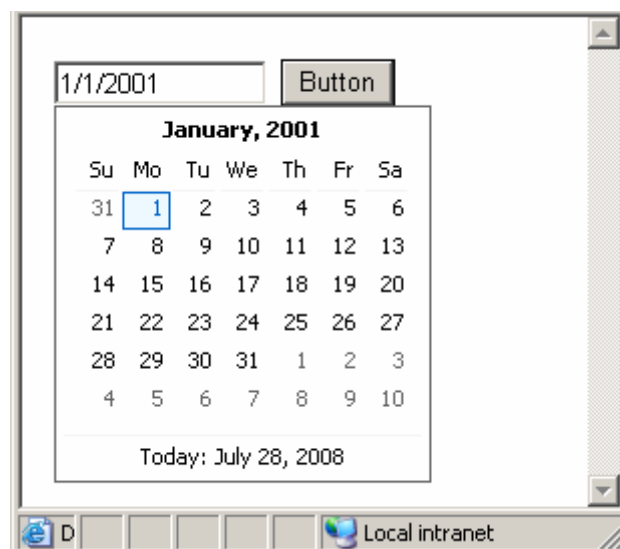
<table border="0px" style="border-collapse:collapse; border:solid 1px purple;width:150px">
    <tr>
        <td align="center" style="background-color:Purple; color:White;font-weight:bold">
            Main Menu
        </td>
    </tr>
</table>
```

```

<td>
    <asp:LinkButton runat="server" ID="InkHome" CssClass="HLink" Text="Trang chủ"
PostBackUrl="~/Default.aspx"></asp:LinkButton><br />
    <asp:LinkButton runat="server" ID="LinkButton1" CssClass="HLink" Text="Giới
thiệu"></asp:LinkButton> <br />
    <asp:LinkButton runat="server" ID="LinkButton2" CssClass="HLink" Text="Hướng
dẫn"></asp:LinkButton><br />
    <asp:LinkButton runat="server" ID="LinkButton8" CssClass="HLink" Text="Liên
hệ"></asp:LinkButton>
    <hr />
    <asp:LinkButton runat="server" ID="LinkButton" CssClass="HLink" Text="Nhập hồ sơ cán
bộ"></asp:LinkButton> <br />
    <asp:LinkButton runat="server" ID="LinkButton3" CssClass="HLink" Text="Sửa đổi hồ
sơ"></asp:LinkButton><br />
    <asp:LinkButton runat="server" ID="LinkButton4" CssClass="HLink" Text="Tìm
kiếm"></asp:LinkButton><br />
    <asp:LinkButton runat="server" ID="LinkButton5" CssClass="HLink" Text="Thống
kê"></asp:LinkButton><br />
    <asp:LinkButton runat="server" ID="LinkButton6" CssClass="HLink" Text="In
ấn"></asp:LinkButton><br />
</td>
<tr>
<td align="center" style="background-color:Purple; color:White;font-weight:bold">
    Đăng nhập
</td>
</tr>
<tr>
<td>
    User name: <br />
    <asp:TextBox runat="server" ID="txtUserName" Width="92%"></asp:TextBox> <br />
    Password: <br />
    <asp:TextBox runat="server" ID="txtPassword" Width="92%"></asp:TextBox>
    <asp:Button runat="server" ID="cmdLogin" Text="Đăng nhập" />
</td>
</tr>
</table>

```

**Ví dụ 2:** Tạo một textbox để nhập ngày tháng (có tích hợp Calendar) dùng chung trong toàn ứng dụng.



Kết quả.

#### File MyCalendar.ascx

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="MyCalendar.ascx.cs"
Inherits="MyCalendar" %>
```

```
<%@ Register assembly="AjaxControlToolkit"
namespace="AjaxControlToolkit" tagprefix="cc1" %>
```

```
<table>
  <tr>
    <td>
      <asp:TextBox runat="server" ID="txtNT" style="width:100px"></asp:TextBox>
    </td>
  </tr>
</table>

<cc1:CalendarExtender ID="CalendarExtender1" runat="server"
  TargetControlID="txtNT">
</cc1:CalendarExtender>
```

#### File MyCalendar.ascx.cs

```
using System;
using System.Web.UI;

public partial class MyCalendar : System.Web.UI.UserControl
{
    private string gt;
    public string GiaTri
    {
        set
        {
            gt = value;
            txtNT.Text = gt;
        }
        get
        {
            return gt;
        }
    }

    protected void Page_Load(object sender, EventArgs e)
    {
    }
}
```

## BÀI 8: THỰC HÀNH

### Mục tiêu:

Kết thúc bài thực hành này học viên có thể:

- ❖ Tạo một số điều khiển UCC phục vụ cho hệ thống Quản lý cán bộ
- ❖ Thêm và sử dụng được các thuộc tính, phương thức cho điều khiển UCC

### Nội dung:

**Bài 1:** Tạo một điều khiển Login, có giao diện như sau:



Chương trình minh họa:

#### **Trang Login.ascx**

```
<%@ Control Language="C#"
AutoEventWireup="true"
CodeFile="Login.ascx.cs" Inherits="Login" %>
```

```
<table runat="server" id="NoiDungLogin" style="border:solid 1px purple;border-collapse:collapse">
  <tr>
    <td colspan="2" style="text-align:center; background-color:purple;color:White">
      Đăng nhập
    </td>
  </tr>
  <tr>
    <td style="text-align:center;width:40%">User name:</td>
    <td>
      <asp:TextBox runat="server" ID="txtUserID" Width="99%"></asp:TextBox>
    </td>
  </tr>
  <tr>
    <td style="text-align:center; width:40%">Password:</td>
    <td>
      <asp:TextBox runat="server" ID="txtPassword" Width="99%" TextMode="Password">
      </asp:TextBox>
    </td>
  </tr>
  <tr>
    <td></td>
    <td>
      <asp:Button runat="server" ID="cmdCancel" Text="Cancel" />
      <asp:Button runat="server" ID="cmdLogin" Text="Login" />
    </td>
  </tr>
</table>
```

**Bài 2:** Tạo điều khiển Login1 có giao diện như bài 1 nhưng thêm thuộc tính cho phép đặt độ rộng của điều khiển.

**Hướng dẫn:** Phần giao diện code như trên, còn phần code C# sẽ thêm thuộc tính như sau:

**Trang Login.ascx.cs**

```
using System;
using System.Web.UI;

public partial class Login : System.Web.UI.UserControl
{
    private string dorong;
    public string DoRong
    {
        set
        {
            dorong = value;
            NoiDungLogin.Width = dorong;
        }
        get
        {
            return dorong;
        }
    }
    protected void Page_Load (object sender, EventArgs e)
    {

    }
}
```

Tại trang aspx, khi muốn thay đổi độ rộng, chỉ việc thêm thuộc tính DoRong trong thẻ.

**Bài 3:** Yêu cầu như bài 2 nhưng có thêm 3 phương thức. Phương thức CheckAccout() để kiểm tra, nếu User name="asp.net" và password = "123456" thì trả về true, trái lại trả về false; Phương thức GetUserName trả về giá trị trong ô User name; phương thức GetPassword trả về giá trị trong ô Password.

Ngoài ra, khi người dùng bấm vào nút "Login" thì sẽ in ra thông báo (trên một nhãn phía dưới) : "Bạn đã đăng nhập với User name là : ... Password là: ....."

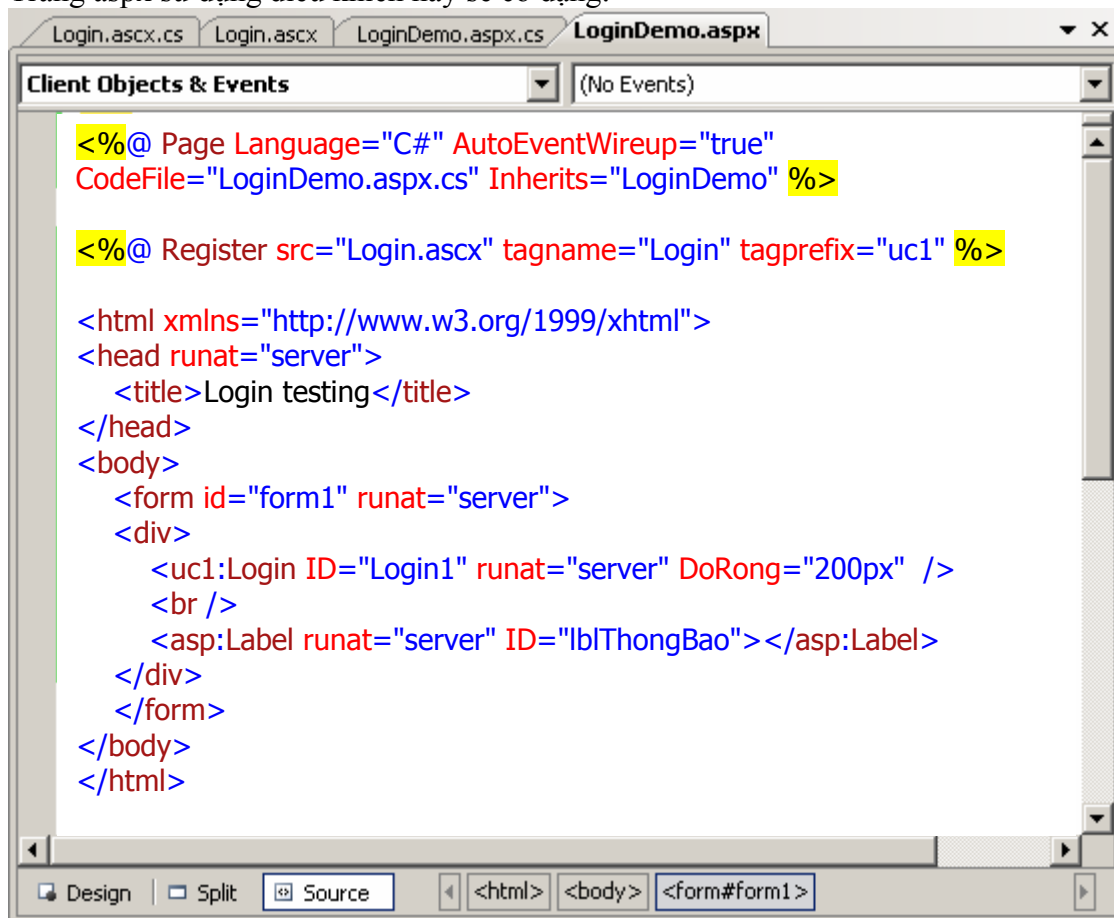
**Hướng dẫn:** Vì bên ngoài không thể truy xuất được vào các thuộc tính của đối tượng con thuộc UCC, do vậy nếu thực sự muốn truy cập thì ta cần phải xây dựng các phương thức Public cung cấp các chức năng truy cập (đọc/ ghi) này.

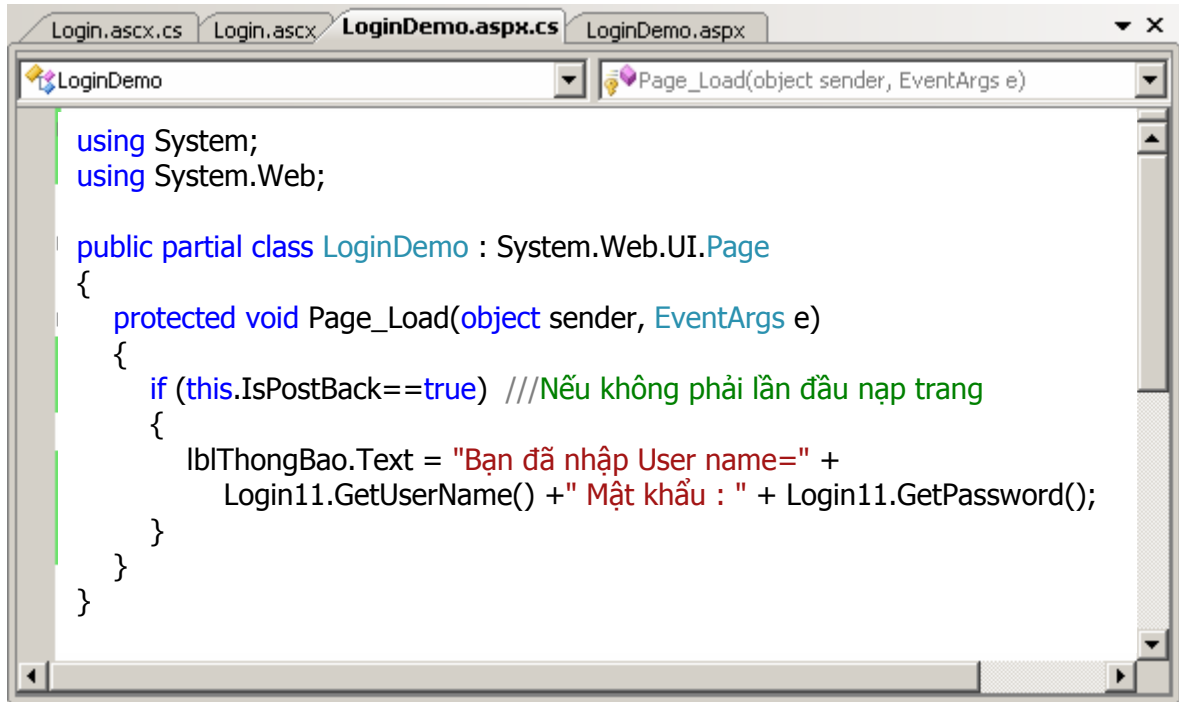
**Viết lại trang Login.ascx.cs như sau (Phần giao diện không đổi)**

```
using System;
public partial class Login : System.Web.UI.UserControl
{
    private string dorong;
    public string DoRong
    {
        set
        {
            dorong = value;
            NoiDungLogin.Width = dorong;
        }
    }
}
```

```
    }
    get
    {
        return dorong;
    }
}
public Boolean CheckAccount ()
{
    return (txtUserID.Text == "asp.net" && txtPassword.Text == "123456");
}
/// Lấy User name trong ô User name
public string GetUserName ()
{
    return txtUserID.Text;
}
/// lấy Password trong ô Password
public string GetPassword ()
{
    return txtPassword.Text;
}
protected void Page_Load (object sender, EventArgs e)
{
}
}
```

Trang aspx sử dụng điều khiển này sẽ có dạng:





Nội dung trang C# của trang ASPX

**Bài 4:** Tạo phần Header và Footer để có thể chèn vào các trang. Giao diện như sau:



Nội dung của điều khiển Header.ascx (Học viên có thể tùy biến)

Tạo một UCC, có tên **Header.ascx** như sau:

```
<%@ Control Language="C#" AutoEventWireup="true"
    CodeFile="Header.ascx.cs" Inherits="Header"
%>
```

```
<style type="text/css">
```

```
A
{
    float:right;
    width:80px;
    text-decoration:none;
    color:white;
    background-color:purple;
    border-right:1px solid white;
    text-align:center;
}
```

```
A:Hover
{
    background-color:#ff3300;
    color:Purple;
}
```



```
}
</style>

<table border="0px" style="width:790px;border-collapse:collapse">
  <tr style="height:80px;background-image:url('body_bg.gif')">
    <td style="text-align:center;
      vertical-align:middle;
      font-family:compact;
      font-size:20pt;
      font-weight:bold">
      HUMAN RESOURCE MANAGEMENT SYSTEM - V2.0
    </td>
  </tr>

  <tr style="height:30pt">
    <td style="text-align:right;vertical-align:top;
      font-weight:bold;text-decoration:none; color:White">
      <a href="LoginDemo.aspx">Login</a>
      <a href="Products.aspx">Products</a>
      <a href="Downloads.aspx">Download</a>
      <a href="Contact.aspx">Contact</a>
      <a href="Forums.aspx">Forum</a>
    </td>
  </tr>
</table>
```

**Bài 5:** Tạo Menu cho hệ thống phần mềm quản lý cán bộ.

(Làm lại bài trong tài liệu dùng cho lý thuyết). Nhưng thêm một số hyperlink khác..

**Bài 6:** Lắp ghép header, footer và Menu để được trang như sau:

HUMAN RESOURCE MANAGEMENT SYSTEM - V2.0				
		Forum	Contact	Download
		Products	Login	
<div><div><div>Main Menu</div><div><a href="#">Trang chủ</a> <a href="#">Giới thiệu</a> <a href="#">Liên hệ</a> <a href="#">Nhập hồ sơ cán bộ</a> <a href="#">Sửa đổi hồ sơ</a> <a href="#">Tìm kiếm</a> <a href="#">Thống kê</a> <a href="#">In ấn</a></div></div><div><div>Đăng nhập</div><div>User name: <input type="text"/></div><div>Password: <input type="password"/></div><div>Đăng nhập</div></div></div>				
<hr/>				
<p>Phần mềm quản lý cán bộ - phiên bản 2.0 ©Bản quyền thuộc về Trung tâm đào tạo Hưng Yên - Aptech 2008 Tel: 0321-713.179; E-Mail: <a href="mailto:aptech@utehy.edu.vn">aptech@utehy.edu.vn</a>; website: <a href="http://www.aptech.utehy.vn">www.aptech.utehy.vn</a></p>				

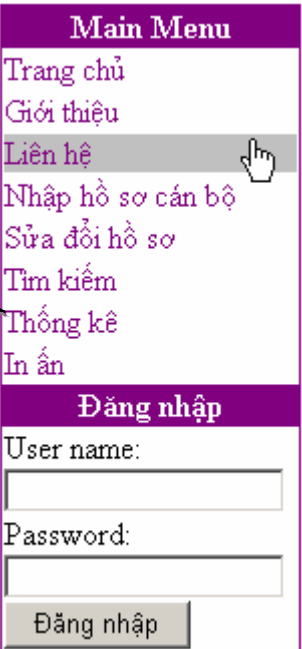
Trang code giao diện của UCC **MainMenu.ascx.cs**

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="MainMenu.ascx.cs"
Inherits="MainMenu" %>
```

```
<style type="text/css">
.A_MainMenu
{
    float:none;
    width:100px;
    text-decoration:none;
    color:purple;
    text-align:left;
}
```

```
A:Hover
{
    background-color:silver;
    color:Purple;
}
</style>
```

Giao diện UCC của  
**MainMenu.ascx**



```
<table border="0px" style="border-collapse:collapse;
border:solid 1px purple;width:150px">
<tr>
<td align="center" style="background-color:Purple; color:White;font-weight:bold">
Main Menu
</td>
</tr>
<tr>
<td>
<asp:LinkButton Width="100%" CssClass="A_MainMenu" runat="server"
ID="InkHome" Text="Trang chủ" PostBackUrl="~/HeaderDEMO.aspx">
</asp:LinkButton>
</td>
</tr>
<tr>
<td>
<asp:LinkButton Width="100%" CssClass="A_MainMenu"
runat="server" ID="LinkButton1" Text="Giới thiệu"></asp:LinkButton>
</td>
</tr>
<tr>
<td>
<asp:LinkButton Width="100%" CssClass="A_MainMenu"
runat="server" ID="LinkButton8" Text="Liên hệ"></asp:LinkButton>
</td>
</tr>
<tr>
```

```
<td>
    <asp:LinkButton Width="100%" CssClass="A_MainMenu"
        runat="server" ID="LinkButton" Text="Nhập hồ sơ cán bộ"></asp:LinkButton>
</td>
</tr>

<tr>
<td>
    <asp:LinkButton Width="100%" CssClass="A_MainMenu"
        runat="server" ID="LinkButton3" Text="Sửa đổi hồ sơ"></asp:LinkButton>
</td>
</tr>

<tr>
<td>
    <asp:LinkButton Width="100%" CssClass="A_MainMenu"
        runat="server" ID="LinkButton4" Text="Tìm kiếm"></asp:LinkButton>
</td>
</tr>

<tr>
<td>
    <asp:LinkButton Width="100%" CssClass="A_MainMenu"
        runat="server" ID="LinkButton5" Text="Thống kê"></asp:LinkButton>
</td>
</tr>

<tr>
<td>
    <asp:LinkButton Width="100%" CssClass="A_MainMenu"
        runat="server" ID="LinkButton6" Text="In ấn"></asp:LinkButton>
</td>
</tr>

<tr>
<td align="center" style="background-color:Purple; color:White;font-weight:bold">
    Đăng nhập
</td>
</tr>
<tr>
<td>
    User name: <br />
    <asp:TextBox runat="server" ID="txtUserName" Width="92%">
    </asp:TextBox> <br />
    Password: <br />
    <asp:TextBox runat="server" ID="txtPassword" Width="92%"></asp:TextBox>
    <asp:Button runat="server" ID="cmdLogin" Text="Đăng nhập" />
</td>
</tr>
</table>
```

## BÀI 9: Các đối tượng trong ASP.NET

Trong bất kỳ ứng dụng nào, dù là winform based hay webform based thì việc giao tiếp (tương tác) với người dùng và giữa các webform với nhau là điều bắt buộc. Ví dụ ta cần phải lấy thông tin đặt hàng do người dùng nhập vào và hiển thị trở lại người dùng một số thông tin hữu ích khác, như kết quả thanh toán... hay một trang chuyển tiếp kết quả cho một trang khác để xử lý v.v...

Ở các bài trước, để làm điều này chúng ta thực hiện dễ dàng thông qua các server controls như textbox, listbox, dropdownlist, label,... Tuy nhiên những điều khiển này chỉ có tác dụng trong một Page còn các trang khác thì hoàn toàn không thể đọc/ghi giá trị nằm trong các điều khiển này.

Để thực hiện việc giao tiếp (truyền dữ liệu) giữa các webform ASP.NET cung cấp một tập các điều khiển giúp ta làm việc đó một cách dễ dàng, đó là: Đối tượng Request và đối tượng Response.

Trong bài học này, chúng ta cũng tìm hiểu thêm một số đối tượng khác cũng rất hay dùng khi xây dựng ứng dụng là đối tượng Server, Application và Session.

### 9.1 Request Object

#### 9.1.1 Đối tượng Request dùng để làm gì ?

Request là một đối tượng của ASP.NET, nó cho phép đọc các thông tin do các trang khác gửi (Submit) đến.

Mô hình gửi/đọc giá trị:

**Trang Gửi**

**Post**

**Trang nhận**

**Form1**

User name :	<input type="text" value="aptech"/>
Password:	<input type="password" value="*****"/>

.....  
**Request("txtUserName") => aptech**  
**Request("txtPassword") => 123456**  
.....

#### 9.1.2 Các thành phần (thuộc tính và phương thức) chính

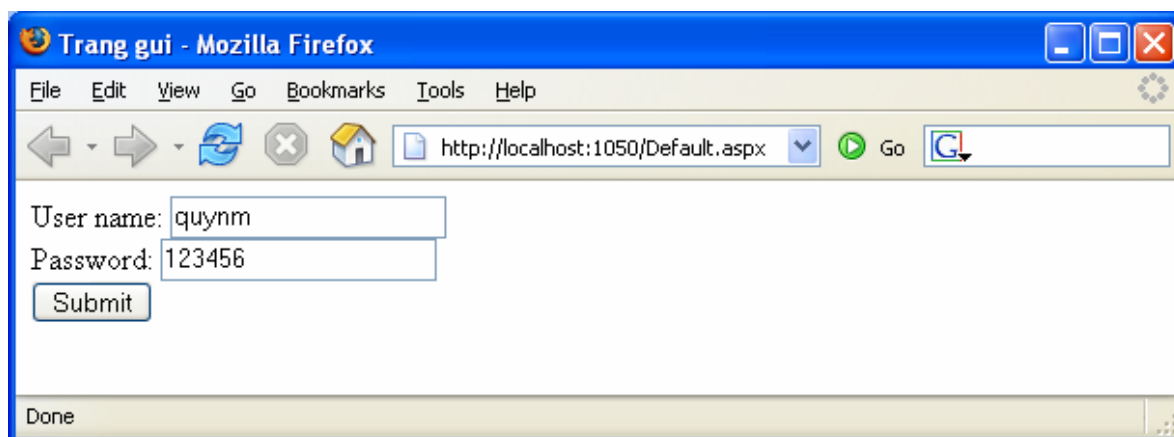
Phương thức:

- ❖ Request.QueryString.Get("Tên\_Phần tử cần đọc"): Để đọc giá trị của một phần tử được gửi theo phương thức Get (Method = "Get")
- ❖ Phương thức Request.Form.Get("Tên\_Phần tử cần đọc"): Để đọc giá trị của một phần tử được gửi theo phương thức Post (Method = "Post").

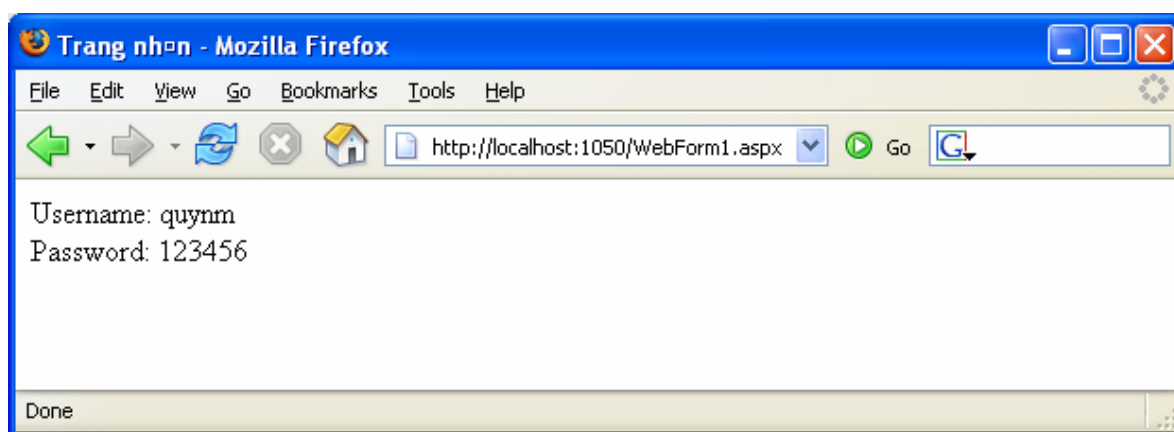
**Chú ý:** Có thể dùng Request.Form.GetValues và Request.Form.GetValues để đọc.

#### 9.1.3 Ví dụ sử dụng

Xây dựng 2 trang web : trang Default.aspx, trong đó có 2 textbox chứa tên và mật khẩu. Khi người dùng click vào nút submit thì gửi tên và mật khẩu sang trang Webform1.aspx để hiển thị.

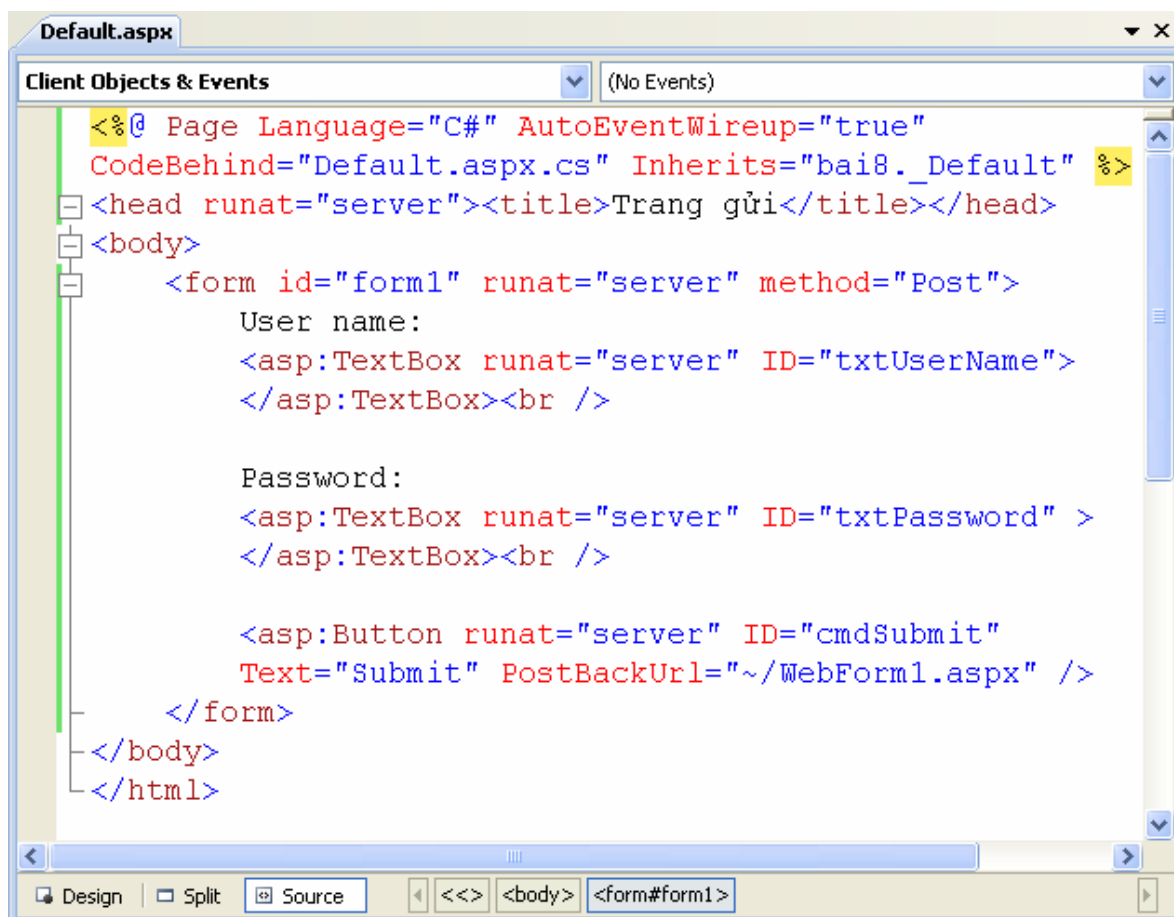


Trang nguồn (gửi): Default.aspx

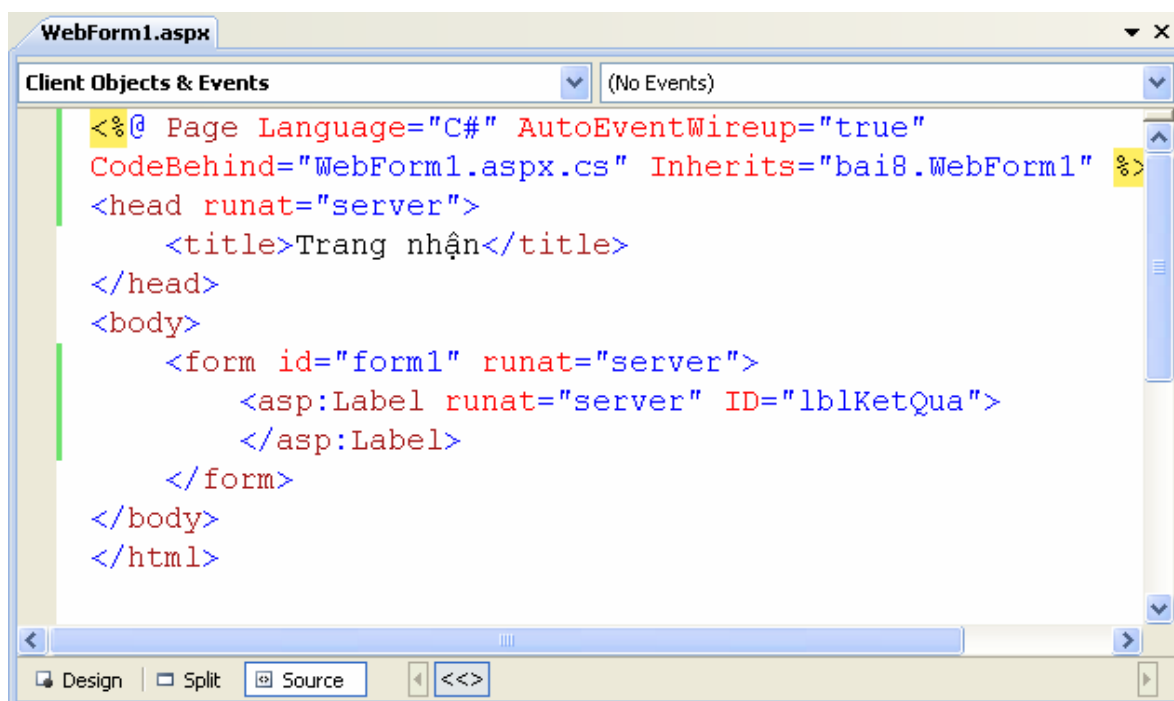


Kết quả nhận về.

Code của 2 trang sẽ như sau:



Default.aspx



Webform1.aspx



Code xử lý của trang webform1.aspx.cs

## 9.2 Response Object

### 9.1.1 Đối tượng Response dùng để làm gì ?

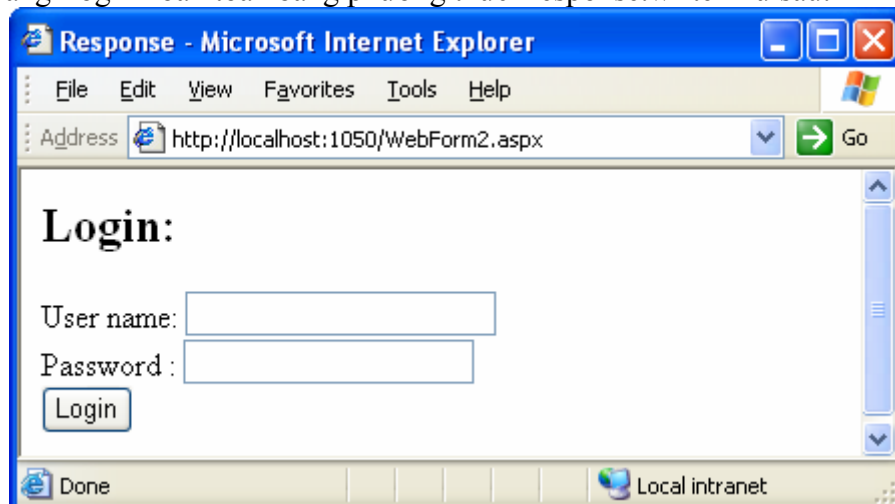
Đối tượng này được dùng để gửi nội dung (một chuỗi) bất kỳ về cho trình duyệt.

### 9.1.2 Các thành phần (thuộc tính và phương thức) chính

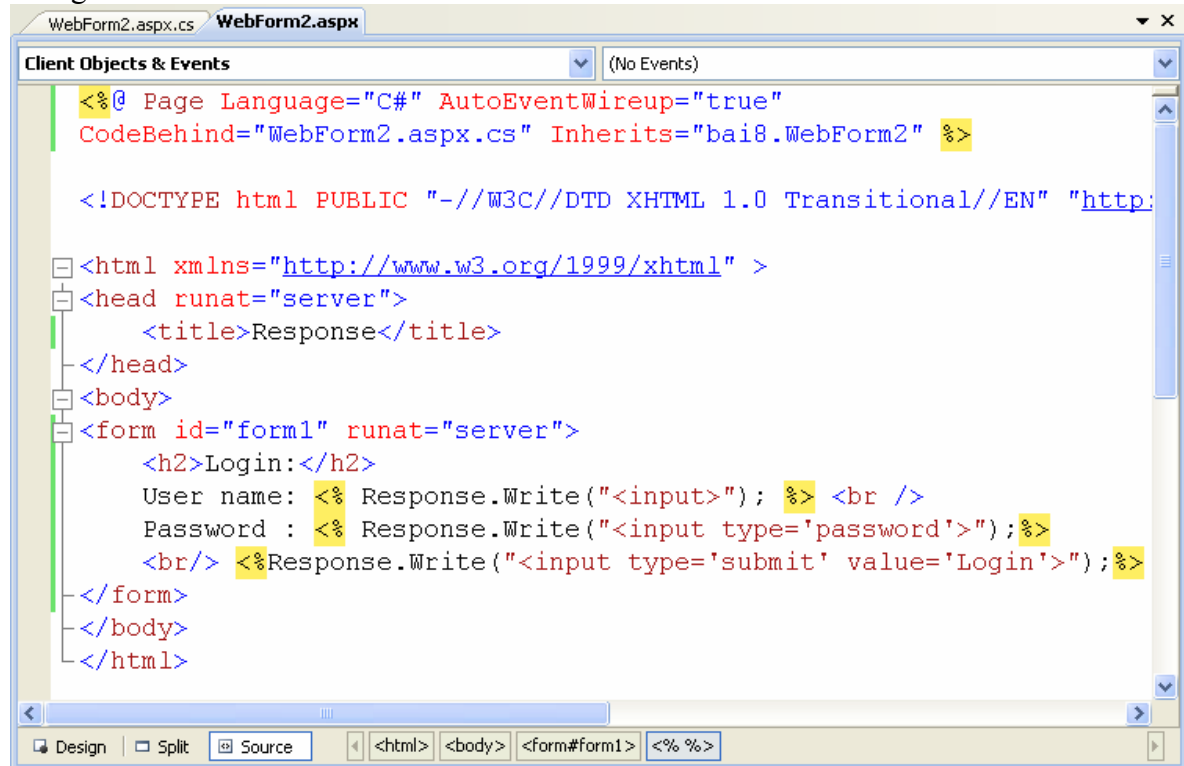
- ❖ Phương thức: Response.write(<Biểu thức>) dùng để gửi giá trị biểu thức truyền vào cho phía trình duyệt.
- ❖ Phương thức: Flush dùng để đưa dữ liệu còn trong bộ đệm phía server về cho phía trình duyệt.
- ❖ Phương thức Response.Redirect("địa chỉ URL"): Chuyển tới một trang khác.

### 9.1.3 Ví dụ sử dụng

Tạo một trang Login hoàn toàn bằng phương thức Response.write như sau:



Trang code sẽ như sau:



```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm2.aspx.cs" Inherits="bai8.WebForm2" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http:

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
<title>Response</title>
</head>
<body>
<form id="form1" runat="server">
<h2>Login:</h2>
User name: <% Response.Write("<input>"); %> <br />
Password : <% Response.Write("<input type='password'>"); %>
<br/> <%Response.Write("<input type='submit' value='Login'>"); %>
</form>
</body>
</html>
```

## 9.3 Server Object

### 9.3.1 Đối tượng Server dùng để làm gì ?

- ❖ Dùng để tạo các đối tượng COM
- ❖ Lấy thông tin về tên máy
- ❖ Ánh xạ đường dẫn ảo thành đường dẫn vật lý.

### 9.3.2 Các thành phần (thuộc tính và phương thức) chính

- ❖ CreateObject("COM Specification") → Ít dùng trong ứng dụng .NET
- ❖ MachineName: String; Trả về tên của máy tính server đang chạy.
- ❖ Mappath("Virtual path"): Trả về đường dẫn vật lý của đường dẫn ảo tương ứng.

### 9.3.3 Ví dụ sử dụng

- ❖ In ra tên của máy chủ hiện hành: Response.Write(Server.MachineName);
- ❖ Cho biết đường dẫn thực sự trên ổ cứng (thư mục vật lý) của trang hiện hành (trang default.aspx) : **Server.Mappath("default.aspx");**
- ❖ Cho biết đường dẫn vật lý ứng với tệp QLCB.Mdb, biết rằng tệp này nằm trong một thư mục con là "App\_Data": **Server.Mappath("App\_Data/QLDB.MDB");**

## 9.4 Session Object

### 9.4.1. Biến Session

Khi vào một website, người dùng có thể duyệt rất nhiều trang web của website đó. Nếu muốn lưu trữ thông tin về khách thăm này trong cả phiên làm việc thì có thể lưu vào các biến, gọi là biến Session.



Nói cách khác, biến session là một biến mà mọi trang trong một phiên (Session) đều có thể truy xuất.

#### 9.4.2. Đối tượng Session

Là đối tượng dùng để quản lý (tạo, đọc, ghi) các biến session và một số thông số khác.

+ Cú pháp để **tạo biến** Session như sau:

```
Session.Add("Tên_Biến", "Giá trị khởi tạo");
```

Lưu ý: Tên biến phải đặt trong cặp dấu nháy kép. <Giá trị> có thể là xâu ký tự hoặc số...

Ví dụ : Tạo một biến tên là MaNguoiDung và gán giá trị là TK34

```
Session.Add("MaNguoiDung", "TK34");
```

+ Cú pháp để đọc giá trị của một biến session như sau:

```
Session.Contents["Tên_Biến"] hoặc dùng chỉ số: Session.Contents[i];
```

+ Cú pháp để ghi (thay đổi) giá trị của biến session:

```
Session.Contents["Tên_Biến"] = <Giá trị mới>
```

Ví dụ:

```
Response.write("Mã người dùng là : " & Session.Contents["MaNguoiDung"])
```

Riêng với đối tượng Session, nó còn có các sự kiện. Các sự kiện này tự động được gọi mỗi khi một phiên làm việc được tạo ra. Các sự kiện này có tên là On\_Start và On\_End. Các sự kiện này được đặt trong file Global.asax.

### 9.5 Application Object

#### 9.5.1 Đối tượng Application dùng để làm gì ?

Dùng để quản lý các biến có phạm vi toàn ứng dụng. Có tác dụng đến mọi người dùng.

#### 9.5.2. Khái niệm biến toàn ứng dụng

Biến toàn ứng dụng là biến có tác dụng đối với mọi người dùng truy cập vào website. Mọi trang aspx.cs đều có thể truy cập đến biến này và dù ở bất kỳ thời điểm nào.

#### 9.5.3. Đối tượng Application

Dùng để quản lý (Tạo, đọc, ghi) các biến có phạm vi toàn ứng dụng.

+ Cú pháp tạo biến Application:

```
Application.Add ("Tên_Biến", <Giá trị khởi tạo>);
```

+ Ví dụ: Tạo biến So\_Nguoi\_Truy\_Cap

```
Application.Add("So_Nguoi_Truy_Cap", 0)
```

+ Truy xuất đến biến Application:

```
Application.Contents["Tên_Biến"] hoặc chỉ số: Application.Contents[i]
```

+ Ví dụ : Đọc và ghi biến Application.

```
Application.Contents["So_Nguoi_Truy_Cap"] =  
Application.Contents["So_Nguoi_Truy_Cap"] + 1
```

```
Response.write("Bạn là vị khách thứ: " &  
Application.Contents["So_Nguoi_Truy_Cap"])
```

Ngoài ra, đối tượng Application còn có 2 phương thức thường dùng là **Application.Lock()**: Để khóa không cho người khác sửa đổi các biến toàn cục và **Application.Unlock()** để mở khóa .

Đối tượng Application cũng có 2 sự kiện đó là Application\_OnStart và Application\_OnEND. Sự kiện OnStart chỉ được kích hoạt duy nhất một lần khi yêu cầu đầu tiên phát sinh. Sự kiện OnEND được kích hoạt khi dịch vụ web dừng (unload).

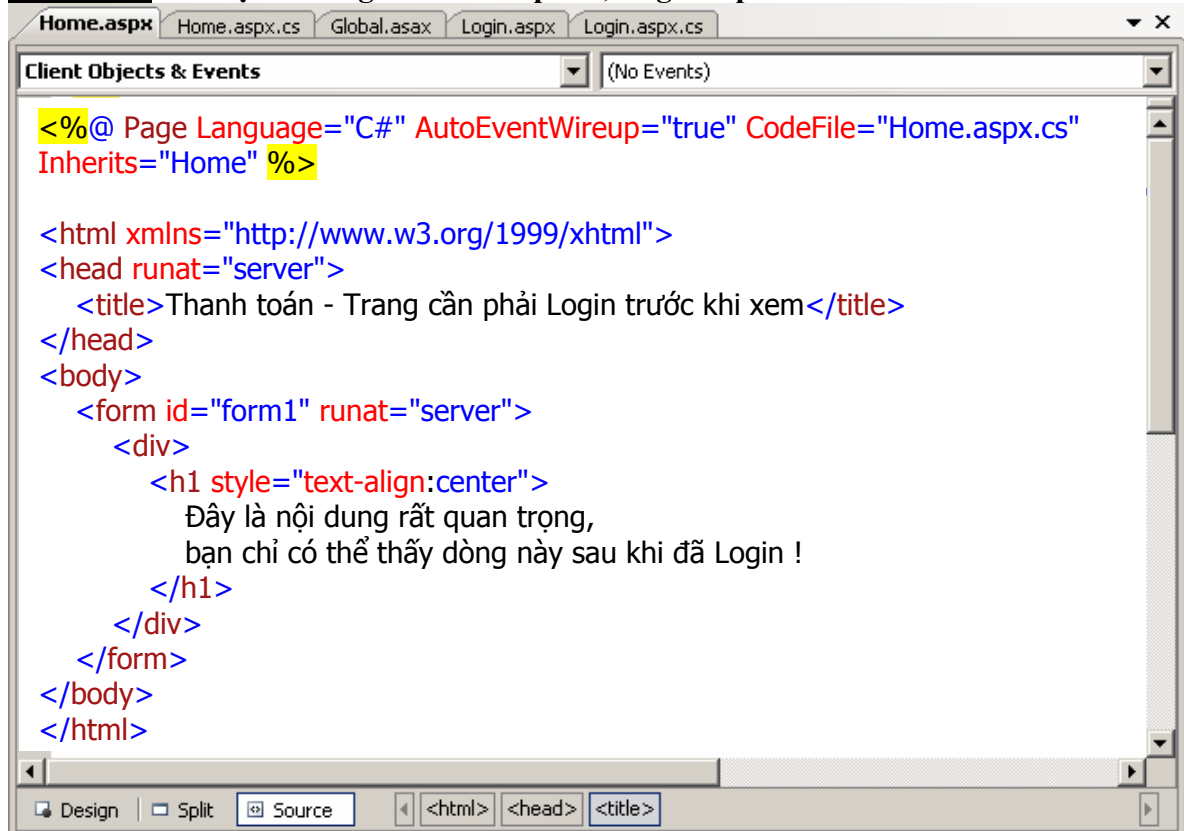
Đối tượng Application có 2 phương thức là Lock và Unlock. Khi gọi phương thức Lock (khóa) thì tất cả các ứng dụng không được phép thay đổi các giá trị Application. Để các ứng dụng khác được phép thay đổi các biến Application thì gọi phương thức Unlock.

Mã lệnh viết cho 2 sự kiện này cũng được đặt trong file Global.asa.

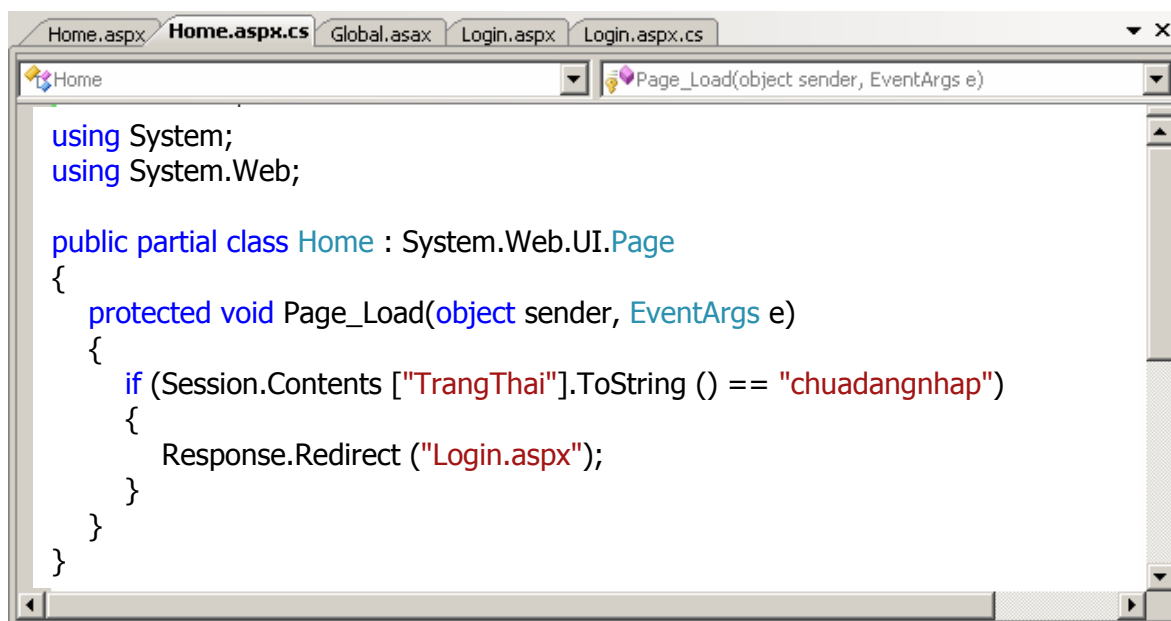
## Một số bài tập tổng hợp:

**Bài 1:** Tạo một trang Login, nếu người dùng nhập user name và mật khẩu tương ứng là **asp.net** và **123456** thì được phép truy cập các trang **Index.aspx**, trái lại mỗi lần người dùng truy cập đến trang Index.aspx thì đều được chuyển tới trang Login.aspx.

**Minh họa:** Cần tạo 3 trang là Home.aspx/cs, Login.aspx/cs và Global.asax như sau:



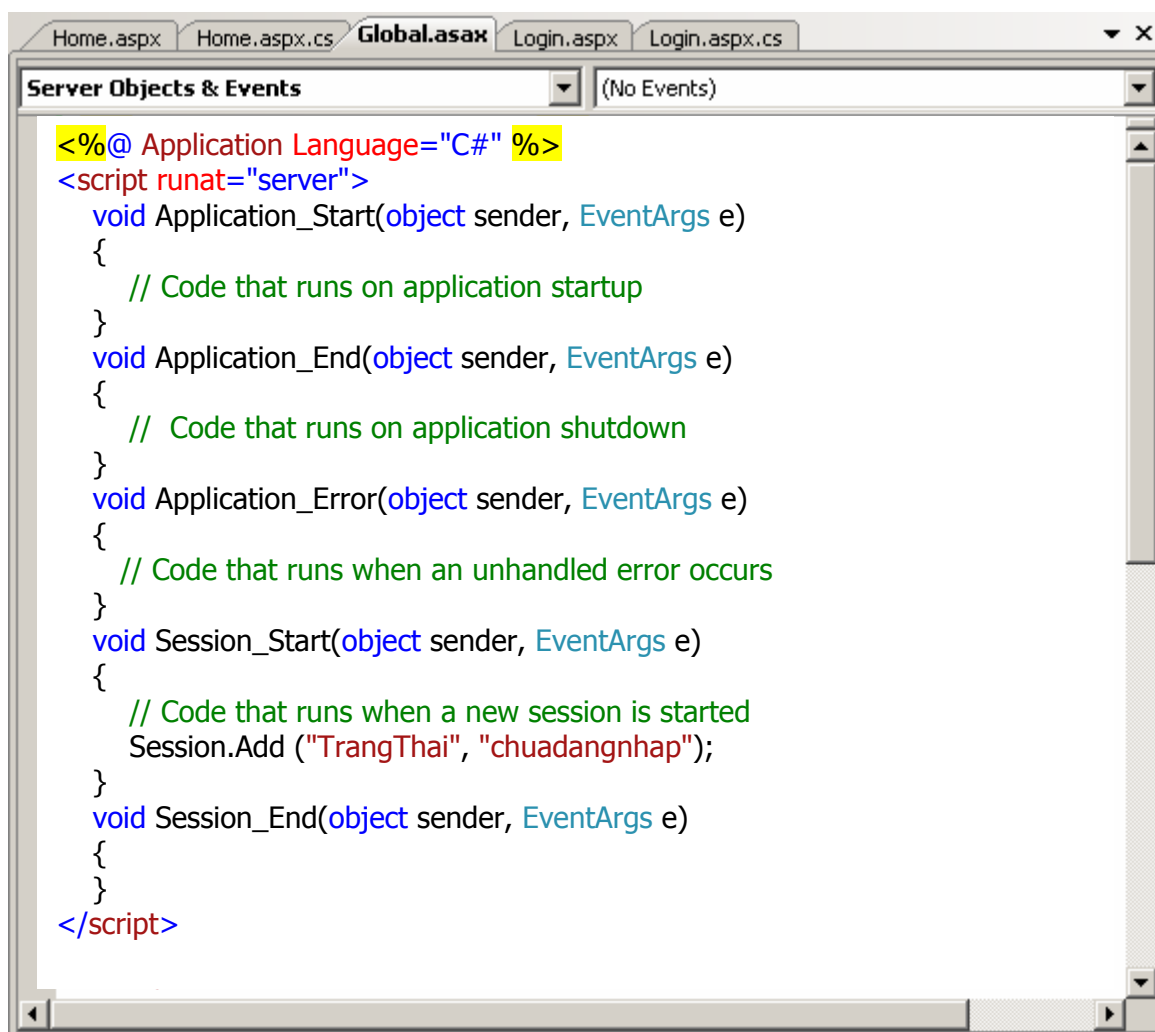
Nội dung trang Home.aspx



```
using System;
using System.Web;

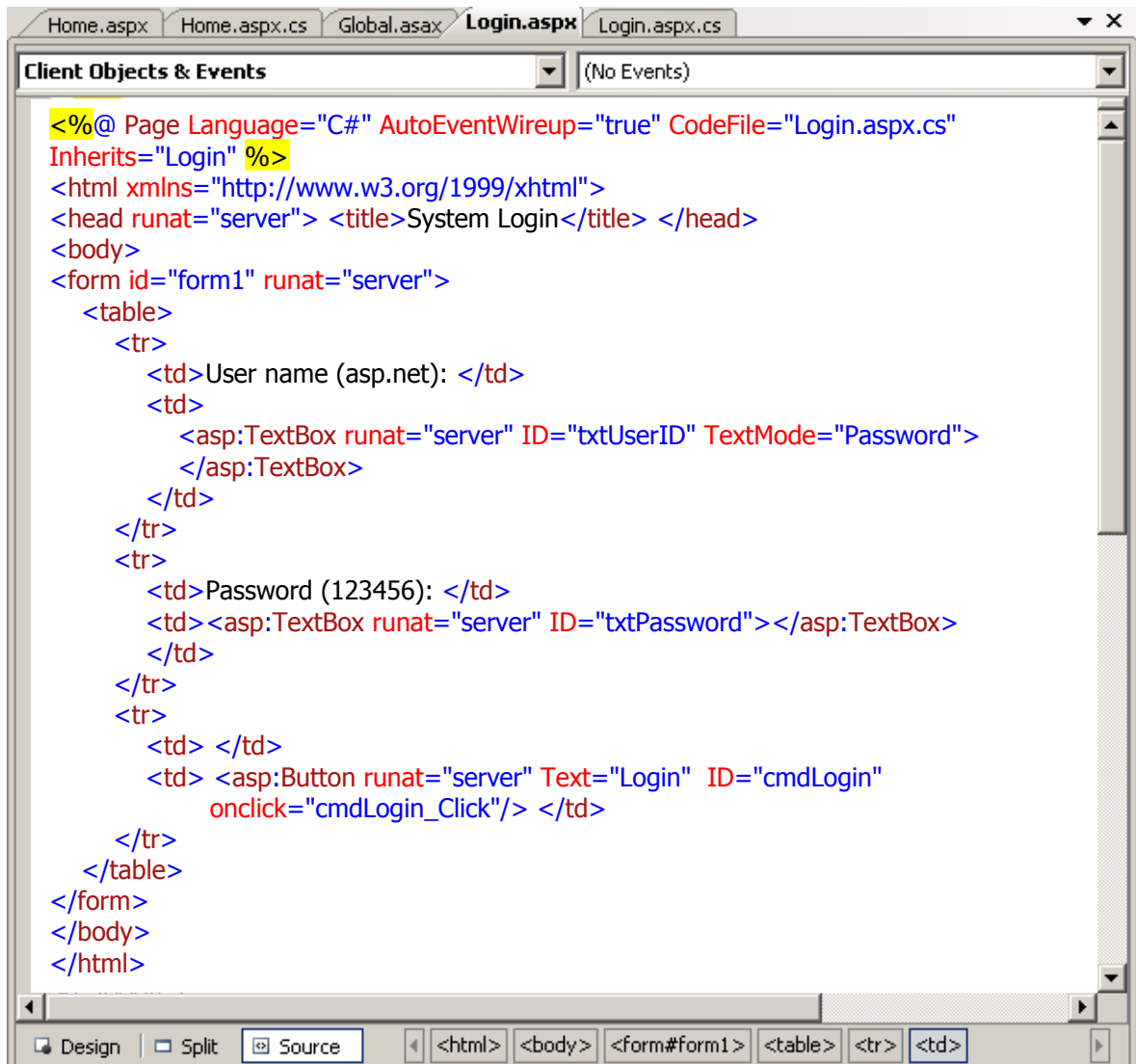
public partial class Home : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (Session.Contents ["TrangThai"].ToString () == "chuadangnhap")
        {
            Response.Redirect ("Login.aspx");
        }
    }
}
```

Trang Home.aspx.cs



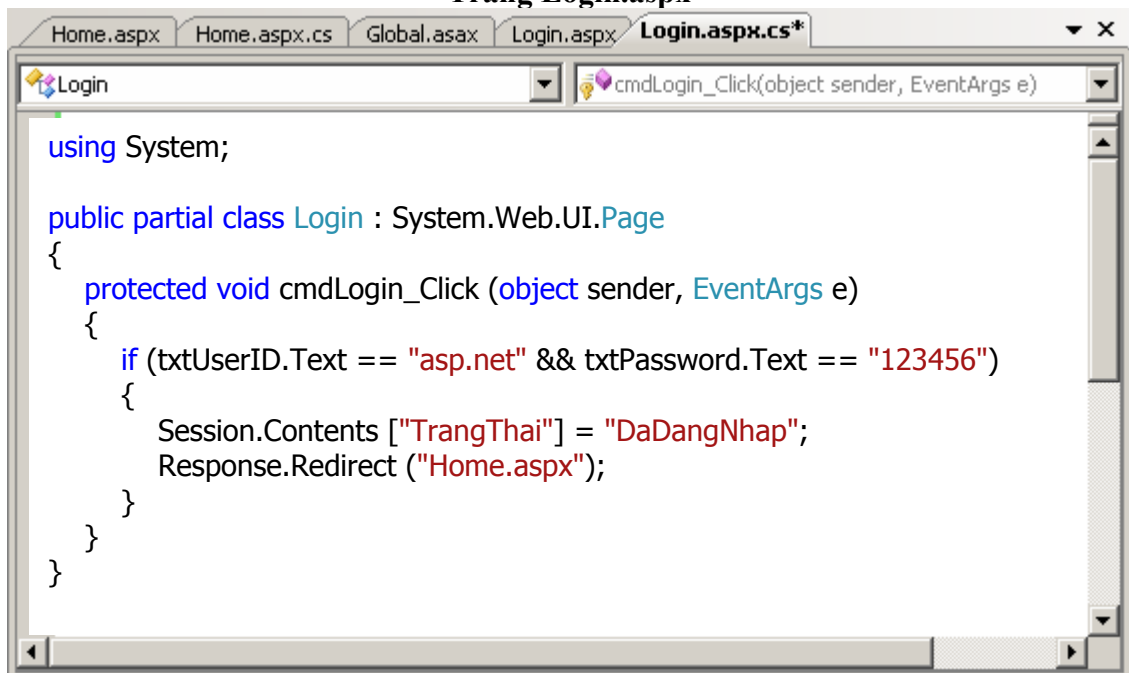
```
<%@ Application Language="C#" %>
<script runat="server">
    void Application_Start(object sender, EventArgs e)
    {
        // Code that runs on application startup
    }
    void Application_End(object sender, EventArgs e)
    {
        // Code that runs on application shutdown
    }
    void Application_Error(object sender, EventArgs e)
    {
        // Code that runs when an unhandled error occurs
    }
    void Session_Start(object sender, EventArgs e)
    {
        // Code that runs when a new session is started
        Session.Add ("TrangThai", "chuadangnhap");
    }
    void Session_End(object sender, EventArgs e)
    {
    }
</script>
```

Trang Global.asax



```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Login.aspx.cs"
Inherits="Login" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server"> <title>System Login</title> </head>
<body>
<form id="form1" runat="server">
  <table>
    <tr>
      <td>User name (asp.net): </td>
      <td>
        <asp:TextBox runat="server" ID="txtUserID" TextMode="Password">
        </asp:TextBox>
      </td>
    </tr>
    <tr>
      <td>Password (123456): </td>
      <td><asp:TextBox runat="server" ID="txtPassword"></asp:TextBox>
      </td>
    </tr>
    <tr>
      <td></td>
      <td><asp:Button runat="server" Text="Login" ID="cmdLogin"
onclick="cmdLogin_Click"/> </td>
    </tr>
  </table>
</form>
</body>
</html>
```

Trang Login.aspx



```
using System;

public partial class Login : System.Web.UI.Page
{
  protected void cmdLogin_Click (object sender, EventArgs e)
  {
    if (txtUserID.Text == "asp.net" && txtPassword.Text == "123456")
    {
      Session.Contents ["TrangThai"] = "DaDangNhap";
      Response.Redirect ("Home.aspx");
    }
  }
}
```

Trang Login.aspx.cs

**Bài 2:** Tạo một trang đếm số lượng người truy cập. Dùng biến tệp text để lưu.

Hướng dẫn: Tạo 2 trang là Index.aspx/cs và Global.asax với nội dung sau:

#### Trang Index.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Index.aspx.cs"
Inherits="Index" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Home Page - Hit counter</title>
</head>
<body>
    <form id="form1" runat="server">
        <h1>Chào mừng bạn đã đến website của chúng tôi</h1>
        <asp:Label runat="server" ID="lblSLKhach"></asp:Label>
    </form>
</body>
</html>
```

#### Trang Index.aspx.cs

```
using System;
public partial class Index : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        lblSLKhach.Text="Bạn là vị khách thứ: " +
            Application.Contents["SLTruyCap"].ToString();
    }
}
```

#### Trang Global.asax

```
<%@ Application Language="C#" %>

<script runat="server">
    void Application_Start(object sender, EventArgs e)
    {
        Application.Lock ();

        System.IO.StreamReader sr;
        sr = new System.IO.StreamReader (Server.MapPath ("SL.txt"));
        string S = sr.ReadLine ();
        sr.Close ();

        Application.UnLock ();

        //Tạo một biến Applciation là SLTruyCap và khởi tạo giá trị S
        Application.Add ("SLTruyCap", S);
    }
}
```

```
void Application_End(object sender, EventArgs e)
{
    // Code that runs on application shutdown
}
void Application_Error(object sender, EventArgs e)
{
    // Code that runs when an unhandled error occurs
}
void Session_Start(object sender, EventArgs e)
{
    //Tăng số lượng người truy cập lên 1 khi có một người mới thăm
    Application.Contents ["SLTruyCap"] =
        int.Parse (Application.Contents ["SLTruyCap"].ToString ()) + 1;

    //Lưu vào file SL.txt (mở và ghi đè)
    System.IO.StreamWriter sw;
    sw = new System.IO.StreamWriter (Server.MapPath ("SL.txt"));
    sw.Write (Application.Contents ["SLTruyCap"].ToString ());
    sw.Close ();
}
void Session_End(object sender, EventArgs e)
{
}
</script>
```

Sau khi tạo, chạy file Index.aspx để kiểm chứng sẽ thấy rằng số lượng người truy cập luôn luôn tăng lên bất kể là server có tắt hay máy tính bị trục trặc.

Đây là cách được dùng chính thức để đếm số lượng lượt người truy cập. Bạn hoàn toàn có thể cải tiến để hiển thị số lượng người truy cập bằng hình ảnh cho sinh động hơn.

## BÀI 10: THỰC HÀNH

### BÀI 11. Truyền dữ liệu giữa các webpage, MasterPage và gỡ rối (Debug) chương trình.

#### 11.1 Truyền (Post) dữ liệu giữa các trang bằng mã lệnh C#

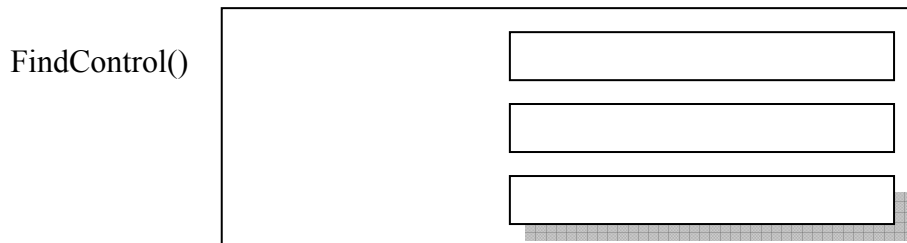
Như bài trước đã đề cập, để truyền (Post) dữ liệu từ một trang X đến trang Y nào đó ta thường tạo một Button trong đó có thuộc tính `PostBackURL = Y`. Với cách này thì mỗi khi chúng ta click chuột lên Button thì nó sẽ chuyển thẳng đến trang Y mà không phụ thuộc vào mã lệnh xử lý bên trong. Điều này sẽ bất lợi nếu như việc chuyển tới trang Y còn tùy thuộc vào kết quả xử lý hiện tại.

Ví dụ:

Tạo một trang có một textbox và một nút nhấn. Khi người dùng click nút nhấn (submit) thì kiểm tra nếu rỗng thì thông báo sai, trái lại thì Post tới trang XuLy.aspx.

#### 11.2 Truy xuất đến các phần tử bằng phương thức FindControl

Phương thức FindControl cho phép ta tìm kiếm (trò/ truy xuất) tới một phần tử nằm ở bên trong nó. Ví dụ: Tạo một trang có một textbox và một nút nhấn. Khi người dùng click vào nút nhấn thì hiển thị dòng chữ "Bạn đã dùng FindControl để truy xuất!".



#### 11.3 Truy xuất đến trang gửi thông qua thuộc tính PreviousPage.

Khi một trang X post (gửi) dữ liệu đến một trang Y, thì thuộc tính PreviousPage trong trang Y sẽ trở tới trang X. Hay có thể viết hình tượng là: **Y.PreviousPage = X.**

Ví dụ: Tạo một trang Login.aspx, Sau khi click nút Login, sẽ post đến trang XuLyDangNhap.aspx. Tại trang này sẽ đọc dữ liệu nằm trong textbox username, Password.

#### 11.4 MasterPage

Master Page là một trang đặc biệt cho phép các trang khác hiển thị bên trong nó. MasterPage thường dùng để tạo ra một mẫu sẵn (Ví dụ chứa Header, MainMenu và Footer), còn các trang nội dung không phải kèm thêm các phần Header, footer này.

Cách tạo: Chọn trang web dạng MasterPage

Cách cho trang khác hiển thị bên trong nó: Khi tạo form, chọn loại web content. Từ lần sau, chỉ cần tick vào phần: Select MasterPage.

Ví dụ: Tạo một trang Master chứa 3 nội dung là Header, MainMenu và Footer. Và 2 trang khác Home.aspx và GioiThieu.aspx (Main menu có *ZZ2* liên kết các trang này).



## Header

Trang chủ  
Giới thiệu  
Sản phẩm

Vùng này dùng để hiển thị các trang khi click hyperlink menu cạnh

## Footer

## **11.5 Gỡ rối**

Có 2 loại lỗi:

- Lỗi biên dịch (Compile error): Là loại lỗi được phát hiện ngay khi dịch chương trình
- Lỗi lập trình (logic) hay còn gọi là lỗi Run-time: Lỗi xảy ra khi chạy chương trình. Việc tìm ra các lỗi run-time gọi là quá trình gỡ rối (hay Debug).

### **11.5.1 Giới thiệu**

### **11.5.2 Chạy ứng dụng ở chế độ gỡ rối**

### **11.5.3 Khái niệm điểm dừng**

### **11.5.4 Chạy từng dòng lệnh với chế độ Step Into (F8)**

### **11.5.5 Chạy từng dòng lệnh với chế độ Step Over (Shift-F8)**

### **11.5.6 Chạy từng dòng lệnh với chế độ Step Out (Ctrl-Shift-F8)**

## **11.2 Sử dụng Custom Error page**

## **11.3 Ghi các vết gây lỗi (Trace errors)**

## **11.4 Sử dụng công cụ gỡ rối/ Menu Debug**

## **11.5 Tracing lỗi ở mức trang/ Mức toàn ứng dụng**

# **BÀI 12: THỰC HÀNH**

## BÀI 13: CÔNG NGHỆ ADO.NET

### 13.1 Giới thiệu chung

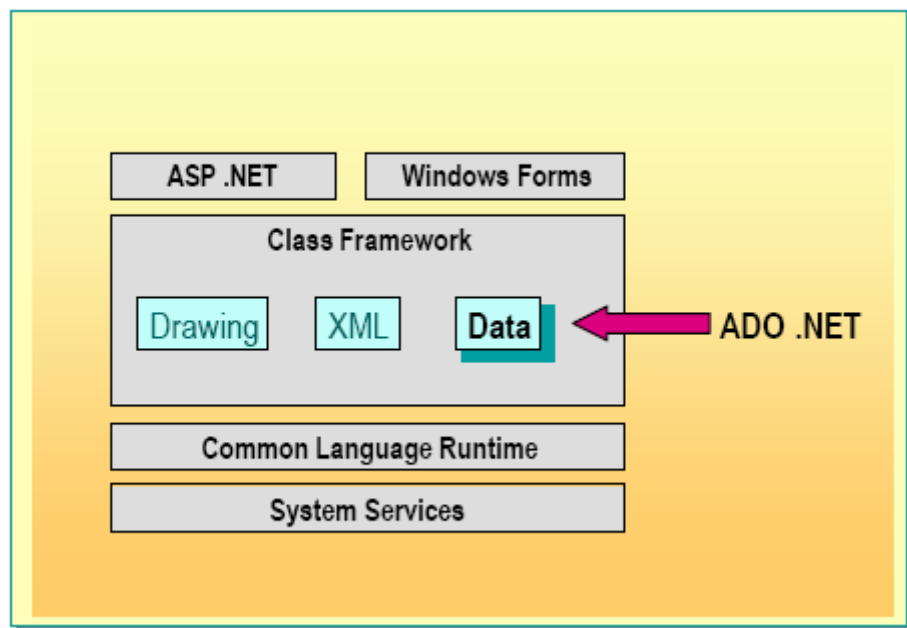
Khi phát triển các ứng dụng trên nền web thì công việc chủ yếu phải giải quyết là xử lý các nghiệp vụ, trong đó phần lớn là xử lý Cơ sở dữ liệu. Trong môi trường phát triển Microsoft .NET tất cả các ứng dụng webform hay winform đều thống nhất sử dụng chung một bộ thư viện để truy xuất và thao tác Cơ sở dữ liệu gọi là ADO.NET (Active Data Object).

- ADO.NET là **một tập các lớp** nằm trong bộ thư viện lớp cơ sở của .NET Framework, cho phép các ứng dụng windows (như C#, VB.NET) hay ứng dụng web (như ASP.NET) thao tác dễ dàng với các nguồn dữ liệu.

- **Mục tiêu chính của ADO.NET là:**

- Cung cấp các lớp để thao tác CSDL trong cả hai môi trường là phi kết nối (Disconnected data) và kết nối (Connected data).
- Tích hợp chặt chẽ với XML (Extensible Markup Language)
- Tương tác với nhiều nguồn dữ liệu thông qua mô tả dữ liệu chung.
- Tối ưu truy cập nguồn dữ liệu (OLE DB & SQL server).
- Làm việc trên môi trường Internet.

- Các lớp của ADO.NET được đặt trong Namespace là System.Data/ System.Data.oledb
- ADO.NET bao gồm 2 Provider (2 bộ thư viện thường dùng) để thao tác với các CSDL là: OLE DB Provider (nằm trong System.Data.OLEDB) dùng để truy xuất đến **bất kỳ CSDL nào có hỗ trợ OLEDB**; SQL Provider (nằm trong System.Data.SqlClient) chuyên dùng để truy xuất đến CSDL SQL Server (Không qua OLE DB nên nhanh hơn). Hiện nay, các hãng thứ ba còn cung cấp các Provider khác như : MySQL, Oracle... provider để cho phép ứng dụng .NET truy xuất đến các cơ sở dữ liệu không phải của Microsoft khác.
- Vị trí của ADO.NET trong kiến trúc của .NET Framework



Vị trí của ADO.NET trong kiến trúc của .net Framework

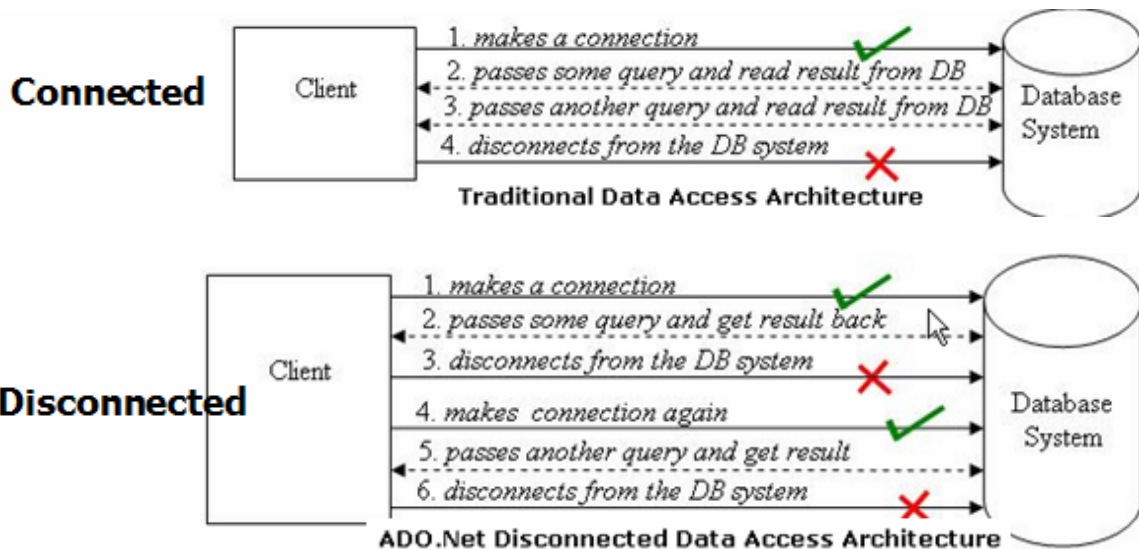
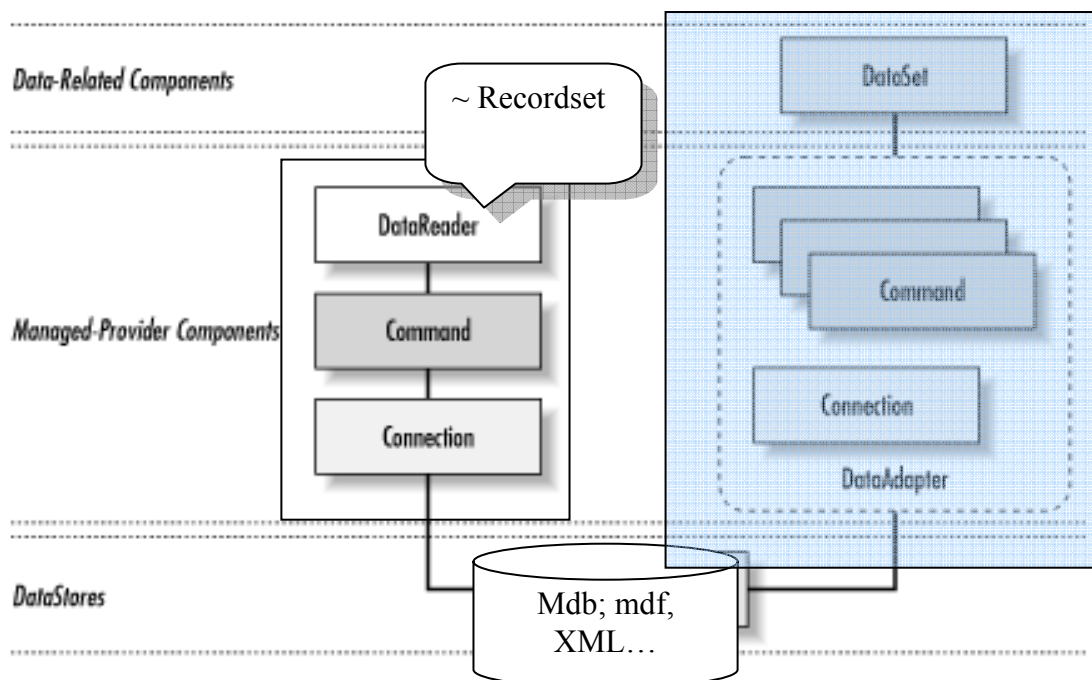
Từ kiến trúc ta thấy rằng: ADO.NET là một thành phần nội tại (Intrinsic) của .NET framework, do vậy nó có thể được sử dụng trong tất cả các ngôn ngữ hỗ trợ .NET như C#, VB.NET... mà không có sự khác biệt nào (Tức là các chức năng cũng như cách sử dụng hoàn toàn giống nhau).

### 13.2 Kiến trúc của ADO.NET

ADO.NET cho phép làm việc ở cả hai chế độ, chế độ Kết nối (Connected) và phi kết nối (Disconnected).

Bộ ba Connection, Command và DataReader: cho phép làm việc ở chế độ Connected; còn DataAdapter, Connection, Command và DataSet làm việc ở chế độ Disconnected.

Trong chế độ Connected thì mỗi khi thao tác (như sửa, xóa, thêm) thì đều đòi hỏi ứng dụng phải kết nối và thao tác trực tiếp với cơ sở dữ liệu (CSDL); còn trong chế độ Disconnected thì vẫn có thể thêm, sửa, xóa dữ liệu trên đối tượng cục bộ; không nhất thiết phải kết nối ngay đến CSDL (Xem mô hình ở dưới).



## 13.3 Các lớp thao tác với CSDL: Connection, Command,....

### 13.3.1 Lớp Connection

+ **Chức năng:** Là đối tượng có nhiệm vụ thực hiện kết nối đến Cơ sở dữ liệu để các đối tượng như Command thao tác với CSDL thông qua Connection này.

+ **Khai báo** (có nhiều cách):

```
public OleDbConnection Cn1;  
public OleDbConnection Cn2 = new OleDbConnection ();  
public OleDbConnection Cn3 = new OleDbConnection ("Provider=Microsoft.jet.....");
```

#### Một số phương thức:

+ **Open:** Dùng để mở kết nối:

Cnn.**Open**(): Mở kết nối đến CSDL do ta chỉ định trong ConnectionString

**Lưu ý:** sau khi gọi phương thức Open, có thể xem đã kết nối thành công hay không thông qua thuộc tính State của Connection:

*if (Cnn.State == 1) “Kết nối thành công !”*

+ **Close():** Dùng để đóng kết nối:

Cnn.**Close**();

→ Thường thì nên viết như sau để tránh lỗi : if (Cnn.State == 1) Cnn.Close();

+ **GetSchema:** Lấy thông tin về CSDL (Ví dụ tên các bảng, các trường trong bảng...)

#### Một số thuộc tính:

+ **State:** Cho biết trạng thái kết nối. (ConnectionState.Open → kết nối đã được mở)

+ **ConnectionString:** Chứa các thông tin để kết nối.

### Ví dụ về một trang thực hiện kết nối đến CSDL C:\Nwind.mdb

```
using System;
```

```
using System.Data;
```

```
using System.Data.OleDb;
```

```
public partial class Lesson_12_Default : System.Web.UI.Page
```

```
{
```

```
    /// <summary>
```

```
    /// Hàm kết nối đến Cơ sở dữ liệu
```

```
    /// </summary>
```

```
    /// <param name="DBFileName">Đường dẫn tới file MDB</param>
```

```
    /// <returns>Trả về đối tượng OleDbConnection hoặc null</returns>
```

```
    public static OleDbConnection OpenDB (string DBName)
```

```
    {
```

```
        try
```

```
        {
```

```
            OleDbConnection Conn = new OleDbConnection ();
```

```
            Conn.ConnectionString="Provider=Microsoft.jet.oledb.4.0;data source="+DBName;
```

```
            Conn.Open ();
```

```
            return Conn;
```

```
        }
```

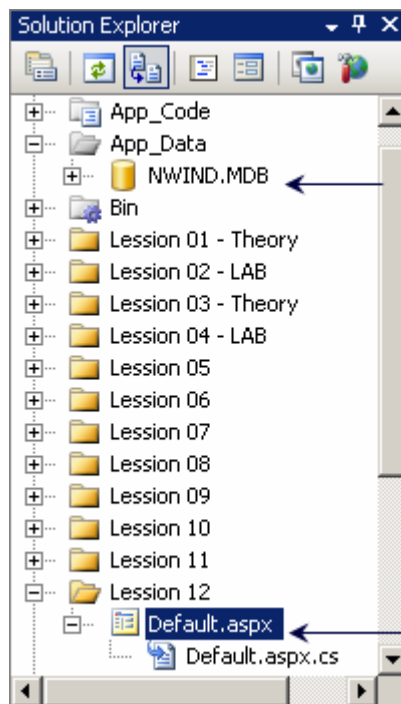
```
catch (Exception ex)
{
    return null;
}
}

// Kết nối đến cơ sở dữ liệu và thông báo kết quả kết nối trên một Label.
protected void Page_Load (object sender, EventArgs e)
{
    OleDbConnection Conn;
    Conn = OpenDB (@"c: \Nwind.mdb");

    if (Conn != null)
    {
        if (Conn.State == ConnectionState.Open)
            lblThongBao.Text = "Đã kết nối thành công ! " ;
        else
        {
            lblThongBao.Text = "Không thể kết nối được !";
        }
    }
}
```

**Chú ý:** Thông thường tệp cơ sở dữ liệu được lưu trong thư mục App\_Data. Khi đó để có thể kết nối đến CSDL này mà không cần biết thư mục hiện được đặt trong ổ C:\ hay D:\ ... thì cần viết đường dẫn của tệp như sau:

```
Conn = OpenDB (Server.MapPath("../App_Data/nwind.mdb"));
```



Vị trí của tệp CSDL nwind.mdb

- Nếu ổ đĩa chứa tệp CSDL được định dạng là NTFS và được đặt quyền truy cập thì cần phải đảm bảo rằng thư mục chứa tệp CSDL có quyền read/write cho người dùng là IUSR\_. (Có thể thay đổi quyền hoặc người dùng truy cập đến thư mục bằng cách Right click lên thư mục đó, chọn Properties, tiếp theo chọn thẻ Security và Add thêm người dùng/ quyền..)

### 13.3.2 Lớp Command

❖ Chức năng: Thực hiện các thao tác đối với CSDL, như Insert, Update, delete, Select. Tuy nhiên, để thực hiện được các lệnh này thì cần phải thông qua một Connection nào đó đang được mở.

❖ Cách tạo (chính tắc):

- ```
OleDbCommand Cmd;  
Cmd = new OleDbCommand ();  
Cmd.CommandText = "Câu lệnh SQL";  
Cmd.Connection= OleDbConnection_Obj;
```
- Hoặc viết gọn hơn:  

```
OleDbCommand Cmd=new OleDbCommand("Lệnh SQL",OleDbConnection_Obj);
```

Trong đó OleDbConnection\_Obj là một OleDbConnection mà trước đó đã Open rồi.  
Câu lệnh SQL: là một xâu chứa câu lệnh SQL bất kỳ.

❖ Một số phương thức dùng để thực thi câu lệnh SQL:

- **int ExecuteNonQuery():** Sử dụng khi CommandText ở trên thuộc dạng Insert, Delete, Update.... Hàm này trả về số bản ghi bị tác động (affected).
- **Object ExecuteScalar():** Sử dụng khi CommandText ở trên là câu lệnh SQL chỉ trả về một kết quả đơn, ví dụ câu lệnh đếm tổng số bản ghi : Select Count(\*) ... Hàm này trả về hàng và cột đầu tiên của kết quả thực thi truy vấn. Các hàng và cột khác bị bỏ qua.
- **OleDbDataReader ExecuteReader():** Dùng khi CommandText là một câu lệnh chọn (Select). Hàm trả về là một đối tượng OleDbDataReader chứa kết quả thực thi câu lệnh (thường là câu lệnh Select).
- **XMLReader ExecuteXMLReader():** Dùng để đọc dữ liệu là một tệp XML. Phương thức này chỉ áp dụng cho một số Provider (ví dụ SqlConnection)

❖ Một số thuộc tính

- **CommandText:** Chứa câu lệnh SQL cần thực thi, ví dụ: "Select \* from Employees", "Insert into Employees (...) values (...)", "Delete from Employees where ..."
- **Connection:** Để cho biết là đối tượng Command sử dụng kết nối nào.
- **CommandType:** Cho biết CommandText chứa StoreProcedure, tên bảng hay là câu lệnh SQL. Mặc định thuộc tính này có giá trị là Text.

```
Cmd.CommandType=CommandType.
```



❖ Ví dụ: Xây dựng một trang web hiển thị tổng số bản ghi của bảng Products trong cơ sở dữ liệu nwind.mdb.

### Default.aspx.cs

```
using System;
using System.Data;
using System.Data.OleDb;

public partial class Lesson_12_Default : System.Web.UI.Page
{
    // Kết nối đến cơ sở dữ liệu và thông báo kết quả trên một Label.
    protected void Page_Load (object sender, EventArgs e)
    {
        // Tạo đối tượng Connection và mở kết nối đến CSDL
        OleDbConnection Conn;
        Conn=new OleDbConnection();
        Conn.ConnectionString="Provider=Microsoft.jet.oledb.4.0; data source=" ;
        Conn.ConnectionString += Server.MapPath("../App_Data/nwind.mdb");
        Conn.Open();

        // Tạo đối tượng Command và thực thi câu lệnh đếm số bảng ghi
        OleDbCommand Cmd;
        Cmd = new OleDbCommand ();
        Cmd.CommandText = "Select Count(*) from Products";
        Cmd.Connection = Conn;

        // Hiển thị kết quả trên Label
        int SL = (int) Cmd.ExecuteScalar();
        lblThongBao.Text = "Số bản ghi trong bảng Products: " + SL.ToString ();

        // Giải phóng kết nối.
        Cmd.Dispose ();
        Conn.Close ();
    }
}
```

**Ví dụ:** Thêm Tên nhà cung cấp vào bảng Suppliers:

Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Command_InsertData.aspx.cs" Inherits="Lesson_12_Command_InsertData"
%>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head runat="server">
```

```
    <title>Insert data</title>
```

```
</head>
```

```
<body>
```

```
    <form id="form1" runat="server">
```

```
        <h2>
```

```
            <asp:TextBox runat="server" ID="txtNCC"></asp:TextBox>
```

```
            <asp:Button runat="server" ID="cmdAdd" Text="Thêm" onclick="cmdAdd_Click"
```



```
/>  
    </h2>  
  </form>  
</body>  
</html>
```

**Trang Code behind:**

```
using System;  
using System.Data;  
using System.Data.OleDb;  
  
public partial class Lesson_12_Command_InsertData : System.Web.UI.Page  
{  
    protected void Page_Load(object sender, EventArgs e)  
    {  
  
    }  
    protected void cmdAdd_Click (object sender, EventArgs e)  
    {  
        // Tạo đối tượng Connection và mở kết nối đến CSDL  
        OleDbConnection Conn;  
        Conn = new OleDbConnection ();  
        Conn.ConnectionString = "Provider=Microsoft.jet.oledb.4.0; data source=";  
        Conn.ConnectionString += Server.MapPath ("./App_Data/nwind.mdb");  
        Conn.Open ();  
  
        // Tạo đối tượng Command và thực thi câu lệnh đếm số bảng ghi  
        OleDbCommand Cmd;  
        Cmd = new OleDbCommand ();  
        Cmd.CommandText = "Insert into Suppliers(CompanyName) values('" + txtNCC.Text +  
        "')";  
        Cmd.Connection = Conn;  
        Cmd.ExecuteNonQuery ();  
  
        Cmd.Dispose ();  
        Conn.Close ();  
    }  
}
```

### 13.3.3 Lớp DataReader

- ❖ Chức năng: Dùng để đón nhận kết quả trả về từ phương thức **ExecuteReader** của đối tượng **Command**. Nó tương tự như một Recordset của ADO, tuy nhiên dữ liệu nhận về là Readonly và chỉ đọc theo chiều tiến.
- ❖ Một số phương thức:
  - Bool Read(): Thực hiện việc đọc một bản ghi (một hàng) trong kết quả, sau đó chuyển tới bản ghi tiếp theo. Hàm này trả về true nếu vẫn còn dữ liệu, false nếu đã đọc hết.
  - DataTable: GetTableSchema() → Trả về một dataTable mô tả thông tin về DataReader như tên các cột.

- String: GetName(int i) → Trả về tên của cột i
- GetInt(int i), GetString(int i),..., GetXXX(int i) → Trả về giá trị của cột i và chuyển về dạng Int, String,...
- ❖ Một số thuộc tính:
  - Boolean: **HasRows** cho biết là DataReader có chứa dữ liệu hay không ?
  - int FieldCount → Cho biết số trường (Cột) của DataReader.
  - Biến DataReader –ví dụ Dr – cho phép đọc dữ liệu của từng ô (cột/ trường) của hàng hiện hành như sau: Dr["Tên trường"/ hoặc chỉ số].
- ❖ Ví dụ : Nạp Tên của tất cả sản phẩm trong bảng Products và đưa vào một ListBox.

**Trang giao diện:**

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="DataReader.aspx.cs" Inherits="Lesson_12_DataReader" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>OleDbDataReader demo</title>
</head>
<body>
    <form id="form1" runat="server">
        <h2> Danh mục sản phẩm</h2>
        <asp:ListBox runat="server" ID="lstDSSP" Rows="20">
        </asp:ListBox>
    </form>
</body>
</html>
```

**Trang Code Behind**

```
using System;
using System.Data;
using System.Data.OleDb;

public partial class Lesson_12_DataReader : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        // Tạo đối tượng Connection và mở kết nối đến CSDL
        OleDbConnection Conn;
        Conn = new OleDbConnection ();
        Conn.ConnectionString = "Provider=Microsoft.jet.oledb.4.0; data source=";
        Conn.ConnectionString += Server.MapPath ("../App_Data/nwind.mdb");
        Conn.Open ();

        // Tạo đối tượng Command và thực thi câu lệnh đếm số bảng ghi
        OleDbCommand Cmd;
        Cmd.CommandText = "Select ProductName from Products";
        Cmd.Connection = Conn;
```

```
OleDbDataReader Dr; ///không có new
Dr = Cmd.ExecuteReader ();

//Duyệt và đưa vào lstDSSP
while (Dr.Read() == true)
{
    lstDSSP.Items.Add (Dr ["ProductName"].ToString());
}

Cmd.Dispose ();
Conn.Close ();
}
}
```

**Câu hỏi:** Viết lstDSSP.Items.Add (Dr [0].ToString()) có được không ?.

### 13.3.7 Lớp DataColumn

❖ Chức năng: Là một thành phần tạo nên DataTable.

❖ Khai báo:

```
DataColumn Dc;
Dc = new DataColumn (Tên_Cột);
```

Hoặc, tạo cột và chỉ định kiểu dữ liệu cho cột:

```
DataColumn Dc;
Dc = new DataColumn ("Hello", System.Type.GetType(Tên_Kiểu)); Trong đó
Tên_Kiểu có thể là String, Int32, ....
```

❖ Một số phương thức:

■

❖ Một số thuộc tính:

- Caption: Tiêu đề của cột
- ColumnName: Tên của cột.

❖ Ví dụ :

Tạo một cột có tên là Họ tên, kiểu String, Tuổi kiểu Int:

```
DataColumn Dc_HVT;
Dc_HVT = new DataColumn ("HoVaTen", System.Type.GetType ("String"));
Dc_HVT.Caption = "Họ và tên";
```

```
DataColumn Dc_Tuoi;
Dc_HVT = new DataColumn ("Tuoi", System.Type.GetType ("Int32"));
Dc_HVT.Caption = "Tuổi";
```

### 13.3.8 Lớp DataTable

❖ Chức năng: Quản lý dữ liệu dạng bảng 2 chiều (Các hàng và các cột).

❖ Khai báo:

```
DataTable Dt ;
```

Dt=new DataTable(); hoặc Dt = new DataTable("Tên\_Bảng");

❖ Một số phương thức:

- DataRow NewRow() : Trả về một đối tượng DataRow;
- Clear(): Xóa tất cả các dữ liệu trong DataTable

❖ Một số thuộc tính:

- Columns: Là một tập hợp, quản lý toàn bộ các cột (Thêm, xóa, sửa...) của DataTable. Columns lại có các phương thức để thêm/xóa cột.
- Rows: Là một tập hợp, quản lý toàn bộ các hàng trong DataTable. Rows cũng có các phương thức để thêm/xóa hàng.

❖ Truy xuất đến ô [i,j] của bảng: Tên\_Bảng.Rows[i][j]. Có thể dùng vòng lặp kiểu như  
for (i=0; i < Dt.Rows.Count; i++)  
for (j=0; j < Dt.Columns.Count; j++)  
{  
    ... Dt.Rows[i][j] ...  
}

để duyệt toàn bộ các ô trong Table.

❖ Ví dụ : Tạo một bảng có 2 cột là Họ tên (Kiểu String) và Tuổi (Kiểu Int32)

```
.....  
DataColumn Dc_HVT;  
Dc_HVT = new DataColumn ("HoVaTen", Type.GetType ("String"));  
Dc_HVT.Caption = "Họ và tên";  
  
DataColumn Dc_Tuoi;  
Dc_HVT = new DataColumn ("Tuoi", Type.GetType ("Int32"));  
Dc_HVT.Caption = "Tuổi";  
  
DataTable Dt ;  
Dt=new DataTable();  
Dt.Columns.Add (Dc_HVT);    // Tạo cột họ tên nhờ vào Column ở trên  
Dt.Columns.Add(Dc_Tuoi);    // Tạo cột tuổi.  
.....
```

Hoặc có thể thêm ngắn gọn hơn: **Dt.Columns.Add("HoVaTen",Type.GetType("String"));**

### 13.3.9 Lớp DataRow

❖ Chức năng: Là một đối tượng để quản lý một hàng của một DataTable.

❖ Khai báo: DataRow Dr; Lưu ý: Vì Dr phụ thuộc vào bảng (DataTable) nên nó chỉ được tạo ra bởi một DataTable có sẵn, không thể tạo DataRow theo kiểu:  
**DataRow Dr=new DataRow() !!!**

❖ Truy xuất các cột (ô) trong một DataRow như sau:

- Dr[Chỉ số] hoặc Dr[Tên\_Cột]. Trong đó Dr: là một biến kiểu DataRow

❖ Ví dụ : Tạo một bảng có hai cột Họ tên và Tuổi, sau đó chèn vào bảng này 2 bản ghi có giá trị tương ứng là {"Nguyễn Văn An", 30} và {"Nguyễn Văn Bình", 20}.

```
.....  
DataTable Dt ;  
Dt=new DataTable();
```

```
Dt.Columns.Add (Dc_HVT);
Dt.Columns.Add(Dc_Tuoi);

DataRow Dr;

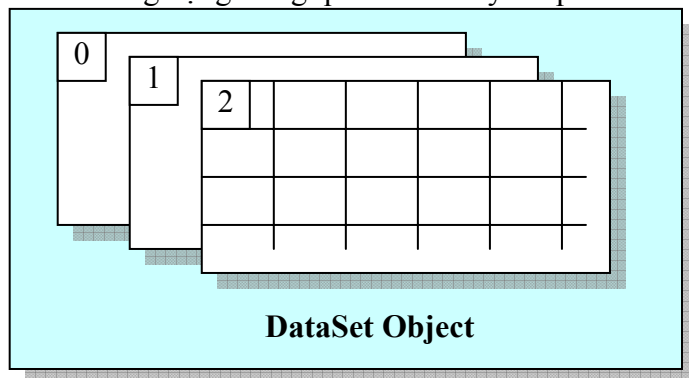
Dr = Dt.NewRow ();           // Tạo một hàng trắng
Dr ["HoVaTen"] = "Nguyễn Văn An";
Dr ["Tuoi"] = 30;
Dt.Rows.Add (Dr);           // Thêm vào bảng

Dr = Dt.NewRow ();           // Tạo một hàng trắng
Dr ["HoVaTen"] = "Nguyễn Văn Bình";
Dr ["Tuoi"] = 20;
Dt.Rows.Add (Dr);           // Thêm vào bảng
```

.....

### 13.3.10 Lớp DataSet

- ❖ Chức năng: Là một đối tượng chứa các DataTable. Nó là nơi lưu trữ dữ liệu tạm thời cho ứng dụng trong quá trình xử lý. Lớp DataSet này nằm trong System.Data.



- ❖ Khai báo:
  - DataSet Ds;
  - DataSet Ds = new DataSet();
- ❖ Một số phương thức:
  -
- ❖ Một số thuộc tính:
  - Tables: Chứa tất cả các bảng chứa trong Dataset.
  - Tables[i] hoặc Tables[Tên\_Bảng] : Tham chiếu đến một bảng cụ thể trong Dataset.
- ❖ Ví dụ : Xem ví dụ mục 13.3.11

### 13.3.11 Lớp DataAdapter

- ❖ Chức năng: Đóng vai trò cầu nối / Chuyển đổi dữ liệu giữa Nguồn dữ liệu (DataSource) và các đối tượng thao tác dữ liệu (như DataSet chẳng hạn).
- ❖ Một số phương thức:
  - Fill (DataSet, Tên\_Cho\_DataSet): Điền dữ liệu lấy được vào DataSet.

- Update(DataSet/DataTable...) : Cập nhật dữ liệu trong DataSet, DataTable ngược trở về Cơ sở dữ liệu.
- ❖ Một số thuộc tính:
  - SelectCommand, UpdateCommand, DeleteCommand, InsertCommand: trả về hoặc cho phép thiết lập các câu lệnh SQL để Chọn (Select), Cập nhật (Update), Delete, Insert vào Cơ sở dữ liệu.
- ❖ Ví dụ: Hiển thị toàn bộ bảng Suppliers ra màn hình

#### Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="DataSet.aspx.cs"
Inherits="Lesson_12_DataSet" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>DataSet demo</title>
</head>
<body>
  <form id="form1" runat="server">
    <h2>Danh sách nhà cung cấp</h2>
    <asp:DataGrid runat="server" ID="dgrNCC">
    </asp:DataGrid>
  </form>
</body>
</html>
```

#### Trang Code

```
using System;
using System.Data;
using System.Data.OleDb;

public partial class Lesson_12_DataSet : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        // Tạo đối tượng Connection và mở kết nối đến CSDL
        OleDbConnection Conn;
        Conn = new OleDbConnection ();
        Conn.ConnectionString = "Provider=Microsoft.jet.oledb.4.0; data source=";
        Conn.ConnectionString += Server.MapPath ("../App_Data/nwind.mdb");
        Conn.Open ();

        // Tạo đối tượng Command và select toàn bộ bảng Suppliers
        OleDbCommand Cmd;
        Cmd = new OleDbCommand ();
        Cmd.CommandText = "Select * from Suppliers";
        Cmd.Connection = Conn;

        OleDbDataAdapter Da;
        Da=new OleDbDataAdapter();
```

```
Da.SelectCommand=Cmd;  
  
DataSet DsNCC = new DataSet ();  
Da.Fill (DsNCC, "DS_NCC");  
  
// Hiển thị trên một bảng  
dgrNCC.DataSource = DsNCC.Tables ["DS_NCC"];  
dgrNCC.DataBind ();  
}  
}
```

## BÀI 14: THỰC HÀNH

**Mục tiêu:** Kết thúc bài học này, học viên có thể:

- ❖ Tạo được cơ sở dữ liệu cho hệ thống quản lý cán bộ
- ❖ Sử dụng được các lớp truy xuất của ADO.NET
- ❖ Hiển thị dữ liệu trên trình duyệt bằng các điều khiển ASP Server controls.
- ❖ Hoàn thiện chức năng Nhập hồ sơ cán bộ, trong đó có lưu vào Database.

**Nội dung:**

1. Tạo cơ sở dữ liệu Access QLCB.MDB có những bảng với cấu trúc như sau:

❖ Bảng tblCanBo

	Field Name	Data Type	Description
	id	AutoNumber	
	HoVaTen	Text	Họ và tên
	NgaySinh	Date/Time	Ngày sinh
	GioiTinh	Yes/No	Giới tính. Yes=Nam, No = Nữ
	DiaChi	Text	Địa chỉ (255)
	SoDienThoai	Text	Số điện thoại (50)
	NgayVaoDang	Date/Time	Ngày vào đảng (dd/MM/yyyy)
	MaBangCap	Text	Mã bằng cấp (DH, ThS, TS)
	MaPhongBan	Text	Mã phòng ban
	MaChuyenMon	Text	Mã chuyên môn (KS, CN, ThS, Ts)
	MaChucVu	Text	Mã chức vụ
	BanThan	Memo	Ghi chú về bản thân.

❖ Bảng Bằng cấp: tblBangCap

	Field Name	Data Type	Description
	MaBangCap	Text	Mã bằng cấp (DH, ThS, TS)
	MoTa	Text	Mô tả về bằng, nơi cấp, năm cấp v.v...

❖ Bảng phòng ban: tblPhongBan

	Field Name	Data Type	Description
	MaPhong	Text	Mã phòng ban
	TenPhong	Text	Tên phòng ban
	SoDienThoai	Text	Số điện thoại liên hệ của Phòng ban
	DiaChi	Text	Địa chỉ của phòng ban (Phòng máy, tầng mấy...)

❖ Bảng trình độ chuyên môn: tblChuyenMon

	Field Name	Data Type	Description
	MaChuyenMon	Text	Mã chuyên môn (KS hóa, Ths Toán, TS văn chương...)
	MoTa	Text	Mô tả

❖ Bảng chức vụ: tblChucVu

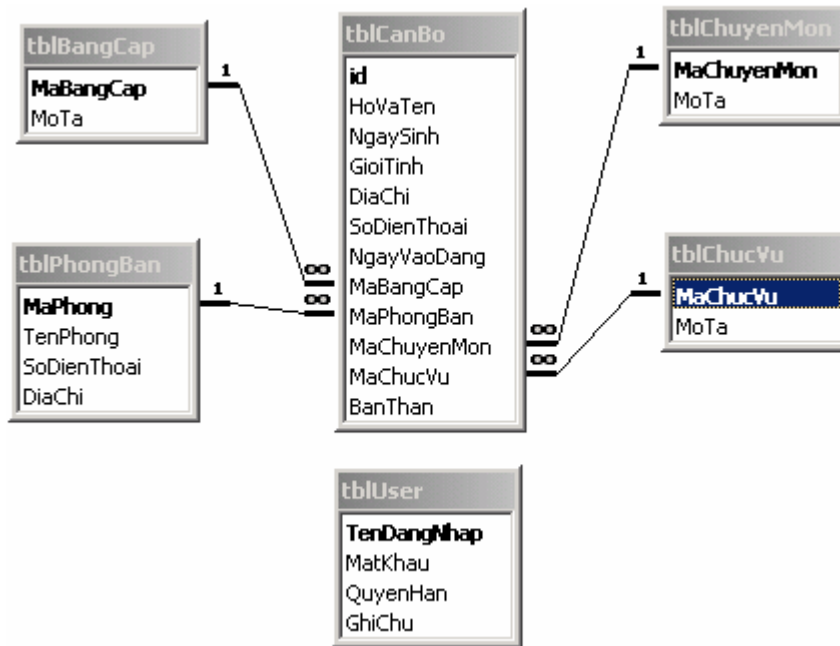
	Field Name	Data Type	Description
	MaChucVu	Text	Mã chức vụ (GD, TP, PTP, ...)
	MoTa	Text	Mô tả (Trưởng phòng Khoa học, Giám đốc kỹ thuật...)



❖ Bảng người dùng

	Field Name	Data Type	Description
	TenDangNhap	Text	Tên đăng nhập
	MatKhai	Text	Mật khẩu
	QuyenHan	Number	Quyền hạn (1=thấp nhất, 5=cao nhất). Default=1
	TrangThai	Number	1=Active, 0 = unactive (Bị khóa)
	NgayTao	Date/Time	Ngày tạo tài khoản này.
	GhiChu	Text	Ghi chú thêm

❖ Mối quan hệ giữa các bảng (Chú ý chọn Cascading Update/ Delete khi tạo)



2. **Nhập dữ liệu cho bảng.** Ở đây chỉ hướng dẫn cách nhập mẫu cho bảng tblUser, vì bảng này chứa cả dữ liệu dạng Xâu, số, bool, datetime. Học viên tạo các bảng khác một cách tương tự.

2.1 Thiết kế trang giao diện (Chú ý: Các trang cần phải được gắn vào trong hệ thống giao diện đã được xây dựng ở bài trước, ví dụ: đưa vào MasterPage)

Trang giao diện NhapNguoiDung.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="NhapNguoiDung.aspx.cs"
Inherits="Lesson_13_NhapNguoiDung" %>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Nhập thông tin người dùng</title>
    <style type="text/css">
        .CanPhai {text-align:right; font-style:italic}
    </style>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <table style="border:solid 1px purple; border-collapse:collapse;">
```

```
<tr>
  <td colspan="2" style="background-color:Purple;color:White">
    <h3 style="margin:3px 3px 3px 3px">Nhập thông tin người dùng</h3>
  </td>
</tr>

<tr>
  <td class="CanPhai">
    Tên đăng nhập
  </td>
  <td>
    <asp:TextBox ID="txtUserName" runat="server"></asp:TextBox>
  </td>
</tr>

<tr>
  <td class="CanPhai">
    Mật khẩu
  </td>
  <td>
    <asp:TextBox ID="txtPassword" runat="server"></asp:TextBox>
  </td>
</tr>

<tr>
  <td class="CanPhai">
    Quyền hạn
  </td>
  <td>
    <asp:DropDownList ID="ddlQuyenHan" runat="server"
      style="text-align:left">
      <asp:ListItem>1</asp:ListItem>
      <asp:ListItem>2</asp:ListItem>
      <asp:ListItem>3</asp:ListItem>
      <asp:ListItem>4</asp:ListItem>
      <asp:ListItem>5</asp:ListItem>
    </asp:DropDownList>
  </td>
</tr>

<tr>
  <td class="CanPhai">
    Ghi chú
  </td>
  <td>
    <asp:TextBox ID="txtGhiChu" runat="server"
      Rows="2" TextMode="MultiLine"></asp:TextBox>
  </td>
</tr>

<tr>
  <td>
  </td>
  <td>
  </td>
  <td>
    <asp:Button ID="cmdThem" runat="server" Text="Thêm" onclick="cmdThem_Click" />
  </td>
</tr>
</table>
</div>
</form>
```

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

## 2.2 Viết code behind.

### NhapNguoiDung.aspx.cs

```
using System;
using System.Data;
using System.Data.OleDb;

public partial class Lesson_13_NhapNguoiDung : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void cmdThem_Click (object sender, EventArgs e)
    {
        // Tạo đối tượng Connection và mở kết nối đến CSDL
        OleDbConnection Conn;
        Conn = new OleDbConnection ();
        Conn.ConnectionString = "Provider=Microsoft.jet.oledb.4.0; data source=";
        Conn.ConnectionString += Server.MapPath ("../App_Data/QLCB.mdb");
        Conn.Open ();

        string strSQL;
        string TenDN=txtUserName.Text;
        string MatKhai=txtPassword.Text;
        string GhiChu=txtGhiChu.Text;
        string NgayTao=DateTime.Now.Month.ToString() + "/" + DateTime.Now.Day.ToString();
        NgayTao += "/" + DateTime.Now.Year.ToString();
        int QuyenHan=int.Parse(ddlQuyenHan.Text);

        strSQL="Insert into tbluser(TenDangNhap, MatKhai, QuyenHan, NgayTao, GhiChu) ";
        strSQL += " values ('" + TenDN + "','" + MatKhai +
            "','" + QuyenHan + "','" + NgayTao + "','" + GhiChu + "')";

        // Tạo đối tượng Command và select toàn bộ bảng Suppliers
        OleDbCommand Cmd;
        Cmd = new OleDbCommand ();
        Cmd.CommandText = strSQL;
        Cmd.Connection = Conn;
        Cmd.ExecuteNonQuery ();

        txtUserName.Text = "";
        txtPassword.Text = "";
        txtGhiChu.Text = "";
        ddlQuyenHan.Text = "1";

        txtUserName.Focus ();
    }
}
```

### 3. Hiển thị dữ liệu trong bảng lên trình duyệt

#### 3.1 Trang giao diện

##### HienThiNguoiDung.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="HienThiNguoiDung.aspx.cs"
Inherits="Lesson_13_HienThi" %>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Danh sách người dùng</title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <h2> Danh sách người dùng</h2>
            <asp:DataGrid runat="server" id="dgrDSNS"></asp:DataGrid>
        </div>
    </form>
</body>
</html>
```

#### 3.2 Trang code behind

##### Trang HienThiNguoiDung.aspx.cs

```
using System;
using System.Data;
using System.Data.OleDb;
public partial class Lesson_13_HienThi : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        // Tạo đối tượng Connection và mở kết nối đến CSDL
        OleDbConnection Conn;
        Conn = new OleDbConnection ();
        Conn.ConnectionString = "Provider=Microsoft.jet.oledb.4.0; data source=";
        Conn.ConnectionString += Server.MapPath ("../App_Data/QLCB.mdb");
        Conn.Open ();

        // Tạo đối tượng Command và select toàn bộ bảng Suppliers
        OleDbCommand Cmd;
        Cmd = new OleDbCommand ();
        Cmd.CommandText = "Select * from tblUser";
        Cmd.Connection = Conn;

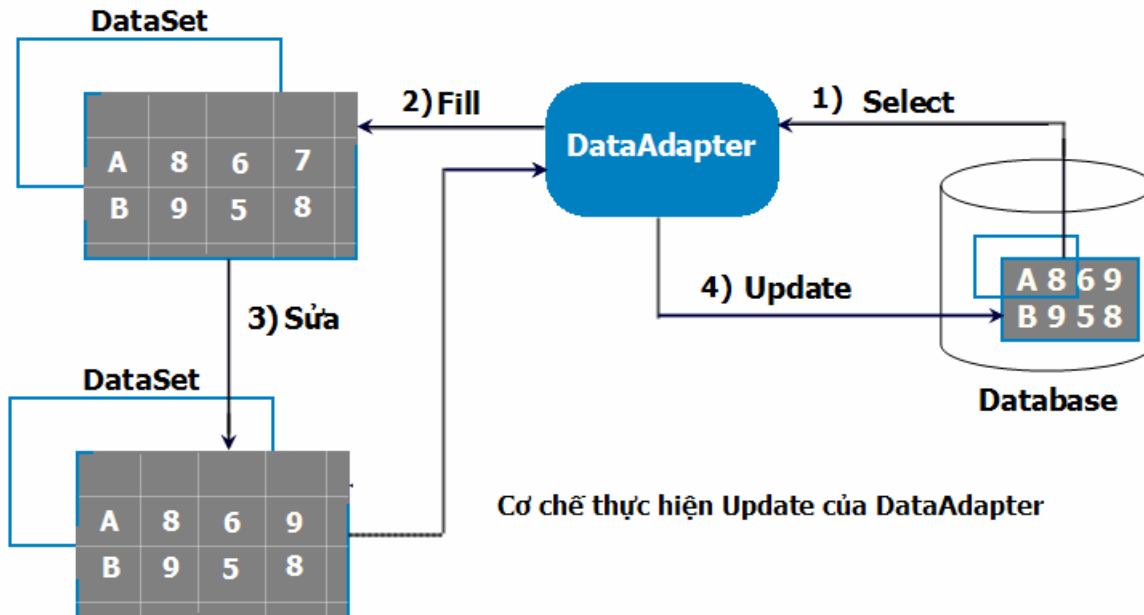
        OleDbDataAdapter Da;
        Da = new OleDbDataAdapter ();
        Da.SelectCommand = Cmd;

        DataSet DsNCC = new DataSet ();
        Da.Fill (DsNCC, "DS_User");

        // Hiển thị trên một bảng
        dgrDSNS.DataSource = DsNCC.Tables ["DS_User"];
        dgrDSNS.DataBind ();
    }
}
```

#### 4. Cập nhật dữ liệu bằng DataSet và DataAdapter

Để cập nhật dữ liệu vào trong CSDL, ta có thể dùng câu lệnh SQL dạng như "UPDATE ...WHERE ...." và thực thi bằng phương thức ExecuteNonQuery của đối tượng Command. Tuy nhiên còn có một cách khác để cập nhật nữa, đó là dùng phương thức Update của đối tượng DataSet và DataAdapter. Mô hình dạng như sau:



Các bước cần phải thực hiện khi Update bằng DataAdapter:

1. Tạo kết nối đến CSDL
2. Tạo đối tượng Command và đặt câu lệnh Select cho thuộc tính CommandText. Việc đặt câu lệnh Select ở đây là để về sau DataAdapter biết được các trường của bảng gồm những trường nào ?
3. Điền (Fill) dữ liệu vào một DataSet.
4. Chỉnh sửa dữ liệu trong các bảng của DataSet
5. Tạo một thể hiện của CommandBuilder (OleDbCommandBuilder/ SqlCommandBuilder)
6. Gọi phương thức Update của DataAdapter để cập nhật thực sự vào Database.

#### Ví dụ:

Sửa đổi trạng thái đăng nhập (Trường TrangThai) của tất cả người dùng trong bảng tblUser thuộc cơ sở dữ liệu QLCB.mdb thành 1.

#### Trang giao diện UpdatewithDataAdapter.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UpdatewithDataAdapter.aspx.cs"
Inherits="Lesson_13_UpdatewithDataAdapter" %>
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title>Cập nhật dữ liệu thông qua Data Adapter</title>
</head>
<body>
<form id="form1" runat="server">
<h2>Sửa trạng thái đăng nhập thành 1</h2>
<asp:Button runat="server" ID="cmdEnableAllUser"
Text="Enable all users"
```

```
        onclick="cmdEnableAllUser_Click" />
    </form>
</body>
</html>
```

#### Trang code behind: UpdatewithDataAdapter.aspx.cs

```
using System;
using System.Data;
using System.Data.OleDb;

public partial class Lesson_13_UpdatewithDataAdapter : System.Web.UI.Page
{
    protected void cmdEnableAllUser_Click (object sender, EventArgs e)
    {
        // Tạo đối tượng Connection và mở kết nối đến CSDL
        OleDbConnection Conn;
        Conn = new OleDbConnection ();
        Conn.ConnectionString = "Provider=Microsoft.jet.oledb.4.0; data source=";
        Conn.ConnectionString += Server.MapPath ("../App_Data/QLCB.mdb");
        Conn.Open ();

        // Tạo đối tượng Command và select toàn bộ bảng tblUser
        OleDbCommand Cmd;
        Cmd = new OleDbCommand ();
        Cmd.CommandText = "Select * from tbluser";
        Cmd.Connection = Conn;

        OleDbDataAdapter Da;
        Da = new OleDbDataAdapter ();
        Da.SelectCommand = Cmd;

        // Điền dữ liệu vào DataSet
        DataSet Ds=new DataSet();
        Da.Fill(Ds,"DSND");

        // Sửa dữ liệu của bảng tblUser trong Dataset
        for (int i=0; i < Ds.Tables["DSND"].Rows.Count; i++)
        {
            Ds.Tables["DSND"].Rows [i]["TrangThai"] = 1;
        }

        //Cập nhật trở lại Cơ sở dữ liệu
        OleDbCommandBuilder CmdBuilder = new OleDbCommandBuilder (Da);
        Da.Update (Ds,"DSND"); /// Cập nhật bảng DSND trong Ds vào Cơ sở dữ liệu

        //Giải phóng các đối tượng không còn sử dụng
        Cmd.Dispose ();
        Da.Dispose ();
        Conn.Close ();
    }
}
```

## BÀI 15: Tìm hiểu và ứng dụng cơ chế Data Binding

### 15.1 Giới thiệu DataBinding

ASP.NET cung cấp cho chúng ta rất nhiều điều khiển để cho phép hiển thị cũng như tiếp nhận thông tin từ người dùng. Có những điều khiển cho phép chúng ta hiển thị những thông tin tĩnh (Static – tức là giá trị xác định được ngay khi lập trình), một số hiển thị được cả những thông tin động (Dynamic - tức là được tính toán khi chạy chương trình). Để việc hiển thị thông tin động này một cách đơn giản và nhanh chóng, ASP.NET cung cấp cho chúng ta một đặc tính gọi là "Data Binding" (Tạm dịch: "Gắn kết dữ liệu").

Thuật ngữ Data Binding ở đây được hiểu là "Gắn/ liên kết các điều khiển với một nguồn dữ liệu nào đó để hiển thị, thao tác...tự động". Từ "Data" cũng cần phải được hiểu theo nghĩa rộng, nó không nhất thiết phải là cái gì đó liên quan đến Cơ sở dữ liệu như ta thường nghĩ mà có thể là một thuộc tính, một mảng, một tập hợp, một danh sách, một trường dữ liệu trong bảng CSDL....hay tổng quát là một biểu thức trả về giá trị.

Điểm khác biệt chính của cơ chế Data binding so với các cơ chế liên kết dữ liệu khác đó là ở tính khai báo. Việc khai báo này không phải ở trong file Code behind (\*.CS) mà là trong file giao diện (\*.ASPX). Điều này làm cho code và điều khiển tách biệt và sáng sủa hơn.

Bài học này sẽ giới thiệu đặc tính Databinding bằng các điều khiển ASP.NET Server.

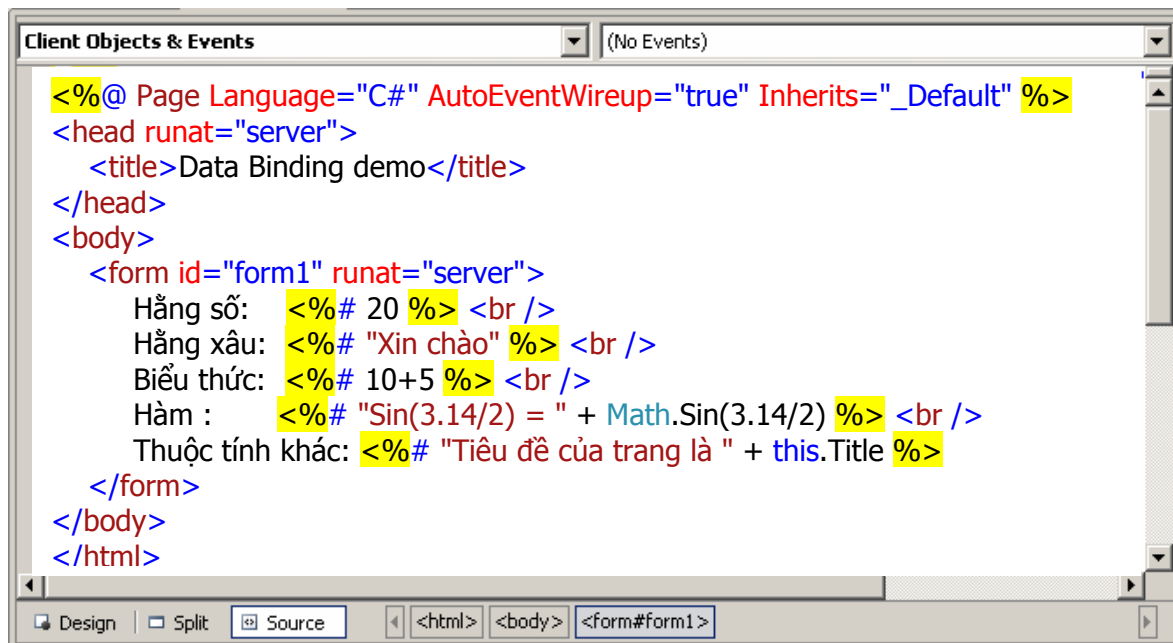
### 15.2 Data Binding

#### 15.2.1 Dạng gắn kết dữ liệu đơn (Single DataBinding)

Trong ASP.NET, có thể gắn một giá trị đơn lẻ vào trang được gọi là Single Data Binding.

Cú pháp để gắn dữ liệu đơn vào trang như sau: **<%# Giá\_Trị %>**.

Trong đó: Giá\_Trị có thể là một hằng, một biến, một hàm, một biểu thức hoặc có thể là một thuộc tính khác. Ví dụ:



Có thể gắn kết tới một biểu thức, một biến, thuộc tính ... bất kỳ.

**Chú ý:** Trong thủ tục Page\_Load cần thêm lệnh **this.DataBind()** để thực sự gắn kết.

### 15.2.2 Dạng gắn kết dữ liệu có sự lặp lại (Repeated Data Binding)

Có rất nhiều trường hợp dữ liệu cần hiển thị là một danh sách (Ví dụ Mảng, bảng, DataReader,...) hay tổng quát là một tập hợp các mục (Collection Items ). Trong trường hợp như vậy, ta hoàn toàn có thể dùng cơ chế DataBinding trong ASP.NET để gắn kết quả đó vào một điều khiển dạng danh sách (Ví dụ ListBox, DropDownList, CheckedList,...) để hiển thị mà không cần phải viết nhiều dòng code.

Các điều khiển cho phép gắn kết dữ liệu thường có 3 thuộc tính với các ý nghĩa như sau:

- ❖ **DataSource**: Là thuộc tính để chỉ đến nguồn dữ liệu cần gắn kết. Nguồn dữ liệu này phải là một tập hợp. Ví dụ: DataTable, Array,...
- ❖ **DataSourceID**: Chỉ đến một đối tượng cung cấp nguồn dữ liệu. Có thể sử dụng hoặc thuộc tính DataSourceID hoặc DataSource nhưng không được cả hai.
- ❖ **DataTextField**: Cho biết là gắn kết với trường nào của mỗi mục dữ liệu.

#### Ví dụ 1:

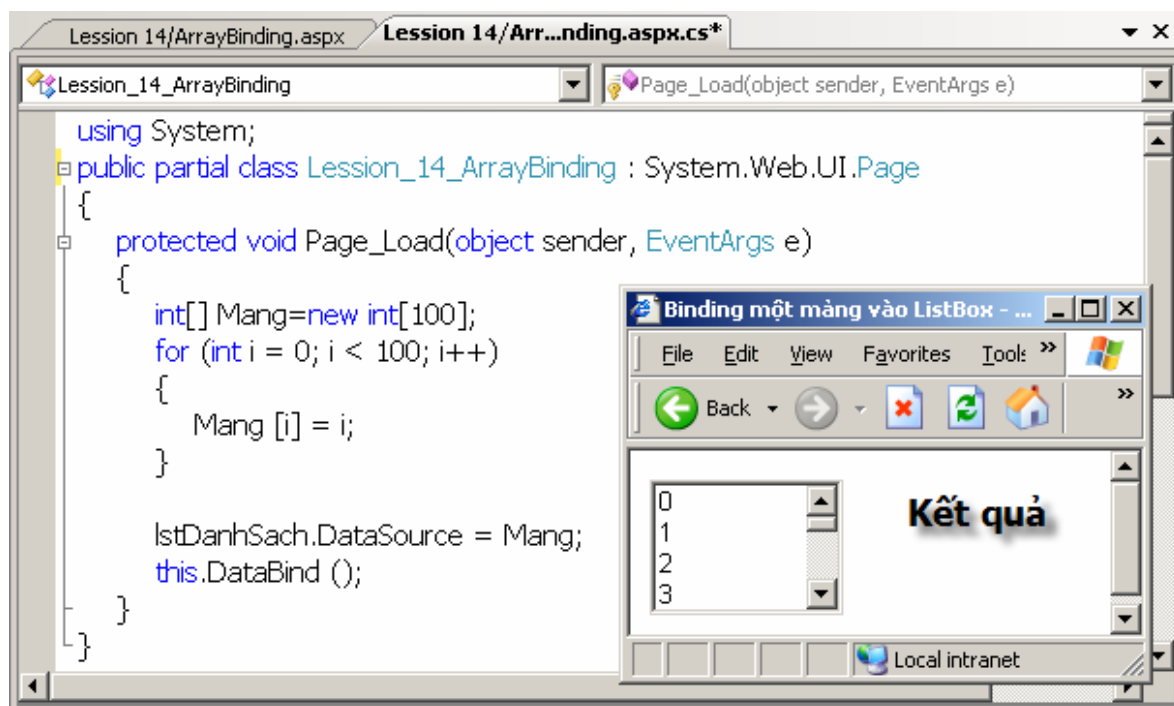
Tạo một mảng có 100 phần tử (từ 0 đến 99), sau đó hiển thị trên một Listbox thông qua cơ chế DataBinding:



Trang giao diện: ArrayBinding.aspx

**Chú ý:** Vì ở đây mỗi một phần tử của mảng chỉ có một giá trị duy nhất, do vậy trong 3 thuộc tính DataSource, DataSourceID và DataTextField ta chỉ cần đặt giá trị duy nhất là DataSource khi thực hiện bind mảng này vào ListBox để hiển thị. Cụ thể như trong trang Code behind sau đây. Nhắc lại rằng, để việc Bind dữ liệu thực sự diễn ra, ta cần gọi phương thức DataBind của điều khiển hoặc DataBind của trang (Nếu gọi phương thức DataBind của trang thì tất cả các điều khiển con thuộc trang sẽ tự động gọi phương thức DataBind của riêng nó).



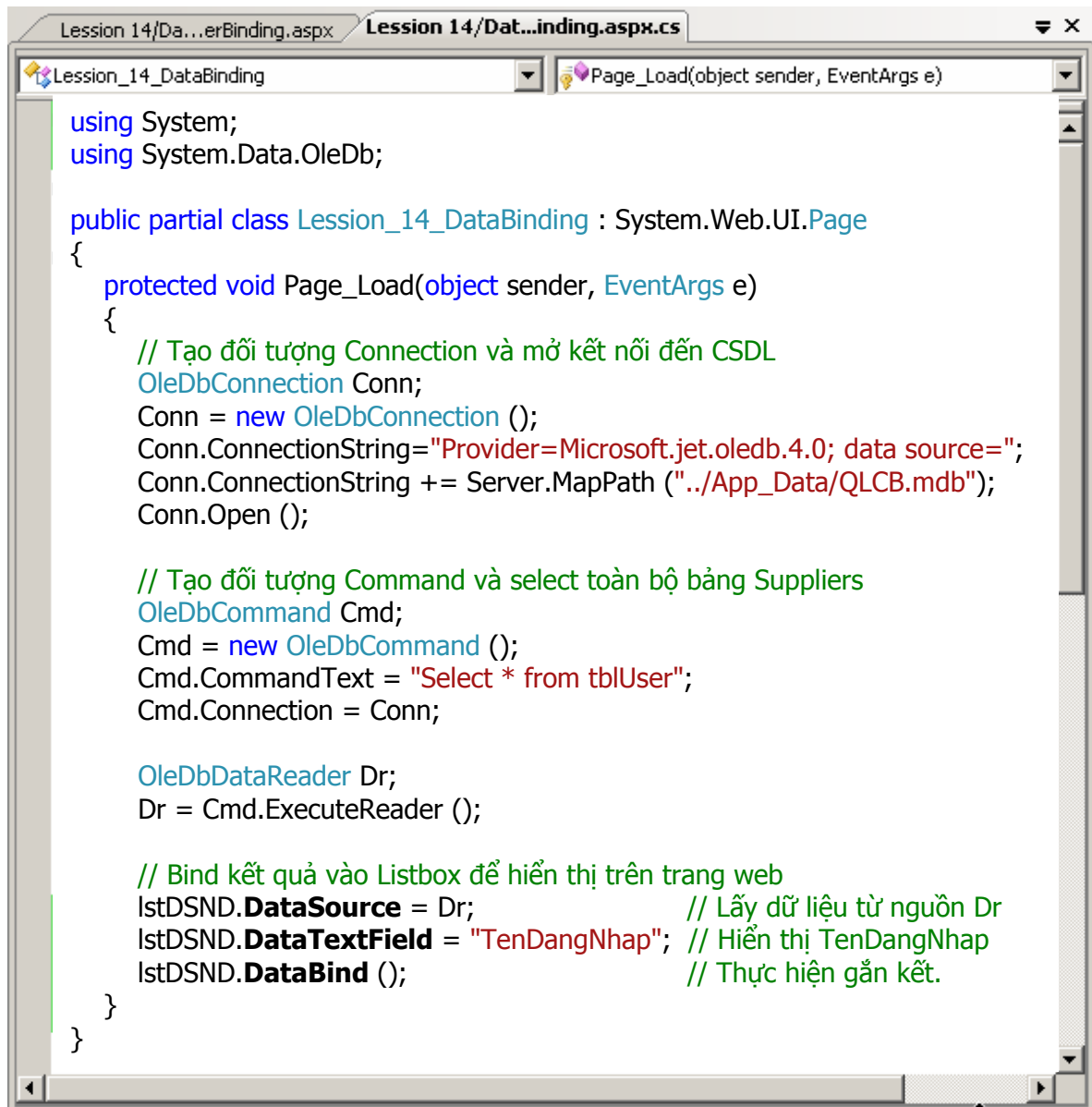


Trang Code và kết quả: ArrayBinding.aspx.cs

**Ví dụ 2:** Dùng cơ chế `DataBind`, hiển thị tất cả tên đăng nhập trong bảng `tblUser` thuộc `CSDL QLCB.MDB` vào một `ListBox`:



Trang giao diện



Trang code thực hiện việc Binding

**Chú ý:**

- Ta gắn kết dữ liệu giữa Listbox với DataReader được bởi vì Dr chứa một tập hợp phần tử
- Vì mỗi mục dữ liệu của nguồn dữ liệu (Dr) trong trường hợp này lại chứa nhiều trường, do vậy ta cần phải chỉ rõ thêm là Listbox sẽ gắn/lấy trường nào ra để hiển thị thông qua việc gán tên trường cần hiển thị cho thuộc tính DataTextField.
- Sau cùng, cần phải gọi phương thức DataBind () để thực hiện gắn kết và hiển thị thực sự.

## 15.3 Các điều khiển Data Source (Data source controls).

### 15.3.1 Giới thiệu về DataSource controls

Ở bài trước chúng ta đã sử dụng các đối tượng truy xuất dữ liệu (như datareader) kết hợp với vòng lặp (while (Dr.DataRead() == true)) để duyệt và đọc toàn bộ bản ghi lấy về. Tuy nhiên, còn một cách khác để đọc dữ liệu mà không phải viết code đó là dùng các điều khiển Data Source.

Hiện tại ASP.NET cung cấp một số Data source controls sau đây:

- **SqlDataSource**: Cho phép truy xuất tới bất kỳ nguồn dữ liệu sử dụng trình điều khiển (Provider) của ADO.NET. Bao gồm OleDb, SqlClient, ODBC, Oracle. (bài này sẽ sử dụng SqlDataSource để minh họa)
- **ObjectDataSource**: Truy xuất tới nguồn dữ liệu do người dùng tự định nghĩa.
- **AccessDataSource**: Truy xuất tới nguồn dữ liệu Access
- **XmlDataSource**: Truy xuất tới nguồn dữ liệu là file XML.

Ý tưởng của DataSource control là: "*Bạn chỉ việc đặt vài thông số kết nối và câu lệnh sql, sau đó có thể gắn vào control này để lấy lại dữ liệu.*". Việc gắn và lấy dữ liệu này thực hiện dễ dàng thông qua các thuộc tính khi khai báo control. Tuy nhiên, với DataSource control thì không chỉ có vậy, nó còn cho phép thực hiện các thao tác cập nhật khác như Update, delete,...

### 15.3.2 Sử dụng SqlDataSource để chọn (Select) dữ liệu

a) Tạo điều khiển SqlDataSource:

Cú pháp:

```
<asp:SqlDataSource runat="server" ProviderName="Tên_Provider"
    ConnectionString="Đường dẫn tới file CSDL hoặc thông số kết nối"
    SelectCommand/ UpdateCommand/ DeleteCommand ="Câu lệnh SQL tương ứng"
    ID="Giá trị ID">
```

**Lưu ý:** Để cho linh hoạt, thông số kết nối thường được đặt trong file cấu hình (web.config).

Để đọc cấu trúc kết nối này ta có thể thực hiện thông qua ký pháp dạng: <%\$ %>.

**Ví dụ:** Tạo một điều khiển SqlDataSource để đọc toàn bộ nội dung của bảng tblUser trong cơ sở dữ liệu QLCB.

Nội dung của file web.config như sau:

web.config

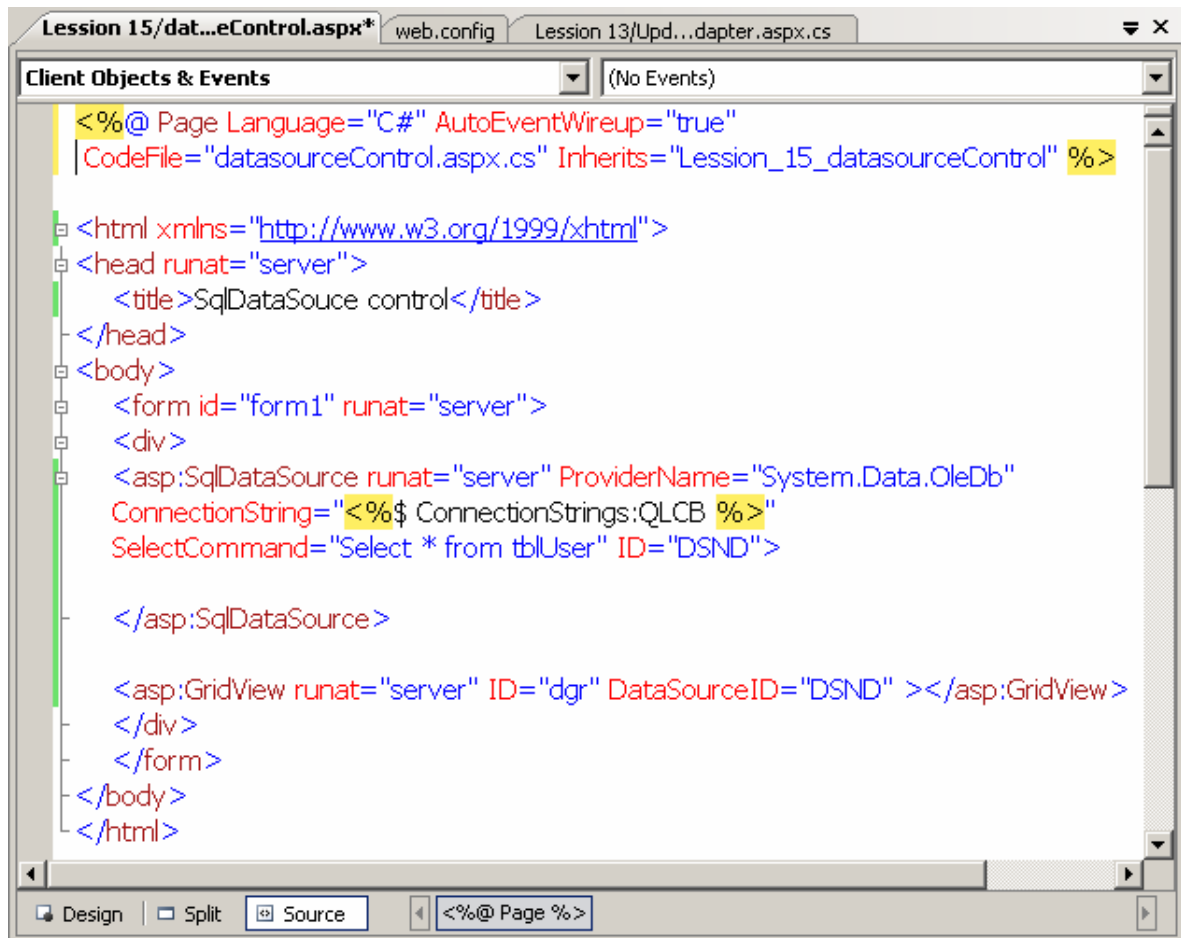
```
.....
<connectionStrings>
  <add name="QLCB" connectionString="Provider=Microsoft.jet.oledb.4.0;
    Data Source=|DataDirectory|\QLCB.mdb"/>
</connectionStrings>
.....
```

Trong đó: |DataDirectory| sẽ cho ta đường dẫn tới thư mục App\_Data.

Nếu cơ sở dữ liệu cần kết nối là SQL Server thì cấu trúc sẽ có dạng:

web.config

```
.....
<connectionStrings>
  <add name="QLCB" connectionString="Provider=SQLOLEDB.1;Integrated
Security=SSPI;Persist Security Info=False;Initial Catalog=QLCB;Data Source=localhost"/>
</connectionStrings>
.....
```



Tạo một SqlDataSource và select dữ liệu trong bảng tblUser

Sau khi đã tạo một nguồn dữ liệu ở trên rồi thì việc sử dụng nguồn dữ liệu này để hiển thị khá đơn giản.

**Ví dụ:** Hiển thị bảng tblUser vừa lấy được ở trên.

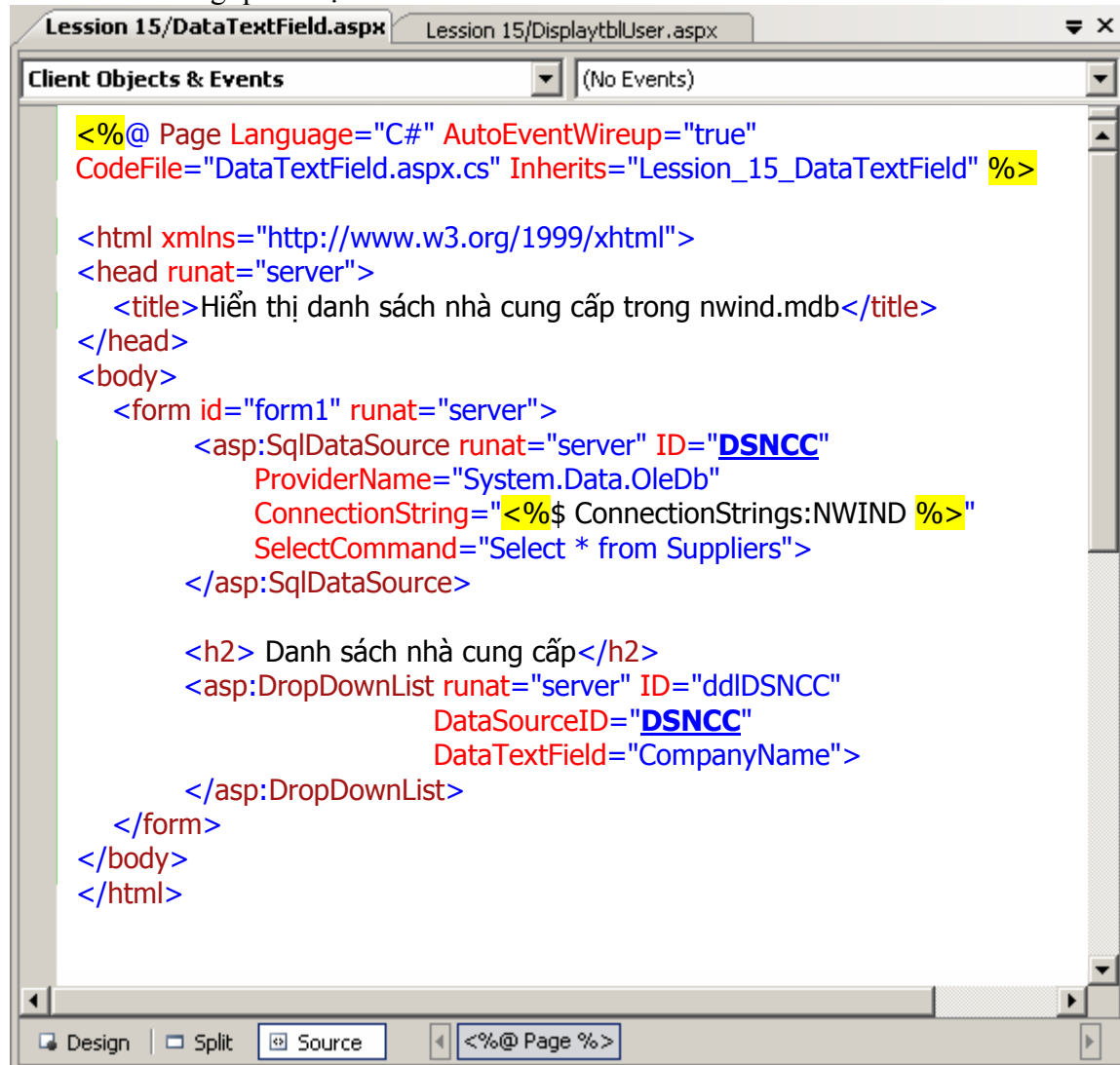
#### Hiển thị danh sách người dùng

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="datasourceControl.aspx.cs" Inherits="Lesson_15_datasourceControl" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title>SqlDataSource control</title>
</head>
<body>
<form id="form1" runat="server">
<asp:SqlDataSource runat="server" ProviderName="System.Data.OleDb"
ConnectionString="<%%$ ConnectionStrings:QLCB %>"
SelectCommand="Select * from tblUser" ID="DSND" >
</asp:SqlDataSource>

<asp:GridView runat="server" ID="dgr" DataSourceID="DSND" >
</asp:GridView>
</form>
</body>
</html>
```

**Ví dụ 2:** Hiển thị tên các nhà cung cấp trong một Dropdownlist;

Phân tích: Ở đây Dropdownlist chỉ hiển thị được một cột (trường dữ liệu) trong khi Dr chứa cả một bảng (có nhiều cột). Do vậy, cần phải chỉ cho Dropdownlist biết là gắn với trường nào của Dr thông qua thuộc tính DataTextField.



Trong đó: Xâu kết nối NWIND của file web.config có giá trị như sau:

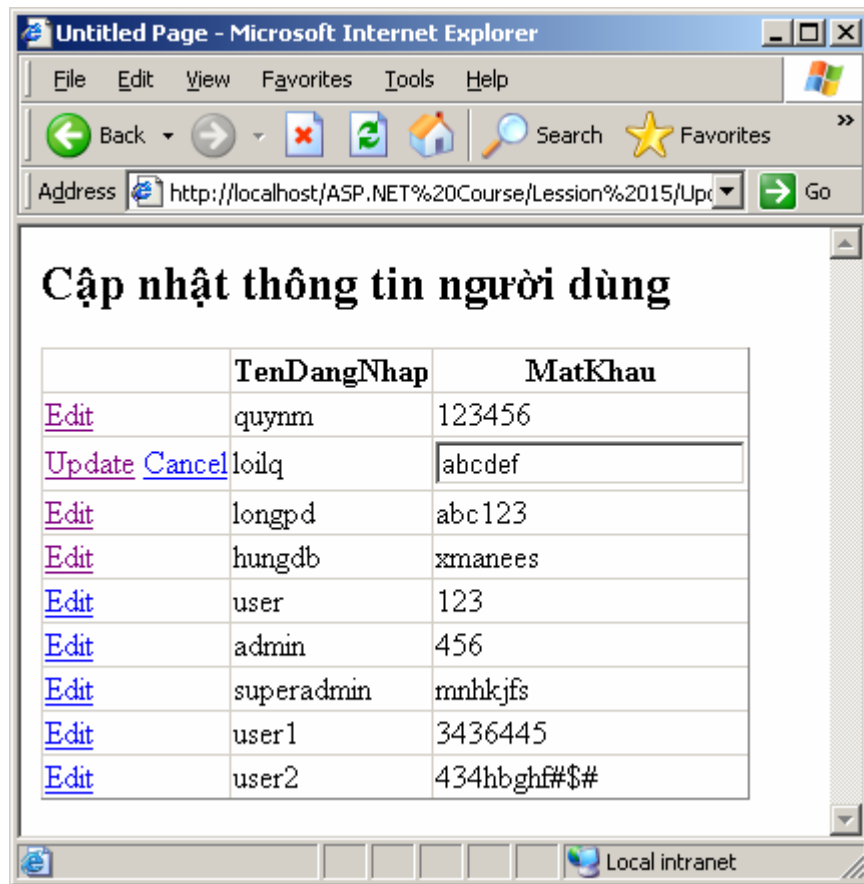
```
<connectionStrings>
<add name="NWIND" connectionString="Provider=Microsoft.jet.oledb.4.0;
Data Source=|DataDirectory|\nwind.mdb"/>
</connectionStrings>
```

### 15.3.3 Sử dụng SqlDataSource để cập nhật dữ liệu

Để cập nhật dữ liệu thì trong khai báo điều khiển SqlDataSource ta cần cung cấp cụ thể câu lệnh Update cho thuộc tính UpdateCommand.

**Lưu ý:** Khi thực hiện Update, thông thường ta sẽ truyền giá trị vào thông qua các tham số. Do vậy, cần phải Add các tham số (parameter) này trước khi thực hiện thao tác Update.

**Ví dụ 1:** Hiển thị bảng tblUser trong gridview, nhưng có thêm chức năng cập nhật.



Giao diện trang web

Để có thể cập nhật được thực sự vào CSDL, thực hiện mấy công việc sau:

1. Hiện thị cột Update, trong gridview: Đặt thuộc tính `AutoGenerateEditButton="true"`.
2. Thêm thuộc tính `UpdateCommand` cho `SqlDataSource`
3. Truyền tham số và giá trị cho các trường mà ta muốn cập nhật.
4. Gọi phương thức `Update` của `SqlDataSource`.

Trong đó 1) và 2) viết trong trang ASPX; 3) 4) viết trong sự kiện `Row_Updating` của `GridView`.

#### Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UpdateDeleteUser.aspx.cs" Inherits="Lesson_15_DisplaytblUser" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Cập nhật người dùng</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:SqlDataSource runat="server" ID="DSND"
      ProviderName="System.Data.OleDb"
      ConnectionString="<%%$ ConnectionStrings:QLCB %>"
      SelectCommand="Select TenDangNhap, MatKhau from tbluser"
      UpdateCommand="Update tblUser set MatKhau=@MatKhau where
        TenDangNhap=@TenDangNhap"
    </asp:SqlDataSource>

    <h2>Cập nhật thông tin người dùng</h2>
```

```
<asp:GridView runat="server" ID="dgrDS"
    DataKeyNames="TenDangNhap"
    DataSourceID="DSND"
    AutoGenerateEditButton="true"
    onrowupdating="dgrDS_RowUpdating" >
</asp:GridView>
</form>
</body>
</html>
```

#### Trang code

```
using System;
using System.Data;
using System.Data.OleDb;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;

public partial class Lesson_15_DisplaytblUser : System.Web.UI.Page
{
    protected void dgrDS_RowUpdating (object sender, GridViewUpdateEventArgs e)
    {
        string TenDN;
        string MKMoi;

        TenDN = e.NewValues ["TenDangNhap"].ToString ();
        MKMoi = e.NewValues ["MatKhai"].ToString ();

        // Tạo 2 tham số với giá trị tương ứng là TenDN và MKMoi,
        // Sau đó add vào UpdateCommand của đối tượng SQLDataSource

        DSND.UpdateParameters.Add ("MatKhai", MKMoi);
        DSND.UpdateParameters.Add ("TenDangNhap", TenDN);

        DSND.Update ();
    }
}
```

**Phân tích:** Trong thuộc tính UpdateCommand ở trên, ta có dòng:

**UpdateCommand="Update tblUser set MatKhai=@MatKhai where**

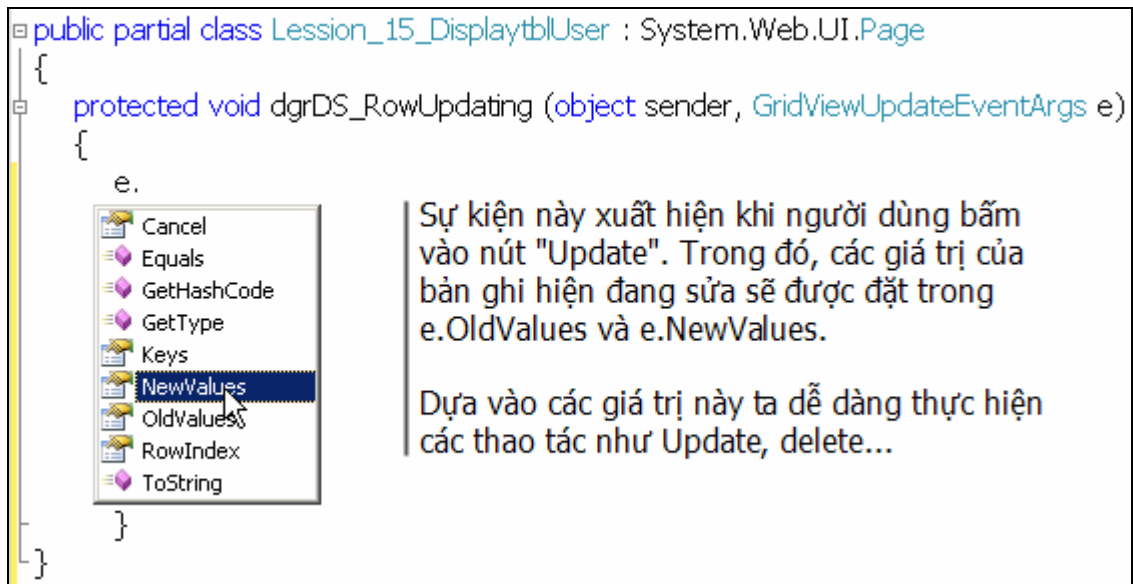
**TenDangNhap=@TenDangNhap"**

Ở đây @MatKhai và @TenDangNhap được gọi là các tham số. Tham số này rất đa dạng, nó có thể là nội dung của một textbox hay cũng có thể do ta tạo ra tường minh bằng câu lệnh dạng: **SqlDataSource.UpdateCommand.Parameters.Add("Tên","Giá trị") ....**

Trong trường hợp này ta sẽ tạo bằng phương thức **Add**, sau đó truyền giá trị là nội dung mà người dùng vừa sửa đổi.

**Câu hỏi :** Làm thế nào để lấy được giá trị mà người dùng vừa sửa ?.

Rất may cho chúng ta là khi người dùng sửa đổi nội dung và bấm vào nút "Update" (bên cạnh Gridview) thì Gridview sẽ Postback trở lại Server và kích hoạt sự kiện **RowUpdating**:



Khi postback (gửi về) server, GridView sẽ gửi kèm các thông tin về hàng (bản ghi) hiện đang được sửa chữa, cụ thể gồm: Các giá trị cũ (OldValues), các giá trị mới (NewValues), chỉ số của hàng đang sửa và có thể cả giá trị khóa của bản ghi (nếu trong GridView ta đặt thuộc tính DataKeyNames)

Để truy xuất đến các giá trị mới/cũ này ta viết: e.OldValues[Chỉ số / tên trường], e.NewValues[Chỉ số hoặc tên trường]. Để truy xuất đến giá trị khóa của bản ghi hiện hành, ta viết e.Keys[Chỉ số / hoặc tên trường] (Ví dụ e.Keys[MaSanPham]...)

Sau khi lấy được các giá trị này, ta sẽ tạo ra parameters tương ứng và gọi phương thức Update() của điều khiển SqlDataSource.

#### 15.3.4 Xóa bản ghi trong CSDL bằng SqlDataSource

Để xóa bản ghi, ta cũng tiến hành tương tự như khi cập nhật, đó là phải thêm thuộc tính DeleteCommand cho điều khiển SqlDataSource, tạo và truyền tham số trong sự kiện RowDeleting. Như vậy, nội dung trang web ở trên sẽ được bổ sung thêm như sau:

##### Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UpdateDeleteUser.aspx.cs" Inherits="Lesson_15_DisplaytblUser" %>
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title>Cập nhật người dùng</title>
</head>
<body>
<form id="form1" runat="server">
<asp:SqlDataSource runat="server" ID="DSND"
ProviderName="System.Data.OleDb"
ConnectionString="<%= $ConnectionStrings:QLCB %>"
SelectCommand="Select TenDangNhap, MatKhai from tbluser"
UpdateCommand="Update tbluser set MatKhai=@MatKhai where
TenDangNhap=@TenDangNhap"
DeleteCommand="delete from tblUser where
TenDangNhap=@TenDangNhap">
</asp:SqlDataSource>
```



```
<h2>Cập nhật thông tin người dùng</h2>
<asp:GridView runat="server" ID="dgrDS"
    DataKeyNames="TenDangNhap"
    DataSourceID="DSND"
    AutoGenerateEditButton="true"
    onrowupdating="dgrDS_RowUpdating" >
</asp:GridView>
</form>
</body>
</html>
```

#### Trang code

```
using System;
using System.Data;
using System.Data.OleDb;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;

public partial class Lesson_15_DisplaytblUser : System.Web.UI.Page
{
    protected void dgrDS_RowDeleting (object sender, GridViewDeleteEventArgs e)
    {
        string TenDN = e.Keys ["TenDangNhap"].ToString (); ///Lấy giá trị khóa
        DSND.DeleteParameters.Add ("TenDangNhap", TenDN);
        DSND.Delete ();
    }

    protected void dgrDS_RowUpdating (object sender, GridViewUpdateEventArgs e)
    {
        string TenDN, MKMoi;

        TenDN = e.NewValues ["TenDangNhap"].ToString ();
        MKMoi = e.NewValues ["MatKhau"].ToString ();

        // Tạo 2 tham số với giá trị tương ứng là TenDN và MKMoi,
        // Sau đó add vào UpdateCommand của đối tượng SQLDataSource
        DSND.UpdateParameters.Add ("MatKhau", MKMoi);
        DSND.UpdateParameters.Add ("TenDangNhap", TenDN);
        DSND.Update ();
    }
}
```

**Chú ý:** Thứ tự thêm tham số phải giống như thứ tự sử dụng tham số trong thuộc tính UpdateCommand, DeleteCommand của SqlDataSource.

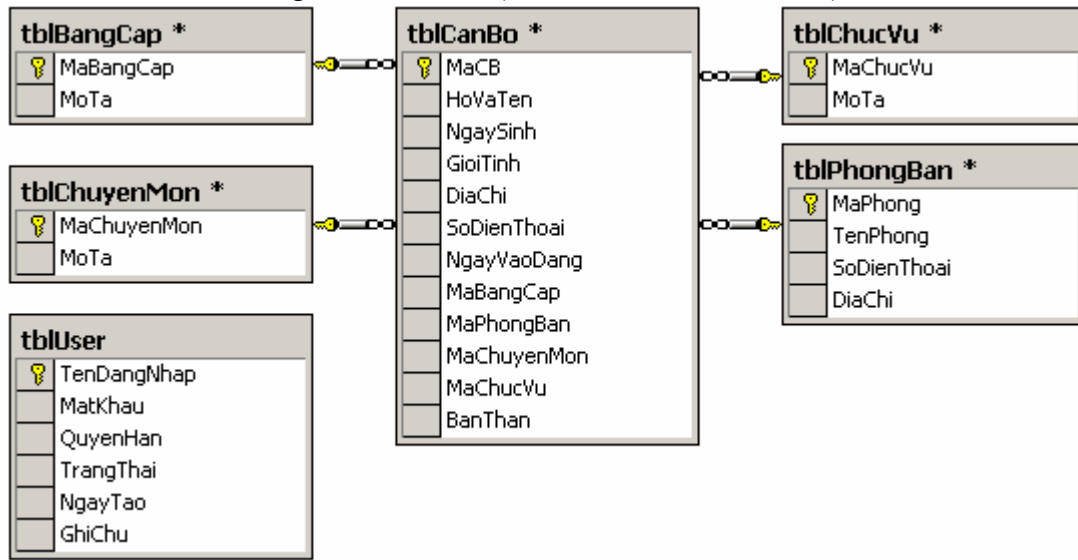
## BÀI 16: THỰC HÀNH

**Mục tiêu:** Kết thúc bài thực hành, học viên có thể:

- ❖ Đọc và hiển thị dữ liệu sử dụng cơ chế DataBinding
- ❖ Đọc và hiển thị dữ liệu sử dụng cơ chế DataBinding và điều khiển SqlDataSource
- ❖ Cập nhật dữ liệu sử dụng SqlDataSource và GridView.

**Nội dung:**

Cơ sở dữ liệu được sử dụng: QLCB.MDF (cơ sở dữ liệu SQL Server)



**Bài 1.1:** Hiển thị danh sách cán bộ (bao gồm Họ tên, ngày sinh, giới tính, địa chỉ, điện thoại) trong một Gridview, sử dụng cơ chế DataBinding.

**Hướng dẫn:** Đọc dữ liệu bằng DataReader và Command. Sau đó "Gắn" GridView với DataReader bằng cách đặt thuộc tính DataSource của GridView:

### Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="DSCB_DataBinding.aspx.cs" Inherits="Lesson_17_DSCB_DataBinding" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title>Danh sách cán bộ - version 1.0</title>
</head>
<body>
<form id="form1" runat="server">
<h3>Danh sách cán bộ<hr />
<asp:Button ID="cmdDisplay" runat="server" onclick="cmdDisplay_Click"
Text="Hiển thị" />
</h3>
<asp:GridView runat="server" ID="dgrDSCB">
</asp:GridView>
</form>
</body>
</html>
```

```
Trang Code
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class Lesson_17_DSCB_DataBinding : System.Web.UI.Page
{
    private SqlConnection Conn = new SqlConnection ();
    private SqlCommand Cmd = new SqlCommand ();
    private SqlDataReader Dr;

    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void cmdDisplay_Click (object sender, EventArgs e)
    {
        Conn.ConnectionString =
        System.Configuration.ConfigurationManager.ConnectionStrings ["strConn"].ToString();
        Conn.Open ();

        Cmd.CommandText = "Select HoVaTen, NgaySinh, GioiTinh, DiaChi, SoDienThoai from tblCanBo";
        Cmd.Connection = Conn;

        Dr=Cmd.ExecuteReader ();

        //Gắn kết với Gridview để hiển thị
        dgrDSCB.DataSource = Dr;
        dgrDSCB.DataBind ();

        Cmd.Dispose ();
        Conn.Close ();
    }
}
```

**Bài 1.1** Hiển thị thông tin về người dùng (Gồm họ tên, bằng cấp, chức vụ) mỗi khi người dùng chọn Đơn vị trong một Dropdownlist.

Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="DSCB_PhongBan.aspx.cs" Inherits="Lesson_17_DSCB_PhongBan" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Thông tin về cán bộ theo phòng ban</title>
</head>
<body>
    <form id="form1" runat="server">
        <h3>Danh sách phòng</h3>
```

```
<asp:DropDownList runat="server" ID="ddlDSPhong" AutoPostBack="true">
</asp:DropDownList>

<br />
<h3>Danh sách cán bộ trực thuộc</h3>
<asp:GridView runat="server" ID="dgrDSCB"></asp:GridView>
</form>
</body>
</html>
```

#### Trang code

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;

public partial class Lesson_17_DSCB_PhongBan : System.Web.UI.Page
{
    private SqlConnection Conn = new SqlConnection ();
    private SqlCommand Cmd = new SqlCommand ();
    private SqlDataReader Dr;

    protected void Page_Load(object sender, EventArgs e)
    {
        Conn.ConnectionString =
            System.Configuration.ConfigurationManager.ConnectionStrings ["strConn"].ToString ();
        Conn.Open ();

        if (IsPostBack == false)
        {
            Cmd.CommandText = "Select MaPhong from tblPhongBan";
            Cmd.Connection = Conn;

            Dr = Cmd.ExecuteReader ();
            ddlDSPhong.DataSource = Dr;
            ddlDSPhong.DataTextField = "MaPhong";
            ddlDSPhong.DataBind ();
        }
        else
        {
            Cmd.CommandText="Select tblCanBo.HoVaTen, tblBangCap.MoTa, tblChucVu.MoTa " +
                " FROM tblCanBo, tblBangCap, tblChucVu " +
                "WHERE tblCanBo.MaBangCap = tblBangCap.MaBangCap and " +
                " tblCanBo.MaChucVu=tblChucVu.MaChucVu and " +
                " tblCanBo.MaPhongBan = '" + ddlDSPhong.Text + "'";
            Cmd.Connection = Conn;
        }
    }
}
```

```

        Dr = Cmd.ExecuteReader ();
        dgrDSCB.DataSource = Dr;
        dgrDSCB.DataBind ();
    }

    Cmd.Dispose ();
    Conn.Close ();
}

```

**Bài 2:** Xây dựng các trang cho phép cập nhật thông tin về người dùng, phòng ban, chức vụ, chuyên môn, bằng cấp. Sử dụng cơ chế DataBinding, kết hợp với GridView tương tự như trong bài lý thuyết.

**Bài 3:** bổ sung vào các trang ở trên chức năng "Thêm mới". để cho phép thêm mới các bản ghi. Giao diện có dạng như sau:

**Danh sách cán bộ**

HoVaTen	NgaySinh	GioiTinh	DiaChi	SoDienThoai
Nguyễn Minh Quý		<input checked="" type="checkbox"/>	Hưng Yên	0321-713319
Lê Quang Lợi		<input checked="" type="checkbox"/>	Hưng Yên	
Phạm Ngọc Hưng		<input checked="" type="checkbox"/>	Quảng Ninh	
Nguyễn Đình Hân		<input checked="" type="checkbox"/>	Hưng Yên	
Đặng Bá Hùng		<input checked="" type="checkbox"/>	Hải Dương	
Nguyễn Thị Hiền		<input type="checkbox"/>	Hưng yên	012344567
Nguyễn Văn Huyền		<input checked="" type="checkbox"/>	Hưng Yên	213424324

Họ tên:   
 Mật khẩu:   
 Quyền hạn :   
 Trạng thái: ☒

## BÀI 17: Làm việc với GridView

### 17.1 Giới thiệu tổng quan

GridView có lẽ là một điều khiển trình diễn dữ liệu quan trọng nhất của ASP.NET. Nó cho phép gắn và hiển thị dữ liệu ở dạng bảng, trong đó mỗi hàng là một bản ghi, mỗi cột ứng với một trường dữ liệu. Ngoài việc hiển thị, GridView còn có rất nhiều tính năng khác mà trước đây người ta phải viết rất nhiều dòng code mới có được, ví dụ: Định dạng, phân trang, sắp xếp, sửa đổi, xóa dữ liệu.

GridView có thể gắn kết dữ liệu với các nguồn như DataReader, SqlDataSource, ObjectDataSource hay bất kỳ nguồn nào có cài đặt System.Collections.Enumerable.

Trong bài học này, chúng ta sẽ đi tìm hiểu và sử dụng một số tính năng nổi bật của GridView, từ đó có thể áp dụng làm Project cho môn học.

### 17.2 Tìm hiểu lớp GridView

#### 17.2.1 Các thuộc tính và cột thuộc tính

GridView ngoài việc hiển thị thuần túy các trường của một nguồn dữ liệu, nó còn cho phép hiển thị dưới các hình thức khác (ví dụ dưới dạng nút, dạng HyperLink, dạng checkbox...), các cột khác hỗ trợ cho việc thao tác dữ liệu như Select, Update, Delete hoàn toàn có thể tùy biến trong GridView.

Để chỉnh sửa các cột dạng này, click chọn "smart tag" của GridView và chọn "Edit Field" hoặc chọn thuộc tính Columns của GridView trong cửa sổ thuộc tính.

Loại cột	Mô tả
<b>BoundField</b>	Hiển thị giá trị của một trường thuộc nguồn dữ liệu.
<b>ButtonField</b>	Hiển thị một nút lệnh cho mỗi mục trong GridView. Nút này cho phép bạn có thể tạo ra các nút tùy biến kiểu như Add hay Remove.
<b>CheckBoxField</b>	Hiển thị một checkbox ứng với mỗi mục trong GridView. Cột này thường được dùng để hiển thị các trường kiểu Boolean (Yes/No).
<b>CommandField</b>	Hiển thị các nút lệnh đã được định nghĩa sẵn để thực hiện các thao tác select, edit, hay delete.
<b>HyperLinkField</b>	Hiển thị giá trị của một trường dưới dạng siêu liên kết (hyperlink). Loại cột này cho phép bạn gắn một trường thứ hai vào URL của siêu liên kết.
<b>ImageField</b>	Hiển thị một ảnh ứng với mỗi mục trong GridView.
<b>TemplateField</b>	Hiển thị nội dung tùy biến của người dùng cho mỗi mục dữ liệu trong GridView, theo như mẫu định sẵn. Loại cột này cho phép ta tạo ra các cột tùy biến.

**Các thuộc tính:**

Thuộc tính	Mô tả
<b>GridLines</b>	Ẩn, hiện các đường viền của GridView.
<b>ShowHeader</b>	Cho phép Hiện/ ẩn phần Header
<b>ShowFooter</b>	Cho phép Hiện/ ẩn phần Footer
<b>PageSize</b>	Get/Set cho biết mỗi trang chứa bao nhiêu dòng.
<b>PageCount</b>	Cho biết số trang của nguồn dữ liệu mà nó gắn kết
<b>PageIndex</b>	Get/Set chỉ số của trang đang được hiển thị
<b>AllowPaging</b>	Có cho phép phân trang không ( true = có)
<b>AllowSorting</b>	Có cho phép sắp xếp không (true=có)
<b>AutoGenerateColumns</b>	Có tự động sinh ra các cột ứng với các cột trong nguồn dữ liệu hay không ? Mặc định = true (có)
<b>AutoGenerateDeleteButton</b>	Tự động tạo ra cột Delete (true = tự động)
<b>AutoGenerateUpdateButton</b>	Tự động tạo ra cột Update (true = tự động)
<b>AutoGenerateSelectButton</b>	Tự động tạo ra cột Select (true = tự động)
<b>EditIndex</b> <b>(SelectedIndex)</b>	Đặt hàng nào đó về chế độ edit. EditIndex = 2 → hàng thứ 3 (chỉ số 2) sẽ về chế độ edit. Nếu đặt EditIndex = -1 thì sẽ thoát khỏi chế độ Edit.
<b>SelectedIndex</b>	Trả về chỉ số của dòng đang chọn
<b>Rows</b>	Một tập hợp chứa các hàng của GridView.
<b>Columns</b>	Một tập hợp chứa các cột của GridView.

### 17.2.2 Các style áp dụng cho GridView

GridView rất linh hoạt trong việc trình bày dữ liệu, nó cho phép ta định dạng các phần thông qua style. Ví dụ ta có thể định dạng cho phần Header, footer, các mục dữ liệu, các hàng chẵn-lẻ v.v...

Bảng dưới đây sẽ giải thích rõ ý nghĩa một số thuộc tính:

Thuộc tính style	Mô tả
<b><u>AlternatingRowStyle</u></b>	Style áp dụng cho các hàng dữ liệu chẵn-lẻ trong GridView. Khi đặt thuộc tính này thì các hàng sẽ được hiển thị với định dạng luân phiên giữa <u>RowStyle</u> và <u>AlternatingRowStyle</u> .
<b><u>EditRowStyle</u></b>	Style để hiển thị hàng hiện đang được sửa (Edit).
<b><u>FooterStyle</u></b>	Style áp dụng cho phần Footer.
<b><u>HeaderStyle</u></b>	Style áp dụng cho phần Header.
<b><u>PagerStyle</u></b>	Style áp dụng cho phần phân trang (các trang << 1 2 3 ... >>).
<b><u>RowStyle</u></b>	Style áp dụng cho các hàng dữ liệu trong GridView control. Khi <u>AlternatingRowStyle</u> được thiết lập thì sẽ áp dụng luân phiên giữa <u>RowStyle</u> và <u>AlternatingRowStyle</u> .
<b><u>SelectedRowStyle</u></b>	Style áp dụng cho hàng đang được chọn (Selected)của <b>GridView</b> .

### 17.2.3 Các sự kiện

GridView có rất nhiều sự kiện quan trọng, các sự kiện này khi kích hoạt sẽ cung cấp cho ta những thông tin hữu ích trong quá trình xử lý. Ví dụ, khi chúng ta click nút Update, nó sẽ kích hoạt sự kiện Updating và trả về cho chúng ta các giá trị mà người dùng vừa sửa....

Dưới đây là bảng tổng hợp một số sự kiện hay dùng nhất:

Tên sự kiện	Mô tả
<b><u>PageIndexChanged</u></b>	Xuất hiện khi ta click chọn các nút ( << 1 2 3 >>) trong hàng phân trang.
<b><u>PageIndexChanging</u></b>	Xuất hiện khi người dùng click chọn các nút ( << 1 2 3 >>) trong hàng phân trang nhưng TRƯỚC khi GridView thực hiện việc phân trang. Ta có thể hủy việc phân trang tại sự kiện này.
<b><u>RowCancelingEdit</u></b>	Xuất hiện khi nút Cancel được click nhưng trước khi thoát khỏi chế độ Edit.
<b><u>RowCommand</u></b>	Xuất hiện khi một nút được click.
<b><u>RowCreated</u></b>	Xuất hiện khi một hàng mới được tạo ra. Thường được sử dụng để sửa nội dung của hàng khi nó vừa được tạo ra.
<b><u>RowDataBound</u></b>	Xuất hiện khi một hàng dữ liệu được gắn vào GridView. Tại đây ta có thể sửa đổi nội dung của hàng đó.



<b><u>RowDeleted</u></b>	Xuất hiện khi nút Delete của một hàng được click, nhưng <b>sau khi</b> GridView đã delete bản ghi từ nguồn.
<b><u>RowDeleting</u></b>	Xuất hiện khi nút Delete được click nhưng <b>trước khi</b> GridView xóa bản ghi từ nguồn. Tại đây có thể Cancel việc Delete.
<b><u>RowEditing</u></b>	Xuất hiện khi nút Edit được click, nhưng <b>trước khi</b> GridView về chế độ sửa.
<b><u>RowUpdated</u></b>	Xuất hiện khi nút Update được click, nhưng <b>sau khi</b> GridView update hàng dữ liệu.
<b><u>RowUpdating</u></b>	Xuất hiện khi nút Update được click, nhưng <b>trước khi</b> GridView update hàng dữ liệu.
<b><u>SelectedIndexChanged</u></b>	Xuất hiện khi nút Select của hàng được click nhưng <b>sau khi</b> GridView xử lý xong thao tác Select.
<b><u>SelectedIndexChanging</u></b>	Xuất hiện khi nút Select của hàng được click nhưng <b>trước khi</b> GridView xử lý xong thao tác Select.
<b><u>Sorted</u></b>	Xuất hiện khi Hyperlink (tiêu đề cột) được click, nhưng <b>sau khi</b> GridView thực hiện việc sắp xếp.
<b><u>Sorting</u></b>	Xuất hiện khi Hyperlink (tiêu đề cột) được click, nhưng <b>trước khi</b> GridView thực hiện việc sắp xếp. Sự kiện này khi xảy ra, nó sẽ cung cấp cho chúng ta thông tin về tên cột vừa được click. Dựa vào đó ta có thể thực hiện việc sắp xếp một cách dễ dàng.

#### 17.2.4 Các phương thức

Tên phương thức	Mô tả
<b>DataBind()</b>	Gắn kết dữ liệu giữa GridView và nguồn dữ liệu (đặt các thuộc tính DataSource, DataTextField hoặc DataSourceID).
<b>DeleteRow(int)</b>	Xóa một dòng trong GridView
<b>UpdateRow(int i, bool Valid)</b>	Cập nhật một dòng trong GridView.
<b>Sort(Biểu thức sx, hướng sx)</b>	Sắp xếp dựa trên biểu thức và hướng.

## 17.3 Các tính năng hỗ trợ của GridView

### 17.3.1 Phân trang

Để thực hiện phân trang, cần đặt thuộc tính AllowPaging = True. Khi phân trang, có thể tùy biến hiển thị các trang (hiển thị dạng các số 1 2 3 hay mũi tên < >) bằng cách đặt các thuộc tính con trong PagerSettings.

#### Ví dụ:

Hiển thị tất cả người dùng trong bảng tblUser trong GridView nhưng sử dụng cơ chế phân trang. Mỗi trang gồm 5 bản ghi.

#### Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="AllUserwithPaging.aspx.cs" Inherits="Lesson_18_AllUserwithPaging" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Danh sách người dùng</title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:SqlDataSource ID="SqlDataSource1" runat="server"
            ConnectionString="<%%$ ConnectionStrings:QLCBCConnectionString %>"
            SelectCommand="SELECT * FROM tblUser">
        </asp:SqlDataSource>

        <asp:GridView DataSourceID="SqlDataSource1" runat="server"
            AllowPaging="true"
            PageSize="5"
            PagerStyle-HorizontalAlign="Center"
            PagerSettings-Mode="NumericFirstLast">
        </asp:GridView>

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

Nếu gắn kết gridView với DataReader/ DataTable/ DataSet thì cần phải chỉ định rõ chỉ số trang cần hiển thị khi sự kiện **PageIndexChanging** được kích hoạt, cụ thể như sau:

#### Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="PagingwithDataReaderBinding.aspx.cs"
Inherits="Lesson_18_PagingwithDataReaderBinding" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Danh sách người dùng - Paging với DataSet</title>
</head>
<body>
```

```
<form id="form1" runat="server">
  <asp:Button runat="server" ID="cmdHienThi" Text="Hiển thị"
    onclick="cmdHienThi_Click" />

  <asp:GridView runat="server" ID="dgrDSND"
    AllowPaging="true"
    AutoGenerateColumns="true"
    PagerStyle-HorizontalAlign="Center"
    PagerSettings-Mode="NumericFirstLast"
    onpageindexchanging="dgrDSND_PageIndexChanging">
  </asp:GridView>
</form>
</body>
</html>
```

#### Trang Code behind

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;

public partial class Lesson_18_PagingwithDataReaderBinding : System.Web.UI.Page
{
    protected void NapDuLieu()
    {
        //Kết nối đến csdl SQL. Chú ý lệnh đọc sâu kết nối trong tệp web.config
        SqlConnection Cn = new SqlConnection ();
        Cn.ConnectionString=ConfigurationManager.ConnectionStrings ["QLCBConnectionString"].ToString();
        Cn.Open ();

        SqlDataAdapter Da = new SqlDataAdapter ("Select * from tblUser", Cn);
        DataSet Ds = new DataSet ();
        Da.Fill (Ds, "DSND");

        /// Bind dữ liệu vừa lấy được vào GridView để hiển thị
        dgrDSND.DataSource = Ds.Tables ["DSND"];
        dgrDSND.DataBind ();

        //Giải phóng khi không còn sử dụng
        Ds.Dispose ();
        Da.Dispose ();
        Cn.Close ();
    }

    protected void cmdHienThi_Click (object sender, EventArgs e)
    {
        NapDuLieu (); //Gọi hàm nạp dữ liệu
    }

    //Sự kiện này kích hoạt khi người dùng click số trang trên GridView
    // Khi người dùng click trang nào thì ta hiển thị trang đó trên browser
}
```

```
// Chỉ số của trang vừa chọn nằm trong e.NewPageIndex
protected void dgrDSND_PageIndexChanging (object sender, GridViewPageEventArgs e)
{
    dgrDSND.PageIndex = e.NewPageIndex; // Đặt lại chỉ số trang cần hiển thị
    NapDuLieu ();                       // Nạp lại dữ liệu
}
}
```

### 17.3.2 Tính năng tự động sắp xếp

Tính năng này cho phép dữ liệu trong GridView sẽ tự động được sắp xếp theo giá trị của cột mà người dùng click. Ở đây ta có thể sắp xếp theo chiều tăng (Ascending) hoặc giảm (Descending).

Để bật tính năng này, cần đặt thuộc tính AllowSorting = true trong GridView.

Khi người dùng click chuột vào một cột tiêu đề nào đó của GridView thì sự kiện Sorting sẽ được kích hoạt, tại đây ta cần phải chỉ rõ cho GridView biết là sắp theo cột nào (SortExpression) và theo chiều tăng hay giảm (SortDirection).

#### Ví dụ:

Hiện thị danh sách người dùng trong bảng tblUser, khi người dùng click vào một cột thì sắp theo chiều tăng dần.

#### Trang giao diện

```
%@ Page Language="C#" AutoEventWireup="true" CodeFile="Sorting.aspx.cs"
Inherits="Lesson_18_AllUserwithPaging" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Danh sách người dùng</title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:SqlDataSource ID="SqlDataSource1" runat="server"
            ConnectionString="<%"$ ConnectionStrings:QLCBConnectionString %>"
            SelectCommand="SELECT * FROM tblUser">
        </asp:SqlDataSource>

        <asp:GridView DataSourceID="SqlDataSource1" runat="server" id="dgrDSND"
            AllowPaging="True"
            AllowSorting="True"
            PageSize="5"
            PagerStyle-HorizontalAlign="Center"
            PagerSettings-Mode="NumericFirstLast"
        </asp:GridView>
    </form>
</body>
</html>
```

#### Trang code

```
using System;
using System.Collections;
using System.Configuration;
public partial class Lesson_18_AllUserwithPaging : System.Web.UI.Page
```

```
{
    protected void dgrDSND_Sorting (object sender, GridViewSortEventArgs e)
    {
        e.SortDirection= SortDirection.Descending;
    }
}
```

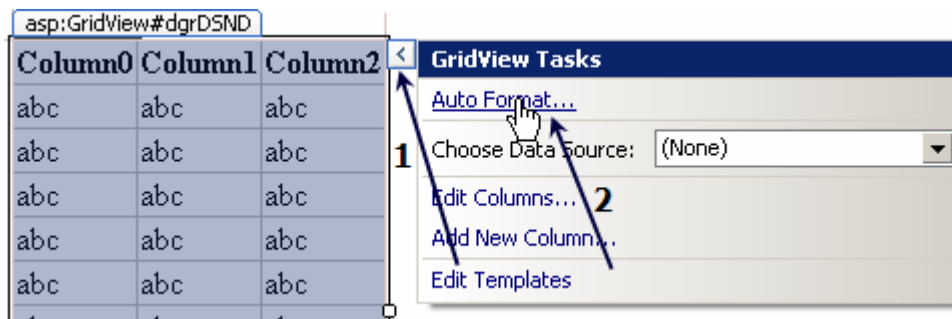
### 17.3.3 Các mẫu hiển thị - Template

ASP.NET cung cấp cho chúng ta sẵn một số Template (mẫu) để hiển thị GridView cũng khá đẹp. Vì vậy, bạn có thể sử dụng ngay các template này khi xây dựng ứng dụng.

Cách thức chọn template cho GridView như sau:

b1: Mở trang ở chế độ Design

b2: Chọn GridView và chọn smart tag, tiếp theo chọn AutoFormat



b3: Chọn Format trong danh sách.

Hiển thị						
TenDangNhap	HoVaTen	MatKhau	QuyenHan	TrangThai	NgayTao	GhiChu
loilq	Lê Quang Lợi	123456	4	1	2/8/2008	Staff
longpt	Phí Đức Long	123456	2	1	2/8/2008	Staff
nguyetdt	Đinh Thị Nguyệt	123456	5	1	2/8/2008	Staff
quynm	Nguyễn Minh Quý	123456	2	1	1/1/2008	Lecturer
tannd	Nguyễn Duy Tân	123456	1	1	2/8/2008	Staff
user	user	123	1	1	2/8/2008	Staff
uyenpt	Phạm Thị Uyên	123456	5	1	2/8/2008	Staff
1 2						

**Tổ hợp màu được chọn từ Template có sẵn.**

Sau khi chọn Template, Asp sẽ tự động tạo ra các thuộc tính (thẻ) tương ứng trong GridView, tại đây bạn có thể tiếp tục tùy biến thêm theo như ý muốn.

## 17.4 Tạo các cột tùy biến HyperLink, BoundColumn...

### 17.4.1 Tạo cột BoundField thủ công

Để tạo các cột thủ công, cần đặt thuộc tính AutoGenerateColumns = "False", sau đó soạn thủ công các cột trong cửa sổ Edit Columns.

*Ví dụ: Hiển thị danh sách người dùng nhưng các cột tạo thủ công*

Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UserList_ColumnManual.aspx.cs" Inherits="Lesson_18_AllUserwithPaging" %>
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Danh sách người dùng</title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:SqlDataSource ID="SqlDataSource1" runat="server"
            ConnectionString="<%$ ConnectionStrings:QLCBConnectionString %>"
            SelectCommand="SELECT * FROM tblUser">
        </asp:SqlDataSource>

        <asp:GridView DataSourceID="SqlDataSource1" runat="server"
            AllowPaging="True"
            PageSize="5"
            PagerStyle-HorizontalAlign="Center"
            PagerSettings-Mode="NumericFirstLast" AllowSorting="True"
            AutoGenerateColumns="False">
            <Columns>
                <asp:BoundField DataField="TenDangNhap" HeaderText="Tên đăng nhập"
                    SortExpression="tendangnhap" />

                <asp:BoundField DataField="HoVaTen" HeaderText="Họ và tên"
                    SortExpression="HoVaTen" />

                <asp:BoundField DataField="MatKhai" HeaderText="Mật khẩu"
                    SortExpression="MatKhai" />
            </Columns>
        </asp:GridView>
    </form>
</body>
</html>
```

### 17.4.2 Tạo một cột hyperlink,

Hiển thị danh sách người dùng (bảng tbluser) trong đó có thêm cột "**Chi tiết**" để khi người dùng click chuột vào hyperlink này thì chuyển đến trang hiển thị chi tiết người dùng. Trong ASP.NET, GridView có khả năng hiển thị (render) các trường có chứa HyperLink thành các thẻ <a href ...> trên trình duyệt. Do vậy, về ý tưởng chúng ta cần phải tạo một cột mới có chứa sẵn Hyperlink sau đó "Chèn" trường này vào cột Hyperlink của GridView. Chú ý: Trong SQL, thông thường trong câu lệnh SELECT chúng ta chỉ chọn các trường sẵn có trong bảng CSDL, tuy nhiên hoàn toàn có thể tạo ra một cột mới kiểu như sau:

SELECT Ten + Ho as **HoVaTen** FROM ...

**Trong đó:** Ten, Ho là 2 trường của bảng, HoVaTen là một cột mới (Do ta tạo ra ngay trong câu lệnh SELECT, còn trong bảng CSDL thì không có trường này)

*Hay một ví dụ khác:*

SELECT TenHang, NgayXua, SoLuong, DonGia, SoLuong \* DonGia As **ThanhTien** ...  
Trường **ThanhTien** là một trường mới. Giá trị của nó bằng giá trị SoLuong \* DonGia của bản ghi hiện tại.

Để tạo cột hiển thị được HyperLink, GridView cung cấp thẻ : **<asp: HyperLinkField>**

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UserList_ColumnHyperlink.aspx.cs" Inherits="Lesson_18_AllUserwithPaging"
%>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head runat="server">
```

```
<title>Danh sách người dùng</title>
```

```
</head>
```

```
<body>
```

```
<form id="form1" runat="server">
```

```
<asp:SqlDataSource ID="SqlDataSource1" runat="server"
```

```
ConnectionString="<%$ ConnectionStrings:QLCBConnectionString %>"
```

```
SelectCommand="SELECT TenDangNhap, HoVaTen, MatKhai,
'UserDetail.aspx?TenDangNhap=' + TenDangNhap as ChiTiet FROM tblUser">
```

```
</asp:SqlDataSource>
```

```
<asp:GridView DataSourceID="SqlDataSource1" runat="server"
```

```
AllowPaging="True" PageSize="5"
```

```
PagerStyle-HorizontalAlign="Center"
```

```
PagerSettings-Mode="NumericFirstLast" AllowSorting="True"
```

```
AutoGenerateColumns="False">
```

```
<Columns>
```

```
<asp:BoundField DataField="TenDangNhap" HeaderText="Tên đăng nhập"
SortExpression="tendangnhap" />
```

```
<asp:BoundField DataField="HoVaTen" HeaderText="Họ và tên"
SortExpression="HoVaTen" />
```

```
<asp:BoundField DataField="MatKhai" HeaderText="Mật khẩu"
SortExpression="MatKhai" />
```

```
<asp:HyperLinkField HeaderText="Chi tiết" DataNavigateUrlFields="ChiTiet"
Text="Xem chi tiết" />
```

```
</Columns>
```

```
</asp:GridView>
```

```
</form>
```

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

### 17.4.3 Tạo cột Image

Tương tự như cột HyperLink, GridView cũng có một cột chuyên để hiển thị hình ảnh (ImageField) nếu trường dữ liệu gắn với nó chứa đường dẫn đến ảnh nằm trong ứng dụng.

Để tạo cột cho phép hiển thị Image, dùng thẻ <asp:ImageField DataImageUrlField ../>

#### Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UserList_ColumnImage.aspx.cs" Inherits="Lesson_18_AllUserwithPaging" %>
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Danh sách người dùng</title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:SqlDataSource ID="SqlDataSource1" runat="server"
            ConnectionString="<%$ ConnectionStrings:QLCBConnectionString %>"
            SelectCommand="SELECT TenDangNhap, HoVaTen, MatKhau,
                HinhAnh FROM tblUser">
        </asp:SqlDataSource>

        <asp:GridView DataSourceID="SqlDataSource1" runat="server"
            AllowPaging="True"
            PageSize="5"
            PagerStyle-HorizontalAlign="Center"
            PagerSettings-Mode="NumericFirstLast" AllowSorting="True"
            AutoGenerateColumns="False" >

            <Columns>
                <asp:BoundField DataField="TenDangNhap" HeaderText="Tên đăng nhập"
                    SortExpression="tendangnhap" />
                <asp:BoundField DataField="HoVaTen" HeaderText="Họ và tên"
                    SortExpression="HoVaTen" />
                <asp:BoundField DataField="MatKhau" HeaderText="Mật khẩu"
                    SortExpression="MatKhau" />
                <asp:ImageField DataImageUrlField="HinhAnh" HeaderText="Hình ảnh">
            </asp:ImageField>
            </Columns>
        </asp:GridView>

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



## 17.5 Tạo và xử lý các cột **Select, Edit, Delete, Update...**

### 17.5.1 Thêm cột **Select, Edit - Update, Delete**

GridView không chỉ hiển thị được các bảng dữ liệu mà còn hỗ trợ rất tốt trong việc chỉnh sửa và xóa dữ liệu. Đặc biệt khi nguồn dữ liệu là `SqlDataSource` thì việc sửa và xóa hoàn toàn tự động, không cần phải viết bất kỳ dòng code nào. Để bật tính năng này, cần bổ sung thêm thuộc tính vào GridView với giá trị là `true` cho **`AutoGenerateSelectColumn`**, **`AutoGenerateEditColumn`**, **`AutoGenerateDeleteColumn`**. Cụ thể như sau:

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="AddEditUpdateDeleteColumn.aspx.cs"
Inherits="Lesson_18_AllUserwithPaging" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

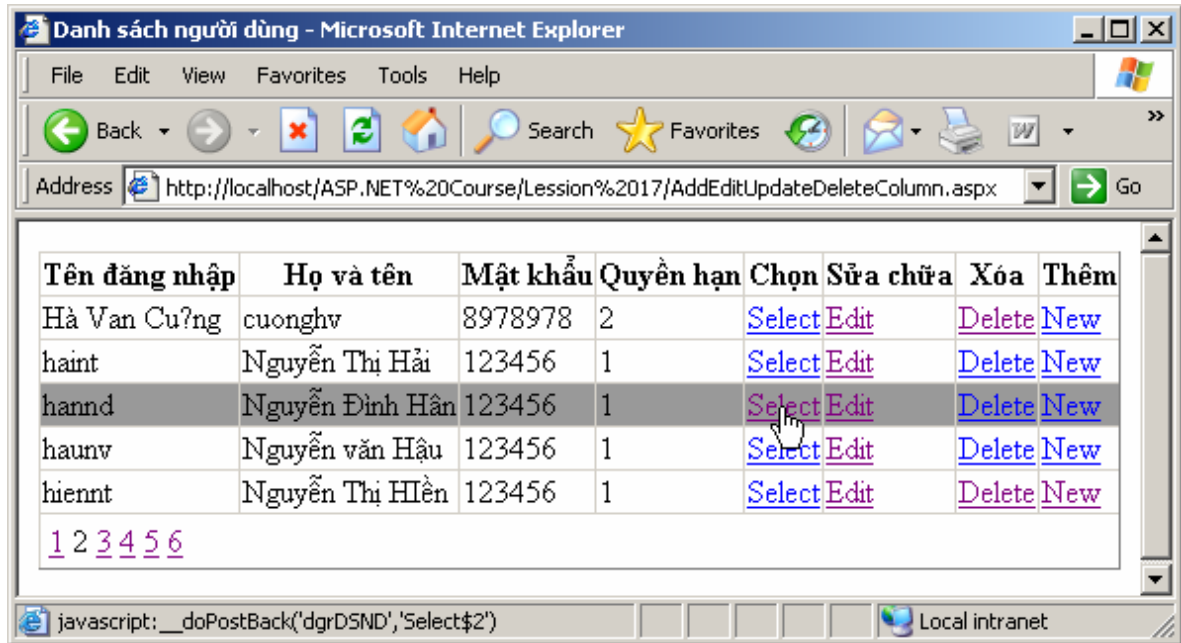
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Danh sách người dùng</title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:SqlDataSource ID="SqlDataSource1" runat="server"
            ConnectionString="<%$ ConnectionStrings:QLCBConnectionString %>"
            SelectCommand="SELECT TenDangNhap, HoVaTen, MatKhai FROM tblUser">
        </asp:SqlDataSource>

        <asp:GridView DataSourceID="SqlDataSource1" runat="server" ID="dgrDSND"
            AllowPaging="True" PageSize="5"
            AutoGenerateColumns="False"
            DataKeyNames="TenDangNhap">

            <Columns>
                <asp:BoundField DataField="TenDangNhap" HeaderText="Tên đăng nhập"/>
                <asp:BoundField DataField="HoVaTen" HeaderText="Họ và tên"/>
                <asp:BoundField DataField="MatKhai" HeaderText="Mật khẩu" />
                <asp:BoundField DataField="QuyềnHan" HeaderText="Quyền hạn" />

                <asp:CommandField HeaderText="Chọn" ShowSelectButton="True" />
                <asp:CommandField HeaderText="Sửa chữa" ShowEditButton="True"/>
                <asp:CommandField HeaderText="Xóa" ShowDeleteButton="True" />
                <asp:CommandField HeaderText="Thêm" ShowInsertButton="True" />
            </Columns>

            <SelectedRowStyle BackColor="#999999" />
        </asp:GridView>
    </form>
</body>
</html>
```



Hiển thị các cột lệnh (Select, edit, delete)

### 17.5.2 Cập nhật dữ liệu

**Ví dụ 1:** Xây dựng trang web cho phép hiển thị và cập nhật (Update) trường **Họ và tên** và **Mật khẩu** của bảng tblUser:

Các bước cần tiến hành:

- Tạo một nguồn dữ liệu SqlDataSource
- Thêm thuộc tính UpdateCommand với câu lệnh cập nhật Sql.
- Tạo GridView và đặt thuộc tính **DataKeyNames** = "Tên trường Khóa của bảng CSDL"
- Gắn kết GridView với SqlDataSource bằng cách đặt DataSourceID của GridView = ID của SqlDataSource.

#### Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
```

```
CodeFile="AddEditUpdateDeleteColumn.aspx.cs"
```

```
Inherits="Lesson_18_AllUserwithPaging" %>
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head runat="server">
```

```
<title>Danh sách người dùng</title>
```

```
</head>
```

```
<body>
```

```
<form id="form1" runat="server">
```

```
<asp:SqlDataSource ID="SqlDataSource1" runat="server"
```

```
ConnectionString="<%= $ ConnectionStrings:QLCBConnectionString %>"
```

```
UpdateCommand=
```

```
"UPDATE tblUser SET MatKhau=@MatKhau, QuyenHan=@QuyenHan  
WHERE TenDangNhap=@TenDangNhap">
```

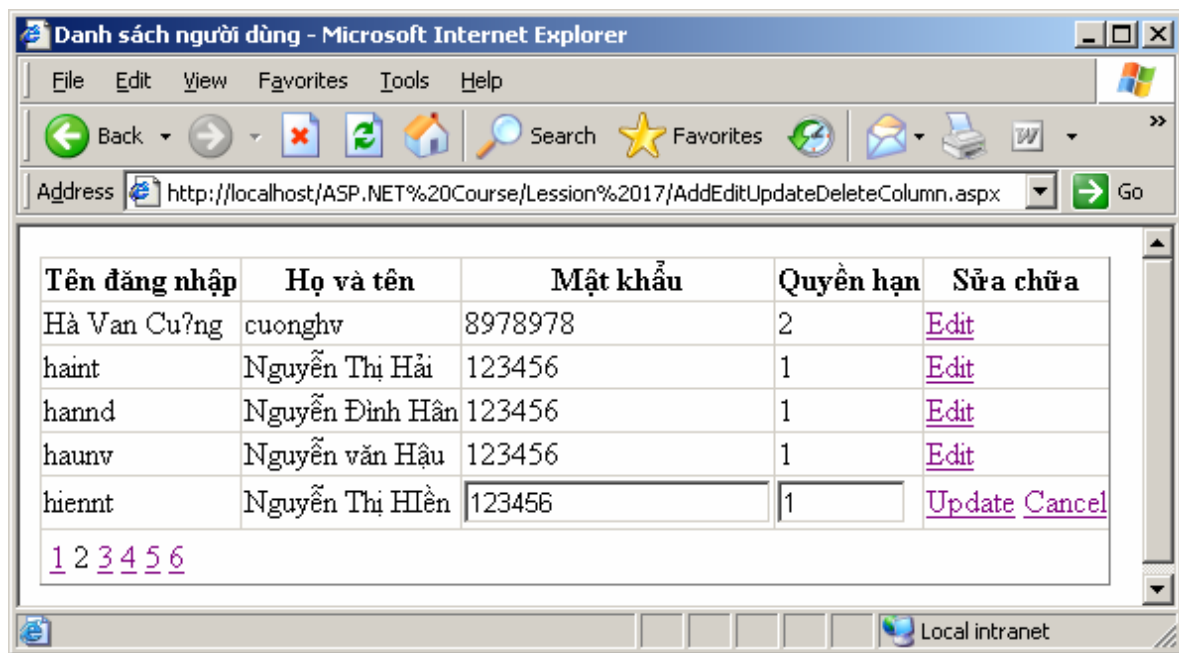
```
</asp:SqlDataSource>
```

```
<asp:GridView DataSourceID="SqlDataSource1" runat="server" ID="dgrDSND"
```

```
AllowPaging="True" PageSize="5"
```

```
AutoGenerateColumns="False"
DataKeyNames="TenDangNhap">

<Columns>
    <asp:BoundField DataField="TenDangNhap" HeaderText="TênĐN" ReadOnly="true"/>
    <asp:BoundField DataField="HoVaTen" HeaderText="Họ và tên" ReadOnly="true"/>
    <asp:BoundField DataField="MatKhau" HeaderText="Mật khẩu" />
    <asp:BoundField DataField="QuyenHan" HeaderText="Quyền hạn" />
    <asp:CommandField HeaderText="Sửa chữa" ShowEditButton="True"/>
</Columns>
</asp:GridView>
</form>
</body>
</html>
```



Kết quả sau khi chạy trang ở trên.

**Chú ý:** Khi sử dụng nguồn dữ liệu là SqlDataSource và trong câu lệnh Update/Delete nếu ta đặt tên các tham số giống với tên của các trường dữ liệu (ví dụ MatKhau=@MatKhau) thì SqlDataSource sẽ tự động tạo các tham số sau đó truyền giá trị mà người dùng vừa mới nhập giúp chúng ta. Vì vậy không cần phải viết các câu lệnh cập nhật tường minh.

**Trang Code:** Không phải viết vì khai báo ở trên sẽ tự động cập nhật !!!.

### 17.5.3 Xóa dữ liệu

**Ví dụ 2:** Xây dựng trang web cho phép xóa bản ghi trực tiếp trên GridView.

Việc xóa cũng hoàn toàn tương tự như Update. Tức là ta cũng cần phải thêm thuộc tính DeleteCommand vào trong SqlDataSource. cụ thể như sau:

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="AddEditUpdateDeleteColumn.aspx.cs" Inherits="Lesson_18_AllUserwithPaging" %>

<html xmlns="http://www.w3.org/1999/xhtml">
```

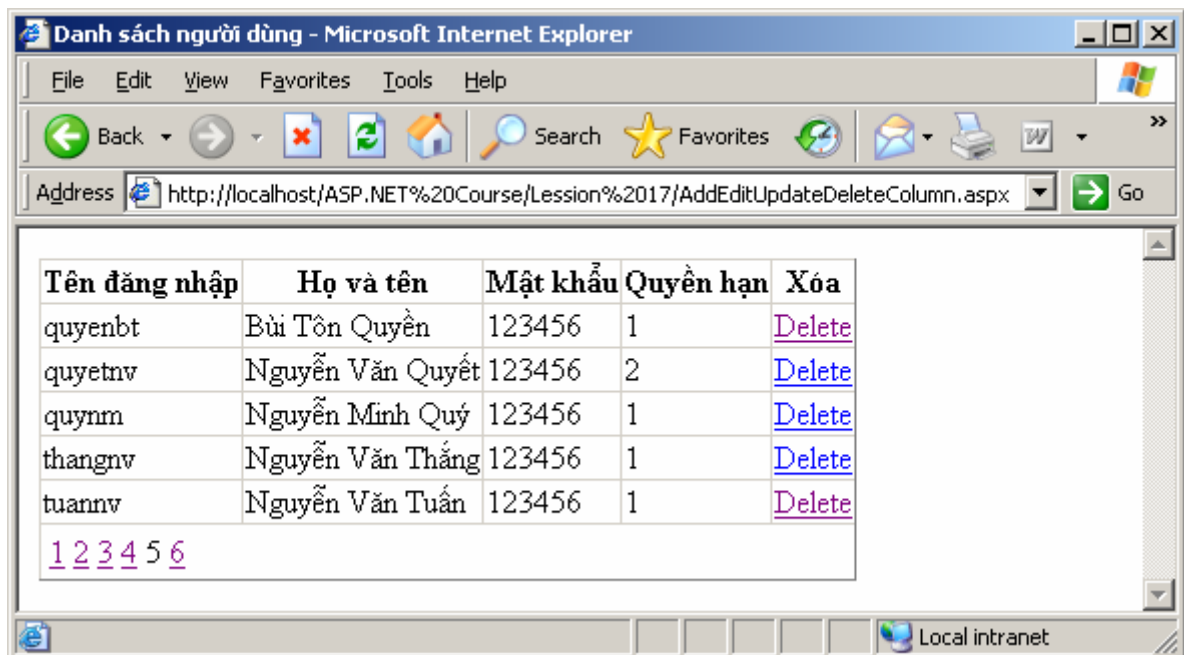
```
<head runat="server">
  <title>Cập nhật danh sách người dùng</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
      ConnectionString="<%%$ ConnectionStrings:QLCBConnectionString %>"
      SelectCommand="SELECT TenDangNhap, HoVaTen, MatKhai FROM tblUser"
      DeleteCommand="Delete tblUser where TenDangNhap = @TenDangNhap">
    </asp:SqlDataSource>

    <asp:GridView DataSourceID="SqlDataSource1" runat="server" ID="dgrDSND"
      AllowPaging="True" PageSize="5"
      AutoGenerateColumns="False"
      DataKeyNames="TenDangNhap">

      <Columns>
        <asp:BoundField DataField="TenDangNhap" HeaderText="Tên" ReadOnly="true" />
        <asp:BoundField DataField="HoVaTen" HeaderText="Họ và tên" ReadOnly="true"/>
        <asp:BoundField DataField="MatKhai" HeaderText="Mật khẩu" />
        <asp:BoundField DataField="QuyenHan" HeaderText="Quyền hạn" />

        <asp:CommandField HeaderText="Sửa chữa" ShowEditButton="True" />
        <asp:CommandField HeaderText="Xóa" ShowDeleteButton="True" />
      </Columns>
    </asp:GridView>

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



Xóa trực tiếp trên GridView.

## BÀI 18: THỰC HÀNH

**Mục tiêu:** Kết thúc bài thực hành này học viên có thể:

- Sử dụng được khả năng sắp xếp và phân trang của GridView với các cột sinh ra tự động hoặc thủ công.
- Cập nhật dữ liệu (Sửa và Xóa) trực tiếp trên GridView.
- Tạo ra các cột tùy biến: HyperLink và Image

### **Nội dung:**

**Bài 1:** Hiển thị thông tin vắn tắt (gồm Họ và tên, địa chỉ, chức vụ) về cán bộ trong cơ sở dữ liệu QLCB. Trong đó thực hiện phân trang với kích thước trang được lưu trong file web.config, ngoài ra còn cho phép người dùng sắp xếp bảng hiển thị bằng cách click chuột lên các cột tiêu đề.

### **Hướng dẫn:**

Để lưu thông số nào đó trong file web.config, cần thêm phần tử vào cặp thẻ <AppSettings>. Ví dụ: Ta cần lưu thiết lập về kích thước của trang, khi đó web.config sẽ có dạng:

```
.....
<appSettings>
  <add key="PageSize" value="5"/>
</appSettings>
.....
```

Khi đó, để đọc giá trị này, ta viết:

```
System.Configuration.ConfigurationManager.AppSettings ["PageSize"].ToString ();
```

### **Minh họa:**

#### **Trang giao diện**

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="PagingSorting.aspx.cs" Inherits="Lesson_18_PagingSorting" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Hiển thị và sắp xếp với GridView.</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
      ConnectionString="<%%$ ConnectionStrings:QLCBConnectionString %>"
      SelectCommand="SELECT TenDangNhap, HoVaTen, MatKhau FROM tblUser">
    </asp:SqlDataSource>

    <asp:GridView runat="server" ID="dgrDSND"
      AllowSorting="True"
      AllowPaging="True"
      DataSourceID="SqlDataSource1">
    </asp:GridView>
  </form>
</body>
</html>
```

#### Trang Code behind

```
using System;
using System.Collections;
using System.Configuration;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;

public partial class Lesson_18_PagingSorting : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        dgrDSND.PageSize=int.Parse(
            System.Configuration.ConfigurationManager.AppSettings ["PageSize"].ToString ());
    }
}
```

#### Chú ý:

Nếu muốn sắp xếp theo nhiều cột thì cần đưa danh sách tên cột (trường đó) vào thuộc tính SortExpression. Các phần tử cách nhau bởi dấu chấm phẩy.

**Bài 2:** Tạo một trang hiển thị thông tin chi tiết về người dùng trong bảng tblUser. Trong đó có thêm cột chi tiết (dạng Hyperlink) để khi người dùng click chọn thì chuyển tới một trang mới là UserDetail.aspx, tại đây sẽ hiển thị toàn bộ các thông tin khác của người dùng vừa được chọn.

Tên đăng nhập	Họ và tên	Mật khẩu	Chi tiết
admin	Admin	54355433	<a href="#">Xem chi tiết</a>
anhlt	Lưu Tuấn Anh	123456	<a href="#">Xem chi tiết</a>
chiennnd	Nguyễn Đình Chiến	123456	<a href="#">Xem chi tiết</a>
chuanpm	Phạm Minh Chuẩn	123456	<a href="#">Xem chi tiết</a>
dungvt	Vũ Thị Dung	123456	<a href="#">Xem chi tiết</a>
1 2 3 4 5 6			

Giao diện

#### Hướng dẫn:

Cần tạo thêm một cột – ví dụ ChiTiet - ở ngay trong câu lệnh SELECT, sau đó "Chèn" cột mới này vào trường HyperLinkField của GridView.

#### Minh họa:

##### Trang giao diện (Không có Code behind)

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UserList_ColumnHyperlink.aspx.cs" Inherits="Lesson_18_AllUserwithPaging"
%>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Danh sách người dùng</title>
```

```
</head>
<body>
  <form id="form1" runat="server">
    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
      ConnectionString="<%$ ConnectionStrings:QLCBConnectionString %>"
      SelectCommand="SELECT TenDangNhap, HoVaTen, MatKhau,
        'ChiTietNguoiDung.aspx?TenDangNhap=' + TenDangNhap as ChiTiet
        FROM tblUser">
    </asp:SqlDataSource>

    <asp:GridView DataSourceID="SqlDataSource1" runat="server"
      AllowPaging="True"
      PageSize="5"
      PagerStyle-HorizontalAlign="Center"
      PagerSettings-Mode="NumericFirstLast" AllowSorting="True"
      AutoGenerateColumns="False" CellPadding="4" >

      <Columns>
        <asp:BoundField DataField="TenDangNhap" HeaderText="Tên đăng nhập"
          SortExpression="tendangnhap" />
        <asp:BoundField DataField="HoVaTen" HeaderText="Họ và tên"
          SortExpression="HoVaTen" />
        <asp:BoundField DataField="MatKhau" HeaderText="Mật khẩu"
          SortExpression="MatKhau" />
        <asp:HyperLinkField DataNavigateUrlFields="ChiTiet" HeaderText="Chi tiết"
          Text="Xem chi tiết" />
      </Columns>
    </asp:GridView>
  </form>
</body>
</html>
```



### UserDetail.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UserDetail.aspx.cs" Inherits="Lesson_18_UserDetail" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Chi tiết người dùng</title>
    <style type="text/css">
        .CotTrai { text-align:right; font-style:italic; font-weight:bold}
    </style>
</head>
<body>
    <form id="form1" runat="server">
        <h2 style="border-bottom:solid 2px">Chi tiết người dùng đặt tại đây</h2>
        <table>
            <tr>
                <td class="CotTrai">Họ và tên</td>
                <td><asp:Label runat="server" ID="lblHVT"></asp:Label></td>
            </tr>
            <tr>
                <td class="CotTrai">Tên đăng nhập</td>
                <td><asp:Label runat="server" ID="lblTenDN"></asp:Label></td>
            </tr>
            <tr>
                <td class="CotTrai">Mật khẩu</td>
                <td><asp:Label runat="server" ID="lblMK"></asp:Label></td>
            </tr>
            <tr>
                <td class="CotTrai">Quyền hạn</td>
                <td><asp:Label runat="server" ID="lblQH"></asp:Label></td>
            </tr>
            <tr>
                <td class="CotTrai">Trạng thái</td>
                <td><asp:Label runat="server" ID="lblTT"></asp:Label></td>
            </tr>
            <tr>
                <td colspan="2"><asp:Image runat="server" ID="imgPhoto"/></td>
            </tr>
        </table>
        <hr />
        <a href="#" onclick="history.go(-1);"> &lt; Trở về</a>
    </form>
</body>
</html>
```

### Trang Code UserDetail.aspx.cs

```
using System;
using System.Collections;
using System.Configuration;
using System.Data.SqlClient;
using System.Web;
using System.Web.Security;
using System.Web.UI;
```



```
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;

public partial class Lesson_18_UserDetail : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        SqlConnection Cn = new SqlConnection ();
        Cn.ConnectionString=
            System.Configuration.ConfigurationManager.ConnectionStrings
                ["QLCBCConnectionString"].ToString ();

        Cn.Open ();

        SqlCommand Cmd = new SqlCommand ("Select * from tblUser where
            TenDangNhap= '" + Request.QueryString ["TenDangNhap"] + "'",Cn);
        SqlDataReader Dr = Cmd.ExecuteReader ();

        // Hiển thị thông tin chi tiết ra các nhãn và thẻ Image
        if (Dr.Read())
        {
            lblHVT.Text = Dr ["HoVaTen"].ToString();
            lblMK.Text = Dr ["MatKhau"].ToString ();
            lblTenDN.Text = Dr ["TenDangNhap"].ToString ();
            lblTT.Text = Dr ["TrangThai"].ToString ();
            lblQH.Text = Dr ["QuyenHan"].ToString ();
            imgPhoto.ImageUrl = Dr ["HinhAnh"].ToString ();
        }
        Cmd.Dispose ();
        Cn.Close ();
    }
}
```

**Bài 3:** Tạo một trang **PhoToNguoiDung**, ở đó hiển thị 3 cột thông tin là HoVaTen, TrangThai, QuyenHan và Ảnh tương ứng. Giả sử thư mục ảnh có tên là Images và nằm cùng với trang web.

**Hướng dẫn:**

Cần tạo thêm một cột mới – ví dụ đặt tên là **DuongDanAnh** – sau đó "Chèn" trường này vào ImageField của GridView.

**Minh họa:**

Trang code giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="UserList_ColumnImage.aspx.cs" Inherits="Lesson_18_AllUserwithPaging"
%>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <style type="text/css">
        img {width:50px; height:50px}
    </style>
    <title>Danh sách người dùng</title>
```

```
</head>
<body>
  <form id="form1" runat="server">
    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
      ConnectionString="<%$ ConnectionStrings:QLCBConnectionString %>"
      SelectCommand="SELECT TenDangNhap, HoVaTen, HinhAnh FROM tblUser">
    </asp:SqlDataSource>

    <asp:GridView DataSourceID="SqlDataSource1"
      runat="server" ID="dgrDSND"
      DataKeyNames="TenDangNhap"
      AllowPaging="True"
      PageSize="5"
      PagerStyle-HorizontalAlign="Center"
      AllowSorting="True"
      AutoGenerateColumns="False">
      <Columns>
        <asp:BoundField DataField="TenDangNhap" HeaderText="Tên đăng nhập"
          SortExpression="tendangnhap" />
        <asp:BoundField DataField="HoVaTen" HeaderText="Họ và tên"
          SortExpression="HoVaTen" />
        <asp:BoundField DataField="MatKhai" HeaderText="Mật khẩu"
          SortExpression="MatKhai" />
        <asp:ImageField DataImageUrlField="HinhAnh" HeaderText="Hình ảnh" >
        </asp:ImageField>
      </Columns>
    </asp:GridView>
  </form>
</body>
</html>
```

**Kết quả ➔**

Tên đăng nhập	Họ và tên	Mật khẩu	Hình ảnh
admin	Admin	54355433	
anhlt	Lưu Tuấn Anh	123456	
chiennnd	Nguyễn Đình Chiến	123456	
chuanpm	Phạm Minh Chuẩn	123456	
dungvt	Vũ Thị Dung	123456	
1 2 3 4 5 6			

**Bài 4:** Tạo một trang CapNhatCanBo, trong đó cho phép người dùng sửa đổi Họ tên và Địa chỉ trực tiếp. (Trên GridView.) và xóa một cán bộ bất kỳ.

**Hướng dẫn:** Làm tương tự như bài lý thuyết (Thêm các cột Edit, Delete vào GridView)

**Minh Họa:**

**Trang giao diện (Không có Code ở trang CS)**

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="EditUpdateDeleteCB.aspx.cs" Inherits="Lesson_18_AllUserwithPaging" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title>Danh sách cán bộ</title>
</head>
<body>
<form id="form1" runat="server">
<asp:SqlDataSource ID="SqlDataSource1" runat="server"
ConnectionString="<%%$ ConnectionStrings:QLCBConnectionString %>"
SelectCommand="SELECT * FROM tblCanBo"
UpdateCommand="Update tblCanBo set HoVaTen=@HoVaTen, DiaChi=@DiaChi
where MaCanBo=@MaCanBo"
DeleteCommand="Delete tblCanBo where MaCanBo = @MaCanBo">
</asp:SqlDataSource>

<asp:GridView DataSourceID="SqlDataSource1" runat="server" ID="dgrDSCB"
AllowPaging="True" PageSize="5"
AutoGenerateColumns="False"
DataKeyNames="MaCanBo">

<Columns>
<asp:BoundField DataField="HoVaTen" HeaderText="Họ và tên" />
<asp:BoundField DataField="DiaChi" HeaderText="Địa chỉ" />
<asp:BoundField DataField="NgaySinh" HeaderText="NS" ReadOnly="true"/>
<asp:BoundField DataField="SoDienThoai" HeaderText="ĐT" ReadOnly="true"/>
<asp:CommandField HeaderText="Sửa chữa" ShowEditButton="True" />
<asp:CommandField HeaderText="Xóa" ShowDeleteButton="True" />
</Columns>
</asp:GridView>
</form>
</body>
</html>
```

Họ và tên	Ngày sinh	Ngày sinh	Số điện thoại	Sửa chữa	Xóa
Nguyễn Minh Quý	Hưng Yên		0321-713319	<a href="#">Edit</a>	<a href="#">Delete</a>
Lê Quang Lợi	Hưng Yên City			<a href="#">Edit</a>	<a href="#">Delete</a>
<input type="text" value="Phạm Ngọc Hưng"/>	<input type="text" value="Quảng Ninh"/>			<a href="#">Update</a> <a href="#">Cancel</a>	
Nguyễn Đình Hân	Hưng Yên			<a href="#">Edit</a>	<a href="#">Delete</a>
Đặng Bá Hùng	Hải Dương			<a href="#">Edit</a>	<a href="#">Delete</a>
1 2					

## BÀI 19: Sử dụng Templates

### 19.1 Giới thiệu tổng quan

Trong các bài trước, chúng ta đã tìm hiểu khả năng trình diễn dữ liệu rất mạnh của GridView, nó có thể phân trang tự động, sắp xếp tự động và cập nhật dữ liệu tự động... Tuy nhiên, trong một số trường hợp cụ thể thì chúng ta vẫn rất cần phải có sự trình bày linh hoạt hơn, ví dụ: Chúng ta muốn trình bày dữ liệu theo chiều dọc chẳng hạn. v.v...

Rất may là GridView (và các điều khiển khác) đều có cơ chế để hỗ trợ cho chúng ta khả năng tùy biến rất cao – đó là TEMPLATE !.

Template – dịch ra Tiếng Việt – có thể hiểu là các Mẫu định sẵn. Cái mẫu này do lập trình viên tự thiết kế. Trong mẫu này, chúng ta có thể thêm bất cứ thẻ HTML hay thẻ server hợp lệ. Mỗi lần xử lý một mục dữ liệu (một bản ghi) thì hệ thống sẽ tự động đưa mục dữ liệu này vào bên trong mẫu.

Bài học này sẽ giới thiệu thêm về Template của các điều khiển nâng cao là GridView, DataList và Repeater.

### 19.2 Các điều khiển hỗ trợ Templates

#### 19.2.1 Một số điều khiển hỗ trợ Template thường dùng

Trong asp.NET có rất nhiều điều khiển hỗ trợ khả năng tùy biến bằng Templates. Một số điều khiển phổ biến hỗ trợ Template bao gồm:

- ❖ GridView
- ❖ DataGrid
- ❖ DataList
- ❖ Repeater

#### 19.2.2 Các loại Template

Các điều khiển đều có một số phần hiển thị tương tự nhau: Như phần tiêu đề (Header), phần nội dung dữ liệu (DataItem) và phần chân trang (Footer). Ý nghĩa cụ thể của từng phần này như sau:

- ❖ **HeaderTemplate**: Là phần cho phép định dạng tiêu đề trước nội dung hiển thị. Thường là tên các trường trong CSDL,...
- ❖ **ItemTemplate**: Là phần cho phép định dạng cách thức hiển thị mục dữ liệu (bản ghi). Phần này được tạo ra mỗi khi một dòng dữ liệu được gắn kết vào điều khiển.
- ❖ **AlternateTemplate**: Phần này cho phép định dạng cách thức hiển thị của phần tử thuộc hàng chẵn.
- ❖ **FooterTemplate**: Cho phép định dạng chân trang sau nội dung hiển thị.

Ngoài ra, mỗi điều khiển còn có thêm các thuộc tính khác nữa.

Khi sử dụng Template, thông thường chúng ta có mong muốn là hiển thị một số trường dữ liệu nào đấy. Vì vậy, asp.net cung cấp một hàm gọi là **Eval**("Tên\_Mục/Trường dữ liệu") cho phép chúng ta lấy dữ liệu của một trường thuộc bản ghi hiện hành. Giá trị của hàm Eval() này được đưa vào trong thẻ hiển thị thông qua cơ chế DataBind, kiểu như sau:

Họ và tên: <b> <%# Eval ("HoVaTen") %> </b>
---------------------------------------------

## 19.3 Repeater control, DataList control, GridView control

### 19.3.1 Tạo template với GridView.

**Ví dụ 1:** Tạo một trang web hiển thị danh sách **Họ và tên** người dùng trong bảng tblUser. Ở bài trước ta đã hiển thị được một cách dễ dàng, tuy nhiên sau đây chúng ta sẽ sử dụng Template của GridView.

#### Trang giao diện

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="DisplaytblUser.aspx.cs" Inherits="Lesson_19_DisplaytblUser" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head id="Head1" runat="server">
  <title>Danh sách người dùng - sử dụng Templates</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
      ConnectionString="<%$ ConnectionStrings:QLCBConnectionString %>"
      SelectCommand="SELECT * FROM tblUser">
    </asp:SqlDataSource>

    <asp:GridView ID="GridView1" runat="server"
      DataSourceID="SqlDataSource1"
      AutoGenerateColumns="false" AllowPaging="true">

      <Columns>
        <asp:TemplateField HeaderText="Họ và tên">
          <ItemTemplate>
            <%# Eval("HoVaTen") %>
          </ItemTemplate>
        </asp:TemplateField>
      </Columns>

    </asp:GridView>
  </form>
</body>
</html>
```

Đặt nội dung bất kỳ (Có thể là text, html, server controls ...)

**Ví dụ 2:** Như ví dụ 1 nhưng đặt họ tên vào các TextBox

#### Trang giao diện

```
<asp:GridView ID="GridView1" runat="server"
  DataSourceID="SqlDataSource1"
  AutoGenerateColumns="false" AllowPaging="true">
  <Columns>
    <asp:TemplateField HeaderText="Họ và tên">
      <ItemTemplate>
        <asp:TextBox runat="server"
          Text="<%# Eval("HoVaTen") %>">
```

```
        </asp:TextBox>
    </ItemTemplate>
</asp:TemplateField>
</Columns>
</asp:GridView>
```

**Ví dụ 3:** Hiển thị họ tên và TrangThai. Trong đó TrangThai được đặt trong ngoặc ngay sau họ và tên.

**Trang giao diện**

```
<asp:GridView ID="GridView1" runat="server"
    DataSourceID="SqlDataSource1"
    AutoGenerateColumns="false" AllowPaging="true">
    <Columns>
        <asp:TemplateField HeaderText="Họ và tên">
            <ItemTemplate>
                <%# Eval("HoVaTen") %>
                ( <%# Eval("TrangThai") %> )
            </ItemTemplate>
        </asp:TemplateField>
    </Columns>
</asp:GridView>
```

**Ví dụ 4:** Hiển thị họ tên và ảnh

**Trang giao diện**

```
<%@ Page Language="C#" AutoEventWireup="true"
    CodeFile="DisplaytblUserOnTextBox.aspx.cs" Inherits="Lesson_19_DisplaytblUser" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head id="Head1" runat="server">
    <title>Danh sách người dùng - sử dụng Templates</title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:SqlDataSource ID="SqlDataSource1" runat="server"
            ConnectionString="<%%$ ConnectionStrings:QLCBConnectionString %>"
            SelectCommand="SELECT * FROM tblUser">
        </asp:SqlDataSource>

        <asp:GridView ID="GridView1" runat="server"
            DataSourceID="SqlDataSource1"
            AutoGenerateColumns="false" AllowPaging="true">
            <Columns>
                <asp:TemplateField HeaderText="Họ và tên & ảnh ">
                    <ItemTemplate>
                        <%# Eval("HoVaTen") %> <br />
```

```
<asp:Image runat="server" ImageUrl='<%# Eval("HinhAnh") %>' />
</ItemTemplate>
</asp:TemplateField>
</Columns>

</asp:GridView>
</form>
</body>
</html>
```

### **Thêm nhiều cột:**

Để thêm nhiều cột, ta cần thêm cặp thẻ **<asp:TemplateField > </asp:TemplateField>**. Trong đó, mỗi cặp thẻ sẽ tương ứng với một CỘT của GridView .

**Ví dụ 5:** Hiển thị các trường HoVaTen, MatKhanh, *Hình ảnh* trên GridView, mỗi trường hiển thị trên một cột và **Họ tên** chữ đậm.

#### **Trang giao diện**

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="DisplaytblUserMultiCols.aspx.cs" Inherits="Lesson_19_DisplaytblUser" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head id="Head1" runat="server">
<title>Danh sách người dùng - sử dụng Templates</title>
</head>
<body>
<form id="form1" runat="server">
<asp:SqlDataSource ID="SqlDataSource1" runat="server"
ConnectionString="<%%$ ConnectionStrings:QLCBConnectionString %>"
SelectCommand="SELECT * FROM tblUser">
</asp:SqlDataSource>

<asp:GridView ID="GridView1" runat="server"
DataSourceID="SqlDataSource1"
AutoGenerateColumns="false" AllowPaging="true">
<Columns>

<asp:TemplateField HeaderText="Họ và tên">
<ItemTemplate>
<b><%%# Eval("HoVaTen") %></b> <br />
</ItemTemplate>
</asp:TemplateField>

<asp:TemplateField HeaderText="Trạng thái">
<ItemTemplate>
<i><%%# Eval("TrangThai") %></i> <br />
</ItemTemplate>
</asp:TemplateField>
```

```
<asp:TemplateField HeaderText="Hình ảnh">
  <ItemTemplate>
    <asp:image runat="server"
      ImageUrl='<%# Eval("HinhAnh") %>'
      Width="100px" Height="100px"
    />
  </ItemTemplate>
</asp:TemplateField>
</Columns>
</asp:GridView>
</form>
</body>
</html>
```

**Ví dụ 6:** Hiển thị dữ liệu trong một cột

```
..... Như ví dụ 5 .....
<asp:GridView ID="GridView1" runat="server" Width="100%"
  DataSourceID="SqlDataSource1" ShowHeader="false"
  AutoGenerateColumns="false" AllowPaging="true">
  <Columns>
    <asp:TemplateField HeaderText="THÔNG TIN NGƯỜI DÙNG">
      <ItemTemplate>
        <b> Họ và tên: <%# Eval("HoVaTen") %></b> <hr />
        Tên đăng nhập: <%# Eval("TenDangNhap") %> <br />
        Quyền hạn : <%# Eval("QuyenHan") %> <br />
        Trạng thái: <%# Eval("TrangThai") %> <br />
        Hình ảnh : <br /><asp:Image runat="server"
          ImageUrl='<%# Eval("HinhAnh") %>' />
      </ItemTemplate>
    </asp:TemplateField>
  </Columns>
</asp:GridView>
..... Như ví dụ 5 .....
```

**Cập nhật dữ liệu**

Để cho phép sửa dữ liệu của một trường, ta cần đặt thuộc tính text của trường đó vào trong phương thức Bind, dạng như sau:

```
<asp:TextBox
  Text='<%# Bind("Tên_Trường") %>'
  runat="server" id="textBox"/>
</asp:TextBox>
```

**Ví dụ 7:** Cập nhật hồ sơ cán bộ (Chỉ cập nhật trường "Bản thân")

```
..... Như ví dụ 5 .....
<asp:GridView ID="GridView1" runat="server" Width="100%"
  DataSourceID="SqlDataSource1" DataKeyNames="MaCanBo"
  AutoGenerateColumns="false" AllowPaging="true">
```



```

<Columns>
  <asp:TemplateField HeaderText="THÔNG TIN CÁN BỘ">
    <ItemTemplate>
      <b> Họ và tên: <%# Eval("HoVaTen") %> </b> <hr />
      Mã Cán bộ: <%# Eval("MaCanBo") %> <br />
      Ngày sinh : <%# Eval("NgaySinh") %> <br />
      Địa chỉ: <%# Eval("DiaChi") %> <br />
      Bản thân : <%# Eval("BanThan") %>
    </ItemTemplate>

    <EditItemTemplate>
      <b> Họ và tên: <%# Eval("HoVaTen") %> </b> <hr />
      Mã Cán bộ: <%# Eval("MaCanBo") %> <br />
      Ngày sinh : <%# Eval("NgaySinh") %> <br />
      Địa chỉ: <%# Eval("DiaChi") %> <br />
      Bản thân :
      <asp:TextBox TextMode="MultiLine" runat="server"
        id="txtBanThan" Text="<%# Bind("BanThan") %>" />
    </EditItemTemplate>
  </asp:TemplateField>
  <asp:CommandField ButtonType="Button" EditText="Sửa" ShowEditButton="true" />
</Columns>
</asp:GridView>
..... Như ví dụ 5 .....

```

### 19.3.2 Tạo template với DataList

DataList cho phép ta hiển thị dưới dạng danh sách. Danh sách này có thể chia làm nhiều cột như một số ứng dụng thường thấy trên Internet.

**Ví dụ 8:** Hiển thị ảnh như hình trên;

```

.....
<body>
  <form id="form1" runat="server">
    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
      ConnectionString="<%$ ConnectionStrings:QLCBConnectionString %>"
      SelectCommand="SELECT * FROM tblUser">
    </asp:SqlDataSource>

    <asp:DataList runat="server" ID="DataList1"
      DataSourceID="SqlDataSource1"
      RepeatDirection="Horizontal"
      RepeatColumns="3"
      RepeatLayout="Table">
      <ItemTemplate>
        Họ và tên: <b> <%# Eval("HoVaTen") %> </b> <br />
        Quyền hạn : <%# Eval("QuyenHan") %> <br />
        <asp:Image ID="Image1" Width="100px" Height="100px"
          runat="server" ImageUrl="<%# Eval("HinhAnh") %>" />
        <hr />
      </ItemTemplate>
    </asp:DataList>
  </form>

```

```
</form>
</body>
```



Kết quả sau khi chạy trang ở trên

### 19.3.3 Tạo Template với Repeater (light-weight)

Repeater cũng là một điều khiển có khả năng hiển thị dữ liệu dưới dạng danh sách. Khi được gắn với nguồn dữ liệu, nó sẽ lần lượt thực thi nội dung nằm trong phần Template mỗi khi một bản ghi được đọc từ nguồn.

Tuy điều khiển này không có khả năng phân trang, sắp xếp như GridView như nó là một điều khiển chiếm ít tài nguyên của hệ thống (vì vậy được gọi là điều khiển Light-weight), do đó chúng ta có thể dùng trong những trường hợp mà ở đó tài nguyên đóng vai trò quan trọng.

Repeater cho phép chúng ta tùy biến các mục tương tự như GridView và DataList, đó là sử dụng Template.

**Ví dụ 9:** Hiển thị họ tên và ảnh minh họa người dùng trong bảng tblUser sử dụng điều khiển Repeater.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Repeater.aspx.cs"
Inherits="Lesson_19_Repeater" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Hiển thị danh sách người dùng</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
      ConnectionString="<%$ ConnectionStrings:QLCBConnectionString %>"
      SelectCommand="SELECT * FROM tblUser">
    </asp:SqlDataSource>

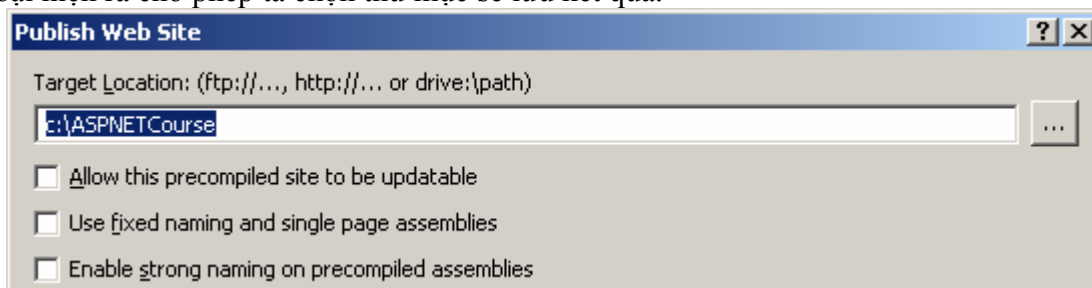
    <asp:Repeater runat="server" ID="rptDSND"
      DataSourceID="SqlDataSource1">
      <ItemTemplate>
        Họ và tên: <%# Eval("HoVaTen") %> <br />
        <asp:Image runat="server"
          ImageUrl='<%# Eval("HinhAnh") %>'
          Width="100px" Height="100px" />
        <hr />
      </ItemTemplate>
    </asp:Repeater>

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

## 20. Đóng gói website

Sau khi hoàn tất dự án thì một khâu quan trọng nữa cần phải thực hiện để đảm bảo vấn đề bản quyền đó là biên dịch và xuất bản ứng dụng web. Việc xuất bản (Publish) ứng dụng sẽ giúp biên dịch các file code behind (file CS) thành các assemblies (file DLL). Khi đó, ứng dụng chạy sẽ nhanh hơn và bảo mật hơn.

Để xuất bản web, Right click vào Tên của Solution và chọn mục **Publish**. Sau đó một hộp thoại hiện ra cho phép ta chọn thư mục sẽ lưu kết quả.



Ứng dụng được biên dịch này có thể copy vào Server để thực thi.

## BÀI 20: THỰC HÀNH

**Mục tiêu:** Kết thúc bài học này, học viên có thể

- ❖ Sử dụng được tính năng Template để trình diễn và cập nhật dữ liệu theo yêu cầu
- ❖ Sử dụng được DataList để trình diễn dữ liệu dưới dạng cột.
- ❖ Sử dụng ListView kết hợp với DataPager để hiển thị và duyệt các bản ghi.

**Nội dung:**

**Bài 1:** Hiện thị danh sách cán bộ (bao gồm họ tên, địa chỉ, điện thoại) trong bảng tblCanBo. Trong đó Họ tên chữ **đậm**.

**Hướng dẫn:**

- Đặt thuộc tính AutoGenerateColumns = "false"
- Tự tạo các cột cho GridView bằng cặp thẻ `<asp:TemplateField> ...</>`
- Đặt nội dung cần hiển thị (ở đây là Họ tên, địa chỉ, điện thoại) vào các cột bằng cách đặt bên trong cặp thẻ `<asp:DataItemTemplate> </>`

Họ và tên	Địa chỉ	Điện thoại
Nguyễn Minh Quý	Hưng Yên	0321-713319
Lê Quang Lợi	Hưng Yên City	0913-713777
Phạm Ngọc Hưng	Quảng Ninh	0912-713777
Nguyễn Đình Hân	Hưng Yên-hải dương	0912-713777
Đặng Bá Hùng	Hải Dương	0912-713777
Nguyễn Thị Hiền	Hưng yên	012344567

Danh sách cán bộ

**Minh họa:**

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="ThongTinCanBo.aspx.cs" Inherits="Lesson_20_ThongTinCanBo" %>
<head runat="server"> <title>Thông tin về cán bộ</title> </head>
<body>
    <form id="form1" runat="server">
        <asp:SqlDataSource ID="SqlDataSource1" runat="server"
            ConnectionString="<%%$ ConnectionStrings:QLCBConnectionString %>"
            SelectCommand="SELECT * FROM tblCanBo">
        </asp:SqlDataSource>

        <asp:GridView runat="server" ID="DSCB"
            DataSourceID="SqlDataSource1" AutoGenerateColumns="false">
            <Columns>
                <asp:TemplateField HeaderText="Họ và tên">
                    <ItemTemplate>
```

```

        <b><%# Eval("HoVaTen") %></b>
    </ItemTemplate>
</asp:TemplateField>

<asp:TemplateField HeaderText="Địa chỉ">
    <ItemTemplate>
        <%# Eval("DiaChi") %>
    </ItemTemplate>
</asp:TemplateField>

<asp:TemplateField HeaderText="Điện thoại">
    <ItemTemplate>
        <%# Eval("SoDienThoai") %>
    </ItemTemplate>
</asp:TemplateField>
</Columns>
</asp:GridView>
</form>
</body>
</html>

```

**Bài 2:** Bổ sung thêm trường Photo vào bảng tblCanBo, trường Photo này sẽ lưu đường dẫn tới file ảnh của mỗi cán bộ. Sau đó xây dựng trang web hiển thị thông tin cán bộ (bao gồm các trường Họ tên, địa chỉ và ảnh tương ứng).

**Hướng dẫn:**

- Vì trường Photo đã chứa đường dẫn đến file ảnh rồi, vì vậy để hiển thị hình ảnh thay vì văn bản text thuần túy, ta sẽ tạo thêm phân tử <asp:Image>, trong đó thuộc tính ImageUrl sẽ được gán giá trị của trường Photo tương ứng. cụ thể là :  
 <asp:Image ID="Image1" runat="server" ImageUrl='<%# Eval("Photo") %>' />
- Để đặt kích thước ảnh như nhau, có thể thêm thuộc tính Width và Height
- Trước giá trị <%# Eval("Photo") %> cần có thêm cặp dấu nháy đơn để đảm bảo tính đúng đắn khi đường dẫn ảnh chứa dấu cách.



Yêu cầu về giao diện

**Minh họa:**

```
<asp:GridView runat="server" ID="DSCB" Width="100%" GridLines="Both"
DataSourceID="SqlDataSource1" AutoGenerateColumns="false">
  <Columns>
    <asp:TemplateField HeaderText="Họ và tên">
      <ItemTemplate>
        <%# Eval("HoVaTen") %>
      </ItemTemplate>
    </asp:TemplateField>

    <asp:TemplateField HeaderText="Địa chỉ">
      <ItemTemplate>
        <%# Eval("DiaChi") %>
      </ItemTemplate>
    </asp:TemplateField>

    <asp:TemplateField HeaderText="Ảnh">
      <ItemTemplate>
        <asp:Image runat="server" Width="100px" Height="100px"
          ImageUrl='<%# Eval("Photo") %>' />
      </ItemTemplate>
    </asp:TemplateField>
  </Columns>
</asp:GridView>
```

**Bài 3:** Hiển thị danh sách cán bộ dạng chi tiết. Thông tin được dàn trang theo chiều dọc (Flow).

**Hướng dẫn:** Tạo một cột duy nhất, nhưng mỗi dòng của cột đó sẽ chứa tất cả các trường thông tin cần hiển thị. Với mỗi hàng được tạo ra, ta sẽ đặt vào một table có kích thước cố định, bảng này có 1 hàng và 2 cột. Cột thứ nhất sẽ chứa thông tin ở dạng text như họ tên, địa chỉ, điện thoại và mô tả bản thân. Cột thứ hai sẽ hiển thị hình ảnh tương ứng.



Kết quả

**Minh họa:**

```
<asp:GridView runat="server" ID="DSCB" Width="100%" GridLines="None"
DataSourceID="SqlDataSource1" AutoGenerateColumns="false">
  <Columns>
    <asp:TemplateField HeaderText="Trích ngang danh sách cán bộ">
      <ItemTemplate>
        <table border="0px" width="300px">
          <tr>
            <td style="width:200px;border-top:solid 1px">
              <b>
                <%# Eval("HoVaTen") %>
              </b><br />
                <%# Eval("DiaChi") %> <br />
                <%# Eval("SoDienThoai") %> <br />
                <%# Eval("BanThan") %>
              </td>
            <td style="width:100px">
              <asp:Image runat="server"
                Width="100px"
                Height="100px"
                ImageUrl='<%# Eval("PhoTo") %>'
              />
            </td>
          </tr>
        </table>
      </ItemTemplate>
    </asp:TemplateField>
  </Columns>
</asp:GridView>
```

**Bài 3:** Hiển thị danh sách người dùng dưới dạng Flow (tuyến tính) như bài 3, nhưng có thêm chức năng cập nhật và Delete. Thông tin hiển thị gồm Họ tên, Địa chỉ, Điện thoại. Trong đó, trường địa chỉ sẽ hiển thị ở dạng Text Multiline khi sửa.

**Hướng dẫn:**

Tạo các template như bài 3, nhưng thêm 2 command là Edit và Delete. Trong đó các trường muốn sửa chữa sẽ đặt vào các Textbox và dùng hàm <%# Bind("Tên\_Trường")%>.

**Minh họa:**



Họ và tên	Sửa chữa	Xóa
Nguyễn Minh Quý Địa chỉ: Hưng Yên, Việt nam Điện thoại: 0321-713319	<a href="#">Sửa</a>	<a href="#">Xóa</a>
<b>Họ tên:</b> <input type="text" value="Lê Quang Lợi"/>		
<b>Địa chỉ:</b> <input type="text" value="Hưng Yên City"/>		
<b>Điện thoại:</b> <input type="text" value="0913-713777"/>		
<a href="#">Lưu</a> <a href="#">Hủy bỏ</a>		
Trần Thị Phương Địa chỉ: Yên Bái Điện thoại: 0912-713777	<a href="#">Sửa</a>	<a href="#">Xóa</a>
Phí Đức Long Địa chỉ: Hưng Yên Điện thoại: 0912-713777	<a href="#">Sửa</a>	<a href="#">Xóa</a>

Kết quả.

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="DSCB_ImageField.aspx.cs" Inherits="Lesson_20_DSCB_ImageField" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Cập nhật danh sách cán bộ</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
      ConnectionString="<%$ ConnectionStrings:QLCBConnectionString %>"
      SelectCommand="SELECT * FROM tblCanBo"

      UpdateCommand="Update tblCanBo SET
HoVaTen=@HoVaTen, DiaChi=@DiaChi, SoDienThoai=@SoDienThoai
Where MaCanBo=@MaCanBo"

      DeleteCommand="Delete tblCanBo where MaCanBo=@MaCanBo">
    </asp:SqlDataSource>
```



```

<asp:GridView runat="server" ID="DSCB" Width="100%" GridLines="both"
    DataKeyNames="MaCanBo"
    DataSourceID="SqlDataSource1" AutoGenerateColumns="false">
    <Columns>
        <asp:TemplateField HeaderText="Họ và tên">
            <ItemTemplate>
                <b><%# Eval("HoVaTen") %></b> <br />
                Địa chỉ: <%# Eval("DiaChi") %> <br />
                Điện thoại: <%# Eval("SoDienThoai") %>
            </ItemTemplate>

            <EditItemTemplate>
                <b>Họ tên: </b><br />
                <asp:TextBox runat="server" ID="txtHVT" Width="98%"
                    Text= '<%# Bind("HoVaTen") %>' >
                </asp:TextBox>
                <br />

                <b>Địa chỉ: </b><br />
                <asp:TextBox runat="server" ID="txtDC" Width="98%"
                    TextMode="MultiLine" Rows="5"
                    Text= '<%# Bind("DiaChi") %>'>
                </asp:TextBox>
                <br />

                <b>Điện thoại: </b> <br />
                <asp:TextBox runat="server" ID="txtDT" Width="98%"
                    Text= '<%# Bind("SoDienThoai") %>'>
                </asp:TextBox>
            </EditItemTemplate>
        </asp:TemplateField>

        <asp:CommandField
            HeaderText="Sửa chữa"
            EditText=" Sửa"
            UpdateText="Lưu"
            ButtonType="Link"
            CancelText=" Hủy bỏ"
            ShowEditButton="true" />

        <asp:CommandField
            HeaderText="Xóa"
            DeleteText=" Xóa"
            ButtonType="Link"
            ShowDeleteButton="true" />
    </Columns>
</asp:GridView>
</form>
</body>
</html>

```

Phần template cho chế độ thường (chưa sửa)

Phần template cho chế độ soạn thảo (edit mode)

**Bài 4:** Hiện thị thông tin trích ngang về người dùng trong bảng tblUser, trong đó dưới mỗi người dùng thêm một Hyperlink là "**Xem chi tiết**" để khi người dùng click vào hyperlink này thì mở trang **UserDetail.aspx** và hiện thị chi tiết thông tin về người dùng đó. Yêu cầu thêm: Danh sách này hiển thị làm 3 cột.

**Hướng dẫn:** Để hiển thị thông tin dưới dạng cột, ta sử dụng điều khiển DataList. Trong mỗi Hyperlink ta sẽ tạo liên kết đến trang UserDetail.aspx và truyền cho trang này ID (trong trường hợp này là TenDangNhap) của người dùng tương ứng. Dựa vào ID này, trang UserDetail.aspx sẽ đọc (dùng Request.QueryString["TenNguoiDung"]) sau đó select thông tin ứng với ID đó và hiển thị.

**Minh họa:**

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="DataListControl.aspx.cs"
Inherits="Lesson_19_RepeaterControl" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Danh sách cán bộ</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
      ConnectionString="<%$ ConnectionStrings:QLCBConnectionString %>"
      SelectCommand="SELECT * FROM tblUser">
    </asp:SqlDataSource>

    <asp:DataList runat="server" ID="DataList1"
      DataSourceID="SqlDataSource1"
      RepeatDirection="Horizontal"
      RepeatColumns="3"
      RepeatLayout="Table">

      <ItemTemplate >
        <b> <%# Eval("HoVaTen") %> </b> <br />

        Quyền hạn : <%# Eval("QuyenHan") %> <br />

        <asp:Image Width="100px" Height="100px"
          runat="server" ImageUrl= '<%# Eval("HinhAnh") %>' /><br />

        <a href='UserDetail.aspx?TenDangNhap=<%#Eval("TenDangNhap") %>'>
          Xem chi tiết
        </a>
        <hr />
      </ItemTemplate>
    </asp:DataList>
  </form>
</body>
</html>
```



Kết quả.

**Bài 4:** Hiển thị danh sách người dùng dưới dạng cột và thực hiện phân trang.

**Hướng dẫn:**

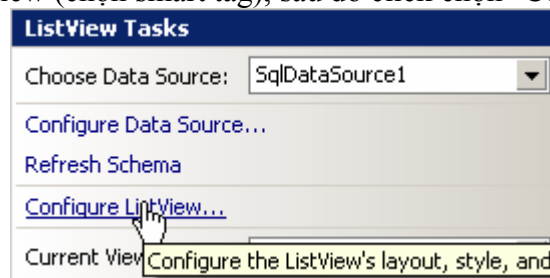
Để thực hiện tạo các cột, có thể sử dụng điều khiển DataList, ngoài ra ta còn một điều khiển khác cũng rất mạnh cho phép hiển thị dưới dạng các cột nhưng có thêm khả năng phân trang, đó là: ListView controls.

**Các bước thực hiện:**

B1. Tạo nguồn dữ liệu SqlDataSource như những bài trước.

B2. Tạo một ListView và gắn với nguồn dữ liệu SqlDataSource (như bài trước)

B3. Cấu hình cho ListView (chọn smart tag), sau đó click chọn "Config ListView..."

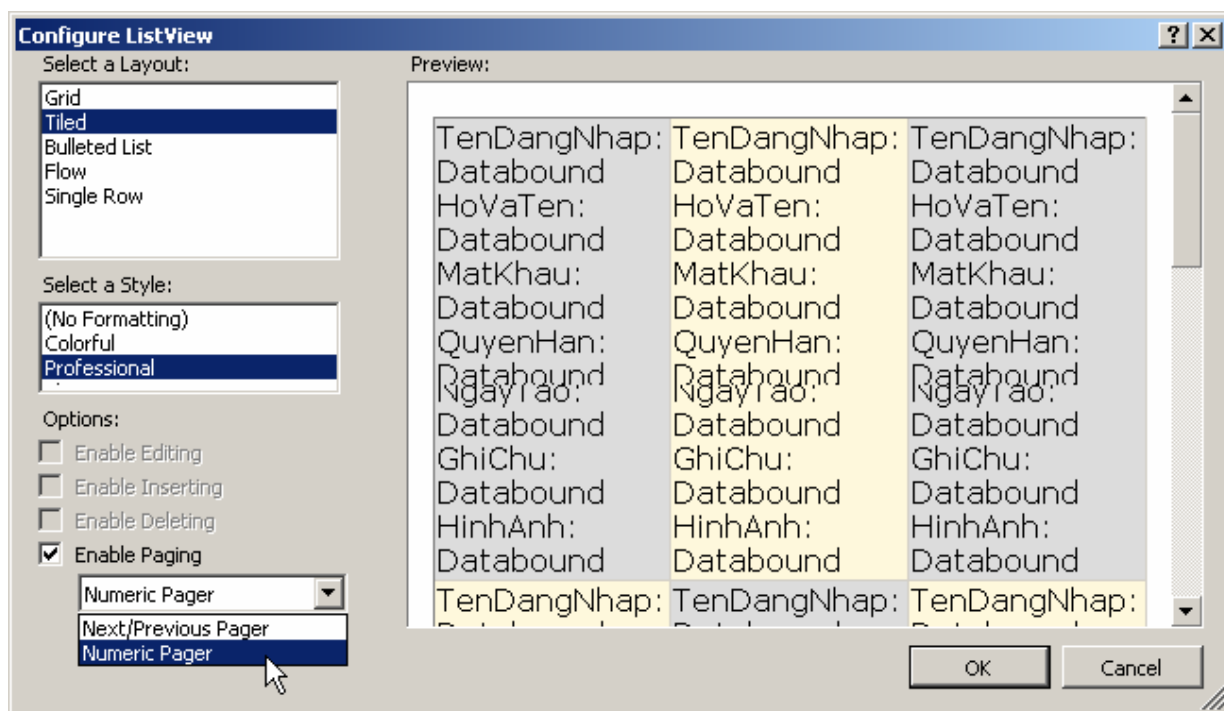


B4: Chọn các thông số như hình bên dưới.

B5: Mở trang web.

**Lưu ý:** Chúng ta hoàn toàn có thể thay đổi format của ListView bằng các vào Source code editor để sửa.

Một cách phân trang khác là tạo một DataPager và gắn vào ListView, Khi đó ta chỉ cần đặt thuộc tính `PagedControlID="ListView1"`. Cách này có ưu điểm là phân trang có thể đặt ở bất kỳ vị trí nào trên màn hình.



Kết quả hiển thị