

MỤC LỤC

Chương 1: TỔNG QUAN NGÔN NGỮ C#	1
1.1 Giới thiệu C#	1
1.2 Các khái niệm cơ bản	1
1.2.1 Cấu trúc chương trình.....	1
1.2.2 Định danh (identifier)	2
1.2.3 Không gian tên (namespace).....	3
1.2.4 Toán tử `'	4
1.2.5 Từ khóa using	4
1.2.6 Từ khóa static	4
1.2.7 Biến (variable)	6
1.2.8 Từ khóa var	8
1.2.9 Lệnh và khối lệnh	8
1.2.10 Biểu thức.....	9
1.2.11 Chú thích (comment).....	10
1.3 Cấu trúc điều kiện, lựa chọn	11
1.3.1 Câu lệnh điều kiện if.....	11
1.3.2 Câu lệnh lựa chọn switch	12
1.4 Cấu trúc Lặp.....	13
1.4.1 Câu lệnh for.....	13
1.4.2 Câu lệnh while	15
1.4.3 Câu lệnh do while.....	17
1.4.4 Câu lệnh For Each	18
1.5 Return, Break, Continue	18
1.6 Xử lý ngoại lệ (Exception).....	19
1.6.1 Ngoại lệ là gì?	19
1.6.2 Cấu trúc câu lệnh try... catch... finally....	20
1.6.3 Phát sinh và bắt giữ ngoại lệ	20
1.6.4 Câu lệnh <i>throw</i>	21
1.6.5 Dẫn xuất một ngoại lệ	21
1.7 Các bước Debug chương trình.....	22
1.8 Bài tập	24
Chương 2: HÀM, MẢNG, TẬP HỢP	26

2.1	Hàm	26
2.1.1	Định nghĩa	26
2.1.2	Khai báo:	26
2.1.3	Lời gọi hàm.....	26
2.1.4	Truyền tham số cho hàm	27
2.2	Mảng	29
2.2.1	Giới thiệu mảng trong C#	29
2.2.2	Mảng 1 chiều	29
2.2.3	Mảng nhiều chiều	31
2.2.4	Mảng răng cưa.....	31
2.3	Tập hợp - Collections	32
2.3.1	List.....	32
2.3.2	ArrayList.....	33
2.3.3	Hashtable	34
2.4	Kiểu liệt kê - Enumeration.....	35
2.5	Kiểu cấu trúc - Struct	36
2.6	Optional & Named Parameters	36
2.6.1	Optional Parameter – Tham số mặc định.....	36
2.6.2	Named Parameter – Tham số được đặt tên	37
2.7	Bài tập	38
	Chương 3: ĐỐI TƯỢNG (OBJECT) VÀ LỚP (CLASS)	45
3.1	Khái niệm lớp (Class) & Đối tượng (Object).....	45
3.1.1	Lớp	45
3.1.2	Đối tượng	45
3.2	Thuộc tính (Field) & Phương thức (Method)	46
3.2.1	Thuộc tính (Field).....	46
3.2.2	Phương thức (Method).....	46
3.2.3	Bảng tầm vực thuộc tính truy cập	46
3.3	Constructors & Object Initialize	46
3.3.1	Constructor.....	46
3.3.2	Object Initialize	47
3.4	Properties và Automatic Properties	47
3.4.1	Properties	47

3.4.2	Automatic Properties	48
3.5	Static và Extension Method	48
3.5.1	Từ khóa static	48
3.5.2	Phương thức mở rộng	48
3.6	Kiểu Anonymous Type	49
3.7	Bài tập	49
Chương 4:	THIẾT KẾ GIAO DIỆN WEB	52
4.1	Ngôn ngữ HTML	52
4.1.1	Một số khái niệm	52
4.1.2	Giới thiệu HTML	53
4.1.3	Một số tag thường dùng	53
4.1.4	Cấu trúc của 1 trang web	54
4.1.5	Soạn thảo trang web	54
4.2	Bảng định kiểu – CASCADING STYLE SHEET (CSS)	58
4.2.1	Giới thiệu	58
4.2.2	Khởi động nhanh	58
4.2.3	Tạo style định dạng	59
4.2.4	Các thuộc tính CSS	60
4.2.5	Bộ chọn (Selector)	62
4.2.6	Qui tắc nạp chồng	69
4.3	JAVASCRIPT VÀ JQUERY	70
4.3.1	Giới thiệu	70
4.3.2	Đưa javascript vào trang	70
4.3.3	Jquery	70
Chương 5:	TỔNG QUAN VỀ ỨNG DỤNG WEB	72
5.1	Giới thiệu Ứng dụng Web	72
5.2	Nguyên lý hoạt động của ứng dụng Web	72
5.3	Các khái niệm	73
5.3.1	Web client (Browser)	73
5.3.2	Web server	73
5.3.3	Giao thức HTTP	73
5.3.4	Client Scripting và Server Scripting	74
5.4	Kiến trúc công nghệ ứng dụng web	74

5.5	Giới thiệu về ASP & ASP.NET	74
5.5.1	Giới thiệu về ASP	74
5.5.2	Giới thiệu về ASP.NET	76
Chương 6:	WEB SERVER CONTROL.....	79
6.1	Mô hình Tổ chức mã	79
6.2	Sự kiện của trang ASP.NET	81
6.2.1	Các sự kiện	81
6.2.2	Ví dụ sự kiện trang	82
6.3	Cơ chế PostBack	84
6.3.1	ViewState:	84
6.3.2	Thuộc tính IsPostBack:	84
6.3.3	Thuộc tính MaintainScrollPositionOnPostBack.....	86
6.4	Kẽ thura để mở rộng.....	86
6.5	ASP.NET Web Server Control	88
6.5.1	Giới thiệu	88
6.5.2	Các điều khiển cơ bản.....	89
6.5.3	Các điều khiển nâng cao	95
6.6	Các thành phần giao diện thường dùng của jQuery UI	105
6.6.1	Thiết kế theme.....	105
6.6.2	Liên kết thư viện JQuery	107
6.6.3	JQuery button	107
6.6.4	JQuery Datepicker	108
6.6.5	JQuery Tabs	110
6.6.6	JQuery Accordion	111
6.6.7	JQuery Dialog	111
6.7	Validation Controls	113
6.7.1	Kiểm lỗi của ASP.NET	114
6.7.2	Điều khiển RequiredFieldValidator.....	115
6.7.3	Điều khiển CompareValidator	115
6.7.4	Điều khiển Range Validator	115
6.7.5	Điều khiển Regular ExpressionValidator.....	116
6.7.6	Điều khiển CustomValidator	117
6.7.7	Điều khiển Validation Summary	119

6.8 Kiểm lỗi với JQuery	120
6.8.1 Khởi động nhanh với Jquery Validation.....	121
6.8.2 Luật kiểm lỗi do người dùng định nghĩa.....	124
Chương 7: TỔ CHỨC WEBSITE	127
7.1 MasterPage	127
7.1.1 Giới thiệu.....	127
7.1.2 Tạo MasterPage	128
7.1.3 Đánh dấu vùng thay đổi (ContentPlaceHolder).....	128
7.1.4 Áp dụng MasterPage.....	129
7.2 Web User Control.....	130
7.2.1 Giới thiệu.....	130
7.2.2 Tạo và sử dụng.....	131
7.3 Quốc tế hóa website	133
7.3.1 Giới thiệu.....	133
7.3.2 Tạo tài nguyên cục bộ	133
7.3.3 Quốc tế hóa trang web I18N.aspx.....	135
7.4 Theme & Skin.....	137
7.4.1 Giới thiệu.....	137
7.4.2 Tạo tập tin Skin.....	138
7.4.3 Áp dụng Skin cho các điều khiển	139
Chương 8: CHIA SẼ DỮ LIỆU	141
8.1 Đổi tượng Request, Response, Server.....	141
8.1.1 Đổi tượng Request	141
8.1.2 Đổi tượng Response	142
8.1.3 Ví dụ minh họa Request, Response	143
8.1.4 Đổi tượng Server	145
8.2 Truyền tham số giữa các trang ASP.NET	148
8.2.1 Truyền dữ liệu theo phương thức POST.....	148
8.2.2 Truyền dữ liệu theo phương thức GET.....	148
8.2.3 Truyền dữ liệu sử dụng Cross-Page.....	150
8.3 Session và Cookies.....	153
8.3.1 Đổi tượng Cookies	153
8.3.2 Đổi tượng Session	154

8.4	Đối tượng Application.....	156
8.4.1	Mục đích.....	156
8.4.2	Sử dụng biến Application	157
8.4.3	Thí dụ minh họa sử dụng biến Application	157
8.5	Tập tin Global.asax	158
8.5.1	Mục đích.....	158
8.5.2	Cấu trúc tập tin Global.asax	159
8.5.3	Các sự kiện trong tập tin Global.asax	160
8.6	Tập tin Web.config.....	161
8.6.1	Cấu trúc tập tin web.config	161
8.6.2	Thiết lập các cấu hình.....	161
Chương 9:	SQL SERVER 2012	164
9.1	Tổng quan về SQL và cơ sở dữ liệu quan hệ.....	164
9.1.1	Khái niệm SQL	164
9.1.2	Vai trò của SQL	164
9.1.3	Mô hình dữ liệu quan hệ	165
9.2	Bảng và ràng buộc.....	165
9.2.1	Bảng (Table).....	165
9.2.2	Khóa chính của bảng (Primary Key)	165
9.2.3	Mối quan hệ (Relationship) và khóa ngoại (Foreign Key).....	165
9.3	Sơ lược về câu lệnh SQL.....	166
9.3.1	Các câu lệnh	166
9.3.2	Quy tắc sử dụng tên trong SQL	166
9.4	Kiểu dữ liệu.....	167
9.5	Toán tử.....	167
9.6	View, Stored Procedure, Trigger, Function	168
9.6.1	Bảng ảo – View	168
9.6.2	Stored Procedure	168
9.6.3	Trigger	169
9.6.4	Function	169
Chương 10:	ADO.NET & WEB FORM	170
10.1	Kiến Trúc ADO.NET	170
10.1.1	ADO.Net là gì?	170

10.1.2	Kiến trúc ADO.Net	170
10.2	Data Provider	171
10.2.1	Đối tượng Connection	171
10.2.2	Đối tượng Command	173
10.3	Data Classes.....	174
10.3.1	DataTable.....	174
10.3.2	DataSet	176
10.3.3	DataView.....	177
10.3.4	DataRelation	177
10.4	Kết nối Và Thao tác Dữ liệu	178
10.4.1	Tạo kết nối	179
10.4.2	Truy xuất dữ liệu.....	179
10.4.3	Thao tác dữ liệu	181
10.5	Ràng buộc (Binding) Dữ liệu với Data Controls	183
10.6	ADO.Net & SQL DataSource Control	192
Chương 11:	LINQ TO SQL & SQL.NET	199
11.1	Kiến trúc 2 lớp	199
11.2	Xây dựng ứng dụng ASP.NET theo kiến trúc 2 Lớp.....	200
11.3	ObjectDataSource control	205
11.4	LINQ to SQL.....	207
11.4.1	Mô hình hóa cơ sở dữ liệu dùng LINQ to SQL.....	207
11.4.2	Lớp DataContext trong LINQ to SQL	209
11.4.3	Một số thao tác dữ liệu trong LINQ to SQL	209
11.4.4	Các bước phát triển ứng dụng LINQ to SQL	210
11.5	LinqDataSource Control	213
11.6	LINQ to DataSet	215
Chương 12:	AJAX & ASP.NET	217
12.1	Giới thiệu Ajax	217
12.2	Cơ chế làm việc của ajax	217
12.2.1	Cơ chế truyền thông đồng bộ	218
12.2.2	Cơ chế truyền thông bất đồng bộ	218
12.3	Các bước xây dựng một ứng dụng ajax.....	219
12.4	ASP.NET Ajax	221



12.4.1	ASP.NET ajax và cơ chế gọi ngược.....	221
12.4.2	ASP.NET Ajax Server Control	223
12.5	Giới thiệu Ajax Control Toolkit.....	226
12.5.1	Tải Ajax Control Toolkit.....	226
12.5.2	Thêm Ajax Control Toolkit vào VS Toolbox	227
12.5.3	Một số điều khiển trong Ajax Control Toolkit.....	229
12.6	Bài Tập Thực Hành	243
12.7	Sử dụng Ajax với JQuery	250
Chương 13:	WEB SERVICE	252
13.1	Giới thiệu Web Service	252
13.2	Kiến trúc Web Service	252
13.3	Xây dựng ứng dụng Web Service	253
13.4	Xây dựng ứng dụng Web Client.....	254
13.5	Asynchronous Call trong Web Service	256
Chương 14:	TIỆN ÍCH.....	257
14.1	Authentication	257
14.1.1	Form Authentication	257
14.1.2	Passport Authentication	257
14.1.3	Windows Authentication.....	258
14.2	Authorization	258
14.2.1	File Authorization	258
14.2.2	Url Authorization	258
14.2.3	Các bước cấu hình Authentication và Authorization.....	259
14.3	Mã hóa dữ liệu.....	261
14.3.1	Thuật toán MD5	261
14.3.2	Thuật toán SHA-1.....	261
14.3.3	Thuật toán SHA5 (SHA-2 512 bit)	262
14.3.4	Ứng dụng thuật toán mã hóa	262
14.4	Khảo sát ngôn ngữ XML	263
14.4.1	Khái niệm về XML.....	263
14.4.2	Cấu trúc tài liệu XML.....	263
14.4.3	Tạo một tài liệu XML.....	264
14.4.4	Sử dụng lớp Xdocument.....	264

14.5 Sử Dụng Linq To Xml	269
14.5.1 Tham chiếu đến file tài liệu	270
14.5.2 Lưu tài liệu	270
14.5.3 Tham chiếu đến các phần tử trong tài liệu.....	271
14.5.4 Tham chiếu đến thuộc tính.....	271
14.5.5 Thêm phần tử vào tài liệu	271
14.5.6 Tìm kiếm phần tử.....	271
14.5.7 Cập nhật phần tử	272
14.5.8 Xóa phần tử.....	273
14.6 Giới thiệu URL Routing	274
Chương 15: WEB TIN TỨC	278
15.1 Database.....	278
15.2 Tạo website tin tức	278
15.2.1 Tạo mới Website	278
15.2.2 Tạo kết nối database	279
15.3 Viết code trang chủ.....	279
15.3.1 Tạo trang Master Page.....	279
15.3.2 Thiết kế giao diện.....	279
15.3.3 Tạo các hàm xử lý :	281
15.4 Tạo trang Default.aspx:.....	282
15.5 Tạo Web User Control Menu	285
15.6 Tạo Web User Control Tin mới nhất.....	287
15.7 Tạo Web User Control tin xem nhiều	288
15.8 Tạo Web User Control Login	289
15.9 Tạo trang TinTrongLoai.aspx	291
15.10 Tạo trang ChiTietTin.aspx.....	293
15.11 Viết code trang quản trị.....	295
15.11.1 Tạo trang BackEndMasterPage	295
15.11.2 Tạo trang Default.aspx	296
15.11.3 Tạo trang TheLoai.aspx	296
15.11.4 Tạo trang LoaiTin.aspx	301
15.11.5 Tạo trang Tin.aspx	305
Chương 16: WEBSITE BÁN HÀNG	313

16.1	Tài nguyên & CSDL	313
16.1.1	Tài nguyên	313
16.1.2	Lược đồ CSDL	313
16.1.3	Tạo website bán hàng	315
16.1.4	Tạo kết nối database	315
16.2	Master Page và trang chủ	316
16.2.1	Tạo trang Master Page	316
16.2.2	Thiết kế giao diện	316
16.2.3	Tạo trang Default.aspx	317
16.2.4	Tạo WebUserControl Danh mục sản phẩm	320
16.3	Trang SPTheoLoai.aspx	322
16.4	Trang ChiTietSP.aspx	325
16.5	Trang đăng nhập (Login.aspx)	329
16.6	Xử lý nút Mua hàng	330
16.7	Trang GioHang.aspx	331
16.8	Trang ThanhToan.aspx	331
16.9	Trang lịch sử mua hàng	333
16.10	Phần Quản trị	333
16.10.1	Layout trang Quản trị	333
16.10.2	Quản lý Chủng loại	334
16.10.3	Quản lý Đơn hàng	334
16.10.4	Quản lý Sản phẩm	335
16.10.5	Quản lý Tài khoản người dùng	337
16.11	Yêu cầu thêm	338
	Chương 17: UPLOAD FILE LÊN HOST	339

Chương 1:

TỔNG QUAN NGÔN NGỮ C#

Sau khi học xong chương này, học viên có khả năng:

- Mô tả được các kiểu dữ liệu cơ bản trong C#.
- Sử dụng được các cấu trúc lệnh.
- Xử lý ngoại lệ và biết Debug chương trình.

1.1 Giới thiệu C#

C# (đọc là Cee Sharp) là ngôn ngữ lập trình cho nền tảng .Net platform. Phiên bản đầu tiên năm 2002, C# trải qua các phiên bản 1.0, 2.0, 3.0, 4.0 và nay là 5.0. C# được phát triển bởi đội ngũ kỹ sư của Microsoft, trong đó người dẫn đầu là Anders Hejlsberg và Scott Wiltamuth.

Ngôn ngữ C# khá đơn giản, với khoảng 80 từ khóa và hơn 10 kiểu dữ liệu được xây dựng sẵn. Tuy nhiên, nó có ý nghĩa cao khi thực thi những khái niệm lập trình hiện đại. C# bao gồm tất cả những hỗ trợ cho cấu trúc, thành phần component, lập trình hướng đối tượng. Những tính chất đó hiện diện trong một ngôn ngữ lập trình hiện đại. Và C# hội đủ những điều kiện như vậy, hơn nữa nó được xây dựng trên nền tảng của hai ngôn ngữ mạnh nhất là C++ và Java.

Trong ngôn ngữ C#, mọi thứ liên quan đến khai báo lớp đều được tìm thấy trong phần khai báo của nó. Định nghĩa một lớp trong ngôn ngữ C# không đòi hỏi phải chia ra tập tin header và tập tin nguồn giống như trong ngôn ngữ C++. Hơn thế nữa, ngôn ngữ C# hỗ trợ kiểu XML, cho phép chèn các tag XML để phát sinh tự động các document cho lớp.

C# cũng hỗ trợ giao diện interface. Một lớp chỉ có thể kế thừa duy nhất từ một lớp cha (tức là không cho đa kế thừa như trong ngôn ngữ C++), tuy nhiên một lớp có thể thực thi nhiều giao diện. Khi một lớp thực thi một giao diện thì nó sẽ cung cấp chức năng thực thi giao diện. C# cũng hỗ trợ cấu trúc, nhưng khái niệm về ngữ nghĩa của nó thay đổi khác với C++. Trong C#, một cấu trúc được giới hạn, là kiểu dữ liệu nhỏ gọn, và khi tạo thể hiện thì nó yêu cầu ít hơn về hệ điều hành và bộ nhớ so với một lớp. Một cấu trúc không thể kế thừa từ một lớp (hoặc kế thừa một cấu trúc khác), nhưng một cấu trúc có thể thực thi một giao diện.

C# cung cấp những đặc tính hướng thành phần (component-oriented): những thuộc tính, những sự kiện. Lập trình hướng thành phần được hỗ trợ bởi CLR cho phép lưu trữ metadata với mã nguồn cho một lớp. Metadata mô tả cho một lớp, bao gồm những phương thức và những thuộc tính của nó, cũng như những bảo mật cần thiết và những thuộc tính khác. Mã nguồn chứa đựng những logic cần thiết để thực hiện những chức năng của nó... Do vậy, một lớp được biên dịch như là một khối self-contained, môi trường hosting biết được cách đọc metadata của một lớp và mã nguồn cần thiết mà không cần những thông tin khác để sử dụng nó.

1.2 Các khái niệm cơ bản

1.2.1 Cấu trúc chương trình

Bắt đầu từ chương trình "Hello World" đơn giản trong C#:

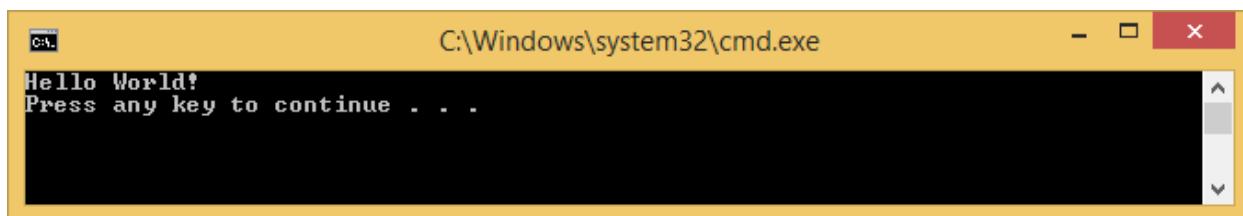
```
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Hello World!");
    }
}
```

Giải thích:

- Phần đầu của chương trình là các khai báo thư viện với từ khóa **using**, theo sau là tên của thư viện cần khai báo.
- Toàn bộ chương trình được “đóng gói” trong một **namespace**. Bạn sẽ rõ hơn về **namespace** trong các phần sau.
- Bản thân chương trình trong C# là một lớp (*class*), như bạn thấy, có tên là **Program**. Lớp này chứa hàm *Main* – điểm bắt đầu của chương trình.
- Hàm Main ở trên chỉ chứa duy nhất một câu lệnh: **Console.WriteLine("Hello World!");**

Để viết ra màn hình dòng chữ: Hello World

Gõ tổ hợp phím *CTRL+F5* để chạy chương trình, bạn sẽ được kết quả:

**Hàm Main:**

Trong C#, hàm *Main()* được viết ký tự hoa đầu, và có thể trả về giá trị *void* hay *int*. Khi chương trình thực thi, CLR gọi hàm *Main()* đầu tiên, hàm *Main()* là đầu vào của chương trình, và mỗi chương trình phải có một hàm *Main()*. Đôi khi chương trình có nhiều hàm *Main()* nhưng lúc này ta phải xác định các chỉ dẫn biên dịch để CLR biết đâu là hàm *Main()* đầu vào duy nhất trong chương trình.

1.2.2 Định danh (identifier)

Định danh được sử dụng để đặt cho các đối tượng trong chương trình như: tên biến, tên kiểu dữ liệu, tên hàm, tên lớp, tên thuộc tính,...

Ngôn ngữ lập trình cũng giống như ngôn ngữ tự nhiên, chúng đều có cú pháp và ngữ nghĩa. Do đó việc đặt tên cho các đối tượng trong chương trình là rất quan trọng, làm sao phải đảm bảo được hai yếu tố: *đúng cú pháp* và *dễ đọc*.

Quy tắc đặt tên:

- Tên (định danh – identifier) là một chuỗi kí tự bắt đầu bằng một chữ cái hoặc dấu gạch nối “ _ ”, được dùng để đặt cho các đối tượng trong chương trình (như lớp, thuộc tính, phương thức, biến, kiểu dữ liệu,.. ..)
- C# phân biệt chữ in hoa và in thường (case sensitive).

Chú ý:

- Tên không được bắt đầu bằng một chữ số.
- Tên không được trùng với từ khóa.
- Tên không được chứa khoảng trắng

Ví dụ: đặt tên như sau là **sai** cú pháp:

```
int class = 3;           //tên trùng với từ khóa (class)
double 1abc = 12.3;      //tên bắt đầu bằng chữ số 1
```

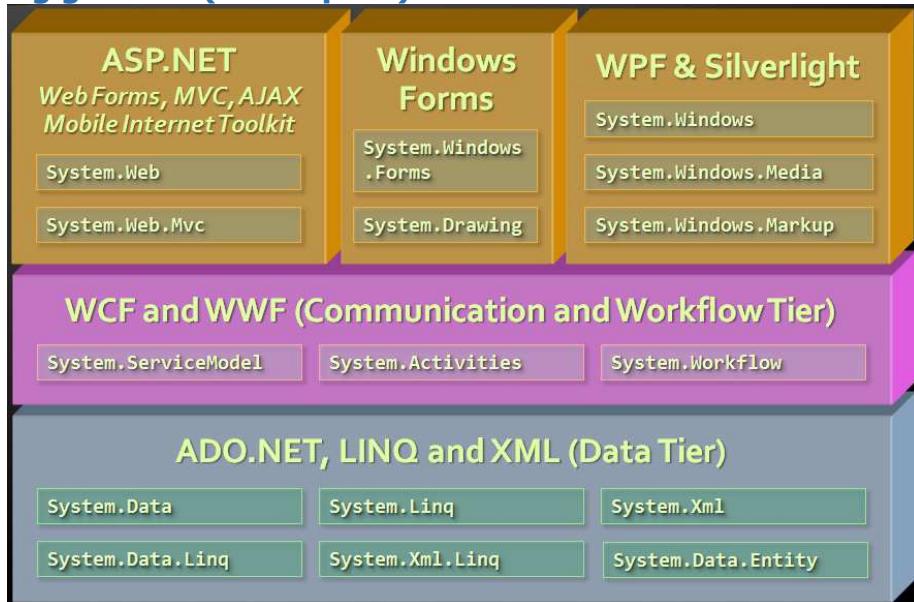
Trong việc đặt tên ngoài những quy tắc bắt buộc, các lập trình viên thường tìm cách đặt tên sao cho dễ đọc.

Ví dụ:

Tên của biến (**myDictionary**) thường được đặt theo cách đặt tên cú pháp lạc đà.

Tên của hàm (**DrawLine**) và thuộc tính (**ColorBackground**) đặt theo cú pháp Pascal.

1.2.3 Không gian tên (namespace)



.NET Framework cung cấp một thư viện các lớp đồ sộ, có tên là FCL (Framework Class Library). Trong đó Console chỉ là một lớp nhỏ trong hàng ngàn lớp trong thư viện. Mỗi lớp có một tên riêng, vì vậy FCL có hàng ngàn tên như *ArrayList*, *Dictionary*, *FileSelector*,...

Điều này làm nảy sinh vấn đề, người lập trình không thể nào nhớ hết được tên của các lớp trong .NET Framework. tệ hơn nữa là sau này có thể ta tạo lại một lớp trùng tên với lớp đã có chẳng hạn. Ví dụ trong quá trình phát triển một ứng dụng ta cần xây dựng một lớp từ điển và lấy tên là *Dictionary*, và điều này dẫn đến sự tranh chấp khi biên dịch vì C# chỉ cho phép một tên duy nhất. Khi đó chúng ta phải đổi tên của lớp từ điển mà ta vừa tạo thành một cái tên khác chẳng hạn như *myDictionary*. Do đó sẽ làm cho việc phát triển các ứng dụng trở nên phức tạp, cồng kềnh. Đến một sự phát triển nhất định nào đó thì chính là cơn ác mộng cho nhà phát triển.

Để giải quyết vấn đề này là ta tạo ra một *namespace*. *Namespace* sẽ hạn chế phạm vi của một tên, làm cho tên này chỉ có ý nghĩa trong vùng đã định nghĩa. Các *namespace* để phân thành các vùng cho các lớp trùng tên không tranh chấp với nhau.

Như vậy nếu .NET framework có xây dựng một lớp *Dictionary* bên trong *namespace System.Collections*, và tương ứng ta có thể tạo một lớp *Dictionary* khác nằm trong *namespace Lab2*, điều này hoàn toàn không dẫn đến sự tranh chấp với nhau.

Một ví dụ về namespace:

```
namespace Lab2
{
  namespace NS1
  {
    class class1
```

```

{
    static public void method1()
    {
        //làm công việc gì đó
    }
}
namespace NS2
{
    class class1
    {
        static public void method1()
        {
            //làm công việc gì đó
        }
    }
    class Program
    {
        static void Main(string[] args)
        {
            Lab2.NS1.class1.method1(); //gọi phương thức method1 trong namespace
NS1
            Lab2.NS2.class1.method1(); //gọi phương thức method1 trong namespace
NS2
        }
    }
}

```

1.2.4 Toán tử `.'

Trong ví dụ trên dấu `.' được sử dụng để truy cập đến phương thức hay dữ liệu trong một lớp (trong trường hợp này phương thức là `method1()`), và ngăn cách giữa tên lớp đến một *namespace* xác định (*namespace* `Lab2.NS1` và lớp `class1`). Việc thực hiện này theo hướng từ trên xuống, trong đó mức đầu tiên *namespace* là `Lab2`, tiếp theo là *namespace* `NS1`, tên lớp `class1` và cuối cùng là truy cập đến các phương thức hay thuộc tính của lớp.

1.2.5 Từ khóa using

Để làm cho chương trình gọn hơn, và không cần phải viết từng *namespace* cho từng đối tượng, C# cung cấp từ khóa là *using*, sau từ khóa này là một *namespace* hay subnamespace với mô tả đầy đủ trong cấu trúc phân cấp của nó.

Ví dụ: bằng việc khai báo

```
using Lab2.MyFolder;
```

Ta có thể viết gọn hơn:

```
StatementDemo test = new StatementDemo();
```

Thay vì:

```
MyFolder.StatementDemo test = new MyFolder.StatementDemo();
```

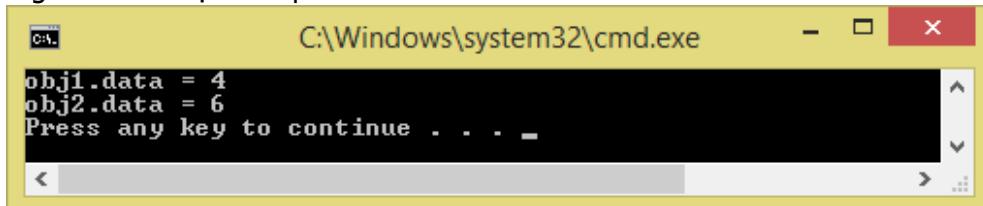
1.2.6 Từ khóa static

Theo mặc định, các thành phần trong một lớp là *non static*, có nghĩa là khi một đối tượng thuộc lớp này được tạo (khi ta gọi hàm dựng – *constructor*) thì một vùng nhớ riêng được cấp phát để lưu trữ các thành phần của đối tượng này.

Ví dụ:

```
class MyClass
{
    public int data;
    public void Method()
    {
        //làm công việc gì đó
    }
}
class Program
{
    static void Main()
    {
        MyClass obj1 = new MyClass();
        MyClass obj2 = new MyClass();
        obj1.data = 4;
        obj2.data = 6;
        Console.WriteLine("obj1.data = " + obj1.data.ToString());
        Console.WriteLine("obj2.data = " + obj2.data.ToString());
        obj1.Method();
        obj2.Method();
    }
}
```

Chạy chương trình ta được kết quả:



Điều này khẳng định: mỗi đối tượng của `MyClass` lưu dữ liệu của mình một cách riêng biệt. Tương tự như vậy cho các thành phần khác (phương thức, thuộc tính,...).

Trong một số trường hợp, chúng ta cần các đối tượng sử dụng chung một thành phần nào đó. C# giải quyết vấn đề này bằng cách “chuyển” thành phần này cho lớp thay vì “để” ở đối tượng.

Ví dụ:

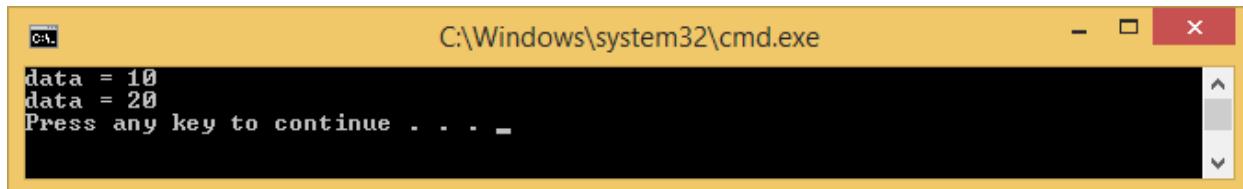
```
class OtherClass
{
    static public int data;
    public void DisplayData()
    {
        Console.WriteLine("data = " + data.ToString());
    }
    static public void Method()
    {
        //làm công việc gì đó
    }
}
class Program
{
    static void Main()
    {
        OtherClass.data = 10;
```

```

        OtherClass obj1 = new OtherClass();
        obj1.DisplayData();
        OtherClass obj2 = new OtherClass();
        OtherClass.data = 20;
        obj2.DisplayData();
        OtherClass.Method(); //phương thức của lớp!
    }
}

```

Ta có kết quả:



Kết quả này chứng tỏ dòng lệnh `OtherClass.data = 20;` đã tác động đến "thành phần" dữ liệu của đối tượng `obj2`. Hay chính xác hơn là chỉ có một vùng dữ liệu được tạo ra cho mọi đối tượng của lớp `OtherClass`.

1.2.7 Biến (variable)

Khái niệm:

Biến là khái niệm rất quan trọng trong các ngôn ngữ lập trình. Biến là một vùng nhớ (trong bộ nhớ sơ cấp – RAM) được đặt tên, dùng để lưu trữ dữ liệu. Mỗi biến có một kiểu dữ liệu xác định.

Cú pháp Khai báo:

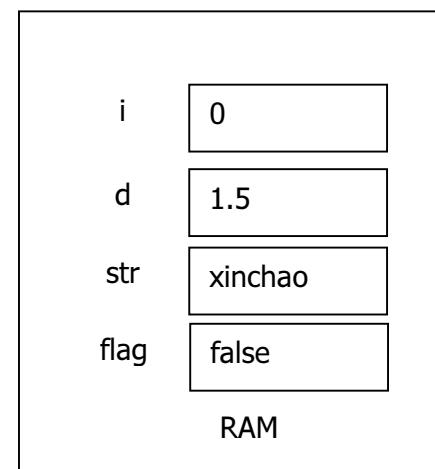
<Tên kiểu> Tên_bien;

Ví dụ sau đây khai báo và khởi tạo giá trị ban đầu cho các biến:

```

static void Main(string[] args)
{
    int i = 0;
    double d = 1.5;
    string str = "xin chao";
    bool flag = false;
}

```

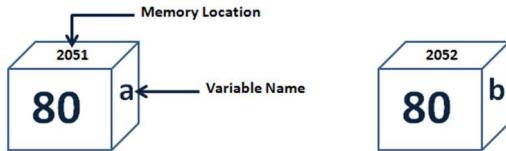


Biến được lưu trữ như thế nào?

Tùy thuộc vào kiểu dữ liệu mà mối quan hệ giữa tên biến và vùng dữ liệu chứa giá trị của biến sẽ khác nhau. Cụ thể:

- Các biến thuộc kiểu giá trị - value type (như int, double, enum,.. ..) thì tên biến là tên vùng nhớ trực tiếp chứa giá trị của biến.

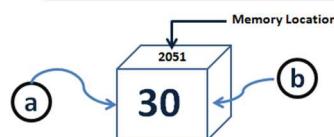
VALUE TYPE - Example



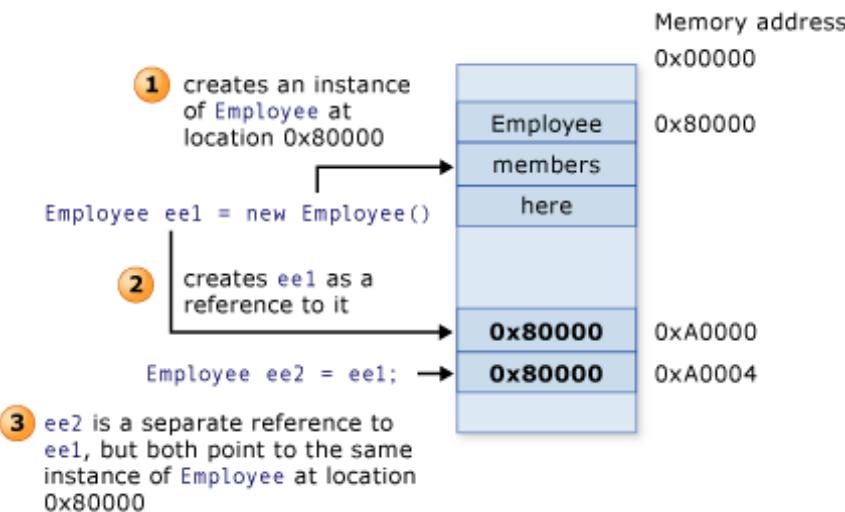
```
int a;
a=80;
b=a; This will just create a copy of "a" in other  
memory location with variable named "b"
```

- Còn các biến thuộc kiểu tham chiếu - reference type (như class, string,.. ..) thì tên biến là tên vùng nhớ chứa giá trị tham chiếu đến giá trị thực của biến (giá trị thực của biến được lưu trong một vùng nhớ khác).

Reference Type - Example



```
Employee a=new Employee();
a.age = 26;
Employee b=a; This will just create a new variable which will  
point to same location as "a" points.
b.age = 30
```



Tâm vực của biến (variable scope): cho biết biến có hiệu lực ở đâu và khi nào.

Biến mức lớp: là các biến được khai báo như một *fields non static* của một lớp. Biến này có tầm vực hoạt động trong toàn bộ lớp mà nó được khai báo. Ví dụ:

```
class VariableScope
{
    int data;//biến mức lớp.
    //...
}
```

Biến mức phương thức (hàm): là các biến được khai báo (cục bộ) trong một phương thức hoặc một hàm. Chúng chỉ có tầm vực hoạt động trong phương thức hoặc hàm này. Ví dụ:

```
class VariableScope
{
    int data; //biến có tầm vực lớp.
    //...
    public int TinhTong(int n)
    {
        int i, t = 0; //biến mức phương thức, hàm.
        for (i = 1; i <= n; i++) t += i;
        return t;
    }
    //truy xuất biến t và i ở đây là không hợp lệ
}
```

1.2.8 Từ khóa var

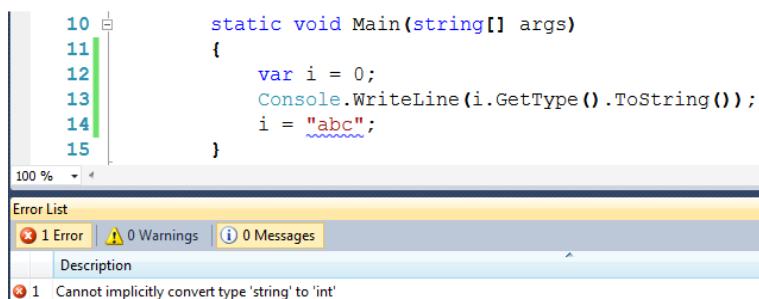
Mục đích: để khai báo biến (trong phạm vi hàm, phương thức)

Yêu cầu: phải khởi tạo giá trị khi khai báo.

Kiểu dữ liệu: ngầm định kiểu dữ liệu của biến là kiểu của biểu thức bên phải phép gán.

Khai báo tường minh	Sử dụng từ khóa var
<p>Explicit[direct]</p> <p>You are <u>directly</u> defining the data types.</p>	<p>Implicit[indirect]</p> <p>By looking at the <u>right hand side</u> data the <u>left hand side</u> <u>data type</u> is defined.</p>

Ví dụ:



1.2.9 Lệnh và khôi lệnh

Lệnh là một khôi (*block*) mã thực hiện một công việc xác định. Lệnh trong C# kết thúc bằng dấu ";"

C# phân chia các lệnh theo nhóm, như sau:

Loại	Từ khóa
Lệnh lựa chọn (rẽ nhánh)	if, else, switch, case
Lệnh lặp	do, for, foreach, in, while
Lệnh nhảy (jump)	break, continue, default, goto, return
Lệnh điều khiển ngoại lệ	throw, try, catch, finally
Đã / chưa kiểm tra	checked, unchecked
Lệnh fixed	fixed
Lệnh khóa	lock

Lệnh gán (assignment statement): sử dụng để gán giá trị cho một biến.

Ví dụ: `var i = 0;`

Các lệnh xuất nhập console

C# cung cấp các phương thức *static* gắn với lớp *Console* để nhập và xuất dữ liệu bàn phím – màn hình.

- Nhập dữ liệu từ bàn phím:

```
Console.Read();      //đọc 1 kí tự
Console.ReadLine(); //đọc 1 dòng
Console.ReadKey();   //đọc 1 phím
```

- Xuất dữ liệu ra màn hình:

```
Console.Write();
Console.WriteLine(); //viết xong xuống dòng mới
```

Ví dụ:

```
class CommandDemo
{
    double diem;
    string hoTen;
    public void NhapGiaTriChoBien()
    {
        Console.Write("nhap ho ten = ");
        hoTen = Console.ReadLine();
        Console.Write("nhap diem = ");
        diem = double.Parse(Console.ReadLine());
        Console.WriteLine("xin chao {0}, diem cua ban = {1}", hoTen, diem);
    }
}
```

Khối lệnh: là một nhóm các câu lệnh đặt trong cặp dấu { và }. Toàn bộ khối lệnh được xem như một lệnh (đơn).

Ví dụ:

```
if (a > b)
{
    temp = a;
    a = b;
    b = temp;
}
```

1.2.10 Biểu thức

Tương tự như trong toán học, biểu thức bao gồm các toán hạng và toán tử (phép toán). Ví dụ:

`i*d + Math.Sqrt(Math.Sin(Math.PI)) + str.Length`

Là một biểu thức gồm các toán hạng, ở đây là các biến (i, d, str,.. ..) và các toán tử (*, +,...)

Mỗi biểu thức có một giá trị (khi biết giá trị của các toán hạng trong biểu thức đó).

Các loại phép toán:

- Phép toán số học: +, -, *, /
- Phép toán logic: &&, ||, !
- Phép toán quan hệ: >, <, >=, <=, !=
- Phép toán tăng giảm: ++, --
- Phép gán: =, +=, -=, *=, /=

Trong một biểu thức, toán hạng có thể là hằng, biến, hàm, hay một biểu thức con.

1.2.11 Chú thích (comment)

Chú thích chỉ đơn thuần là các văn bản được sử dụng để giải thích, ghi chú trong chương trình. Chúng là công cụ để các nhà phát triển “giao tiếp” với nhau (có khi là với chính họ) chứ không phải với máy tính!

Có nhiều cách để chú thích trong C#. Ví dụ:

```
/*
 * chú thích trên nhiều dòng
 *
 */
```

```
//chú thích trên một dòng
```

```
///chú thích lặp lại
///chú thích lặp lại
///chú thích lặp lại
```

Đặc biệt là chú thích dạng XML, thường được sử dụng cho các phương thức, các hàm. Ví dụ:

```
///<summary>
///Returns the Square of the specified number
</summary>
///<param name = "x">The number to square</param>
///<returns>The squared value.</returns>
static public double Square(double x)
{
    return x * x;
}
```

Vì C# hỗ trợ cơ chế “gợi ý” khi gọi các phương thức được chú thích theo dạng này:

```
double result = NS1.class1.Square()
```

double class1.Square(double x)
Returns the Square of the specified number
x: The number to square

Ví dụ: Viết *comment* cho hàm.

Bước 1: Viết hàm

```
static public dynamic CodeSnippetDemo(dynamic param1, dynamic param2)
{
    return param1 + param2;
}
```

Bước 2: Đặt con trỏ tại dòng phía trên tiêu đề của hàm, gõ *///*. Net sẽ tự động sinh comment:

```
/// <summary>
///
/// </summary>
/// <param name="param1"></param>
/// <param name="param2"></param>
/// <returns></returns>
```

```
static public dynamic CodeSnippetDemo(dynamic param1, dynamic param2)
{
    return param1 + param2;
}
```

Bước 3: Viết nội dung cho comment

```
/// <summary>
/// Demo CodeSnippet
/// </summary>
/// <param name="param1">Tham số thứ 1</param>
/// <param name="param2">Tham số thứ 2</param>
/// <returns>Trả về tổng 2 tham số</returns>
static public dynamic CodeSnippetDemo(dynamic param1, dynamic param2)
{
    return param1 + param2;
}
```

Bước 4: Gọi hàm

```
CodeSnippetDemo (
    dynamic Program.CodeSnippetDemo(dynamic param1, dynamic param2)
    Demo CodeSnippet
    param1: Tham số thứ 1
```

1.3 Cấu trúc điều kiện, lựa chọn

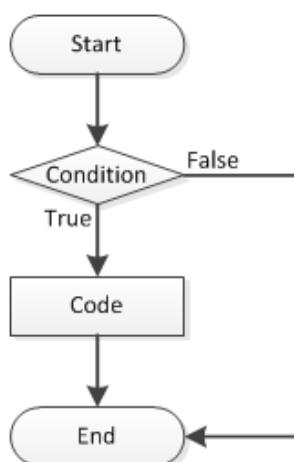
Trong quá trình xử lý, đôi khi chúng ta phải chọn 1 trong 2 hoặc nhiều công việc. Tùy thuộc vào một điều kiện nào đó, C# cung cấp cho chúng ta các câu lệnh *if* và *switch* để thực hiện điều này.

1.3.1 Câu lệnh điều kiện if

* Cú pháp:

```
if(conditional expression)
    <statement1>
    else
        <statement2>
```

* Sơ đồ khối:



* Các ví dụ:

Ví dụ 1: Hàm sau đây trả về số lớn nhất trong 3 số mà nó nhận vào.

```
///<summary>Tìm số lớn nhất trong 3 số</summary>
///<param name = "x">Số thứ nhất</param>
///<param name = "y">Số thứ hai</param>
///<param name = "z">Số thứ ba</param>
///<returns>Số lớn nhất</returns>
static public double SoLonNhat(double x, double y, double z)
{
    double max = x;
    if (max > y) max = y;
    if (max > z) max = z;
    return max;
}
```

* *Chú ý:* Các câu lệnh if có thể lồng nhau nhiều cấp.

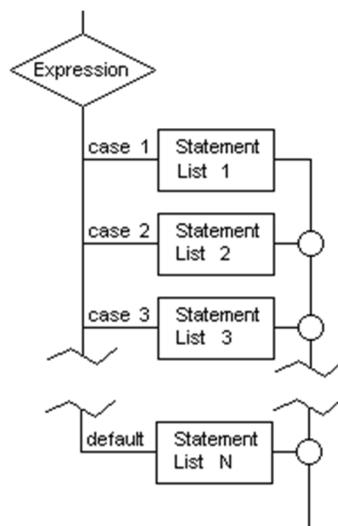
1.3.2 Câu lệnh lựa chọn switch

Câu lệnh *switch* cung cấp một cấu trúc điều khiển lựa chọn để thực hiện một công việc nào đó dựa trên giá trị của biểu thức tại thời điểm run-time.

* Cú pháp:

```
switch(Expression)
{
    case label_1: <statement_1>
    case label_2: <statement_2>
    case label_3: <statement_3>
    ...
    case label_k: <statement_k>
    [default: <statement_k+1>]
}
```

* Sơ đồ khôi:



* Các ví dụ:

Ví dụ 1: Hàm sau đây sẽ tính số tiền khách hàng phải trả khi thuê xe. Dữ liệu đầu vào là số ngày thuê và loại xe. Mỗi loại xe có một giá thuê riêng, sử dụng cấu trúc *switch* để xác định giá thuê xe dựa trên loại xe. Nếu số ngày thuê lớn hơn 10 thì giảm giá thuê xe 10%.

```
///<summary>Tính tiền thuê xe</summary>
///<param name = "n">Số ngày thuê</param>
///<param name = "loai">Loại xe</param>
///<returns>Số tiền phải trả</returns>
static public double TienThueXe(int n, string loai)
{
    double gia = 0;
    switch (loai.ToUpper())
    {
        case "A": gia = 1000000; break;
        case "B": gia = 700000; break;
        case "C": gia = 500000; break;
    }
    if (n > 10) gia = gia * 0.9;
    return gia * n;
}
```

Ví dụ 2: Đoạn mã sau xác định số ngày của tháng bất kỳ trong năm 2000

```
int thang;
int soNgay;
Console.WriteLine("thang = ");
thang = int.Parse(Console.ReadLine());
switch (thang)
{
    case 1:
    case 3:
    case 5:
    case 7:
    case 8:
    case 10:
    case 12: soNgay = 31; break;
    case 4:
    case 6:
    case 9:
    case 11: soNgay = 30; break;
    case 2: soNgay = 29; break;
    default: soNgay = 0; break;
}
Console.WriteLine("so ngay = " + soNgay.ToString());
```

* Chú ý:

- Nhãn *default* trong cấu trúc *switch* là tùy chọn. Nếu có không có nhãn nào trước đó có giá trị bằng với giá trị của biểu thức thì *<statement_k+1>* sau nhãn *default* sẽ được thực hiện.
- Có thể sử dụng *break* để thoát khỏi cấu trúc *switch* khi cần thiết (Xem ví dụ 2 ở trên).

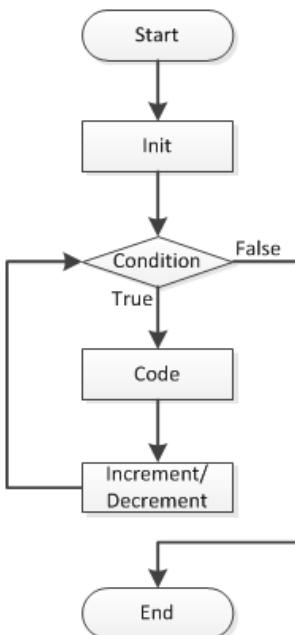
1.4 Cấu trúc Lặp

1.4.1 Câu lệnh for

* Cú pháp:

```
for(<init>;<condition>;<increment/decrement>)
    <statement(s)>
```

* Sơ đồ khôi:



* Các ví dụ:

Ví dụ 1: Tính tổng $S = 1 + 2 + \dots + n$

```
class Program
{
    static void Main(string[] args){
        int tong = 0;
        Console.Write("n = ");
        int n = int.Parse(Console.ReadLine());
        for (int i = 1; i <= n; i++) tong += i; //tong = tong + i;
        Console.WriteLine("tong = {0}", tong);
    }
}
```

Ví dụ 2: Tính tổng các phần tử dương trong mảng a.

```
class Program
{
    static void Main(string[] args){
        int tong = 0;
        //khai báo và khởi tạo mảng a:
        int[] a = { 1, -23, 6, 7, 3, -4, 8 };

        for (int i = 0; i < a.Length; i++)
            if(a[i] > 0) tong += a[i]; //tong = tong + a[i];
        Console.WriteLine("Tong cac phan tu duong = {0}", tong);
    }
}
```

Tương tự Ví dụ 2, chúng ta có thể sử dụng *break* và *continue* thay cho lệnh *if*:

```
for (int i = 0; true; i++)
{
    if (i == a.Length) break;//thoát for gần nhất
    if (a[i] <= 0) continue;
    tong = tong + a[i];
}
```

Ví dụ 3: Tính tổng $S = 1! + 2! + \dots + k!$

Cách 1: Sử dụng *for* lồng nhau

```
class Program
{
    static void Main(string[] args)
    {
        int tong = 0;
        Console.WriteLine("n = ");
        int n = int.Parse(Console.ReadLine());
        for (int i = 1; i <= n; i++)
        {
            //tính i giai thừa
            int gaiThua = 1;
            for (int j = 1; j <= i; j++)
                gaiThua *= j;
            //cộng vào tổng:
            tong += gaiThua;
        }
        Console.WriteLine("Tong = {0}", tong);
    }
}
```

Cách 2: Cải tiến cách 1

Nhận xét rằng tại bước lặp thứ i ở cách 1 chúng ta đã tính lại i giai thừa bằng cách nhân từ 1 đến i . Tuy nhiên, trước đó, tại bước $i-1$ chúng ta đã tính $i-1$ giai thừa rồi. Do vậy có thể tính i giai thừa ngay bằng lệnh:

`gaiThua = gaiThua * i;`

Chương trình được sửa lại chỉ dùng 1 vòng lặp *for* như sau:

```
int gaiThua = 1;
int tong = 0;
for (int i = 1; i <= n; i++)
{
    gaiThua *= i;
    tong += gaiThua;
}
```

* Chú ý:

- Có thể sử dụng *break* và *continue* trong *for* (xem ví dụ 3)
- Các câu lệnh *for* có thể lồng nhau nhiều cấp.

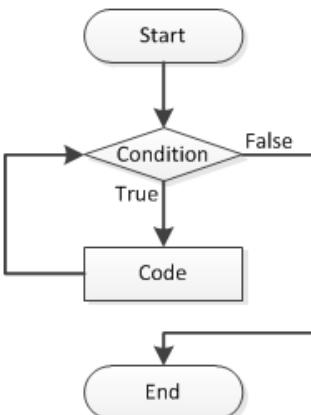
1.4.2 Câu lệnh while

Trong các ngôn ngữ lập trình *while* được xem là câu lệnh lặp điều kiện trước. Nghĩa là trong khi điều kiện còn thỏa mãn thì khối lệnh trong thân *while* còn tiếp tục được thực hiện.

* Cú pháp:

```
while(conditional expression)
{
    <statement(s)>
}
```

* Sơ đồ khối:



* Các ví dụ:

Ví dụ 1: Hàm sau đây sẽ đếm số chữ số của số nguyên n (với n là tham số đầu vào của hàm)

```
public static int DemSoChuSo(int n)
{
    int dem = 0;
    while (n > 0)
    {
        ++dem;
        n = n / 10;
    }
    return dem;
}
```

Ví dụ 2: Hàm sau đây tính tổng các số nguyên tố trong một mảng (số nguyên tố là số chỉ chia hết cho 1 và chính nó).

```
public static int TongcacSoNguyenTo(int[] a)
{
    int tong = 0;
    int so;
    int i = -1;
    while (true)
    {
        ++i;
        if (i == a.Length) break;
        //kiểm tra xem a[i] có nguyên tố hay không:
        so = 2;
        while (a[i] % so != 0) ++so;
        //nếu a[i] không nguyên tố thi bỏ qua
        if (so < a[i]) continue;
        //ngược lại thì cộng vào tổng:
        tong += a[i];
    }
    return tong;
}
```

* Chú ý: Chúng ta cũng có thể sử dụng *break* và *continue* trong lệnh *while* tương tự như ở trong lệnh *for*.

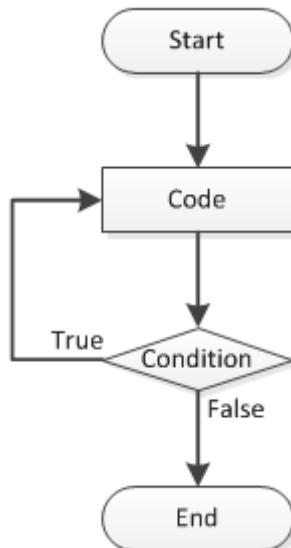
1.4.3 Câu lệnh do while

Do while được gọi là câu lệnh lặp điểu kiện sau. Nghĩa là việc kiểm tra điều kiện chỉ được thực hiện sau khi đã thực hiện khối lệnh *<statement(s)>* sau *do*

* Cú pháp:

```
do
{
    <statement(s)>
}
While(<conditional expression>);
```

* Sơ đồ khối:



* Các ví dụ:

Ví dụ 1: Chương trình sau yêu cầu người sử dụng nhập vào một giá trị số. Nếu không đúng là số thì yêu cầu nhập lại.

```
static void Main(string[] args)
{
    int so;
    bool thanhCong;
    do
    {
        Console.WriteLine("Hay nhap vao mot so = ");
        thanhCong = int.TryParse(Console.ReadLine(), out so);
        if (!thanhCong) Console.WriteLine("Gia tri khong hop le,
            vui long nhap lai!");
    }while (!thanhCong);
    Console.WriteLine("so ban da nhap = {0}", so);
}
```

* *Chú ý:* Tương tự như các lệnh *for* và *while*, chúng ta cũng có thể sử dụng *break* và *continue* trong lệnh *do while*.

1.4.4 Câu lệnh For Each

For Each là một cấu trúc lặp trong C#. For Each khác với tất cả các vòng lặp khác: For Each không có điểm bắt đầu, không có điểm kết thúc, cũng như không có bước nhảy giữa các lần lặp. For Each đơn giản là duyệt từng phần tử có bên trong mảng.

Cấu trúc:

```
foreach (string name in arr)
{
    //to do here
}
```

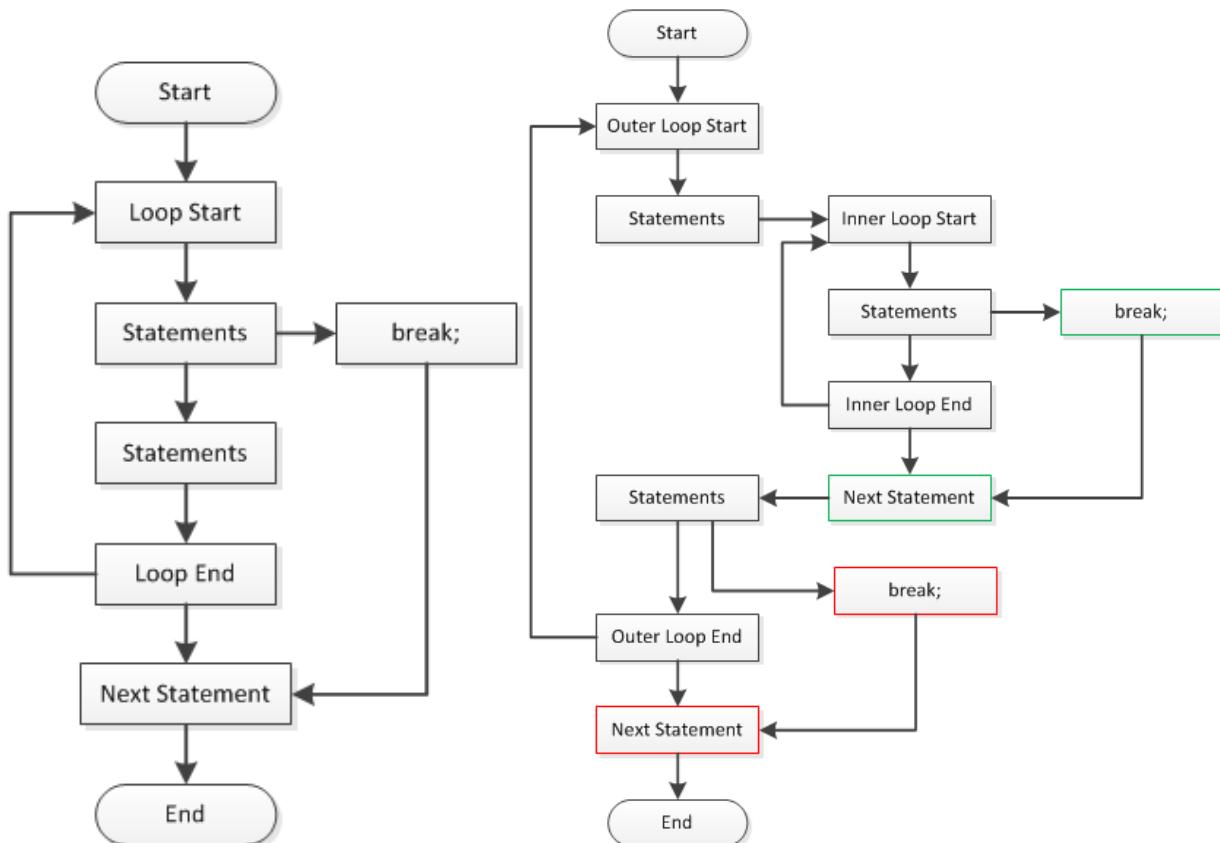
Ví dụ:

```
int[] fibarray = new int[] { 0, 1, 1, 2, 3, 5, 8, 13 };
foreach (int element in fibarray)
{
    Console.WriteLine(element);
}
Console.WriteLine();
```

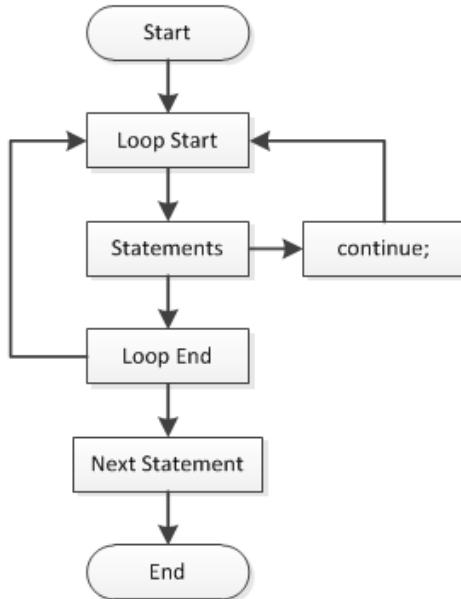
1.5 Return, Break, Continue

Trong phần này chúng ta trình bày rõ thêm về các lệnh *return*, *break* và *continue*

Break: cho phép thoát khỏi các lệnh *switch*, *for*, *while* và *do while* **gần nhất**.



Continue: bỏ qua việc thực hiện <statement(s)> và tiếp tục lặp với giá trị mới của biến lặp.



Return: lệnh này sử dụng để trả về giá trị cho hàm (trong trường hợp hàm có kiểu *void* thì không cần lệnh *return*) và kết thúc hàm đó.

Ví dụ: Hàm Main sau đây sẽ tính và in ra căn bậc 2 của một số thực.

```

static void Main(string[] args)
{
    double so;
    Console.WriteLine("Hay nhap vao mot so = ");
    if (double.TryParse(Console.ReadLine(), out so) == false)
    {
        Console.WriteLine("Gia tri ban nhap khong phai la so");
        return;
    }
    if (so < 0)
    {
        Console.WriteLine("Gia tri ban nhap < 0");
        return;
    }
    Console.WriteLine("Can bac 2 cua {0} = {1}", so, Math.Sqrt(so));
    return;
}
  
```

1.6 Xử lý ngoại lệ (Exception)

1.6.1 Ngoại lệ là gì?

Ngôn ngữ C# cũng cho phép xử lý những lỗi và các điều kiện không bình thường với những ngoại lệ. Ngoại lệ là một đối tượng đóng gói những thông tin về sự cố của một chương trình không bình thường. Chúng ta phần chia các “sự cố” thành bug, lỗi, và ngoại lệ. Một bug là một lỗi lập trình có thể được sửa chữa trước khi mã nguồn được chuyển giao. Những ngoại lệ thì không được bảo vệ và tương phản với những bug. Mặc dù một bug có thể là nguyên nhân sinh ra ngoại lệ, chúng ta cũng không dựa vào những ngoại lệ để xử lý những bug trong chương trình, tốt hơn là chúng ta nên sửa chữa những bug này.

Lỗi gây ra có thể do lỗi của người sử dụng. Ví dụ, người sử dụng nhập vào một số nhưng họ lại nhập vào ký tự chữ cái. Một lần nữa, lỗi có thể làm xuất hiện ngoại lệ, nhưng chúng ta có thể ngăn ngừa điều này bằng cách bắt giữ lỗi với mã hợp lệ. Những lỗi có thể được đoán trước và được ngăn ngừa. Thậm chí nếu chúng ta xóa tất cả những bug và dự đoán tất cả các lỗi của người dùng, chúng ta cũng có thể gặp phải những vấn đề không mong đợi, như là xuất hiện trạng thái thiếu bộ nhớ (out of memory), thiếu tài nguyên hệ thống,... Những nguyên nhân này có thể do các chương trình khác cùng hoạt động ảnh hưởng đến, chúng ta không thể ngăn ngừa các ngoại lệ này, nhưng có thể xử lý chúng để chúng không thể làm tổn hại đến chương trình.

Khi một chương trình gặp một tình huống ngoại lệ, như là thiếu bộ nhớ thì nó sẽ tạo một ngoại lệ. Khi một ngoại lệ được tạo ra, việc thực thi của các chức năng hiện hành sẽ bị treo cho đến khi nào việc xử lý ngoại lệ tương ứng được tìm thấy. Điều này có nghĩa rằng nếu chức năng hoạt động hiện hành không thực hiện việc xử lý ngoại lệ, thì chức năng này sẽ bị chấm dứt và hàm gọi sẽ nhận sự thay đổi đến việc xử lý ngoại lệ. Nếu hàm gọi này không thực hiện việc xử lý ngoại lệ, ngoại lệ sẽ được xử lý sớm bởi CLR, điều này dẫn đến chương trình của chúng ta sẽ kết thúc.

Một trình xử lý ngoại lệ là một khối lệnh chương trình được thiết kế xử lý các ngoại lệ mà chương trình phát sinh. Xử lý ngoại lệ được thực thi trong câu lệnh catch. Một cách lý tưởng thì nếu một ngoại lệ được bắt và được xử lý, thì chương trình có thể sửa chữa được vấn đề và tiếp tục thực hiện hoạt động. Thậm chí nếu chương trình không tiếp tục, bằng việc bắt giữ ngoại lệ chúng ta có cơ hội để in ra những thông điệp có ý nghĩa và kết thúc chương trình một cách rõ ràng.

Nếu đoạn chương trình của chúng ta thực hiện mà không quan tâm đến bất cứ ngoại lệ nào mà chúng ta có thể gặp (như khi giải phóng tài nguyên mà chương trình được cấp phát), chúng ta có thể đặt đoạn mã này trong khối finally, khi đó nó sẽ chắc chắn sẽ được thực hiện thậm chí ngay cả khi có một ngoại lệ xuất hiện.

1.6.2 Cấu trúc câu lệnh try... catch... finally...

Để nắm rõ các bước xử lý ngoại lệ trong C#, bạn hãy xem xét cấu trúc của lệnh *try catch finally* sau đây (và ý nghĩa của từng phần):

```
try
{
    //các lệnh có nguy cơ tạo ra ngoại lệ
}
catch(Exception ex)
{
    //các lệnh xử lý khi xảy ra ngoại lệ
}
finally
{
    //các lệnh sẽ thực hiện dù có ngoại lệ hay không
}
```

1.6.3 Phát sinh và bắt giữ ngoại lệ

Trong ngôn ngữ C#, chúng ta chỉ có thể phát sinh (*throw*) những đối tượng các kiểu dữ liệu là *System.Exception*, hay những đối tượng được dẫn xuất từ kiểu dữ liệu này. Namespace *System* của CLR chứa một số các kiểu dữ liệu xử lý ngoại lệ mà chúng ta có thể sử dụng trong chương trình. Những kiểu dữ liệu ngoại lệ này bao gồm *ArgumentNullException*, *InvalidOperationException*, *InvalidOperationException*, và *OverflowException*, cũng như nhiều lớp khác nữa.

1.6.4 Câu lệnh throw

Để phát tín hiệu một sự không bình thường trong một lớp của ngôn ngữ C#, chúng ta phát sinh một ngoại lệ bằng cách sử dụng từ khóa *throw*. Dòng lệnh sau tạo ra một thể hiện mới của *System.Exception* và sau đó *throw* nó:

throw new System.Exception();

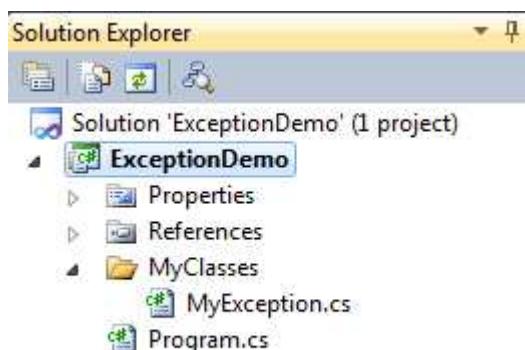
Khi phát sinh ngoại lệ thì ngay tức khắc sẽ làm ngừng việc thực thi trong khi *CLR* sẽ tìm kiếm một trình xử lý ngoại lệ. Nếu một trình xử lý ngoại lệ không được tìm thấy trong phương thức hiện thời, thì *CLR* tiếp tục tìm trong phương thức gọi cho đến khi nào tìm thấy. Nếu *CLR* trả về lớp *Main()* mà không tìm thấy bất cứ trình xử lý ngoại lệ nào, thì nó sẽ kết thúc chương trình.

Ví dụ:

```
static void Main(string[] args)
{
    int tuoi;
    try
    {
        Console.WriteLine("nhap tuoi cua ban = ");
        tuoi = int.Parse(Console.ReadLine());
        if (tuoi < 0) throw new Exception("khong hop le vi tuoi < 0");
        Console.WriteLine("tuoi = {0}", tuoi);
        //các lệnh xử lý khi tuoi>0 sẽ được viết ở đây:
    }
    catch(Exception ex)
    {
        Console.WriteLine(ex.Message);
    }
    finally
    {
        //rỗng
    }
}
```

1.6.5 Đẫn xuất một ngoại lệ

Bước 1: Tạo ứng dụng mới



Bước 2: Viết mã cho lớp MyException

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
namespace ExceptionDemo.MyClasses
{
    class MyException:Exception
    {
        public MyException(string message) : base(message)
        {
        }
    }
}
```

Bước 3: Sử dụng lớp MyException

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using ExceptionDemo.MyClasses;

namespace ExceptionDemo
{
    class Program
    {
        static void Main(string[] args)
        {
            double soTien;
            try
            {
                Console.Write("nhap so tien = ");
                bool thanhCong = double.TryParse(Console.ReadLine(), out soTien);
                if (!thanhCong || soTien < 0) throw new MyException("so tien khong hop le");
                Console.WriteLine("so tien da nhap = {0}", soTien);
            }
            catch (MyException ex)
            {
                Console.WriteLine(ex.Message);
            }
        }
    }
}
```

Bước 4: Kiểm thử chương trình với 3 trường hợp (nhập số tiền = abc, số tiền = -123 và số tiền = 456)

1.7 Các bước Debug chương trình

* Biên dịch chương trình:

- Cách 1: Gõ phím *F6*
- Cách 2: Gõ tổ hợp phím *CTRL+SHIFT+B*
- Cách 3: Vào thực đơn *Build* chọn lệnh *Build Solution*

* Debug chương trình: Gõ phím *F5*

* Debug chương trình từng bước: Gõ phím *F10*

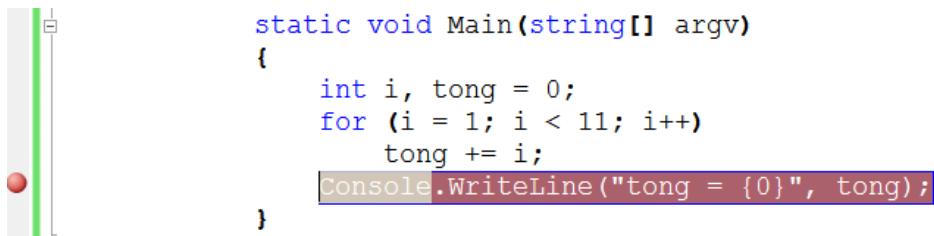
* Đặt/Xóa breakpoint: *F9*

Ví dụ:

Bước 1: Tạo dự án mới với các hàm sau:

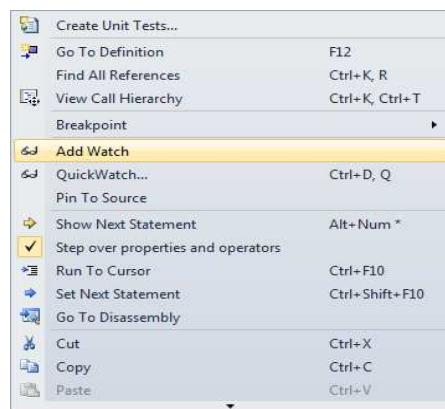
```
static void Main(string[] args)
{
    int i, tong = 0;
    for (i = 1; i < 11; i++) tong += i;
    Console.WriteLine("tong = {0}", tong);
}
```

Bước 2: Đặt breakpoint



Bước 3: Gõ F10 để debug

Bước 4: Click phải chuột tại dòng lệnh đã đặt breakpoint chọn Add Watch



Bước 5: Gõ tên của đối tượng (biến) cần kiểm tra vào cửa sổ Watch

Name	Type
i	int
tong	int
i<11	bool

Chú ý: tại bước này bạn có thể thêm, và xóa các đối tượng cần kiểm tra trong cửa sổ Watch

Bước 6: Tiếp tục gõ F10 nhiều lần để quan sát giá trị của các đối tượng cần kiểm tra.

Name	Type
i	int
tong	int
i<11	bool

Bước 7: Gõ SHIFT+F5 để dừng debug.

Bước 8: Gõ lại F9 để xóa breakpoint đã đặt

1.8 Bài tập

NHẬP XUẤT CƠ BẢN

- Viết chương trình nhập vào hai số thực dương chỉ chiều dài và chiều rộng của hình chữ nhật. Xuất ra màn hình chu vi và diện tích hình chữ nhật đó.
- Viết chương trình nhập vào độ dài cạnh của hình vuông. Xuất ra màn hình chu vi và diện tích hình vuông đó.
- Viết chương trình nhập vào bán kính của hình tròn. Xuất ra màn hình chu vi và diện tích hình tròn đó.
- Viết chương trình nhập vào họ tên (HoTen), điểm toán (Toan), điểm lý (Ly), điểm hóa (Hoa) của một sinh viên. In ra màn hình họ tên (dạng chữ HOA), điểm trung bình (DTB) lấy hai số thập phân của sinh viên theo công thức $DTB = (Toan * 2 + Ly + Hoa)/4$.
- Viết chương trình nhập vào họ tên, năm sinh một người bất kỳ. Sau đó in ra màn hình các kết quả sau: họ tên, năm sinh, tuổi hiện tại và tuổi ở năm 2020 của họ. (Sử dụng: `DateTime.Now` lấy ngày giờ hiện tại).
- Nhập vào 1 số thực x bất kỳ, xuất ra kết quả của đa thức $Y = 3x^2 + 4x - 7$.

IF ... ELSE

- Giải phương trình bậc 2 ($ax^2 + bx + c = 0$).
 - Nhập vào các hệ số a, b và c
 - Biện luận và giải phương trình
 - Vô nghiệm ($\Delta < 0$)
 - Nghiệm kép ($\Delta=0$)
 - 2 Nghiệm phân biệt ($\Delta > 0$)
 - Hướng dẫn:
 - `double delta = b*b - 4*a*c;`
 - `X1 = (-b + Math.sqrt(delta))/(2*a);`
 - `X2 = (-b - Math.sqrt(delta))/(2*a);`
- Viết chương trình nhập vào một số nguyên dương chỉ năm, cho biết năm đó có là năm nhuận hay không?

Hướng dẫn:

- Năm nhuận (là năm có 366 ngày, tháng 2 có 29 ngày) là năm chia hết cho 4, nếu năm chia hết cho 100 thì năm đó phải chia hết cho 400.

Thuật giải:

Nếu `nam % 400 == 0` thì

`//Năm nhuận`

Ngược lại, nếu (`nam % 4 == 0`) và (`nam % 100 != 0`) thì

`//Năm nhuận`

`Ngược lại`

`//Năm thường`

- Nhập vào độ dài 3 cạnh của một tam giác. Xuất ra thông báo tam giác vuông (bình phương một cạnh bằng tổng bình phương 2 cạnh còn lại), tam giác cân (hai cạnh bằng nhau), tam giác đều (ba cạnh bằng nhau), tam giác thường hoặc bộ ba số không hợp lệ.

10. Nhập vào tiền thực lãnh của tháng (năm) và số người phụ thuộc, tính thuế thu nhập cá nhân phải nộp theo luật thuế áp dụng từ tháng 7 năm 2013 như sau:

Bậc thuế	Phần thu nhập tính thuế/năm (tr. đồng)	Phần thu nhập tính thuế/tháng (tr. đồng)	Thuế suất (%)
1	Đến 60	Đến 5	5
2	Trên 60 đến 120	Trên 5 đến 10	10
3	Trên 120 đến 216	Trên 10 đến 18	15
4	Trên 216 đến 384	Trên 18 đến 32	20
5	Trên 384 đến 624	Trên 32 đến 52	25
6	Trên 624 đến 960	Trên 52 đến 80	30
7	Trên 960	Trên 80	35

Giảm trừ gia cảnh mỗi người 09 triệu/tháng và mỗi người phụ thuộc 3.6 triệu/tháng.

SWITCH ... CASE

11. Tính số tiền khách hàng phải trả khi thuê xe. Dữ liệu đầu vào là số ngày thuê xe và loại xe. Mỗi loại xe có một giá thuê riêng: loại A: 1,000,000 đ/ngày; loại B: 700,000 đ/ngày, loại C: 500,000 đ/ngày. Nếu số ngày thuê lớn hơn 10 thì giảm giá thuê xe 10%.

LẮP (FOR, WHILE,...)

12. Viết chương trình nhập số N sau đó tính các tổng sau:

- a. $S_1 = 1 + 2 + 3 + \dots + N$
- b. $S_2 = 1 + 1/2 + 1/3 + \dots + 1/N$
- c. $S_3 = 11 + 22 + 33 + \dots + NN$
- d. $S_4 = 1 * 2 * 3 * \dots * N$
- e. $S_5 = 1 + 1/2! + 1/3! + \dots + 1/N!$ (trong đó $N! = 1 * 2 * 3 * \dots * N$)
- f. $S_6 = 1/(1 * 2) + 1/(2 * 3) + 1/(3 * 4) + \dots + 1/(N * (N + 1))$

13. Nhập vào số nguyên dương $N \leq 150$. In ra giá trị bình phương các số từ 1 đến N .

14. Nhập vào một số nguyên, xuất ra số chữ số và tổng các chữ số của nó.

VD: Nhập vào 123456, xuất ra 6 và 21 ($1 + 2 + 3 + 4 + 5 + 6 = 21$)

-----oOo-----

Chương 2: **HÀM, MẢNG, TẬP HỢP**

Sau khi học xong bài này, học viên có khả năng :

- Trình bày và sử dụng được Hàm
- Trình bày và sử dụng được Mảng, Tập hợp, kiểu Cấu trúc, kiểu Liệt kê
- Mô tả được các phương pháp truyền tham số cho Hàm
- Thực hiện được cách đặt tên tham số và giá trị tùy chọn cho tham số

2.1 Hàm

2.1.1 Định nghĩa

Hàm là đoạn chương trình thực hiện trọn vẹn một công việc nhất định. Hàm chia cắt việc lớn bằng nhiều việc nhỏ, giúp chương trình sáng sủa, dễ sửa lỗi, dễ quản lý.

2.1.2 Khai báo:

```
<Kiểu dữ liệu trả về> <Tên hàm> ([Danh sách tham số])  
{  
    <Danh sách các lệnh>;  
}
```

Trong đó:

- Kiểu dữ liệu trả về: **void, float, int, double, ...**, hay kiểu người dùng định nghĩa.
- Tên hàm: do người dùng đặt thông thường đặt sao cho gợi nhớ về công dụng của hàm (thường đặt theo Pascal Case).
- Tham số: có thể có hoặc không. Tham số nếu có là các biến có giá trị, biến kết quả mà hàm sử dụng.

Ví dụ 1: Hàm output_hello dưới đây không có tham số truyền vào, trả về kiểu string.

```
static string output_hello()  
{  
    return "Hello, welcome to ASP.NET class!";  
}
```

Ví dụ 2: Hàm input khởi tạo giá trị cho mảng số nguyên, không có giá trị trả về (trả về **null**).

```
static void input(int[] a)  
{  
    //khởi tạo ngẫu nhiên các phần tử số nguyên  
    Random rd = new Random();  
    for (int i = 0; i < a.Length; i++)    a[i] = rd.Next(100);  
}
```

2.1.3 Lời gọi hàm

- Mục đích: Yêu cầu thực thi 1 tác vụ (hàm) với dữ liệu cụ thể.
- Cú pháp gọi hàm:
Tên_hàm (các dữ liệu cho từng tham số)

- Lưu ý:

- Cần truyền đủ số lượng và đúng kiểu tham số đã khai báo khi gọi hàm.
- Có thể gọi hàm lồng nhau.
- Đối với hàm có kết quả trả về, lời gọi hàm thường gán kết quả cho biến hay đặt trong biểu thức xử lý.
- Đối với hàm không có kết quả trả về, lời gọi hàm nằm riêng một dòng lệnh.

- Ví dụ:

- Gọi hàm không truyền tham số: `string s = output_hello();`
- Gọi hàm có truyền tham số: `input(a); // a là mảng một chiều`

2.1.4 Truyền tham số cho hàm

2.1.4.1 Dạng INT

Đây là kiểu truyền tham số mặc định cho hàm. Thân hàm chỉ tham khảo giá trị của tham số mà không thay đổi giá trị của tham số.

Ví dụ: Khai báo và định nghĩa hàm:

```
static int Tong(int n)
{
    int s = 0;
    for (int i = 1; i <= n; i++) s += i;
    return s;
}
```

Gọi hàm:

```
static void Main(string[] args)
{
    // ví dụ truyền tham số
    int n = 5;
    Console.WriteLine("Truoc khi goi ham n = {0}", n);
    int s = Tong(n);
    Console.WriteLine("Sau khi goi ham n = {0}", n);
}
```

Kết quả:



2.1.4.2 DẠNG OUT

- Thân hàm cấp phát/khởi tạo giá trị của tham số (chỉ được gán giá trị cho tham số).
- Ra khỏi hàm giá trị tham số thay đổi.
- Khi gọi hàm, thêm chữ `out` vào trước tên tham số.

Ví dụ:

Khai báo và định nghĩa hàm:

```
static int Tong2(out int n)
{
    int s = 0;
    for (int i = 1; i <= 5; i++)    s += i;
    n = 100;
    return s;
}
```

Gọi hàm:

```
static void Main(string[] args)
{
    //ví dụ truyền tham số
    int n = 5;
    Console.WriteLine("Truoc khi goi ham n = {0}", n);
    int s = Tong2(out n);
    Console.WriteLine("Sau khi goi ham n = {0}", n);
}
```

Kết quả:



2.1.4.3 DẠNG REF

- Ra khỏi hàm giá trị tham số thay đổi.
- Được phép thao tác đọc/sửa giá trị của tham số.
- Khi gọi hàm, thêm chữ **ref** vào trước tên tham số.

Ví dụ:

Khai báo và định nghĩa hàm:

```
static int Tong3(ref int n)
{
    int s = 0;
    for (int i = 1; i <= n; i++) s += i;
    n = 100;
    return s;
}
```

Gọi hàm:

```
static void Main(string[] args)
{
    //ví dụ truyền tham số
    int n = 5;
    Console.WriteLine("Truoc khi goi ham n = {0}", n);
    int s = Tong3(ref n);
    Console.WriteLine("Sau khi goi ham n = {0}", n);
}
```

Kết quả:

```
C:\Windows\system32\cmd.exe
Truoc khi goi ham n = 5
Sau khi goi ham n = 100
```

2.2 Mảng

2.2.1 Giới thiệu mảng trong C#

- Mảng – thành phần quan trọng trong cấu trúc dữ liệu – là tập hợp các phần tử có cùng kiểu dữ liệu được truy xuất thông qua một tên duy nhất.
- Các loại mảng: mảng một chiều (One-Dimensional Array), mảng nhiều chiều (Multidimensional Array) và mảng răng cưa (Jagged Array). Các thuộc tính và phương thức của mảng là một thể hiện của lớp **System.Array**.
- Khi chúng ta tạo một mảng có kiểu dữ liệu giá trị, mỗi thành phần sẽ chứa giá trị mặc định của kiểu dữ liệu.

Ví dụ: Với khai báo **int myIntArray = new int[5]** ; thì:

- Mỗi thành phần của mảng được thiết lập giá trị là 0 (giá trị mặc định của số nguyên).
- Những kiểu tham chiếu trong một mảng không được khởi tạo giá trị mặc định, chúng được khởi tạo giá trị **null**.

2.2.2 Mảng 1 chiều

- Chỉ số mảng bắt đầu từ 0, tức là phần tử đầu tiên ở vị trí 0, phần tử cuối cùng ở vị trí <số_phần_tử> - 1.
- Mảng có thể được khai báo với kích thước cố định (bị giới hạn bởi số lượng phần tử) hoặc động (có thể thay đổi kích thước mảng tùy tình hình thực tế).
- Mảng là kiểu đối tượng (object), do đó sau khi khai báo mảng cần tạo thể hiện của mảng bằng từ khóa **new**.

```
kiểu dữ liệu>[] <tên mảng> ;
```

Ví dụ:

```
double[] doubleArray = new double[5];
char[] charArray = new char[5];
bool[] boolArray = new bool[2];
string[] stringArray = new string[10];
```

- Khởi tạo giá trị cho mảng:
 - Có thể khởi tạo ngay khi khai báo và tạo thể hiện

```
int[] staticIntArray = new int[3] {1, 3, 5};
string[] strArray = new string[] {"Mahesh Chand", "Mike Gold", "Raj Beniwal", "Praveen Kumar", "Dinesh Beniwal"};
```

- Hoặc gán trực tiếp từng giá trị cho mảng:

```
int[] staticIntArray = new int[2];
staticIntArray[0] = 1;
staticIntArray[1] = 3;
```

- Truy xuất mảng: Sử dụng toán tử [].

```
staticIntArray[0] = 1;
```

2.2.2.1 Một số thuộc tính và phương thức thường dùng của lớp System.Array

Thành viên	Mô tả
Sort()	Phương thức sắp xếp giá trị tăng dần trong mảng một chiều
Reverse()	Phương thức sắp xếp giá trị giảm dần trong mảng một chiều
Length	Thuộc tính chiều dài của mảng
SetValue()	Phương thức thiết lập giá trị cho một thành phần xác định trong mảng
Rank	Thuộc tính trả về số chiều của mảng

2.2.2.2 Ví dụ:

Các thao tác với mảng một chiều các số nguyên tối đa 20 phần tử.

```
static void XuatMang(int [] a)
{
    for (int i = 0; i < a.Length; i++)
        Console.Write(a[i] + " ");
    Console.WriteLine();
}
static void Main(string[] args)
{
    //ví dụ mảng 1 chiều
    //khai báo mảng 1 chiều tối đa 20 phần tử
    int[] a = new int[20];

    //khởi tạo giá trị ngẫu nhiên cho mảng
    Random rd = new Random();
    for (int i = 0; i < a.Length; i++)    a[i] = rd.Next(100);

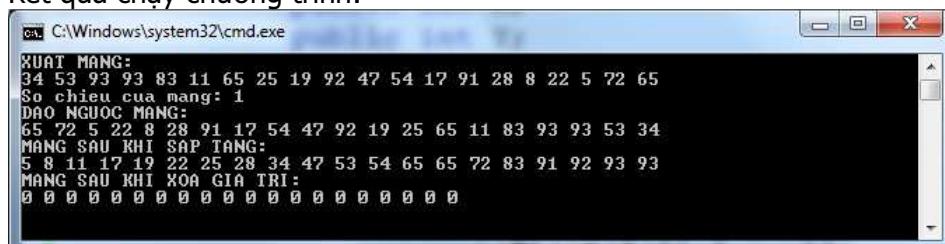
    Console.WriteLine("XUAT MANG:");
    XuatMang(a);
    Console.WriteLine("So chieu cua mang: {0}", a.Rank);

    Console.WriteLine("DAO NGUOC MANG:");
    Array.Reverse(a);
    XuatMang(a);

    Console.WriteLine("MANG SAU KHI SAP TANG:");
    Array.Sort(a);
    XuatMang(a);

    //xóa mảng
    Console.WriteLine("MANG SAU KHI XOA GIA TRI:");
    Array.Clear(a, 0, a.Length);
    XuatMang(a);
}
```

Kết quả chạy chương trình:



2.2.3 Mảng nhiều chiều

- Mảng nhiều chiều được biết đến như mảng hình chữ nhật với số chiều nhiều hơn 1, thường được gọi là ma trận (matrix). Số lượng phần tử là bằng nhau ở mỗi chiều.
- Khai báo mảng:

```
<kiểu dữ liệu>[ , ] <tên mảng>
```

Ví dụ:

```
int[ , ] myRectangularArray ;
```

- Khởi tạo thành phần của mảng

```
int[, ] myRectangularArray = new int[sodong , socot];
int[, ] numbers = new int[3, 2] { { 1, 2 }, { 3, 4 }, { 5, 6 } };
string[, ] names = new string[2, 2] { { "Rosy", "Amy" }, { "Peter", "Albert" } };
```

hoặc:

```
int[, ] numbers = new int[, ] { { 1, 2 }, { 3, 4 }, { 5, 6 } };
string[, ] names = new string[, ] { { "Rosy", "Amy" }, { "Peter", "Albert" } };
```

hoặc là:

```
int[, ] numbers = { { 1, 2 }, { 3, 4 }, { 5, 6 } };
string[, ] names = { { "Rosy", "Amy" }, { "Peter", "Albert" } };
```

hoặc là:

```
int[, ] numbers = new int[3, 2];
numbers[0, 0] = 1;
numbers[1, 0] = 2;
numbers[2, 0] = 3;
numbers[0, 1] = 4;
numbers[1, 1] = 5;
numbers[2, 1] = 6;
```

- Duyệt mảng 2 chiều

```
for (int i = 0; i < sodong; i++)
{
    for (int j = 0; j < socot; j++)
    {
        Xử lý numbers[i,j];
    }
}
```

2.2.4 Mảng răng cưa

- Là mảng nhiều chiều nhưng số lượng phần tử ở mỗi chiều khác nhau.
- Khai báo mảng:

```
<kiểu dữ liệu>[ ][ ] <tên mảng>;
```

Ví dụ: Khai báo mảng răng cưa 2 chiều có 3 dòng.

```
int[][] intJaggedArray = new int[3][];
```

- Khởi tạo thành phần của mảng.

Ví dụ: Mảng răng cửa 2 chiều có 3 dòng, ứng với mỗi dòng ta khởi tạo số lượng cột khác nhau.

```
intJaggedArray[0] = new int[2];
intJaggedArray[1] = new int[4];
intJaggedArray[2] = new int[6];
```

hoặc:

```
intJaggedArray[0] = new int[2]{2, 12};
intJaggedArray[1] = new int[4]{4, 14, 24, 34};
intJaggedArray[2] = new int[6] {6, 16, 26, 36, 46, 56};
```

- Duyệt mảng răng cửa:

Ví dụ: duyệt mảng răng cửa 3 dòng đã khai báo và khởi tạo ở trên.

```
for (int i = 0; i < intJaggedArray.Length; i++)
{
    System.Console.Write("Element ({0}): ", i);
    for (int j = 0; j < intJaggedArray[i].Length; j++)
    {
        System.Console.Write("{0}{1}", intJaggedArray[i][j], j ==
            (intJaggedArray[i].Length - 1) ? "" : " ");
    }
    System.Console.WriteLine();
}
```

Kết quả chạy chương trình:

```
C:\Windows\system32\cmd.exe
Element <0>: 2 12
Element <1>: 4 14 24 34
Element <2>: 6 16 26 36 46 56
```

2.3 Tập hợp - Collections

- Là cấu trúc dữ liệu dùng để lưu trữ danh sách các phần tử có kiểu dữ liệu khác nhau. Số lượng phần tử không hạn chế.
- Các lớp này nằm trong namespace `System.Collections` hoặc `System.Collections.Generic` và thường có chung một giao diện.

2.3.1 List

- Phải chỉ định rõ kiểu dữ liệu của từng phần tử.
- Khai báo:

```
List<kiểu_dữ_liệu> <tên_biến> = new List<kiểu_dữ_liệu>();
```

Ví dụ:

```
using System;
using System.Collections.Generic;

class Program
{
```

```

static void Main()
{
    // Use the List type.
    List<string> list = new List<string>();
    list.Add("cat");
    list.Add("dog");

    foreach (string element in list)
    {
        Console.WriteLine(element);
    }
}

```

Kết quả:

cat
dog

- Truy xuất phần tử thông qua toán tử []. Ví dụ: list[1];
- Một số phương thức và thuộc tính thường dùng:

Thuộc tính/Phương thức	Ý nghĩa
Capacity	Trả về tổng số phần tử tối đa
Count	Trả về số phần tử thật sự
Add(obj)	Thêm một phần tử kiểu obj
AddRange(arr_obj)	Thêm một mảng các phần tử
Clear()	Xóa tất cả phần tử
Contains(T)	Xác định danh sách có chứa phần tử T hay không?
Exist(lamda_expression)	Kiểm tra có phần tử nào thỏa mãn lamda expression hay không? (true hoặc false)
Find(lamda_expression)	Kiểm tra có phần tử nào thỏa mãn lamda expression hay không? (true hoặc false)
Insert(index, T)	Chèn phần tử T vào vị trí index
InsertRange(index, T_collection)	Chèn vào vị trí index danh sách các phần tử
RemoveAll(lamda_expression)	Xóa tất cả phần tử thỏa mãn lamda expression
RemoveAt(index)	Xóa phần tử tại vị trí index
Sort()	Sắp xếp các phần tử

2.3.2 ArrayList

- Không chỉ định rõ kiểu dữ liệu của từng phần tử. Do đó, các phần tử có thể có kiểu dữ liệu khác nhau.
- Khai báo:

```
ArrayList <tên biến> = new ArrayList();
```

Ví dụ:

```

using System;
using System.Collections.Generic;

class Program
{
    static void Main()
    {
        // Use the ArrayList type.
        ArrayList alist = new ArrayList();
        alist.Add("cat");
    }
}

```

```

alist.Add(1999);
alist.Add("dog");

for (int i=0; i < alist.Count; i++)
{
    Console.WriteLine(alist[i].ToString());
}
}
    
```

Kết quả:

cat

1999

dog

- Truy xuất phần tử thông qua toán tử [], chú ý ép kiểu nếu cần.

Ví dụ: `(int) alist[1];`

- Một số phương thức và thuộc tính thường dùng:

Thuộc tính/Phương thức	Ý nghĩa
Capacity	Trả về tổng số phần tử tối đa có thể chứa
Count	Trả về số phần tử thật sự
Add()	Thêm một phần tử kiểu obj vào cuối ArrayList
AddRange(arr_obj)	Thêm một mảng các phần tử vào cuối ArrayList
Clear()	Xóa tất cả phần tử
Contains(T)	Xác định phần tử T có nằm trong ArrayList hay không?
Exist(lamda_expression)	Kiểm tra có phần tử nào thỏa mãn lamda expression hay không? (true hoặc false)
Find(lamda_expression)	Kiểm tra có phần tử nào thỏa mãn lamda expression hay không? (true hoặc false)
Insert(index, T)	Chèn phần tử T vào vị trí index
InsertRange(index, T_collection)	Chèn vào vị trí index danh sách các phần tử
Remove()	Xóa phần tử đầu tiên trong ArrayList
RemoveAll(lamda_expression)	Xóa tất cả phần tử thỏa mãn lamda expression
RemoveAt(index)	Xóa phần tử tại vị trí index
Reverse()	Đảo ngược thứ tự các phần tử
Sort()	Sắp xếp các phần tử

2.3.3 Hashtable

- Hashtable là kiểu từ điển, mỗi phần tử bao gồm 1 cặp [key-value]. Hashtable không cần khai báo kiểu dữ liệu cho key, value.
- Hashtable dùng tối ưu cho việc truy xuất nhanh. Các cặp key là không trùng nhau.
- Khai báo:

`Hashtable <tên biến> = new Hashtable();`

Ví dụ:

```

using System;
using System.Collections.Generic;
    
```

```
class Program
{
    static void Main()
    {
        Hashtable ht = new Hashtable();
        ht.Add("pet", "cat");
        ht.Add(1, 1999);
        ht.Add("nick", "mylovepet");

        for (DictionaryEntry de in ht)
            Console.WriteLine("{0} --> {1}", de.Key, de.Value);
    }
}
```

Kết quả:

```
cat
dog
```

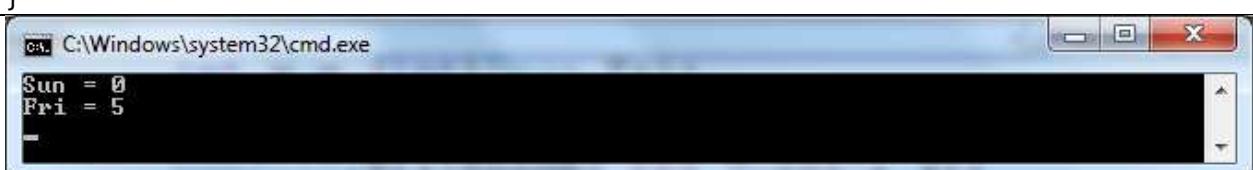
- Truy xuất phần tử thông qua toán tử [], chú ý ép kiểu nếu cần.
Ví dụ: `(int) ht[1];`
- Một số phương thức và thuộc tính thường dùng:

Thuộc tính/Phương thức	Ý nghĩa
Count	Trả về số phần tử có trong Hashtable
Keys	Tập hợp các khóa
Values	Tập hợp các giá trị ứng với khóa
Add(key, value)	Thêm một phần tử với key, value xác định
Clear()	Xóa tất cả phần tử bao gồm cả key, value
Contains(T)	Xác định phần tử T có nằm trong Hashtable hay không?
ContainsKey(k)	Xác định có phần tử nào trong Hashtable có khóa k
ContainsValue(v)	Xác định có phần tử nào trong Hashtable có giá trị là v
Remove(key)	Xóa phần tử có khóa key

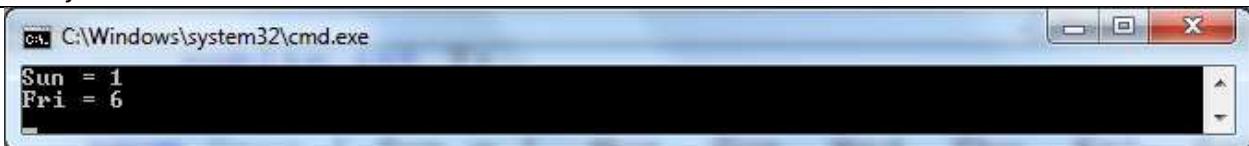
2.4 Kiểu liệt kê - Enumeration

Kiểu tập hợp là kiểu dữ liệu mà giá trị của nó lấy trong một tập hợp các hằng số cho trước. Kiểu dữ liệu mặc định của các phần tử liệt kê là kiểu `int` (số nguyên). Mặc định, phần tử đầu tiên trong danh sách liệt kê là 0, các phần tử tiếp theo được tăng lên 1.

```
enum Days {Sun, Mon, Tue, Wed, Thu, Fri, Sat};
enum Days {Sun = 1, Mon, Tue, Wed, Thu, Fri, Sat};
enum Days { Sun, Mon, Tue, Wed, Thu, Fri, Sat };
static void Main(string[] args)
{
    int x = (int)Days.Sun;
    int y = (int)Days.Fri;
    Console.WriteLine("Sun = {0}", x);
    Console.WriteLine("Fri = {0}", y);
}
```



```
enum Days { Sun = 1, Mon, Tue, Wed, Thu, Fri, Sat };
static void Main(string[] args)
{
    int x = (int)Days.Sun;
    int y = (int)Days.Fri;
    Console.WriteLine("Sun = {0}", x);
    Console.WriteLine("Fri = {0}", y);
}
```



2.5 Kiểu cấu trúc - Struct

Là kiểu giá trị được dùng để đóng gói nhóm nhỏ các biến có liên quan với nhau.

Ví dụ:

```
class Program
{
    struct Point {
        public int X;
        public int Y;
    }
    static void Main(string[] args)
    {
        Point p;
        p.X = 10;
        p.Y = 20;
        Console.WriteLine("Point ({0},{1})", p.X, p.Y);
    }
}
```

2.6 Optional & Named Parameters

2.6.1 Optional Parameter – Tham số mặc định

- Tham số mặc định dùng cho trường hợp không truyền giá trị của tham số.

Ví dụ: Hàm Optional() bên dưới có tham số truyền vào kiểu string, nếu không truyền giá trị tham số thì sẽ lấy giá trị mặc định (Test).

```
static void Main(string[] args)
{
    Optional();
    Optional("Another value");
}
static void Optional(string Value="Test")
{
    Console.WriteLine(Value);
}
```

Kết quả:

Another value

Test

- Tham số mặc định phải truyền từ phải sang trái, liên tục nhau.

Ví dụ:

```
static void Optional2(string Value1="Test", string Value2) {
    Console.WriteLine(Value1 + Value2);
}
```

```
static void Optional3(string Value1="Test", string Value2,
string Value3="OK") {
    Console.WriteLine(Value1 + Value2 + Value3);
}
```

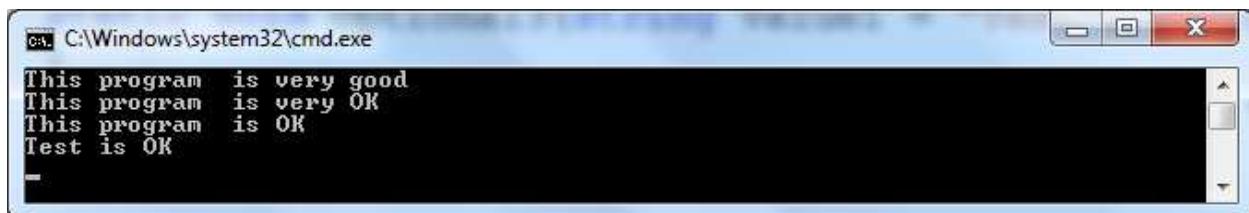
Hàm Optional2, Optional3 bắt buộc tham số Value2 phải là tham số mặc định. Ta sửa tham số Value2 của Optional3 như sau:

```
static void Optional3(string Value1 = "Test", string Value2 = "is",
string Value3="OK") {
    Console.WriteLine(Value1 + Value2 + Value3);
}
```

Thực hiện chương trình với dữ liệu sau:

```
static void Main(string[] args)
{
    Optional3("This program ","is very", "good");
    Optional3("This program ","is very");
    Optional3("This program ");
    Optional3();
}
```

Kết quả nhận được:



2.6.2 Named Parameter – Tham số được đặt tên

Sử dụng tham số được đặt tên làm chương trình dễ đọc, dễ trình bày. Bằng cách sử dụng tên tham số chính thức, chúng ta có thể đổi thứ tự các tham số thực tế.

Ví dụ sau đây sử dụng 4 cách gọi hàm khác nhau sử dụng tham số được đặt tên.

```
class Program
{
    static void Main()
    {
        // Call the Test method several times in different ways.
        Test(name: "Perl", size: 5); //Cach 1
        Test(name: "Dot", size: -1); //Cach 2
        Test(6, "Net"); //Cach 3
        Test(7, name: "Google"); //Cach 4
    }
}
```

```

static void Test(int size, string name)
{
    Console.WriteLine("Size = {0}, Name = {1}", size, name);
}
    
```

Output

Size = 5, Name = Perl

Size = -1, Name = Dot

Size = 6, Name = Net

Size = 7, Name = Google

Cách gọi 1,2 hoán đổi vị trí các tham số.

2.7 Bài tập

HÀM

- Viết chương trình thực hiện chức năng nhập vào 1 số nguyên từ n bàn phím ($n > 20$), sau đó tính tổng x (x được nhập từ bàn phím) các số chẵn đầu tiên từ 1 -> n. Nếu người nhập $n < 20$ thì thông báo nhập lại.
 - Viết hàm nhập và kiểm tra số nguyên n.
 - Viết hàm tính tổng các số chẵn.

HƯỚNG DẪN - GỢI Ý

Bước 1: Thiết kế hàm nhập và kiểm tra $n > 20$. *Chú ý kiểu truyền tham số cho hàm.*

```

static void nhap(out int n) {
    int x;
    while(true){
        Console.WriteLine("Nhập vào số nguyên");
        x = int.Parse(Console.ReadLine());
        if (x > 20) break;
        Console.WriteLine("Vui lòng nhập n > 20.");
    }
    n = x;
}
    
```

Bước 2: Thiết kế hàm tính tổng các số chẵn từ 1 đến n

```

static int tongsochan(int n)
{
    int s = 0;
    for (int i = 1; i <= n; i++)
        if (i % 2 == 0) s += i;
    return s;
}
    
```

Bước 3: Thực hiện gọi hàm

```

static void Main(string[] args){
    int n; nhap(out n);
    Console.WriteLine("Tổng các số chẵn từ 1 đến {0} là: {1}", n,
                      tongsochan(n));
}
    
```

Kết quả màn hình chạy với dữ liệu nhập vào lần lượt là 1, 19, 21.

```
C:\Windows\system32\cmd.exe
Nhap so nguyen: 1
Uui long nhap n > 20.
Nhap so nguyen: 19
Uui long nhap n > 20.
Nhap so nguyen: 21
Tong cac so chan tu 1 den 21 la: 110
Press any key to continue . . .
```

Bước 4: Tinh chỉnh hàm tính tổng

```
static int tongsochan2(int n)
{
    int s = 0;
    for (int i = 2; i <= n; i+=2)
        s += i;
    return s;
}
```

2. Viết chương trình đếm xem có bao nhiêu nguyên tố từ x → y với x, y là 2 số nguyên được nhập từ bàn phím.

- Viết hàm nhập số nguyên.
- Viết hàm kiểm tra số nguyên tố
- Viết hàm đếm số nguyên tố

HƯỚNG DẪN - GÓI Ý

Gợi ý: Thuật toán kiểm tra số nguyên tố:

Số nguyên tố là số chỉ có 2 ước số: 1 và chính nó (chia hết cho 1 và chính nó). Nếu n không chia hết cho mọi số nguyên i trong khoảng từ 2 đến căn bậc 2 của n thì n là số nguyên tố.
 Ví dụ: n = 9 , canbac2(9) = 3.

i = 2 và i = 3 , 9 không chia hết cho 2 nhưng chia hết cho 3 => 9 không phải là số nguyên tố (vì số nguyên tố là số chỉ chia hết cho 1 và chính nó).

Sử dụng hàm Math.Sqrt để tính căn bậc 2

Gợi ý: Hàm kiểm tra số nguyên tố

```
//Hàm kiểm tra số nguyên tố
static bool kiemtraSNT(int n)
{
    bool flag = (n > 1) ? true : false;
    for(int i = 2; i <= Math.Sqrt(n); i++)
        if (n % i == 0)
        {
            flag = false;
            break; //thoát vòng lặp
        }
    return flag;
}
```

MÃNG

3. Khai báo mảng một chiều các số nguyên tối đa 100 phần tử. Viết chương trình:

- Viết hàm nhập vào giá trị cho các phần tử trong mảng.
- Viết hàm xuất mảng 1 chiều các số nguyên.
- Viết hàm tính tổng các phần tử trong mảng.
- Viết hàm tìm số lớn nhất, số nhỏ nhất trong mảng 1 chiều.

- Viết hàm đếm số lượng số nguyên dương chẵn có trong mảng.
- Viết hàm xuất giá trị tổng, trung bình cộng các giá trị của các phần tử trong mảng.
- **Viết hàm đếm số lần xuất hiện của phần tử x (tham số) trong mảng a.**
- **Viết hàm trả về mảng chứa các mảng phần tử chẵn trong mảng a.**
- Viết hàm main thực hiện:
 - o Khai báo mảng
 - o Gọi hàm nhập, xuất mảng
 - o Xuất kết quả tổng
 - o Xuất số lớn nhất, số nhỏ nhất
 - o Xuất số lượng số nguyên dương chẵn
 - o Xuất giá trị tổng, trung bình cộng

Code minh họa

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Mang
{
    class Program
    {
        //Hàm nhập mảng số nguyên a
        public static void Nhap(int[] a)
        {
            Console.WriteLine("NHAP MANG");
            for (int i = 0; i < a.Length; i++)
            {
                Console.Write("Nhập phần tử thu {0}: ", i);
                a[i] = int.Parse(Console.ReadLine());
            }
        }
        //Hàm xuất mảng số nguyên a có n phần tử
        public static void Xuat(int[] a)
        {
            Console.WriteLine("XUAT MANG");
            for (int i = 0; i < a.Length; i++)
            {
                Console.Write(a[i] + " ");
            }
            Console.WriteLine();
        }

        static void Main(string[] args)
        {
            int n;//số phần tử của mảng
            Console.Write("Nhập số phần tử của mảng: ");
            n = int.Parse(Console.ReadLine());

            //Khai báo mảng số nguyên a có n phần tử
            int[] a = new int[n];

            //Gọi hàm nhập mảng
            Nhap(a);

            //gọi hàm xuất mảng
        }
    }
}

```

```

Xuat(a);

//gọi hàm tính tổng các phần tử trong mảng
Console.WriteLine("Tổng các phần tử trong mảng là: {0}", tinh tong(a));

//gọi hàm tìm phần tử lớn nhất
Console.WriteLine("Phần tử lớn nhất của mảng là: {0}", a[timMax(a)]);

//gọi hàm đếm số phần tử nguyên dương chẵn
Console.WriteLine("Số lượng phần tử nguyên dương chẵn là: {0}", dem(a));
}

//Hàm tính tổng các phần tử có trong mảng
public static int tinh tong(int[] a)
{
    int s = 0;
    for (int i = 0; i < a.Length; i++)
        s += a[i];
    return s;
}

//Hàm tìm vị trí phần tử lớn nhất trong mảng
public static int timMax(int[] a)
{
    int vitri = 0;
    for (int i = 1; i < a.Length; i++)
        if (a[i] > a[vitri])
            vitri = i;
    return vitri;
}

//Hàm đếm số lượng số nguyên dương chẵn có trong mảng
public static int dem(int[] a)
{
    int dem = 0;
    for (int i = 1; i < a.Length; i++)
        if ((a[i] > 0) && (a[i] % 2 == 0))
            dem++;
    return dem;
}
}

```

Lưu ý: Một số hàm học viên tự viết, giảng viên hướng dẫn nếu có thời gian.

4. Khai báo 1 mảng nguyên 2 chiều 4 dòng, 5 cột. Viết chương trình:

 - Nhập giá trị cho các phần tử trong mảng (giá trị = số thứ tự dòng + số thứ tự cột).
 - In giá trị các phần tử trong mảng.
 - In giá trị lớn nhất, giá trị nhỏ nhất của các phần tử trong mảng.
 - In tổng số các giá trị, trung bình cộng các giá trị của các phần tử trong mảng.
 - Viết hàm thực hiện sắp xếp các phần tử trong mảng tăng dần (từ trái sang phải, từ trên xuống dưới).
 - Viết hàm tìm phần tử x có trong mảng hay không ? nếu tìm thấy xuất thông báo.
 - Viết hàm thực hiện tính tổng các phần tử trên đường chéo chính của mảng a (những phần tử có vị trí dòng = vị trí cột).
 - Viết hàm thực hiện sắp xếp các phần tử trên dòng chẵn tăng dần và dòng lẻ giảm dần.

Code minh họa:

```
namespace Mang2C
{
    class Program
    {
        //Hàm nhập mảng 2 chiều số nguyên a có m dòng, n cột
        public static void Nhap(int[,] a, int m, int n)
        {
            Console.WriteLine("NHAP MANG");
            for (int i = 0; i < m; i++)
            {
                for (int j = 0; j < n; j++)
                {
                    Console.Write("Nhập phần tử thu [{0}][{1}]: ", i, j);
                    a[i,j] = int.Parse(Console.ReadLine());
                }
            }
        }
        //Hàm xuất mảng 2 chiều số nguyên a có m dòng, n cột
        public static void Xuat(int[,] a, int m, int n)
        {
            Console.WriteLine("XUAT MANG");
            for (int i = 0; i < m; i++)
            {
                for (int j = 0; j < n; j++)
                    Console.Write("a[{0}][{1}] = {2}", i, j, a[i,j]);
                Console.WriteLine(); //xuống hàng
            }
            Console.WriteLine();
        }
        static void Main(string[] args)
        {
            int m = 4, n = 5;

            //Khai báo mảng số nguyên có 4 hàng 5 cột
            int[,] a = new int[4,5];

            //Gọi hàm nhập mảng
            Nhap(a, m, n);

            //gọi hàm xuất mảng
            Xuat(a, m, n);

            //gọi hàm tính tổng các phần tử trong mảng
            Console.WriteLine("Tổng các phần tử trong mảng là: {0}",
            tinh tong(a, m, n));

            //gọi hàm tìm phần tử lớn nhất
            Console.WriteLine("Phần tử lớn nhất của mảng là: {0}", timMax(a,
            m, n));
        }

        //Hàm tính tổng các phần tử có trong mảng
        public static int tinh tong(int[,] a, int m, int n)
```

```

    {
        int s = 0;
        for (int i = 0; i < m; i++)
            for(int j = 0; j < n; j++) s += a[i,j];
        return s;
    }

    //Hàm tìm giá trị lớn nhất
    public static int timMax(int[,] a, int m, int n)
    {
        int max = a[0,0];
        for (int i = 0; i < m; i++)
            for (int j = 0; j < n; j++)
                if (a[i, j] > max) max = a[i, j];
        return max;
    }
}

```

Lưu ý: Một số hàm học viên tự viết, giảng viên hướng dẫn nếu có thời gian.

5. Viết 1 ứng dụng Console thực hiện các chức năng sau với ArrayList hoặc Hashtable
 - Nhập các phần tử kiểu chuỗi từ bàn phím, thêm vào ArrayList khi nào người dùng nhập vào chuỗi “stop” thì ngừng.
 - Xuất các phần tử ra màn hình.
 - Tìm phần tử, trả về **true** nếu tìm thấy, ngược lại trả về **false**
 - Xóa một phần tử.
 - Thêm 1 phần tử vào vị trí index bất kỳ với index nhập từ bàn phím.

HƯỚNG DẪN:

Trường hợp sử dụng Hashtable

```

Hashtable ht = new Hashtable();
int i = 1;
string s;
do
{
    s = Console.ReadLine();
    if (s == "stop") break;
    ht.Add(i, s);
    i++;
} while (true);
XuatPhanTu(ht); //goi ham xuat cac phan tu trong HashTable
Console.WriteLine("nhap phan tu can tim:");
string x = Console.ReadLine();
if (ht.ContainsValue(x) == true)
    Console.WriteLine("tim thay");
else
    Console.WriteLine("Khong tim thay");

```

Trường hợp sử dụng ArrayList

```

//Khai bao va su dung ArrayList
ArrayList al = new ArrayList();
string n;
do
{

```

```

n = Console.ReadLine();
if (n == "stop") break;
al.Add(n);
} while (true);
XuatMang(al);
Console.WriteLine("So phan tu hien co:{0}", al.Count);
Console.WriteLine("\n Nhập phan tu can tim:");
string x = Console.ReadLine();
int vitri = al.BinarySearch(x);
if (vitri >= 0)
{
    Console.WriteLine("\nTim thay");
    Console.WriteLine("Xoa phan tu(y/n)");
    string ch = Console.ReadLine();
    if (ch.ToUpper() == "Y")
        al.RemoveAt(vitri);
    Console.WriteLine("Mang sau khi xoa:");
    XuatMang(al);
}
else
    Console.WriteLine("\nKhong tim thay");
//Dao tat ca cac phan tu:
al.Reverse();
XuatMang(al);
//Dao mot phan cua mang: al.reverse(vitri,soluong);

```

6. Thông tin một mặt hàng bao gồm: Mã mặt hàng (MaMH - số nguyên), tên mặt hàng (TenMH - chuỗi), số lượng (SoLuong - số nguyên), đơn giá (DonGia - số thực).
 - Khai báo cấu trúc mặt hàng nói trên, bao gồm cả hàm tạo và hàm tính thành tiền.
 - Viết hàm thêm một mặt hàng vào trong danh sách.
 - Viết hàm tìm một mặt hàng dựa vào mã mặt hàng, trả về kiểu bool (có hay không).
 - Viết hàm xuất danh sách các mặt hàng.
 - viết hàm xóa mặt hàng dựa vào mã mặt hàng.
 - Viết hàm main thực hiện các chức năng sau:
 - o nhập vào danh sách các mặt hàng. Nhập xong mỗi mặt hàng hỏi người dùng có tiếp tục nhập hay không?
 - o xuất danh sách các mặt hàng.
 - o nhập vào mã mặt hàng, tiến hành tìm kiếm mặt hàng nói trên. Nếu tìm thấy, xóa mặt hàng và xuất danh sách các mặt hàng (sau khi xóa).

-----oOo-----

Chương 3: **ĐỐI TƯỢNG (OBJECT) VÀ LỚP (CLASS)**

Sau khi học xong bài này, học viên có khả năng :

- Xây dựng được Lớp và thành phần trong lớp
- Trình bày và tạo được đối tượng sử dụng Constructors & Object Initialize
- Khai báo và sử dụng được phương thức mở rộng
- Khai báo và sử dụng được kiểu Anonymous Type

3.1 Khái niệm lớp (Class) & Đối tượng (Object)

3.1.1 Lớp

Lớp là một khái niệm mô tả cho những thực thể có chung tính chất và hành vi, là một khuôn mẫu cho các đối tượng.

Lớp đối tượng là một cấu trúc dữ liệu linh hoạt có thể lưu trữ dữ liệu và thực thi hành động bao gồm hai thành phần sau :

- Thành phần thuộc tính (dữ liệu) bao gồm các thông tin liên quan đến lớp.
- Thành phần phương thức (hành động) bao gồm các hành động liên quan đến lớp đó.

Mỗi lớp đối tượng có thể có nhiều thuộc tính và nhiều phương thức.

Khai báo lớp bằng cách sử dụng từ khoá **class**. Cú pháp đầy đủ như sau:

[Thuộc tính] [Bổ sung truy cập] class <Tên lớp> [: Lớp cơ sở]

{

```
// Các thuộc tính
<Thuộc tính>
// Các phương thức
<Phương thức>
```

}

Ví dụ: Khai báo lớp Diem biểu diễn thông tin một điểm trong mặt phẳng Oxy.

```
class Diem
{
    // Các thuộc tính
    private int x; // x viet thuong
    private int y; // y viet thuong

    // Các phương thức
    public override string ToString() // Xuat
    {
        return "(" + x + ", " + y + ")";
    }
}
```

3.1.2 Đối tượng

Đối tượng là những đại diện cho lớp, mọi đối tượng đều có chung tính chất và hành vi mà lớp định nghĩa.

Ví dụ: Khai báo đối tượng dinhA – là thể hiện của lớp Diem đã định nghĩa ở trên.

```
Diem dinhA = new Diem();
```

Sau khi khai báo, đối tượng dinhA có đầy đủ các thuộc tính và phương thức của lớp Diem.

3.2 Thuộc tính (Field) & Phương thức (Method)

3.2.1 Thuộc tính (Field)

- Fields là các phần tử dùng để thể hiện các biến trong lớp.
- Fields là những thông tin có thể thay đổi được.

3.2.2 Phương thức (Method)

- Phương thức (method) chính là các hàm (function) được tạo trong lớp (class).
- Tên của phương thức thường được đặt theo tên của hành động.

3.2.3 Bảng tầm vực thuộc tính truy cập

Thuộc tính	Giới hạn truy cập
public	Không hạn chế. Những thành viên được đánh dấu public có thể được dùng bất kỳ các phương thức của lớp, bao gồm cả những lớp khác.
private	Thành viên trong lớp được đánh dấu private chỉ được dùng các phương thức của lớp này mà thôi.
protected	Thành viên trong lớp được đánh dấu protected chỉ được dùng các phương thức của lớp này; và các phương thức của lớp dẫn xuất từ lớp này.
internal	Thành viên trong lớp được đánh dấu là internal được dùng các phương thức của bất kỳ lớp nào cùng khối hợp ngữ với lớp này.
protected internal	Thành viên trong lớp được đánh dấu là protected internal được dùng các phương thức của lớp này; các phương thức của lớp dẫn xuất từ lớp này; và các phương thức của bất kỳ lớp nào trong cùng khối hợp ngữ với lớp này.

- private** thì thuộc tính/phương thức đó chỉ được sử dụng trực tiếp bên trong lớp đó.
- public** thì thuộc tính/phương thức đó có thể được sử dụng trực tiếp bên trong lớp lẫn bên ngoài lớp.

3.3 Constructors & Object Initialize

3.3.1 Constructor

- Constructor là phương thức đặc biệt của lớp, được gọi thực hiện khi lớp được tạo ra.
- Constructors có tên giống như tên của Class.
- Constructors không có giá trị trả về.
- Ví dụ: Một số hàm constructor cho lớp Diem.

```
// Các phương thức khởi tạo
public Diem()
{
    x = 0;
    y = 0;
}
```

```

public Diem(int xx, int yy)
{
    x = xx;
    y = yy;
}

public Diem(Diem p)
{
    X = p.X;
    Y = p.Y;
}

```

Sử dụng hàm constructor:

```

Diem dinhA = new Diem();
Diem dinhB = new Diem(5, 5);
Diem dinhC = new Diem(dinhB);

```

3.3.2 Object Initialize

Khởi tạo đối tượng kiểu object Initialize gồm có 3 cách sau:

- Cách thông thường: Khởi tạo các đối tượng sau đó gán các thuộc tính.

Ví dụ:

```

Diem dinhA = new Diem();
dinhA.X = 10;
dinhA.Y = 10;

```

- Gán các thuộc tính ngay khi khởi tạo đối tượng:

Ví dụ:

```

Diem dinhB = new Diem() { X = 10, Y = 20 };

```

- Khởi tạo đối tượng với kiểu anonymous:

Ví dụ:

```

var dinhC = new { X = 9, Y = 11};

```

3.4 Properties và Automatic Properties

3.4.1 Properties

Properties là phần tử dùng để cập nhật và truy xuất đến đặc điểm của một đối tượng – field ở mức private. Properties được định nghĩa bằng 2 phần, phần thứ nhất giống như định nghĩa Fields, phần thứ 2 có thêm 2 phần tử get và set.

Ví dụ: Property cho thuộc tính hoành độ x trong lớp Diem.

```

class Diem
{
    // Các thuộc tính
    private int x; // x viết thường
    private int y; // y viết thường

    // Các phương thức Properties
    public int X // X viết hoa
    {
        get { return x; } // x viết thường
    }
}

```

```
set { x = value; } // x viet thuong
}
}
```

3.4.2 Automatic Properties

Để đơn giản hóa việc định nghĩa các get/set giống nhau ở các properties, automatic properties cho phép người dùng khai báo một cách chung chung `get; set;`. Thay vào đó trình biên dịch có thể tự động tạo ra các private field và những thao tác get/set mặc định cho chúng.

Ví dụ:

```
public class HangHoa {
    public string MaHang { get; set; }
    public string TenHang { get; set; }
    public int SoLuong { get; set; }
}
```

3.5 Static và Extension Method

3.5.1 Từ khóa static

- Các thành viên (biến, phương thức) tĩnh (static) cho phép chúng ta truy cập trực tiếp mà không cần phải tạo thể hiện (đối tượng) của lớp.
- Mọi thao tác truy xuất thông qua tên class.
- Static member:
 - Dữ liệu thuộc mức lớp.
 - Độc lập với các đối tượng.
 - Chỉ có một thể hiện (instance) duy nhất.
 - Dữ liệu được cấp phát khi chương trình bắt đầu chạy.
- Static method:
 - Chỉ sử dụng được biến static.

Ví dụ:

```
class StaticClass
{
    //static member
    static int count;

    //static method
    public static void Print()
    {
        Console.WriteLine("Count = " + count);
    }
}
```

Khai báo sử dụng hàm print():

```
StaticClass.Print();
```

3.5.2 Phương thức mở rộng

- Extension Methods (phương thức mở rộng) là phương thức được viết thêm vào một class **static** hiện có mà không cần một cấp thừa kế, biên dịch lại, hoặc sửa đổi mã nguồn gốc. Extension Methods được viết dưới dạng hàm tĩnh (static), tức là bạn sẽ gọi hàm này mà không cần phải khởi tạo một đối tượng.

- Khai báo phương thức mở rộng:

```
public static <kiểu trả về hàm> tên_hàm (<this|> <kiểu đối tượng>
mở_rộng> tên_đối_tượng)
{
    //Nội dung hàm
}
```

Ví dụ: Cài đặt phương thức đổi sang chữ hoa chuỗi cho trước là phương thức được thêm vào lớp string đã có.

```
static class Program
{
    public static string doisangchuhoa(<this|> string s)
    {
        return s.ToUpper();
    }
    static void Main(string[] args)
    {
        string s = "Hello eVery body!";
        Console.WriteLine(s.doisangchuhoa());
    }
}
```

3.6 Kiểu Anonymous Type

Anonymmous Type - kiểu dữ liệu trừu tượng - được dùng khi khai báo đối tượng chưa xác định được kiểu. Kiểu dữ liệu của biến sẽ được xác định khi gán giá trị cụ thể cho biến.

```
static void Main(string[] args)
{
    var a1 = new { Item100 = 1234, Item200 = "Hello World", Item300 = true };
    Console.WriteLine(a1.Item100 * 2);           // 246
    Console.WriteLine(a1.Item200.ToUpper());      // HELLO WORLD
    Console.WriteLine(a1.Item300 ? "One" : "Two"); // One
}
```

3.7 Bài tập

Bài tập 1: SinhVien

Tạo ứng dụng Console, thêm vào 1 class tên SinhVien bao gồm các thành phần dữ liệu: MaSV, HoTen, NgaySinh, DiaChi, DienThoai.

- Khai báo và định nghĩa các Constructor tham số và không tham số.
- Khai báo và định nghĩa các properties để truy cập đến giá trị của các thành phần dữ liệu (get, set) và viết thêm 1 properties chỉ đọc (get) dùng để lấy LayTuoia của SinhVien.
- Viết hàm Main() để kiểm tra lớp trên.

Bài tập 2: HìnhHoc

Tạo ứng dụng Console, thêm vào 1 class tên HinChuNhat gồm các thành phần dữ liệu: mChieuCao, mChieuRong.

- Viết các constructors không tham số và có tham số để khởi tạo các giá trị cho mChieuCao và mChieuRong.
- Viết các properties để truy cập đến 2 thành phần trên.
- Viết 2 phương thức trả về chu vi ($mChieuDai+mChieuRong$) x 2 và diện tích hình chữ nhật ($mChieuDai \times mChieuRong$)
- Viết hàm Main để kiểm tra lớp HinChuNhat: khởi tạo đối tượng, các giá trị cho đối tượng, gọi các phương thức,

Bài tập 3: QLHinhHoc

Tạo ứng dụng Console, thực hiện các yêu cầu sau:

- Thêm vào 1 class HinHoc gồm các thành phần biểu diễn diện tích (mDienTich) và chu vi (mChuVi). Viết phương thức để xuất giá trị của mDienTich và mChuVi ra màn hình.
- Thêm vào 1 class HinChuNhat kế thừa từ lớp HinHoc biểu diễn thông tin hình chữ nhật bao gồm thuộc tính riêng của nó là mChieuDai, mChieuRong.
 - Khai báo thành phần dữ liệu cần thiết để biểu diễn hình chữ nhật.
 - Khai báo và định nghĩa các constructor cần thiết.
 - Khai báo và định nghĩa các properties để truy cập đến giá trị của các thành phần dữ liệu mChieuDai và mChieuRong (get, set).
 - Viết 2 phương thức tính diện tích hình chữ nhật ($mChieuDai \times mChieuRong$) và chu vi ($mChieuDai+mChieuRong$) x 2, kết quả gán vào thuộc tính mDienTich, mChuVi.
- Thêm 1 lớp tên HinTron kế thừa từ lớp HinHoc và viết thêm các thành phần sau :
 - Khai báo thêm các thành dữ liệu: mBanKinh.
 - Khai báo và định nghĩa các Constructor tham số và không tham số để khởi tạo các giá trị cho các thành phần dữ liệu.
 - Khai báo và định nghĩa các properties để truy cập đến giá trị của các thành phần dữ liệu (get, set).
 - Viết các phương thức tính chu vi và diện tích hình tròn (cách viết giống như lớp hình chữ nhật).
- Viết hàm Main() để kiểm tra các các constructor, các properties, các phương thức của các lớp trên.

Bài tập 4: QLNhanVien

Tạo 1 ứng dụng Console, thực hiện các yêu cầu sau:

- Thêm vào 1 class tên Nguoi bao gồm các thành phần dữ liệu: HoTen, NgaySinh, DiaChi.
 - Khai báo và định nghĩa các constructor tham số và không tham số.

- Khai báo và định nghĩa các properties để truy cập đến giá trị của các thành phần dữ liệu (get, set) và viết thêm 1 properties chỉ đọc (get) dùng để lấy LayTuoi của Nguoi.
- Viết 1 phương thức tên XemThongTin(): xuất giá trị các thành phần dữ liệu ra màn hình.
- Thêm 1 lớp tên SinhVien kế thừa từ lớp Nguoi và viết thêm các thành phần sau:
 - Khai báo thêm các thành dữ liệu: string MaSV , string MaLop, string Email, string DienThoai.
 - Khai báo và định nghĩa các Constructor tham số và không tham số để khởi tạo các giá trị cho các thành phần dữ liệu .
 - Khai báo và định nghĩa các properties để truy cập đến giá trị của các thành phần dữ liệu (get, set) kiểm tra dữ liệu.
- Thêm 1 lớp tên NhanVien kế thừa từ lớp Nguoi và viết thêm các thành phần sau:
 - Khai báo thêm các thành dữ liệu: string MaNhanVien , string Email, string DienThoai, DateTime NgayLamViec, string MaCongTy.
 - Khai báo và định nghĩa các Constructor tham số và không tham số để khởi tạo các giá trị cho các thành phần dữ liệu.
 - Khai báo và định nghĩa các properties để truy cập đến giá trị của các thành phần dữ liệu (get, set) và kiểm tra dữ liệu.
- Viết hàm Main() để kiểm tra các constructor, các properties, các phương thức của các lớp trên.
- Mở rộng thêm cho các đối tượng khác như GiamDoc, CaSi. Tất cả các lớp nên override lại phương thức *ToString()*.
- Sử dụng List để khai báo mảng các đối tượng.

```
List<Nguoi> ds = new List<Nguoi>();  
ds.Add(new SinhVien{.....});  
ds.Add(new NhanVien{.....});  
ds.Add(new Nguoi{.....});  
ds.Add(new SinhVien{.....});  
  
//duyet danh sach  
...
```

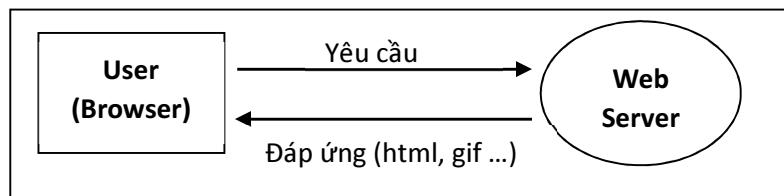
-----oOo-----

Chương 4: THIẾT KẾ GIAO DIỆN WEB

4.1 Ngôn ngữ HTML

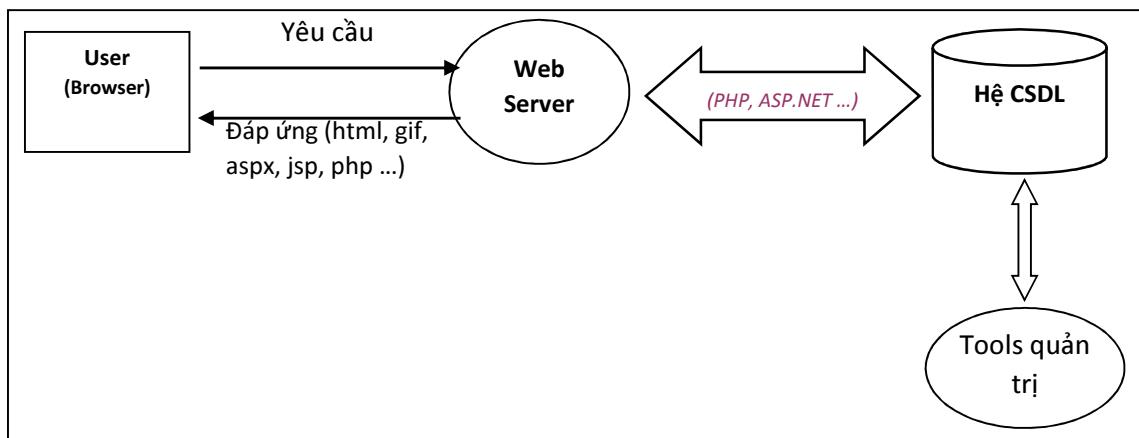
4.1.1 Một số khái niệm

- a) **Trang web:** Trang web (tĩnh) là một file dạng text chứa dữ liệu và các tag HTML. Khi hiển thị trong trình duyệt web, dữ liệu sẽ được hiển thị theo quy định của các tag mà nó nằm bên trong. Dữ liệu trong trang web có thể là văn bản, hình ảnh, âm thanh, video...
- b) **Hyperlink:** là 1 liên kết chỉ đến 1 trang web khác. Một trang web có thể chứa nhiều link.
- c) **Website:** Là tập hợp nhiều trang web thể hiện thông tin của 1 tổ chức, 1 chủ đề nào đó. Mỗi website có 1 trang trang chủ chứa các hyperlink liên kết đến các trang khác trong website. Người xem sẽ vào website bắt đầu từ trang chủ; từ trang này, nhờ các link trong đó mà họ sẽ đến được các trang khác trong toàn website.
- d) **Browser :** Là chương trình dùng để xem các trang web. Các trình duyệt web nổi tiếng là Internet Explorer, FireFox, Google Chrome.
- e) **WebServer:** Là các máy “phục vụ web”, đây là các máy tính trên Internet/Intranet có cài chương trình webserver. Webserver sẽ trả về cho người sử dụng trang web mà họ yêu cầu để họ xem. Webserver liên lạc với browser qua giao thức http(s). Một Webserver có thể chứa nhiều website. Hai chương trình webserver nổi tiếng nhất là: IIS và Apache.
- f) **Http:** là giao thức để browser và web server trao đổi với nhau nhằm đưa trang web cho người dùng xem.
- g) **Web tĩnh:** Là trang web chỉ có tag html và dữ liệu, tất cả đều gõ trực tiếp trong trang chứ không đặt ở nơi khác. File có tên mở rộng là. html hoặc. htm.



Trong mô hình web tĩnh, user yêu cầu 1 trang web html, trang web này đã được thiết kế sẵn và đặt trên webserver, trang web không hề có tương tác đến CSDL. Webserver chỉ việc lấy file html trả về cho user. Vậy là xong.

- h) **Web động:** Là trang web có truy xuất đến cơ sở dữ liệu (Database) hoặc có tương tác với webserver để thực hiện 1 chức năng cao cấp nào đó. Một trang web động có thể trả về những kết quả khác nhau tùy theo yêu cầu của người sử dụng. Thiết kế web động đòi hỏi người thiết kế có nhiều kiến thức: HTML, Javascript, Database, WebServer, ... tốn nhiều công sức và thời gian. Mô hình Web động:



Trong mô hình web động, Webserver sẽ tương tác với các chương trình “hậu trường” phía sau nó (PHP, ASP.NET...) để thực hiện 1 số việc nào đó (thường là kết nối cơ sở dữ liệu), các chương trình này lấy dữ liệu trong hệ quản trị cơ sở dữ liệu và thực hiện định dạng (nếu cần) rồi đưa về cho webserver để webserver sẽ trả về cho user.

Các chương trình “hậu trường” như PHP, ASP.NET... là chương trình trung gian, là cầu nối giữa Webserver và cơ sở dữ liệu. Sở dĩ có chúng là vì tương tác với cơ sở dữ liệu không phải là mục tiêu của webserver, nhiệm vụ chính của WebServer là tương tác với user để trả về trang web (qua giao thức http).

4.1.2 Giới thiệu HTML

- HTML (*Hyper Text Markup Language*) là một ngôn ngữ để quy định cách hiển thị thông tin trong trang web. HTML gồm nhiều lệnh, mỗi lệnh gọi là 1 tag. Mỗi tag quy định một cách thức hiển thị dữ liệu trong trang web. Ví dụ như: chữ đậm, chữ nghiêng, màu chữ ... Người xem trang web không thấy các tag mà chỉ thấy các dữ liệu được định dạng bởi các tag. Nói đơn giản : **HTML là 1 ngôn ngữ dùng để tạo ra các trang web.**
- Các tag cùng với dữ liệu trong đó được lưu trong 1 file text, gọi là trang web. File này thường có tên mở rộng là .html hoặc .htm.
- Ví dụ: Nếu bạn gõ như sau khi tạo trang web:

Lớp: ASP.NET

 Họ tên: <u><i>Nguyễn Văn Tèo</i></u>

thì kết quả hiện trong Browser sẽ thế này:

Lớp: **ASP.NET**
 Họ tên: *Nguyễn Văn Tèo*

- Tên tag không quan trọng chữ thường chữ hoa, tên tag phải đặt trong 2 dấu < >, thường có mở và đóng. Một số tag chỉ có mở như <hr>,
, .

4.1.3 Một số tag thường dùng

, 	Chữ đậm
<i>, 	Chữ nghiêng
<u>	Chữ gạch dưới
	Tạo 1 hyperlink
	Chèn hình
<hr>	Chèn 1 đường gạch ngang

<p align=cáchcanh>	Tạo paragraph mới
 	Xuống hàng, không tạo paragraph mới

4.1.4 Cấu trúc của 1 trang web

- Một trang web thường có mở đầu và kết thúc bởi tag **html**
- Tag **head** chứa những thông tin để quản lý và hoạt động nội tại bên trong trang web, không hiện ra cho user xem.
- Tag **title** là tiêu đề của trang web, bao giờ cũng nằm trong tag head
- Tag **body** chứa dữ liệu hiện ra trong trang web cho user xem.



4.1.5 Soạn thảo trang web

- Để soạn 1 trang web, bạn dùng 1 chương trình soạn văn bản để gõ. Có thể dùng Notepad, Wordpad hoặc MS Word. Bạn nên dùng notepad vì đây là chương trình đơn giản, dễ dùng, notepad lưu file dưới dạng text – loại file của trang web. Nếu bạn dùng Wordpad hoặc MS Word thì phải chọn dạng file là Text.
- Cách mở Notepad : Nhấn Start ➔ (All) Programs ➔ Accessories ➔ Notepad
- Lưu ý: khi soạn thảo trang web, bạn gõ các tag và các dữ liệu bên trong tag. Một đoạn dữ liệu có thể nằm bên trong nhiều tag. Hãy nhớ là các tag được lồng nhau chứ không nên gõ so le nhau. Ví dụ:
 - Gõ thế này là đúng: <i> Nhất nghệ </i> ➔ tag i lồng (nằm trong tag b).
 - Gõ thế này là sai: <i> Nhất nghệ </i> ➔ 2 tag so le nhau. Mặc dù gõ thế này đôi khi browser cũng hiện ra nhưng đó là “hên”, là nhờ browser thông minh và cấu trúc trang web của bạn đơn giản, trong những trường hợp khác hoặc browser khác thì chưa chắc.

Ví dụ:

1. Mở Notepad và gõ nội dung sau:

```

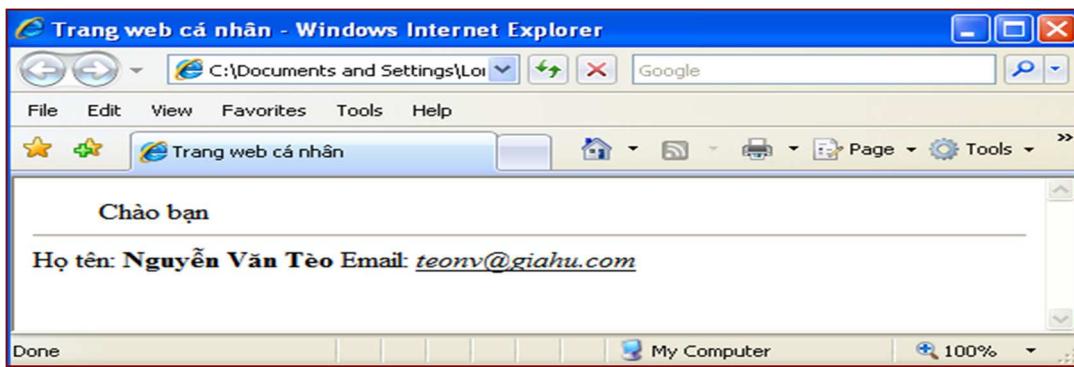
<html>
  <head>
    <title>Trang web cá nhân</title>
  </head>
  <body>
    <marquee> Chào bạn </marquee>
    <hr>
  </body>
</html>

```

```

        Họ tên: <b> Nguyễn Văn Tèo</b>
        Email: <i><u> teonv@giahu.com </u></i>
</body>
</html>
```

- Nhấn menu File → Save để lưu file. Lưu vào **Desktop**, tên file là **vidu.html**, Encoding là **UTF-8**.
- Đóng Notepad.
- Nhấn đúp vào file **vidu.html** trên Desktop để xem kết quả.



2. Thêm lệnh xuống hàng giữa Họ tên và Email:

- Thêm sau tag lệnh xuống hàng:

- Đóng Notepad và lưu lại.
- Nạp lại trang vidu.html (phím F5) để xem những thay đổi



3. Đưa hình (h.jpg) vào trang web:

- Xem Source HTML của trang
- Nhấn sau tag <hr> và gõ như sau:

```
<img src=h.jpg>
```

- Mở thêm 1 cửa sổ Browser. Vào Google, tìm 1 hình nào đó (cỡ lớn) và lưu vào Desktop với tên h.jpg.
- Nạp lại trang vidu.html để xem kết quả

4. Chính kích thước và title, cách canh và đường viền cho hình:

- Xem Source HTML của trang
- Nhấn sau chữ và bổ sung để được như sau:

Xong thì lưu lại

- Nạp lại trang vidu.html để xem kết quả

Chào bạn



Họ tên: **Nguyễn Văn Tèo**
Email: teonv@giahu.com

- Sửa chữ **left** thành **right** và xem kết quả

5. Chính direction (chiều cuộn), behavior (cách cuộn), scrolldelay (thời gian dừng), scrollamount (khoảng cách nhảy) của marquee.

- Xem Source HTML của trang
- Nhập sau chữ <marquee và bổ sung để được như sau:

<marquee behavior=scroll direction=right scrolldelay=10 scrollamount=1> Chào bạn </marquee>

Xong thì lưu lại

- Nạp lại trang vidu.html để xem kết quả
- Có thể sửa chữ **scroll** thành **slide** hoặc **alternate**
- Có thể sửa chữ **right** thành **left** hoặc **up** hoặc **down**
- Có thể sửa số **10**, số **1** thành những giá trị khác (>=1)

6. Thêm liên kết vào trang (liên kết nằm trong tag a)

- Xem Source HTML của trang.
- Nhập sau chữ </i> , Enter xuống hàng vào nhập vào:

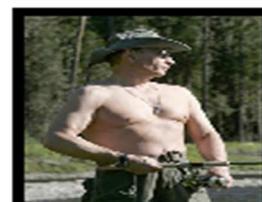
<p>Website thường xem: </p>
 Nhất nghệ

 Google

Xong thì lưu lại.

- Nạp lại trang vidu.html để xem kết quả:

Chào bạn



Họ tên: **Nguyễn Văn Tèo**
Email: teonv@giahu.com

Website thường xem:

[Nhất nghệ](#)
[Google](#)

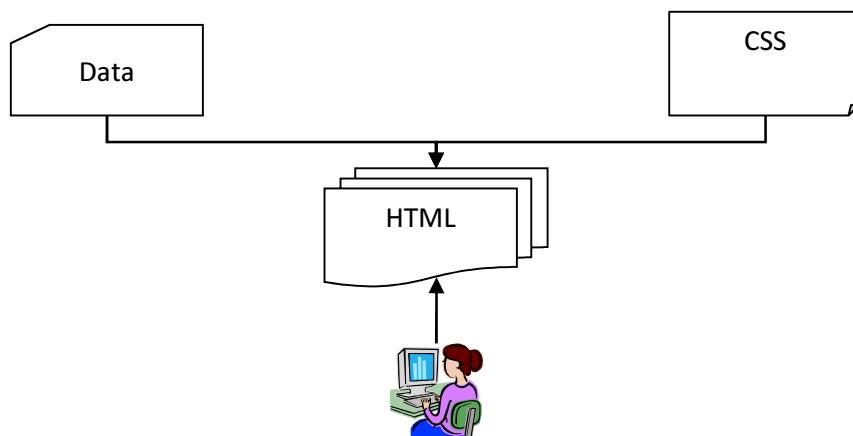
7. Các tag HTML căn bản

Tên Tag	Cú pháp	Định nghĩa
<a>	Tên hiển thị	Tạo liên kết đến trang abc.html
	Nội dung đoạn văn	In đậm
 	Nội dung đoạn văn bắt đầu một dòng mới	Xuống dòng, không qua đoạn mới
<bgsound>	<bgsound delay="1" loop="-1" src="start.wav">	Nhạc nền cho trang web
<center>	<CENTER>Canh giữa chữ</CENTER>	Canh giữa
<div>	<div>.....</div>	div chứa 1 vùng dữ liệu trong trang
	 Nội dung 	Định dạng kiểu chữ nghiêng
<h1> to <h6>	<h1>Tiêu đề 1 </h1> <h2>Tiêu đề 2 </h2> <h3>Tiêu đề 3 </h3> <h4>Tiêu đề 4 </h4> <h5>Tiêu đề 5 </h5> <h6>Tiêu đề 6 </h6>	Tạo tiêu đề (cấp 1 đến cấp 6)
<hr>	<hr color="#FF0000">	Tạo một đường gạch ngang
<i>	<i> Nội dung </i>	Chữ in nghiêng
<iframe>	<iframe name="content_frame" width="488" height="244" src="welcome.htm"> </iframe>	Tạo 1 iframe (iframe là 1 vùng trong trang chứa 1 trang web khác)
		Chèn hình vào văn bản.
<marquee>	<marquee direction="left" loop="-1" scrollamount="2" width="100%">Chữ cuộn</marquee>	Là tag dùng để cuộn (hình, văn bản).
<p>	<p>Nội dung đoạn văn bản.</p>	Paragraph
<small>	<small>Nội dung văn bản</small>	Chữ nhỏ
	 Nội dung văn bản	Bao quanh 1 vùng text để định dạng
	Nội dung đoạn văn bản	Chữ đậm
<sub>	_{Nội dung đoạn văn bản}	Chữ subscript (chữ xuống dưới+nhỏ)
<sup>	^{Nội dung đoạn văn bản}	Chữ superscript (chữ lên cao+nhỏ)
<u>	<u>Nội dung đoạn văn bản</u>	Gạch dưới

4.2 Bảng định kiểu – CASCADING STYLE SHEET (CSS)

4.2.1 Giới thiệu

- CSS là 1 kỹ thuật dùng để định dạng các tag trong trang web. CSS giúp định dạng trang web rất nhanh nhờ nhiều kiểu định dạng tag, class, element... Bạn không thể định dạng 1 trang web cho đẹp khi không có sự am hiểu về CSS. Nếu làm được điều này, chúng ta có được các lợi điểm sau:
 - ✓ Dễ quản lý, bảo trì.
 - ✓ Tái sử dụng. Một qui luật kiểu dáng có thể áp dụng cho nhiều thành phần web khác nhau.
 - ✓ Cải thiện tốc độ.
 - Giảm lượng thông tin truyền tải.
 - Cách hiển thị của trình duyệt



- Style: Là 1 tập hợp các đặc điểm định dạng cho các thành phần trong trang. Để định dạng, ta chuyển sang chế độ code, rồi định nghĩa các style bên trong tag <style>. Tag <style> cần đặt trong tag head.

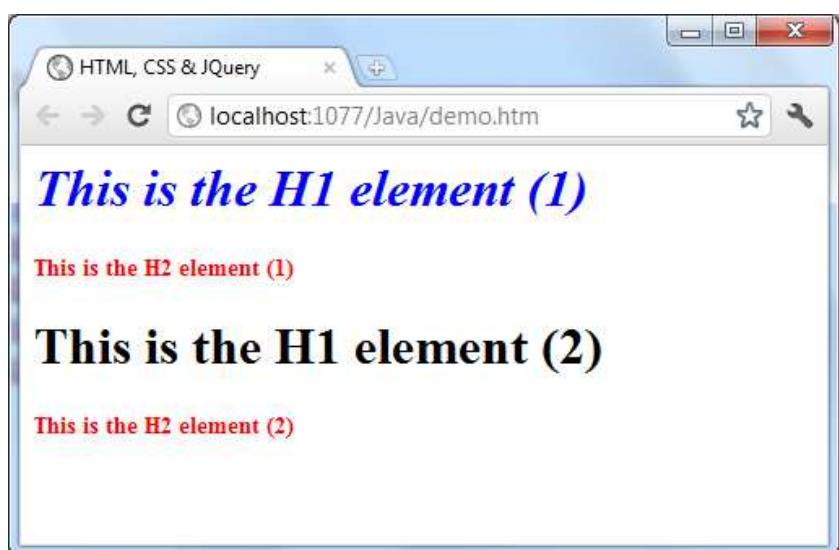
4.2.2 Khởi động nhanh

Để không gặp khó khăn của các qui luật của CSS, hãy tìm hiểu một ví dụ đơn giản về nó. Qua đó chúng ta có dịp làm quen với các khái niệm của CSS.

Mã nguồn HTML

```

<HTML>
<HEAD>
    <title>HTML, CSS & JQuery</title>
    <STYLE TYPE="text/css">
        H2{color:red;font-size: 14px;}
    </STYLE>
</HEAD>
<H1 style="color:blue;font-style:italic;">
    This is the H1 element (1)</H1>
<H2>This is the H2 element (1)</H2>
<H1>This is the H1 element (2)</H1>
<H2>This is the H2 element (2)</H2>
</HTML>
    
```



Kết quả thực hiện

Phân tích ví dụ:

- Các thẻ `<h2>` nhỏ và màu đỏ là do dòng mã CSS `H2{color:red;font-size: 14px;}`. Còn thẻ `<h1>` nghiêng và có màu xanh do thuộc tính style chứa CSS `style="color:blue;font-style:italic;"`.
- Phương pháp định nghĩa CSS cho 1 thẻ phù hợp cho áp dụng hàng loạt ngược lại phương pháp inline (sử dụng thuộc tính style) thì các CSS được tạo ra chỉ để áp dụng cho từng trường hợp đơn lẻ.

4.2.3 Tạo style định dạng

4.2.3.1 Tạo style định dạng cho 1 tag trong toàn trang:

Muốn định dạng tag nào thì style sẽ giống như tag muốn định dạng.

Ví dụ: định dạng tag **p** và tag **a**

```
<style>
p {color:#F00}
a { color:#039; text-decoration:none}
<style>
```

4.2.3.2 Tạo style định dạng cho 1 đối tượng cụ thể có tên:

Muốn định dạng cụ thể 1 tag nào đó theo tên do bạn đặt thì tạo style bắt đầu bằng dấu **#**.

Ví dụ sau định dạng cho 1 tag có tên là **box**

```
<style>
#box { width:300px; height:150px; text-align:justify}
<style>
```

Chú ý: tag phải đặt tên theo id khớp với style đã tạo thì mới có tác dụng

```
<div id="box"> .... </div>
```

4.2.3.3 Tạo style định dạng cho tag bên trong 1 đối tượng có tên:

Muốn định dạng tag bên trong 1 vùng thì tạo style theo công thức sau: **#TênVùng tag**

Ví dụ sau định dạng cho các tag **a** trong vùng có tên là **box**

```
<style>
#box a { color: magenta; text-transform: uppercase}
<style>
```

4.2.3.4 Tạo style dạng class và set class

Muốn tạo class gõ theo công thức sau: **.TênClass** Ví dụ sau tạo class. tieude

```
<style>
.caption {color:#993; padding:5px; margin:0px; text-align:center }
<style>
```

Set class: Tạo class xong, muốn tag nào định dạng theo class thì chỉ định thông số class.

Ví dụ:

```
<h4 class="caption">Tin xem nhiều</caption>
```

4.2.3.5 Tạo file css và nhúng vào trang web

❖ Tạo file css

- Nhấp menu File → New → chọn CSS → Create → Lưu với tên file. css
- Khai báo các style

❖ Nhúng file css vào trang web

```
<link href="c1.css" rel="stylesheet" type="text/css" />
```

Trong trang web cần áp dụng CSS, gõ code trên để nhúng file css, trong đó c1.css là tên file css muốn nhúng.

4.2.4 Các thuộc tính CSS

Khi thiết kế trang web với CSS thì vốn kiến thức CSS là khối lượng các thuộc tính CSS mà bạn có được. Sau đây trình bày danh sách các thuộc tính thường dùng nhất có phân loại.

Các thuộc tính CSS thường sử dụng để định nghĩa cho văn bản trên trang web như font chữ, màu sắc, chế độ hiển thị...

4.2.4.1 Định dạng chữ

Thuộc tính	Mô tả
font-family: Verdana, Geneva, sans-serif;	Chỉ định tên font. Các font được liệt kê các nhau dấu phẩy. Áp dụng cho font được tìm thấy trước
font-size: 12px;	Kích thước font
font-style: italic;	Kiểu font (italic: chữ nghiêng)
line-height: 12px;	Độ cao của mỗi hàng
font-weight: bold;	Độ đậm của font chữ: bold (đậm), 100 (độ đậm 100)
font-variant: small-caps;	Chữ hoa nhỏ, ký tự đầu lớn hơn
text-transform: uppercase;	đổi chữ hoa, chữ thường (capitalize: chữ hoa đầu từ, uppercase: toàn chữ hoa, lowercase: toàn chữ thường)
color: #F00;	Màu sắc có thể dùng mã (Red, Green, Blue) hoặc tên màu
text-decoration: none;	Trang trí chữ: Underline: gạch dưới chữ, Strikethrough: gạch giữa chữ, Overline: gạch đầu chữ, None: không gạch
text-align: center;	canh chữ (left, right, center, justify)
text-shadow	màu bóng của chữ
letter-spacing: 2em;	Khoảng cách giữa các ký tự
text-align: justify;	Canh lề: left, right, center, justify
text-indent: 5px;	Khoảng thụt vào đầu dòng
vertical-align: middle;	Canh lề đứng: top, bottom, middle, base-line
word-spacing: 4em;	Khoảng cách giữ các từ
letter-spacing: 2em;	Khoảng cách giữa các ký tự

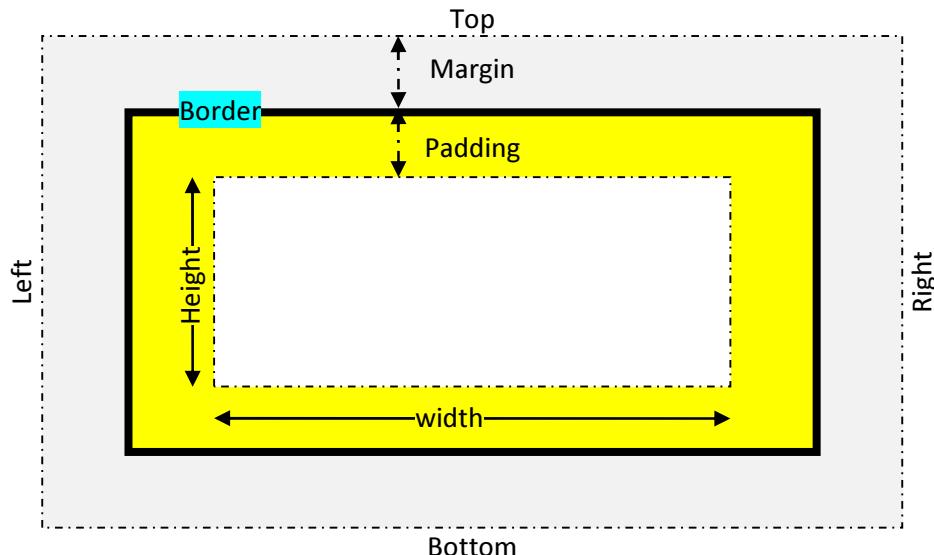
4.2.4.2 Background

Nền chỉ có 2 loại là màu và ảnh. Nếu là ảnh thì cần điều chỉnh chế độ lặp lại (lát). Trong trường hợp không lặp bạn cần điều chỉnh vị trí đặt ảnh nền.

Thuộc tính	Mô tả
background-color: #F00;	Màu nền
background-attachment: fixed;	Chế độ cuộn ảnh nền ✓ Fixed = cố định ảnh nền khi cuộn nội dung ✓ Scroll = ảnh nền cuộn theo nội dung
background-image: url(anh/abc.jpg);	Ảnh nền
background-repeat: repeat;	Chế độ lặp: ✓ None: không lặp ✓ Repeat: lặp cả 2 chiều ✓ repeat-x: lặp chiều ngang ✓ repeat-y: lặp chiều đứng
background-position: left center;	Vị trí đặt ảnh nền trường hợp không lặp
Background-size: 100% 100%	Kích thước ảnh nền (width height)

4.2.4.3 Box

Mô hình của một hộp gồm kích thước, lề, phần đệm, được viền, chế độ xếp hộp.



Thuộc tính	Mô tả
height: 222px;	Chiều cao
width: 111px;	Chiều rộng
margin: 6px;	Khoảng cách từ lề của đối tượng với những đối tượng bên ngoài. Sử dụng margin-top, margin-right, margin-bottom, margin-left nếu muốn định nghĩa riêng mỗi cạnh.
padding: 4px;	Phần đệm bên trong hộp (Khoảng cách từ lề của đối tượng với nội dung bên trong). Sử dụng padding -top, padding -right, padding -bottom, padding -left nếu muốn định nghĩa riêng mỗi cạnh.
border: medium dotted #F00;	Đường kẻ theo thứ tự độ dày, kiểu, màu . Sử dụng border -top, border -right, border -bottom, border -left nếu muốn định nghĩa riêng mỗi cạnh.
float: left;	Gâm (chế độ xếp hộp) vào trái: left (gâm trái), right(gâm phải)
clear: right;	Hủy bỏ chế độ gâm: left(xóa gâm trái), right(xóa gâm phải), both(xóa gâm cả 2 bên)

4.2.4.4 Border

- border-style: kiểu đường viền.
- border-width: độ dày.
- border-color: Màu đường viền.
- border-radius: bo tròn góc.
- box-shadow: tạo bóng cho đối tượng định dạng

4.2.4.5 List

Để điều chỉnh ``, `` và `` bạn cần sử dụng các thuộc tính css sau đây.

Thuộc tính	Mô tả
list-style-position: inside;	Vị trí đặt dấu danh sách
list-style-type: square;	Kiểu dấu danh sách: <i>disc: tròn đen; circle: tròn trắng; square: vuông...</i>
list-style-image: url(xyz/abc.gif);	hình dùng thay thế ký tự bullet

4.2.4.6 Layer

Để tạo ra và điều chỉnh các thông số của nó, bạn cần học các thuộc tính CSS sau đây là đủ.

Thuộc tính	Mô tả
overflow: scroll;	Điều khiển chế độ tràn: scroll, visible, hidden
position: relative;	Chế độ vị trí của layer. Absolute (vị trí tuyệt đối so với layer mẹ), relative (vị trí tương đối tức đặt tại vị trí đặt thẻ)
visibility: visible;	Ẩn hiện layer
left: 0px;	Vị trí layer tính từ bên trái
top: 0px;	Vị trí layer tính từ bên trên
right: 0px;	Vị trí layer tính từ bên phải
bottom: 0px;	Vị trí layer tính từ bên dưới
z-index: 111;	Chiều z hướng từ trong màn hình ra người dùng. Layer nào có z-index cao hơn sẽ nằm trên.

4.2.5 Bộ chọn (Selector)

Bộ chọn (selector) là nơi định nghĩa các qui luật kiểu dáng để áp dụng cho các thành phần trên trang web. Có 3 loại bộ chọn cơ bản là Class, ID và HTML

4.2.5.1 Bộ chọn HTML (HTML Selector)

- ✓ Định nghĩa: định nghĩa kiểu dáng bổ sung cho các thẻ HTML

<thẻ>{<khai báo các thuộc tính CSS>}

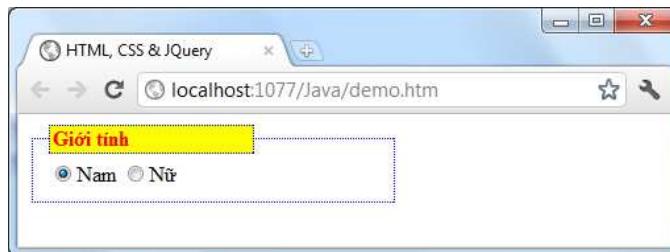
- ✓ Áp dụng: Tự động áp dụng các qui luật CSS trong phần khai báo cho tất cả các thẻ có tên là <thẻ>
- ✓ Ví dụ sau đây định nghĩa lại thẻ <fieldset> và legend với các thuộc tính kích thước (width), đường kẻ (border), màu chữ (color), màu nền (background-color).

```

<HTML>
<HEAD>
    <title>HTML, CSS & JQuery</title>
    <STYLE TYPE="text/css">
        FIELDSET{
            width: 250px;
            border: 1px dotted #0000FF;
        }
        LEGEND{
            font-weight: bold;
            color: #FF0000;
            background-color: #FFFF00;
            border: 1px dotted #0000FF;
            width: 150px;
        }
    </STYLE>
</HEAD>
<body>
    <fieldset>
        <legend>Giới tính</legend>
        <input type="radio" name="rdoGioiTinh" checked/>Nam
        <input type="radio" name="rdoGioiTinh" />Nữ
    </fieldset>
</body>
</HTML>

```

Kết quả hiển thị



4.2.5.2 Bộ chọn lớp (Class Selector)

- ✓ Định nghĩa: định nghĩa một lớp được bắt đầu bởi dấu chấm(.) bên trong khai báo nhiều thuộc tính CSS để áp dụng cho bất kỳ thẻ nào chỉ định bởi thuộc tính class của nó.

.<Tên lớp>{<khai báo các thuộc tính CSS>}

- ✓ Áp dụng: tất cả các thẻ sử dụng thuộc tính class có giá trị là <Tên lớp>. Chú ý thuộc tính class của mỗi thẻ có thể chỉ đến nhiều class cùng một lúc (cách nhau khoản trắng).
- ✓ Ví dụ sau định nghĩa 2 bộ chọn lớp sau đó thẻ <H1> áp dụng một cùn thẻ <DIV> áp dụng cả hai để tận dụng các đặc điểm tổng hợp.

```
<HTML>
<HEAD>
<title>HTML, CSS & JQuery</title>
<STYLE TYPE="text/css">
. MyHeader{
    font-family: Arial, Helvetica, sans-serif;
    font-weight: bold;
    font-style: italic;
    font-size: 14px;
    color: #FF0000;
}

```

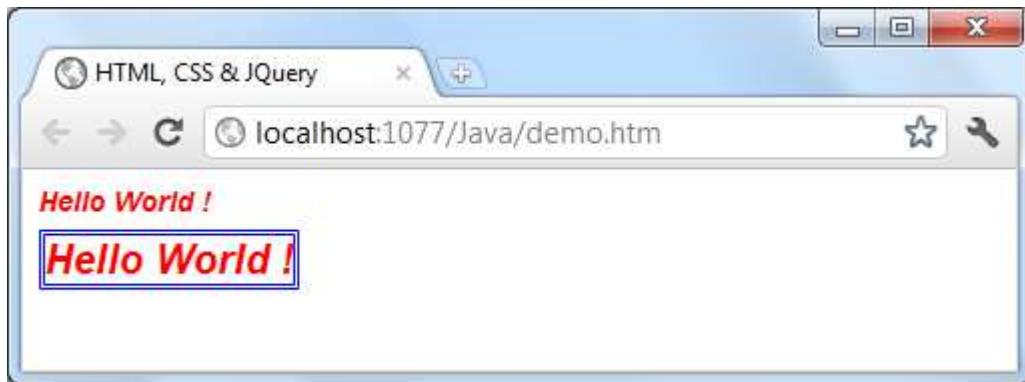
```
. MyBorder{
    border: 3px double blue;
    font-size: 20px;
    display: inline;
}
</STYLE>
</HEAD>
<body>
<h1 class="MyHeader">Hello World !</h1>
<div class="MyHeader MyBorder">Hello World !</div>
</body>
</HTML>
```

Trong đó

- ✓ font-family: tên font chữ

- ✓ font-weight: độ đậm
- ✓ font-style: kiểu chữ
- ✓ font-size: kích thước chữ
- ✓ color: màu chữ

Kết quả thực hiện



4.2.5.3 Bộ chọn định danh (ID Selector)

- ✓ Định nghĩa: giống như bộ chọn lớp nhưng khởi đầu với dấu rào (#)

#<tên định danh>{<khai báo các thuộc tính css>}

- ✓ Áp dụng: tất cả các thẻ sử dụng thuộc tính id với giá trị là <tên định danh>
- ✓ Ví dụ sau định nghĩa bộ chọn định danh tên là #MyPara sau đó áp dụng cho một thẻ <P> trong trang web. Chú ý thẻ <P> còn lại không hề bị ảnh hưởng gì.

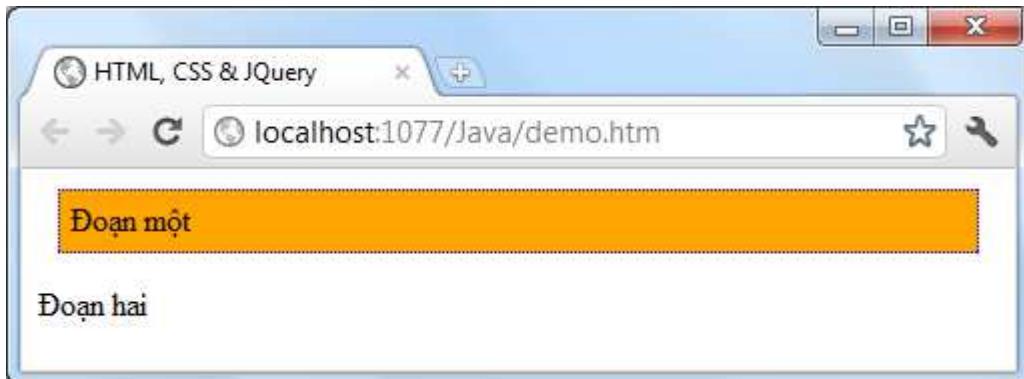
```

<HTML>
<HEAD>
<title>HTML, CSS & JQuery</title>
<STYLE TYPE="text/css">
#MyPara
{
    background-color: orange;
    background-image: url(images/abc.gif);
    text-align: justify;
    margin: 10px;
    padding: 5px;
    border: 1px dotted #0000FF;
}
</STYLE>
</HEAD>
<body>
<p id="MyPara">Đoạn một</p>
<p>Đoạn hai</p>
</body>

```

</HTML>

Kết quả thực hiện



4.2.5.4 Bộ chọn cho liên kết

Định nghĩa: định nghĩa CSS cho siêu liên kết. Với liên kết có bốn trạng thái sử dụng là chưa thăm (chưa click), đã thăm, có chuột và tích cực (đang chọn). Vì vậy để định nghĩa CSS áp dụng cho liên kết bạn không chỉ định nghĩa CSS cho thẻ <A> mà còn định nghĩa cả 4 trạng thái của nó. Sau đây là cú pháp chung định nghĩa CSS cho siêu liên kết

A: {<khai báo các thuộc tính CSS>}

A:link {<khai báo các thuộc tính CSS>}

A:visited{<khai báo các thuộc tính CSS>}

A:hover{<khai báo các thuộc tính CSS>}

A:active{<khai báo các thuộc tính CSS>}

Tên style	Ý nghĩa
a:link	định dạng cho tag a chưa được nhấp trong trang
a:visited	định dạng cho tag a đã được nhấp trong trang
a:hover	định dạng cho tag a trong trang đang được đưa chuột vào
a	định dạng cho tag a trong trang
#menu a:link	định dạng cho tag a chưa nhấp trong vùng có tên là menu
#menu a:visited	định dạng cho tag a đã được nhấp trong vùng có tên là menu
#menu a:hover	định dạng cho tag a đang đưa chuột vào trong vùng menu
#menu a	định dạng cho tag a trong vùng có tên là menu
.menu A:link	định dạng cho tag a (chưa nhấp) trong vùng có class là menu
.menu A:visited	định dạng cho tag a (đã nhấp) trong vùng có class là menu
.menu A:hover	định dạng tag a (đang đưa chuột vào) trong class là menu
.menu A	định dạng cho tất cả liên kết trong vùng có class là menu
.tieudetin	định dạng cho liên kết có class là tieudetin

- ✓ Áp dụng: Tất cả các liên kết trong trang có định nghĩa CSS cho liên kết
- ✓ Ví dụ

<HTML>

<HEAD>

<title>HTML, CSS & JQuery</title>

<STYLE TYPE="text/css">

```
A{  
    font-family: Arial; font-size: 16px; text-decoration: none; }  
A:link{ color: Blue; }  
A:visited { color: Green; }  
A:hover{  
    text-decoration: underline; color: Red;  
    border: 1px dotted Red; background-color: Yellow;  
}  
A:active { color: Orange; }  
</STYLE>  
</HEAD>  
<body>  
    <a href="#1">Link 1</a> | <a href="#2">Link 2</a> | <a href="#3">Link 3</a> |  
    <a href="#4">Link 4</a> | <a href="#5">Link 5</a> |  
</body>  
</HTML>
```

Kết quả thực hiện: Link3 đang có chuột, Link2 đã bị click trước đó

[Link 1](#) | [Link 2](#) | [Link 3](#) | [Link 4](#) | [Link 5](#) |

4.2.5.5 Nhiều bộ chọn cùng kiểu

- ✓ Định nghĩa: định nghĩa nhiều bộ chọn đồng một số kiểu dáng.

<bộ chọn 1>, <bộ chọn 2>, ..., <bộ chọn n>{<khai báo các thuộc tính css>}

- ✓ Áp dụng: áp dụng các khai báo css cho tất cả các thẻ có chỉ định sử dụng css thỏa mãn với các bộ chọn được liệt kê cách nhau dấu phẩy.
- ✓ Ví dụ

```
<html>  
<head>  
    <title>HTML, CSS & JQuery</title>  
    <style type="text/css">  
        #A,.B, DIV INPUT, H2  
        {  
            font-weight: bold; font-style: italic; color: #FF0000; font-size: 11pt;  
        }  
    </style>  
</head>  
<body>  
    <div class="B">Công cha như núi thái sơn</div>  
    <div id="A">Nghĩa mẹ như nước trong nguồn chảy ra</div>  
    <div><input value="Một lòng thờ mẹ kính cha" size="55" /></div>  
    <h2>Cho tròn đạo hiệu mới là đạo con</h2>  
    <input value="Thẻ input này không bị ảnh hưởng gì" size="55" />  
</body>  
</html>
```

Kết quả thực hiện

*Công cha như núi thái sơn
Nghĩa mẹ như nước trong nguồn chảy ra
Một lòng thờ mẹ kính cha*

Cho tròn đạo hiệu mới là đạo con

Thẻ input này không bị ảnh hưởng gì

Lưu ý: "DIV INPUT" có nghĩa là định nghĩa css cho các thẻ <INPUT> đặt trong các thẻ <DIV>. Vì vậy trong bài này thẻ <input> không đặt trong <div> không hề chịu tác dụng của css đã định nghĩa.

4.2.5.6 Bộ chọn khoanh vùng

- ✓ Định nghĩa: định nghĩa CSS cho các vùng khác nhau trên trang. Như vậy chúng ta cần xác định vùng cần áp dụng và bộ chọn chứa các CSS để áp dụng.

<vùng> <bộ chọn>{<khai báo các thuộc tính css>}

- ✓ Áp dụng CSS của bộ chọn cho các thẻ đặt trong <vùng> và chỉ định áp dụng bộ chọn.
- ✓ Ví dụ:

```
<HTML>
<HEAD>
<title>HTML, CSS & JQuery</title>
<STYLE TYPE="text/css">
    /*--võ bọc bên ngoài rộng 900px, canh giữa, nền trắng--*/
    .container{width:900px; margin: 0px auto; background-color: White;}
    /*--đầu trang cao 100px--*/
    .top{height: 100px; background-color: Red;}
    /*--menu trang cao 22px, canh giữa--*/
    .menu{height: 22px; background-color: Yellow; text-align:center}
    /*--giữa trang cao tối thiểu 400px--*/
    .middle{min-height: 400px;}
    /*--giữa-trái cao như middle, rộng 250px, gầm trái--*/
    .middle_left{
        float:left; width: 250px; min-height:inherit;
        background-color: Aqua;
    }
    /*--giữa-phải cao như middle, rộng 650px, gầm phải--*/
    .middle_right{
        float:right; width: 645px; min-height:inherit;
        background-color: White;
    }
    /*--chân trang không gầm, cao 22px--*/
    .bottom{clear:both; height: 22px; background-color: Yellow;}
    /*--fieldset trong. middle_left cao tối thiểu 150--*/
    .middle_left fieldset{min-height: 150px;}
    /*--li trong. middle_left không dùng dấu, kẽ chân--*/
    .middle_left li{list-style-type:none; border-bottom: 1px dotted red;}
    /*--liên kết trong. menu cách nhau 20px--*/
    .menu a{padding: 0px 10px 0px 10px;}
    /*--liên kết trong. middle_left chữ HOA nhỏ--*/
    .middle_left a{font-variant:small-caps;}
    /*--liên kết có chuột trong. middle_left in đậm--*/
    .middle_left a:hover{font-weight: bold;}
    /*--liên kết chung cho toàn trang--*/

```

```

a{text-decoration: none;}
a:link, a:active, a:visited{color: Blue;}
a:hover{color: Red;}
body{background-color: Gray;}

</STYLE>
</HEAD>
<body>
<div class="container">
    <div class="top"></div>
    <div class="menu">
        <a href="#1">Home</a> | <a href="#2">About Us</a> |
        <a href="#3">Contact Us</a> | <a href="#4">Feedback</a> |
        <a href="#5">FAQs</a>
    </div>
    <div class="middle">
        <div class="middle_left">
            <fieldset>
                <legend>Member Info</legend>
            </fieldset>
            <fieldset>
                <legend>Products</legend>
                <li><a href="#1">Nokia</a></li>
                <li><a href="#2">Samsung</a></li>
                <li><a href="#3">Sony Ericsson</a></li>
                <li><a href="#4">Motorola</a></li>
                <li><a href="#5">Apple</a></li>
                <li><a href="#5">Seamen</a></li>
            </fieldset>
            <fieldset>
                <legend>Online Support</legend>
            </fieldset>
        </div>
        <div class="middle_right"></div>
    </div>
    <div class="bottom"></div>
</div>
</body>
</HTML>
```

Các bạn lưu ý các điểm sau:

- ✓ Khoanh vùng liên kết cho 2 vùng khác nhau là. menu và. middle_left:

```

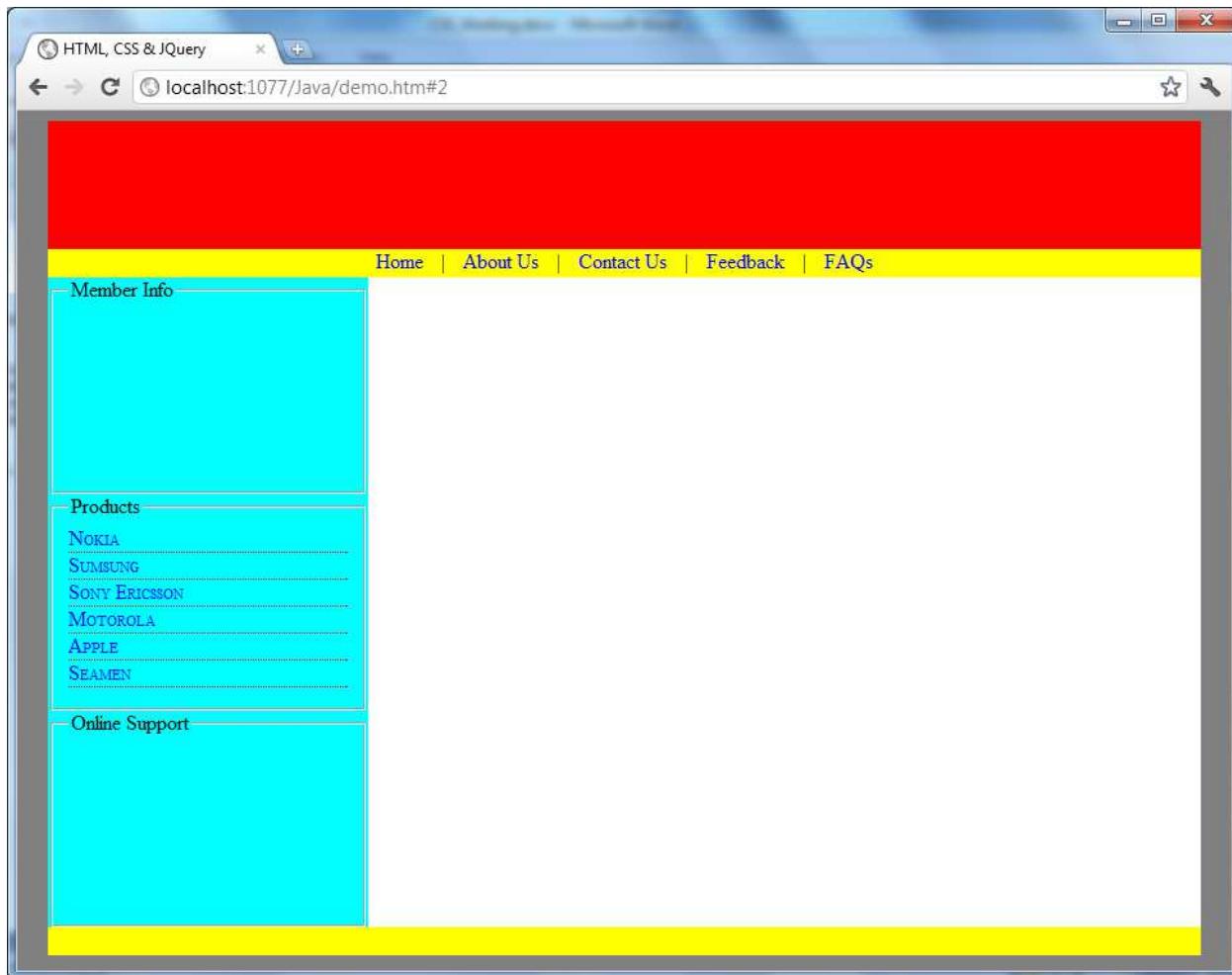
/*--liên kết trong. menu cách nhau 20px--*/
.menu a{padding: 0px 10px 0px 10px;}
/*--liên kết trong. middle_left chữ HOA nhỏ--*/
.middle_left a{font-variant:small-caps;}
/*--liên kết có chuột trong. middle_left in đậm--*/
.middle_left a:hover{font-weight: bold;}
```

Chúng ta cũng khoanh vùng cho các <fieldset> và đặt trong middle_left

```

/*--fieldset trong. middle_left cao tối thiểu 150--*/
.middle_left fieldset{min-height: 150px;}
/*--li trong. middle_left không dùng dấu, kẽ chân--*/
.middle_left li{list-style-type:none; border-bottom: 1px dotted red;}
```

Kết quả thực hiện:



4.2.6 Qui tắc nạp chồng

Nếu nhiều bộ chọn cùng được áp dụng cho cùng một thẻ thì phần khác nhau sẽ được gộp lại và phần giống nhau sẽ bị bộ chọn có độ ưu tiên cao hơn nạp chồng lên các bộ chọn có độ ưu tiên thấp hơn.

Thể loại và vị trí định nghĩa của các bộ chọn sẽ ảnh hưởng đến độ ưu tiên của chúng khi áp dụng lên một thẻ. Hãy ghi nhớ 2 qui tắc sau:

- ✓ Phân biệt theo thể loại:
 - **Nội tuyến** -> Bộ chọn **ID** -> Bộ chọn **Class** -> Bộ chọn **HTML** -> Mặc định
- ✓ Phân biệt theo vị trí định nghĩa:
 - **Nội tuyến** -> **Nhúng** -> **Liên kết ngoài**

Ví dụ sau thẻ <H1> chịu ảnh hưởng cả 3 bộ chọn khác nhau. Khi chạy sẽ cho dòng chữ "Nạp Chồng" màu vàng

```
<HTML>
<HEAD>
<TITLE>Job application</TITLE>
<style type="text/css">
H1{color: green;}
#xyz{color: yellow;}
. abc{color: Red;}
</style>
</HEAD>
```

```
<BODY>
    <H1 class="abc" id="xyz">Nạp chồng</H1>
</BODY>
</HTML>
```



4.3 JAVASCRIPT VÀ JQUERY

4.3.1 Giới thiệu

- Là ngôn ngữ lập trình dạng script thực thi trong browser.
- Javascript giúp trang web có tính tương tác : đổi màu 1 đối tượng khi đưa chuột vào, đổi nội dung của 1 tag, đưa ra các thông báo cần giao tiếp, phóng to hình.
- Trong trang web, mã lệnh javascript được đặt bên trong tag script.
- Mã lệnh javascript có thể đặt trực tiếp trong trang html hoặc có đặt trong 1 file riêng (*.js). Khi đó trang html muốn dùng code javascript thì link đến file js.
- Mỗi lệnh javascript kết thúc là dấu ;

4.3.2 Đưa javascript vào trang

4.3.2.1 Nhập trực tiếp code trong HTML

Là viết mã lệnh Javascript trực tiếp trong file HTML với tag script. Ví dụ:

```
<script> alert("Chào bạn"); </script>
```

4.3.2.2 Viết mã javascript trong file riêng

Là viết mã lệnh trong file. js nằm ngoài trang web. Sau đó nhúng file js vào trang web

a. Tạo file javascript:

- Nhấp menu File → New → Javascript → Create
- Gõ mã lệnh javascript. Ví dụ: gõ

```
<script>
    hoten=prompt("Bạn ơi bạn tên gì?");
    alert("Chào bạn " + hoten);
</script>
```

- Lưu file với tên mở rộng là. js

b. Liên kết file js đến trang html

- Mở 1 file html. Nhấp vị trí muốn chèn (thường trong tag head) rồi kéo thả file js vào

4.3.3 Jquery

- jQuery là một thư viện giúp viết code JavaScript dễ hơn.
- JQuery có nhiều hàm giúp định dạng, thay đổi nội dung trang web, tạo nhiều hiệu ứng như mở dần, chạy dọc chạy ngang, tạo request ajax v.v..

- jQuery cho phép tạo ra các Plugin.

4.3.3.1 Nhúng jquery vào trang web

- Vào trang chủ www.jquery.com và download phiên bản mới nhất (file js), chép vào folder website của bạn, rồi insert vào trang web bằng tag script.
- Hoặc chèn Jquery từ site chính thức của Jquery:

```
<html> <head>
<script type="text/javascript" src="http://code.jquery.com/jquery-latest.js">
</script>
<script type="text/javascript">
$(document).ready(function(){
    // Mã jquery của bạn
});
</script>
</head>
```

4.3.3.2 Sử dụng jQuery cơ bản

Lệnh đầu tiên trong jQuery mà bạn cần biết là lệnh theo dõi document ready (ready event), lệnh này theo dõi và đợi cho đến khi document sẵn sàng. Code Jquery thường đặt trong sự kiện này để chúng thực thi khi tài liệu sẵn sàng

```
<script type="text/javascript">
$(document).ready(function(){
    // Mã jquery của bạn
});
</script>
```

4.3.3.3 Chọn các phần tử HTML

Thao tác cơ bản của Jquery là chọn các phần tử trong tài liệu HTML và thực hiện một việc gì đó. Cú pháp như sau: **\$(query).action()**

\$ là kí hiệu đặc biệt, xác định đây là câu lệnh jQuery. **action** là hàm sẽ tác động lên các phần tử được chọn (click, change...). **query** là cách chọn phần tử.

```
$("#a")      chọn tất cả các tag <a>
$("div.intro")  chọn tất cả các tag <div> có class là "intro".
$("p#tieude")  chọn tất cả các tag <p> có id là "tieude".
$(this)      chọn phần tử hiện hành
$(".tensp")   chọn các phần tử có class là tensp
$("#left")     chọn phần tử có tên là left
```

4.3.3.4 Một số Jquery plugin

- Nivo Slider: <http://docs.dev7studios.com/jquery-plugins/nivo-slider>
- Slide tab: <http://lopatin.github.io/sliderTabs/>
- Lightbox: <http://www.jackmoore.com/colorbox/>
- Lightbox: <http://fancyboxapps.com/fancybox/>,
- Lightbox: <http://leandrovieira.com/projects/jquery/lightbox/>
- Lightbox: <http://lokeshdhakar.com/projects/lightbox2/>
- Tooltip: <http://craigsworks.com/projects/qtip/>,
- Tooltip: <http://vadikom.com/demos/poshytip/>

Chương 5: **TỔNG QUAN VỀ ỨNG DỤNG WEB**

Sau khi học xong bài này, học viên có khả năng :

- *Trình bày được kiến trúc ứng dụng WEB*
- *Mô tả được cú pháp ngôn ngữ HTML, DHTML và JavaScript*
- *Trình bày được các mô hình code cho Client side và Server side*
- *Cài đặt và cấu hình được ứng dụng ASP.NET trên Web Server*
 - *Ứng dụng thương mại điện tử và mô hình Client-Server*
 - *Ngôn ngữ HTML, DHTML và JavaScript*
 - *Phát triển ứng dụng ASP.NET với Web Forms*
 - *Cấu trúc trang ASP.NET và các mô hình code Client side & Server side*
 - *Giới thiệu về Web Server IIS 7*

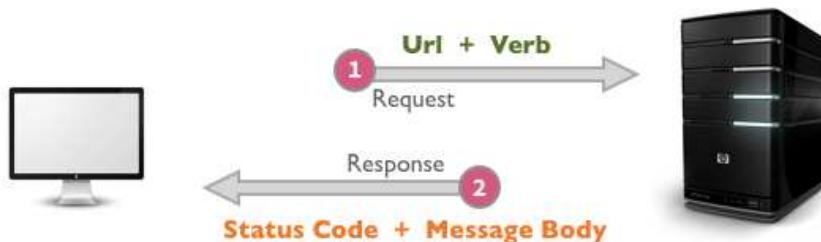
5.1 Giới thiệu Ứng dụng Web

Hàng ngày chúng ta mở máy tính, vào các trang web nổi tiếng như Google, Yahoo để tìm kiếm, đọc và gửi email. Chúng ta cũng thường vào các trang báo để đọc tin tức; các trang đào tạo để học hành; các trang bán hàng để tìm và mua hàng; tham gia vào diễn đàn để tranh luận. Tất cả rất tuyệt vời và thật sự có ý nghĩa với cuộc sống mỗi một con người trong thời đại internet của chúng ta.

Sự phát triển mạnh mẽ về công nghệ thông tin, đặc biệt là internet, nhiều lĩnh vực ngày nay như : thương mại, y tế, giáo dục..., nhu cầu trao đổi thông tin thực sự là cần thiết, giúp cho công việc được triển khai nhanh, chính xác, dễ dàng và tiết kiệm chi phí, thông tin được cập nhật kịp thời. Do đó vấn đề đặt ra là chúng ta cần phải có một ứng dụng cho phép trao đổi thông tin mọi lúc, mọi nơi, dễ sử dụng,... thông qua mạng. Ứng dụng Web đáp ứng được các yêu cầu đặt ra và sau đây là các lý do tại sao chúng ta phải sử dụng Web :

- ✓ Dễ dàng trao đổi và chia sẻ thông tin thông tin qua mạng.
- ✓ Sử dụng giao diện đồ họa giúp cho người dùng dễ sử dụng.
- ✓ Hỗ trợ về multimedia như : hình ảnh, âm thanh, phim ảnh,...
- ✓ Hỗ trợ nhiều chương trình(web-browser) để truy cập Web.
- ✓ Hỗ trợ truy cập web trên các thiết bị di động: Tablet, SmartPhone,...
- ✓ Hỗ trợ nhiều ngôn ngữ để phát triển Web: ASP, ASP.NET, JSP, PHP...

5.2 Nguyên lý hoạt động của ứng dụng Web



Để có được kết quả hiển thị của trang web yêu cầu, các bước thực hiện truyền thông xảy ra ở phía hậu cảnh bao gồm:

- ✓ Chuyển đổi “url” thành “ip”
- ✓ Gửi request đến Web Server
- ✓ Web Server thực hiện các xử lý cần thiết theo request
- ✓ Kết quả được response đến Browser
- ✓ Web Browser trình bày dữ liệu trên kết quả trả về và các thẻ markup

5.3 Các khái niệm

5.3.1 Web client (Browser)

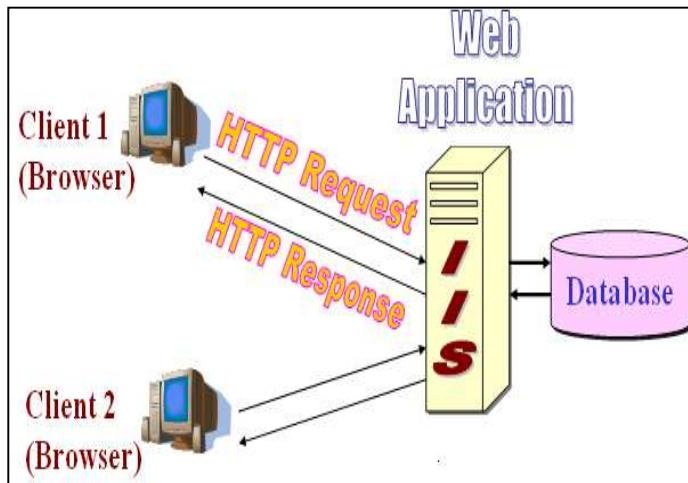
Máy khách (Client) thường là Web Browser sẽ sử dụng chương trình để truy cập đến các trang web gọi là trình duyệt web hay browser. Hiện rất nay có nhiều trình duyệt web như : Internet Explorer, Nescape, Mozilla FireFox, ...

5.3.2 Web server

Các máy chủ (Server) chứa các ứng dụng Web, sẵn sàng truy xuất các trang web hay các tài liệu và gửi về cho client khi nhận được yêu cầu từ phía Client. Hiện nay có rất nhiều Web server và chạy trên nhiều hệ thống như : Apache, Microsoft, Sun,...

5.3.3 Giao thức HTTP

Quá trình giao tiếp giữa client và server được thực hiện thông qua giao thức chuẩn HTTP(HyperText Transfer Protocol). Hình minh họa sau mô tả việc truy cập ứng dụng Web.



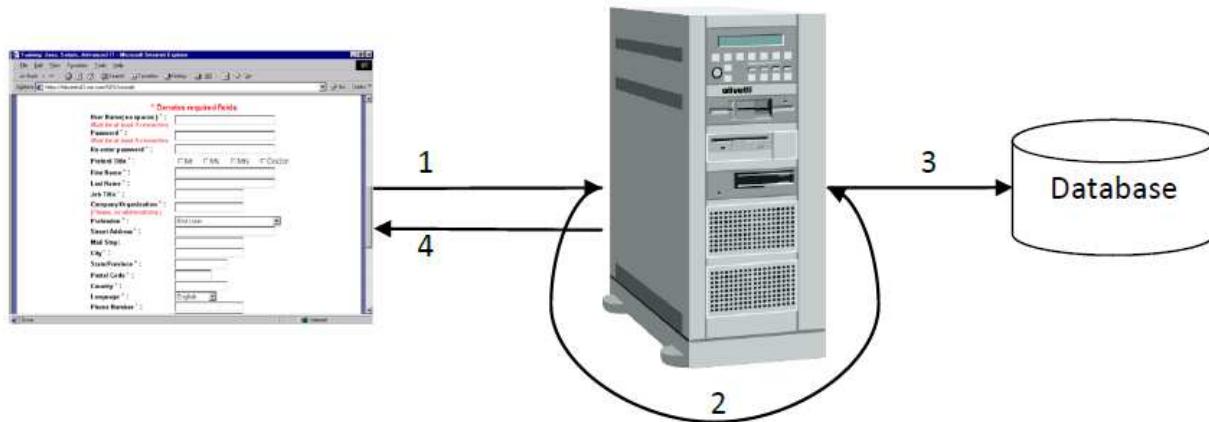
- ✓ Web được phát triển trên mô hình client-server.
- ✓ Giao thức HTTP: Quá trình giao tiếp giữa client và server được thực hiện thông qua giao thức chuẩn HTTP (HyperText Transfer Protocol).
- ✓ Mô hình gồm hai thành phần chính là: máy khách(client) và máy phục vụ(server). Máy phục vụ(server) sẽ chứa các ứng dụng Web và các ứng dụng Web này sẽ được quản lý tập trung bởi trình quản lý gọi là Web Server (IIS,...). Các máy khách(client) truy cập đến ứng dụng web sử dụng trình duyệt web(browser).
- ✓ Client sử dụng giao thức HTTP Request để gửi yêu cầu(trang web) lên Server, Server xử lý và sử dụng giao thức HTTP Response để gửi kết quả về cho Client.

5.3.4 Client Scripting và Server Scripting

Các ngôn ngữ dùng để viết mã cho trang web. Một trang web được xử lý ở Server và trả kết quả về cho Client. Do đó các ngôn ngữ viết mã cho trang web được chia thành hai dạng:

- ✓ **ClientScript:** được xử lý tại trình duyệt (Browser) trên máy Client. Các ngôn ngữ dùng để viết là :VBScript, JavaScript, DHTML...
 - JavaScript là ngôn ngữ phổ biến sử dụng nhiều nhất hiện nay. JavaScript được dùng để kiểm tra việc nhập liệu, kiểm tra trình duyệt,...
 - DHTML: là sự kết hợp của HTML, Style Sheet (CSS) và JavaScript nhằm làm cho trang web dễ tương tác, điều khiển và giảm bớt việc xử lý phía Server.
 - VBScript là ngôn ngữ script của Microsoft. Chức năng của VBScript cũng giống như JavaScript.
- ✓ **Server Scripting:** được xử lý tại Web server trên máy Server. Các ngôn ngữ dùng để viết là : ASP, ASP.NET, PHP, JSP,... Trong giáo trình này chúng ta sẽ khảo sát ngôn ngữ ASP.NET.

5.4 Kiến trúc công nghệ ứng dụng web



Trình duyệt giúp người sử dụng giao tiếp với ứng dụng web cài đặt phía server. Phần mềm trung gian (ứng dụng web) này sẽ nhận và xử lý các yêu cầu của người sử dụng. Nếu ứng dụng cần truy vấn hay lưu trữ thông tin, nó sẽ kết nối với CSDL để được trợ giúp bởi các hệ quản trị CSDL.

1. Thông tin trên form chuyển đến Web Server thông qua request
2. Server thực hiện các xử lý được cài đặt (server script – Server Side)
3. Kết nối và thao tác CSDL
4. Kết quả xử lý được trả về qua response (HTML)

5.5 Giới thiệu về ASP & ASP.NET

5.5.1 Giới thiệu về ASP

Active Server Page (ASP) do Microsoft phát triển là môi trường lập trình phía server (server side scripting) hỗ trợ mạnh trong việc xây dựng các ứng dụng thương mại điện tử (các trang Web động). Các ứng dụng ASP rất dễ viết và dễ sửa đổi, đồng thời tích hợp các công nghệ sẵn có của Microsoft như : COM,...

Từ khoảng cuối thập niên 90, ASP (Active Server Page) đã được nhiều lập trình viên lựa chọn để xây dựng và phát triển ứng dụng web động trên máy chủ sử dụng hệ điều hành Windows. ASP đã thể hiện được những ưu điểm của mình với mô hình lập trình thủ tục đơn giản, sử dụng hiệu quả các đối tượng COM: ADO (ActiveX Data Object) - xử lý dữ liệu, FSO (File System Object) - làm việc với hệ thống tập tin... Đồng thời, ASP cũng hỗ trợ nhiều ngôn ngữ: VBScript, JavaScript. Chính những ưu điểm đó, ASP đã được yêu thích trong một thời gian dài.

Một ứng dụng ASP được triển khai trên Web Server là IIS (Internet Information Service) có sẵn trong môi trường Windows. Để có thể triển khai ứng dụng ASP trên các môi trường khác ta phải cài đặt các thư viện hỗ trợ ASP.

❖ Đặc điểm của trang ASP

- ✓ Là một tập tin văn bản (text file) có phần mở rộng .asp. Phần mở rộng này sẽ giúp Web server yêu cầu trình xử lý trang asp (ASP engine) trước khi trả về cho trình duyệt.
- ✓ Ngôn ngữ script thông dụng nhất để viết mã của ASP là VBScript. Ngoài ra ta cũng có thể viết mã bằng các ngôn ngữ khác như: JavaScript, Perl, Python,...nếu trên Web server có cài đặt các bộ xử lý ngôn ngữ này.
- ✓ Các đoạn mã viết trong trang ASP sẽ được các bộ xử lý ngôn ngữ trên Web server xử lý tuân tự từ trên xuống dưới. Kết quả của việc xử lý này là trả về trang mã HTML cho web server và web server sẽ gửi trang HTML này về cho trình duyệt (Browser). Do đó tại trình duyệt không thể thấy được các đoạn mã chương trình đã viết trong trang ASP.
- ✓ Một trang ASP gồm 4 phần :
 - Dữ liệu văn bản(text)
 - Các thẻ (tag) HTML.
 - Các đoạn mã chương trình phía client đặt trong cặp thẻ <SCRIPT> và </SCRIPT>.
 - Mã chương trình ASP được đặt trong cặp thẻ <% và %>.

3 thành phần đầu tiên là cấu trúc của một trang HTML thông thường, do đó có thể xem một trang ASP là một trang HTML được nhúng thêm phần xử lý viết bằng mã ASP (VBScript, JavaScript...).

Ví dụ: Minh họa trang ASP.

```
<% @Language=VBScript %>
<HTML>
<BODY>
<CENTER>
<I>
<FONT COLOR="HOTPINK" size = 5>
<P> <B>Welcome to my website</B>. Today is
<%
Response.Write Date()
%>
</FONT>
</I>
</CENTER>
</BODY>
</HTML>
```

❖ Ưu điểm

- ✓ Trang ASP được diễn dịch một cách tự động.

- ✓ Xây dựng ứng dụng Web động.
- ✓ Xử lý dữ liệu động hiệu quả.

❖ **Nhược điểm:**

- ✓ Chỉ sử dụng hai ngôn ngữ kịch bản phi định kiểu(non-type): VBScript và JavaScript.
- ✓ Sử dụng trình thông dịch cho các trang ASP.
- ✓ Các đoạn mã lệnh và giao diện (HTML) trộn lẫn với nhau.
- ✓ Không sử dụng lại được (reuse) các đoạn mã.
- ✓ Không hỗ trợ cơ chế bẫy lỗi (Debug).

5.5.2 Giới thiệu về ASP.NET

ASP vẫn còn tồn đọng một số khó khăn như Code ASP và HTML lẫn lộn, do không được biên dịch trước (dễ bị mất source code và hạn chế về mặt tốc độ thực hiện), không có hỗ trợ cache, quá trình xử lý Postback khó khăn, ...

Đầu năm 2002, Microsoft giới thiệu một kỹ thuật lập trình Web khá mới mẻ với tên gọi ban đầu là ASP+, tên chính thức sau này là ASP.Net. Với ASP.Net, không những không cần đòi hỏi bạn phải biết các tag HTML, thiết kế web, mà nó còn hỗ trợ mạnh lập trình hướng đối tượng trong quá trình xây dựng và phát triển ứng dụng Web. ASP.Net là kỹ thuật lập trình và phát triển ứng dụng web ở phía Server (Server-side) dựa trên nền tảng của Microsoft. Net Framework.

Hầu hết, những người mới đến với lập trình web đều bắt đầu tìm hiểu những kỹ thuật ở phía Client (Client-side) như: HTML, Java Script, CSS (Cascading Style Sheets). Khi Web browser yêu cầu một trang web (trang web sử dụng kỹ thuật client-side), Web server tìm trang web mà Client yêu cầu, sau đó gửi về cho Client. Client nhận kết quả trả về từ Server và hiển thị lên màn hình.

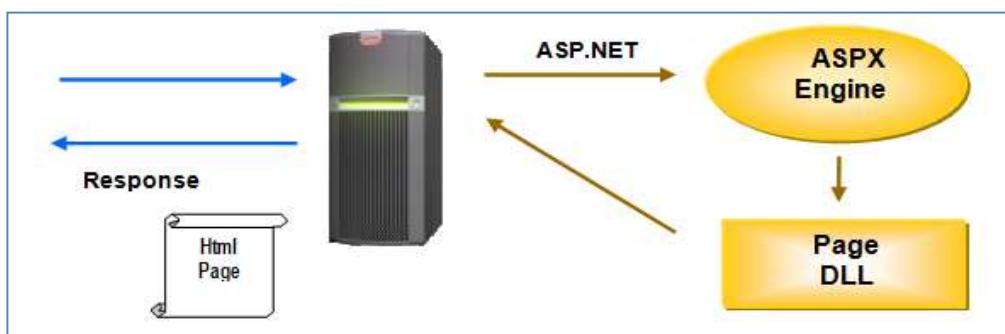
ASP.Net sử dụng kỹ thuật lập trình ở phía server thì hoàn toàn khác, mã lệnh ở phía server sẽ được biên dịch và thi hành tại Web Server. Sau khi được Server đọc, biên dịch và thi hành, kết quả tự động được chuyển sang HTML/JavaScript/CSS và trả về cho Client. Tất cả các xử lý lệnh ASP.Net đều được thực hiện tại Server và do đó, gọi là kỹ thuật lập trình ở phía server.

ASP.Net được Microsoft phát triển qua nhiều phiên bản từ ASP.Net 1.0, 1.1, 2.0, 4.0 và gần đây nhất là phiên bản ASP.NET 5.0 chạy trên. Net Framework 4.5 sử dụng môi trường phát triển tích hợp (IDE) Visual Studio.NET 2012. Trong giáo trình này chúng sử dụng ASP.NET 4.0.

❖ **Các ưu điểm của ASP.Net**

- ✓ ASP chỉ sử dụng VBScript và JavaScript mà không sử dụng được các ngôn ngữ mạnh khác: Visual Basic, C++... Trong khi đó ASP.NET cho phép viết nhiều ngôn ngữ : VBScript, JavaScript, C#, Visual Basic.Net,...
- ✓ ASP.Net sử dụng phong cách lập trình mới: Code behide – Tách code riêng, giao diện riêng. Dễ đọc, dễ quản lý và bảo trì.
- ✓ Trong các trang ASP chúng ta phải viết mã để kiểm tra dữ liệu nhập từ người dùng, ASP.NET hỗ trợ các validation controls để kiểm tra chúng ta không cần viết mã,...
- ✓ Hỗ trợ phát triển Web được truy cập trên các thiết bị di động: PocketPC, Smartphone...
- ✓ Hỗ trợ nhiều web server control.
- ✓ Hỗ trợ thiết kế và xây dựng MasterPage lồng nhau.
- ✓ Hỗ trợ bẫy lỗi (debug) JavaScript.

- ✓ Cho phép người dùng thiết lập giao diện trang Web theo sở thích cá nhân sử dụng Theme, Profile, WebPart.
- ✓ Tăng cường các tính năng bảo mật (security).
- ✓ Hỗ trợ kỹ thuật truy cập dữ liệu mới LINQ.
- ✓ Hỗ trợ kỹ thuật xây dựng các ứng dụng đa phương tiện SilverLight.
- ✓ Hỗ trợ kỹ thuật bắt đồng bộ ASP.Net Ajax.
- ✓ ASP.Net hỗ trợ mạnh mẽ bộ thư viện phong phú và đa dạng của .Net Framework, làm việc với XML, Web Service, truy cập cơ sở dữ liệu qua ADO.Net, ...
- ✓ ASPX và ASP có thể cùng hoạt động trong 1 ứng dụng.
- ✓ Kiến trúc lập trình giống ứng dụng trên Windows.
- ✓ Hỗ trợ quản lý trạng thái của các control.
- ✓ Tự động phát sinh mã HTML cho các Server control tương ứng với từng loại Browser.
- ✓ Hỗ trợ nhiều cơ chế Cache.
- ✓ Triển khai cài đặt : Không cần lock, không cần đăng ký DLL, cho phép nhiều hình thức cấu hình ứng dụng.
- ✓ Hỗ trợ quản lý ứng dụng ở mức toàn cục : Global.aspx có nhiều sự kiện hơn, quản lý session trên nhiều Server, không cần Cookies.
- ✓ Trang ASP.Net được biên dịch trước. Thay vì phải đọc và thông dịch mỗi khi trang web được yêu cầu, ASP.Net biên dịch những trang web động thành những tập tin DLL mà Server có thể thi hành nhanh chóng và hiệu quả. Yếu tố này làm gia tăng tốc độ thực thi so với kỹ thuật thông dịch của ASP.



Ví dụ: Minh họa trang ASP.NET hiển thị ngày hiện hành.

Mã ASP.NET

```

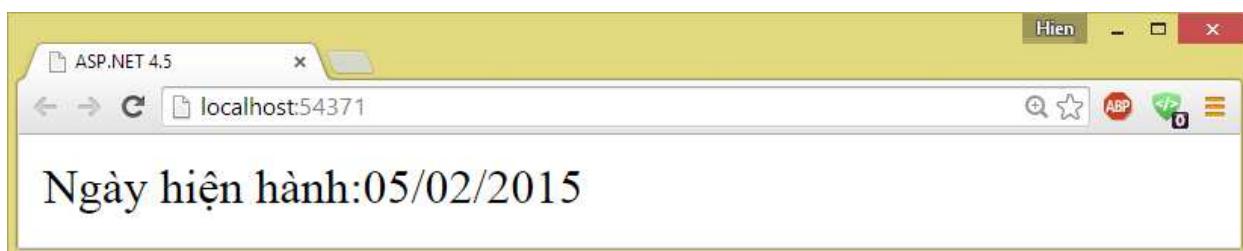
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="_Default" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
<title>ASP.NET 4.5</title>
</head>
<body>
<form id="form1" runat="server">
<div>
<asp:Label ID="lbMsg" runat="server" Text="Ngày hiện hành:"></asp:Label>
</div>
</form>
</body>
</html>
    
```

Mã C# Code behind

```
using System;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;

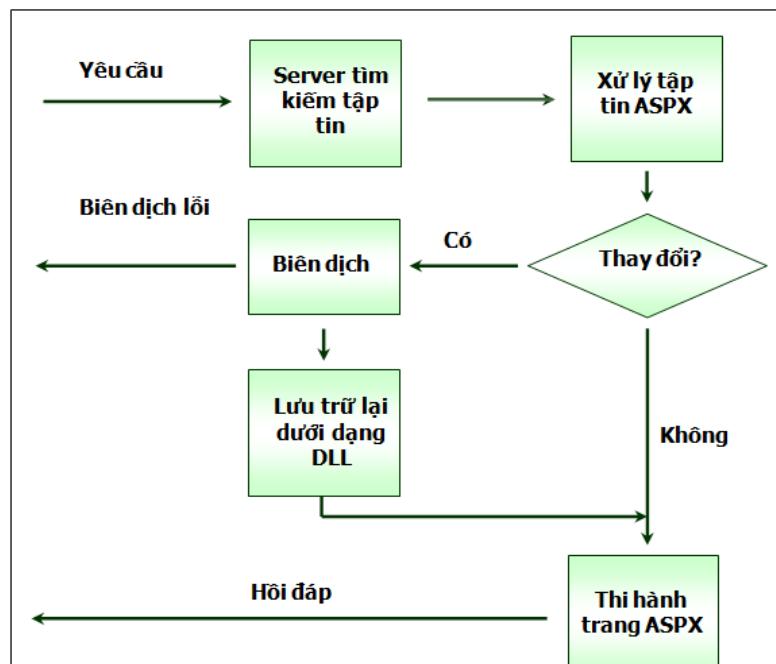
public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        lbMsg.Text = lbMsg.Text + DateTime.Now.ToString("dd/MM/yyyy");
    }
}
```

Kết quả thực hiện



❖ Quá trình xử lý tập tin ASPX

Khi Web Server nhận được yêu cầu từ phía client, nó sẽ tìm kiếm tập tin được yêu cầu thông qua chuỗi URL được gởi về, sau đó, tiến hành xử lý theo sơ đồ sau:

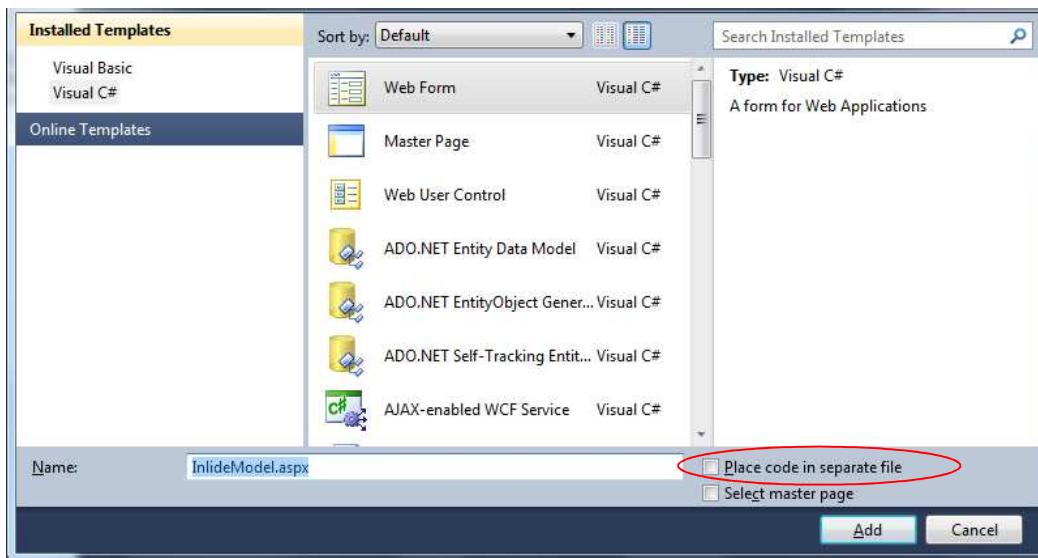


Chương 6: WEB SERVER CONTROL

6.1 Mô hình Tổ chức mã

Mô hình inline: trong mô hình này, phần mã ASP.NET và mã HTML được viết cùng một trang. Mã ASP.NET được viết ở phần `<script runat="server">....</script>` nằm trong trang ASP.NET nhưng không trộn lẫn với mã HTML dành cho phần nội dung (content section).

Chẳng hạn trang **InlineModel.aspx** có phần mã ASP.NET cùng với phần mã HTML.



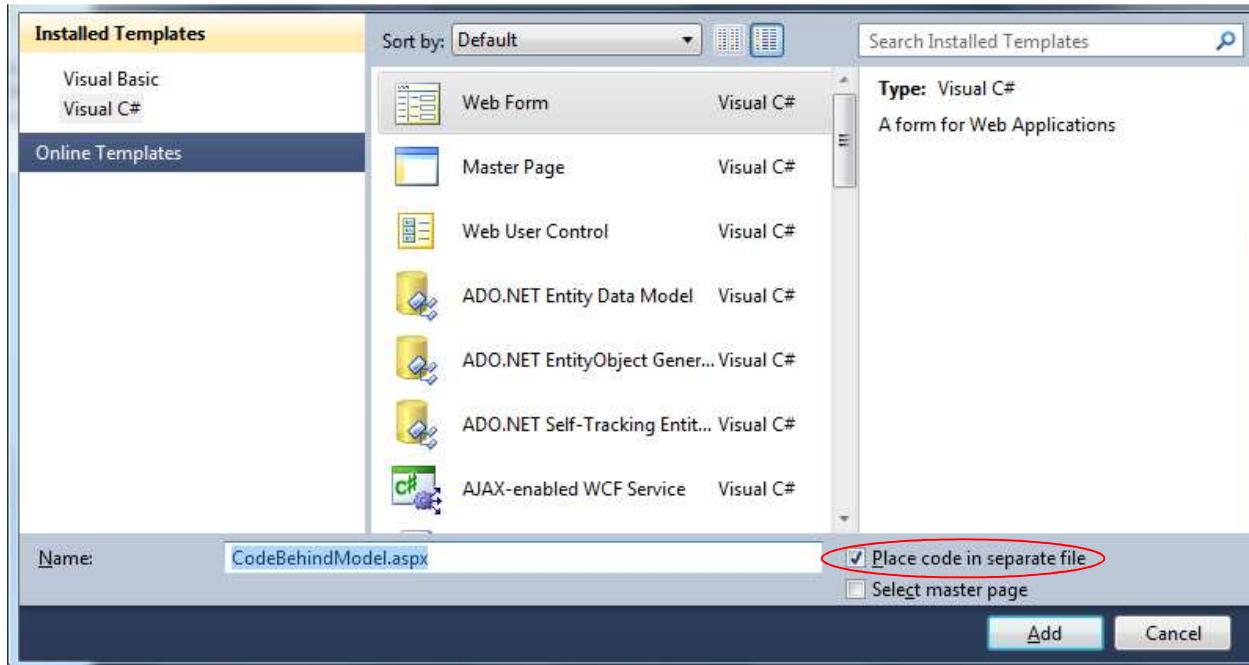
```
<%@ Page Language="C#" %>

<%-- Phần mã ASP.NET(code section) --%>
<script runat="server">
void Page_Load(object sender, EventArgs e)
{
    lblChao.Text = "Lập trình Web với ASP.Net";
}
</script>
```

```
<%-- Phần mã HTML(content section) --%>
<html>
<head id="Head1" runat="server">
<title>Chao mung</title>
</head>
<body>
<form id="form2" runat="server">
<div>
<asp:Label ID="lblChao" runat="server"/>
</div>
</form>
</body>
</html>
```



Mô hình code behind: trong mô hình này, phần mã ASP.NET được tách biệt trong một tập tin riêng biệt với phần mã HTML.



Khi đó chúng ta sẽ có 2 trang được tạo ra: **CodeBehindModel.aspx** (chứa mã ASP.NET) và **CodeBehindModel.aspx.cs** (chứa mã C#).

Trang ASP.NET - CodeBehindModel.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="CodeBehindModel.aspx.cs"
Inherits="CodeBehindModel" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Chao Mung</title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Label ID="lblChao" runat="server"/>
        </div>
    </form>
</body>
</html>
```

Code Behind - CodeBehindModel.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class CodeBehindModel : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        lblChao.Text = "Lập trình Web với ASP.Net";
    }
}
```



So với CodeInline thì Code Behind viết mã lệnh dễ hơn do tách được phần giao diện và phần mã HTML riêng biệt đồng thời có thể sử dụng lại các đoạn mã đã viết trong tập tin **.aspx.cs**.

6.2 Sự kiện của trang ASP.NET

Có nhiều sự kiện xảy ra trong suốt quá trình thực hiện một trang ASP.NET. Chúng ta cần phân biệt rõ thứ tự thực hiện để cài đặt mã nguồn một cách chính xác cho các đoạn mã chạy tại những thời điểm khác nhau phù hợp với yêu cầu logic của nghiệp vụ ứng dụng. Chẳng hạn, công tác bảo mật nên chạy trước tiên, công tác hiển thị nên chạy sau cùng, các thao tác nghiệp vụ khác nên chạy vào các thời điểm phù hợp.

Ví dụ: Thao tác bỏ hàng vào giỏ phải xảy ra trước khi thông kê thông tin hàng hóa trên giỏ hàng.

6.2.1 Các sự kiện

Sự kiện Init: xảy ra ở 3 thời điểm khác nhau của giai đoạn khởi tạo

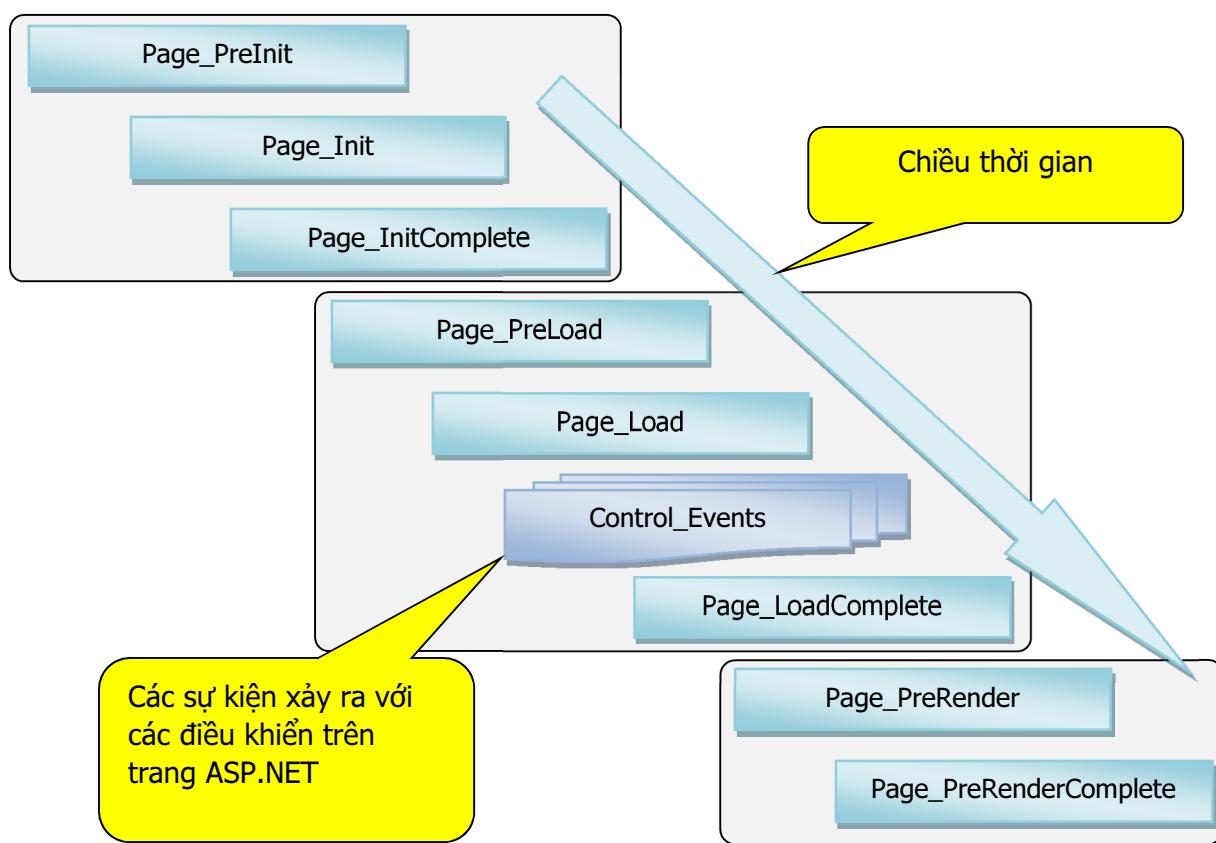
- ✓ **Page_PreInit**(object sender, EventArgs e): viết mã điều khiển sự kiện xảy ra trước khi trang web được khởi tạo (ví dụ như kiểm soát vai trò của người dùng).
- ✓ **Page_Init**(object sender, EventArgs e): viết mã điều khiển sự kiện xảy ra ngay sau khi trang web được khởi tạo.
- ✓ **Page_InitComplete**(object sender, EventArgs e): viết mã điều khiển sự kiện xảy ra sau khi tất cả Page_Init trên của các controls trên trang web đã được khởi tạo.

Sự kiện Load: xảy ra ở 3 thời điểm khác nhau của giai đoạn tải trang ASP.NET

- ✓ **Page_Preload**(object sender, EventArgs e): viết mã điều khiển sự kiện trước khi trang web được tải để thực thi.
- ✓ **Page_Load**(object sender, EventArgs e): viết mã điều khiển sự kiện ngay sau khi trang web đã tải đầy đủ. Lúc này bạn đã có thể làm việc với các server controls.
- ✓ **Page_LoadComplete**(object sender, EventArgs e): viết mã điều khiển sự kiện sau khi tất cả các Page_Load trên các control đã xảy ra hoàn toàn.

Sự kiện PreRender: xảy ra ở 2 thời điểm khác nhau của giai đoạn sinh mã HTML

- ✓ **Page_PreRender**(object sender, EventArgs e): viết mã điều khiển sự kiện ngay trước khi trang web sinh mã HTML. Đây là lúc thích hợp nhất để cấp dữ liệu cho các control hiển thị lên giao diện của trang.
- ✓ **Page_PreRenderComplete**(object sender, EventArgs e): viết mã điều khiển sự kiện sau khi kết thúc hoàn toàn việc sinh mã HTML của ASP.NET và các control trên nó.



6.2.2 Ví dụ sự kiện trang

Xem xét ví dụ sau để hiểu hơn về hoạt động của các sự kiện:

Mã ASP.NET

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

```
<!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></title>
</head>
<body>
<form id="form1" runat="server">
<div>
<asp:Button ID="Button1" runat="server" Text="Button" onclick="Button1_Click" />
</div>
</form>
</body>
</html>
```

Mã C#

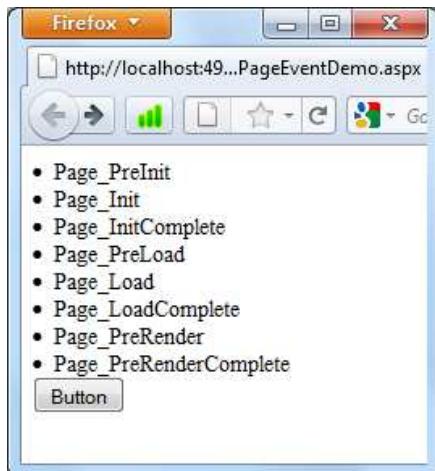
```
using System;
using System.Collections.Generic;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
```

```

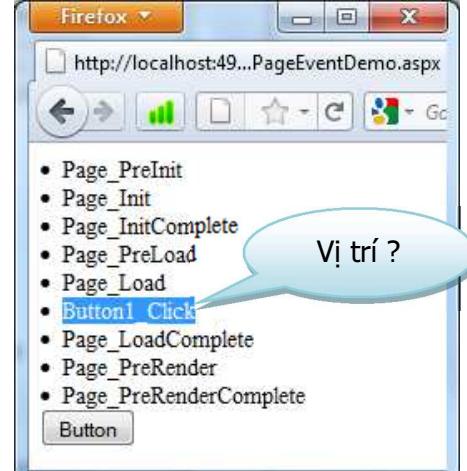
public partial class PageEventDemo : System.Web.UI.Page
{
    protected void Page_PreInit(object sender, EventArgs e)
    {
        Response.Write("<li>Page_PreInit</li>");
    }
    protected void Page_Init(object sender, EventArgs e)
    {
        Response.Write("<li>Page_Init</li>");
    }
    protected void Page_InitComplete(object sender, EventArgs e)
    {
        Response.Write("<li>Page_InitComplete</li>");
    }
    protected void Page_PreLoad(object sender, EventArgs e)
    {
        Response.Write("<li>Page_PreLoad</li>");
    }
    protected void Page_Load(object sender, EventArgs e)
    {
        Response.Write("<li>Page_Load</li>");
    }
    protected void Page_LoadComplete(object sender, EventArgs e)
    {
        Response.Write("<li>Page_LoadComplete</li>");
    }
    protected void Page_PreRender(object sender, EventArgs e)
    {
        Response.Write("<li>Page_PreRender</li>");
    }
    protected void Page_PreRenderComplete(object sender, EventArgs e)
    {
        Response.Write("<li>Page_PreRenderComplete</li>");
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        Response.Write("<li>Button1_Click</li>");
    }
}

```

Kết quả hiển thị



Chưa click nút Button



Đã click nút Button

Hãy chú ý vào **vị trí** xảy ra của các sự kiện trang và sự kiện của nút Button.

Chú ý quan trọng: trên các web user control (sẽ được nghiên cứu sau) chỉ có 3 sự kiện trang: Page_Init, Page_Load và Page_PreRender và đều **xảy ra sau** sự kiện tương ứng của trang ASP.NET chứa nó. Nắm rõ qui luật này để việc chia sẻ dữ liệu giữa trang web và web user control trên nó được tốt hơn.

6.3 Cơ chế PostBack

6.3.1 ViewState:

Khi bạn xem mã nguồn trang trên các trang viết bằng ASP.NET thì bạn sẽ thấy có một trường ẩn có tên là **_VIEWSTATE** chứa dữ liệu trạng thái của trang web đang xem.



```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>
    ASP.NET 4.5
</title></head>
<body>
    <form method="post" action="Default.aspx" id="form1">
<div class="aspNetHidden">
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
value="ydb1+4F2aDmXRd4eLgMx01XN12htra4iEs5Lx1QMrZAr8eKKVi5FPVeCC0wb1Ge1Tyc8Y
KvFbEx0QCZyiNhQ3SiGznvmb4pWRSG7dSSY22Kfzu1Ey7PhFjmgBPGlwugnxW9dHVQmDuITIXCF
VsLUA3wyoCIbO6EhyMGqwu9PY8=" />
</div>

```

Thực ra giá trị của thuộc tính value là thông tin của các thành phần trên trang web đã được mã hóa và lưu giữ lại ở đây. Thông tin này sẽ được chuyển ngược về server để phục hồi trạng thái cho các thành phần trên trang web một cách tự động (khi bạn tương tác lên các nút nhấn) mà người lập trình không cần phải quan tâm.

Khi lập trình, chúng ta có thể lợi dụng cơ chế này để duy trì và chia sẻ các giá trị giữa các lần tương tác trên cùng một trang bằng các đọc/ghi trạng thái trang ViewState["tên"] [= "giá trị"].

Ví dụ: lưu lời chào để chia sẻ giữa các lần tương tác của trang

- ✓ Lưu lời chào (ghi vào trạng thái trang):

```
ViewState["chao"] = "Chào bạn";
```

- ✓ Lấy lời chào (đọc từ trạng thái trang):

```
String chao = ViewState["chao"]
```

6.3.2 Thuộc tính IsPostBack:

Như đã trình bày trên đây là thông tin trạng thái trang sẽ được gửi ngược trở lại server để phục hồi lại thông tin cho các thành phần trước đó của trang một cách tự động khi người dùng tương tác trang lên (nhấp nút). Tuy nhiên khi gọi trang ASP.NET chạy lần đầu tiên (hay theo phương thức GET) thì chẳng có thông tin trạng thái nào được tạo ra trước đó vì vậy sẽ không có thông

tin trạng thái được gửi ngược về server. Để phân biệt 2 trường hợp này, trang ASP.NET cung cấp cho chúng ta 1 thuộc tính là **IsPostBack**. Nếu thuộc tính này có giá trị true thì có nghĩa là người dùng đang “đẩy ngược” trở lại trang này. Một số tình huống sau sẽ tạo ra “đẩy ngược”.

- ✓ Nhấp nút: **Button, LinkButton, ImageButton**
- ✓ Chọn mục trên ListControl (**DropDownList, ListBox, CheckBoxList, RadioButtonList**) với thuộc tính AutoPostBack được thiết lập là true.
- ✓ Các sự kiện xảy ra với các DataControl: **SelectedIndexChanged, ItemDataBound,...**

Ví dụ: Cung cấp dữ liệu cho DropDownList lần đầu tiên, sau đó nhấp nút Button thì DropDownList sẽ không được cấp lại mà nhờ vào sự phục hồi trạng thái trang tự động của ASP.NET.



Mã nguồn ASP.NET

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Chao Mung</title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:DropDownList ID="DropDownList1" runat="server">
            </asp:DropDownList>
            <asp:Button ID="Button1" runat="server" Text="Button" />
        </div>
    </form>
</body>
</html>
```

Mã nguồn code behind (C#)

```
using System;
using System.Collections.Generic;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
```

```
public partial class PostBackDemo : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            DropDownList1.Items.Add("Vietnam");
            DropDownList1.Items.Add("United States");
        }
    }
}
```

Trong ví dụ trên nếu chúng ta không dùng lệnh if(!**IsPostBack**) thì sau mỗi lần nhấp nút thì sẽ có 2 mục nữa được thêm vào trong DropDownList.

6.3.3 Thuộc tính MaintainScrollPositionOnPostBack

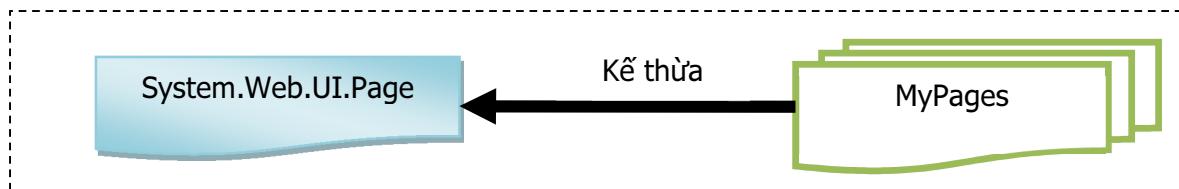
Trong trường hợp trang web bị “**dẩy ngược**” và nội dung của trang web vượt quá kích thước hiển thị của màn hình và bạn đang đọc ở phần giữa của trang web màn hình sẽ hiển thị phần đầu của trang web. Nếu giá trị của thuộc tính này là **true**, trình duyệt web sẽ vẫn **giữ nguyên vị trí** mà bạn đang đọc sau khi tải lại. Giá trị mặc định là **false**.

6.4 Kế thừa để mở rộng

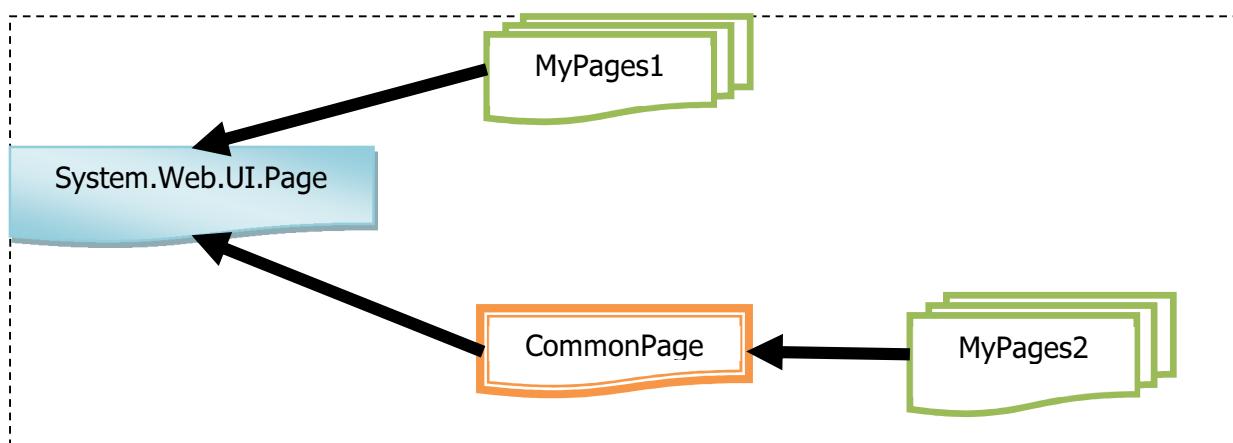
Theo mô hình lập trình hướng đối tượng, chúng ta có thể mở rộng lớp bằng cách kế thừa một lớp đã tồn tại và viết bổ sung các phương thức mới. Trên quan điểm đó, chúng ta thấy trang web ASP.NET là một lớp được kế thừa từ lớp System.Web.UI.Page đã được Microsoft viết ra trước đó trong .Net framework.

```
public partial class PostBackDemo : System.Web.UI.Page
{
    ...
}
```

Có thể hình dung mô hình thừa kế các trang trong website của chúng ta như hình sau:

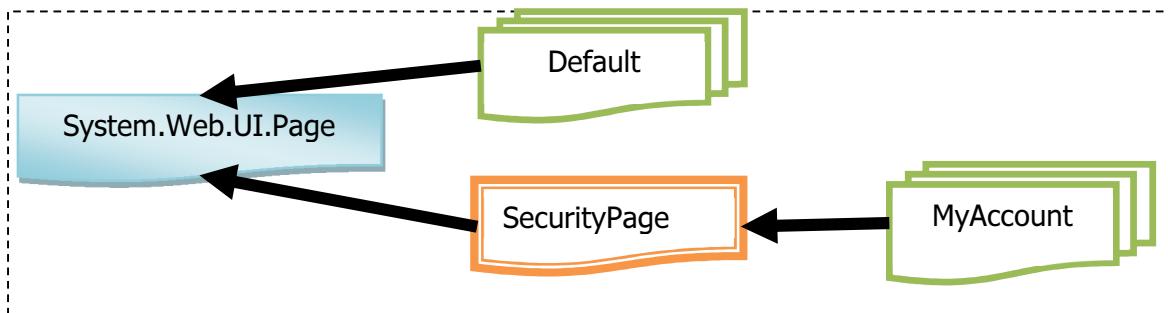


Mở rộng mô hình này để tận hưởng lợi ích mà lập trình hướng đối tượng mang lại cho chúng ta.



Nếu tổ chức được như mô hình này, bạn sẽ tận dụng được các chức năng dùng chung viết ở lớp CommonPage cho tất cả các trang web được viết ở nhóm 2 (MyPages2), còn các trang thuộc nhóm 1 (MyPages1) vẫn được viết như bình thường.

Ví dụ sau cho phép kiểm soát các trang kế thừa từ trang SecurityPage.



Mã nguồn sẽ được cài đặt như sau:

Trang SecurityPage.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

public class SecurityPage: System.Web.UI.Page
{
    protected void Page_PreInit(object sender, EventArgs e)
    {
        if (Session["user"] == null)
        {
            Response.Redirect("Login.aspx");
        }
    }
}
  
```

Trang MyAccount.aspx.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

public class MyAccount: SecurityPage
{
    ...
}
  
```

Trang Default.aspx.cs

```

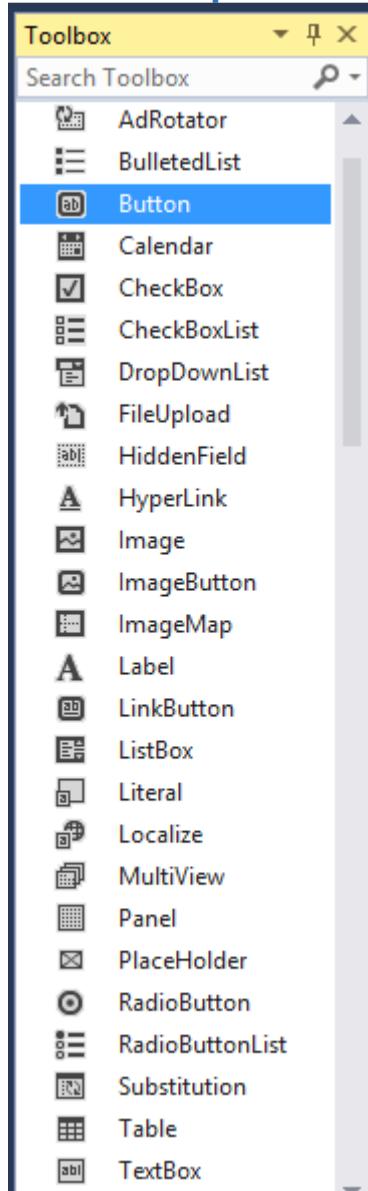
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

public class MyAccount: System.Web.UI.Page
{
    ...
}
  
```

Nếu bạn chạy trang Default.aspx thì vẫn hoạt động bình thường. Còn nếu bạn chạy trang MyAccount.aspx mà trang session chưa có biến user thì nó sẽ tự chuyển về trang Login.aspx.

6.5 ASP.NET Web Server Control

6.5.1 Giới thiệu



Web Server Controls là các thành phần web chạy phía server và sinh mã HTML cho trang web. Hình bên trái là thanh Toolbox chứa các server control trong Visual Studio dùng để kéo vào các trang web ASP.NET trong quá trình phát triển.

Làm việc với ASP.NET, lập trình viên chỉ cần thao tác các thuộc tính, gọi các phương thức... mà có thể không quan tâm đến mã HTML được sinh ra như thế nào. Dựa vào trạng thái hiện tại của các control ASP.NET sẽ phát sinh mã HTML sau đó gửi về client một cách tự động.

Dưới đây là các lý do bạn nên sử dụng ASP.Net Web Control:

- ✓ Đơn giản, tương tự như các điều khiển trên Windows Form.
- ✓ Đồng nhất: Các điều khiển Web server có các thuộc tính giống nhau -> dễ tìm hiểu và sử dụng.
- ✓ Hiệu quả: Các điều khiển Web Server tự động phát sinh ra các tag HTML theo từng loại Browser.

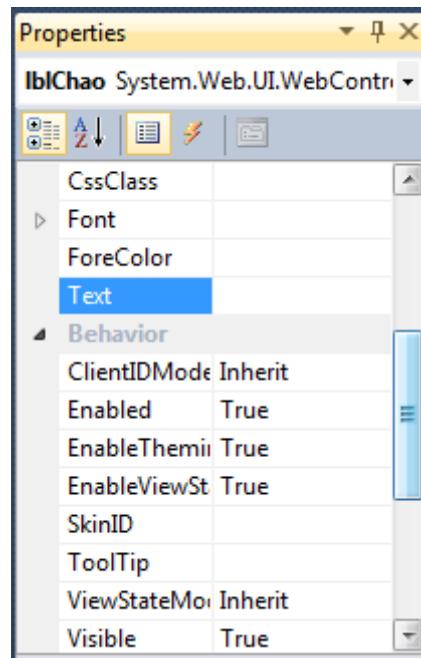
Các controls ASP.Net là các lớp, do đó ta có thể mở rộng các chức năng của các control.

Mỗi control có rất nhiều thuộc tính khác nhau. Các thuộc tính này bạn có thể thay đổi lúc thiết kế bằng cách nhập vào thanh thuộc tính (Properties - Hình bên phải) của nó hoặc có thể thay đổi bằng cách lập trình.

IblChao.Text = "Chào các bạn";

Ở đây **IblChao** là ID của server control còn **Text** là tên thuộc tính.

Dù mỗi control có rất nhiều thuộc tính, song tất cả chúng có chung một số thuộc tính. Sau đây là danh sách các thuộc tính thường dùng của tất cả các control.



Bảng liệt kê các thuộc tính chung của Web Control:

Thuộc tính	Kiểu	Ý nghĩa
(ID)	String	Tên của điều khiển (duy nhất)
AccessKey	String	Quy định ký tự để di chuyển nhanh đến vị trí điều khiển – ký tự xử lý phím nóng
Attributes	AttributeCollection	Tập hợp các thuộc tính của điều khiển HTML
BackColor	Color	Quy định màu nền của điều khiển
BorderColor	Color	Quy định màu đường viền của điều khiển
BorderStyle	BorderStyle	Quy định kiểu đường viền của điều khiển
BorderWidth	Unit	Quy định độ rộng của đường viền
CssClass	String	Quy định hình thức hiển thị thông qua tên class CSS
Enabled	Boolean	Quy định điều khiển có được hiển thị hay không, mặc định là True – được phép hiển thị.
Font	FontInfo	Quy định Font hiển thị cho điều khiển
ForeColor	Color	Quy định màu chữ hiển thị trên điều khiển
Height	Unit	Quy định chiều cao của điều khiển
ToolTip	String	Dòng chữ sẽ hiện khi rê chuột vào điều khiển
Width	Unit	Quy định độ rộng của điều khiển

6.5.2 Các điều khiển cơ bản

6.5.2.1 Label

```
<asp:Label ID="IblChao" runat="server" Text="Chào"></asp:Label>
```

Label thường được sử dụng để hiển thị và trình bày nội dung trên trang web. Nội dung được hiển thị trong label được xác định thông qua thuộc tính Text. Thuộc tính Text có thể nhận và hiển thị nội dung với các tag HTML.

Ví dụ:

```
IblChao.Text = "Đây là chuỗi văn bản thường";
```

6.5.2.2 HyperLink

<asp:HyperLink NavigateUrl="hi.aspx" ID="InkHi" runat="server">Hello</asp:HyperLink>

Điều khiển này được sử dụng để tạo ra các liên kết siêu văn bản.

Các thuộc tính thường dùng

Thuộc tính	Mô tả
ImageUrl	Qui định hình hiển thị trên điều khiển.
Text	Chuỗi văn bản sẽ được hiển thị trên điều khiển. Trong trường hợp cả 2 thuộc tính ImageURL và Text được thiết lập, thuộc tính ImageURL được ưu tiên, thuộc tính Text sẽ được hiển thị như Tooltip
NavigateUrl	Đường dẫn cần liên kết đến
Target	Xác định cửa sổ sẽ hiển thị cho mỗi liên kết

Ví dụ:

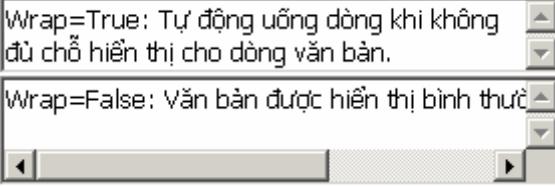
```
InkHi.Text = "Trang chủ ASP.NET";
InkHi.ImageUrl = "~/Hinh/Asp_net.jpg";
InkHi.NavigateUrl = "http://www.asp.net";
```

6.5.2.3 TextBox

<asp:TextBox ID="txtName" runat="server"></asp:TextBox>

TextBox là điều khiển được dùng để nhập và hiển thị dữ liệu. TextBox thường được sử dụng nhiều với các ứng dụng trên windows form.

Các thuộc tính

Thuộc tính	Mô tả
Text	Nội dung chứa trong Textbox
TextMode	Qui định chức năng của Textbox, có các giá trị sau: ✓ SingleLine: Hiển thị và nhập liệu 1 dòng văn bản ✓ MultiLine: Hiển thị và nhập liệu nhiều dòng văn bản ✓ Password: Hiển thị dấu * thay cho các ký tự có trong Textbox.
Rows	Trong trường hợp thuộc tính TextMode = MultiLine, thuộc tính Rows sẽ qui định số dòng văn bản được hiển thị.
Maxlength	Qui định số ký tự tối đa được nhập vào cho TextBox
Wrap	Thuộc tính này qui định việc hiển thị của văn bản có được phép tự động xuống dòng khi kích thước ngang của của điều khiển không đủ để hiển thị dòng nội dung văn bản. Giá trị mặc định của thuộc tính này là True - tự động xuống dòng. Ví dụ: 
AutoPostBack	Thuộc tính này qui định điều khiển có được phép tự động PostBack về Server khi nội dung trong Textbox bị thay đổi hay không. Giá trị mặc định của thuộc tính này là False - không tự động PostBack.

6.5.2.4 Image

<asp:ImageButton ImageUrl="logo.gif" ID="imgLogo" runat="server" />

Điều khiển này được dùng để hiển thị hình ảnh lên trang Web.

Các thuộc tính

Thuộc tính	Mô tả
ImageUrl	Đường dẫn đến tập tin hình ảnh cần hiển thị
AlternateText	Chuỗi văn bản sẽ hiển thị khi tập tin được thiết lập trong thuộc tính ImageUrl không tồn tại
ImageAlign	Vị trí hiển thị giữa hình và nội dung văn bản

6.5.2.5 Button, ImageButton, LinkButton

```
<asp:Button ID="Button1" runat="server" Text="OK" />
<asp:LinkButton ID="LinkButton1" runat="server">OK</asp:LinkButton>
<asp:ImageButton ImageUrl="ok.gif" ID="ImageButton1" runat="server" />
```

Các điều khiển Button, ImageButton, LinkButton mặc định đều là các nút Submit Button, mỗi khi được nhấn vào sẽ PostBack về Server.

Khi chúng ta thiết lập giá trị thuộc tính CommandName cho các điều khiển này, chúng ta gọi tên chung cho các điều khiển này là Command Button.

Các thuộc tính

Thuộc tính	Mô tả
Text	Chuỗi văn bản hiển thị trên điều khiển.
CommandName	Tên lệnh. Được sử dụng trong sự kiện Command.
CommandArgument	Thông tin bổ sung thông số cho nút gây ra sự kiện.
CausesValidation	Trang web mặc định kiểm tra tính hợp lệ dữ liệu mỗi khi được PostBack. Các điều khiển Button, ImageButton, LinkButton luôn PostBack về Server mỗi khi được nhấn -> luôn kiểm tra tính hợp lệ dữ liệu trên trang web. Muốn trang Web bỏ qua việc kiểm tra dữ liệu khi được nhấn, gán trị cho thuộc tính này = False. Giá trị mặc định của thuộc tính này là True.
PostBackUrl	Dùng post back sang trang khác

Các thuộc tính sau chỉ áp dụng cho ImageButton

ImageUrl	Đường dẫn đến tập tin hình ảnh cần hiển thị.
ImageAlign	Chuỗi văn bản sẽ hiển thị khi tập tin được thiết lập trong thuộc tính ImageUrl không tồn tại
AlternateText	Vị trí hiển thị giữa hình và nội dung văn bản

Sự kiện quan trọng

Sự kiện	Mô tả
Click	Xảy ra khi click vào nút

Ví dụ:



Button, LinkButton và ImageButton

6.5.2.6 Checkbox, RadioButton

```
<asp:CheckBox Text="Single ?" ID="CheckBox1" runat="server" />
<asp:RadioButton Text="Male" ID="RadioButton1" runat="server" />
<asp:RadioButton Text="Female" ID="RadioButton2" runat="server" />
```

Các điều khiển này thường được sử dụng để cung cấp trạng thái

Các thuộc tính

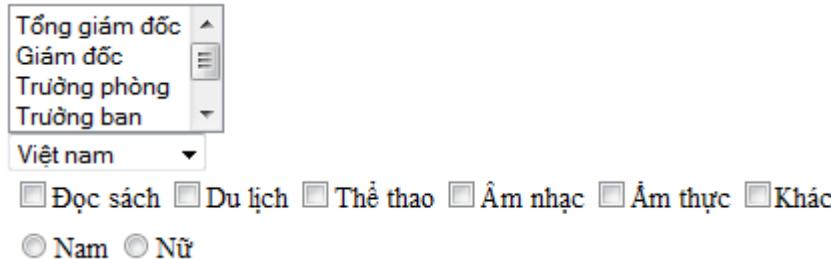
Thuộc tính	Mô tả
Checked	Cho biết trạng thái của mục chọn - có được chọn hay không
TextAlign	Qui định vị trí hiển thị của điều khiển so với chuỗi văn bản. <input type="checkbox"/> Right <input type="checkbox"/> Left <input type="checkbox"/>
AutoPostBack	Thuộc tính này qui định điều khiển có được phép tự động PostBack về Server khi các mục chọn của điều khiển bị thay đổi. Giá trị mặc định của thuộc tính này là False - không tự động Postback.
GroupName (RadioButton)	Tên nhóm. Thuộc tính này được sử dụng để nhóm các điều khiển RadioButton lại thành 1 nhóm.

Ví dụ:

Bạn đang học tiếng nước nào ? Bạn đã học tiếng Anh bao lâu ?

<input checked="" type="checkbox"/> Anh <input checked="" type="checkbox"/> Pháp <input type="checkbox"/> Hoa <input type="checkbox"/> Nhật	<input type="radio"/> < 1 năm <input checked="" type="radio"/> 1 -> 3 năm <input type="radio"/> 3 -> 5 năm <input type="radio"/> > 5 năm
--	---

6.5.2.7 Listbox, DropDownList, CheckBoxList và RadioButtonList



ListBox và DropDownList là điều khiển hiển thị danh sách lựa chọn mà người dùng có thể chọn một hoặc nhiều (chỉ dành cho ListBox). Các mục lựa chọn có thể được thêm vào danh sách thông qua lệnh hoặc ở cửa sổ thuộc tính (Property Windows).

Các thuộc tính

Thuộc tính	Mô tả
Items	Đây là tập hợp chứa các mục chọn của điều khiển. Ta có thể thêm vào mục chọn vào thời điểm thiết kế thông qua cửa sổ ListItem Collection Editor, hoặc thông qua lệnh.
AutoPostBack	Thuộc tính này qui định điều khiển có được phép tự động PostBack về Server khi chỉ số của mục chọn bị thay đổi. Giá trị mặc định của thuộc tính này là False - không tự động PostBack.
Xử lý mục chọn:	Các thuộc tính sau sẽ giúp bạn xác định chỉ số, giá trị của mục đang được chọn. Trong trường hợp điều khiển cho phép chọn nhiều, ta duyệt qua các Item trong tập hợp Items, sử dụng thuộc tính Selected của đối tượng Item để kiểm tra xem mục đó có được chọn hay không.
SelectedIndex	Cho biết chỉ số của mục được chọn. Trong trường hợp chọn nhiều mục, SelectedIndex sẽ trả về chỉ số mục chọn đầu tiên.
SelectedItem	Cho biết mục được chọn. Trong trường hợp chọn nhiều mục, SelectedItem sẽ trả về mục chọn đầu tiên.

SelectedValue	Cho biết giá trị của mục được chọn. Trong trường hợp chọn nhiều mục, SelectedValue sẽ trả về giá trị mục chọn đầu tiên.
---------------	---

Các thuộc tính chỉ áp dụng cho ListBox

Rows (ListBox)	Qui định chiều cao của ListBox theo số dòng hiển thị.
SelectionMode (ListBox)	Thuộc tính này xác định cách thức chọn các mục trong ListBox. SelectionMode chỉ được phép thay đổi trong quá trình thiết kế, vào lúc thực thi chương trình, thuộc tính này chỉ đọc. ✓ Single: Chỉ được chọn một mục có trong danh sách (mặc định). ✓ Multiple: Cho phép chọn nhiều lựa chọn.

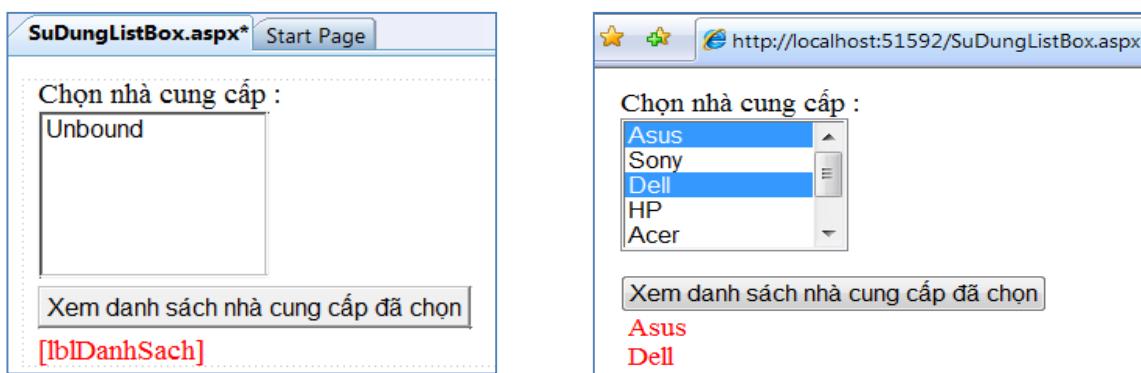
Các thuộc tính chỉ áp dụng cho RadioButtonList và CheckBoxList

RepeatColumns	Qui định số cột hiển thị
RepeatDirection	Qui định hình thức hiển thị ✓ Vertical: Theo chiều dọc ✓ Horizontal: Theo chiều ngang

Sự kiện quan trọng

Sự kiện	Mô tả
SelectedIndexChanged	Xảy ra khi chọn một mục. Sự kiện này chỉ xảy ra khi thuộc tính AutoPostBack được thiết lập là true

Ví dụ: Điều khiển danh sách IstNhaCungCap: SelectionMode=Multiple, Rows=4



Khi thiết kế

Khi nhấp nút

Xử lý các sự kiện: Phần lệnh trong tập tin SuDungListBox.aspx.cs

```

protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        lstNhaCungCap.Items.Add("Asus");
        lstNhaCungCap.Items.Add("Sony");
        lstNhaCungCap.Items.Add("Dell");
        lstNhaCungCap.Items.Add("HP");
        lstNhaCungCap.Items.Add("Acer");
        lstNhaCungCap.Items.Add("GetWay");
        lstNhaCungCap.Items.Add("Lenovo");
    }
}

protected void btnHienThiNhaCungCap_Click(object sender, EventArgs e)
{
    for (int i = 0; i < lstNhaCungCap.Items.Count; i++)
    {
        if (lstNhaCungCap.Items[i].Selected)
        {
            lblDanhSach.Text += lstNhaCungCap.Items[i].Text + "<br>";
        }
    }
}

```

6.5.2.8 Điều khiển Calendar

<asp:Calendar ID="calNgaySinh" runat="server"></asp:Calendar>

Một điều chắc chắn rằng điều khiển Calendar đã quá quen thuộc với các bạn lập trình ứng dụng trên windows, nó có giao diện trực quan, vì vậy, người dùng có thể chọn ngày dễ dàng.

Thuộc tính

Thuộc tính	Mô tả
DayHeaderStyle	Qui định hình thức hiển thị tiêu đề của các ngày trong tuần
DayStyle	Qui định hình thức hiển thị của các ngày trong điều khiển.
NextPrevStyle	Qui định hình thức hiển thị của tháng trước/sau của tháng đang được chọn.
SelectedDayStyle	Qui định hình thức hiển thị của ngày đang được chọn.
SelectedDate	Giá trị ngày được chọn trên điều khiển
TitleStyle	Qui định hình thức hiển thị dòng tiêu đề của tháng được chọn
TodayDayStyle	Qui định hình thức hiển thị của ngày hiện hành (trên server).
WeekendDayStyle	Qui định hình thức hiển thị của các ngày cuối tuần (thứ 7, chủ nhật)
OtherMonthDayStyle	Qui định hình thức hiển thị của các ngày không nằm trong tháng hiện hành.

Sự kiện

Sự kiện	Mô tả
SelectionChanged	X sự kiện này xảy ra khi bạn chọn một ngày khác với giá trị ngày đang được chọn hiện hành
VisibleMonthChanged	X sự kiện này xảy ra khi bạn chọn tháng khác với tháng hiện hành

Ví dụ: Sử dụng điều khiển Calendar



Khi thiết kế



Khi thi hành

Xử lý sự kiện

```
protected void btnXemNgay_Click(object sender, EventArgs e)
{
    lblNgay.Text = "Bạn đã chọn ngày : " + calenda1.SelectedDate.ToString("dd/MM/yyyy");
}
```

6.5.2.9 Điều khiển FileUpload

<asp:FileUpload ID="fupHinh" runat="server" />

Điều khiển này được sử dụng để upload tập tin lên server.

Các thuộc tính lập trình

Thuộc tính	Mô tả
FileName	Tên của file gốc
HasFile	Cho biết file có được upload hay không
FileContent	Lấy Stream dẫn đến file được upload
FileBytes	Mảng byte[] chứa nội dung file được upload

Ví dụ sau đây cho phép lấy tên tập tin vừa được upload và lưu vào thư mục "hinh" đặt tại thư mục gốc của website.

```
protected void btnUpload_Click(object sender, EventArgs e)
{
    String tenTapTin = fupHinh.FileName;
    String duongDanLuuHinh = Server.MapPath("~/hinh/" + tenTapTin);
    fupHinh.SaveAs(duongDanLuuHinh);
}
```

6.5.2.10 Điều khiển Hidden

<asp:HiddenField ID="hidCountry" runat="server" Value="Việt nam" />

Điều khiển này tạo ra trường ẩn (không nhìn thấy), thường dùng để chứa dữ liệu mặc định không cần phải nhập từ người dùng.

Thuộc tính quan trọng quy nhất là **Value** dùng để chứa giá trị nhập sẵn.

6.5.3 Các điều khiển nâng cao

6.5.3.1 Điều khiển AdRotator

<asp:AdRotator ID="adrQuangCao" runat="server" AdvertisementFile="QuangCao.xml" Height="200px" Width="350px" />

Điều khiển AdRotator được dùng để tạo ra các banner quảng cáo cho trang web, nó tự động thay đổi các hình ảnh (đã được thiết lập trước) mỗi khi có yêu cầu, PostBack về server.

Các thuộc tính

Thuộc tính	Mô tả
AdvertisementFile	Tên tập tin dữ liệu (dưới dạng xml) cho điều khiển. Dưới đây là cấu trúc của tập tin Advertisement (*.xml)

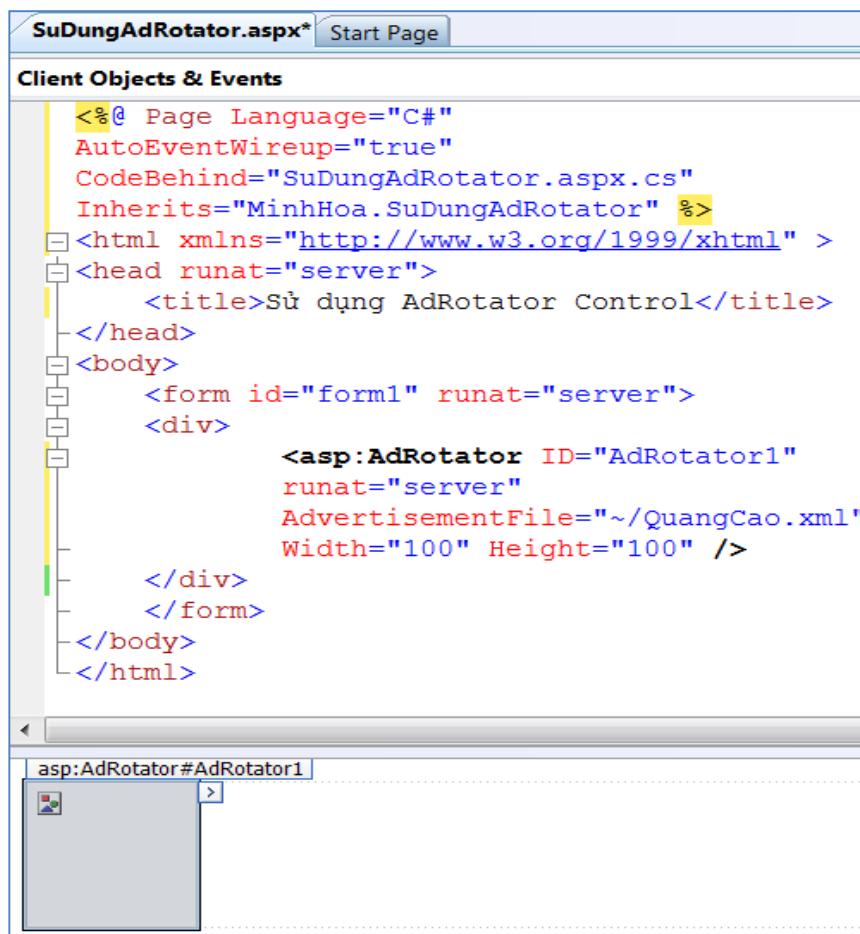
```
<Advertisements>
<Ad>
    <ImageUrl>Hình ảnh</ImageUrl>
    <NavigateUrl>Liên kết</NavigateUrl>
    <AlternateText>Tooltip</AlternateText>
    <Keyword>Từ khóa dùng để lọc quảng cáo</Keyword>
    <Impressions>Tần suất hiển thị</Impressions>
</Ad>
<!-- mô tả cho các quảng cáo khác-->
<Ad> ..... </Ad>
</Advertisements>
```

Lưu ý: Phải nhập đúng các giá trị trong tag như mẫu trên. Các giá trị trong tag có phân biệt chữ Hoa chữ thường. Trong đó **ImageUrl:** Đường dẫn đến một tập tin hình ảnh

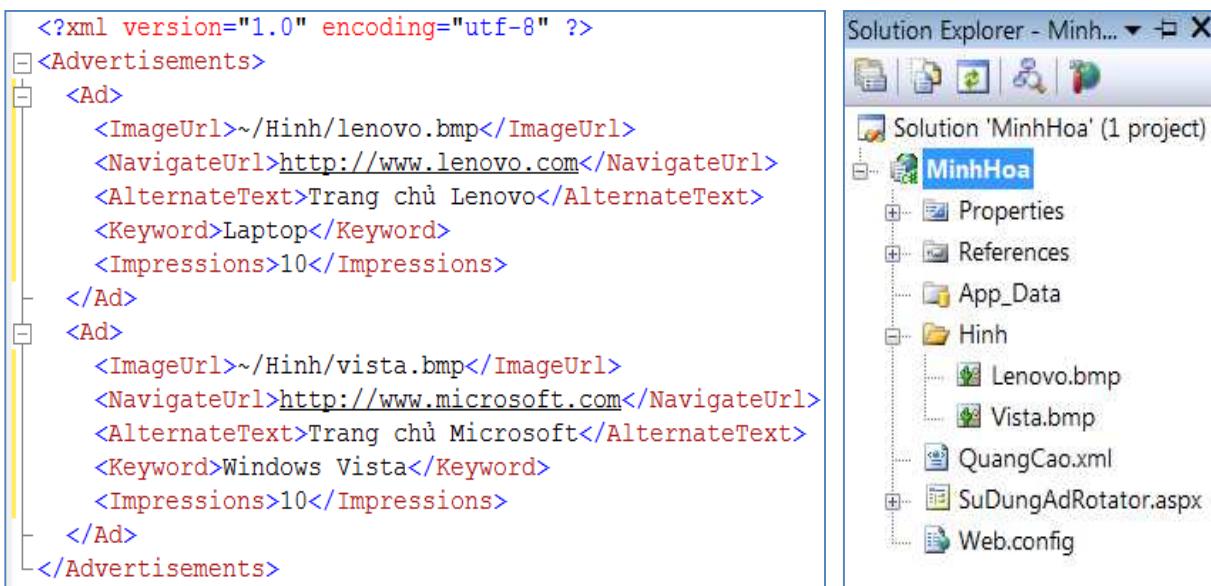
	<p>NavigateUrl: Đường dẫn đến trang web sẽ được liên kết đến khi người dùng nhấp vào hình ảnh đang hiển thị.</p> <p>AlternateText: Giá trị này sẽ được hiển thị nếu như đường dẫn đến tập tin hình ảnh (qua thuộc tính NavigateUrl) không tồn tại. Đối với một số trình duyệt, tham số này được hiển thị như ToolTip của hình quảng cáo.</p> <p>Keyword: Được dùng để phân loại các quảng cáo. Thông qua giá trị này, ta có thể lọc các quảng cáo theo một điều kiện nào đó.</p> <p>Impressions: Tham số này quyết định tầng suất hiển thị của hình ảnh. Giá trị này càng lớn, khả năng hiển thị càng nhiều.</p>
KeywordFilter	<p>Được dùng để chọn lọc và hiển thị những hình quảng cáo có giá trị của tham số Keyword = giá trị của tham số này.</p> <p>Giá trị của tham số này mặc định không được thiết lập nghĩa là hiển thị tất cả những các quảng cáo có trong tập tin XML.</p>
Target	<p>Qui định cửa sổ hiển thị trang liên kết</p> <ul style="list-style-type: none"> ✓ _blank: Trang liên kết sẽ được mở ở một cửa sổ mới. ✓ _self: Trang liên kết sẽ được mở ở chính cửa sổ chưa điều khiển. ✓ _parent: Trang liên kết sẽ được mở ở cửa sổ cha.

Ví dụ: Tạo Quảng cáo sử dụng điều khiển AdRotator

Bước 1 : Thiết kế giao diện



Bước 2 : Chọn project , nhấp phải chuột vào Add | New Item. Trong cửa sổ này chọn XML File đặt tên tập tin QuangCao.xml với nội dung như sau :



Bước 3 : Nhấn Ctrl-F5 để thi hành trang SuDungAdRotator.aspx sau đó nhấn nút **Refresh** để thay đổi hình. Kết quả như 2 hình sau :



6.5.3.2 Điều khiển Panel

```

<asp:Panel ID="Panel1" runat="server" GroupingText="Giới tính"
HorizontalAlign="Center">

    <asp:RadioButtonList ID="rdoGioiTinh" runat="server"
RepeatDirection="Horizontal">

        <asp:ListItem Value="1">Nam</asp:ListItem>
        <asp:ListItem Value="0">Nữ</asp:ListItem>
    </asp:RadioButtonList>
</asp:Panel>

```

Giới tính

Nam Nữ

Panel được sử dụng để nhóm các điều khiển khác thành một nhóm. 2 Thuộc tính thường dùng của điều khiển này là Visible, GroupingText, Width, Height và HorizontalAlign

Thuộc tính	Mô tả
Visible	Ẩn hiện các Panel khi đó tất cả các điều khiển bên trong nó cũng ẩn hiện theo.
GroupingText	Chuỗi chứa đề mục của nhóm
HorizontalAlign	Căn lề các điều khiển bên trong Panel (Left, Right, Center, Justify)
Width	Chiều rộng
Height	Chiều cao

6.5.3.3 Điều khiển PlaceHolder

<asp:PlaceHolder ID="phdMessage" runat="server"></asp:PlaceHolder>

Điều khiển PlaceHolder được sử dụng để dành trước chỗ cho điều khiển khác được tạo ra trong lúc lập trình sau này. Thuộc tính thường dùng của điều khiển này là Visible và Controls. Visible dùng để điều khiển ẩn hiện còn Controls được sử dụng để quản lý các điều khiển bên trong.

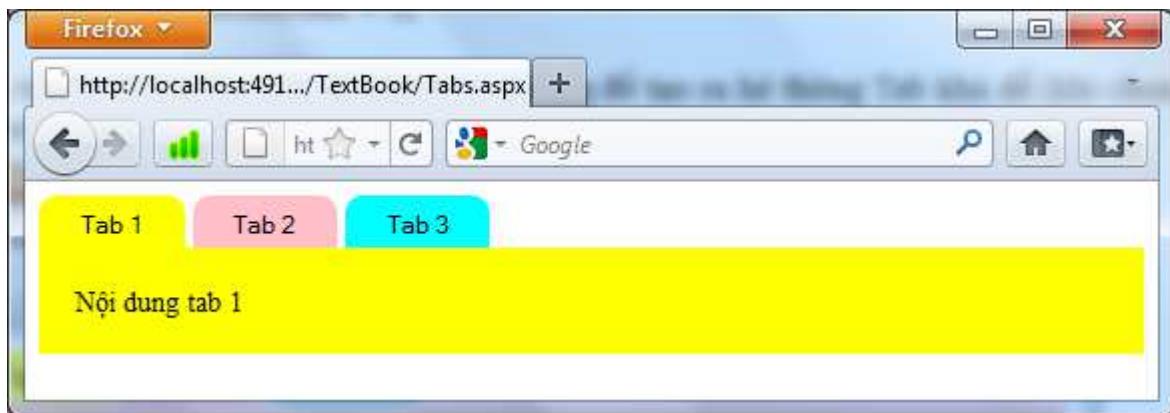
Để đưa một điều khiển vào vị trí dành trước (phdMessage) bằng mã C# ta viết như sau:

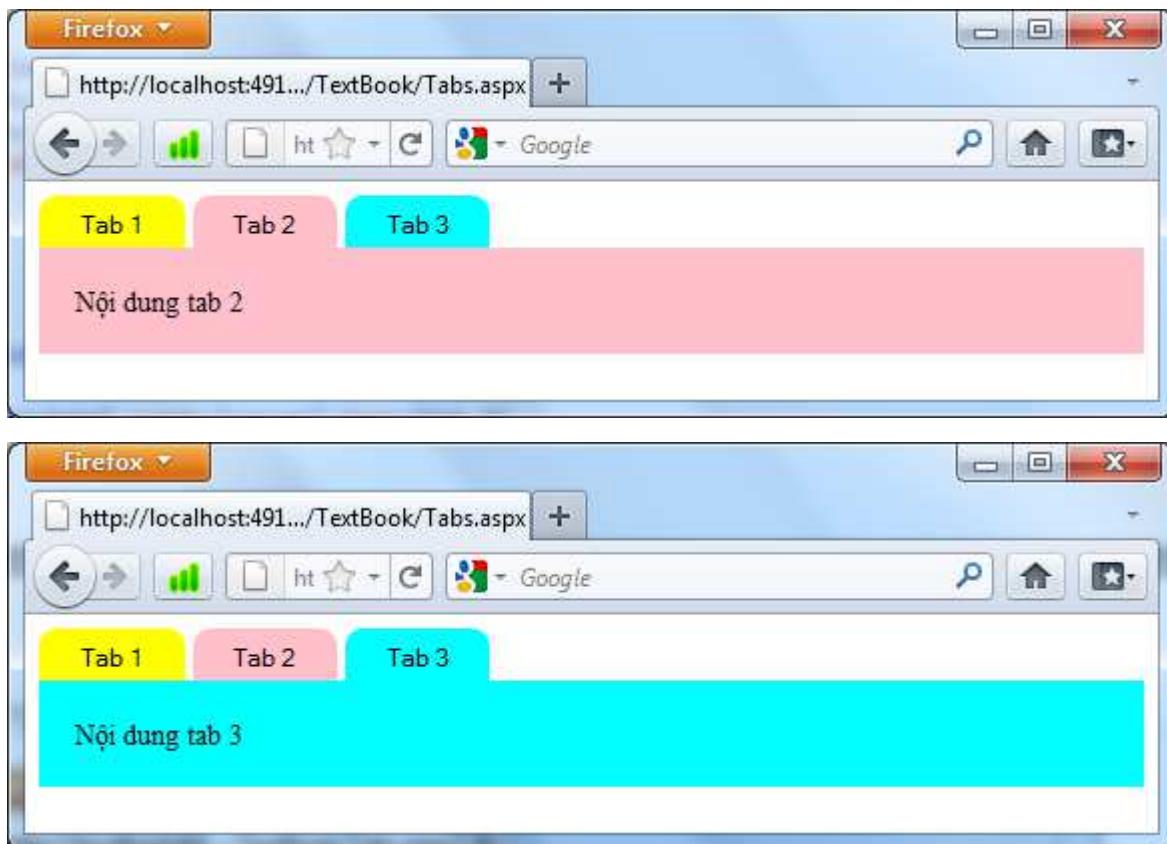
```
protected void Page_Load(object sender, EventArgs e)
{
    Label lblHello = new Label();
    lblHello.Text = "Xin chào";
    phdMessage.Controls.Add(lblHello);
}
```

6.5.3.4 MultiView và View

- MultiView là một điều khiển Standard ToolBox.
- Điều khiển MultiView dùng làm khung chứa nhiều điều khiển View.
- Tại một thời điểm MultiView chỉ cho phép hiển thị một View duy nhất tùy thuộc vào thuộc tính **ActiveViewIndex** (tính từ vị trí 0).

Ví dụ: Nhấp chuột vào nút nào thì hiển thị view tương ứng.





Mã ASP.NET

```

<style>
.tab1-head,.tab2-head,.tab3-head
{
    border-top-left-radius: 10px;
    border-top-right-radius: 10px;
    padding: 5px 20px;
    border:0px;
}
.tab1-head,.tab1-body {background:yellow;}
.tab2-head,.tab2-body {background:pink;}
.tab3-head,.tab3-body {background:aqua;}
.tab1-body,.tab2-body,.tab3-body {padding: 20px;}
</style>

<asp:Button class="tab1-head" ID="btnTab1" runat="server" Text="Tab 1"
    onclick="btnTab1_Click" />
<asp:Button class="tab2-head" ID="btnTab2" runat="server" Text="Tab 2"
    onclick="btnTab3_Click" />
<asp:Button class="tab3-head" ID="btnTab3" runat="server" Text="Tab 3"
    onclick="btnTab2_Click" />
<asp:MultiView ID="MultiView1" runat="server" ActiveViewIndex="0">
    <asp:View ID="View1" runat="server">
        <div class="tab1-body">Nội dung tab 1</div>
    </asp:View>
    <asp:View ID="View2" runat="server">
        <div class="tab2-body">Nội dung tab 2</div>
    </asp:View>

```

```
<asp:View ID="View3" runat="server">
<div class="tab3-body">Nội dung tab 3</div>
</asp:View>
</asp:MultiView>
```

Mã C# code behind

```
protected void btnTab1_Click(object sender, EventArgs e)
{
    MultiView1.ActiveViewIndex = 0;
}
protected void btnTab3_Click(object sender, EventArgs e)
{
    MultiView1.ActiveViewIndex = 1;
}
protected void btnTab2_Click(object sender, EventArgs e)
{
    MultiView1.ActiveViewIndex = 2;
}
```

6.5.3.5 Wizard

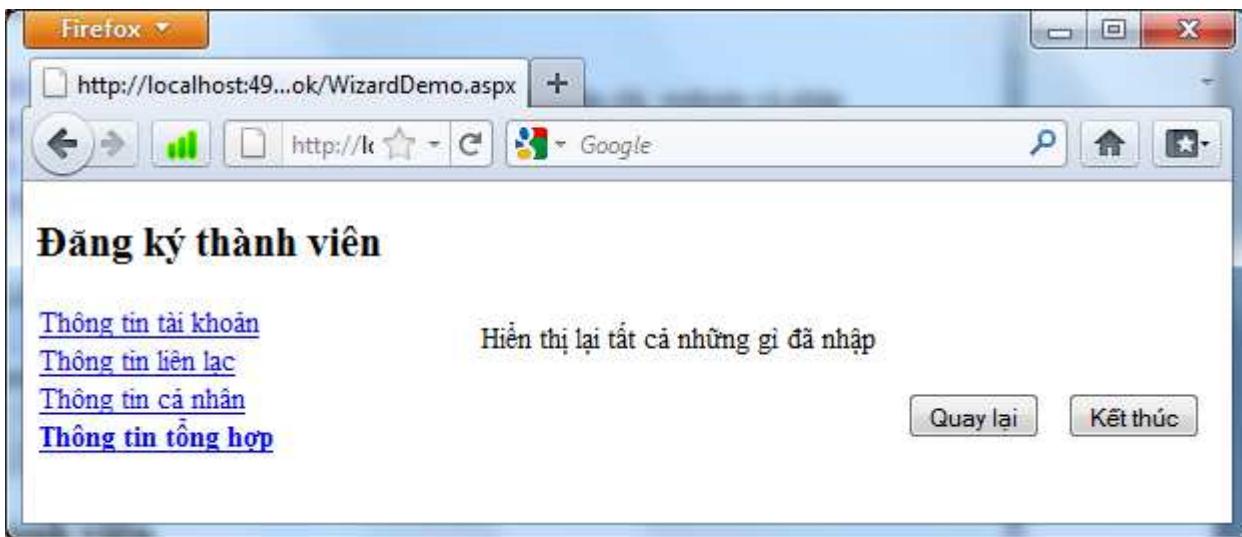
Điều khiển Wizard giống với điều khiển MultiView có thể dùng để chia một Form lớn thành nhiều phần nhỏ, giúp tạo giao diện từng bước (step by step) để thực hiện một vấn đề gồm nhiều phần giao diện khác nhau. Tuy nhiên nó sẽ có thêm một số thuộc tính mà MultiView không hỗ trợ.



```
<asp:Wizard ID="Wizard2" runat="server" ActiveStepIndex="2">
<WizardSteps>
    <asp:WizardStep runat="server" title="Step 1">
        Giao diện bước 1
    </asp:WizardStep>
    <asp:WizardStep runat="server" title="Step 2">
        Giao diện bước 2
    </asp:WizardStep>
    <asp:WizardStep runat="server" Title="Step 3">
        Giao diện bước 2
    </asp:WizardStep>
</WizardSteps>
</asp:Wizard>
```

Ứng dụng Wizard chúng ta có thể thiết kế giao diện đăng ký nhiều bước như phác thảo sau đây:





Một số thuộc tính quan trọng:

- ActiveStep: cho phép bạn lấy thông tin của WizardStep đang kích hoạt
- ActiveStepIndex: cho phép bạn gán hoặc lấy về chỉ số Index của WizardStep đang kích hoạt
- CancelDestinationPageUrl: cho phép bạn chỉ rõ địa chỉ URL được gửi tới khi người sử dụng nhấn nút Cancel
- DisplayCancelButton: Cho phép ẩn hoặc hiện Cancel Button.
- DisplaySlideBar: Cho phép ẩn hoặc hiện SlideBar(hiển thị tất cả các WizardStep)
- FinishDestinationPageUrl: cho phép bạn chỉ định địa chỉ URL được gửi tới khi người dùng nhấn nút Finish
- HeaderText: cho phép bạn chỉ định tiêu đề hiển thị trên đỉnh của điều khiển Wizard.
- WizardSteps: Cho phép bạn lấy thông tin của các điều khiển WizardStep trong điều khiển Wizard

Điều khiển Wizard hỗ trợ các phương thức:

- GetHistory(): cho phép lấy thông tin của các điều khiển Wizard mà đã truy cập.
- GetStepType(): Cho phép bạn trả về kiểu của mỗi WizardStep riêng theo chỉ số, nó có thể là các thuộc tính sau: Auto, Start, Finish hay Step
- MoveTo(): cho phép bạn di chuyển đến một WizardStep.

Điều khiển Wizard hỗ trợ các sự kiện:

- ActiveStepChanged: xảy ra khi một WizardStep trở thành Step được kích hoạt
- CancelButtonClick: xảy ra khi Cancel Button được nhấn.
- FinishButtonClick: xảy ra khi Finish Button được nhấn
- NextButtonClick: Xảy ra khi Next button được nhấn
- PreviousButtonClick: xảy ra khi Previous button được nhấn
- SlideBarButtonClick: xảy ra khi SlideBar button được nhấn

6.5.3.6 TreeView

<asp:TreeView ID="TreeView2" runat="server"></asp:TreeView>



Điều khiển này được sử dụng để tạo giao diện có cấu trúc cây. Mỗi cây có 1 gốc và nhiều node. Mỗi node lại chứa các node con khác.

Chúng ta có thể tạo ra cây theo các phương pháp

- Thiết kế
- Lập trình
- Cả 2 thiết kế và lập trình

Khi làm việc với TreeView, chúng ta cần chú ý đến các sự kiện sau đây.

- **SelectedNodeChanged:** xảy ra khi một node được chọn.
- **TreeNodeCollapsed:** xảy ra khi một node được xả xuống.
- **TreeNodeExpanded:** xảy ra khi một node bị co lại.

Sau đây là một TreeView được tạo ra bằng cả 2 cách thiết kế và lập trình. Chú ý node "NhatNghe" được tạo ra từ lập trình.

Mã ASP.NET

```
<asp:TreeView ID="TreeView1" runat="server"
onselectednodechanged="TreeView1_SelectedNodeChanged"
ontreenodecollapsed="TreeView1_TreeNodeCollapsed"
ontreenodeexpanded="TreeView1_TreeNodeExpanded">
<Nodes>
    <asp:TreeNode Text="My Computer" Value="My Computer">
        <asp:TreeNode Text="C Drive" Value="C Drive">
            <asp:TreeNode Text="Program Files" Value="Program Files"></asp:TreeNode>
            <asp:TreeNode Text="Users" Value="Users"></asp:TreeNode>
            <asp:TreeNode Text="Windows" Value="Windows"></asp:TreeNode>
        </asp:TreeNode>
        <asp:TreeNode Text="D Drive" Value="D Drive">
            <asp:TreeNode Text="Storage" Value="Storage"></asp:TreeNode>
            <asp:TreeNode Text="Softs" Value="Softs">
                <asp:TreeNode Text="WebSofts" Value="WebSofts"></asp:TreeNode>
                <asp:TreeNode Text="Virus" Value="Virus"></asp:TreeNode>
            </asp:TreeNode>
        </asp:TreeNode>
        <asp:TreeNode Text="E Drive" Value="E Drive">
            <asp:TreeNode Text="Drivers" Value="New Node"></asp:TreeNode>
            <asp:TreeNode Text="Ghosts" Value="Ghosts"></asp:TreeNode>
        </asp:TreeNode>
    </asp:TreeNode>
</Nodes>
</asp:TreeView>
```

Mã C# code behind

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        // thêm node "NhatNghe" vào node thứ 2
        TreeView1.Nodes[0].ChildNodes[1].ChildNodes.Add(new TreeNode("NhatNghe"));
    }
}

protected void TreeView1_SelectedNodeChanged(object sender, EventArgs e)
{
    // xử lý node chọn
    TreeNode node = TreeView1.SelectedNode;
}

protected void TreeView1_TreeNodeExpanded(object sender, TreeNodeEventArgs e)
{
    // xử lý node xả xuống
    TreeNode node = e.Node;
}

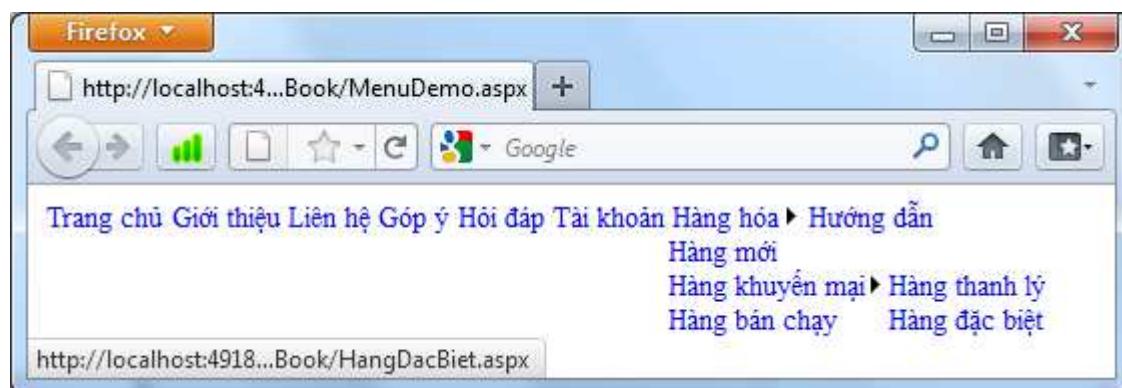
protected void TreeView1_TreeNodeCollapsed(object sender, TreeNodeEventArgs e)
{
    // xử lý node co lại
    TreeNode node = e.Node;
}
```

6.5.3.7 Menu

Điều khiển này giúp tạo ra menu đứng và menu ngang (tùy thuộc vào thuộc tính Orientation là Horizontal hay Vertical).

Mỗi menu chứa nhiều menu item và mỗi menu item có thể chứa nhiều menu item khác tạo thành menu đa cấp.

Chúng ta có thể tạo ra menu bằng thiết kế, lập trình hay kết hợp cả hai. Menu sau đây được tạo ra bằng thiết kế và lập trình.



Mã ASP.NET

```
<asp:Menu ID="Menu1" runat="server" Orientation="Horizontal">
<Items>
    <asp:MenuItem NavigateUrl="TrangChu.aspx" Text="Trang chủ" Value="Trang chủ">
    </asp:MenuItem>
    <asp:MenuItem NavigateUrl="GioiThieu.aspx" Text="Giới thiệu" Value="Giới thiệu">
```

```
</asp:MenuItem>
<asp:MenuItem NavigateUrl="LienHe.aspx" Text="Liên hệ" Value="Liên hệ">
</asp:MenuItem>
<asp:MenuItem NavigateUrl="GopY.aspx" Text="Góp ý" Value="Góp ý"></asp:MenuItem>
<asp:MenuItem NavigateUrl="HoiDap.aspx" Text="Hỏi đáp" Value="Hỏi đáp">
</asp:MenuItem>
<asp:MenuItem NavigateUrl="TaiKhoan.aspx" Text="Tài khoản" Value="Tài khoản">
</asp:MenuItem>
<asp:MenuItem NavigateUrl="HangHoa.aspx" Text="Hàng hóa" Value="Hàng hóa">
<asp:MenuItem NavigateUrl="HangMoi.aspx" Text="Hàng mới" Value="Hàng mới">
</asp:MenuItem>
<asp:MenuItem NavigateUrl="HangKhuyenMai.aspx" Text="Hàng khuyến mại"
Value="Hàng khuyến mại">
<asp:MenuItem NavigateUrl="HangThanhLy.aspx" Text="Hàng thanh lý"
Value="Hàng thanh lý"></asp:MenuItem>
<asp:MenuItem NavigateUrl="HangDacBiet.aspx" Text="Hàng đặc biệt"
Value="Hàng đặc biệt"></asp:MenuItem>
</asp:MenuItem>
<asp:MenuItem NavigateUrl="HangBanChay.aspx" Text="Hàng bán chạy"
Value="Hàng bán chạy"></asp:MenuItem>
</asp:MenuItem>
</Items>
</asp:Menu>
```

Mã C# code behind

```
protected void Page_Load(object sender, EventArgs e)
{
    // tạo menu item
    MenuItem mi = new MenuItem("Hướng dẫn");
    mi.Value = "Hướng dẫn";
    mi.NavigateUrl = "HuongDan.aspx";

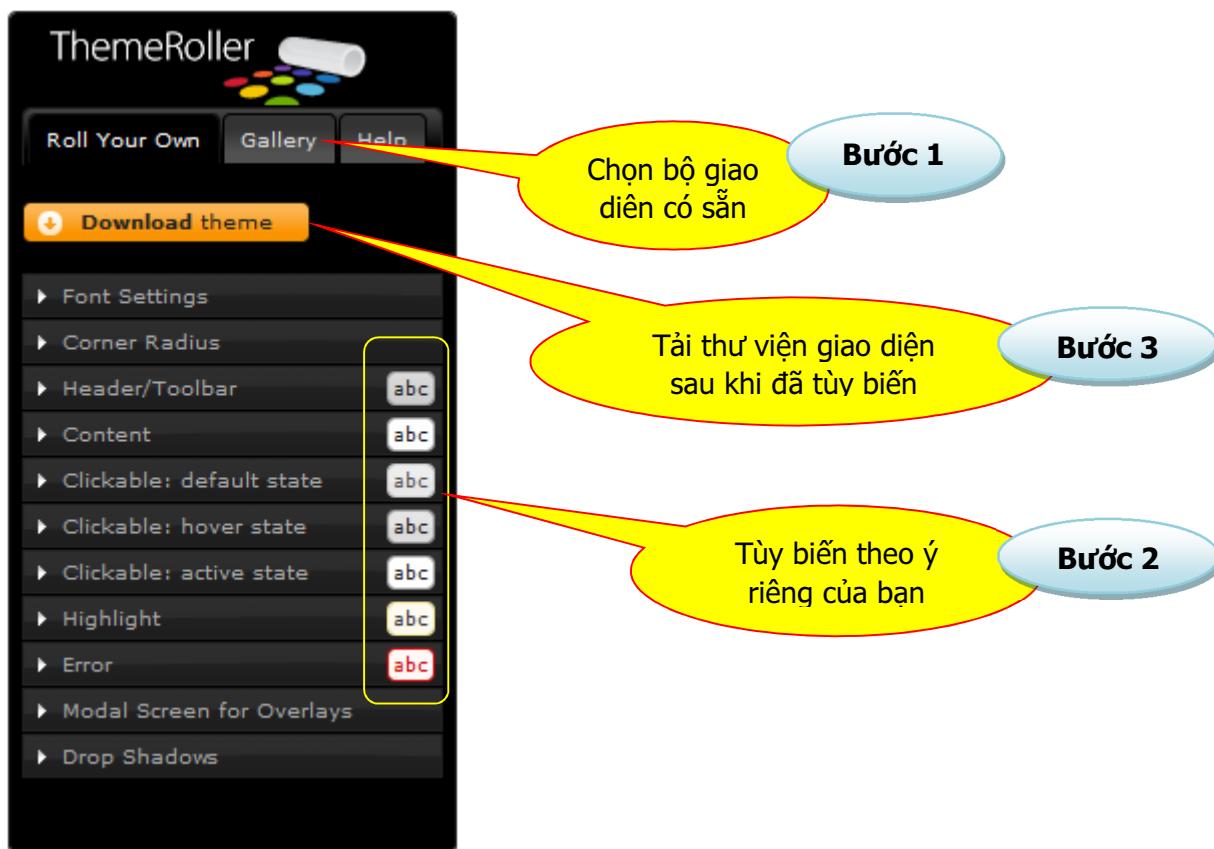
    // thêm menu item vào cuối menu
    Menu1.Items.Add(mi);
}
```

6.6 Các thành phần giao diện thường dùng của jQuery UI

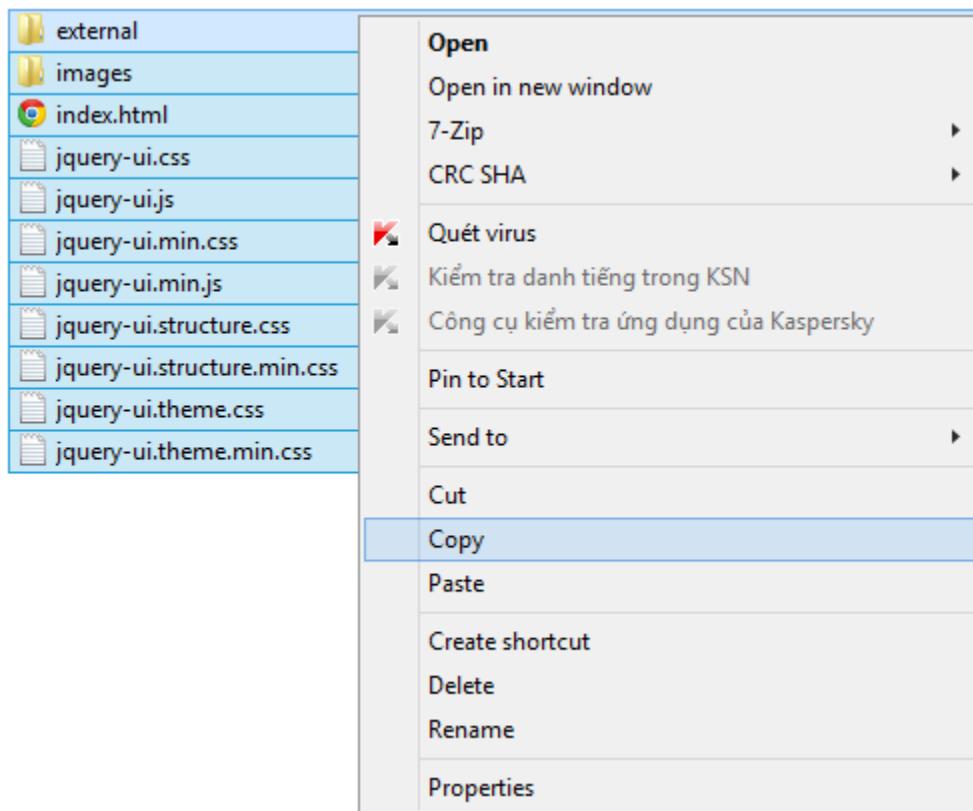
6.6.1 Thiết kế theme

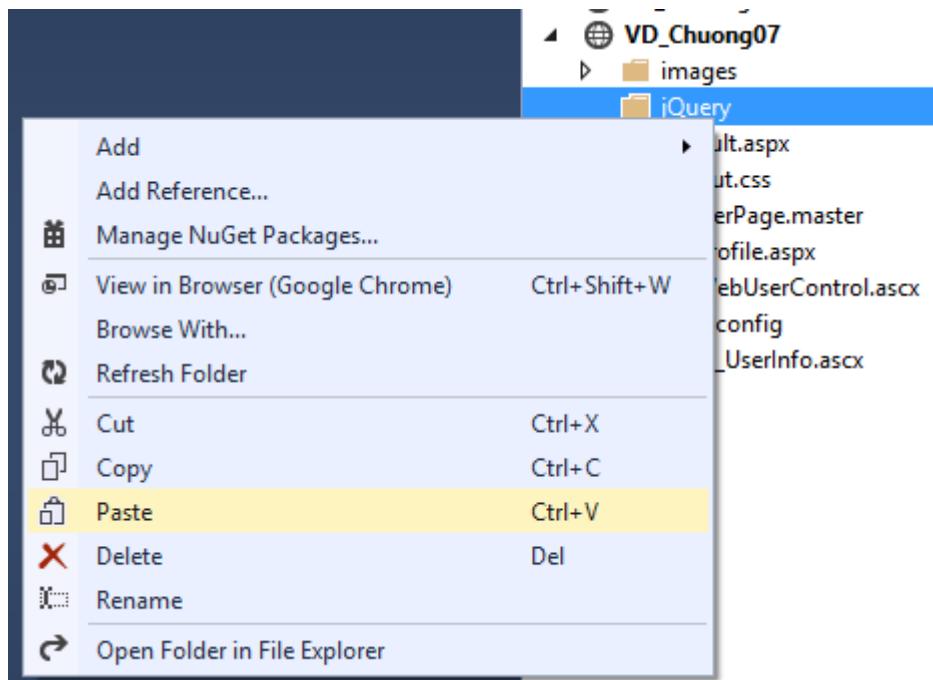
Để các thành phần giao diện của JQuery có theme phù hợp với giao diện hiện tại của ứng dụng web hiện tại, bạn phải thiết kế trực tuyến bộ giao diện trước khi download để sử dụng.

Vào địa chỉ này để thiết kế bộ giao diện: <http://jqueryui.com/themeroller/> để chọn một bộ giao diện ứng ý từ Gallery sau đó sửa đổi đôi chút hoặc thiết kế mới hoàn toàn.



Cuối cùng bạn chỉ việc download thư viện được sinh ra trong quá trình sửa đổi sau đó giải nén và chép các phần cần thiết (thư mục js và css) vào thư mục jquery trong website của bạn.





6.6.2 Liên kết thư viện JQuery

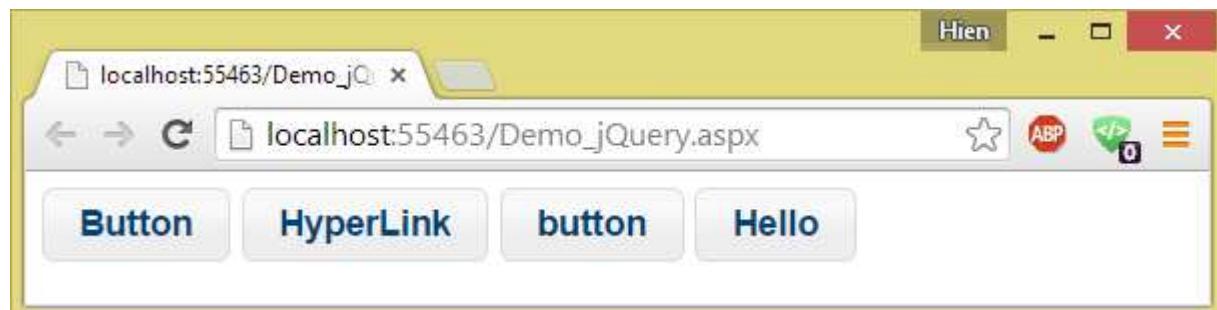
Trước khi viết mã tạo các thành phần giao diện, bạn phải chỉ ra thư viện JQuery bằng 2 liên kết JavaScript và một liên kết css sau:

```
<head runat="server">
    <title></title>
    <script src="jQuery/external/jquery/jquery.js"></script>
    <script src="jQuery/jquery-ui.js"></script>
    <link href="jQuery/jquery-ui.css" rel="stylesheet" />
</head>
```

6.6.3 JQuery button

Để sử dụng kiểu dáng button của JQuery bạn chỉ cần gọi hàm button trên control (nút, liên kết) bạn muốn.

Ví dụ sau đây sẽ tạo kiểu dáng button cho cả `<a>`, `<asp:HyperLink>`, `<input type='button'>` và `<asp:Button>`



Mã nguồn đoạn nhúng:

```
<script>
$(function () {
    $(".my-button").button();
```

```
});  

</script>  
  

<asp:Button CssClass="my-button" ID="Button1" runat="server" Text="Button" />  

<asp:HyperLink CssClass="my-button" ID="lnk"  

runat="server">HyperLink</asp:HyperLink>  

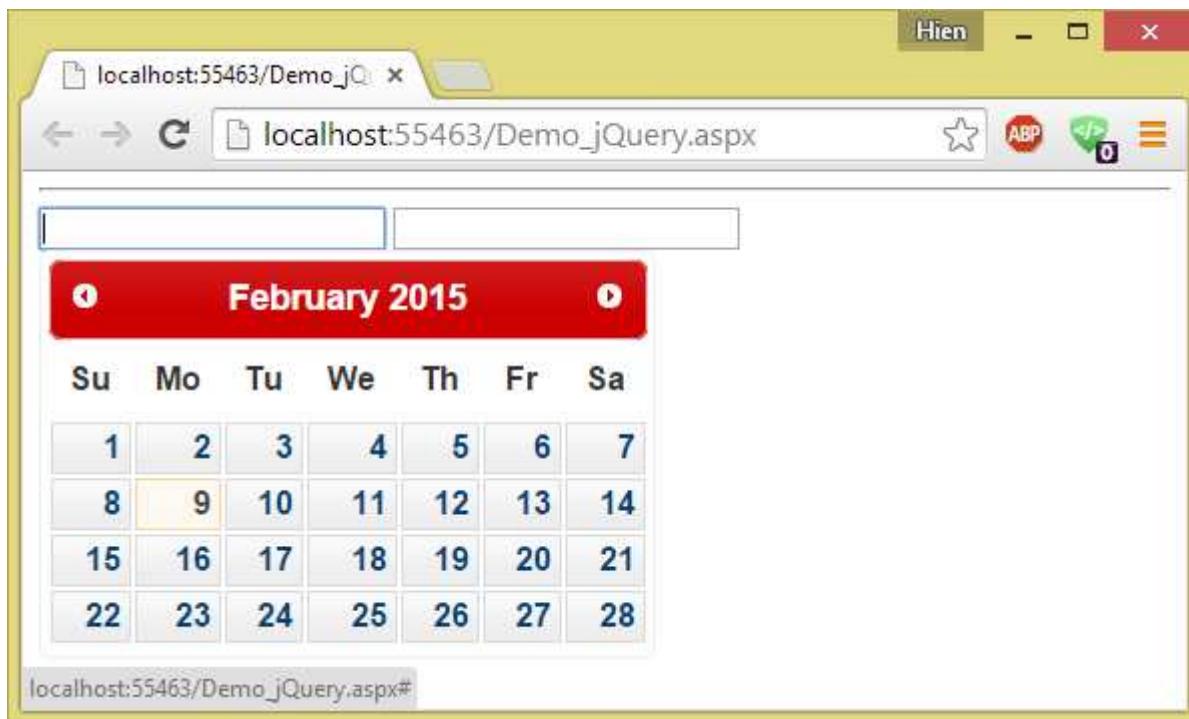
<input class="my-button" id="Button2" type="button" value="button" />  

<a class="my-button" href="">Hello</a>
```

6.6.4 JQuery Datepicker

Thành phần giao diện này cho phép nhập ngày tháng bằng cách chọn từ bảng lịch sổ xuống. Bạn cũng có thể định dạng ngày nhận được tùy thích cũng như hạn chế ngày nhập vào nhờ vào các tùy chọn.

Ví dụ sau minh họa nhập ngày đơn giản:



Mã nguồn đoạn nhúng:

```
<script>  

$(function () {  

    $(".date").datepicker();  

});  

</script>  

<asp:TextBox CssClass="date" ID="TextBox" runat="server" />  

<input class="date" id="Button2" type="text" />  

</div>
```

Nếu bạn muốn định dạng ngày nhận được thì chỉ cần thêm tùy chọn ngày vào hàm datepicker như sau:

```
<script>  

$(function () {  

    $(".date").datepicker({dateFormat:'dd-mm-yy'});  

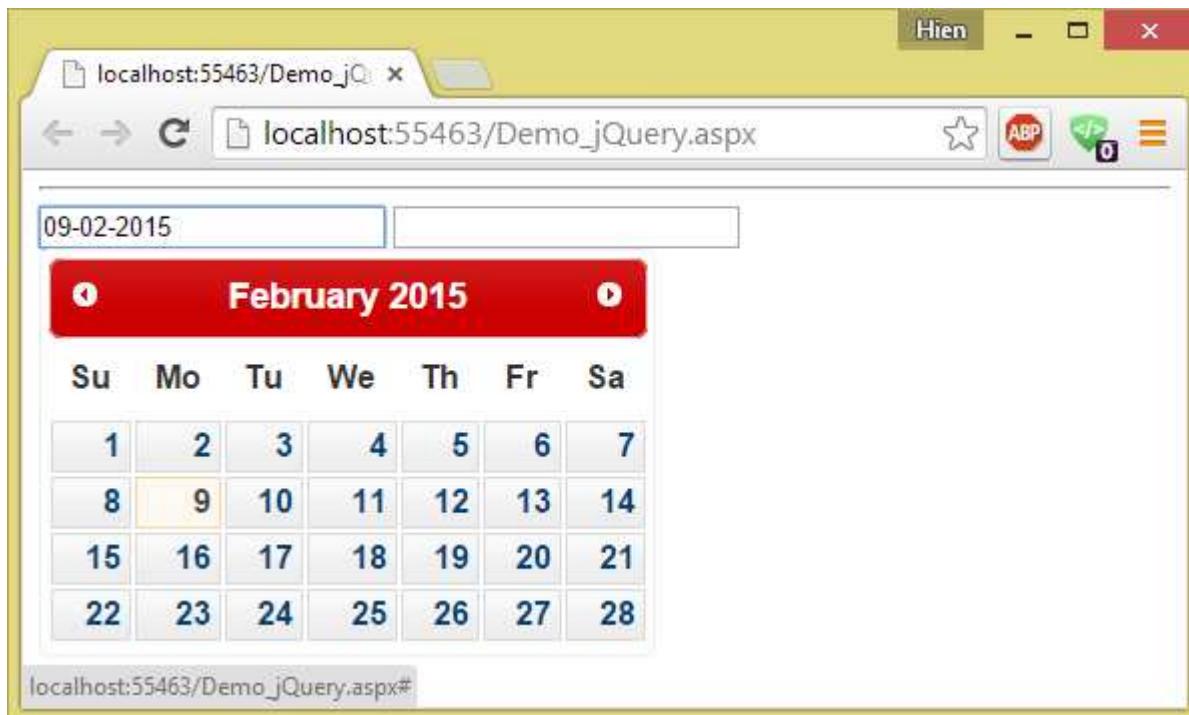
});</script>
```

```
});  

</script>
```

Các ký tự định dạng:

- ✓ y: 2 chữ số năm
- ✓ m: một chữ số tháng
- ✓ d: một chữ số ngày



Nếu bạn muốn hạn chế ngày nhập vào thì thêm tham số minDate và maxDate

```
<script>  

$(function () {  

    $(".date").datepicker({dateFormat: 'yy-mm-dd', minDate: -2, maxDate: "+1M  

+5D"});  

});  

</script>
```

Tham số minDate qui định ngày tối thiểu được phép chọn còn maxDate qui định ngày tối đa được chọn.

Trong ví dụ này chúng ta hiểu ngày tối thiểu là trước ngày hiện tại 2 ngày, còn ngày tối đa là sau ngày hiện tại 1 tháng 5 ngày.

Các ký tự định dạng:

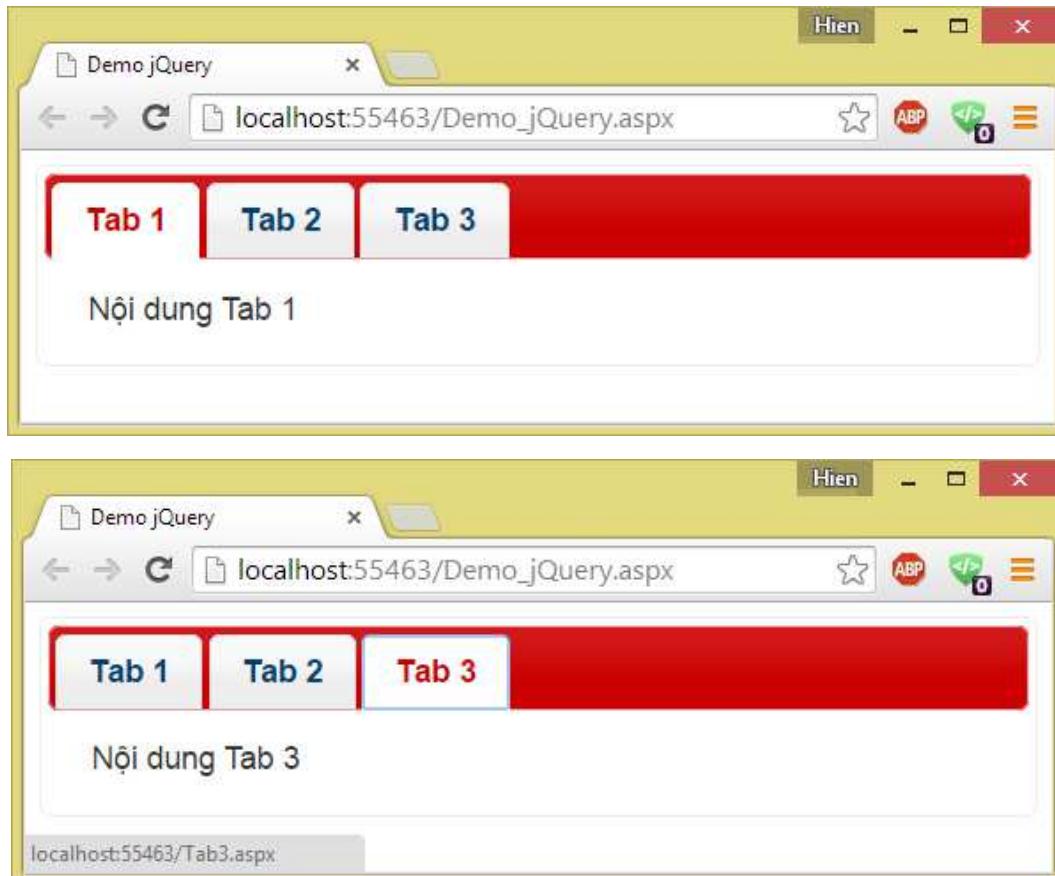
- ✓ D: ngày
- ✓ W: tuần
- ✓ M: tháng
- ✓ Y: năm

Nếu thêm dấu – thì hiểu là trước ngày hiện tại và dấu + thì sau ngày hiện tại.

6.6.5 JQuery Tabs

Thành phần giao diện này rất quen thuộc với chúng ta. Mỗi tab gồm 2 phần là tiêu đề tab và nội dung tab. Nội dung tab có thể là một phần mã HTML cùng trang hoặc nội dung của một trang web khác được tải theo cơ chế Ajax.

Ví dụ sau đây chúng ta tạo ra giao diện 3 tab. Tab1 và Tab2 chỉ đến một phần HTML cùng trang còn Tab3 sẽ tại nội dung của trang Tab3.aspx.



Mã nguồn đoạn nhúng:

```
<div id="mytabs">
    <ul>
        <li><a href="#Tab1">Tab 1</a></li>
        <li><a href="#Tab2">Tab 2</a></li>
        <li><a href="Tab3.aspx">Tab 3</a></li>
    </ul>
    <div id="Tab1">Nội dung Tab 1</div>
    <div id="Tab2">Nội dung Tab 2</div>
</div>
<script>
    $(function () {
        $("#mytabs").tabs();
    });
</script>
```

6.6.6 JQuery Accordion

Thành phần giao diện này cũng gần giống như tabs. Mỗi mục gồm 2 phần là tiêu đề và nội dung. Tại một thời điểm chỉ hiển thị một mục. Tuy nhiên công dụng của thành phần này thường để chứa các menu chức năng để tiết kiệm diện tích trang web.



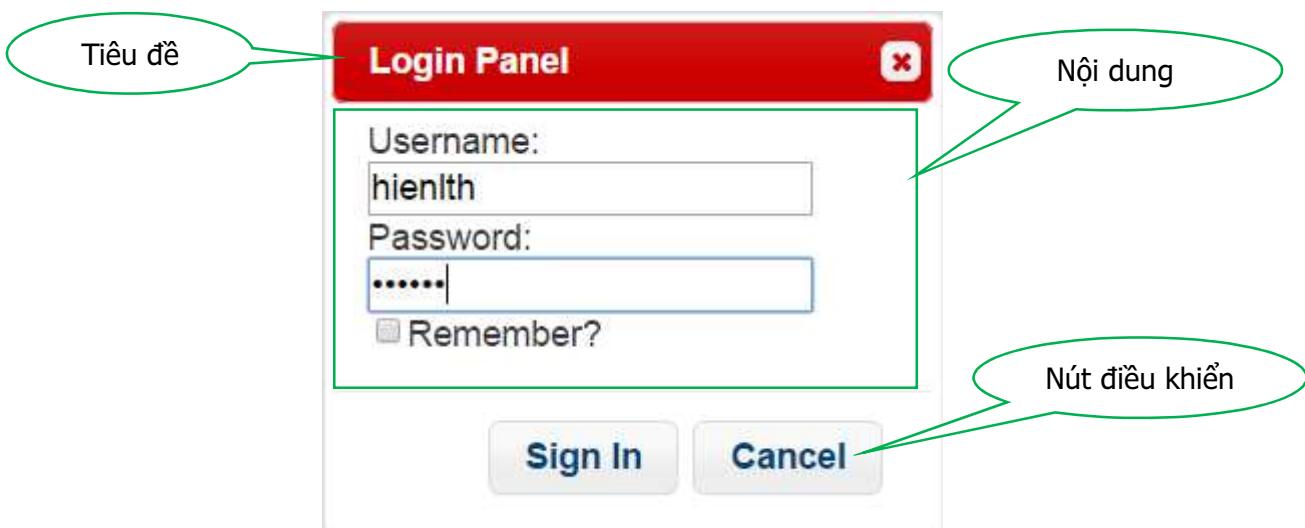
Mã nguồn đoạn nhúng:

```
<script>
$(function () {
    $(".my-acco").accordion();
});
</script>

<div class="my-acco">
    <h3>Tiêu đề mục 1</h3>
    <div>Nội dung mục 1</div>
    <h3>Tiêu đề mục 2</h3>
    <div>Nội dung mục 2</div>
    <h3>Tiêu đề mục 3</h3>
    <div>Nội dung mục 3</div>
</div>
```

6.6.7 JQuery Dialog

Thành phần này dùng để tạo hộp thoại tương tác với người dùng. Mỗi hộp thoại có tiêu đề, nội dung và thậm chí có cả hệ thống nút xử lý tương tác.



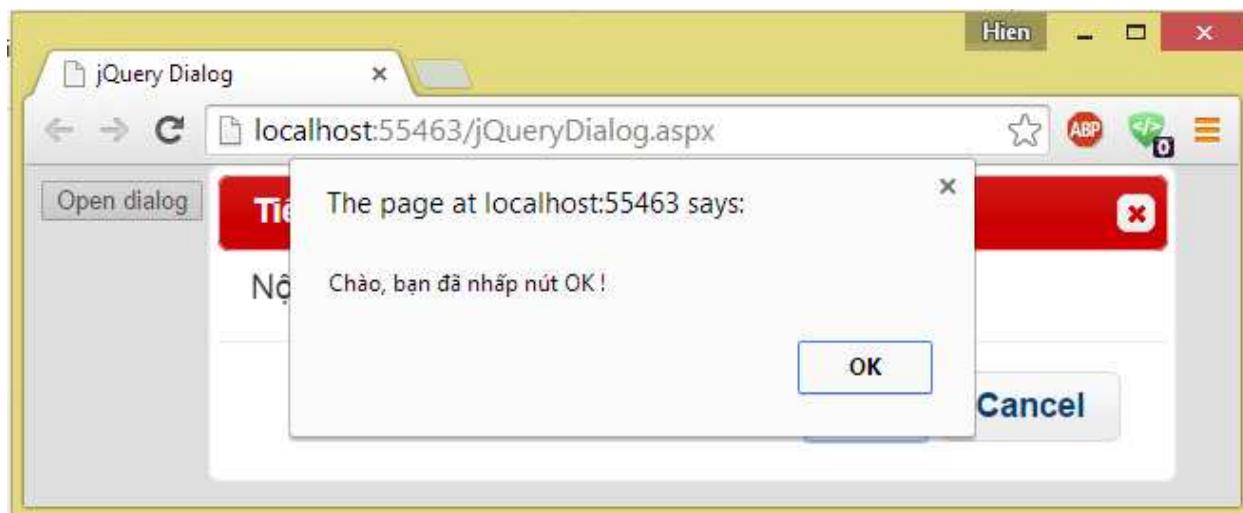
Có 3 công việc chính khi viết mã JQuery làm việc với hộp thoại là:

- ✓ Định nghĩa hộp thoại
- ✓ Mở hộp thoại
- ✓ Xử lý tương tác các nút điều khiển

Ví dụ sau đây cho thấy điều đó:



Sau khi nhấp nút Cancel hoặc dấu chéo góc trên-phải thì hộp thoại được đóng lại. Còn nếu nhấp vào nút OK sẽ nhận được thông báo



Mã nguồn đoạn nhúng:

```
<script>
$(function () {
    // công việc 1: định nghĩa hộp thoại
    $(".my-dialog").dialog({
        width: 500,
        autoOpen: false,
        modal: true,
        show: 'blind',
        hide: 'explode',
    });
});
```

```

buttons: {
    "OK": function () {
        // công việc 3: xử lý tương tác nút sự kiện
        alert("Chào, bạn đã nhấp nút OK !");
    },
    "Cancel": function () {
        // công việc 3: xử lý tương tác nút sự kiện
        $(".my-dialog").dialog("close");
    }
});

// công việc 2: mở hộp thoại
$(".my-button").click(function () {
    $(".my-dialog").dialog("open");
});
});
</script>

<div class="my-dialog" title="Tiêu đề hộp thoại">
    Nội dung hộp thoại
</div>

<input class="my-button" id="Button1" type="button" value="Open dialog" />

```

Qua ví dụ trên ta thấy công việc 1: định nghĩa hộp thoại có cú pháp sau

```
$(".my-dialog").dialog({thiết lập các tùy chọn});
```

Trong trường hợp này chúng ta sử dụng các tùy chọn với ý nghĩa sau

Tùy chọn	Mô tả
width	Định nghĩa chiều rộng của hộp thoại
autoOpen	Giá trị false sẽ không mở hộp thoại khi định nghĩa. Ngược lại giá trị true sẽ mở hộp thoại lúc định nghĩa.
modal	Không cho tương tác lên trang web khi hộp thoại hiện ra nếu là true. Ngược lại sẽ cho tương tác với cửa sổ mẹ.
show	Chỉ ra hiệu ứng lúc mở hộp thoại
hide	Chỉ ra hiệu ứng lúc đóng hộp thoại
buttons	Định nghĩa hệ thống nút điều khiển của hộp thoại

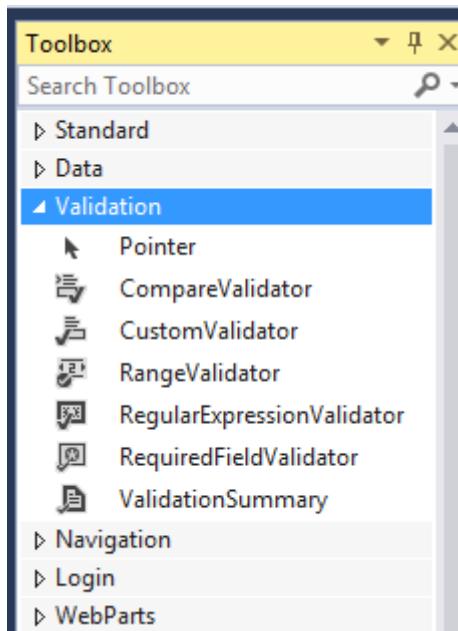
6.7 Validation Controls

Chúng ta đã học về những điều khiển chuẩn của .NET Framework. Chúng được dùng để tiếp nhận dữ liệu từ người dùng hoặc để trình bày dữ liệu cho người dùng xem. Vấn đề nằm ở chỗ làm thế nào để kiểm soát tính hợp lệ của dữ liệu nhập vào từ người dùng? Đó là câu hỏi sẽ được trả lời ở phần này.

Chúng ta có 2 phương pháp kiểm tra tính hợp lệ của dữ liệu là sử dụng các điều khiển kiểm lỗi của ASP.NET và sử dụng kỹ thuật kiểm lỗi của Jquery.

6.7.1 Kiểm lỗi của ASP.NET

ASP.NET cung cấp 6 điều khiển kiểm lỗi sau đây:



RequiredFieldValidator: yêu cầu người sử dụng nhập giá trị vào trường chỉ định trên Form

RangeValidator: kiểm tra giá trị nhập vào có nằm trong một khoảng nhỏ nhất và lớn nhất định trước hay không.

CompareValidator: so sánh giá trị nhập có bằng một giá trị cụ thể hay giá trị của một trường khác trên form hay không.

RegularExpressionValidator: kiểm tra định dạng của giá trị nhập vào có khớp với một biểu thức chính qui hay không...

CustomValidator: kiểm lỗi theo yêu cầu riêng của mình. Bạn có thể kiểm lỗi trên cả client và/hoặc server.

ValidationSummary: cho phép tập hợp và hiển thị lỗi.

Khi xây dựng ứng dụng, chúng ta nên kiểm tra dữ liệu nhập từ người dùng để hạn chế các sai sót dữ liệu nhập nhằm đảm bảo việc thực hiện xử lý dữ liệu được chính xác theo các yêu cầu nghiệp vụ. Nếu chúng ta viết mã để kiểm tra phải mất nhiều thời gian (sử dụng JavaScript hoặc VBScript). ASP.NET Webform hỗ trợ các validation controls để kiểm tra dữ liệu nhập từ người dùng trong các sever controls, mục đích là tránh để người dùng nhập sai hoặc không được bỏ trống các thông tin quan trọng bắt buộc,...

Như các bạn đã biết, mỗi khi PostBack về Server, trang Web luôn kiểm tra tính hợp lệ dữ liệu (nếu có yêu cầu khi thiết kế). Nếu dữ liệu không hợp lệ (bỏ trống, vi phạm miền giá trị, mật khẩu nhập lại không đúng, ...), trang web sẽ không thể PostBack về Server.

➤ Các thuộc tính chung của các validation control

Thuộc tính	Mô tả
ControlToValidate	Tên điều khiển cần kiểm tra. Đây là thuộc tính mà các bạn phải xác định khi sử dụng Validation Control.
Text	Chuỗi thông báo xuất hiện khi có lỗi.
ErrorMessage	Chuỗi thông báo xuất hiện trong điều khiển Validation Summary. Giá trị này sẽ được hiển thị tại vị trí của điều khiển nếu chúng ta không gán giá trị cho thuộc tính Text.
Display	Qui định hình thức hiển thị: None: Không hiển thị thông báo lỗi (vẫn có kiểm tra dữ liệu) Static: Trong trường hợp không có vi phạm dữ liệu, điều khiển không có hiển thị nhưng vẫn chiếm vị trí như trong lúc thiết kế. Dynamic: Trong trường hợp không có vi phạm dữ liệu, điều khiển không chiếm dụng vị trí trên màn hình.
EnableClientScript	Có cho phép thực hiện kiểm tra ở phía Client hay không ?. Giá trị mặc định là true - có kiểm tra.

6.7.2 Điều khiển RequiredFieldValidator

Điều khiển này được dùng để kiểm tra giá trị trong điều khiển phải được nhập. Sử dụng điều khiển này để kiểm tra ràng buộc dữ liệu khác rỗng (bắt buộc nhập).

Thuộc tính	Mô tả
InitialValue	Giá trị khởi động. Giá trị bạn nhập vào phải khác với giá trị của thuộc tính này. Giá trị mặc định của thuộc tính này là chuỗi rỗng.

6.7.3 Điều khiển CompareValidator

Điều khiển này được dùng để so sánh giá trị của một điều khiển với giá trị của một điều khiển khác hoặc một giá trị được xác định trước.

Thông qua thuộc tính Operator, chúng ta có thể thực hiện các phép so sánh như: =, <>, >, >=, <, <= hoặc dùng để kiểm tra kiểu dữ liệu (DataTypeCheck).

Sử dụng điều khiển này để kiểm tra ràng buộc miền giá trị, kiểu dữ liệu, liên thuộc tính.

Lưu ý: Trong trường hợp không nhập dữ liệu, điều khiển sẽ không thực hiện kiểm tra vi phạm.

Thuộc tính	Mô tả
ControlToCompare	Tên điều khiển cần so sánh giá trị. Nếu bạn chọn giá trị của thuộc tính Operator = DataTypeCheck thì không cần phải xác định giá trị cho thuộc tính này
Operator	Qui định phép so sánh, kiểm tra kiểu dữ liệu <ul style="list-style-type: none"> ✓ Equal: = (Đây là giá trị mặc định) ✓ GreaterThan: > ✓ GreaterThanEqual: >= ✓ LessThan: < ✓ LessThanEqual: <= ✓ NotEqual: <> ✓ DataTypeCheck: Kiểm tra kiểu dữ liệu
Type	Qui định kiểu dữ liệu để kiểm tra hoặc so sánh. <ul style="list-style-type: none"> ✓ String ✓ Integer ✓ Double ✓ Date ✓ Currency
ValueToCompare	Giá trị cần so sánh. Trong trường hợp bạn xác định giá trị của cả 2 thuộc tính ControlToCompare và ValueToCompare thì giá trị của điều khiển được qui định bởi thuộc tính ControlToCompare được ưu tiên dùng để kiểm tra

6.7.4 Điều khiển Range Validator

Điều khiển này được dùng để kiểm tra giá trị trong điều khiển phải nằm trong đoạn [min-max]. Sử dụng điều khiển này để kiểm tra ràng buộc miền giá trị của dữ liệu.

Lưu ý: Trong trường hợp không nhập dữ liệu, điều khiển sẽ không thực hiện kiểm tra vi phạm.

Thuộc tính	Mô tả
MinimumValue	Giá trị nhỏ nhất
MaximumValue	Giá trị lớn nhất
Type	Xác định kiểu để kiểm tra dữ liệu. Ta có thể thực hiện kiểm tra trên các kiểu dữ liệu sau: ✓ String ✓ Integer ✓ Double ✓ Date ✓ Currency

6.7.5 Điều khiển Regular Expression Validator

Điều khiển này được dùng để kiểm tra giá trị của điều khiển phải theo mẫu được qui định trước: địa chỉ email, số điện thoại, mã vùng, số chứng minh thư, ...

Lưu ý: Trong trường hợp không nhập dữ liệu, điều khiển sẽ không thực hiện kiểm tra vi phạm.

Thuộc tính	Mô tả
ValidationExpression	Qui định mẫu kiểm tra dữ liệu

Bảng mô tả các ký hiệu thường sử dụng trong Validation Expression

Ký hiệu	Mô tả
A	Ký tự chữ cái (đã được xác định). Ở đây là chữ a
1	Ký tự số (đã được xác định). Ở đây là số 1
[abc]	Một ký tự a, b hoặc c
[ac-f]	Một Ký tự a, c, d, e hoặc f
[^abc]	Một ký tự không phải a, b hoặc c
.	Một ký tự bất kỳ
\.	Ký tự thay thế phải là dấu chấm câu (.)
\w	Tương đương [a-zA-Z0-9_]
\d	Tương đương [0-9]
	Lựa chọn mẫu này hoặc mẫu khác
{n,m}	Xuất hiện ít nhất n lần và nhiều nhất m lần
{n}	Xuất hiện đúng n lần
?	Tương đương {0,1}
*	Qui định số lần xuất hiện: 0 hoặc nhiều lần
+	Qui định số lần xuất hiện: 1 hoặc nhiều lần (ít nhất là 1)

^	Bắt đầu
\$	Kết thúc

6.7.6 Điều khiển CustomValidator

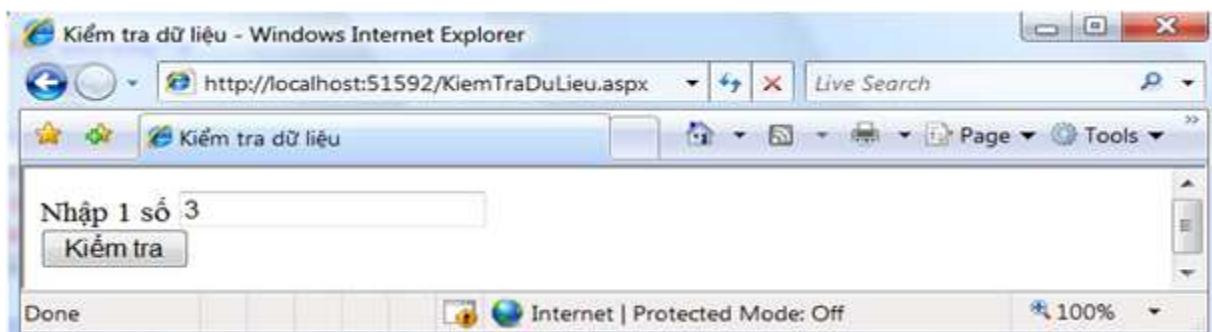
Điều khiển này cho phép bạn tự viết hàm xử lý kiểm tra lỗi theo ý thích cả 2 phía client và server.
Thông thường việc kiểm lỗi phía server xảy ra sau cùng khi các lỗi client đã được kiểm là hợp lệ.

6.7.6.1 Kiểm lỗi phía server

Dựa vào sự kiện **ServerValidate** để cài đặt mã kiểm lỗi phía server.

Ví dụ: Minh họa xử lý kiểm tra dữ liệu nhập tại điều khiển txtSoNguyen có phải là số chẵn hay không?

Bước 1 : Tạo 1 trang KiemTraDuLieu.aspx có giao diện như sau



Mã ASP.NET của trang như sau

```

KiemTraDuLieu.aspx.cs KiemTraDuLieu.aspx* Start Page
Client Objects & Events (No Events)

<body>
    <form id="form1" runat="server">
        <div>
            Nhập 1 số
            <asp:TextBox ID="txtSoNguyen" runat="server"></asp:TextBox>
            <asp:CustomValidator ID="cvKiemTraSoChan" runat="server"
                ControlToValidate="txtSoNguyen" ErrorMessage="*"
                OnServerValidate="cvKiemTraSoChan_ServerValidate">
                Đây không phải là số chẵn !
            </asp:CustomValidator>
            <br />
            <asp:Button ID="btnKiemTra" runat="server" Text="Kiểm tra" />
        </div>
    </form>
</body>



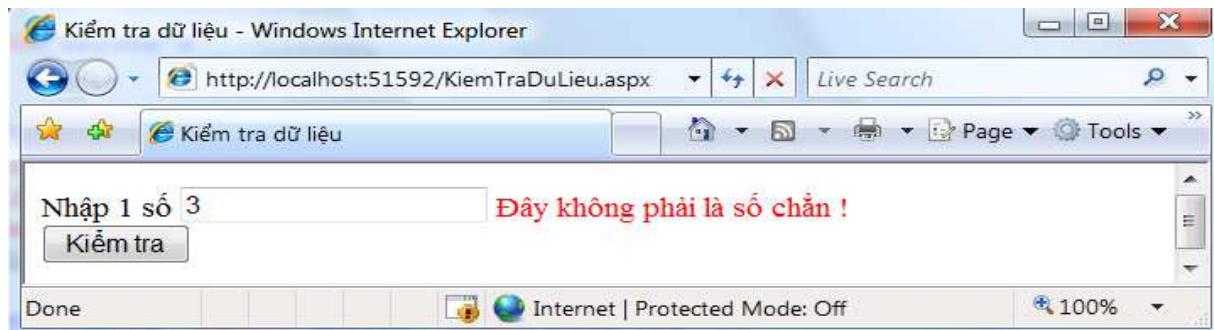
Nhập 1 số 
    Đây không phải là số chẵn !


```

Bước 2 : Mã kiểm lỗi được viết trong sự kiện ServerValidate như sau

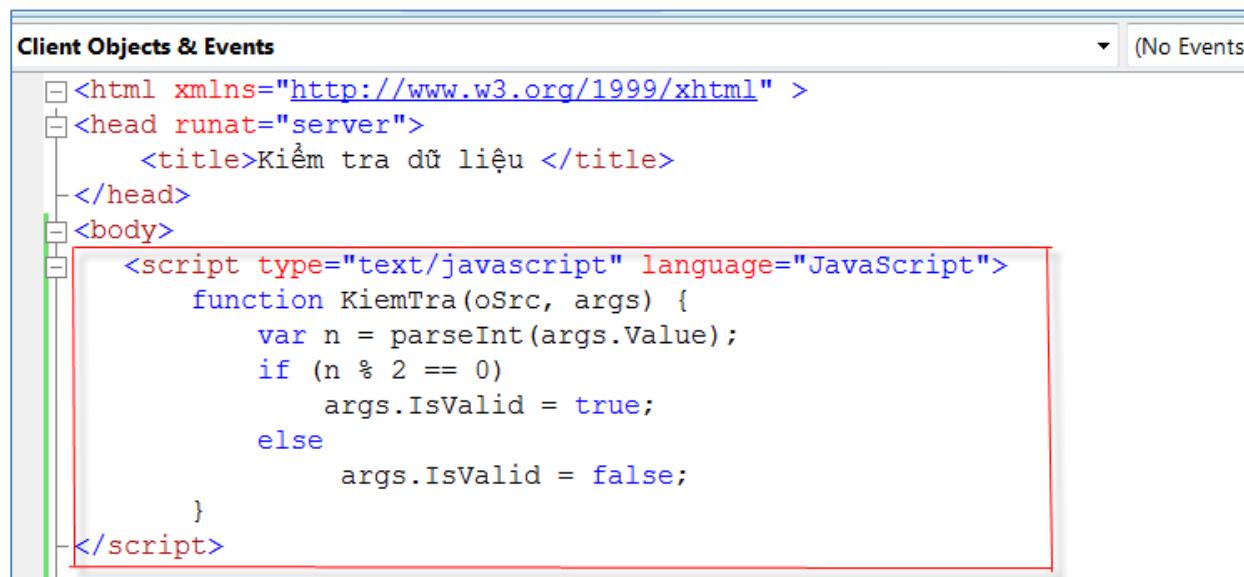
```
public partial class KiemTraDuLieu : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void cvKiemTraSoChan_ServerValidate(object source,
        ServerValidateEventArgs args)
    {
        int n = int.Parse(args.Value);
        if (n % 2 == 0)
            args.IsValid = true;
        else
            args.IsValid = false;
    }
}
```

Bước 3 : Chạy, nhập số 3 và nhấp nút Kiểm tra sẽ nhận được thông báo



6.7.6.2 Kiểm lỗi phía client

Chúng ta có thể viết mã JavaScript kết hợp với CustomValidator để kiểm lỗi phía client. Sau đây là minh họa kiểm tra tại phía Client.



```

<form id="form1" runat="server">
<div>
    Nhập 1 số
    <asp:TextBox ID="txtSoNguyen" runat="server"></asp:TextBox>
    <asp:CustomValidator ID="cvKiemTraSoChan" runat="server"
        ControlToValidate="txtSoNguyen" ErrorMessage="*"
        ClientValidationFunction="KiemTra">
        Đây không phải là số chẵn !
    </asp:CustomValidator>
    <br />
    <asp:Button ID="btnKiemTra" runat="server" Text="Kiểm tra" />
</div>
</form>
</body>
</html>

```

Lưu ý : Khi kiểm tra tại phía Server, sau khi chúng ta nhập vào số nguyên và nhấn **Kiểm tra** để xem kết quả nhưng khi kiểm tra tại phía Client chúng ta chỉ cần nhập vào số nguyên và nhấp chuột ra khỏi TextBox để xem kết quả.

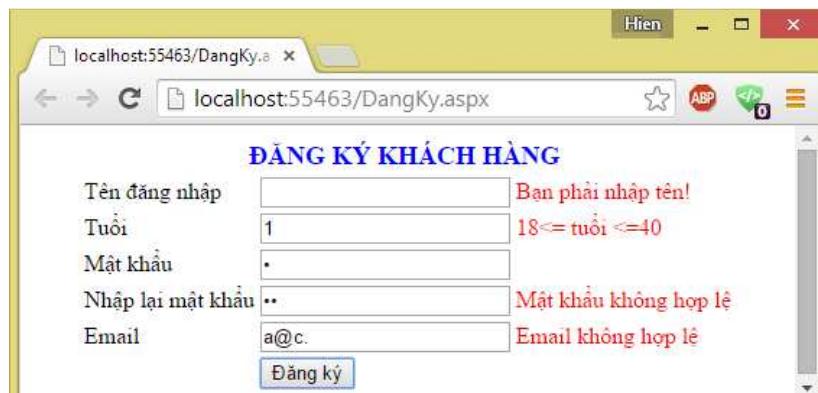
6.7.7 Điều khiển Validation Summary

Điều khiển này được dùng để hiển thị ra bảng lỗi - tất cả các lỗi hiện có trên trang Web. Nếu điều khiển nào có dữ liệu không hợp lệ, chuỗi thông báo lỗi - giá trị thuộc tính ErrorMessage của Validation Control sẽ được hiển thị. Nếu giá trị của thuộc tính ErrorMessage không được xác định, thông báo lỗi đó sẽ không được xuất hiện trong bảng lỗi.

Thuộc tính	Mô tả
HeaderText	Dòng tiêu đề của thông báo lỗi
ShowMessageBox	Qui định bảng thông báo lỗi có được phép hiển thị như cửa sổ MessageBox hay không. Giá trị mặc định của thuộc tính này là False - không hiển thị
ShowSummary	Qui định bảng thông báo lỗi có được phép hiển thị hay không. Giá trị mặc định của thuộc tính này là True - được phép hiển thị.

Ví dụ : Minh họa sử dụng các validation control

Tạo 1 trang ASP.Net tên DangKy.aspx với giao diện và điều kiện ràng buộc như sau :



Các ràng buộc tính hợp lệ của dữ liệu được mô tả như sau:

Tên trường	Mô tả
Tên đăng nhập	Không được bỏ trống
Tuổi	Phải nằm trong khoảng từ 18 đến 40
Nhập lại mật khẩu	Phải chính xác như mật khẩu
Email	Phải đúng dạng email

Bảng mô tả các thuộc tính của các controls

STT	Control	Tên thuộc tính	Giá trị thuộc tính
1	TextBox	ID	txtTenDangNhap
2	TextBox	ID	txtTuoi
3	TextBox	ID	txtMatKhau
4	TextBox	ID	txtNhapLaiMatKhau
5	TextBox	ID	txtEmail
6	RequiredFieldValidator	ControlToValidate	txtTenDangNhap
		ErrorMessage	Bạn phải nhập tên!
7	RequiredFieldValidator	ControlToValidate	txtNhapLaiMatKhau
		ErrorMessage	Bạn phải nhập lại mật khẩu!
8	RangeValidator	ControlToValidate	txtTuoi
		ErrorMessage	18<= tuổi <=40
		MaximumValue	40
		MinimumValue	18
		Type	Integer
9	CompareValidator	ControlToValidate	txtNhapLaiMatKhau
		ControlToCompare	txtMatKhau
		ErrorMessage	Mật khẩu không hợp lệ
10	RegularExpressionValidator	ControlToValidate	txtEmail
		ValidationExpression	Nhấp vào nút ... và chọn Internet e-mail address.
		ErrorMessage	Email không hợp lệ

Sau khi chạy trang DangKy.aspx, nếu ta nhập vào giá trị không hợp lệ và nhấn nút **Đăng ký** thì sẽ xuất hiện thông báo lỗi tương ứng như hình trên.

Chú ý: Nếu tạo dạng **New Project** thì từ VS2012 trở lên cần thêm đoạn sau vào **Web.config**:

```
<appSettings>
  <add key="ValidationSettings:UnobtrusiveValidationMode" value="None" />
</appSettings>
```

6.8 Kiểm lỗi với JQuery

Kiểm duyệt dữ liệu form nhập trước khi chuyển dữ liệu đến server để xử lý là nhiệm vụ cực kỳ quan trọng. Rất nhiều tiêu chí được đặt ra để kiểm duyệt tùy thuộc vào yêu cầu cụ thể của form nhập liệu. Dù sao vẫn có thể liệt kê một số tiêu chí kiểm duyệt chung chung như sau:

- ✓ Không cho để trống ô nhập...
- ✓ Dữ liệu nhập vào phải theo một khuôn dạng nhất định nào đó: email, creditcard, url...
- ✓ Dữ liệu phải nhập vào phải đúng kiểu: số nguyên, số thực, ngày giờ...
- ✓ Dữ liệu nhập vào phải có giá trị tối thiểu, tối đa, trong phạm vi...
- ✓ Dữ liệu nhập phải đúng theo một kết quả tính toán riêng của bạn...

Bây giờ chúng ta hãy khám phá khả năng kiểm duyệt dữ liệu đầu vào của Jquery.

Để sử dụng jQuery Validation, bạn vào trang <http://jqueryvalidation.org/> để tải lấy bản mới nhất.

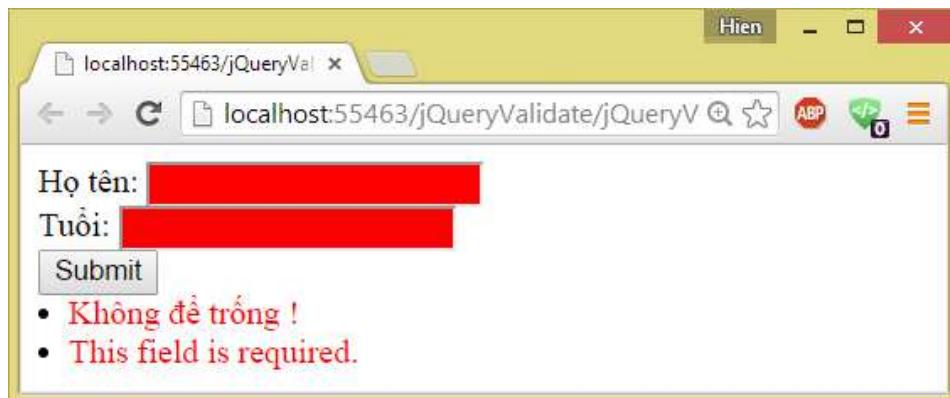
6.8.1 Khởi động nhanh với Jquery Validation

Ví dụ sau đây kiểm tra ô nhập "Tên" phải nhập ít nhất 3 ký tự và ô nhập "Tuổi" phải nhập số từ 20 đến 60.

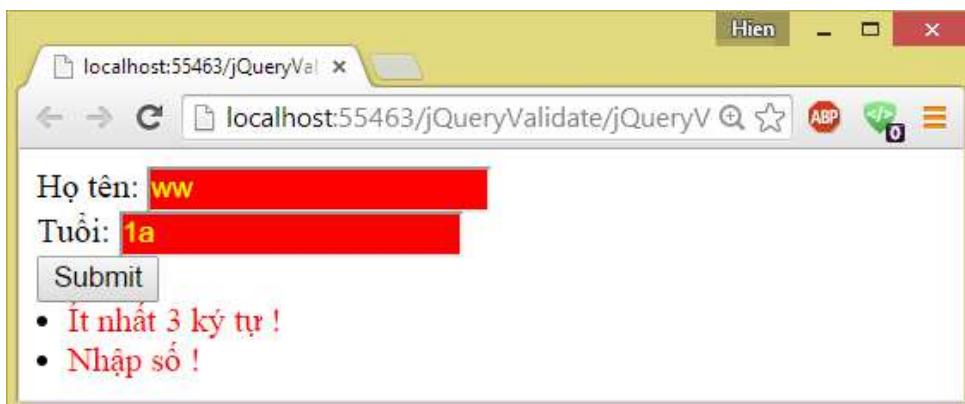
```
<html>
<head>
<script src="jquery-1.11.2.js" type="text/javascript"></script>
<script src="jquery.validate.js" type="text/javascript"></script>
<script type="text/javascript">
$(document).ready(function () {
    $("#form1").validate(
    {
        rules:
        {
            txtName: { required: true, minlength: 3 },
            txtAge: { required: true, digits: true, range: [20, 60] }
        },
        messages:
        {
            txtAge: { digits: "Nhập số !", range: "Chỉ nhập từ 20 → 60 !" },
            txtName: { required: "Không để trống !",
                       minlength: "Ít nhất 3 ký tự !" }
        },
        errorLabelContainer: "#myError",
        wrapper: "li",
        submitHandler: function (form) {
            if (confirm("Dữ liệu form đã hợp lệ. Bạn có muốn submit không ?")) {
                form.submit();
            }
        }
    });
});
</script>
<style type="text/css">
label.error{color:Red;}
input.error { background-color:Red; color:yellow;}
</style>
</head>
<body>
<form id="form1" name="form1" method="post" action="">
<p>
    Name:
    <input name="txtName" type="text" id="txtName" />
</p>
<p>
    Age:
    <input name="txtAge" type="text" id="txtAge" />
</p>
<p>
    <input type="submit" name="Submit" value="Submit" />
</p>
<div id="myError" />
</form>
</body>
</html>
```

Chạy ứng dụng với các tình huống sau

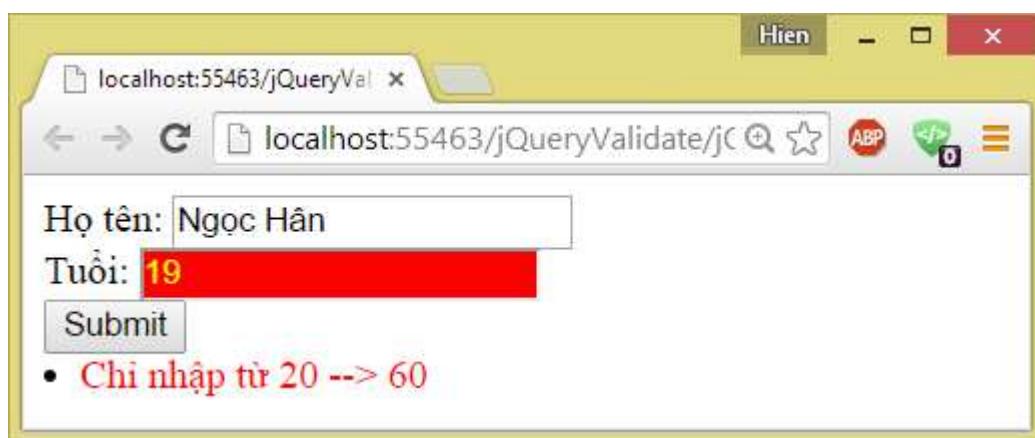
- ✓ Để trống các ô và nhấp nút Submit



- ✓ Nhập dữ liệu không hợp lệ và nhấp nút Submit



Nhập tuổi không liệu hợp lệ và nhấp nút Submit



Phân tích:

Thư viện cần thiết cho việc bẩy lỗi:

```
<script src="jquery.validate.js" type="text/javascript"></script>
Cấu trúc cơ bản của phương thức validate() dùng để cài đặt các tùy chọn bẩy lỗi.
```

```
<script type="text/javascript">
$(document).ready(function () {
```

```

$( "#form1" ).validate(
{
    rules: {<khai báo luật bẩy lỗi cho các trường>},
    messages: {<định nghĩa các thông báo lỗi>},
    errorLabelContainer: ,<khai báo thẻ chứa lỗi>',
    wrapper: ,<khai báo thẻ bọc lỗi>',
    submitHandler: <hàm xử lý submit>
});
});
</script>

```

Trong bài này:

- ✓ Khai báo luật bẩy lỗi cho các trường

```

rules:
{
    txtName: { required: true, minlength: 3 },
    txtAge: { required: true, digits: true, range: [20,60] }
}

```

- txtName: không được để trống, phải có ít nhất 3 ký tự
- txtAge: không được để trống, phải là số nguyên và thuộc khoảng (20, 60)

- ✓ Định nghĩa các thông báo lỗi

Sau đây là danh sách các luật kiểm lỗi trong JQuery

Luật	Mô tả	Ví dụ
required	Bắt buộc nhập	required:true
required	Bắt buộc nhập nếu tập kết quả của selector rỗng	required:"#chkHobby:blank"
required	Bắt buộc nhập nếu kết quả trả về có giá trị false.	required: function(){return true;}
email	Định dạng email	email:true
url	Định dạng url	url:true
date	Định dạng ngày javascript	date:true
number	Số thực	number:true
digits	Số nguyên	digits:true
creditcard	Định dạng creditcard	creditcard:true
minlength	Số ký tự tối thiểu	minlength:10
maxlength	Số ký tự tối đa	maxlength:100
rangelength	Số ký tự từ min đến max	rangelength:[10, 100]
min	Giá trị tối thiểu	min:10
max	Giá trị tối thiểu	max:100
range	Giá trị từ min đến max	range:[10,100]
accept	Kiểu mở rộng file	accept:"doc xsl pdf"
equalTo	So sánh giá trị của phần tử và giá trị của selector	equalTo:"#txtPassword"
remote	Hợp lệ khi kết quả kiểm tra từ xa là false.	remote: "check.aspx"

Chú ý: bạn có 2 cách để khai báo luật bẩy lỗi

- Khai báo trong tùy chọn **rules** như trong ví dụ trên
- Khai báo ngay trong thẻ bạn muốn bẩy lỗi

Ví dụ để kiểm lỗi cho ô nhập txtAge của ví dụ trên, bạn có thể khai báo ngay trên thẻ <input> như sau:

```
<input class="required digits" min="25" max="65" id="txtAge" />
```

6.8.2 Luật kiểm lỗi do người dùng định nghĩa

Trên đây chỉ là danh sách các luật phổ thông hàng ngày. Bạn có thể có những qui luật riêng của mình mà chỉ có bạn mới có thể hiểu và định nghĩa được. Vì vậy Jquery cung cấp cho bạn một cách định nghĩa các luật mới của riêng mình. Hãy xem và phân tích ví dụ sau để hiểu rõ cách để định nghĩa một luật mới.

```
<html>
<head>
<script src="jquery-1.11.2.min.js"></script>
<script src="jquery.validate.js"></script>
<script type="text/javascript">

/*--Định nghĩa hàm kiểm tra số di động việt nam--*/
function fnValidateMobile(value, element) {
    var regex = /(^0[0-9]{9,10}$)/g;
    return this.optional(element) || regex.test(value);
}

/*--Định nghĩa hàm kiểm tra số xe gắn máy sài gòn--*/
function fnValidateSaigonMoto(value, element) {
    var regex = (/^5\d-[A-Z]\d-\d{4}$/g);
    return this.optional(element) || regex.test(value);
}

/*--Định nghĩa hàm kiểm tra IP mạng máy tính--*/
function fnValidateNetuworkIP(value, element) {
    var regex = (/^\d{3}\.\d{3}\.\d{3}\.\d{3}$)/g;
    if (this.optional(element) || regex.test(value)) {
        var nums = value.split(".");
        for (var i = 0; i < nums.length; i++) {
            if (parseInt(nums[i]) > 255) {
                return false;
            }
        }
    } else {
        return false;
    }
    return true;
}

/*--Định nghĩa hàm kiểm tra mục chọn của combo box--*/
function fnValidateSelectOne(value, element) {
    return (element.value != "none");
}

/*--Định nghĩa luật kiểm tra kết hợp với hàm và một thông
báo lỗi nếu kết quả trả về của hàm có giá trị false--*/
$.validator.addMethod("selectone", fnValidateSelectOne, "Please select an item.");
$.validator.addMethod("vinaphone", fnValidateMobile, "Please enter a valid VinaPhone number.");
$.validator.addMethod("saigonmoto", fnValidateSaigonMoto, "Please enter a valid Saigon moto
number.");
$.validator.addMethod("networkip", fnValidateNetuworkIP, "Please enter valid a network IP.");
</script>
<script type="text/javascript">
$(document).ready(function () {
```

```

        $("#form1").validate(
        {
            rules:
            {
                sport: { selectone: true },
                mobile: { vinaphone: true }
            },
            messages:
            {
                sport: { selectone: "Vui lòng chọn môn thể thao" },
                mobile: { vinaphone: "Không phải số di động ở Việt nam" }
            }
        });
    });
</script>
<style type="text/css">
label.error
{
    color: Red;
}
</style>
</head>
<body>
    <h1>Luật kiểm tra tùy biến</h1>
    <form id="form1">
        Số xe máy Sài gòn:
        <input type="text" id="moto" name="moto" class="required saigonmoto">

        Địa chỉ server:
        <input type="text" id="ip" name="ip" class="networkip">

        Số điện thoại di động:
        <input type="text" id="mobile" name="mobile">

        Thể thao:
        <select name="sport" id="sport">
            <option value="none">Chọn môn thể thao</option>
            <option value="baseball">Bóng chày</option>
            <option value="basketball">Bóng rổ</option>
            <option value="volleyball">Bóng chuyền</option>
            <option value="football">Bóng đá</option>
        </select>

        <input class="submit" type="submit" value="Validate">
    </form>
</body>
</html>

```

Trong bài trên chúng ta định nghĩa 4 luật kiểm tra mới là **vinaphone**, **saigonmoto**, **networkip** và **selectone**. Và sau đó áp dụng để kiểm tra dữ liệu cho các thành phần giao diện trên form. Để hiểu được cơ chế định nghĩa và sử dụng chúng ta cần thực hiện các bước sau.



Bước 1: Định nghĩa. cần 2 bước là viết hàm kiểm tra và khai báo luật kiểm với Jquery

✓ **Viết hàm kiểm tra:**

```
/*--Định nghĩa hàm kiểm tra số di động việt nam--*/
function fnValidateMobile(value, element) {
    var regex = /^0[0-9]{9,10}$/g;
    return this.optional(element) || regex.test(value);
}
```

Cú pháp của hàm này phải nhận 2 tham số vào là value (giá trị nhập vào) và element (phần tử gây lỗi). Hàm này phải trả về kết quả true (đã hợp lệ) hoặc false (không hợp lệ). Bạn có thể phân tích giá trị để thực hiện kiểm tra nhờ vào tham số value và thay đổi css hay giá trị của phần tử này thông qua tham số element.

✓ **Khai báo luật kiểm với Jquery**

Sau khi đã định nghĩa hàm kiểm tra, bước tiếp theo là định nghĩa luật kiểm tương ứng với hàm trên và tất nhiên cung cấp thông báo lỗi.

```
/*--Định nghĩa luật kiểm tra kết hợp với hàm, thông báo lỗi nếu kết quả trả về của hàm là false--*/
$.validator.addMethod("vinaphone", fnValidateMobile,
```

"Please enter a valid VinaPhone number."

Sử dụng phương thức `$.validator.addMethod(rule, method, message)` để khai báo luật kiểm. Tham số rule ("vinaphone") là tên luật mới, tham số method ("fnValidateMobile") là tên phương thức kết hợp với luật mới và message ("Please enter a valid VinaPhone number.") là thông báo lỗi.

Bước 2: Sử dụng

Bạn sử dụng các luật mới như các luật đã định nghĩa sẵn trong Jquery. Cụ thể là bạn có thể chỉ định trong tùy chọn rules (rules: {mobile: { vinaphone: true }}) của phương thức validate hoặc chỉ ra trên thẻ cần kiểm tra "`<input type="text" id="moto" name="moto" class="required saigonmoto">`".

Chương 7: TỔ CHỨC WEBSITE

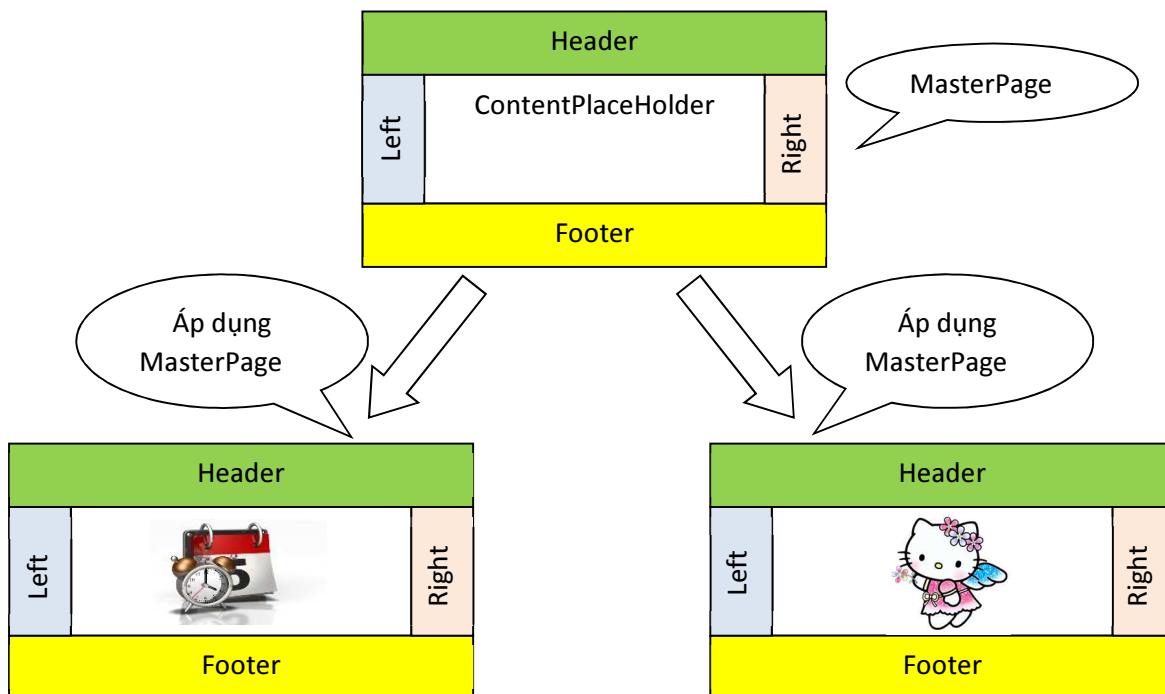
Sau khi học xong bài này, học viên có khả năng :

- Thiết kế và phát triển được trang Master Page , User Controls.
- Thiết kế và sử dụng Theme.

7.1 MasterPage

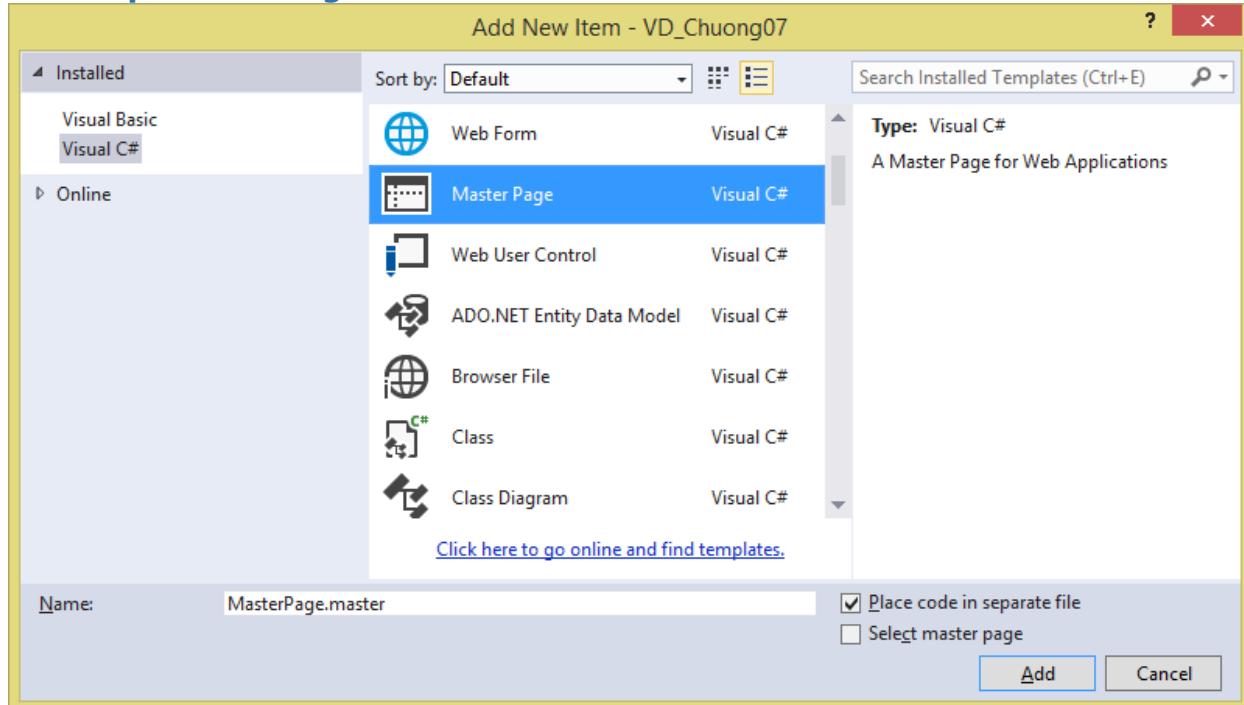
7.1.1 Giới thiệu

- Master Page là trang chứa những điều khiển sẽ được chia sẻ cho nhiều trang khác sử dụng lại. Chẳng hạn, trong web site có nhiều trang có cùng header, footer, banner, menu,... ta có thể xây dựng những thành phần này chỉ một lần trong Master Page, sau đó những trang có liên kết đến Master Page sẽ kế thừa lại những thành phần đó.
- MasterPage là khuôn mẫu giao diện được sử dụng để áp dụng cho một nhóm trang có cùng cấu trúc giao diện trong website.
- Trong mỗi MasterPage có chứa nhiều điều khiển **ContentPlaceHolder** – nơi mà nội dung của các trang áp dụng MasterPage này được phép thay đổi.
- Master Page giúp việc bảo trì web site dễ dàng, tránh trùng lắp code giữa các trang.
- Chúng ta có thể hình dung như hình sau:



Trong hình vẽ bên trên ta thấy chỉ có một vùng được phép thay đổi nội dung ở các trang thành viên. Nói cách khác, các phần Header, Footer, Left và Right gần như không đổi khi chúng ta duyệt qua các trang web áp dụng MasterPage này.

7.1.2 Tạo MasterPage



```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="MasterPage.master.cs"
Inherits="MasterPage" %>
```

```
<!DOCTYPE html>

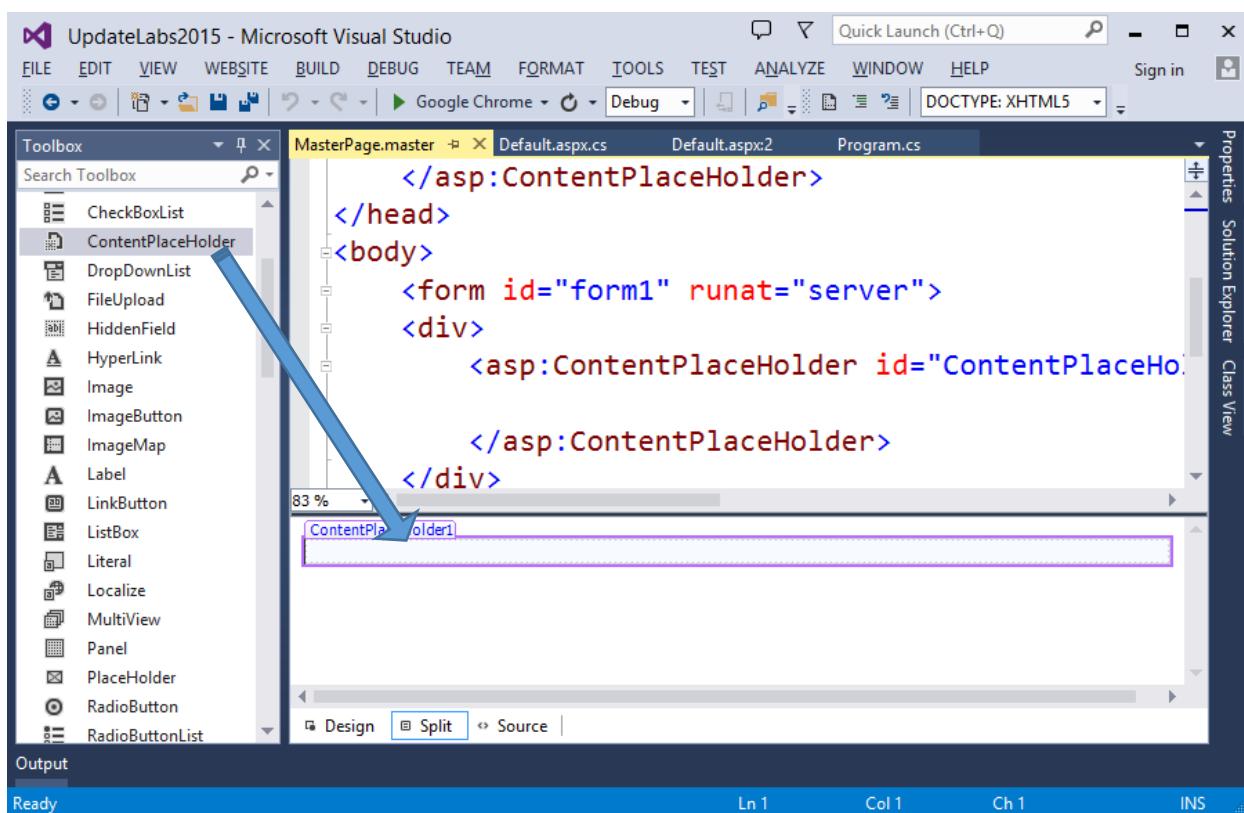
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <asp:ContentPlaceholder id="head" runat="server">
        </asp:ContentPlaceholder>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:ContentPlaceholder id="ContentPlaceholder1" runat="server">
                </asp:ContentPlaceholder>
            </div>
        </form>
    </body>
</html>
```

Trong MasterPage mới tạo ra đã có sẵn 2 vùng cho phép thay đổi (một trên thẻ `<head>` và một trong `<body>`). Thông thường thì chúng ta phải thay thế toàn bộ nội dung MasterPage theo một thiết kế chuyên nghiệp khác. Khi đó các bạn chỉ việc tập trung vào các công việc sau đây:

- ✓ Thay thế nội dung MasterPage bằng mã thiết kế (ngoại trừ chỉ thị `<%@Master...%>`).
- ✓ Thêm thuộc tính `runat="server"` vào thẻ `<head>`.
- ✓ Bọc thẻ `<form runat="server">` ở phần nội dung trang.
- ✓ Đánh dấu các vùng thay đổi ContentPlaceholder.

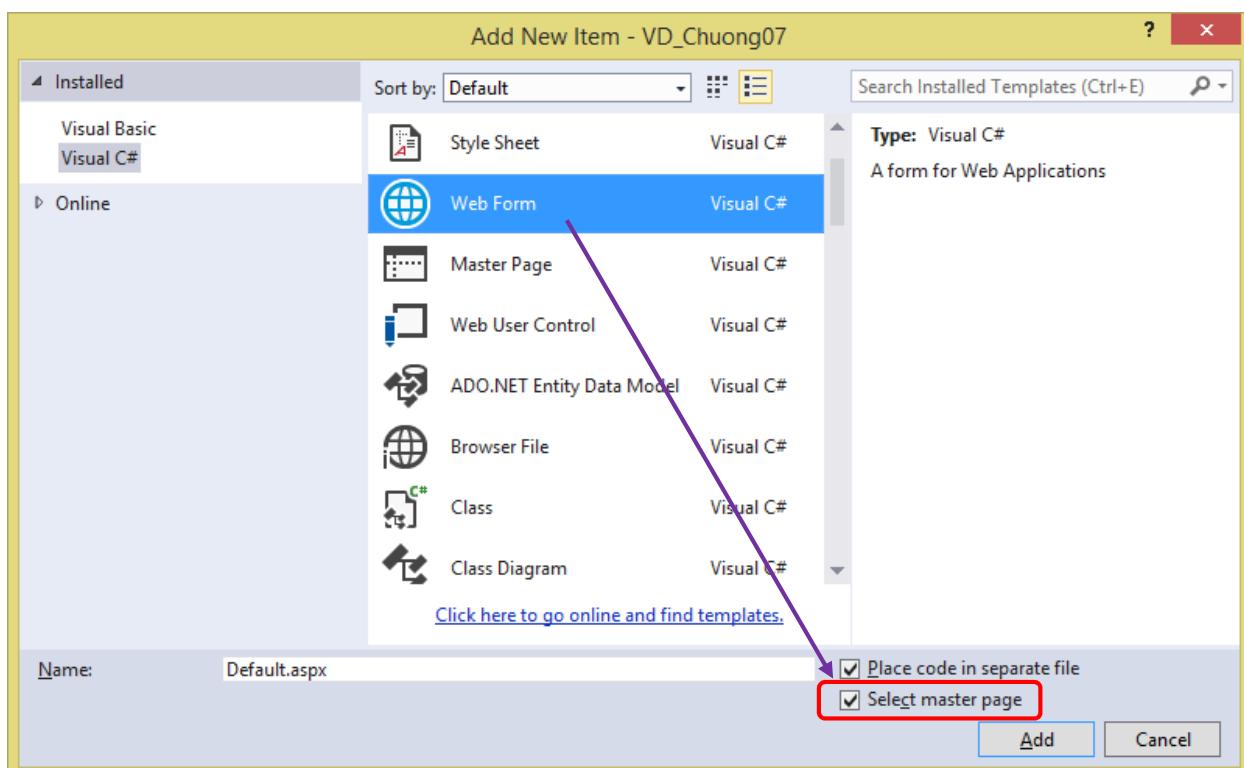
7.1.3 Đánh dấu vùng thay đổi (ContentPlaceholder)

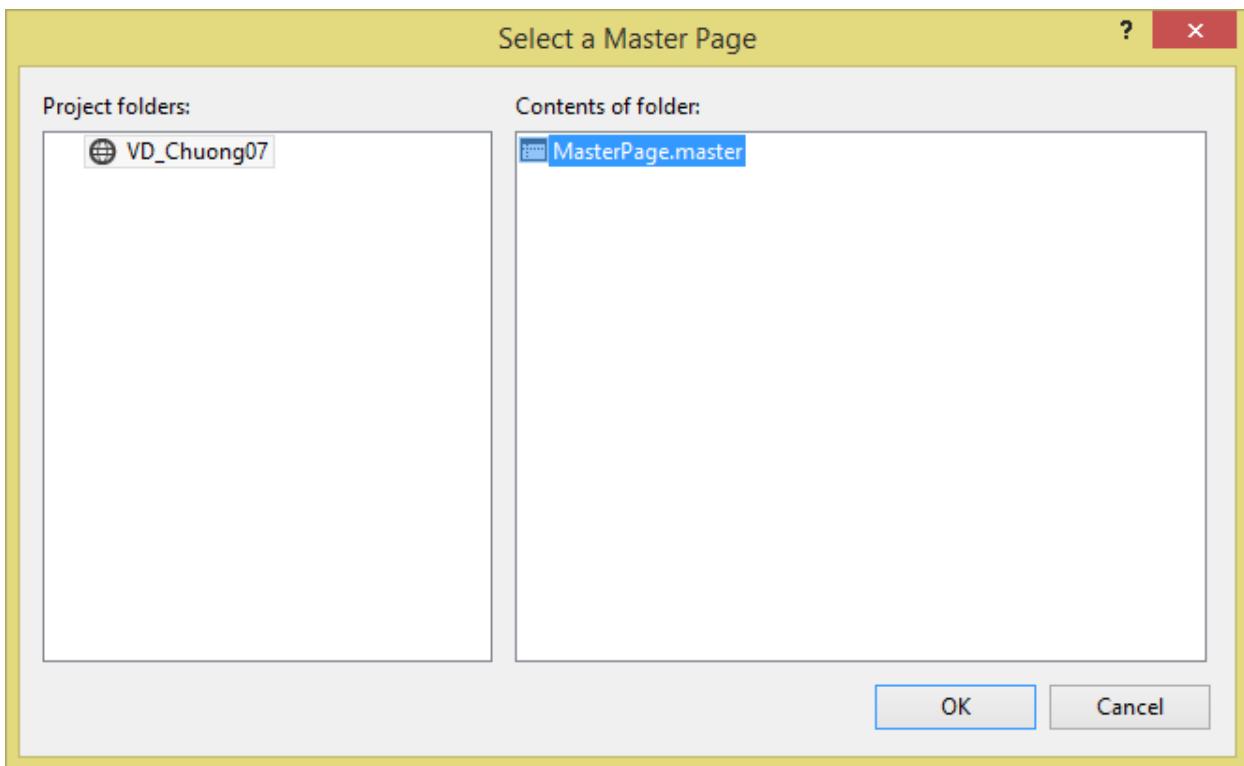
Để đánh dấu một vùng thay đổi được bạn chỉ việc kéo điều khiển ContentPlaceholder trên thanh công cụ vào vị trí mong muốn.



7.1.4 Áp dụng MasterPage

Áp dụng một MasterPage cho một trang ASP.NET rất đơn giản, chỉ việc tích vào checkbox select master page, sau đó lựa chọn master page để áp dụng.





Trang web được tạo ra có mã ASP.NET như sau

```
<%@ Page Title="" Language="C#" MasterPageFile="~/MasterPage.master"
AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
    <!--nội dung của vùng head-->
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceholder1"
Runat="Server">
    <!--nội dung của vùng body-->
</asp:Content>
```

Ta thấy rằng chỉ có các vùng thay đổi được xuất hiện trong các trang thành viên. Phần thiết kế giữa các điều khiển `<asp:Content>` sẽ được thay vào các vùng thay đổi đã giữ chỗ sẵn trong MasterPage.

7.2 Web User Control

7.2.1 Giới thiệu

Web User Control là các control do người dùng định nghĩa với 2 mục đích chính là:

- ✓ Tái sử dụng: control được tạo ra được dùng đi dùng lại nhiều lần trên một hoặc nhiều trang khác nhau.
- ✓ Tổ chức trang web theo phương pháp chia để trị: phân chia năng lực cho mỗi control này. Mã nguồn được tập trung vào các chức năng cụ thể mà control đảm nhiệm.

Web User Control thường được xây dựng từ một tổ hợp của nhiều control ASP.NET và được tổ chức theo một hình thức cụ thể.

Ví dụ:

BÌNH CHỌN WEBSITE

Tuyệt vời
 Rất tốt
 Tốt
 Thường
 Tồi



Bình chọn website

HỖ TRỢ TRỰC TUYẾN

Not Online right now
 Not Online right now

Hỗ trợ trực tuyến

HÀNG HÓA ĐẶC BIỆT

ĐẶC BIỆT NHẤT
YÊU THÍCH NHẤT
CHIA SẺ NHIỀU NHẤT
MUA NHIỀU NHẤT
XEM NHIỀU NHẤT
GIẢM GIÁ NHIỀU NHẤT
HÀNG MỚI NHẤT

Hàng hóa đặc biệt

0 X MÃT HÀNG
0,00 ₫

GIỎ HÀNG 7 6 5 4 1 7

Thông tin giỏ hàng điện tử

THƯ TIN

Newsletter

7.2.2 Tạo và sử dụng

Phải chuột lên website -> Add New Item

Add New Item - VD_Chuong07

Installed

Visual Basic
Visual C#

Online

	Sort by: Default		
	Web Form	Visual C#	Type: Visual C# An ASP.NET server control created using the visual designer
	Master Page	Visual C#	
	Web User Control	Visual C#	
	ADO.NET Entity Data Model	Visual C#	
	Browser File	Visual C#	
	Class	Visual C#	
	Class Diagram	Visual C#	

[Click here to go online and find templates.](#)

Name: MyWebUserControl.ascx

Place code in separate file
 Select master page

Add **Cancel**

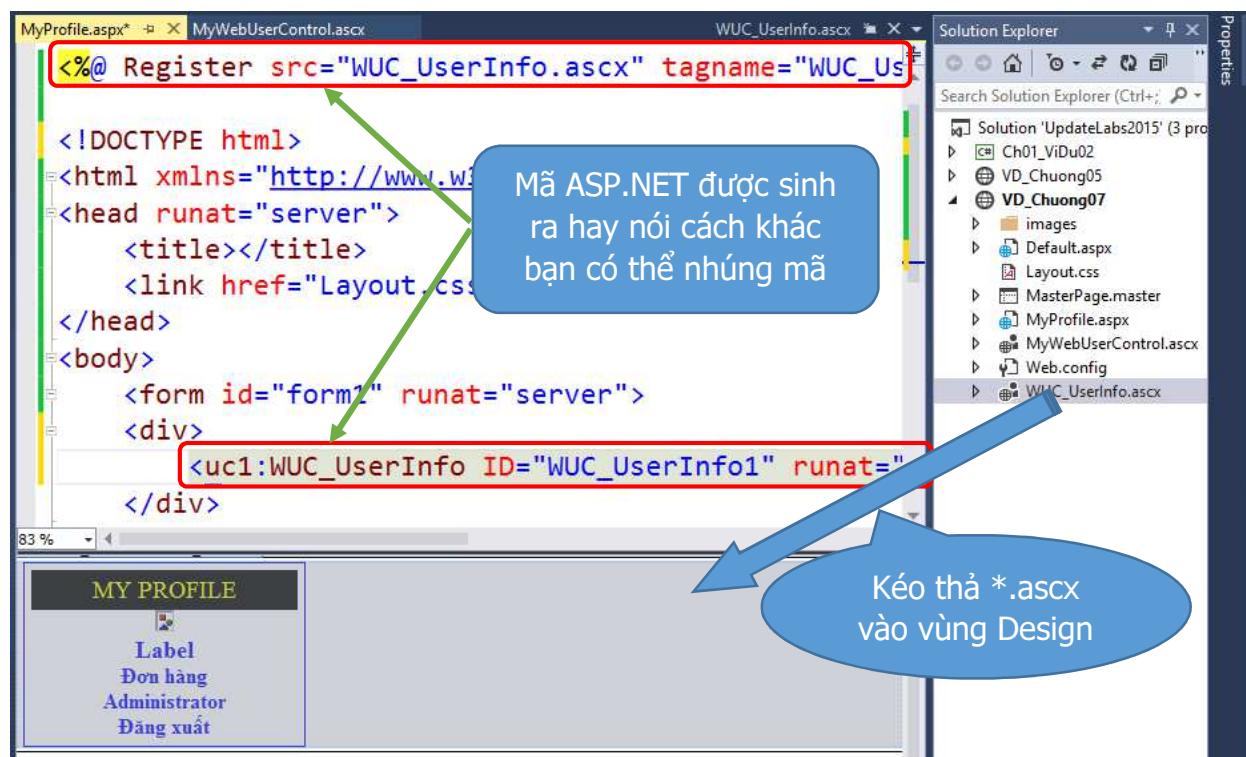
Một tập tin *.ascx được tạo ra có mã ASP.NET như sau

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

Chúng ta có thể thiết kế giao diện như làm việc với trang ASP.NET bình thường (viết mã hoặc/và kéo thả). Giao diện được tạo ra sẽ gom lại thành một mô-đun giao diện độc lập.



Bạn không thể chạy các tập tin *.ascx mà chỉ được bổ sung vào các trang ASP.NET, MasterPage hay Web User Control khác bằng cách kéo tập tin *.ascx và thả vào phần Design của trang muốn sử dụng nó.



7.3 Quốc tế hóa website

7.3.1 Giới thiệu

Ngày nay việc làm cho một Website trở nên đa ngôn ngữ là vô cùng cần thiết. Vì hầu hết người duyệt website đến từ khắp nơi trên thế giới.

Trong phần này sẽ giúp chúng ta thực hiện một website đa ngôn ngữ.

Tổ chức của một website đa ngôn ngữ trong ASP.NET là việc tổ chức các tài nguyên đa ngôn ngữ và sau đó điều khiển việc lựa chọn ngôn ngữ để tài nguyên tương ứng với ngôn ngữ được lựa chọn sẽ được tải vào để hiển thị trên giao diện.

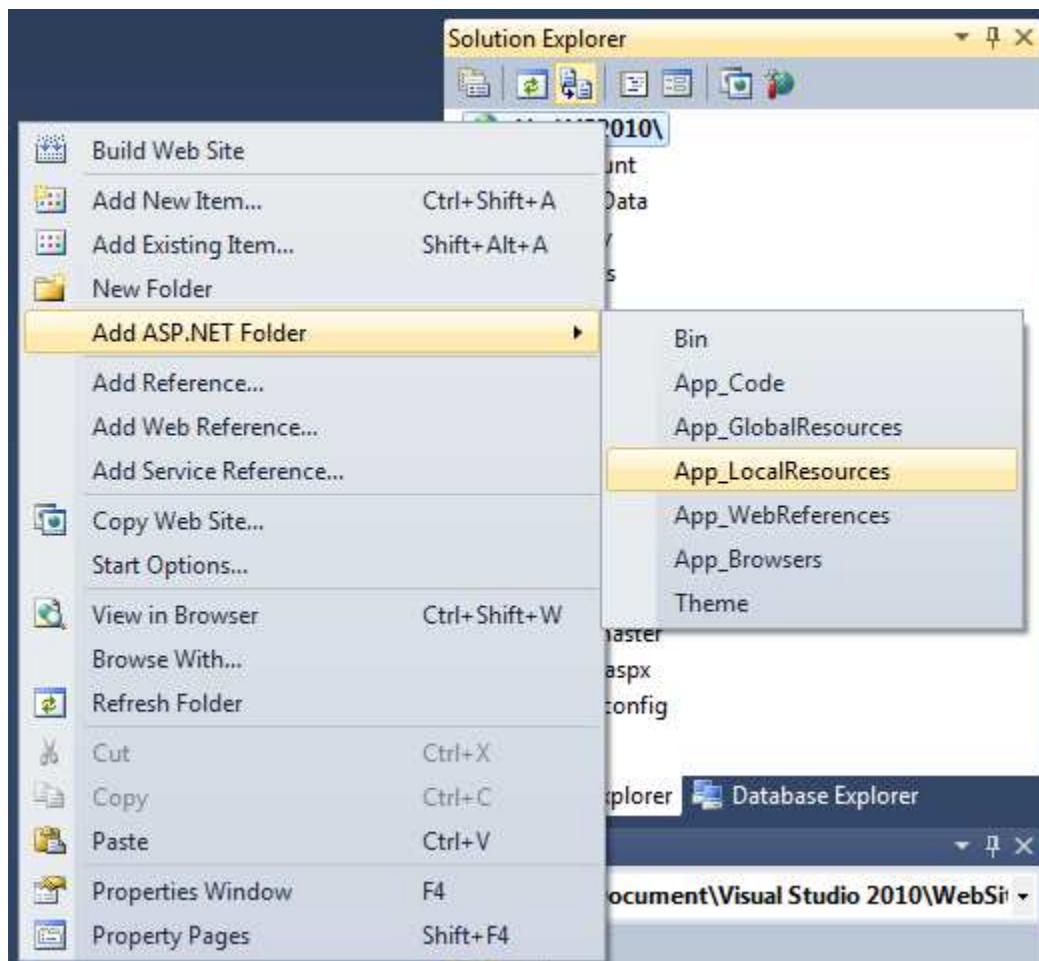
LocalResources là tài nguyên cục bộ của từng tập tin thành phần web (aspx, ascx, master). Mỗi tập tin tài nguyên cục bộ có tên theo cấu trúc sau:

<tên tập tin web>.<mã ngôn ngữ-mã quốc gia>.resx. Ví dụ tập tin tài nguyên tiếng việt của trang abc.aspx là **abc.aspx.vi-VN.resx**, tập tin tài nguyên tiếng việt của control xyz.ascx là **xyz.ascx.vi-VN.resx**.

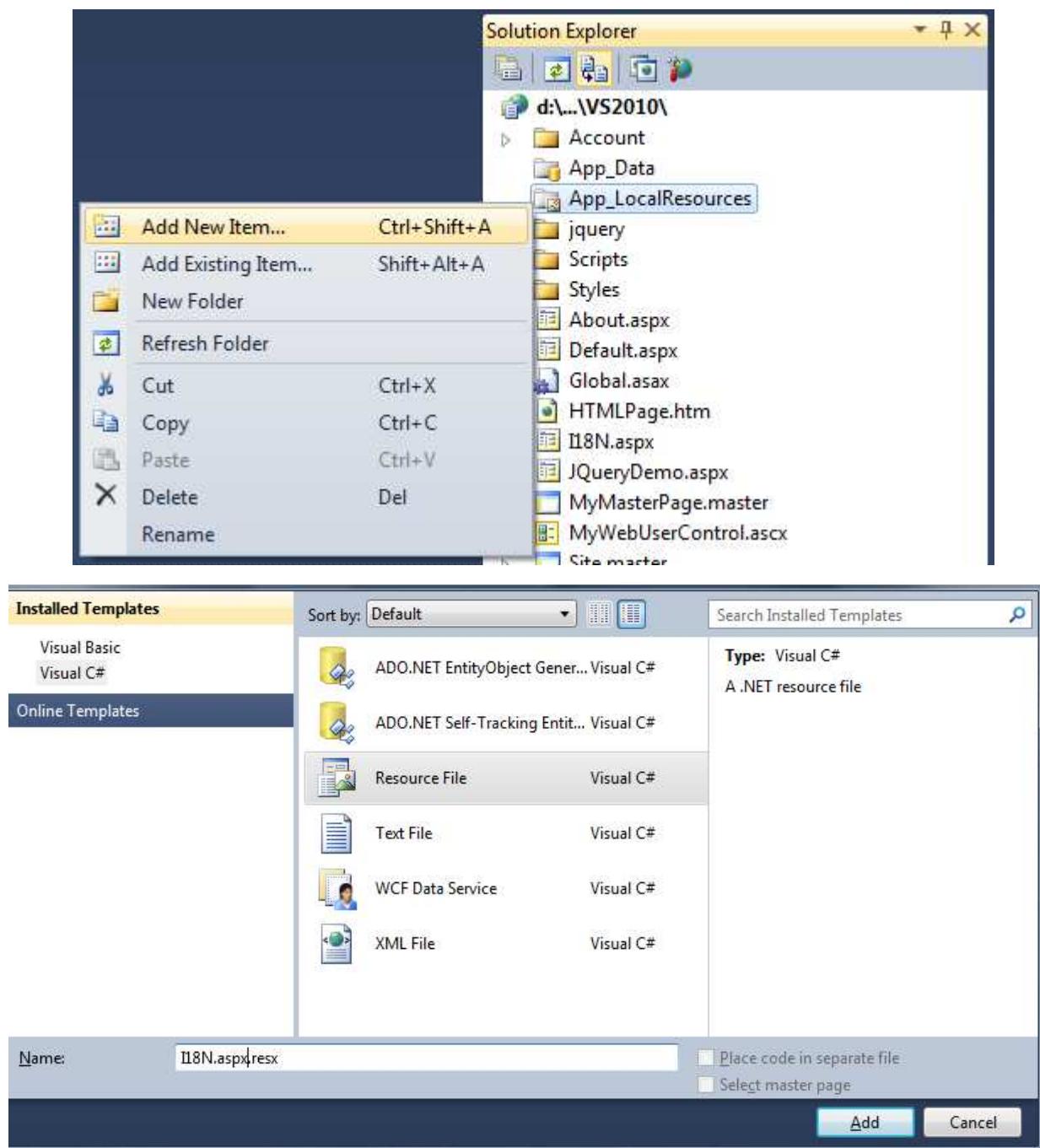
Nội dung của tập tin tài nguyên cục bộ thực chất là danh sách các cặp key=value. Key được sử dụng để truy xuất giá trị hiển thị lên các điều khiển giao diện.

7.3.2 Tạo tài nguyên cục bộ

Các tài nguyên cục bộ được đặt trong thư mục **App_LocalResources** vì vậy trước hết chúng ta cần tạo thư mục này như sau



Sau đó để tạo tập tin tài nguyên cục bộ mặc định cho trang I18N.aspx bằng cách phải chuột lên thư mục App_LocalResources -> Add New Item



Nhập vào tập tin tài nguyên cục bộ mặc định nội dung như sau:

Name	Value	Comment
name	Nhat Nghe Company	
hello	Hello Everybody	

Tạo tập tin tài nguyên cục bộ tiếng Việt bằng cách nhân đôi tập tin I18N.aspx.resx và đổi tên thành I18N.aspx.vi-VN.resx sau đó dịch lại cột Value sang tiếng Việt.

	Name	Value	Comment
*	hello	Chào mọi người	
*	name	Công ty Nhất Nghệ	

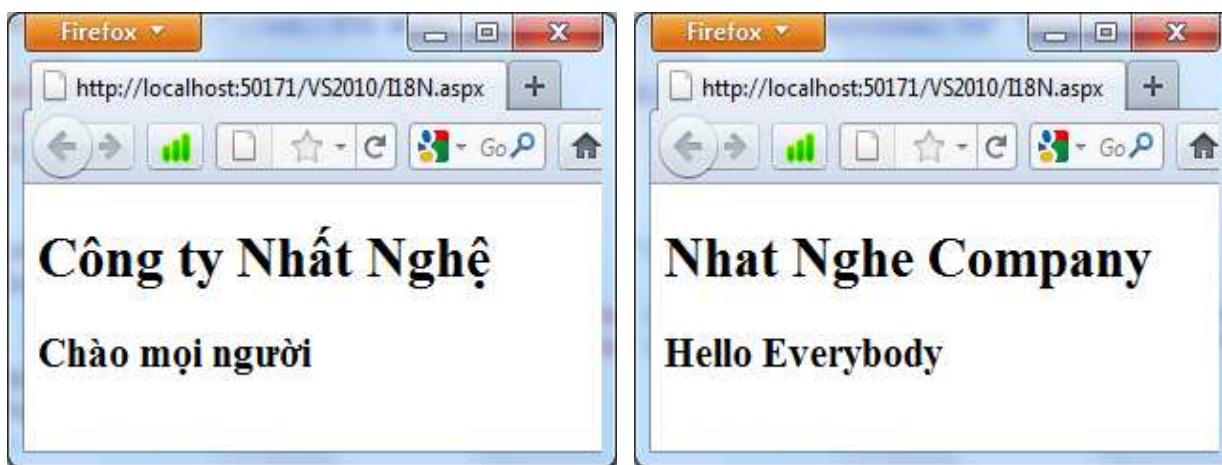
Như vậy cho đến đây, chúng ta đã có 2 tập tin tài nguyên cục bộ tương ứng với 2 ngôn ngữ (mặc định và tiếng Việt).

7.3.3 Quốc tế hóa trang web I18N.aspx

Trang I18N.aspx sẽ hiển thị tiếng Việt

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="I18N.aspx.cs" Inherits="I18N" uiCulture="vi-VN"%>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>
<h1><asp:Label ID="Label1" runat="server" Text="<%$ Resources:name %>"></h1>
<h2><asp:Label ID="Label2" runat="server" Text="<%$ Resources:hello %>"></h2>
</div>
</form>
</body>
</html>
```

Khi bạn chuyển đổi giá trị của thuộc tính uiCulture="en_US" thì trang web sẽ hiển thị tiếng anh.



Tuy nhiên công việc chuyển đổi ngôn ngữ thủ công như trên sẽ rất khó khăn khi áp dụng vào thực tế. Sau đây là cách chuyển đổi ngôn ngữ bằng cách lập trình.



Chọn English

Chọn Tiếng Việt

Mã ASP.NET

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="I18N.aspx.cs" Inherits="I18N"%>

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <a href="?Language=vi-VN">Tiếng Việt</a> | 
    <a href="?Language=en-US">English</a>
    <hr />
    <form id="form1" runat="server">
        <div>
            <h1><asp:Label ID="Label1" runat="server" Text="<%$ Resources:name %>"></h1>
            <h2><asp:Label ID="Label2" runat="server" Text="<%$ Resources:hello %>"></h2>
        </div>
    </form>
</body>
</html>
```

Mã C# code behind:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Globalization;
using System.Threading;
```

```
public partial class I18N : System.Web.UI.Page
{
    /*
     * Viết đè phương thức này để thiết lập ngôn ngữ lựa chọn cho trang web
     */
    protected override void InitializeCulture()
    {
        base.InitializeCulture();

        /*--Lưu ngôn ngữ lựa chọn vào session nếu nhập vào liên kết--*/
        foreach (String key in Request.QueryString.AllKeys)
        {
```

```

if (key.Contains("Language"))
{
    Session["Language"] = Request.QueryString["Language"];
}
}

/*--Lưu ngôn ngữ mặc định vào session--*/
if (Session["Language"] == null)
{
    Session["Language"] = "en-US";
}

/*--Thiết lập ngôn ngữ cho trang web--*/
CultureInfo Culture = new CultureInfo(Session["Language"] as String);
Thread.CurrentCulture = Culture;
Thread.CurrentThread.CurrentCulture = Culture;
}
}

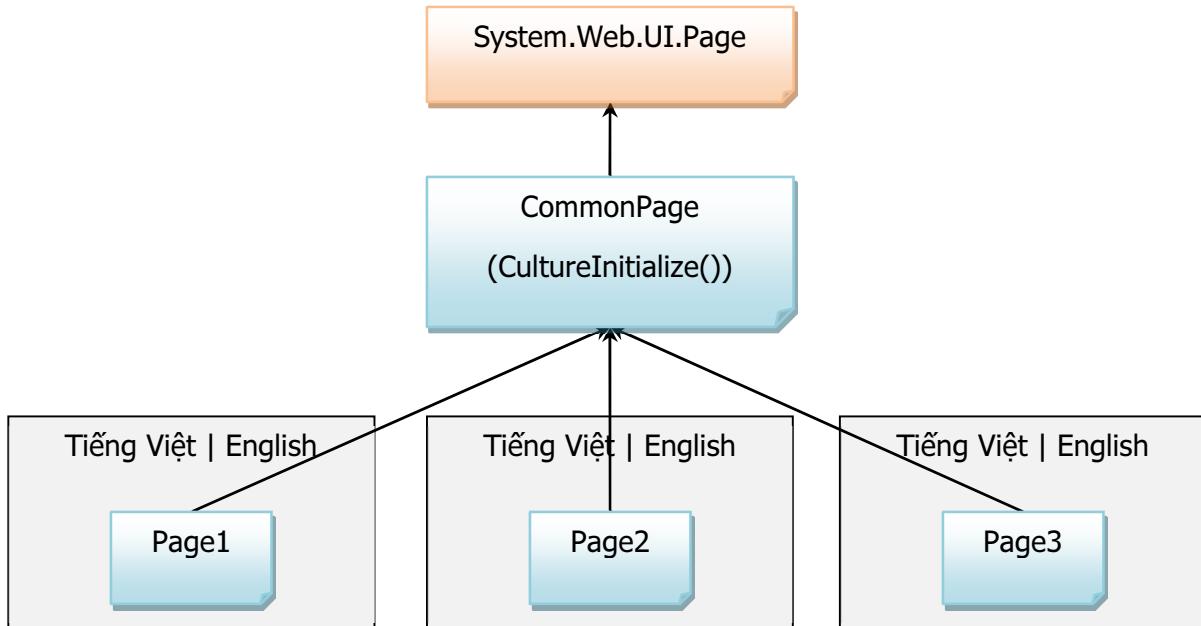
```

Với cách tổ chức trang web như trên, chúng ta đã có thể thiết lập ngôn ngữ bằng cách lập trình. Dù sao, chúng ta vẫn còn một khó khăn lớn khi tổ chức đa ngôn ngữ cho nhiều trang. Khi đó chúng ta phải viết đè phương thức InitializeCulture() cho từng trang riêng biệt. Và chép 2 liên kết chọn ngôn ngữ Tiếng Việt | English vào tất cả các trang ASP.NET.

Để khắc phục trường hợp này, chúng ta tạo ra:

- ✓ Lớp CommonPage (kế thừa từ Page) và viết đè phương thức InitializeCulture(). Sau đó các trang đa ngôn ngữ chỉ việc kế thừa từ CommonPage.
- ✓ MasterPage: chứa 2 liên kết lựa chọn ngôn ngữ.

Mô hình tổ chức như sau:



7.4 Theme & Skin

7.4.1 Giới thiệu

Khi thiết kế trang web, chúng ta thường thiết lập các thông số kiểu dáng phù hợp cho các điều khiển giao diện - công việc này không thể thiếu và xảy ra thường xuyên.

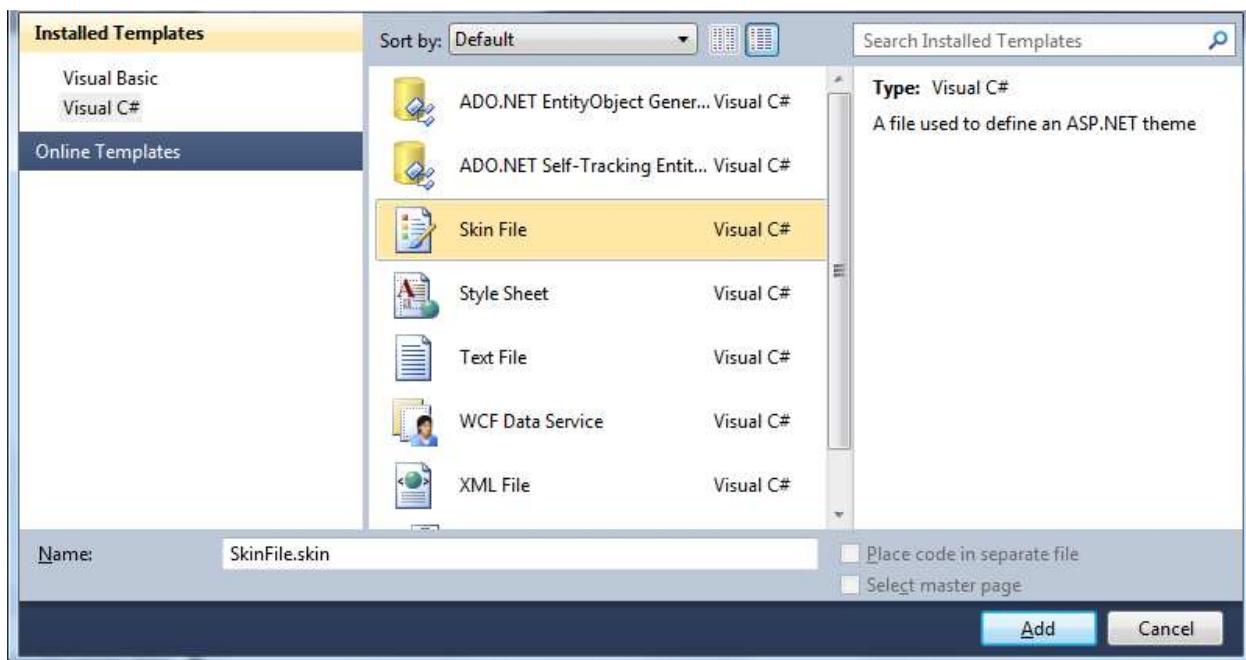
Gần như tất cả các website cần một số kiểu dáng chuẩn cho tất cả các điều khiển giao diện cùng loại trên cùng một trang và thậm chí cả giữa các trang khác nhau. Vấn đề đặt ra là “làm thế nào để cất giữ các kiểu dạng của một số điều khiển để áp dụng cho các điều khiển cùng loại sau này” – Theme và Skin sẽ giúp bạn giải quyết vấn đề trên.

Thực chất Skin được xem là CSS server vì kết quả kết xuất kiểu dáng Skin sẽ sinh mã CSS cho các điều khiển.

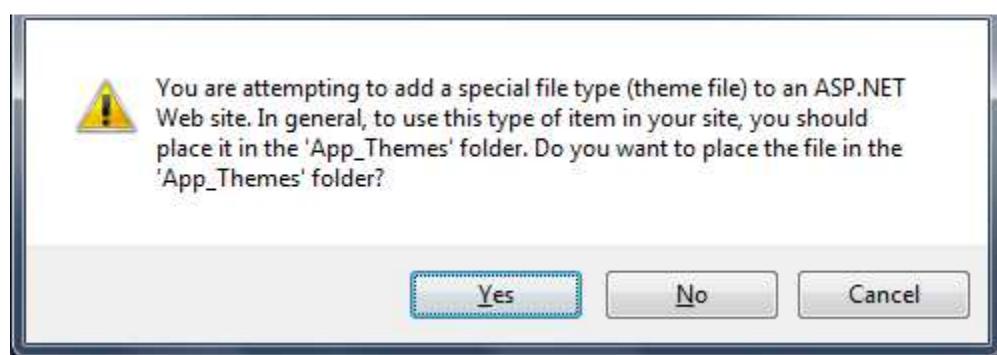
7.4.2 Tạo tập tin Skin

Tập tin Skin là tập tin chứa kiểu dáng của các điều khiển để áp dụng cho các điều khiển trên các trang trong website.

Theo hướng dẫn sau đây để tạo tập tin Skin



Chọn Add



Chọn Yes. Bạn sẽ thấy một cấu trúc thư mục được sinh ra như sau:



Tập tin Skin được tạo ra có sẵn hướng dẫn trên đó như sau

<%--

Default skin template. The following skins are provided as examples only.

1. Named control skin. The SkinId should be uniquely defined because duplicate SkinId's per control type are not allowed in the same theme.

```
<asp:GridView runat="server" SkinId="gridviewSkin" BackColor="White" >
<AlternatingRowStyle BackColor="Blue" />
</asp:GridView>
```

2. Default skin. The SkinId is not defined. Only one default control skin per control type is allowed in the same theme.

```
<asp:Image runat="server" ImageUrl("~/images/image1.jpg" />
--%>
```

Thiết kế kiểu dáng cho các điều khiển dùng chung trên một trang web nào đó sau đó chép mã ASP.NET của điều khiển thiết kế được vào tập tin Skin để lưu giữ và áp dụng cho các điều khiển khác sau này. Lưu ý loại bỏ thuộc tính ID của các điều khiển khỏi tập tin Skin.

Ví dụ sau đây là mã của App_Themes/SkinFile/SkinFile.skin chứa kiểu dáng cho 2 TextBox (một dùng chung-không thuộc tính SkinID và một dùng riêng-có thuộc tính SkinID) và một Calendar.

```
<!--TextBox dùng chung-->
<asp:TextBox runat="server" BackColor="Yellow" BorderColor="Red"
BorderStyle="Dotted" BorderWidth="1px" Height="22px" Width="200px"></asp:TextBox>

<!--Calendar dùng chung-->
<asp:Calendar runat="server" BackColor="White"
BorderColor="#999999" CellPadding="4" DayNameFormat="Shortest"
Font-Names="Verdana" Font-Size="8pt" ForeColor="#0000CC" Height="180px"
Width="200px">
<DayHeaderStyle BackColor="#CCCCCC" Font-Bold="True" Font-Size="7pt" />
<NextPrevStyle VerticalAlign="Bottom" />
<OtherMonthDayStyle ForeColor="#808080" />
<SelectedDayStyle BackColor="#666666" Font-Bold="True" ForeColor="White" />
<SelectorStyle BackColor="#CCCCCC" />
<TitleStyle BackColor="#999999" BorderColor="Black" Font-Bold="True" />
<TodayDayStyle BackColor="#CCCCCC" ForeColor="Black" />
<WeekendDayStyle BackColor="#FFFFCC" />
</asp:Calendar>

<!--TextBox dùng riêng (có SkinID)-->
<asp:TextBox SkinID="NhatNghe" runat="server" BackColor="#66FFFF" BorderColor="Blue"
BorderStyle="Double" BorderWidth="1px" Height="25px" Width="200px"></asp:TextBox>
```

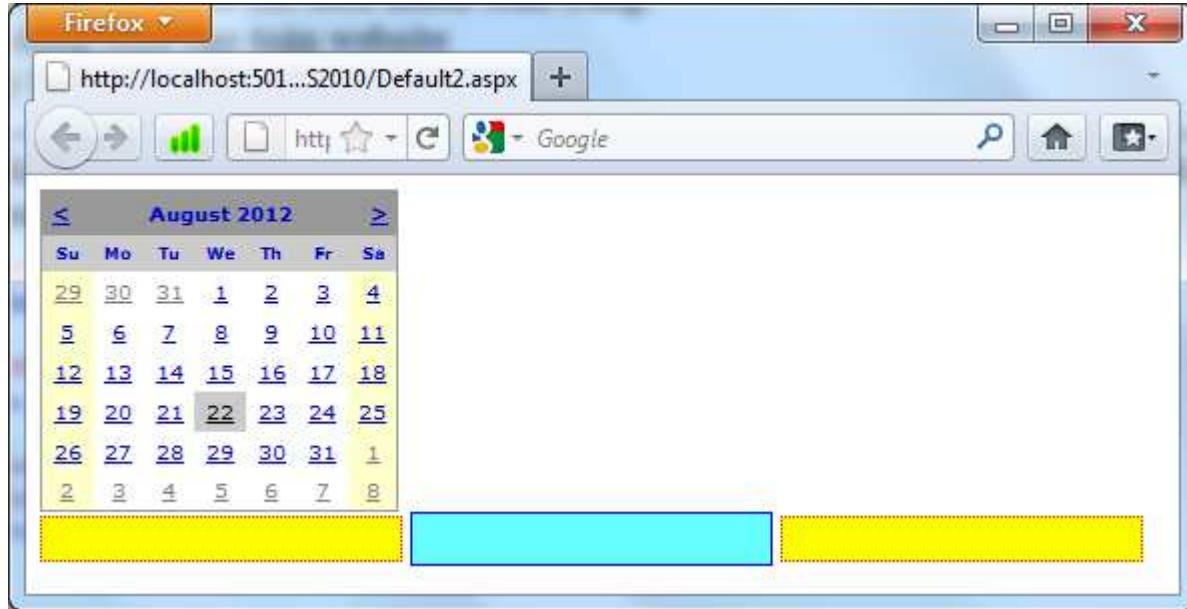
7.4.3 Áp dụng Skin cho các điều khiển

Áp dụng Skin cho các điều khiển bằng cách chỉ rõ Skin ở thuộc tính **StyleSheetTheme="SkinFile"** của chỉ thị @Page. Trang web sau đây áp dụng các kiểu dáng của TextBox và Calendar đã được thiết kế trong SkinFile cho các điều khiển tương ứng của trang.

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

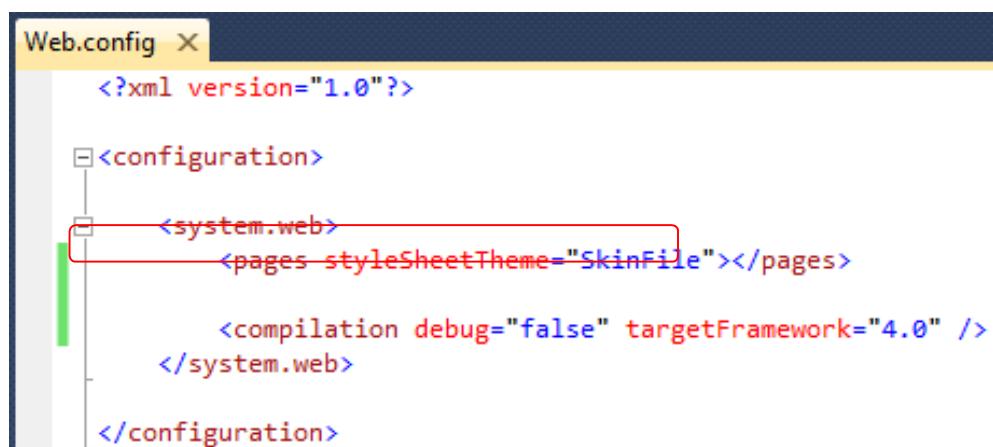
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>
<asp:Calendar ID="Calendar1" runat="server"></asp:Calendar>
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
```

```
<asp:TextBox ID="TextBox3" runat="server" SkinID="NhatNghe"></asp:TextBox>
<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
</div>
</form>
</body>
</html>
```



Ta thấy rõ TextBox3 được áp dụng TextBox đã định nghĩa riêng với thuộc tính SkinID="NhatNghe".

Muốn áp dụng Skin cho tất cả các trang trong website bạn chỉ việc thêm thẻ **<pages styleSheetTheme="SkinFile"></pages>** vào thẻ **<system.web>** của tập tin cấu hình web.config mà không cần khai báo **StyleSheetTheme="SkinFile"** của chỉ thị **@Page**:

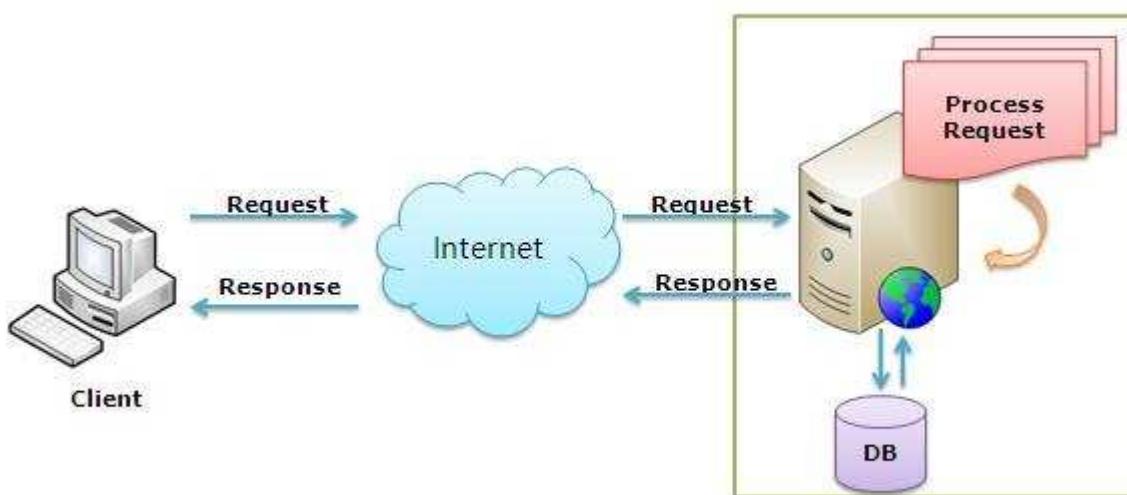


Chương 8: CHIA SẼ DỮ LIỆU

Sau khi học xong bài này, học viên có khả năng :

- Trình bày và sử dụng được các đối tượng Server, Session và Application.
- Trình bày và sử dụng được các phương pháp truyền tham số trong ASP.NET : GET, POST và Cross-Page.
- Mô tả được cách sử dụng Cookie/Session để lưu thông tin.

8.1 Đối tượng Request, Response, Server



HÌNH 1: QUÁ TRÌNH REQUEST - RESPONSE

8.1.1 Đối tượng Request

- Đối tượng Request dùng để đọc các thông tin từ trình duyệt người dùng (browser) gửi đến Web Server. Thông tin này bao gồm các thông số của Form được gửi lên thông qua phương thức POST (mặc định)/GET hay các thông số kèm trang ASP.NET được gọi.
- Đối tượng Request còn dùng để trao đổi các thông tin giữa các trang ASP.NET trong cùng website thông qua các giá trị cookies (*sẽ được giới thiệu ở phần sau*) trên máy client.
- Danh sách các thuộc tính/phương thức của đối tượng Request:

Thuộc tính	Điển giải
AcceptTypes	Trả về chuỗi MIME được hỗ trợ bằng trình khách
ApplicationPath	Trả về đường dẫn ảo của ứng dụng
ClientCertificate	Trả về đối tượng HttpClientCertificate
ContentEncoding	Trả về Tập kí tự của thực thể Body
ContentLength	Trả về chiều dài tính bằng bytes của yêu cầu
ContentType	Trả về loại MIME của yêu cầu
Cookies	Trả về đối tượng HttpCookies
FilePath	Trả về đường dẫn ảo của yêu cầu
Files	Trả về HttpFileCollection của tập nhiều tập tin được tải lên Server
Form	Trả về tập dữ liệu của nội dung gửi từ Form

Header	Trả về tập dữ liệu của nội dung từ HTTP Header
HttpMethod	Trả về phương thức HTTP sử dụng cho yêu cầu
IsLocal	True - Returns the value indicating whether the request is from local computer
IsSecureConnection	False - Returns the value indicating whether the connection is secure (using HTTPS)
Params	Lấy giá trị của đối tượng gửi trong Form (POST/GET) hay từ QueryString
Path	Trả về đường dẫn ảo của yêu cầu. VD: /myfunda/test.aspx
PhysicalApplicationPath	Trả về đường dẫn vật lý của yêu cầu. VD: C:\Web\MyFunda.Web\
PhysicalPath	Trả về đường dẫn vật lý của yêu cầu. VD: C:\Web\MyFunda.Web\test.aspx
QueryString	Trả về tập hợp dữ liệu nội dung từ querystring. VD: q=search&re=afdsf
RawUrl	Trả về URL của yêu cầu. VD: /myfunda/test.aspx?q=search&re=afdsf
RequestType	Phương thức HTTP sử dụng cho Request
Url	Cho biết URL chi tiết. VD: http://localhost/myfunda/test.aspx?q=search&re=afdsf
UserHostAddress	Trả về địa chỉ IP của client. VD: 60.243.141.170
UserHostName	Trả về tên DNS của client.

Phương thức	Diễn giải
BinaryRead	Trả về mảng Byte chứa đựng thông tin nhị phân gửi đến Server
MapPath	Chuyển đổi đường dẫn ảo thành đường dẫn vật lý
SaveAs	Lưu yêu cầu HTTP vào đĩa

- Lấy dữ liệu từ người dùng gửi lên:

- *Request.QueryString["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["Tên_Phần_tử_cần_đọc"]*: Để đọc giá trị của một phần tử được gửi theo phương thức Post (Method = "Post").

8.1.2 Đối tượng Response

- Đối tượng Response được sử dụng để giao tiếp với client, nó quản lý và điều phối thông tin từ Web Server đến các trình duyệt của người dùng.
- Danh sách các thuộc tính và phương thức:

Thuộc tính	Diễn giải
BufferOutput	Có sử dụng hay không bộ nhớ đệm cho kết xuất dữ liệu
Cache	Trả về đối tượng HttpCachePolicy chứa đựng thông tin về quy định Cache của phύc đáp hiện hành
CacheControl	Mặc dù còn hỗ trợ nhưng phương thức này còn đối nghịch trong phương thức của HttpCachePolicy
ContentEncoding	Tập nhận dạng kết xuất, là một trong các giá trị như UnicodeEncoding, UTF7Encoding, UTF8Encoding
ContentType	Loại MIME trả về cho client

Cookies	Trả về một tập của đối tượng HttpCookies
Expires	Mặc dù còn hỗ trợ nhưng phương thức này còn đối nghịch trong phương thức của HttpCachePolicy
ExpiresAbsolute	Mặc dù còn hỗ trợ nhưng phương thức này còn đối nghịch trong phương thức của HttpCachePolicy
Filter	Đối tượng Stream dùng làm bộ lọc dữ liệu kết xuất
Output	Trả về đối tượng TextWriter
OutputStream	Đối tượng Stream dùng để trình bày hàng dữ liệu của body
Status	Gán trạng thái HTTP trả về cho trình khách
StatusCode	Trạng thái HTTP Response
StatusDescription	Gán diễn giải trạng thái HTTP và trả về cho trình khách, thuộc tính này được ưu tiên hơn thuộc tính Status

Phương thức	Điều giải
Clear	Xóa vùng tạm
ClearContent	Xóa nội dung từ Buffer Stream
ClearHeaders	Xóa header từ Buffer Stream
Close	Đóng kết nối với Client
End	Kết thúc tiến trình xử lý trên Server và đẩy dữ liệu tới Client
Flush	Đưa dữ liệu còn trong bộ đệm phía server về cho client
Redirect	Chuyển hướng đến địa chỉ file trong cùng ứng dụng hay URL khác trong lúc thi hành
Write	Ghi thông tin từ các kiểu dữ liệu như Char, Object, String, Array ra trang web
WriteFile	Ghi một luồng dữ liệu ra tập tin chỉ định

Trong đó, hai thương thức chính thường dùng: **Write** và **Redirect**. Ví dụ:

- Chuyển tới trang Chuong08.aspx trong cùng web site:

```
Response.Redirect("Chuong08.aspx");
```

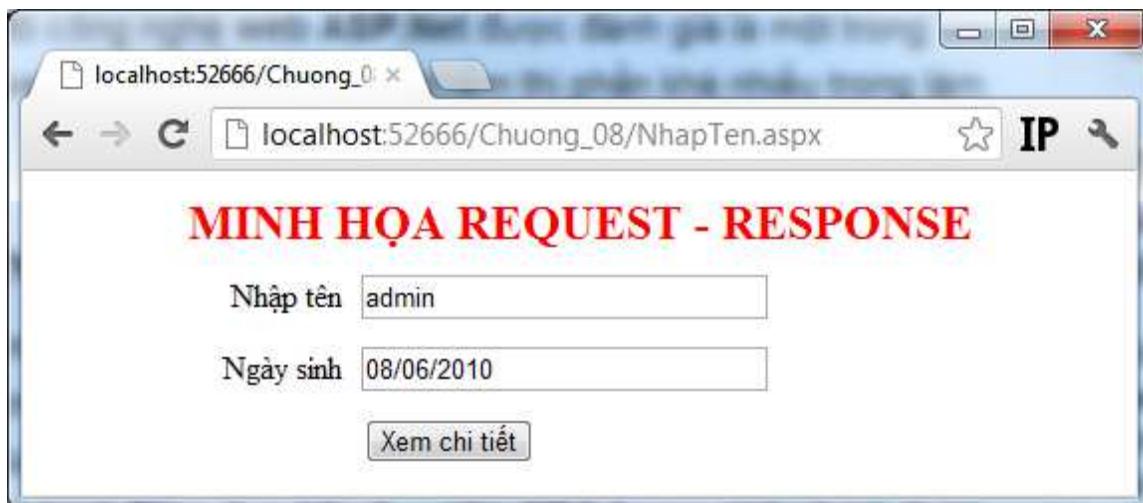
- Chuyển tới trang www.hienlth.info:

```
Response.Redirect("http://www.hienlth.info");
```

8.1.3 Ví dụ minh họa Request, Response

Bước 1: Tạo 2 trang ASP.NET gồm trang NhapTen.aspx và trang XemChiTiet.aspx. Bảng mô tả các thuộc tính của các controls trang NhapTen.aspx.

Control	Tên thuộc tính	Giá trị thuộc tính
Label	Text	Nhập tên
Label	Text	Ngày sinh
TextBox	ID	txtTen
TextBox	ID	txtNgaySinh
Button	Text	Xem chi tiết
	ID	btnXemChiTiet



HÌNH 2: MINH HỌA TRANG NHAPTen.ASPX



HÌNH 3: KẾT QUẢ TRẢ VỀ

Bước 2: Viết lệnh xử lý cho các trang như sau:

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

    protected void btnXemChiTiet_Click(object sender, EventArgs e)
    {
        //Lấy các giá trị đã nhập vào các TextBox
        string strHoTen = txtTen.Text;
        string strNgaySinh = txtNgaySinh.Text;
        //Chuyển quan trang XemChiTiet.aspx
        Response.Redirect("XemChiTiet.aspx?Ten=" + strHoTen + "&NgaySinh=" +
strNgaySinh);
    }
}
```

Phần code behide trang NhapTen.aspx

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

```
//Lấy các giá trị từ trang NhapTen.aspx
string strHoTen = Request.QueryString["Ten"];
string strNgaySinh = Request.QueryString["NgaySinh"];
string s = "Xin chào bạn : <b>" + strHoTen + "</b><br>" +
    "Ngày sinh của bạn là : <b>" + strNgaySinh + "</b>";
//In xuống trình duyệt
Response.Write(s);
}
}
```

Phần code behide trang XemChiTiet.aspx

8.1.4 Đối tượng Server

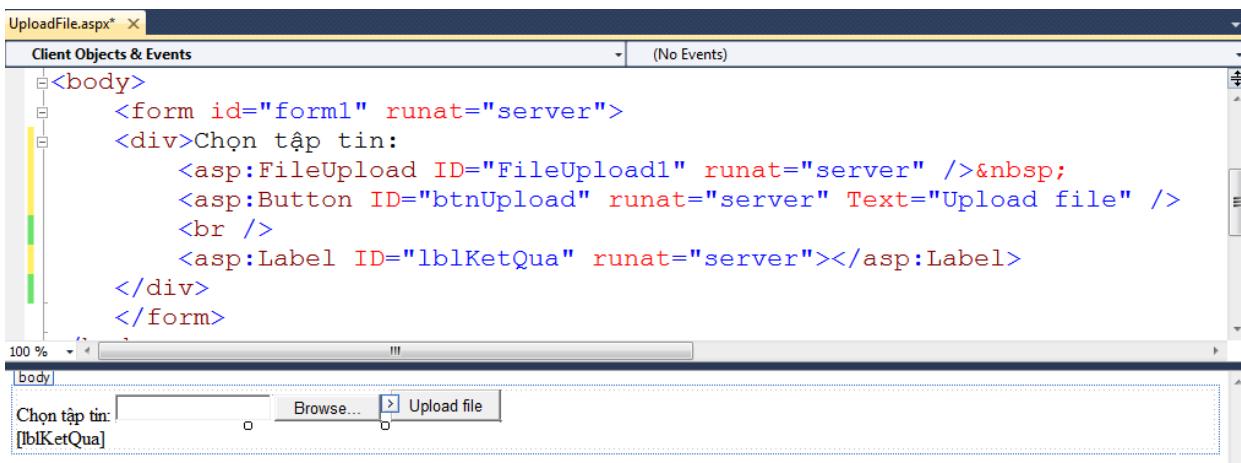
- Đối tượng Server được sử dụng để cung cấp thông tin của Server cho ứng dụng.
- Danh sách Thuộc tính/Phương thức:

Tên thuộc tính	Diễn giải
MachineName	Lấy tên của Web Server
ScriptTimeout	Thiết lập thời gian xử lý tối đa 1 file

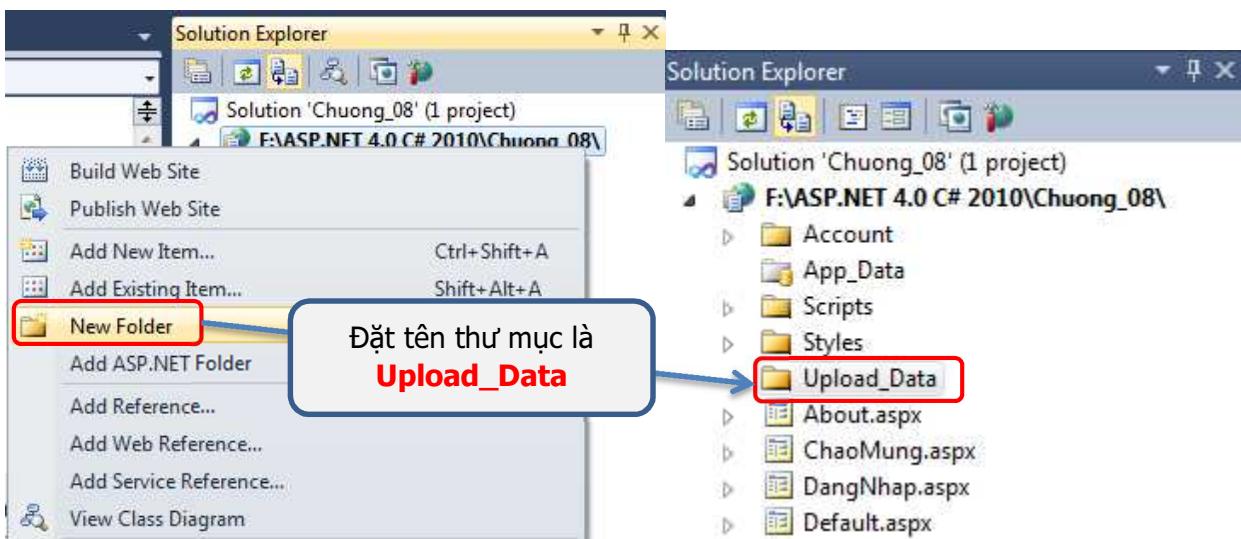
Phương thức	Diễn giải
Excute	Thực thi một trang ASP.NET khác
HtmlDecode	Giải mã chuỗi thành các thẻ HTML tương ứng (ngược với HtmlEncode)
HtmlEncode	Mã hóa HTML thành chuỗi
MapPath	Lấy đường dẫn vật lý hoặc đường dẫn ảo đến một thư mục trên Server
Transfer	Ngừng thi hành trang hiện hành, gửi yêu cầu mới đến trang được gọi thực hiện
UrlDecode	Ngược lại của UrlEncode, giải mã chuỗi mã hóa URL
UrlEncode	Mã hóa URL thành chuỗi (khoảng trắng thành dấu "+", ký tự không thuộc chữ và số sẽ chuyển thành số hexadecimal).

- Phân biệt Response.Redirect với Server.Transfer:
 - Response.Redirect:** Chuyển đến 1 trang mới giống như gõ địa chỉ trang đó trên trình duyệt và request đến server.
 - Server.Transfer:** Cũng chuyển đến trang mới nhưng ở phía Server, làm giảm request đến server, giữ nguyên URL và có thể chuyển cả *các query string và biến trên Form* đến địa chỉ mới, chỉ thực hiện giữa các trang trên cùng 1 host.
- Ví dụ đối tượng Server: Minh họa Upload file lên server.

Bước 1: Thiết kế trang UploadFile.aspx có giao diện như sau:



Dữ liệu sau khi upload thành công sẽ lưu trữ trong thư mục **Upload Data** như cấu trúc file project:



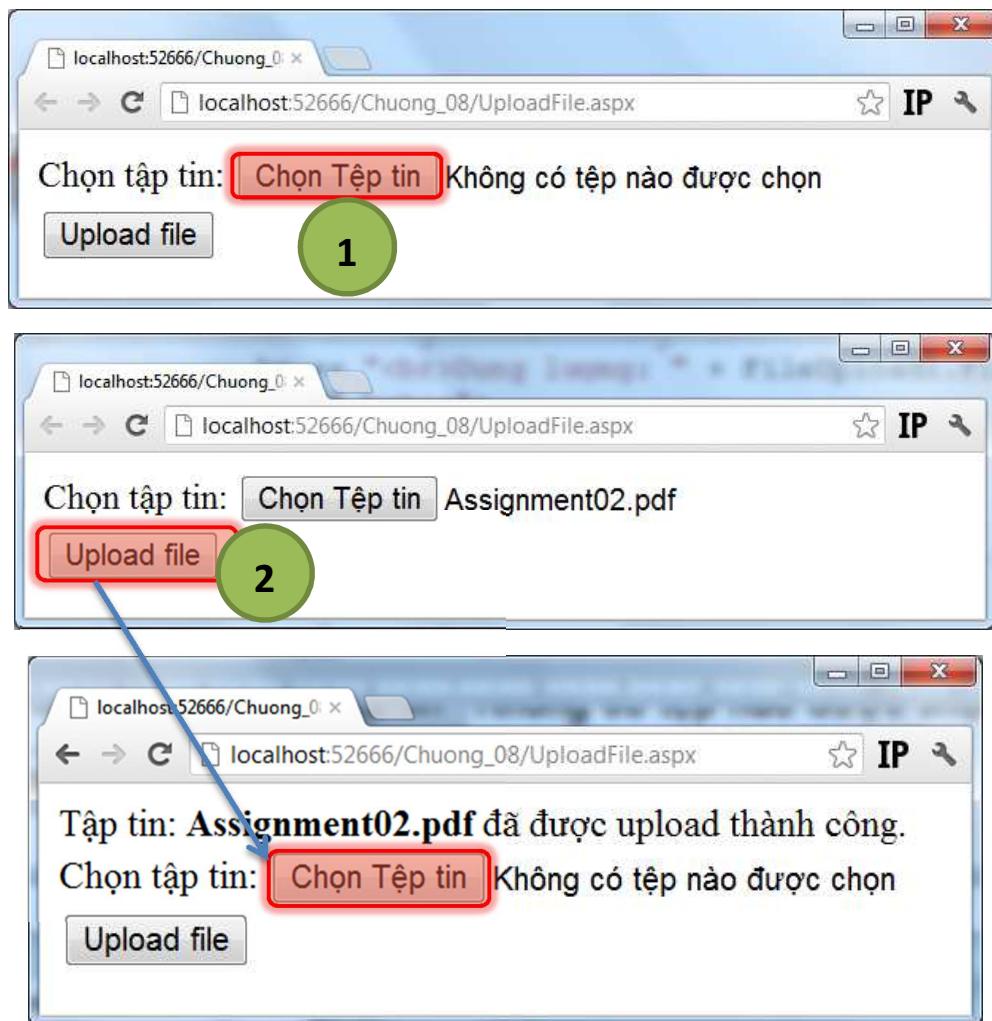
Bước 2: Code behind xử lý cho nút **Upload file**.

```
public partial class UploadFile : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void btnUpload_Click(object sender, EventArgs e)
    {
        //Kiểm tra có tập tin nào được chọn hay không?
        if (FileUpload1.HasFile)
        {
            //Thiết lập đường dẫn cho tập tin
            string filename = "~/Upload_Data/" + FileUpload1.FileName;
            //Lấy đường dẫn vật lý cho tập tin trên ứng dụng
            string s = Server.MapPath(filename);
            //Lưu tập tin về thư mục Upload_Data của ứng dụng
            FileUpload1.SaveAs(s);
            Response.Write("Tập tin: <b>" + FileUpload1.FileName +
                "</b> đã được upload thành công.");
        }
    }
}
```

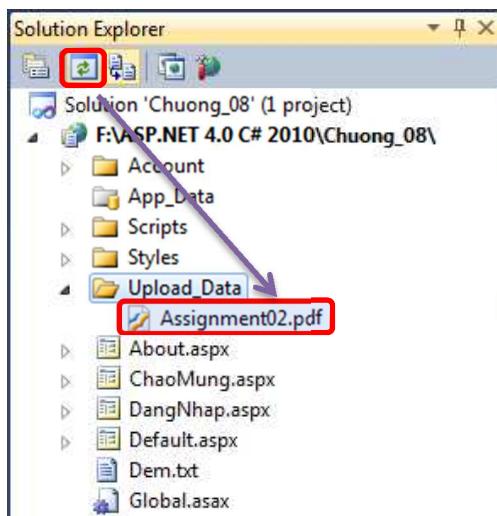
}

Phần code behide trang UploadFile.aspx

Bước 3: Chạy và kiểm tra kết quả.



Để kiểm tra file đã upload lên chưa, click biểu tượng **Refresh** thấy kết quả như hình dưới.



8.2 Truyền tham số giữa các trang ASP.NET

8.2.1 Truyền dữ liệu theo phương thức POST

- Đây là phương pháp mặc định khi thiết kế Form, dữ liệu truyền đi không hiển thị trên thanh địa chỉ (URL) của trình duyệt.
- Lấy dữ liệu từ người dùng gửi đi: `Request.Form["ten_phan_tu"]`
- Ví dụ minh họa:



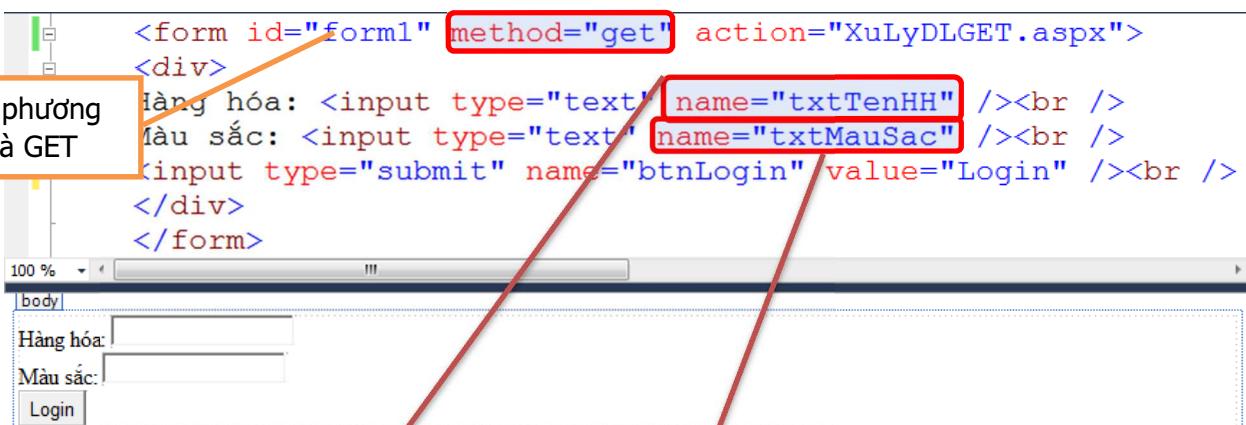
HÌNH 4: THIẾT KẾ TRANG TRUYENDLPOST.ASPX

```
public partial class TruyenDuLieuPOST : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        string s = "Tên hàng hóa: " + Request.Form["txtTenHangHoa"].ToString();
        s += "<br>Màu sắc: " + Request.Form["txtMauSac"].ToString();
        Response.Write(s);
    }
}
```

8.2.2 Truyền dữ liệu theo phương thức GET

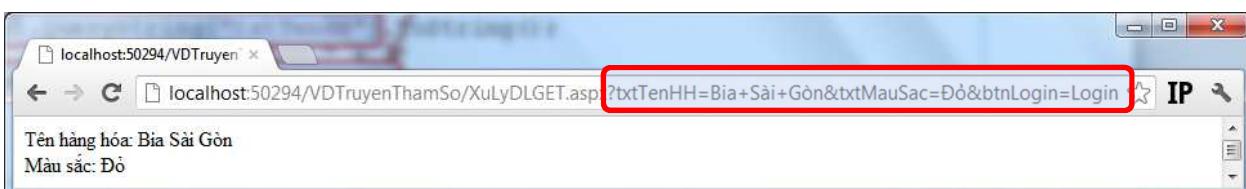
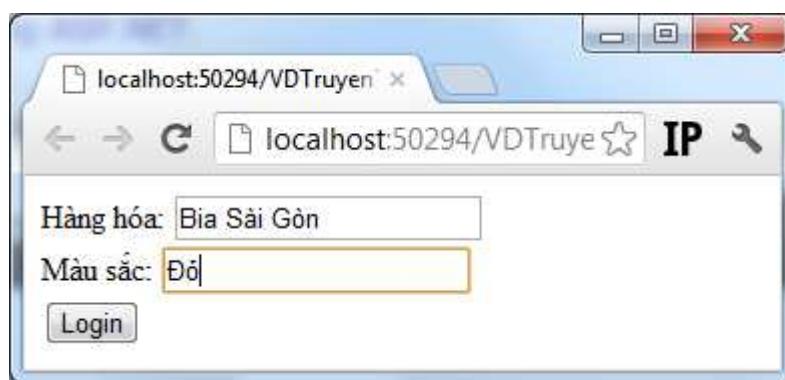
- Truyền dữ liệu theo phương thức GET: Dữ liệu truyền đi hiển thị trên thanh địa chỉ (URL) của trình duyệt. Có 2 cách dùng Form hoặc truyền trực tiếp trên thanh URL(QueryString).
- Lấy dữ liệu từ người dùng gửi đi: `Request.QueryString["ten_phan_tu"]`
- Ví dụ dùng Form để truyền tham số:

Chỉ định rõ phương thức gửi là GET



HÌNH 5: MÀN HÌNH TRANG TRUYENDLGET.ASPX

```
public partial class XuLyDLGET : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        string s = "Tên hàng hóa: " +
Request.QueryString["txtTenHH"].ToString();
        s += "<br>Màu sắc: " +
Request.QueryString["txtMauSac"].ToString();
        Response.Write(s);
    }
}
```



Lưu ý: Với các truyền này, thiết kế dùng các đối tượng HTML control bình thường, không có thuộc tính runat = "server".

- Ví dụ truyền trực tiếp trên URL:

Giả sử có liên kết đến trang ReadNews.aspx như sau:

```
<a href="ReadNews.aspx?typeNews=edu&NewsID=98708">
    Tin Giáo dục
</a>
```

Khi click vào liên kết:



Tại ReadNews.aspx muốn lấy giá `typeNews` and `NewsID`, sử dụng cú pháp:

```
Var1=Request.QueryString["typeNews"]; //Var1=edu
Var2=Request.QueryString["NewsID"]; //Var2=98708
```

8.2.3 Truyền dữ liệu sử dụng Cross-Page

Cross-page postback là dạng postback gửi tới trang khác. Các button control như `Button`, `ImageButton` và `LinkButton` đều có thuộc tính `PostBackUrl`. Để sử dụng Cross-page postback:

- Thiết lập thuộc tính `PostBackUrl` của button control cho web form cần chuyển đến.
- Khi người dùng click vào button, Page sẽ được gửi tới URL mới với tất cả các giá trị của các control trong trang hiện thời.

Trang cross page có thể truy cập trang trước thông qua thuộc tính **PreviousPage**.

```
protected void Page_Load(object sender, EventArgs e)
{
    if (PreviousPage != null)
    {
        lblInfo.Text = "You came from " + PreviousPage.Header.Title;
    }
}
```

Để truy cập chi tiết hơn, các giá trị của control trên form, thì cần phải gán tham chiếu `PreviousPage` cho lớp page tương ứng. Ta có thể thêm thuộc tính vào trang nguồn và đọc chúng ở trang đích.

```
if (PreviousPage != null) {
    SourcePage prevPage = PreviousPage as SourcePage;
    if (prevPage != null)
    {
        lblInfo.Text = "Welcome" + prevPage.NameEntry;
    }
}
```

Tên lớp của trang đích **Thuộc tính của lớp SourcePage**

Ví dụ minh họa:

Tạo 2 trang:

- SendInfo.aspx:** chứa form nhập thông tin User

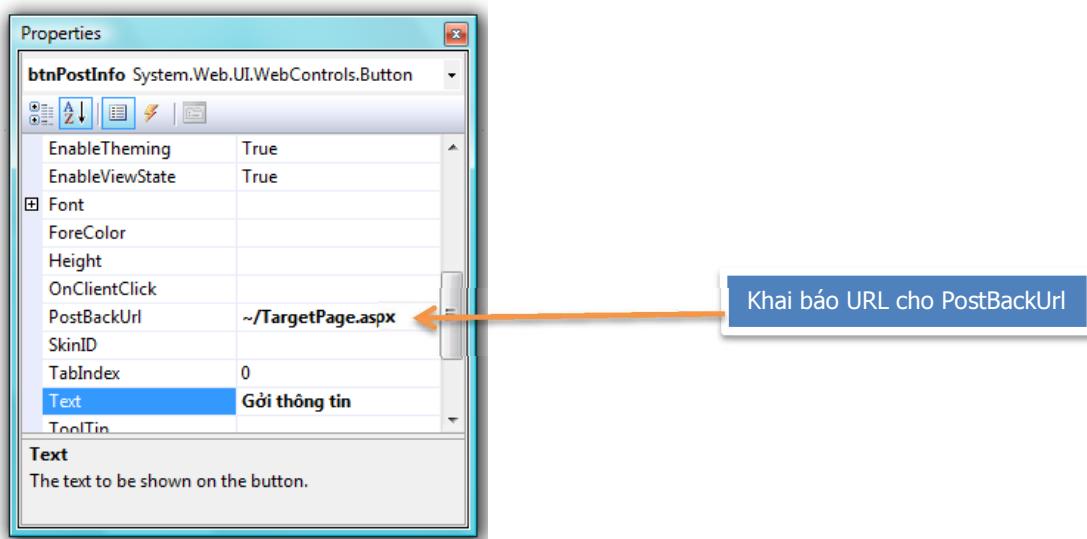
- **TargetPage.aspx:** lấy thông tin người dùng nhập từ trang SendInfo.aspx và hiển thị lên trình duyệt.

Bước 1: Tạo trang SendInfo.aspx có giao diện như sau:

Bước 2: Thiết lập một số thuộc tính các control của trang SendInfo.aspx:

Id control	Kiểu control	Thuộc tính	Diễn giải
txtUsername	TextBox		Nhập tên user
txtEmail	TextBox		Nhập địa chỉ email
txtCompany	TextBox		Nhập tên công ty
btnPostInfo	Button	PostBackUrl = ~/TargetPage.aspx	Button thực thi thao tác Cross Page Postback

Bước 3: Hướng dẫn khai báo thuộc tính PostBackUrl cho button btnPostInfo trả về trang TargetPage.aspx:



Bước 4: Thiết kế giao diện trang TargetPage.aspx.

Trang này lấy thông tin từ SendInfo.aspx và hiển thị ra trình duyệt.

TargetPage.aspx

Thông tin user đăng ký

Tên người dùng	Label
Địa chỉ email	Label
Tên công ty	Label

Id control	Kiểu control	Thuộc tính	Diễn giải
lblUsername	Label	-	Hiển thị tên user
lblEmail	Label	-	Hiển thị email
lblCompany	Label	-	Hiển thị công ty

Bước 5: Viết code xử lý cho trang TargetPage.aspx lấy thông tin từ SourcePage.aspx

```

protected void Page_Load(object sender, EventArgs e) {
    // lấy thông tin từ previous page
    if (PreviousPage != null) {
        // biến đổi tương source tham chiếu đến previous page
        SendInfo source = PreviousPage as SendInfo;
        // lấy các control của previous page (SourcePage.aspx)
        TextBox username =
            (TextBox)source.FindControl("txtUsername");
        TextBox email =
            (TextBox)source.FindControl("txtEmail");
        TextBox company =
            (TextBox)source.FindControl("txtCompany");
        // gán thông tin cho các label
        lblUsername.Text = username.Text;
        lblEmail.Text = email.Text;
        lblCompany.Text = company.Text;
    } // end if
} // end method Page_Load

```

SendInfo là class page
của SendInfo.aspx

ID của control trong SendInfo.aspx

Bước 6: Chạy và demo kết quả.

Chạy trang SendInfo.aspx rồi điền các thông tin theo yêu cầu, sau đó bấm nút “Gửi thông tin” để chuyển tới trang TargetPage.aspx.



8.3 Session và Cookies

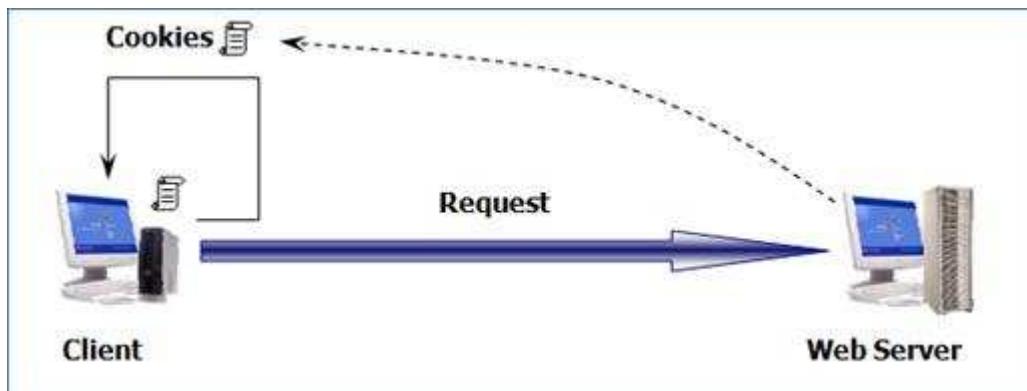
8.3.1 Đối tượng Cookies

Cookie là 1 đoạn dữ liệu được truyền đến browser từ server, đoạn dữ liệu này sẽ được browser lưu trữ trên máy người dùng (đĩa cứng hay bộ nhớ của máy người dùng) và sẽ gửi ngược lên lại server mỗi khi browser tải 1 trang web từ server.

Những thông tin được lưu trữ trong cookie hoàn toàn phụ thuộc vào website trên server.

Mỗi website có thể lưu trữ những thông tin khác nhau trong cookie, ví dụ thời điểm lần cuối bạn ghé thăm website, đánh dấu bạn đã login hay chưa,...

Ngoài ra, mỗi browser quản lý và lưu trữ cookie theo cách riêng của mình, cho nên 2 browser cùng truy cập vào 1 website sẽ nhận được 2 cookie khác nhau.



HÌNH 6: MINH HỌA COOKIES

- Làm việc với Cookies trong ASP.NET
 - Thêm Cookies: `Response.Cookies.Add(<HttpCookie>);`

Thí dụ:

```
HttpCookie ck = new HttpCookie("TenDangNhap");
ck.Value = txtTenDangNhap.Text;
ck.Expires = DateTime.Now.AddDays(15);
Response.Cookies.Add(ck);
```

Trong thí dụ trên, chúng ta đã tạo ra Cookies có tên là TenDangNhap lưu trữ tên đăng nhập của người dùng. Thông tin này sẽ được lưu trữ trên Cookies 15 ngày kể từ ngày hiện hành trên Web Server.

- Lấy giá trị từ Cookies:

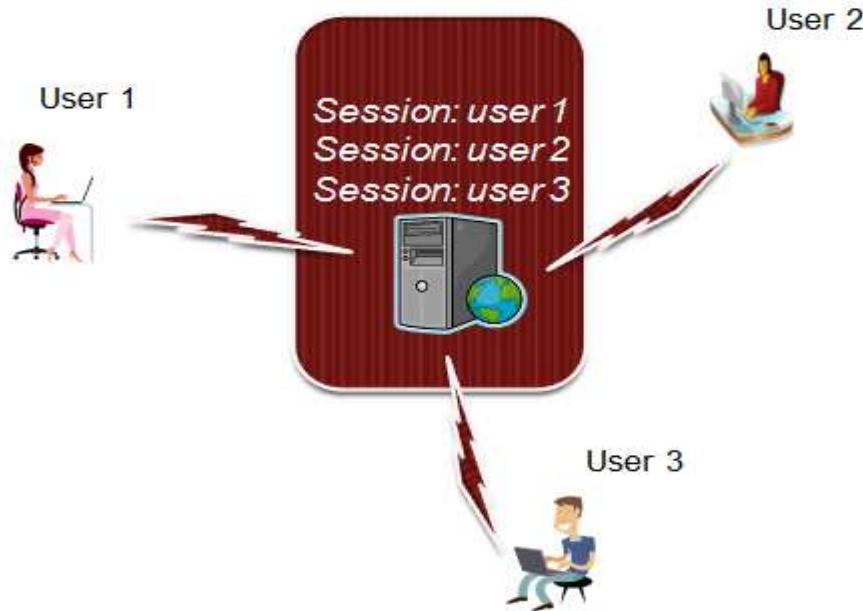
```
HttpCookie ck = Request.Cookies["TenDangNhap"];
string s = ck.Value;
```

Trường hợp Cookies chưa được lưu hoặc đã hết thời hạn duy trì tại Client thì giá trị nhận được là **null**.

- Hủy Cookies: Thiết lập hạn (Expires) của cookies là thời điểm trong quá khứ.

```
<ten_cookies>.Expires = <thoi_diem_trong_qua_khu>;
```

8.3.2 Đối tượng Session



HÌNH 7: SESSION TRONG ASP.NET

- Đối tượng Session được dùng để lưu trữ thông tin của người dùng trong ứng dụng web ứng với một phiên làm việc cụ thể.
- Web Server sẽ tự động tạo một đối tượng Session cho mỗi người dùng mới kết nối vào ứng dụng và lưu trữ trên bộ nhớ của server → tính bảo mật cao rất cao do dữ liệu lưu trữ trên server, không bao giờ chuyển cho client và mỗi client có client SessionId riêng.

- Thời gian sống Session kết thúc sau một khoảng thời gian định nghĩa trước (thường là 20 phút, có thể thay đổi, hoặc khi người lập trình hủy).
- Một số thuộc tính/phương thức:
 - Thuộc tính **Timeout**: chỉ định thời gian (tính bằng phút) mà Web Server duy trì đối tượng Session nếu người dùng không gửi yêu cầu nào về lại Server. Giá trị mặc định là 20 (phút).
 - Phương thức **Abandon()**: Giải phóng toàn bộ giá trị session đã lưu trữ kể cả session identifier.
 - Phương thức **Clear()**: Hủy các giá trị session người dùng lưu trữ ngoại trừ session identifier.
- Sử dụng biến toàn cục với Session:
 - Khai báo biến Session: `Session["Tên Biến"] = <Giá trị>;`
 - Lấy giá trị từ biến Session: `<Biến> = Session["Tên Biến"];`
 - Hủy giá trị từ biến Session: `Session.Remove("Tên Biến");`

8.3.2.1 Ví dụ Session

Ví dụ sau minh họa sử dụng biến session lưu thông tin của người dùng khi đăng nhập.

Bước 1: Thiết kế trang DangNhap.aspx

Control	Tên thuộc tính	Giá trị thuộc tính
Label	Text	Tên đăng nhập
Label	Text	Mật khẩu
TextBox	ID	txtTenDangNhap
TextBox	ID	txtMatKhau
Button	Text	Đăng nhập
	ID	btnDangNhap

ĐĂNG NHẬP

Tên đăng nhập

Mật khẩu

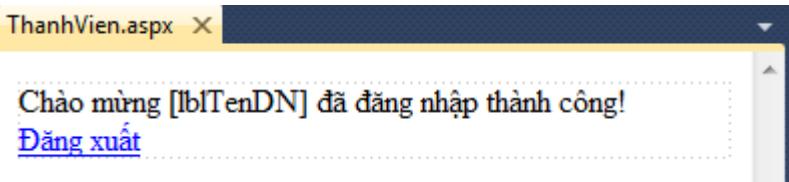
HÌNH 8: THIẾT KẾ GIAO DIỆN TRANG DANGNHAP.ASPX

Bước 2: Viết code behind xử lý sự kiện cho tang DangNhap.aspx:

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

```
//nếu TenDN = admin và Password = 123 thì đăng nhập thành công
if (txtTenDN.Text == "admin" && txtMatKhau.Text == "123")
{
    //ghi nhận session
    Session["TenDN"] = txtTenDN.Text;
    Response.Redirect("ThanhVien.aspx");
}
}
```

Bước 3: Thiết kế trang **ThanhVien.aspx**:



```
public partial class ThanhVien : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
            lblTenDN.Text = Session["TenDN"].ToString();
    }
    protected void LinkButton1_Click(object sender, EventArgs e)
    {
        Session.Remove("TenDN");
    }
}
```

Bước 4: Thực thi các trang

8.4 Đối tượng Application

8.4.1 Mục đích

Đối tượng Application dùng để quản lý tất cả các thông tin trong ứng dụng web. Thông tin được lưu trữ trong đối tượng Application được tạo ra có thể dùng ở bất kỳ trang nào trong site suốt chu kỳ sống của ứng dụng và chỉ được hủy khi kết thúc ứng dụng.

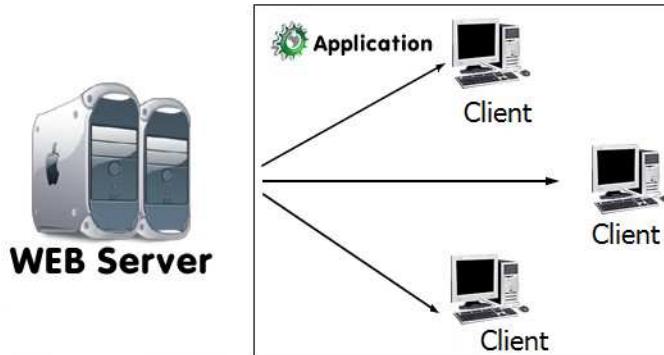
8.4.2 Sử dụng biến Application

- Tạo biến Application:

```
Application["Tên biến"] = <giá trị>;
```

- Lấy giá trị từ biến Application:

```
<biến> = Application["Tên biến"];
```



- Do tại một thời điểm có thể có nhiều người cùng lúc truy cập và thay đổi giá trị của các thông tin được lưu trong đối tượng Application, chúng ta cần Lock() và UnLock() ngay trước và sau khi cập nhật giá trị của biến Application.

- Thí dụ:

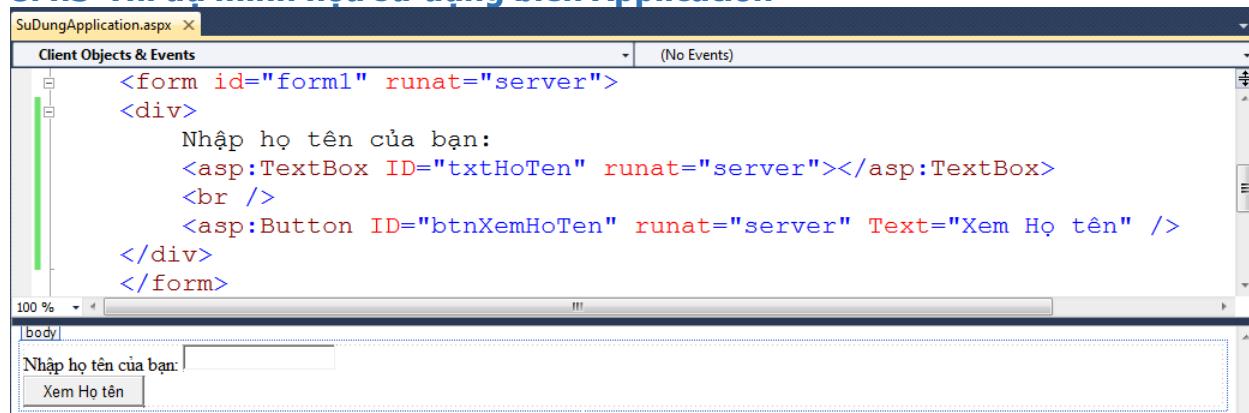
```
Application.Lock()  

Application["SoLanTruyCap"] = 0  

Application["SoNguoiOnLine"] = 0  

Application.UnLock()
```

8.4.3 Thí dụ minh họa sử dụng biến Application



HÌNH 9: THIẾT KẾ TRANG SUĐUNGAPPLICATION.ASPX

Code xử lý sự kiện trang **SuDungApplication.aspx**

```

8 public partial class Chuong08_SuDungApplication : System.Web.UI.Page
9 {
10    protected void Page_Load(object sender, EventArgs e)
11    {
12    }
13    protected void btnXemHoTen_Click(object sender, EventArgs e)
14    {
15        //Khai báo biến ứng dụng
16        Application["HoTen"] = txtHoTen.Text;
17        Response.Redirect("~/LayHoTen.aspx");
18    }
19 }
20

```

Tạo trang **LayHoTen.aspx** và viết lệnh xử lý sự kiện như sau:

```

8 public partial class Chuong08_LayHoTen : System.Web.UI.Page
9 {
10    protected void Page_Load(object sender, EventArgs e)
11    {
12        string s = Application["HoTen"].ToString();
13        Response.Write("Tên của bạn : " + s);
14    }
15 }

```

Kết quả thực hiện:

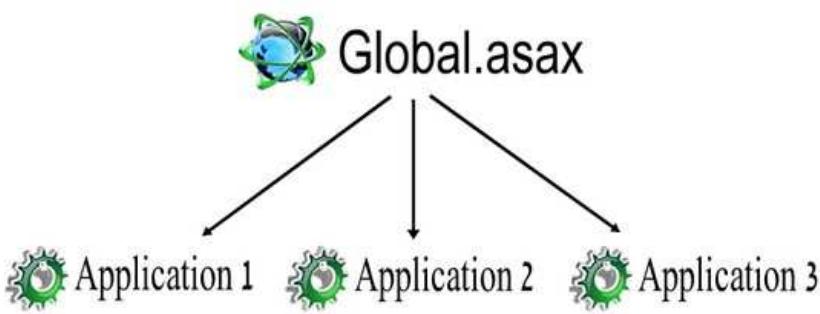


8.5 Tập tin Global.asax

8.5.1 Mục đích

Tập tin Global.asax được dùng để:

- Khai báo và khởi tạo giá trị cho các biến Application, Session.
- Viết xử lý cho các sự kiện của 2 đối tượng Application và Session.



HÌNH 10: TẬP TIN GLOBAL.ASAX TRONG ASP.NET

8.5.2 Cấu trúc tập tin Global.asax

```

<%@ 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
    }

    void Session_End(object sender, EventArgs e)
    {
        // Code that runs when a session ends.
        // Note: The Session_End event is raised only when the sessionstate mode
        // is set to InProc in the Web.config file. If session mode is set to
        StateServer
        // or SQLServer, the event is not raised.
    }
</script>
  
```

8.5.3 Các sự kiện trong tập tin Global.asax

- **Application_Start:** Chỉ xảy ra một lần đầu tiên khi bất kỳ trang nào trong ứng dụng được gọi. Ứng dụng xây dựng tính năng đếm số người online, truy cập trong website.

```
void Application_Start(object sender, EventArgs e)
{
    // Khai báo đếm số người truy cập
    Application["So_luot_truy_cap"] = 0;
    Application["So_nguo_online"] = 0;
}
```

- **Session_Start:** Xảy ra khi có một người dùng mới yêu cầu đến bất kỳ trang aspx của ứng dụng. Khi Session_Start xảy ra, một giá trị duy nhất (SessionID) sẽ được tạo cho người dùng, và giá trị này được sử dụng để quản lý người dùng trong quá trình làm việc với ứng dụng.

```
void Session_Start(object sender, EventArgs e)
{
    // Tăng giá trị biến Application
    Application["So_luot_truy_cap"] =
        (int)Application["So_luot_truy_cap"] + 1;
    Application["So_nguo_online"] =
        (int)Application["So_nguo_online"] + 1;
}
```

- **Application_BeginRequest:** Xảy ra khi mỗi khi có Postback về Server.
- **Application_Error:** Xảy ra khi có lỗi phát sinh trong quá trình thi hành. Thông thường chỉ định trang hiển thị thông báo lỗi.

```
void Application_Error(object sender, EventArgs e)
{
    // Code that runs when an unhandled error occurs
    Response.Redirect("~/ThongBaoLoi.aspx");
}
```

- **Session_End:** Xảy ra khi phiên làm việc không có gởi yêu cầu hoặc làm tươi trang aspx của ứng dụng web trong một khoảng thời gian (mặc định là 20 phút).

```
void Session_End(object sender, EventArgs e)
{
    // Giảm giá trị biến Application
    Application["So_nguo_online"] =
        (int)Application["So_nguo_online"] - 1;
}
```

- **Application_End:** Xảy ra khi dừng hoạt động của WebServer. Ví dụ xử lý ghi nhận thông tin Số lượt truy cập vào cơ sở dữ liệu (nếu cần).

8.6 Tập tin Web.config

8.6.1 Cấu trúc tập tin web.config

Web.config là một tập tin văn bản được sử dụng để lưu trữ thông tin cấu hình của một ứng dụng, được tự động tạo ra khi chúng ta tạo mới ứng dụng web. Tập tin web.config được viết theo định dạng XML.

Web.config được tạo kế thừa các giá trị từ tập tin Windows\Microsoft.NET\Framework\[FrameworkVersion]\CONFIG\machine.config.

```
<?xml version="1.0"?>
<configuration>
    <configSections>
        . . .
    </configSections>

    <system.web>
        . . .
    </system.web>
</configuration>
```

Minh họa tập tin cấu hình Web.config

8.6.2 Thiết lập các cấu hình

- Set ngôn ngữ và Bật/tắt chế độ debug:

```
<compilation debug="true" targetFramework="4.0" />
```

- Cấu hình về xử lý lỗi:

```
<customErrors mode="RemoteOnly" />
```

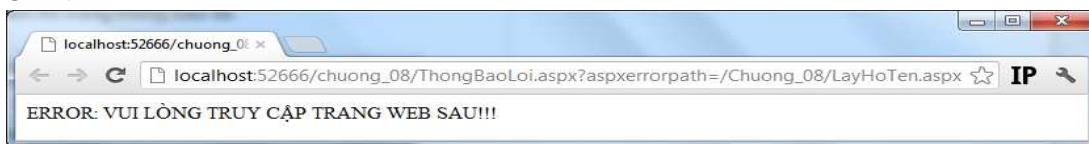
Cấu hình này cho phép chúng ta quản lý việc xử lý lỗi khi có lỗi phát sinh trong ứng dụng. Thuộc tính mode có các giá trị: *RemoteOnly*, *On* và *Off*.

- *RemoteOnly*: Cho phép người dùng thấy thông báo lỗi của hệ thống hoặc trang thông báo lỗi được chỉ định qua defaultRedirect (nếu có). Thông báo lỗi gồm: Mã lỗi và mô tả lỗi tương ứng.
- *Off*: cho phép người dùng thấy lỗi trực tiếp (ở dòng nào, trang gì). Điều này hết sức nguy hiểm khi triển khai ứng dụng.

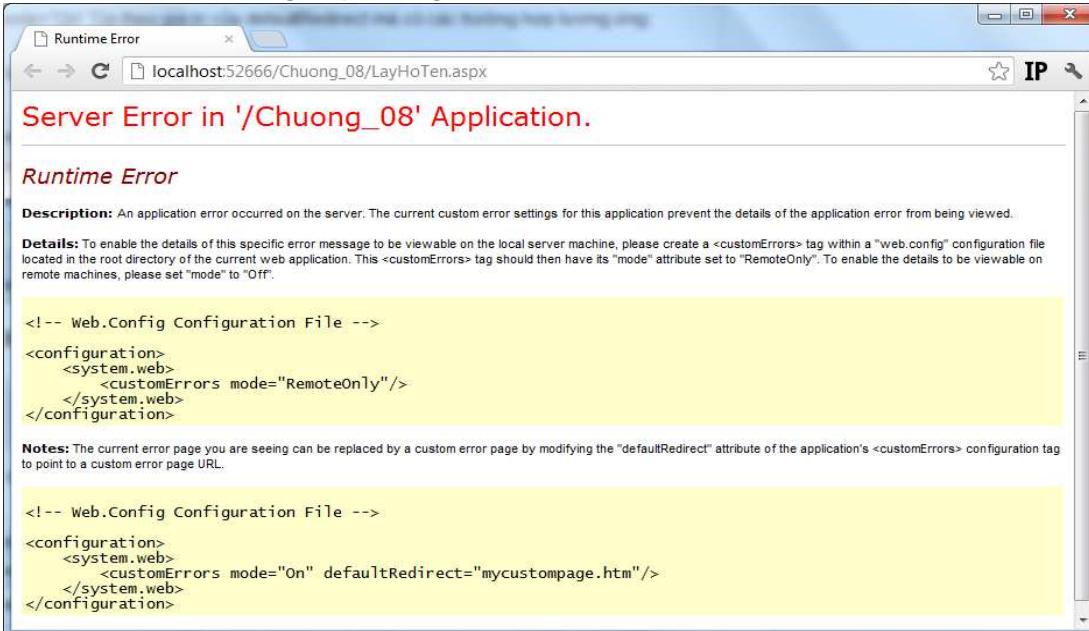


On: Tức là mode="On" Tùy theo giá trị của defaultRedirect mà có các trường hợp tương ứng:

Trường hợp chỉ định thuộc tính defaultRedirect



Trường hợp không chỉ định thuộc tính defaultRedirect



Lưu ý: Nếu có cài đặt sự kiện Application_Error thì sẽ xử lý sự kiện này trước.

- sessionState:

Ví dụ:

```
<sessionState  
    mode="InProc"  
    stateConnectionString="tcpip=127.0.0.1:42424"  
    sqlConnectionString="data source=127.0.0.1;  
    Trusted_Connection=yes"  
    cookieless="false"  
    timeout="20" />
```

- **mode:** Thuộc tính này có 3 giá trị: InProc, sqlserver (lưu trong database), và stateserver (lưu trong bộ nhớ)
 - **stateConnectionString:** Cấu hình địa chỉ và cổng (port) của máy để lưu trữ thông tin của Session trong vùng nhớ (nếu chức năng này được chọn).
 - **sqlConnectionString:** Cấu hình kết nối đến SQL Server được dùng để lưu thông tin Session (nếu chức năng này được chọn).
 - **cookieless:** Nếu giá trị của thuộc tính này = True, thông tin cookie sẽ được lưu trữ trong URL, ngược lại, nếu = False, thông tin cookies sẽ được lưu trữ tại client (nếu client có hỗ trợ)
 - **timeout:** Khoảng thời gian (tính bằng phút) mà đối tượng Session được duy trì. Sau khoảng thời gian này, đối tượng Session sẽ bị huỷ. Giá trị mặc định của thuộc tính này là 20.
- **ConnectionString:** Cấu hình thông tin chuỗi kết nối CSDL:

```
<connectionStrings>  
    <add name="eStore" connectionString="data source=.;Integrated  
        Security=SSPI;AttachDBFilename=|DataDirectory|\eStore.mdf;"  
        providerName="System.Data.SqlClient"/>  
</connectionStrings>
```

Trên đây là một số cấu hình thường dùng trong tập tin Web.config.

Chương 9: **SQL SERVER 2012**

9.1 Tổng quan về SQL và cơ sở dữ liệu quan hệ

9.1.1 Khái niệm SQL

- SQL (Structured Query Language - ngôn ngữ hỏi có cấu trúc) là công cụ sử dụng để tổ chức, quản lý và truy xuất dữ liệu được lưu trữ trong các cơ sở dữ liệu.
- SQL là một hệ thống ngôn ngữ bao gồm tập các câu lệnh sử dụng để tương tác với cơ sở dữ liệu quan hệ.
- SQL được sử dụng để điều khiển tất cả các chức năng mà một hệ quản trị cơ sở dữ liệu cung cấp cho người dùng bao gồm:
 - o **Định nghĩa dữ liệu:** SQL cung cấp khả năng định nghĩa các cơ sở dữ liệu, các cấu trúc lưu trữ và tổ chức dữ liệu cũng như mối quan hệ giữa các thành phần dữ liệu.
 - o **Truy xuất và thao tác dữ liệu:** Với SQL, người dùng có thể dễ dàng thực hiện các thao tác truy xuất, bổ sung, cập nhật và loại bỏ dữ liệu trong các cơ sở dữ liệu.
 - o **Điều khiển truy cập:** SQL có thể được sử dụng để cấp phát và kiểm soát các thao tác của người sử dụng trên dữ liệu, đảm bảo sự an toàn cho cơ sở dữ liệu.
 - o **Đảm bảo toàn vẹn dữ liệu:** SQL định nghĩa các ràng buộc toàn vẹn trong cơ sở dữ liệu nhờ đó đảm bảo tính hợp lệ và chính xác của dữ liệu trước các thao tác cập nhật cũng như các lỗi của hệ thống.

9.1.2 Vai trò của SQL

- SQL không phải là một hệ quản trị cơ sở dữ liệu, do nó không thể tồn tại độc lập.
- SQL là một phần của hệ quản trị cơ sở dữ liệu, nó xuất hiện trong các hệ quản trị cơ sở dữ liệu với vai trò ngôn ngữ và là công cụ giao tiếp giữa người sử dụng và hệ quản trị cơ sở dữ liệu.
- SQL có những vai trò như sau:
 - o **SQL là ngôn ngữ hỏi có tính tương tác:** Người sử dụng có thể dễ dàng thông qua các trình tiện ích để gửi các yêu cầu dưới dạng các câu lệnh SQL đến cơ sở dữ liệu và nhận kết quả trả về từ cơ sở dữ liệu.
 - o **SQL là ngôn ngữ lập trình cơ sở dữ liệu:** Các lập trình viên có thể nhúng các câu lệnh SQL vào trong các ngôn ngữ lập trình để xây dựng nên các chương trình ứng dụng giao tiếp với cơ sở dữ liệu.
 - o **SQL là ngôn ngữ quản trị cơ sở dữ liệu:** Thông qua SQL, người quản trị cơ sở dữ liệu có thể quản lý được cơ sở dữ liệu, định nghĩa các cấu trúc lưu trữ dữ liệu, điều khiển truy cập cơ sở dữ liệu,...
 - o **SQL là ngôn ngữ cho các hệ thống khách/chủ (client/server):** Trong các hệ thống cơ sở dữ liệu khách/chủ, SQL được sử dụng như là công cụ để giao tiếp giữa các trình ứng dụng phía máy khách với máy chủ cơ sở dữ liệu.
 - o **SQL là ngôn ngữ truy cập dữ liệu trên Internet:** Cho đến nay, hầu hết các máy chủ Web cũng như các máy chủ trên Internet sử dụng SQL với vai trò là ngôn ngữ để tương tác với dữ liệu trong các cơ sở dữ liệu.
 - o **SQL là ngôn ngữ cơ sở dữ liệu phân tán:** Đối với các hệ quản trị cơ sở dữ liệu phân tán, mỗi một hệ thống sử dụng SQL để giao tiếp với các hệ thống khác trên mạng, gửi và nhận các yêu cầu truy xuất dữ liệu với nhau.
 - o **SQL là ngôn ngữ sử dụng cho các cổng giao tiếp cơ sở dữ liệu:** Trong một hệ thống mạng máy tính với nhiều hệ quản trị cơ sở dữ liệu khác nhau, SQL thường được sử dụng như là một chuẩn ngôn ngữ để giao tiếp giữa các hệ quản trị cơ sở dữ liệu (HQTCSDL).

9.1.3 Mô hình dữ liệu quan hệ

- CSDL quan hệ là một CSDL trong đó tất cả dữ liệu được tổ chức trong các bảng (table) có mối quan hệ với nhau. Mỗi bảng (table) bao gồm các dòng (record/bản ghi/bộ) và các cột field/trường/thuộc tính).
- Tóm lại, một CSDL bao gồm nhiều bảng (table) có mối quan hệ với nhau (relationship).
- Ví dụ:

Số TT đại lý chung

Bảng: Khách hàng

TT Khách hàng	Tên khách hàng	SDT	TT Đại lý
1231234	Nguyễn thị A	123456	3445
1231235	Vũ văn B	123449	3322
1231236	Trần văn C	223455	2234
1231237	Phạm văn D	334555	4445
1231238	Lê thị S	234444	2222

Bảng: Đại lý

TT Đại lý	Tên đại lý
3322	Phùng A
3445	Trần X
2234	Đỗ Y

9.2 Bảng và ràng buộc

9.2.1 Bảng (Table)

Bảng (table) bao gồm các yếu tố sau:

- Tên của bảng: được xác định duy nhất.
- Cấu trúc của bảng: tập hợp các cột (field/trường/thuộc tính).
- Dữ liệu của bảng: tập hợp các dòng (record/bản ghi/bộ) hiện có trong bảng.

Ví dụ: Table **DONVI**

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	8214514
02	Phòng Tổ chức	8314144
03	Phòng Điều hành	8232356
04	Phòng Đối ngoại	8412345
05	Phòng Tài vụ	8214515

9.2.2 Khóa chính của bảng (Primary Key)

- Mỗi bảng phải có một cột (hoặc một tập các cột) mà giá trị dữ liệu của nó xác định duy nhất một dòng trong tập hợp các dòng trong bảng.
- Một cột (hoặc một tập các cột) có tính chất này gọi là khóa chính của bảng (Primary Key).
- Ví dụ: Table **DONVI** ở trên có khóa chính là MADONVI.

9.2.3 Mối quan hệ (Relationship) và khóa ngoại (Foreign Key)

- Mối quan hệ (Relationship) được thể hiện thông qua ràng buộc giá trị dữ liệu xuất hiện ở bảng này phải có xuất hiện trước ở một bảng khác.
- Một cột (hoặc tập hợp các cột) (field/trường/thuộc tính) trong một bảng mà giá trị của nó được xác định từ khóa chính (Primary Key) của một bảng khác được gọi là khóa ngoại (Foreign Key).

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	8214514
02	Phòng Tổ chức	8314144
03	Phòng Điều hành	8232356
04	Phòng Đối ngoại	8412345
05	Phòng Tài vụ	8214515

MANV	HOTEN	NGAYSINH	DIACHI	DIENTHOAI	MADONVI
NV01001	Nguyễn Ngọc Hoa	15/05/1985	123 Trường Định	38352030	01
NV02001	Lê Thanh Tùng	03/05/1996	32 Trần Phú	38236463	02
NV02002	Hoàng Đình Tùng	08/08/1988	66 Hoàng Diệu	39353535	02
NV03001	Nguyễn Ngọc	19/09/1989	77 Nguyễn Huệ	39292174	03
NV03002	Lý Thanh Tùng	12/02/1992	7 Thành Thái	26636363	03
NV04001	Lê Sao Mai	06/05/1965	123 Lê Lợi	0909123654	04

9.3 Sơ lược về câu lệnh SQL

9.3.1 Các câu lệnh

Câu lệnh		Chức năng
<i>Thao tác dữ liệu</i>		
SELECT	Truy vấn dữ liệu	
INSERT	Thêm mới dữ liệu	
UPDATE	Sửa/Cập nhật dữ liệu	
DELETE	Xóa dữ liệu	
TRUNCATE	Xóa toàn bộ dữ liệu trong bảng	
<i>Định nghĩa dữ liệu</i>		
CREATE TABLE	Tạo bảng	
DROP TABLE	Xóa bảng	
ALTER TABLE	Sửa bảng	
CREATE FUNCTION	Tạo hàm (do người sử dụng định nghĩa)	
ALTER FUNCTION	Sửa đổi hàm	
DROP FUNCTION	Xóa hàm	
CREATE TRIGGER	Tạo trigger	
ALTER TRIGGER	Sửa trigger	
DROP TRIGGER	Xóa trigger	

9.3.2 Quy tắc sử dụng tên trong SQL

- Trong câu lệnh SQL, nếu ta cần chỉ đến một bảng *do một người dùng khác sở hữu* (hiển nhiên là phải được phép) thì *tên của bảng* phải được viết sau *tên của người sở hữu* và *phân cách* với tên người sở hữu *bởi dấu chấm* theo công thức: *tên_người_sở_hữu.tên_bảng*
- Trong câu lệnh SQL, nếu có sử dụng từ *hai cột* trở lên có *cùng tên* trong các bảng khác

nhau thì bắt buộc phải chỉ định thêm *tên bảng trước tên cột*; *tên bảng* và *tên cột* được phân cách nhau bởi dấu chấm theo công thức: *tên_bảng.tên_cột*

9.4 Kiểu dữ liệu

Tên kiểu	Mô tả
CHAR (n)	Kiểu chuỗi với độ dài cố định
NCHAR (n)	Kiểu chuỗi với độ dài cố định hỗ trợ UNICODE
VARCHAR (n)	Kiểu chuỗi với độ dài chính xác
NVARCHAR (n)	Kiểu chuỗi với độ dài chính xác hỗ trợ UNICODE
INTEGER	Số nguyên có giá trị từ -2^{31} đến $2^{31} - 1$
INT	Giống kiểu INTEGER
TINYINT	Số nguyên có giá trị từ 0 đến 255
SMALLINT	Số nguyên có giá trị từ -2^{15} đến $2^{15} - 1$
BIGINT	Số nguyên có giá trị từ -2^{63} đến $2^{63} - 1$
NUMERIC (p, s)	Kiểu số với độ chính xác cố định
DECIMAL (p, s)	Giống kiểu NUMERIC
FLOAT	Số thực có giá trị từ $-1.79E+308$ đến $1.79E+308$
REAL	Số thực có giá trị từ $-3.4E+38$ đến $3.4E+38$
MONEY	Kiểu tiền tệ
BIT	Kiểu bit (có giá trị 0 hoặc 1)
DATETIME	Kiểu ngày giờ (chính xác đến phần trăm của giây)
SMALLDATETIME	Kiểu ngày giờ (chính xác đến phút)
BINARY	Dữ liệu nhị phân với độ dài cố định (tối đa 8000 bytes)
VARBINARY	Dữ liệu nhị phân với độ dài chính xác (tối đa 8000 bytes)
IMAGE	Dữ liệu nhị phân với độ dài chính xác ($\leq 2,147,483,647$ bytes)
TEXT	Dữ liệu kiểu chuỗi với độ dài lớn (tối đa 2,147,483,647 ký tự)
NTEXT	Dữ liệu kiểu chuỗi với độ dài lớn và hỗ trợ UNICODE (tối đa 1,073,741,823 ký tự)

9.5 Toán tử

Toán tử	Ý nghĩa
a) Logic	
AND / OR	Và / Hoặc
b) So sánh	
=	Bằng
>	Lớn hơn
>=	Lớn hơn hay bằng
<	Nhỏ hơn
<=	Nhỏ hơn hay bằng
<> hoặc !=	Khác
c) Danh sách	
IN	Nằm trong danh sách
NOT IN	Không nằm trong danh sách
d) Giới hạn dữ liệu	
BETWEEN	BETWEEN a AND b: nghĩa là $a \leq$ giá trị $\leq b$
NOT BETWEEN	NOT BETWEEN a AND b: nghĩa là (giá trị $< a$) và (giá trị $> b$)
LIKE	Mô tả định dạng dữ liệu sử dụng ký tự đại diện: <ul style="list-style-type: none"> %: ký tự bất kỳ (không hoặc nhiều) _: một ký tự bất kỳ []: một ký tự bất kỳ nằm trong danh sách chỉ định [^]: một ký tự bất kỳ không nằm trong danh sách chỉ định

9.6 View, Stored Procedure, Trigger, Function

SQL Server cho phép bạn tạo ra 4 đối tượng bằng cách lập trình: Stored Procedure, View, Trigger, Function.

9.6.1 Bảng ảo – View

- View có thể được xem như một Table ảo (nó không được lưu trữ lên đĩa về mặt vật lý như Table thông thường) mà dữ liệu của nó được lấy ra từ một câu truy vấn, có chứa cột và dữ liệu từ một hay nhiều Table khác nhau, hay từ những View khác nhau.
- Đặc điểm của View là ta có thể join dữ liệu từ nhiều Table và trả về một tập kết quả đơn. Ngoài ra ta có thể thao tác dữ liệu trước khi trả về cho user bằng cách dùng các lệnh SQL như where, case...
- Lợi ích của View.
 - Có khả năng tăng tính bảo mật. View giúp ta che giấu cấu trúc của câu truy vấn bên trong. Sau khi tạo xong View ta có thể Insert, Delete, Update như 1 Table bình thường.
 - Giảm độ phức tạp. Ví dụ như User chỉ muốn xem thông tin của một vài cột với một điều kiện nào đó, View cung cấp đúng dữ liệu họ muốn mà không có dữ liệu thừa.
- Cú pháp tạo VIEW:

```
CREATE VIEW <tên_view>
AS
<câu_lệnh_sql>
```

- Sử dụng VIEW: giống như table.

9.6.2 Stored Procedure

- SP (Stored Procedure) là tên được đặt cho một nhóm các mệnh đề SQL được tạo ra và lưu trong server database.
- SP cho phép truyền tham số, và có thể được gọi bởi nhiều Client qua mạng với các tham số khác nhau. Khi SP bị thay đổi, tất cả Client sẽ tự động nhận được bản mới, vì SP được lưu ở Server, chứ không phải Client.
- SQL Server cung cấp một số các thủ tục được lưu trữ sẵn trong hệ thống giúp thực hiện một số công việc thường xuyên. Nó được gọi là thủ tục hệ thống – System stored procedures. Còn những thủ tục do người sử dụng tự viết gọi là User stored procedures.
- Lợi ích của SP:
 - Tăng tốc độ thực hiện, tốc độ truy cập dữ liệu nhanh hơn
 - Chương trình được module hóa
 - Nhất quán
 - Nâng cao khả năng bảo mật dữ liệu
- Cú pháp định nghĩa User-defined Stored Procedure:

```
CREATE PROC[EDURE] <procedure_name>
    <@ParaName1> <DataType1> = <DefaultValue1>,
    <@ParaName2> <DataType2> = <DefaultValue2>,
    ...
    <@ParaNameN> <DataTypeN> = <DefaultValueN>
AS
BEGIN
    --SQL statements
END
```

- Thực thi Stored Procedure:

EXEC[UTE] <tên_stored_procedure> [danh_sách_tham_số]

9.6.3 Trigger

- Trigger gắn liền với một bảng và được tự động thực hiện khi có sự thay đổi dữ liệu (Insert, Delete, hay Update tác động trên một bảng cụ thể).
- Tuy nhiên khác với Stored Procedure, Trigger hoàn toàn không có tham số.
- Trigger có thể gọi thực thi Stored Procedure và được lưu trữ, quản lý trên Server Database.
- Dùng Trigger trong trường hợp ta muốn kiểm tra các ràng buộc toàn vẹn trong Database.
- Cú pháp chung để tạo một Trigger như sau:

```
CREATE TRIGGER Ten_Trigger
ON Ten_Bang
FOR {[INSERT] | [UPDATE] | [DELETE]}
AS
BEGIN
    --Cac_Cau_Lenh_Cua_Trigger
END
```

9.6.4 Function

- Function là hàm trong ngôn ngữ lập trình để thực hiện một phép tính hay một xử lý nào đó.
- Mục đích: Dùng để tính giá trị từ 01 hay nhiều câu lệnh SQL
- Cú pháp:

```
CREATE FUNCTION <tên_hàm>
(
    -- danh_sách_tham_số
    <@ParaName1> <DataType>
)
RETURNS <ReturnDataType>
AS
BEGIN
    -- Câu lệnh SQL
    RETURN <kết_quả_trả_về>
END
```

- Ví dụ:

```
CREATE FUNCTION fnDoanhSo
(
    @MaHH INT
)
RETURNS FLOAT
AS
BEGIN
    DECLARE @DoanhSo FLOAT

    SELECT @DoanhSo = SUM(SoLuong * DonGia) FROM ChiTietHoaDon
    WHERE MaHH = @MaHH

    RETURN @DoanhSo
END
```

Và gọi hàm (chú ý thêm dbo trước tên hàm):

```
SELECT MaHH, dbo.fnDoanhSo(MaHH) FROM HangHoa.
```

Chương 10: ADO.NET & WEB FORM

Sau khi học xong chương này, học viên có khả năng:

- Trình bày được các thành phần cơ bản của ADO.Net
- Xây dựng được các trang ASP.Net tương tác với CSDL thông qua Data Providers và Data Classes
- Thực hiện được các ràng buộc dữ liệu từ Data Classes với các DataControls
- Mô tả và sử dụng được Template Field của Data Controls
- Thực hiện được kết nối và thao tác dữ liệu sử dụng SQLDataSource Control

10.1 Kiến Trúc ADO.NET

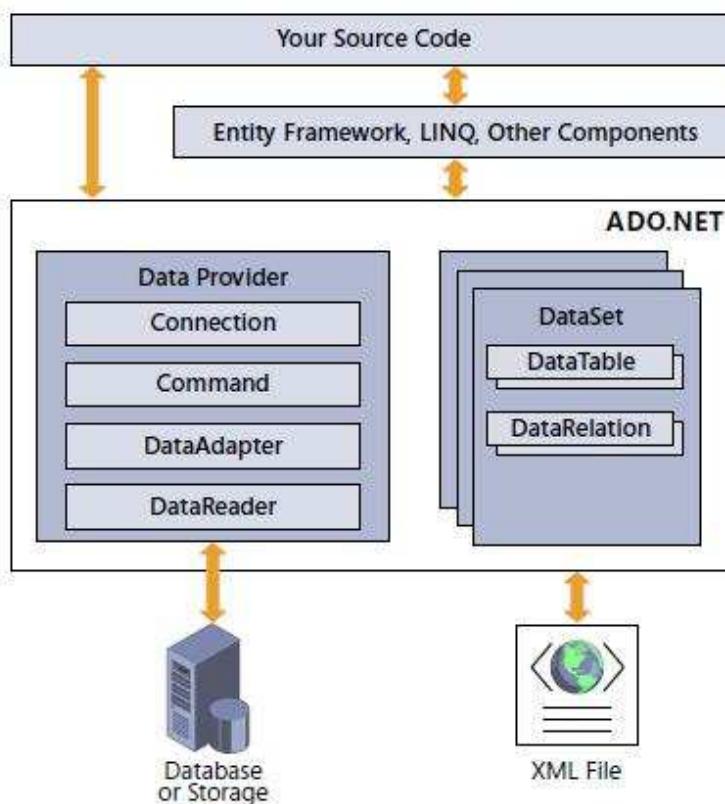
10.1.1 ADO.Net là gì?

Trước đây, trong thế giới lập trình các nhà phát triển sử dụng một phương thức duy nhất để truy xuất dữ liệu đó là: **tương tác trực tiếp với các giá trị được lưu trong bộ nhớ**.

Ngày nay do nhu cầu xử lý dữ liệu lớn, các nhà phát triển cần một giải pháp thống nhất để tương tác với các hệ thống lưu trữ dữ liệu khác nhau (như dữ liệu quan hệ, dữ liệu XML, ...). .Net Framework cung cấp cung cấp thư viện *ADO.NET* đã đáp ứng được nhu cầu đó. Thay vì phải quan tâm đến những chi tiết vụn vặt của từng hệ thống lưu trữ dữ liệu khác nhau, với *ADO.NET* họ chỉ tập trung vào chính nội dung của dữ liệu.

10.1.2 Kiến trúc ADO.Net

Hình sau đây cho thấy kiến trúc của ADO.Net:



Chúng ta có thể chia các lớp của *ADO.NET* thành 2 nhóm: *Provider* (nhóm cung cấp) và *Consumer* (nhóm tiêu thụ). Việc đọc/ghi dữ liệu sẽ được thực hiện bởi các đối tượng *Provider*. Trong khi đó các đối tượng *Consumer* là công cụ để chúng ta thao tác với dữ liệu sau khi đã đọc nó vào bộ nhớ trong. Các đối tượng *Consumer* làm việc khi kết nối đến nguồn dữ liệu đã được ngắt. Còn các đối tượng *Provider* đòi hỏi một kết nối đang được kích hoạt.

Các thành phần của *ADO.NET* sẽ được trình bày chi tiết trong các phần sau.

10.2 Data Provider

.NET *Data Providers* là một tập các đối tượng phục vụ cho việc trao đổi dữ liệu giữa *Data Source* (dữ liệu nguồn) và đối tượng *DataSet*.

ADO.NET cung cấp các *Data Providers* sau:

- Microsoft SQL Server Provider trong namespace *System.Data.SqlClient*.
- *OLE DB Provider* trong namespace the *System.Data.OleDb*.
- *ODBC Provider* trong namespace *System.Data.Odbc*.

Tuy nhiên trong phạm vi giáo trình này, tập trung vào việc thao tác hệ quản trị CSDL SQL Server, do đó chúng tôi sẽ chỉ trình bày việc sử dụng Microsoft SQL Server Provider.

10.2.1 Đối tượng Connection

Tất cả các giao tiếp với dữ liệu bên ngoài đều thông qua đối tượng *Connection*.

Đối tượng *Connection* chịu trách nhiệm quản lý kết nối tới nguồn dữ liệu (*DataSource*). Có 2 dạng *Connection* tương ứng với 2 kiểu dữ liệu SQL Server và OLE DB đó là : *SqlConnection* và *OleDbConnection*. Cả 2 đối tượng này đều implement từ interface *IDbConnection*. Bằng cách sử dụng Interface *IDbConnection*, các nhà cung cấp dịch vụ Database khác nhau có thể tạo ra các cài đặt phù hợp cho Database riêng của họ.

Đối tượng *Connection* của *ADO.NET* chỉ nhận một tham số đầu vào là chuỗi kết nối (*connection string*). Trong chuỗi kết nối, các thông số được cách nhau bằng dấu ";", *connection string* có các thông số sau :

- **Provider** : Tên nhà cung cấp Database, đối với *OLEDB* cần khai báo là *SQLOLEDB*. Đối với SQL Server thì không cần thuộc tính này.
- **DataSource** (hoặc Server) : tên/địa chỉ database server cần kết nối tới.
- **Initial catalog** (hoặc Database) : tên của Database cần truy xuất.
- **User id** : username để đăng nhập vào Database Server.
- **Password** : password để đăng nhập vào Database Server.

Ví dụ:

```
string connectionString = "Data Source=.;Initial Catalog=HSDB;Integrated Security=SSPI;";
```

(Các kiểu kết nối khác nhau bạn có thể tham khảo tại website: <http://www.connectionstrings.com/>)

Sau đây là các thuộc tính, phương thức, sự kiện thông dụng của cả *SqlConnection* và *OleDbConnection* :

Thuộc tính

- **ConnectionString** : thuộc tính thiết lập / lấy chuỗi kết nối.
- **ConnectionTimeout**: thuộc tính thiết lập / lấy thời gian chờ trong khi truy xuất vào database. Khi truy xuất vào Database, chương trình sẽ chờ đúng khoảng thời gian

này nếu chờ qua khoảng thời gian này mà vẫn không kết nối được vào database thì chương trình sẽ báo lỗi.

- **Database**: thuộc tính thiết lập/lấy tên database của đối tượng connection hiện thời.
- **DataSource** : thuộc tính thiết lập/lấy tên của database server của đối tượng connection hiện thời.
- **State** : Lấy trạng thái hiện thời của Connection có các trạng thái sau: *Connecting, Broken, Open, Closed, Executing, Fetching*.

Phương thức

- **BeginTransaction()**: sử dụng cho trường hợp xử lý giao tác của ứng dụng. Việc xử lý giao tác rất có lợi trong khi xử lý dữ liệu từ database vì có lúc trong khi xử lý dữ liệu gặp lỗi thì có thể thực hiện câu lệnh như Rollback hay trong lúc thao tác cũng có thể thực hiện được các giao tác chính như trên SQL Server như Commit...
- **Close()**: đóng 1 connection.
- **Open()**: Mở 1 connection với các thuộc tính hiện hành.

Sự kiện

- **InfoMessage** : Xảy ra khi provider gửi ra 1 lời cảnh cáo hay thông tin.
- **StateChange** : Xảy ra khi trạng thái của connection thay đổi.

Sau đây là đoạn source code sử dụng đối tượng *Connection* để kết nối vào cở sở dữ liệu SQL Server 2008:

```
using System.Data.SqlClient;
static public class DataAL
{
    static private string connectionString = "Data Source=.;Initial Catalog=HSDB;Integrated Security=SSPI;";
    static public bool TestConnection()
    {
        SqlConnection conn = new SqlConnection(connectionString);
        try
        {
            if (conn.State == ConnectionState.Closed)
            {
                conn.Open();
            }
            return true;
        }
        catch
        {
            return false;
        }
        finally
        {
            conn.Close();
        }
    }
    ...
}
```

10.2.2 Đối tượng Command

ADO.NET sử dụng đối tượng *Command* để các truy vấn SQL và các lệnh quản lý dữ liệu trước khi gửi chúng đến đích. *Command* sử dụng các *Parameter* cho phép bạn gọi các thủ tục nội tại (*stored procedures*) hoặc các truy vấn.

Ngoài ra, *DataAdapter* chứa các định nghĩa truy vấn chuẩn để thao tác với dữ liệu, thay vì bạn phải viết các chuỗi truy vấn SQL để thao tác; *DataReader* cung cấp khả năng đọc dữ liệu một cách nhanh chóng.

Sau khi chúng ta đã kết nối vào nguồn dữ liệu, chúng ta cần phải thực hiện thao tác trên các dữ liệu. Để thao tác được với các dữ liệu chúng ta phải dùng đối tượng *Command*. Đối tượng *Command* là đối tượng rất "đa năng", nó vừa xử lý được *SQL Server Stored Procedures* vừa xử lý được các giao tác (*Transaction*).

Tương tự như đối tượng *Connection*, đối tượng *Command* cũng chia ra làm 2 loại tùy theo nguồn dữ liệu : *SqlCommand* (cho SQL Server) và *OleDbCommand* (cho OLE DB).

Một số thuộc tính và phương thức thông dụng của đối tượng *Command*:

Thuộc tính:

- **CommandText** : Thiết lập/ Lấy lệnh thao tác với dữ liệu.
- **CommandTimeout** : Thiết lập/ Lấy thời gian chờ thực hiện lệnh. Sau khi chờ 1 khoảng thời gian nếu vượt quá sẽ báo lỗi.
- **CommandType** : Thiết lập/ Lấy kiểu của đối tượng lệnh (lệnh trực tiếp, stored procedure...)
- **Parameters** : Các tham số truyền vào cho đối tượng *command* (ở kiểu *OleDbParameterCollection*/ *SqlParameterCollection*).
- **Connection** : Thiết lập / lấy kết nối đang được đối tượng *Command* sử dụng.
- **Transaction** : Thiết lập / lấy giao tác mà đối tượng *Command* thực thi.

Phương thức:

- **ExecuteReader()** : Thực thi câu lệnh *CommandText* của đối tượng *Command* và trả về kiểu *DataReader* (*OleDbDataReader* / *SqlDataReader*).
- **ExecuteNonQuery()** : Thực thi câu lệnh *CommandText* của đối tượng *Command*, đây là dạng câu lệnh cập nhật cơ sở dữ liệu (xoá /sửa) nên chỉ trả về số dòng bị ảnh hưởng mà không trả về dòng dữ liệu nào.
- **ExecuteScalar()** : Thực thi câu truy vấn của đối tượng *Command* và chỉ trả về cột đầu tiên của dòng đầu tiên của kết quả. Các kết quả còn lại bị bỏ qua.
- Các hàm khởi tạo của đối tượng *SqlCommand* (Tương tự cho đối tượng *OleDbCommand*):
 - **New()**: không có tham số nào.
 - **New(cmdText as String)**,
trong đó *cmdText* là câu lệnh truyền vào cho đối tượng *Command*.
 - **New(cmdText as String, connectin as SqlConnection)**,
trong đó *cmdText* như trên, *connection* là đối tượng kết nối truyền vào cho đối tượng *Command*.
 - **New(cmdText as String, connection as SqlConnection, transaction as SqlTransaction)**,

trong đó *cmdText, connection* như trên, *Transaction* : là giao tác truyền cho đối tượng *Command*.

10.3 Data Classes

Không gian tên *System.Data* bao gồm các lớp ADO.NET khác nhau cùng làm việc để cung cấp các phương thức truy xuất dữ liệu dạng bảng. Thư viện này chứa 2 nhóm chính, một quản lý dữ liệu trong bộ nhớ chính, và một giao tiếp với các hệ thống lưu trữ dữ liệu ngoài.

Lớp trung tâm của thư viện này là *DataTable* (tương tự như các bảng trong CSDL). *DataTable* chứa các dữ liệu mà chương trình của bạn cần thao tác. Mỗi *DataTable* hoặc rỗng hoặc chứa các dòng dữ liệu – *DataRow*, các giá trị trên mỗi dòng được phân biệt với nhau bởi các cột – *DataColumn*. Ngoài ra còn có các đối tượng khác, như:

- *DataRelation*: biểu diễn mối liên kết giữa các *DataTable*.
- *DataView*: cung cấp các khung nhìn (View) logic cho *DataTable*.
- *DataSet*: tập hợp nhiều *DataTable*.

Các ví dụ về Data Classes:

10.3.1 DataTable

Ví dụ 1: Tạo đối tượng *DataTable* và hiển thị kết quả trong *GridView*

+ Khai báo *DataTable* như một field của form

```
public partial class WebForms_Test : System.Web.UI.Page
{
    DataTable duLieu = null;
    ...
}
```

+ Định nghĩa bảng, tạo các cột, xác định khóa chính

```
private void DinhNghiaCauTrucDataTable()
{
    this.duLieu = new DataTable();
    this.duLieu.Columns.Add("MaHH", typeof(string)).DefaultValue = string.Empty;
    this.duLieu.Columns.Add("TenHH", typeof(string)).DefaultValue = string.Empty;
    this.duLieu.Columns.Add("DonGia", typeof(double)).DefaultValue = 0;
    this.duLieu.Columns.Add("SoLuong", typeof(int)).DefaultValue = 0;
    this.duLieu.Columns.Add("GiamGia", typeof(double)).DefaultValue = 0;
    this.duLieu.Columns.Add("ThanhTien", typeof(double), "DonGia*SoLuong*(1-GiamGia)").DefaultValue = 0;
    this.duLieu.PrimaryKey = new DataColumn[] { this.duLieu.Columns["MaHH"] };
}
```

+ Hàm bổ sung dữ liệu vào *DataTable* *duLieu*

```
private void TaoDuLieu()
{
    DataRow row1 = this.duLieu.NewRow();
    row1["MaHH"] = "HH1";
    row1["TenHH"] = "Tivi";
    row1["DonGia"] = 123;
    row1["SoLuong"] = 10;
    row1["GiamGia"] = 0;
    this.duLieu.Rows.Add(row1);
    DataRow row2 = this.duLieu.NewRow();
    row2["MaHH"] = "HH2";
    row2["TenHH"] = "Laptop";
```

```

row2[ "DonGia" ] = 213;
row2[ "SoLuong" ] = 10;
row2[ "GiamGia" ] = 0.1;
this.duLieu.Rows.Add(row2);
DataRow row3 = this.duLieu.NewRow();
row3[ "MaHH" ] = "HH3";
row3[ "TenHH" ] = "Iphone";
row3[ "DonGia" ] = 312;
row3[ "SoLuong" ] = 30;
row3[ "GiamGia" ] = 0.5;
this.duLieu.Rows.Add(row3);
}

```

+ Load dữ liệu từ DataTable -> GridView:

```

private void Load2GridView()
{
    this.gvData.DataSource = this.duLieu;
    this.gvData.DataBind();
}

```

+ Sự kiện Page_Load:

```

protected void Page_Load(object sender, EventArgs e)
{
    if (!this.IsPostBack)
    {
        this.DinhNghiaCauTrucDataTable();
        this.TaoDuLieu();
        this.Load2GridView();
    }
}

```

+ Kết quả:

SỬ DỤNG DATATABLE ĐƠN GIẢN:

MaHH	TenHH	DonGia	SoLuong	GiamGia	ThanhTien
HH1	Tivi	123	10	0	1230
HH2	Laptop	213	20	0.1	3834
HH3	Iphone	312	30	0.5	4680

+ Nhờ việc xác định khóa chính trong DataTable nên ta có thể dễ dàng thực hiện các thao tác tìm kiếm dựa trên khóa này. Hàm sau đây sẽ tìm mặt hàng có khóa HH2 và in kết quả ra trình duyệt.

```

private void TimKiem()
{
    DataRow result = this.duLieu.Rows.Find("HH2");
    if (result != null)
        Response.Write(result[ "MaHH" ].ToString() + ", " + result[ "TenHH" ].ToString());
}

```

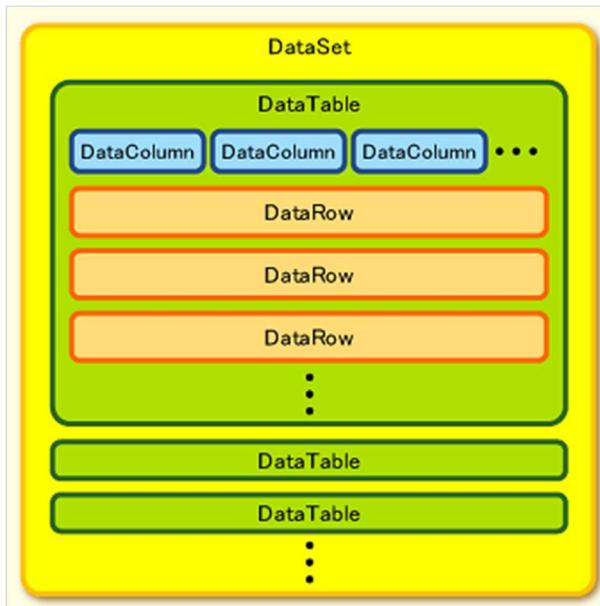
Ví dụ 2: Sử dụng *DataTable* chứa dữ liệu đọc từ bảng trong CSDL. Có nhiều cách để đưa dữ liệu từ CSDL vào *DataTable*. Trong ví dụ này ta sử dụng phương thức *Fill* của đối tượng *SqlDataAdapter*, như sau:

```

private void LoadFromCSDL2DataTable()
{
    string strConnection = "Data Source=.;Initial Catalog=estoreDemo;Integrated
    Security=SSPI;";
    string query = "select * from HangHoa";
    using (SqlConnection conn = new SqlConnection(strConnection))
    {
        SqlDataAdapter adapter = new SqlDataAdapter(query, conn);
        DataTable table = new DataTable();
        adapter.Fill(table);
        foreach (DataRow row in table.Rows)
            Response.Write(row["TenHH"] + "<br/>");
    }
}

```

10.3.2 DataSet



DataSet chứa các *DataTable* cũng như các mối quan hệ giữa chúng. Các mối quan hệ này được thể hiện thông qua đối tượng *DataRelation*. Ví dụ sau đây sẽ load dữ liệu từ 2 bảng *HangHoa*, *Loai* trong CSDL *estoreDemo* vào đối tượng *DataSet*:

```

private DataSet Load2DataSet()
{
    string strConnection = "Data Source=.;Initial Catalog=estoreDemo;Integrated
    Security=SSPI;";
    string query = "select * from HangHoa; select * from Loai";
    using (SqlConnection conn = new SqlConnection(strConnection))
    {
        SqlDataAdapter adapter = new SqlDataAdapter(query, conn);
        DataSet ds = new DataSet();
        adapter.Fill(ds);
        return ds;
    }
}

```

```

private void DataSetDemo()
{
    DataSet ds = this.Load2DataSet();
    Response.Write("Số DataTable trong DataSet = " + ds.Tables.Count.ToString());
    Response.Write("<br/>Dữ liệu trong DataTable thứ nhất:<br/>");
    //truy xuất đến các DataTable:
    foreach(DataRow row in ds.Tables[0].Rows)
        Response.Write(row["TenHH"] + "<br/>");
    Response.Write("Dữ liệu trong DataTable thứ hai:<br/>");
    foreach (DataRow row in ds.Tables[1].Rows)
        Response.Write(row["TenLoai"] + "<br/>");
    //xóa DataTable trong DataSet:
    ds.Tables.RemoveAt(1);
    Response.Write("Số DataTable trong DataSet = " + ds.Tables.Count.ToString());
    //xóa tất cả các DataTable trong DataSet:
    ds.Tables.Clear();
    Response.Write("Số DataTable trong DataSet = " + ds.Tables.Count.ToString());
}

```

10.3.3 DataView

Chúng ta có thể tùy biến *DataView* để lấy một tập con các records trong *DataTable*. Khả năng này cho phép có thể tạo ra nhiều controls cùng liên kết đến một *DataTable* nhưng hiển thị những phần dữ liệu khác nhau.

Do vậy sử dụng *DataView* chúng ta có thể thực hiện các thao tác như sắp xếp, lọc, .. trên dữ liệu. (Bản thân *DataTable* cũng chứa một thuộc tính có tên là *DataView*, thuộc tính này trả về *DataView* mặc định của *DataTable* đó).

Ví dụ sau hiển thị dữ liệu được lọc những mặt hàng có mã nhỏ hơn **1011** và hiển thị trên *GridView*:

```

protected void btnFilter_Click(object sender, EventArgs e)
{
    DataSet ds = this.CreateDataSet();
    DataView dv = new DataView(ds.Tables[0]);
    //loc:
    dv.RowFilter = "MaHH < 1011";
    gvHangHoa.DataSource = dv;
    gvHangHoa.DataBind();
}

```

10.3.4 DataRelation

Đối tượng *DataRelation* được sử dụng để xử lý mối quan hệ giữa các *DataTable* trong *DataSet*.

Trong ví dụ sau, chúng tôi xây dựng một đối tượng *DataSet* chứa các *DataTable* lưu giữ thông tin về sản phẩm (*Product*) và Loại sản phẩm (*Category*). Đồng thời tạo *DataRelation* giữa 2 *DataTable* này.

```

public class MyDataSet : DataSet
{
    public DataTable Category { private set; get; }
    public DataTable Product { private set; get; }
    public DataRelation Relation { private set; get; }
    public MyDataSet()
    {
        this.DataSetName = "DataSetDemo";
        //dinh nghia bang Category:
        this.Category = new DataTable("tblCategory");
        this.Category.Columns.Add("ID", typeof(string));

```

```

        this.Category.Columns.Add("Name", typeof(string));
        this.Category.PrimaryKey = new DataColumn[] {
            this.Category.Columns["ID"] };
        //dinh nghia bang Product:
        this.Product = new DataTable("tblProduct");
        this.Product.Columns.Add("ID", typeof(string));
        this.Product.Columns.Add("Name", typeof(string));
        this.Product.Columns.Add("Price", typeof(double));
        this.Product.Columns.Add("Category", typeof(string));
        this.Product.PrimaryKey = new DataColumn[] { this.Product.Columns["ID"] }
    };
    //chen du lieu:
    this.Category.Rows.Add("C1", "Laptop");
    this.Category.Rows.Add("C2", "Tivi");
    this.Category.Rows.Add("C3", "IPhone");
    this.Product.Rows.Add("P1", "Dell Vostro 1088", 15, "C1");
    this.Product.Rows.Add("P2", "Sony Vaio", 20, "C1");
    this.Product.Rows.Add("P3", "Macbook", 25, "C1");
    this.Product.Rows.Add("P4", "LG", 15, "C2");
    this.Product.Rows.Add("P5", "Sony", 20, "C2");
    this.Product.Rows.Add("P6", "Samsung", 30, "C2");
    //add cac bang -> DataSet:
    this.Tables.Add(this.Category);
    this.Tables.Add(this.Product);
    //tạo relation:
    this.Relation = new DataRelation("relation_Category_Product",
        this.Category.Columns["ID"], this.Product.Columns["Category"]);
    this.Relations.Add(this.Relation);
}
}

```

Nhờ đối tượng *DataRelation*, chúng ta có thể dễ dàng truy xuất được các *Product* theo *Category* như sau:

```

DataRow row = ds.Category.Rows.Find("C1"); //(1)
foreach (DataRow r in row.GetChildRows(ds.Relation)) //(2)
    Response.Write(r["Name"].ToString());

```

Câu lệnh (1) giúp chúng ta tìm được *Category* có *ID* bằng "C1". Sau đó, dựa vào *DataRelation* mà ta có thể truy xuất đến tất cả các dòng trong *Product* có *Category* bằng "C1" như ở câu lệnh (2) bằng phương thức **GetChildRows**

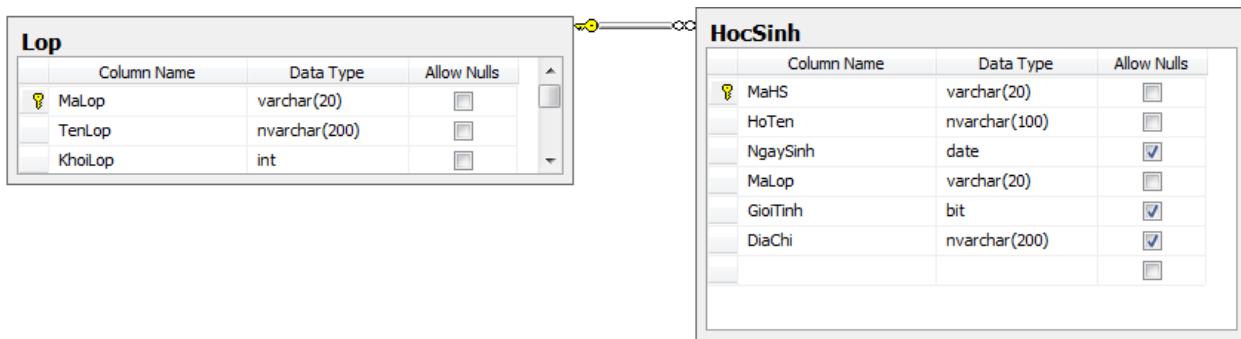
Chú ý: Để nắm chi tiết về các thao tác với các *DataClasses* bạn nên tham khảo các bài labs.

10.4 Kết nối Và Thao tác Dữ liệu

Trong phần này chúng tôi tập trung trình bày các thao tác kết nối, truy xuất dữ liệu với hệ quản trị **SQL Server 2008**. Đồng thời hướng dẫn bạn cách xây dựng một lớp truy xuất dữ liệu để thuận tiện trong việc phát triển dự án.

Giả sử chúng ta có các thông tin về CSDL trên SQL Server 2008 như sau:

- Tên CSDL: **HSDB**
- CSDL này chứa một số bảng mà chúng ta sẽ trực tiếp minh họa:



Chúng ta sẽ tạo một lớp **static** phục vụ việc kết nối và truy xuất dữ liệu:

```
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Data;
using System.Data.SqlClient;

static public class DataAL
{
    //các minh họa sẽ được thể hiện ở đây.
}

10.4.1      Tạo kết nối
```

Các bước tạo kết nối như sau:

- Khai báo chuỗi kết nối.
- Tạo đối tượng kết nối **SqlConnection** gắn với chuỗi kết nối.
- Mở kết nối.

Sau đây là mã minh họa:

```
SqlConnection conn = new SqlConnection(connectionString);
try
{
    if (conn.State == ConnectionState.Closed)
    {
        conn.Open();
    }
    //các xử lý nếu kết nối thành công:
}
catch
{
    //các xử lý nếu kết nối thất bại
}
finally
{
    conn.Close(); //luôn đóng kết nối
}
```

10.4.2 **Truy xuất dữ liệu**

Đọc dữ liệu bằng **SqlDataReader**

```
static public bool ReadDataByDataReader()
{
    string connectionString = "Data Source=.;Initial Catalog=HSDB;Integrated
    Security=SSPI;";
    SqlConnection conn = new SqlConnection(connectionString);
    SqlDataReader reader = null;
```

```

try
{
    if (conn.State == ConnectionState.Closed)
    {
        conn.Open();
    }
    SqlCommand cmd = new SqlCommand();
    cmd.CommandType = CommandType.Text;
    cmd.CommandText = "select HoTen from HocSinh";
    cmd.Connection = conn;
    reader = cmd.ExecuteReader();
    while (reader.Read())
    {
        //hiển thị dữ liệu trên trình duyệt
        HttpContext.Current.Response.Write
        (
            reader.GetString(0).ToString() + "<br/>"
        );
    }
    return true;
}
catch
{
    return false;
}
finally
{
    reader.Close();
    conn.Close();
}
}

```

Đọc dữ liệu bằng DataAdapter

```

static public DataTable ReadDataByDataAdapter()
{
    DataTable dt = new DataTable();
    string connectionString = "Data Source=.;Initial Catalog=HSDB;Integrated
    Security=SSPI;";
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        if (conn.State == ConnectionState.Closed) conn.Open();
        using (SqlCommand cmd = new SqlCommand())
        {
            cmd.CommandType = CommandType.Text;
            cmd.CommandText = "select HoTen from HocSinh";
            cmd.Connection = conn;
            using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))
            {
                adapter.Fill(dt);
            }
        }
    }
    return dt;
}

```

10.4.3 Thao tác dữ liệu

Nói chung để thực hiện các thao tác thêm, cập nhật và xóa dữ liệu trong một bảng chúng ta sử dụng phương thức **ExecuteNonQuery** và thuộc tính **Parameters** của đối tượng **SqlCommand** như các ví dụ dưới đây.

Thêm mới vào bảng Lớp:

```
/// <summary>
/// thêm dữ liệu vào bảng Lop
/// </summary>
/// <param name="maLop"></param>
/// <param name="tenLop"></param>
/// <param name="khoiLop"></param>
/// <returns>Nếu thành công trả về 1, ngược lại trả về 0</returns>
static public int InsertLop(string maLop, string tenLop, int khoiLop)
{
    string connectionString = "Data Source=.;Initial Catalog=HSDB;Integrated
Security=SSPI;";
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        if (conn.State == ConnectionState.Closed) conn.Open();
        using (SqlCommand cmd = new SqlCommand())
        {
            cmd.CommandType = CommandType.Text;
            cmd.CommandText = "insert into Lop (MaLop, TenLop, KhoiLop)
values (@MaLop,@TenLop,@KhoiLop)";
            cmd.Connection = conn;
            cmd.Parameters.AddWithValue("@MaLop", maLop);
            cmd.Parameters.AddWithValue("@TenLop", tenLop);
            cmd.Parameters.AddWithValue("@KhoiLop", khoiLop);
            return cmd.ExecuteNonQuery();
        }
    }
}
```

Ta có thể gọi hàm InsertLop trên như sau:

```
int result = DataAL.InsertLop("L08", "Lớp 8A", 8);
if (result == 1)
    Response.Write("insert thành công");
else
    Response.Write("insert thất bại");
```

Cập nhật một dòng trong bảng Lớp:

```
static public int UpdateLop(string maLop, string tenLop, int khoiLop)
{
    string connectionString = "Data Source=.;Initial Catalog=HSDB;Integrated
Security=SSPI;";
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        if (conn.State == ConnectionState.Closed) conn.Open();
        using (SqlCommand cmd = new SqlCommand())
        {
            cmd.CommandType = CommandType.Text;
            cmd.CommandText = "update Lop set TenLop = @TenLop, KhoiLop =
@KhoiLop where MaLop = @MaLop";
            cmd.Connection = conn;
            cmd.Parameters.AddWithValue("@MaLop", maLop);
```

```

        cmd.Parameters.AddWithValue("@TenLop", tenLop);
        cmd.Parameters.AddWithValue("@KhoiLop", khoiLop);
        return cmd.ExecuteNonQuery();
    }
}
}

```

Sử dụng:

```

int result = DataAL.UpdateLop("L08", "Lớp 7B", 7);
if (result == 1)
    Response.Write("update thành công");
else
    Response.Write("update thất bại");

```

Xóa một dòng trong bảng Lớp:

```

static public int DeleteLop(string maLop)
{
    string connectionString = "Data Source=.;Initial Catalog=HSDB;Integrated
Security=SSPI;";
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        if (conn.State == ConnectionState.Closed) conn.Open();
        using (SqlCommand cmd = new SqlCommand())
        {
            cmd.CommandType = CommandType.Text;
            cmd.CommandText = "delete from Lop where MaLop = @MaLop";
            cmd.Connection = conn;
            cmd.Parameters.AddWithValue("@MaLop", maLop);
            return cmd.ExecuteNonQuery();
        }
    }
}

```

Sử dụng:

```

int result = DataAL.DeleteLop("L08");
if (result == 1)
    Response.Write("delete thành công");
else
    Response.Write("delete thất bại");

```

Các thao tác thống kê:

Việc thống kê dữ liệu trên (các) bảng sẽ sử dụng phương thức **ExecuteScalar** của và thuộc tính **Parameters** của đối tượng **SqlCommand**

Dưới đây là ví dụ đếm số lượng học sinh:

```

static public int CountHocSinh(string maLop)
{
    string connectionString = "Data Source=.;Initial Catalog=HSDB;Integrated
Security=SSPI;";
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        if (conn.State == ConnectionState.Closed) conn.Open();
        using (SqlCommand cmd = new SqlCommand())
        {

```

```

        cmd.CommandType = CommandType.Text;
        cmd.CommandText = "select count(*) from HocSinh where MaLop =
@MaLop";
        cmd.Connection = conn;
        cmd.Parameters.AddWithValue("@MaLop", maLop);
        return (int)cmd.ExecuteScalar();
    }
}
}
    
```

Sử dụng:

```

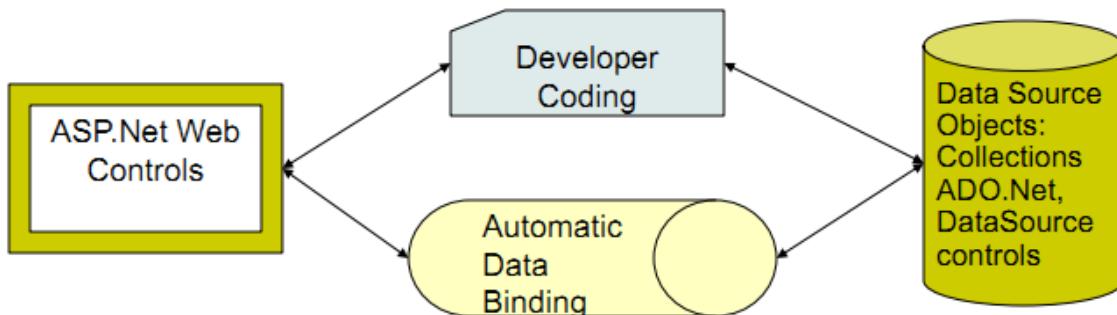
int result = DataAL.CountHocSinh("L000003");
Response.Write("sĩ số = " + result.ToString());
    
```

Nhận xét: Trong các ví dụ minh họa ở trên, vì mục đích giải thích từng bước các thao tác nên các đối tượng, code... được sử dụng lại nhiều lần trong các hàm. Tuy nhiên trong thực tế, chúng ta sẽ xây dựng lớp **DataAL** sao cho hiệu quả hơn. Để nắm rõ vấn đề này bạn nên tìm hiểu trong các bài labs.

10.5 Ràng buộc (Binding) Dữ liệu với Data Controls

Data binding là tạo ra sự liên kết giữa các Web UI controls của .Net với các Data sources (nguồn dữ liệu).

Có 2 cách để đưa dữ liệu vào các controls, thể hiện trong hình sau:



Các data controls được chia thành 2 nhóm: các controls đơn giản (simple) và các controls tích hợp (composite)

Simple Data Controls:

- DropDownList
- CheckBoxList, RadioButtonList
- ListBox, BulletedList

Composite Data Controls:

- ListView
- DataList
- GridView
- Repeater, DetailsView, FormView

Nguồn dữ liệu (Data Sources):

Nói chung, bất kỳ lớp nào hiện thực (implements) giao diện *IList* đều có thể sử dụng làm data source. Các lớp hỗ trợ giao diện (interface) *IList* trong .NET gồm:

Nhóm Collections:

- Array, ArrayList, List<>, HashTable, Dictionary, ..

Nhóm ADO.Net:

- DataReader, DataSet, DataTable, DataView, DataColumn
- DataSource Controls
- Linq, Entity, Object, SQL Server, XML, ...

Quá trình kết nối dữ liệu:

Bước	List Controls + BindingCode	Composite Controls + Binding Code	List Controls + Data Source Controls	Composite Controls + Data Source Controls
1. Chuẩn bị một nguồn dữ liệu	Lập trình để lấy đối tượng data source			Thiết kế một data source control
2. Định nghĩa Web control	Xác định các thuộc tính cho control	Xác định thuộc tính, <i>template field</i> và <i>data binding field</i>	Xác định các thuộc tính cho control	Xác định thuộc tính, <i>template field</i> và <i>data binding field</i>
3. Liên kết data source với web control	Lập trình để xác lập thuộc tính DataSource của control, DataMember, DataTextField, DataValueField			Khai báo để xác định thuộc tính DataSource
4. Kết nối (binding)	Gọi phương thức DataBind()			

Các ví dụ:

Trong phần này chúng tôi xin trình bày việc *binding* với các đối tượng thuộc nhóm *Collection*. Còn việc binding với các đối tượng trong nhóm *ADO.NET* mời bạn tham khảo ở phần sau.

Ví dụ 1: Hàm sau đây sẽ kết nối dữ liệu từ một *list* vào các *list controls* (*DropDownList* và *RadioButtonList*)

```
private void LoadData()
{
    //tạo list:
    List<String> list = new List<string>();
    list.Add("C#");
    list.Add("ASP.Net");
    list.Add("PHP");
    //binding list -> dropdownlist:
    this.ddlDemo.DataSource = list;
    this.ddlDemo.DataBind();
    //binding list -> radiobuttonlist
    this.rblDemo.DataSource = list;
    this.rblDemo.DataBind();
}
```

Kết quả:

DropDownList: C# ▾

C#

RadioButtonList: ASP.Net PHP

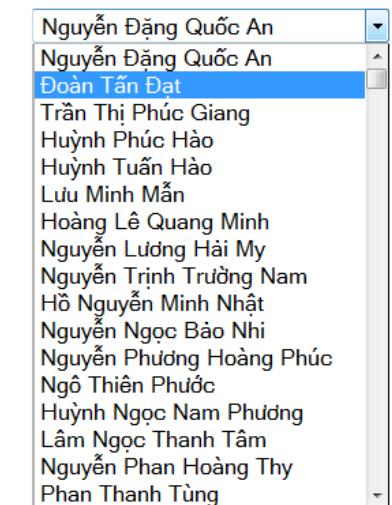
ASP.Net

PHP

Ví dụ 2: Trong ví dụ này chúng ta sử dụng `SqlDataReader` để đọc dữ liệu, sau đó binding vào `DropDownList`. Bạn hãy chú ý đến việc gán giá trị cho các thuộc tính **DataValueField**, **TextField** của `DropDownList`

```
private void LoadDataByDataReader()
{
    string connectionString = "Data Source=.;Initial Catalog=HSDB;Integrated
    Security=SSPI;";
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        if (conn.State == ConnectionState.Closed) conn.Open();
        using (SqlCommand cmd = new SqlCommand())
        {
            cmd.CommandType = CommandType.Text;
            cmd.CommandText = "select MaHS, HoTen from HocSinh";
            cmd.Connection = conn;
            SqlDataReader source = cmd.ExecuteReader();
            this.ddlDemo.DataSource = source;
            this.ddlDemo.DataValueField = "MaHS";
            this.ddlDemo.TextField = "HoTen";
            this.ddlDemo.DataBind();
        }
    }
}
```

Kết quả:



Ví dụ 3: Binding dữ liệu từ đối tượng `DataSet` vào `ListBox`

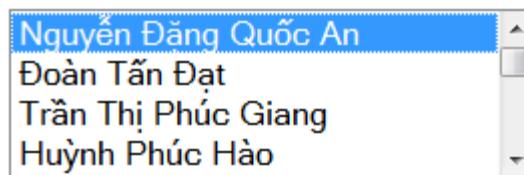
```
private void DataSet2ListBox()
{
    DataSet ds = new DataSet();
    string connectionString = "Data Source=.;Initial Catalog=HSDB;Integrated
    Security=SSPI;";
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        if (conn.State == ConnectionState.Closed) conn.Open();
        using (SqlCommand cmd = new SqlCommand())
        {
            cmd.CommandType = CommandType.Text;
            cmd.CommandText = "select MaHS, HoTen from HocSinh";
            cmd.Connection = conn;
            using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))
```

```

        {
            adapter.Fill(ds, "HocSinh");
        }
        //binding to ListBox:
        lstDemo.DataSource = ds;
        lstDemo.DataMember = "HocSinh"; //tên bảng
        lstDemo.DataTextField = "HoTen";
        lstDemo.DataValueField = "MaHS";
        lstDemo.DataBind();
    }
}

```

Kết quả:

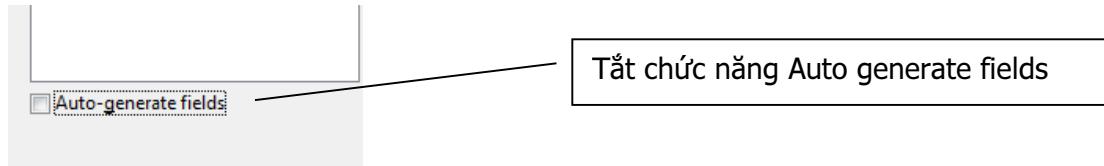


Chú ý: xác định tên của *DataTable* trong *DataSet* muốn binding đến *ListBox* trong câu lệnh sau:

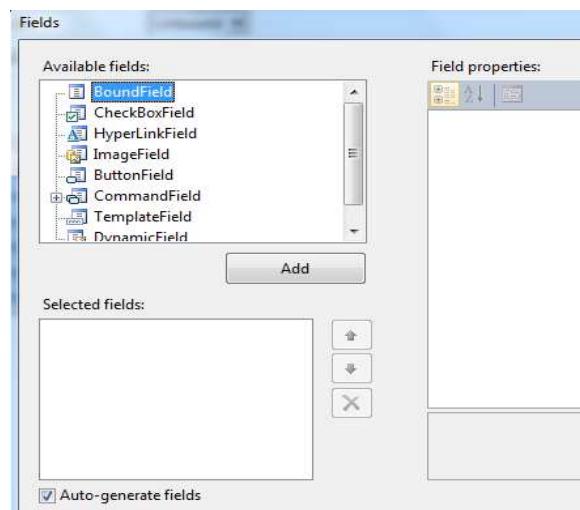
`lstDemo.DataMember = "HocSinh"; //tên bảng`

Templates và Data Binding Fields:

Các *composite controls* sử dụng *fields* để xác định *layout* cho một cột dữ liệu được *binding*. Nếu bạn không tự tạo các *fields* để binding dữ liệu thì, *Net* sẽ tự động sinh ra các *field* tương ứng với dữ liệu mà nó nhận được. Để tắt chức năng này, bạn bỏ hộp chọn:

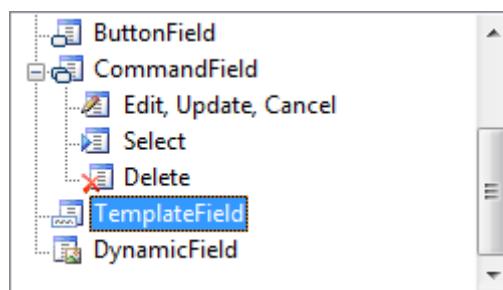
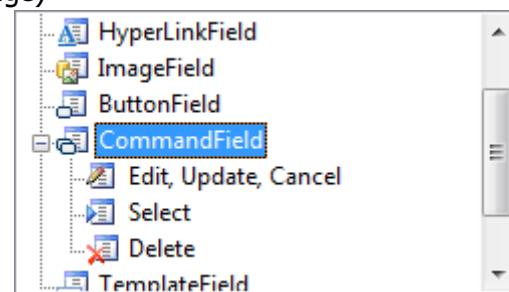


Như minh họa ở hình dưới là các *fields* của *GridView*:

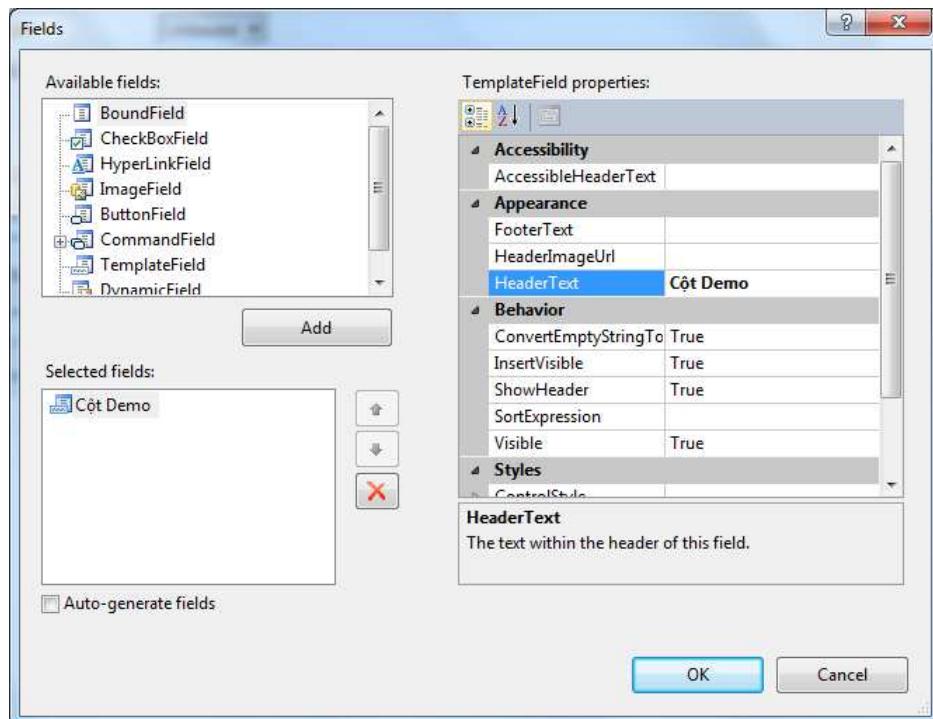


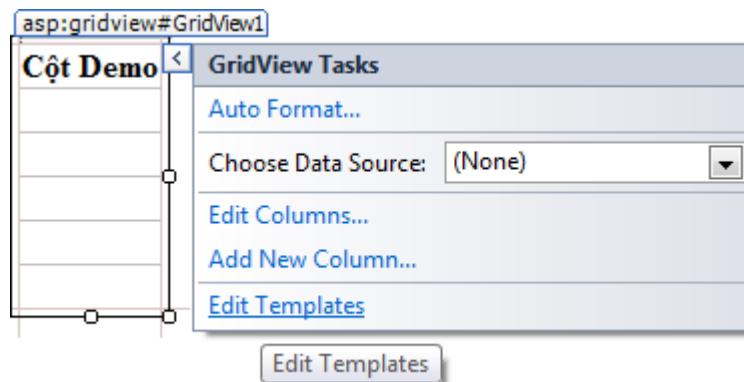
Trong đó:

- *BoundField*: là loại field đơn giản nhất (mặc định), hiển thị dữ liệu dưới dạng text.
- *ImageField*: hiển thị dữ liệu dưới dạng ảnh (image)
- *ButtonField*: hiển thị các link button (cho phép người dùng tương tác)
- *CommandField*: thể hiện các thao tác (lệnh)
- *TemplateField*:

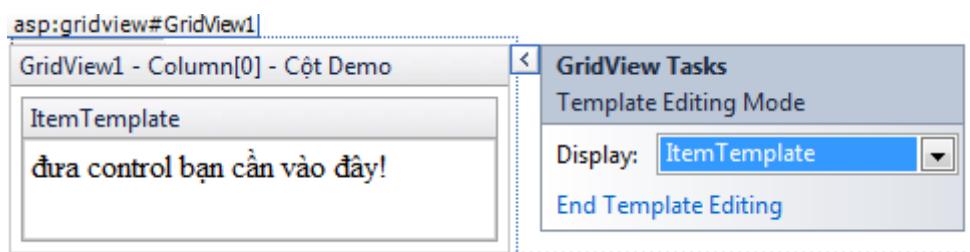


Đây là *field* linh hoạt nhất, nó có nghĩa là bạn chỉ xác định một “mẫu” – *template* chung chung, sau đó bạn phải “đặt” một *control*/khác vào *field* này để kết nối dữ liệu, như minh họa ở dưới:





Chọn *Edit Templates* để edit field này:

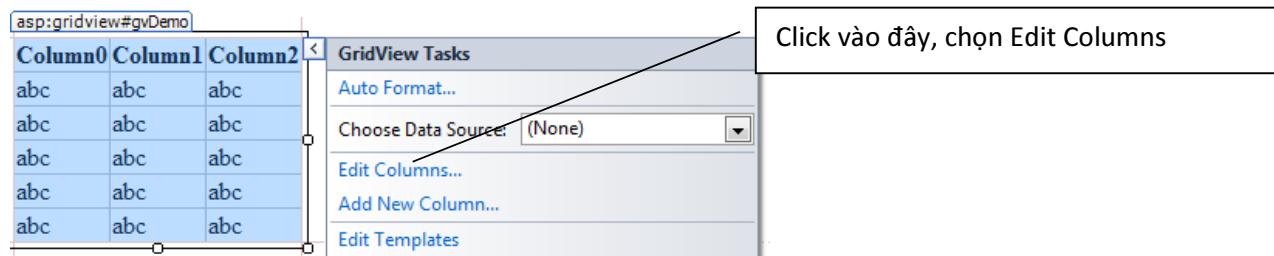


Để binding cho các template field, ta sử dụng:

- `<%# Eval() %>`: chỉ dùng cho việc đọc dữ liệu.
- `<%# Bind() %>`: cả đọc và ghi dữ liệu.

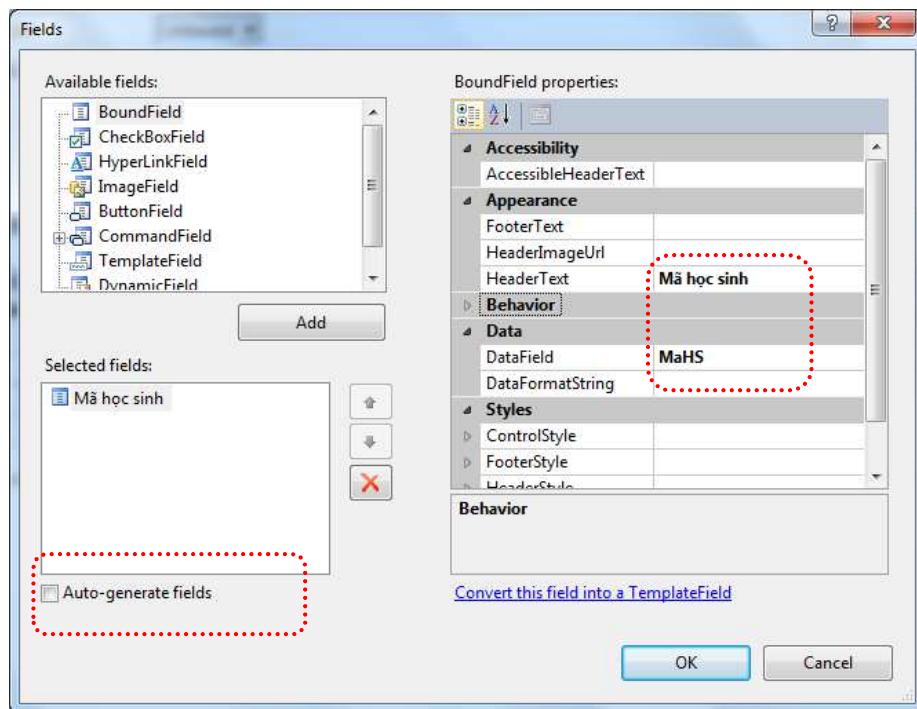
Ví dụ sau đây trình bày các thao tác trên fields của *GridView* để binding dữ liệu từ bảng học sinh:

Bước 1: Tạo GridView



Bước 2: Tạo các fields

Thuộc tính	Mô tả
DataField	Tên của cột trong bảng kết nối
HeaderText	Tên hiển thị trong cột của GridView
DataFormatString	Định dạng dữ liệu trong cột



Làm tương tự cho các cột còn lại...

Bước 3: Viết mã để kết nối dữ liệu

```
private void DataSet2GridView()
{
    DataSet ds = new DataSet();
    string connectionString = "Data Source=.;Initial Catalog=HSDB;Integrated Security=SSPI;";
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        if (conn.State == ConnectionState.Closed) conn.Open();
        using (SqlCommand cmd = new SqlCommand())
        {
            cmd.CommandType = CommandType.Text;
            cmd.CommandText = "select * from HocSinh";
            cmd.Connection = conn;
            using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))
            {
                adapter.Fill(ds, "HocSinh");
            }
            //binding to GridView:
            gvDemo.DataSource = ds;
            gvDemo.DataMember = "HocSinh";
            gvDemo.DataBind();
        }
    }
}
```

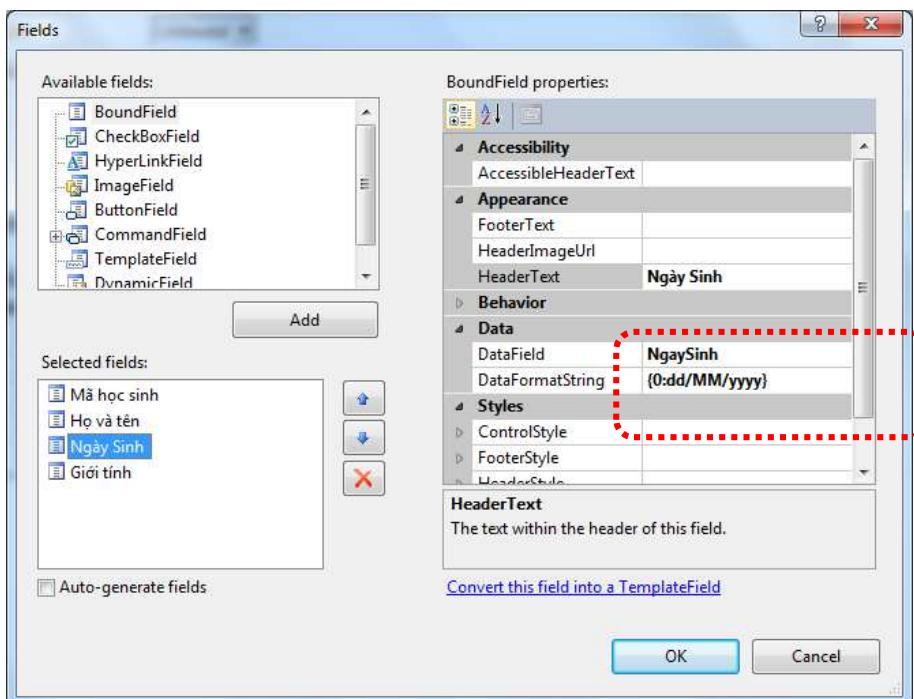
Bước 4: Chạy demo, bạn thu được kết quả sau:

BINDING FIELDS IN GRIDVIEW:

Mã học sinh	Họ và tên	Ngày Sinh	Giới tính
HS000004	Nguyễn Đặng Quốc An	8/10/2006 12:00:00 AM	True
HS000005	Đoàn Tân Đạt	8/10/2006 12:00:00 AM	True
HS000006	Trần Thị Phúc Giang	8/10/2006 12:00:00 AM	True
HS000007	Huỳnh Phúc Hào	8/10/2006 12:00:00 AM	True
HS000008	Huỳnh Tuấn Hào	8/10/2006 12:00:00 AM	True
HS000009	Lưu Minh Mẫn	8/10/2006 12:00:00 AM	True
HS000010	Hoàng Lê Quang Minh	8/10/2006 12:00:00 AM	True
HS000011	Nguyễn Lương Hải My	8/10/2006 12:00:00 AM	True
HS000012	Nguyễn Trịnh Trường Nam	8/10/2006 12:00:00 AM	True
HS000013	Hồ Nguyễn Minh Nhật	8/10/2006 12:00:00 AM	True
HS000014	Nguyễn Ngọc Bảo Nhi	8/10/2006 12:00:00 AM	True

Ở đây có 2 cột hiển thị dữ liệu chưa như mong muốn, đó là cột "Ngày sinh" và cột "Giới tính". Đây chính là lúc chúng ta phải sử dụng đến thuộc tính *DataFormatString* cũng như *TemplateField*.

Bước 5: Đặt *DataFormatString* cho cột "Ngày sinh"



Bước 6: Xử lý cột "Giới tính" bằng TemplateField

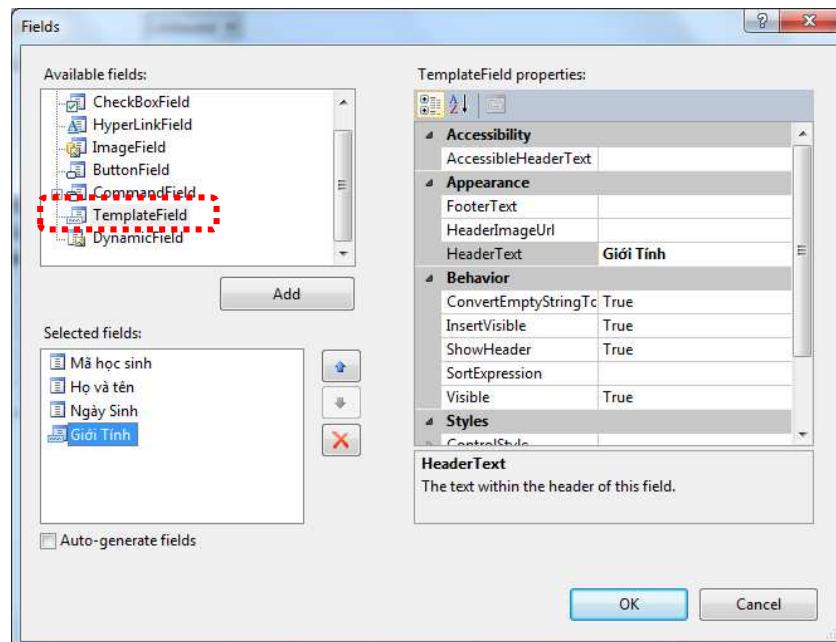
Trước hết trong CodeBehind bạn viết một hàm **public** để định giá trị cho cột này.

```
public string DisplayGioiTinh(object obj)
{
    bool b = bool.Parse(obj.ToString());
```

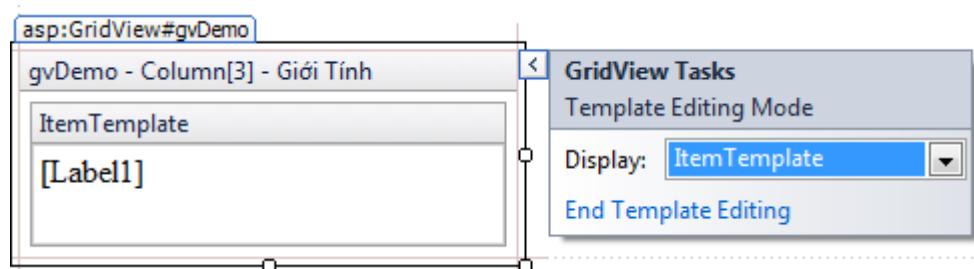
```

if (b) return "Nam";
else return "Nữ";
}
    
```

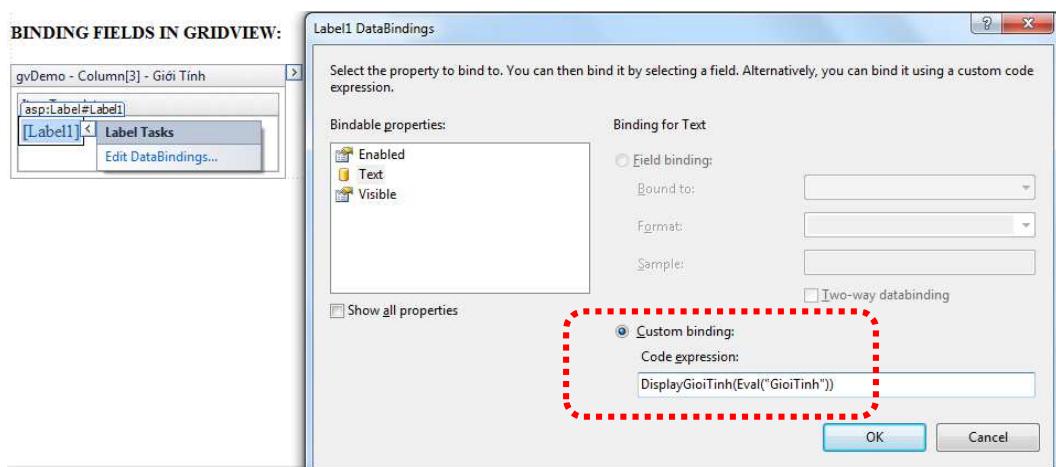
Kế tiếp, tạo lại cột “Giới tính” như sau (chọn *Template Field*)



Edit Template và kéo thả Label vào khung ItemTemplate:



Chọn [Label1] để Edit DataBindings...



Sử dụng hàm DisplayGioiTinh trong khung *Code expression*:

DisplayGioiTinh(Eval("GioiTinh"))

Chạy lại trên trình duyệt, bạn có kết quả:

BINDING FIELDS IN GRIDVIEW:

Mã học sinh	Họ và tên	Ngày Sinh	Giới Tính
HS000004	Nguyễn Đặng Quốc An	10/08/2006	Nam
HS000005	Đoàn Tấn Đạt	10/08/2006	Nam
HS000006	Trần Thị Phúc Giang	10/08/2006	Nam
HS000007	Huỳnh Phúc Hào	10/08/2006	Nam
HS000008	Huỳnh Tuấn Hào	10/08/2006	Nam
HS000009	Lưu Minh Mẫn	10/08/2006	Nam
HS000010	Hoàng Lê Quang Minh	10/08/2006	Nam
HS000011	Nguyễn Lương Hải My	10/08/2006	Nam
HS000012	Nguyễn Trịnh Trường Nam	10/08/2006	Nam
HS000013	Hồ Nguyễn Minh Nhật	10/08/2006	Nam

10.6 ADO.Net & SQL DataSource Control

Các Data source control cung cấp cho bạn khả năng truy xuất dữ liệu mà không cần phải viết mã nguồn. Các Data source control sẽ được thiết kế trực tiếp trên trang .aspx.

Data source control có thể:

- Kết nối để đọc dữ liệu
- Cho phép thực hiện các thao tác cập nhật, thêm, xóa trên dữ liệu.

Có các loại data source control cơ bản như: *ObjectDataSource* và *SqlDataSource* như mô tả trong bảng dưới đây:

Data source cơ bản	Mô tả
ObjectDataSource	Thường sử dụng với các đối tượng thuộc middle-tier (tầng trung gian) để quản lý dữ liệu. Cung cấp các cơ chế sắp xếp và phân trang mà các data source controls khác không hỗ trợ.
SqlDataSource	Cho phép bạn làm việc với các CSDL như Microsoft SQL Server, OLE DB, ODBC, hay Oracle. Control này cũng cung cấp các cơ chế sắp xếp, lọc, và phân trang khi làm việc với đối tượng DataSet.

Tuy nhiên trong phạm vi giáo trình này, chỉ xin trình bày các thao tác với *SqlDataSource*.

Ví dụ: trong ví dụ này chúng ta sử dụng *DropDownList*, *GridView*, *Repeater* kết nối dữ liệu với *SqlDataSource*.

Bước 1: Thiết kế giao diện như sau:

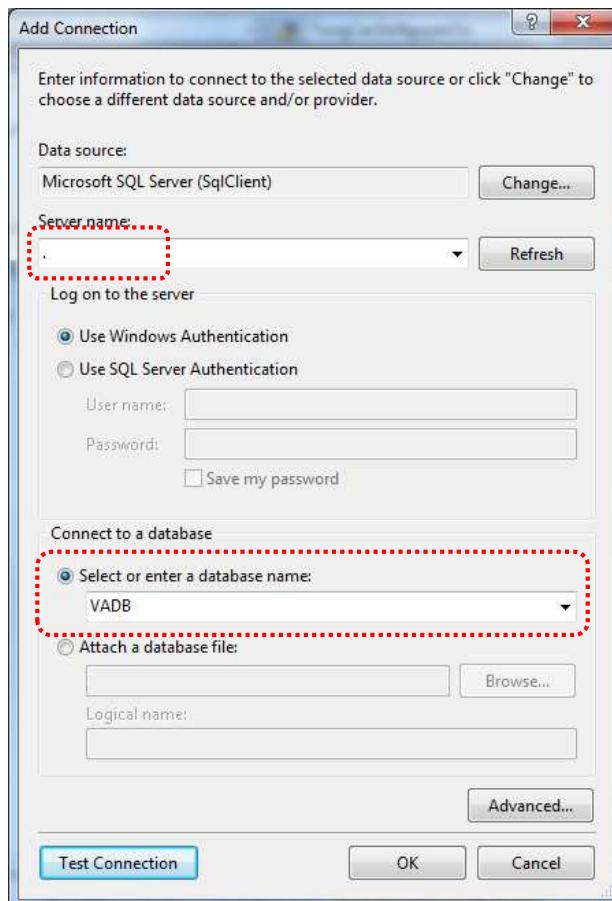
- Tạo *DropDownList*



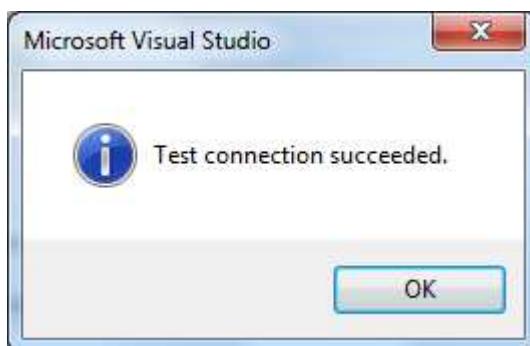
- Tạo *SqlDataSource* lấy danh sách lớp để binding vào *DropDownList*:



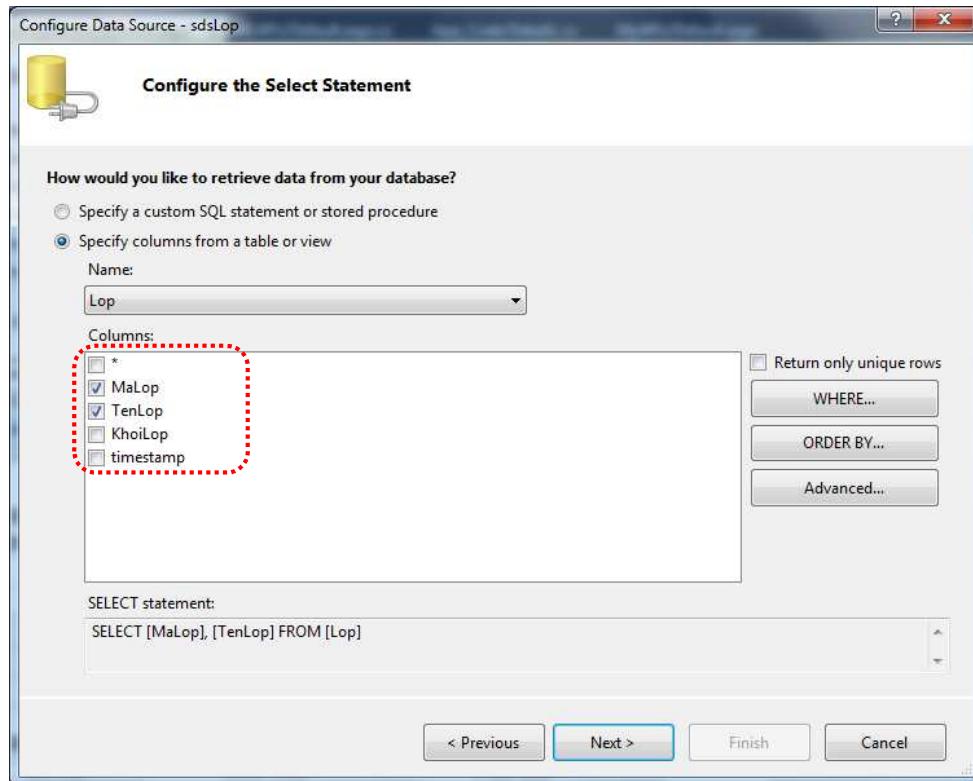
- Cấu hình DataSource cho *SqlDataSource*



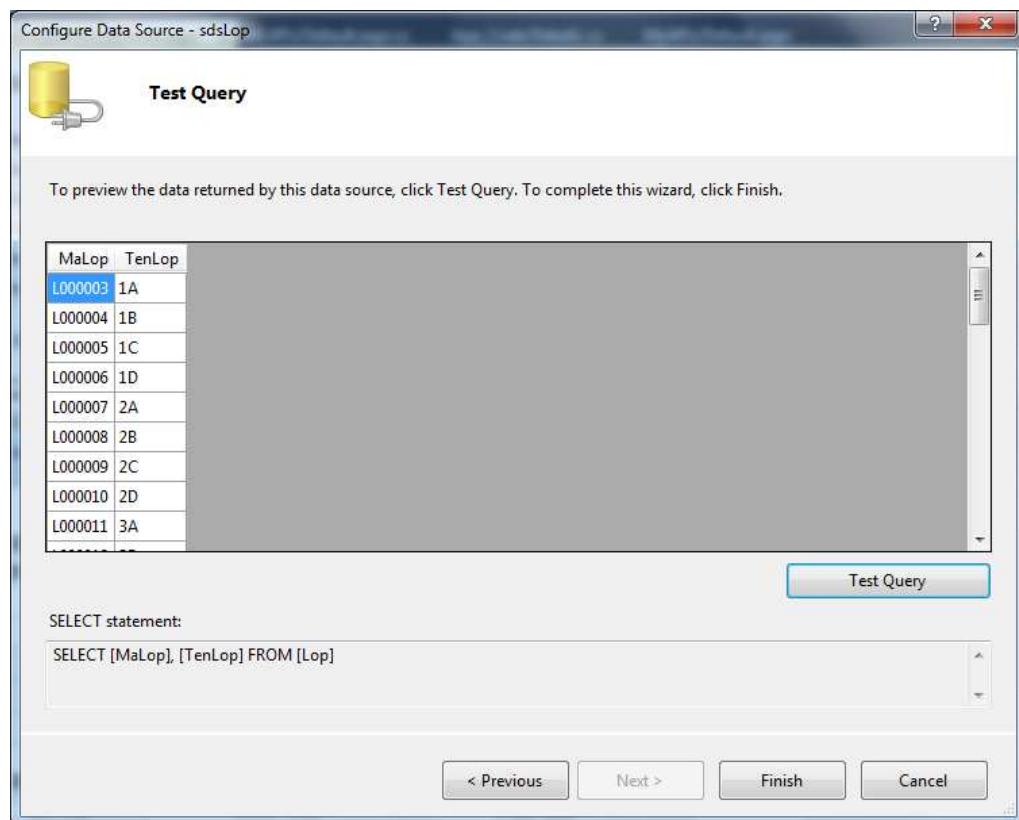
- Kiểm tra kết nối:



- Chọn bảng chứa dữ liệu cần load:

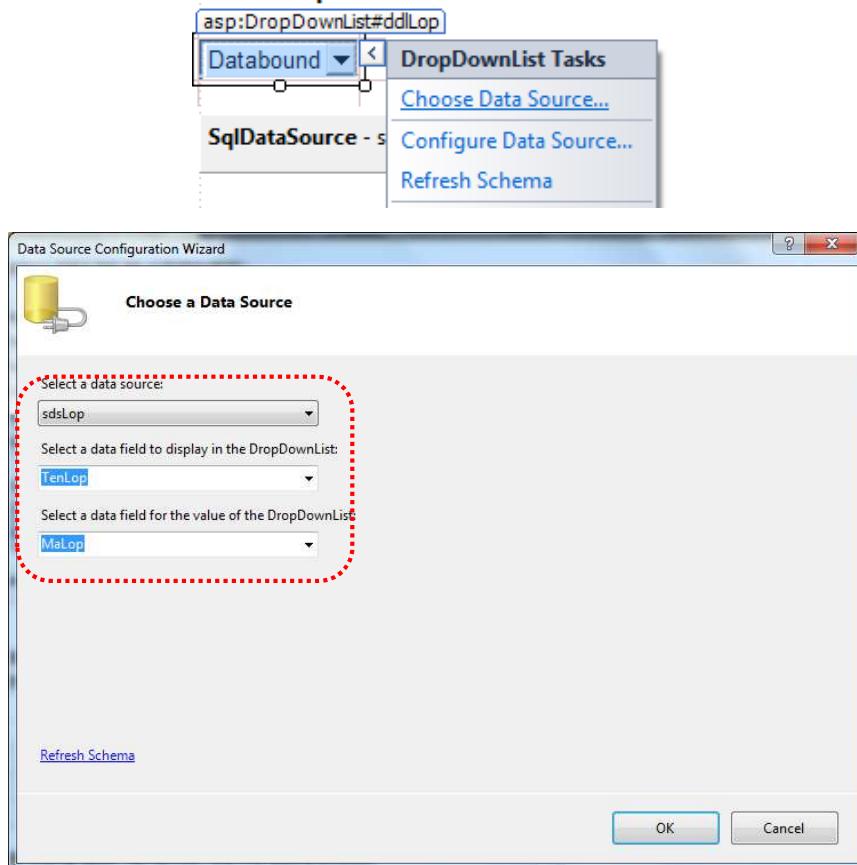


- Kiểm tra truy vấn (Test Query):



Bước 2: Kết nối dữ liệu từ SqlDataSource vào DropDownList:

HÃY CHỌN LỚP:

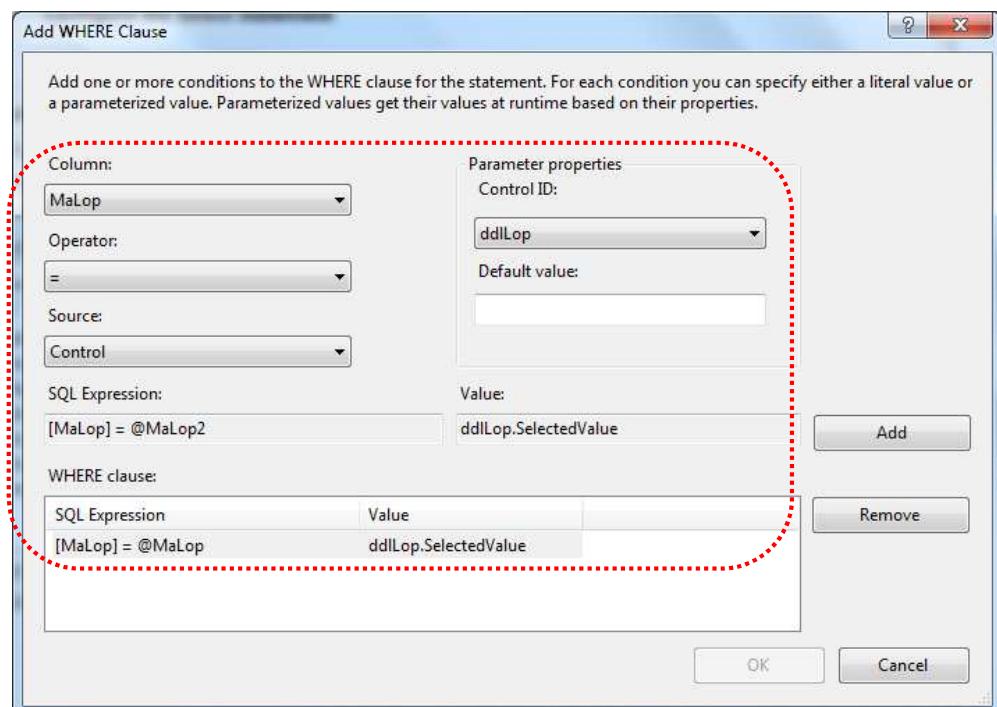
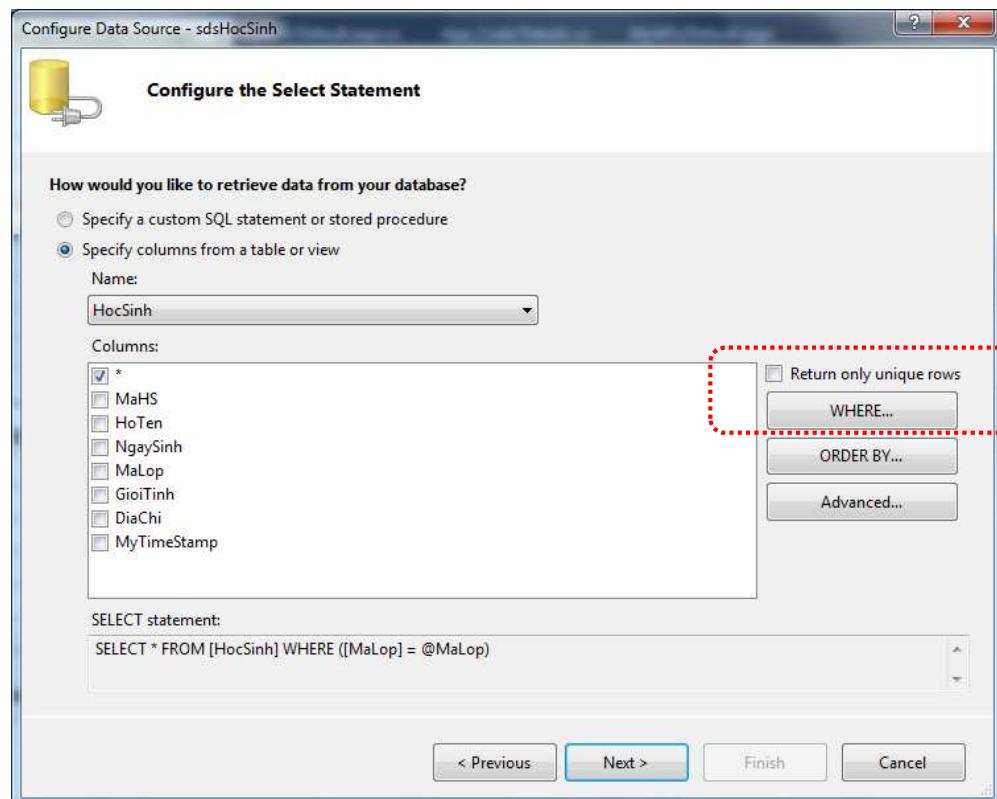


Bước 3:

- Tạo GridView

Column0	Column1	Column2
abc	abc	abc

- Tạo SqlDataSource *sdsHocSinh* để kết nối dữ liệu trong bảng Học sinh vào GridView
 (Tương tự bước 1)



DANH SÁCH HỌC SINH (SỬ DỤNG GRIVIEW):

MaHS	HoTen	NgaySinh	MaLop	GioiTinh	DiaChi
abc	abc	8/25/2012 12:00:00 AM	abc	<input type="checkbox"/>	abc
abc	abc	8/25/2012 12:00:00 AM	abc	<input checked="" type="checkbox"/>	abc
abc	abc	8/25/2012 12:00:00 AM	abc	<input type="checkbox"/>	abc
abc	abc	8/25/2012 12:00:00 AM	abc	<input checked="" type="checkbox"/>	abc
abc	abc	8/25/2012 12:00:00 AM	abc	<input type="checkbox"/>	abc

Bước 4: Chạy thử (chú ý đặt thuộc tính *AutoPostBack* cho *DropDownList* = True)

11C ▾

MaHS	HoTen	NgaySinh	MaLop	GioiTinh	DiaChi
HS000744	Đặng Trần Bảo Ân	8/13/2006 12:00:00 AM	L000040	<input checked="" type="checkbox"/>	
HS000745	Nguyễn Tiên Đạt	8/13/2006 12:00:00 AM	L000040	<input checked="" type="checkbox"/>	
HS000746	Nguyễn Nhựt Hồ	8/13/2006 12:00:00 AM	L000040	<input checked="" type="checkbox"/>	
HS000747	Nguyễn Minh Hưng	8/13/2006 12:00:00 AM	L000040	<input checked="" type="checkbox"/>	
HS000748	Trịnh Quang Khải	8/13/2006 12:00:00 AM	L000040	<input checked="" type="checkbox"/>	
HS000749	Phạm Ngọc Thanh Lam	8/13/2006 12:00:00 AM	L000040	<input checked="" type="checkbox"/>	
HS000750	Đinh Tú Linh	8/13/2006 12:00:00 AM	L000040	<input checked="" type="checkbox"/>	
HS000751	Nguyễn Thị Quỳnh Như	8/13/2006 12:00:00 AM	L000040	<input checked="" type="checkbox"/>	
HS000752	Phạm Trí Tài	8/13/2006 12:00:00 AM	L000040	<input checked="" type="checkbox"/>	
HS000753	Lê Phạm Thiên Thanh	8/13/2006 12:00:00 AM	L000040	<input checked="" type="checkbox"/>	
HS000754	Nguyễn Phúc Xuyên	8/13/2006 12:00:00 AM	L000040	<input checked="" type="checkbox"/>	

Bước 5: Bổ sung *Repeater*, kết nối *Repeater* với *SqlDataSource* "sdsHocSinh" tương tự như với *GridView*.

Bổ sung các Fields cho *Repeater*:

```
<asp:Repeater ID="repHocSinh" runat="server" DataMember="DefaultView"
DataSourceID="sdsHocSinh">
    <HeaderTemplate>
        <table border="1" width="50%">
            <tr>
                <th>
                    Họ tên
                </th>
                <th>
                    Ngày sinh
                </th>
                <th>
                    Giới tính
                </th>
            </tr>
    </HeaderTemplate>
    <ItemTemplate>
        <tr>
            <td>
                <asp:Label ID="HoTen" runat="server" Text='<%#Eval("HoTen")%>'></asp:Label>
            </td>
            <td>
                <asp:Label ID="NgaySinh" runat="server" Text='<%#Eval("NgaySinh")%>'></asp:Label>
            </td>
            <td>
                <asp:Label ID="GioiTinh" runat="server" Text='<%#Eval("GioiTinh")%>'></asp:Label>
            </td>
        </tr>
    </ItemTemplate>
</asp:Repeater>
```

```

<asp:Label ID="NgaySinh" runat="server"
Text='<%#DataBinder.Eval(Container.DataItem,"NgaySinh","{0:dd/MM/yyyy}")%>'></asp:Label>
</td>
<td>
<asp:Label ID="GioiTinh" runat="server"
Text='<%#DisplayGioiTinh(Eval("GioiTinh"))%>'></asp:Label>
</td>
</tr>
</ItemTemplate>
<FooterTemplate>
</table>
</FooterTemplate>
</asp:Repeater>

```

Trong đó **DisplayGioiTinh()** là hàm đã viết sẵn trong Code Behind (như ở phần trên)

```

public string DisplayGioiTinh(object obj)
{
    bool b = bool.Parse(obj.ToString());
    if (b) return "Nam";
    else return "Nữ";
}

```

Bước 6: Kiểm tra kết quả:

Họ tên	Ngày sinh	Giới tính
Đặng Trần Bảo Ân	13/08/2006	Nam
Nguyễn Tiến Đạt	13/08/2006	Nam
Nguyễn Nhựt Hồ	13/08/2006	Nam
Nguyễn Minh Hưng	13/08/2006	Nam
Trịnh Quang Khải	13/08/2006	Nam
Phạm Ngọc Thanh Lam	13/08/2006	Nam
Đinh Tú Linh	13/08/2006	Nam
Nguyễn Thị Quỳnh Như	13/08/2006	Nam
Phạm Trí Tài	13/08/2006	Nam
Lê Phạm Thiên Thanh	13/08/2006	Nam
Nguyễn Phúc Xuyên	13/08/2006	Nam

Chú ý: Các thao tác thêm, xóa, sửa sử dụng *SqlDataSource* sẽ được trình bày cụ thể trong các bài labs.

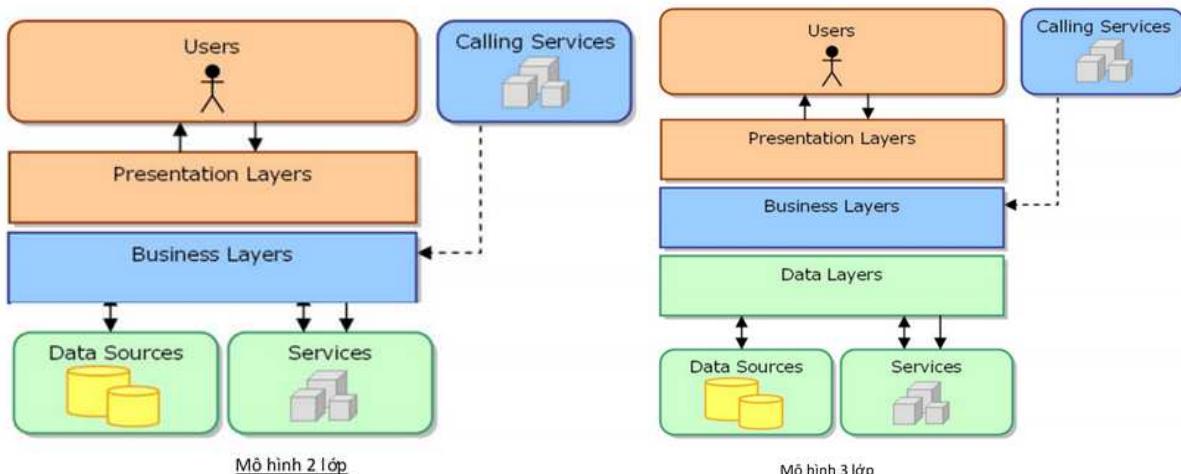
Chương 11: LINQ TO SQL & SQL.NET

Sau khi học xong bài này, học viên có khả năng :

- Xây dựng được ứng dụng ASP.NET tương tác với Cơ sở dữ liệu thông qua mô hình 2 tier
- Sử dụng được LINQ to Object để truy vấn dữ liệu trong mô hình 2 tier
- Mô tả và sử dụng được ObjectDataSource control với các Data Controls
- Xây dựng được ứng dụng ASP.NET tương tác với Cơ sở dữ liệu thông qua LINQ to SQL
- Thực hiện được truy vấn dữ liệu sử dụng LINQ to DataSet

11.1 Kiến trúc 2 lớp

Khi xây dựng các ứng dụng, việc lập trình bắt đầu trở lên phức tạp khi dự án lớn dần. Bởi vậy để dễ quản lý các thành phần của hệ thống, cũng như không bị ảnh hưởng bởi các thay đổi, nhà phát triển hay nhóm các thành phần có cùng chức năng lại với nhau và phân chia trách nhiệm cho từng nhóm để công việc không bị chồng chéo và ảnh hưởng lẫn nhau. Các mô hình lập trình phổ biến : Mô hình 2 lớp (Two Layers), Mô hình 3 lớp (Three Layers) và n lớp (n Layers).



Trong giáo trình này, chúng ta chỉ khảo sát mô hình 2 lớp để áp dụng và phát triển các ứng dụng ASP.NET. Mô hình 2 lớp được cấu thành từ: **Presentation Layers** và **Business Layers**. Các lớp này sẽ giao tiếp với nhau thông qua các dịch vụ (services) mà mỗi lớp cung cấp để tạo nên ứng dụng, lớp này cũng không cần biết bên trong lớp kia làm gì mà chỉ cần biết lớp kia cung cấp dịch vụ gì cho mình và sử dụng nó mà thôi.

▪ **Presentation Layers :**

Lớp này làm nhiệm vụ giao tiếp với người dùng cuối để thu thập dữ liệu và hiển thị kết quả/dữ liệu thông qua các thành phần trong giao diện người sử dụng. Lớp này sẽ sử dụng các dịch vụ do lớp Business Logic cung cấp. Trong .NET chúng ta có thể dùng **Windows Forms**, **ASP.NET** hay **Mobile Forms**,... để hiện thực lớp này.

▪ **Business Logic Layer**

Lớp này thực hiện các *nghiệp vụ chính* của hệ thống và *dịch vụ dữ liệu*, sau đó cung cấp các dịch vụ cho lớp **Presentation**. Lớp này chịu trách nhiệm kiểm tra các ràng buộc logic (constraints), các qui tắc nghiệp vụ (Business Rules), sử dụng các dịch vụ bên ngoài khác để thực hiện các yêu cầu của ứng dụng.

❖ Trong mô hình 3 lớp có thêm Data Layers

- **Data Layers**

Lớp này thực hiện các nghiệp vụ liên quan đến lưu trữ và truy xuất dữ liệu của ứng dụng. Thường lớp này sẽ sử dụng các dịch vụ của các hệ quản trị cơ sở dữ liệu như SQL Server, Oracle, Access, XML... để thực hiện nhiệm vụ của mình. Khi đó **Business Logic Layer** sẽ truy cập dữ liệu từ **Data Layer**.

11.2 Xây dựng ứng dụng ASP.NET theo kiến trúc 2 Lớp

Trong phần này, chúng ta xây dựng ứng dụng ASP.NET gồm các chức năng Thêm, Xóa, Sửa 1 mặt hàng :

The screenshot shows a web page with a form at the top and a table below it. The form has fields for ProductID (93) and ProductName (Caphe), and buttons for Add New, Update, and Delete. The table lists products with columns for ProductID and ProductName.

ProductID	ProductName
93	Caphe
77	Original Frankfurter grüne Soße
76	Lakkalikööri
75	Rhönbräu Klosterbier
74	Longlife Tofu

Bước 1

- 1.1 Tạo ứng dụng ASP.NET đặt tên là SaleApp
- 1.2 Trên SQL Server 2008 tạo 1 database tên Sale gồm 1 bảng tên Products (ProductID int Primary Key, ProductName varchar(50))

Bước 2 Xây dựng Business Logic Layer

2.1 Thêm vào thư mục *App_Code* 2 lớp với 2 tập tin : *ProductData.cs* và *Products.cs*

2.2 Viết code cho từng lớp như sau:

- Lớp Product trong tập tin **Product.cs**

```

App_Code/Products.cs <input type="button" value="X" style="float: right;" />
Product
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

public class Product
{
    public int ProductID { get; set; }
    public string ProductName { get; set; }
}

```

- Lớp ProductData trong tập tin **ProductData.cs**

```
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Collections;
using System.Data;
public class ProductData{
    public ProductData(){ }
    //:@:Thay doi chuoi ket noi cho phu hop
    string strConnection = "server=.\\SQL2K8;database=Sale;uid=sa;pwd=123";
    public void InsertProduct(Product p) {
        SqlConnection cnn = new SqlConnection(strConnection);
        string SQLInsert =
            "Insert Products(ProductID,ProductName) values(@ProID,@ProName)";
        SqlCommand cmd = new SqlCommand(SQLInsert, cnn);
        cmd.Parameters.Add("@ProID", p.ProductName);
        cmd.Parameters.Add("@ProName", p.ProductName);
        try{
            cnn.Open();
            cmd.ExecuteNonQuery();
        }
        catch(Exception ex) {
            throw new Exception("Error:"+ex.Message);
        }
        finally {
            cnn.Close();
        }
    }

    public void UpdateProduct(Product p){
        SqlConnection cnn = new SqlConnection(strConnection);
        string SQLUpdate =
            "Update Products set ProductName = @Name Where ProductID=@ProID";
        SqlCommand cmd = new SqlCommand(SQLUpdate, cnn);
        cmd.Parameters.Add("@Name", p.ProductName);
        cmd.Parameters.Add("@ProID", p.ProductID);
        try{
            cnn.Open();
            cmd.ExecuteNonQuery();
        }
        catch{
            throw new Exception("Error");
        }
        finally {
            cnn.Close();
        }
    }
}
```

```
public void DeleteProduct(Product p) {
    SqlConnection cnn = new SqlConnection(strConnection);
    string SQLInsert =
        "Delete Products where ProductID=@ProID";
    SqlCommand cmd = new SqlCommand(SQLInsert, cnn);
    cmd.Parameters.AddWithValue("@ProID", p.ProductID);
    try{
        cnn.Open();
        cmd.ExecuteNonQuery();
    }
    catch{
        throw new Exception("Error");
    }
    finally{
        cnn.Close();
    }
}

public List<Product> GetProducts(){
    List<Product> data = new List<Product>();
    string SQL = "select ProductID,ProductName from Products";
    SqlConnection cnn = new SqlConnection(strConnection);
    SqlCommand cmd = new SqlCommand(SQL, cnn);
    cnn.Open(); //Phai mo ket noi
    SqlDataReader rd = cmd.ExecuteReader
        (CommandBehavior.CloseConnection);
    if (rd.HasRows) //neu co du lieu tra ve tu cau lenh Select
    {
        while (rd.Read()) //doc tung dong tu bang Products
        {
            Product p = new Product(){
                ProductID=int.Parse(rd["ProductID"].ToString()),
                ProductName = rd.GetString(1)
            };
            data.Add(p);
        }
    }
    //su dung LINQ to Object de sap cot ProductID giam dan va lay 5 product
    return data.OrderByDescending
        (p=>p.ProductID).Take(5).ToList();
}
```

Bước 3 Xây dựng Presentation Layers

3.1 Thiết kế giao diện cho trang Default.aspx như hình sau :

Default.aspx Start Page

ProductID	<input type="text"/>	
ProductName	<input type="text"/>	
<input type="button" value="Add New"/> <input type="button" value="Update"/> <input type="button" value="Delete"/>		
Column0	Column1	Column2
abc	abc	abc

➤ Code HTML

```
<form id="form1" runat="server">
    ProductID &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
    <asp:TextBox ID="txtProductID" runat="server"></asp:TextBox>
    <br />
    ProductName
    <asp:TextBox ID="txtProductName" runat="server"></asp:TextBox>
    <br />
    <asp:Button ID="btnAddNew" runat="server" Text="Add New" />&nbsp;
    <asp:Button ID="btnUpdate" runat="server" Text="Update" />&nbsp;
    <asp:Button ID="btnDelete" runat="server" Text="Delete" />
    <br />
    <asp:GridView ID="gvProducts" runat="server">
    </asp:GridView>
    <br />
</form>
```

3.2 Viết code cho sự kiện **Click** của các Button và **Page_Load** trong tập tin Default.aspx.cs như sau:

```
public partial class _Default : System.Web.UI.Page {
    ProductData pdata = new ProductData();
    protected void Page_Load(object sender, EventArgs e) {
        if (!IsPostBack)
            LoadData();
    }
    public void LoadData() {
        gvProducts.DataSource = pdata.GetProducts();
        gvProducts.DataBind();
    }
    protected void btnAddNew_Click(object sender, EventArgs e) {
        int ProID = int.Parse(txtProductID.Text);
        Product p = new Product { ProductID = ProID, ProductName = txtProductName.Text };
        pdata.InsertProduct(p);
        LoadData();
    }
    protected void btnUpdate_Click(object sender, EventArgs e) {
        int ProID = int.Parse(txtProductID.Text);
        Product p = new Product { ProductID = ProID, ProductName = txtProductName.Text };
        pdata.UpdateProduct(p);
        LoadData();
    }
    protected void btnDelete_Click(object sender, EventArgs e) {
        int ProID = int.Parse(txtProductID.Text);
        Product p = new Product { ProductID = ProID };
        pdata.DeleteProduct(p);
        LoadData();
    }
}
```

Bước 4 : Nhấn Ctrl+F5 để chạy ứng dụng và kiểm tra các chức năng.

ProductID	<input type="text" value="93"/>
ProductName	<input type="text" value="Caphe"/>
<input type="button" value="Add New"/> <input type="button" value="Update"/> <input type="button" value="Delete"/>	
ProductID	ProductName
93	Caphe
77	Original Frankfurter grüne Soße
76	Lakkalikööri
75	Rhönbräu Klosterbier
74	Longlife Tofu

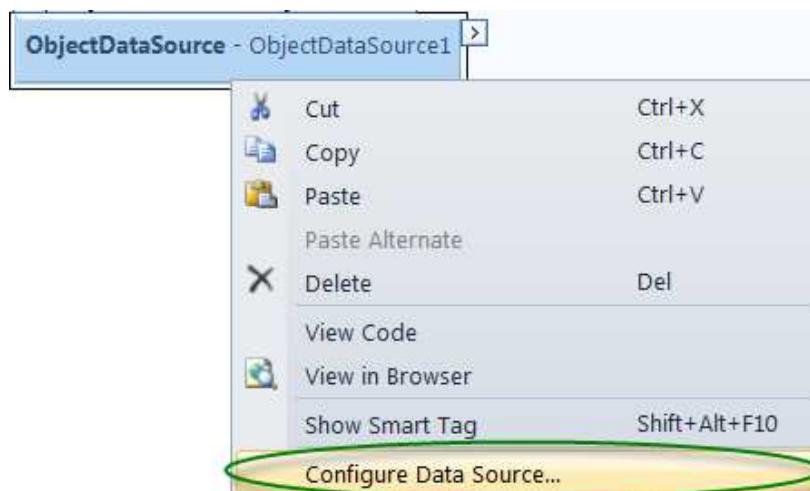
11.3 ObjectDataSource control

Khi ta phát triển ứng dụng theo mô hình 2 lớp hay 3 lớp, tất cả chức năng của ứng dụng sẽ thực thi thông qua Business Layers và chúng ta cũng phải viết code cho các sự kiện của các controls ở tầng Presentation. ObjectDataSource controls sẽ thực thi các chức năng của các Data controls một cách tự động thông qua các phương thức đã khai báo trong lớp ProductData. Như vậy, trang Default.aspx thay vì thiết kế và viết code như trên sẽ được thiết kế lại như sau :

The screenshot shows a Windows application interface. At the top, there's a title bar with the text "FormView - FormView1". Below the title bar is a toolbar with several icons. The main area contains a "FormView" control. Inside the FormView, there is a grid table with three columns labeled "Column0", "Column1", and "Column2". The grid contains five rows, each with the value "abc" in all three columns. Below the FormView is another control labeled "ObjectDataSource - ObjectDataSource1".

- Cấu hình cho ObjectDataSource1 theo các bước sau đây :

Bước 1. Chọn ObjectDataSource1 -> nhấp phải chuột và chọn **Config Data Source...**



Bước 2. Trên hộp thoại Config Data Source, chọn ProductData | nhấp Next để cấu hình các lệnh select, insert, update và delete và nhấn Finish để kết thúc.

Configure Data Source - ObjectDataSource1

Choose a Business Object

Select a business object that can be used to retrieve or update data (App_Code directory for this application).

Choose your business object:

Product

ProductData

SELECT UPDATE INSERT DELETE

Choose a method of the business object that returns data to associate with the SELECT operation. The method can return a DataSet, DataReader, or strongly-typed collection.

Example: GetProducts(Int32 categoryId), returns a DataSet.

Choose a method:

GetProducts(), returns List<Product>

Method signature:

GetProducts(), returns List<Product>

Bước 3.

- Thiết lập thuộc tính DataSource là ObjectDataSource1 cho GridView control và FormView Control.
- Thiết lập thuộc tính DataKeyNames là ProductID cho FormView control

ProductID: 0
 ProductName: abc
[Edit](#) [Delete](#) [New](#)

ProductID	ProductName
0	abc
1	abc
2	abc
3	abc
4	abc

ObjectDataSource - ObjectDataSource1

Bước 4. Nhấn Ctrl+F5 để chạy ứng dụng và kiểm tra các chức năng

ProductID: 77

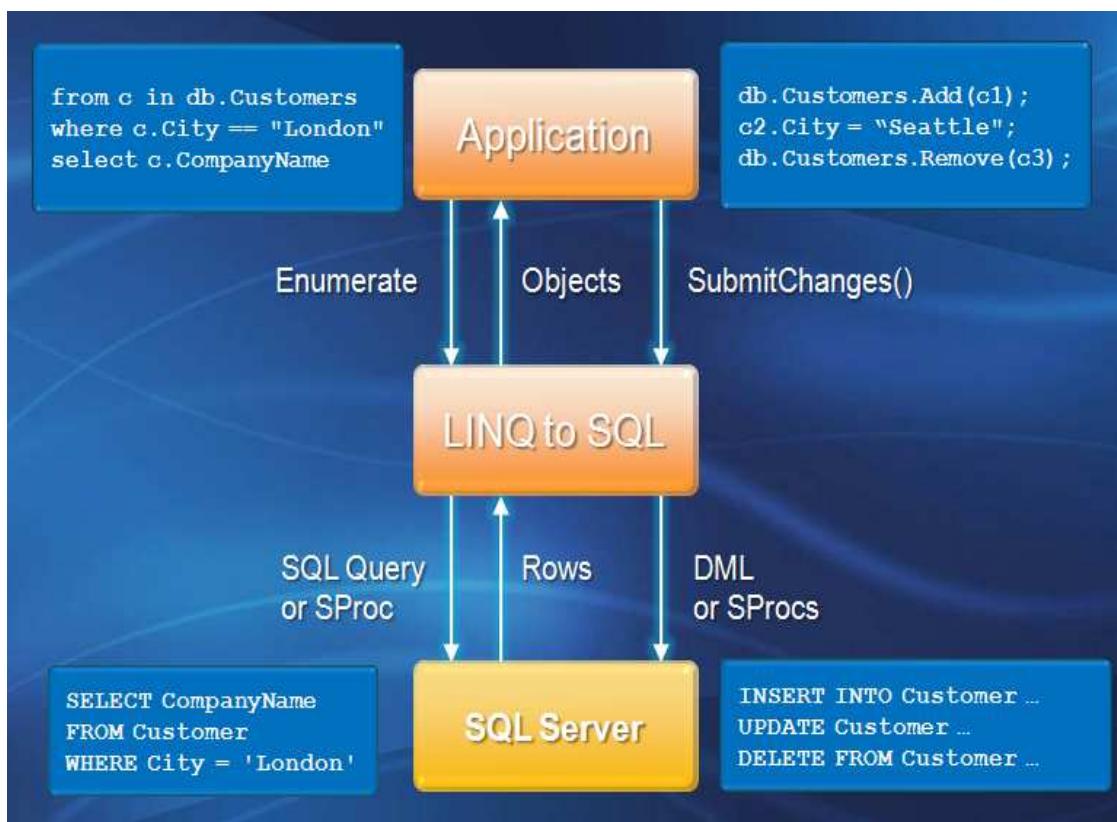
ProductName: Original Frankfurter grüne Soße

[Edit](#) [Delete](#) [New](#)

ProductID	ProductName
77	Original Frankfurter grüne Soße
76	Lakkalikööri
75	Rhönbräu Klosterbier
74	Longlife Tofu
73	Röd Kaviar

11.4 LINQ to SQL

LINQ to SQL là một phiên bản hiện thực hóa của O/RM (object relational mapping) có bên trong .NET Framework bản “Orcas” (.NET Framework 3.5), nó cho phép các nhà phát triển mô hình hóa một cơ sở dữ liệu dùng các lớp .NET. Sau đó có thể truy vấn cơ sở dữ liệu (CSDL) dùng LINQ, cũng như thực hiện các tác vụ cập nhật, thêm, xóa dữ liệu từ đó.

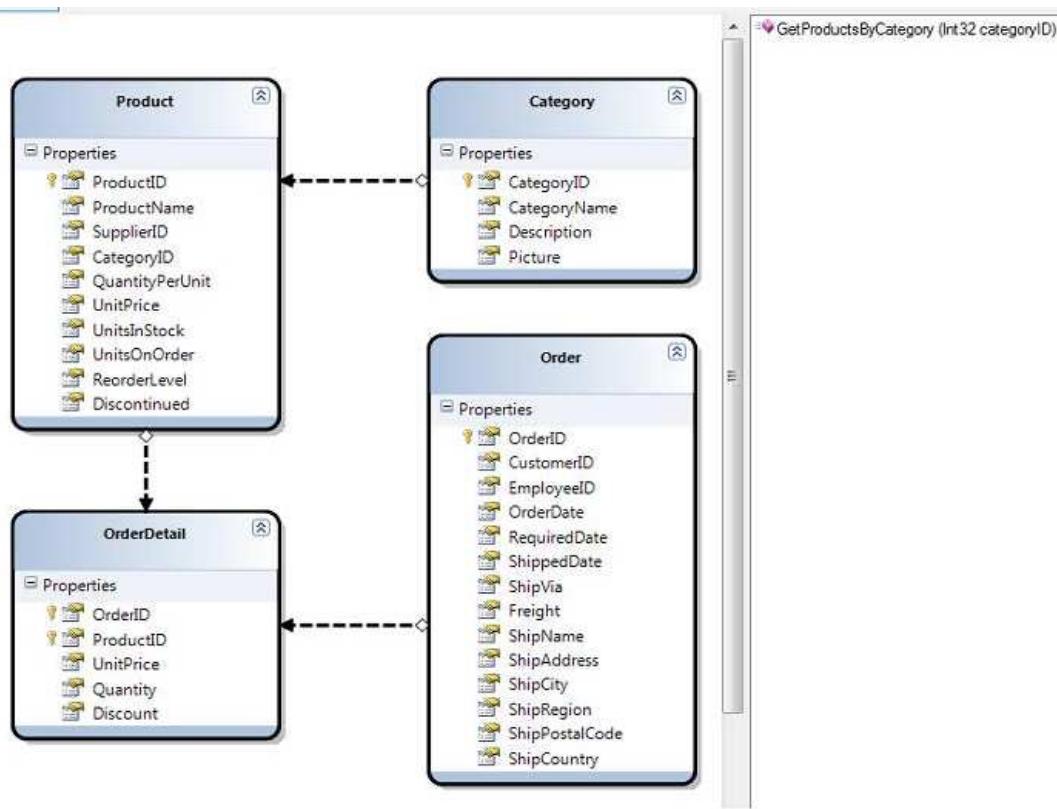


Mô hình hoạt động của LINQ to SQL

11.4.1 Mô hình hóa cơ sở dữ liệu dùng LINQ to SQL

LINQ to SQL hỗ trợ đầy đủ transaction, view và các stored procedure (SP). Nó cũng cung cấp một cách dễ dàng để thêm khả năng kiểm tra tính hợp lệ của dữ liệu và các quy tắc vào trong mô hình dữ liệu.

- Visual Studio “Orcas” đã tích hợp thêm một trình thiết kế LINQ to SQL như một công cụ dễ dàng cho việc mô hình hóa một cách trực quan các CSDL dùng LINQ to SQL.
- Bằng cách dùng trình thiết kế LINQ to SQL, chúng ta có thể dễ dàng tạo một mô hình cho CSDL mẫu “Northwind” sau đây:



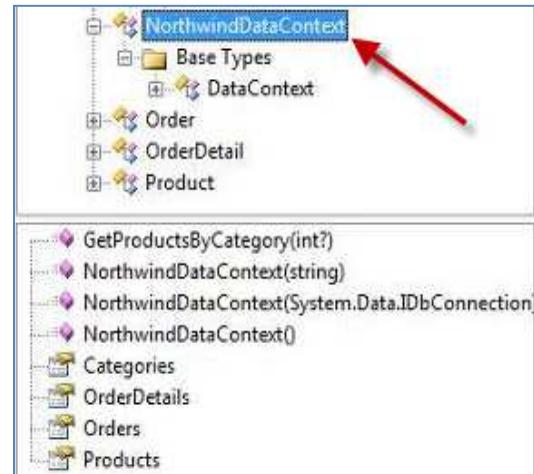
- Mô hình LINQ to SQL ở trên định nghĩa bốn lớp thực thể: Product, Category, Order và OrderDetail. Các thuộc tính của mỗi lớp ánh xạ vào các cột của bảng tương ứng trong CSDL. Mỗi instance của một lớp biểu diễn một dòng trong bảng dữ liệu.
- Các mũi tên giữa bốn lớp thực thể trên biểu diễn quan hệ giữa các thực thể khác nhau, chúng được tạo ra dựa trên các mối quan hệ primary-key/foreign-key trong CSDL. Hướng của mũi tên chỉ ra mối quan hệ là một – một hay một – nhiều. Các thuộc tính tương ứng sẽ được thêm vào các lớp thực thể trong các trường hợp này. Lấy ví dụ, lớp Category ở trên có một mối quan hệ một nhiều với lớp Product, điều này có nghĩa nó sẽ có một thuộc tính "Categories" là một tập hợp các đối tượng Product trong Category này. Lớp Product cũng sẽ có một thuộc tính "Category" chỉ đến đối tượng "Category" chứa Product này bên trong.
- Bảng các phương thức tay phải bên trong trình thiết kế LINQ to SQL ở trên chứa một danh sách các Store Procedure (SP) để tương tác với mô hình dữ liệu của chúng ta. Trong mô hình trên ta đã thêm một thủ tục có tên `GetProductsByCategory` mà nó nhận vào một categoryId và trả về một dãy các Product. Chúng ta sẽ tìm hiểu cách gọi được thủ tục này trong một đoạn code tiếp theo.

11.4.2 Lớp DataContext trong LINQ to SQL

Khi chúng ta lưu màn hình thiết kế LINQ to SQL, Visual Studio sẽ lưu các lớp .NET biểu diễn các thực thể và quan hệ bên trong CSDL mà chúng ta vừa mô hình hóa. Cứ mỗi một file LINQ to SQL chúng ta thêm vào solution, một lớp DataContext sẽ được tạo ra, nó sẽ được dùng khi cần truy vấn hay cập nhật lại các thay đổi. Lớp DataContext được tạo sẽ có các thuộc tính để biểu diễn mỗi bảng được mô hình hóa từ CSDL, cũng như các phương thức cho mỗi Store Procedure mà chúng ta đã thêm vào.

11.4.3 Một số thao tác dữ liệu trong LINQ to SQL

Một khi đã mô hình hóa CSDL dùng trình thiết kế LINQ to SQL, chúng ta có thể dễ dàng viết các đoạn lệnh để làm việc với nó. Dưới đây là một số ví dụ minh họa về các thao tác chung khi xử lý dữ liệu :



- Lấy danh sách các Products**

Đoạn lệnh dưới đây dùng cú pháp LINQ để lấy về một tập IEnumerable các đối tượng Product. Các sản phẩm được lấy ra phải thuộc phân loại "Beverages"

```

NorthwindDataContext db = new NorthwindDataContext();

var products = from p in db.Products
               where p.Category.CategoryName == "Beverages"
               select p;

```

- Cập nhật thông tin Product**

Đoạn lệnh dưới đây dùng cú pháp LINQ để cập nhật UnitPrice và UnitsInStock của Product

```

NorthwindDataContext db = new NorthwindDataContext();

Product product = db.Products.Single(p => p.ProductName == "Chai" );
product.UnitPrice = 99;
product.UnitsInStock = 5;
db.SubmitChanges();

```

- Thêm mới 1 Category**

Đoạn lệnh dưới đây dùng cú pháp LINQ để thêm 1 Category vào bảng Categories

```

NorthwindDataContext db = new NorthwindDataContext();

// Create new Category
Category category = new Category();
category.CategoryName = "Scott's Toys";
// Add category to database and save changes
db.Categories.Add(category);
db.SubmitChanges();

```

- Xóa 1 Product**

Đoạn lệnh dưới đây dùng cú pháp LINQ để xóa 1 Product trong bảng Products

```
NorthwindDataContext db = new NorthwindDataContext();
var p = from p in db.Products
        where p.ProductID=1
        select p;
db.Products.DeleteOnSubmit(p);
db.SubmitChanges();
```

- Gọi 1 thủ tục (store procedure)**

Đoạn lệnh dưới đây dùng cú pháp LINQ gọi 1 store procedure

```
NorthwindDataContext db = new NorthwindDataContext();
var products = db.GetProductsByCategory(5);
foreach (Product product in products)
{
    //Viết code ở đây...
}
```

11.4.4

Các bước phát triển ứng dụng LINQ to SQL

Bước 1

1.1 Tạo ứng dụng ASP.NET đặt tên là TestLINQtoSQL gồm trang Default.aspx như sau:

Column0	Column1	Column2
abc	abc	abc

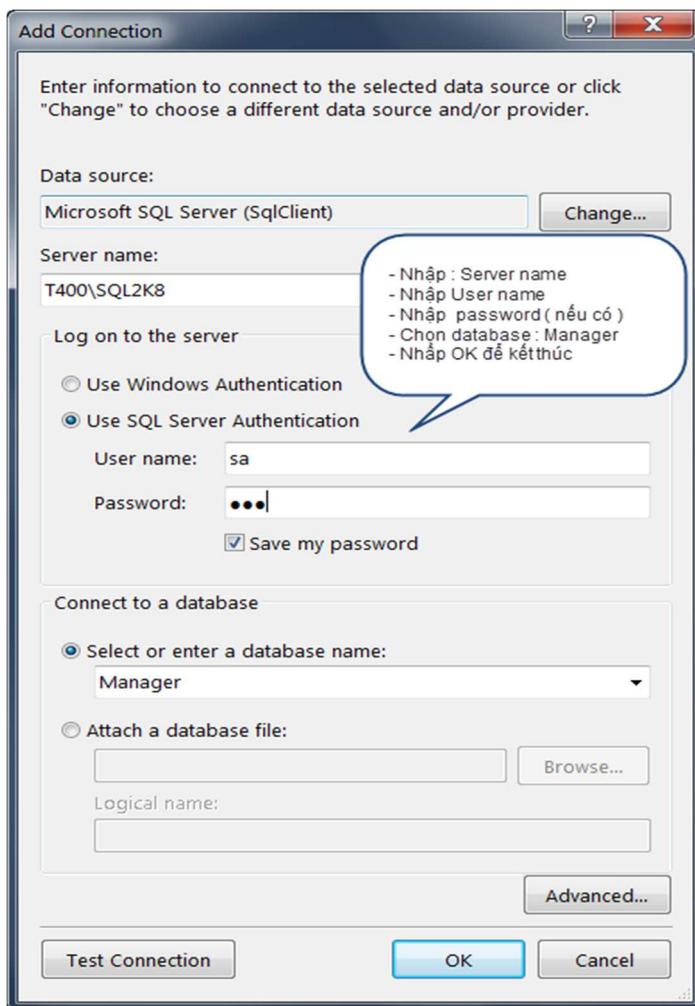
1.2 Trên SQL Server 2008 tạo 1 cơ sở dữ liệu tên Manager gồm 1 bảng tên Products
 (ProductID int Primary Key, ProductName varchar(50))

dbo.Products: Table(ibm.Manager)		
	Column Name	Data Type
	Allow Nulls	
ProductID	int	<input type="checkbox"/>
ProductName	varchar(50)	<input type="checkbox"/>

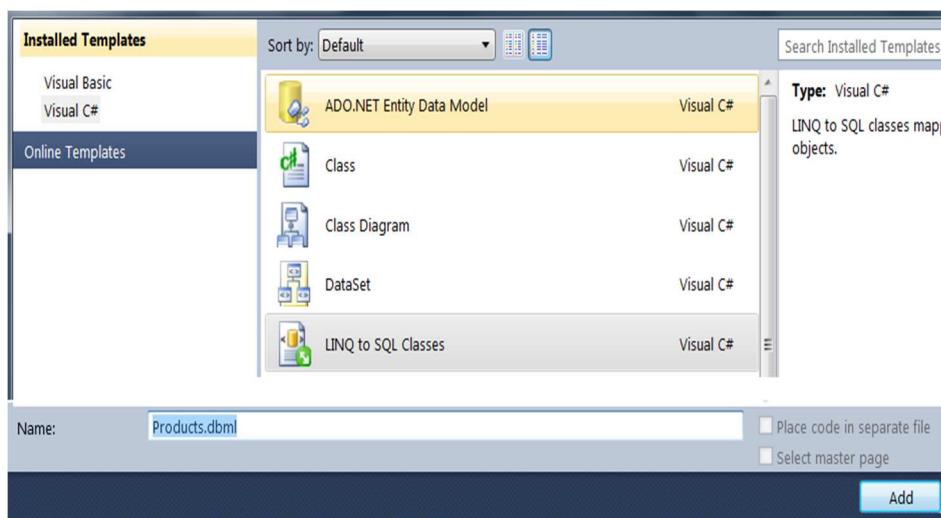
Bước 2 Xây dựng lớp LINQ to SQL

2.1 Tạo DataConnection để kết nối với cơ sở dữ liệu Manager

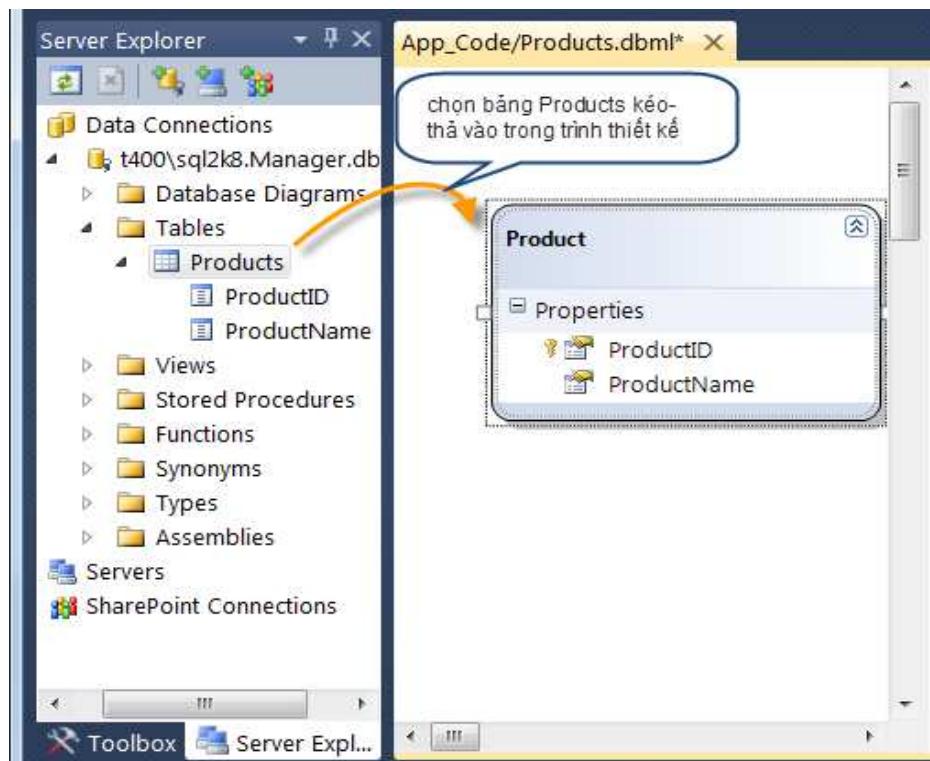
Trên thanh công cụ *Server Explorer | Data Connection*, nhấp phải chuột chọn *Add New Connection* (để mở hộp thoại *Server Explorer* vào *View | Server Explorer*)



2.2 Vào menu *Project | Add New Items* , trên hộp thoại *Add New Items* , chọn **LINQ to SQL Classes** và đặt tên **Products.dbml** , nhấp **Add** để kết thúc.



Từ hộp thoại *Server Explorer* chọn bảng *Products* kéo thả vào *Products.dbml* và nhấn **Shift+F6** để biên dịch ứng dụng.



Bước 3 Viết code cho *sự kiện Page_Load* và *sự kiện Click* của các Button controls trong tập tin Default.aspx.cs

```

protected void Page_Load(object sender, EventArgs e){
    if (!IsPostBack)
        LoadData();
}
public void LoadData() {
    ProductsDataContext pdata = new ProductsDataContext();
    gvProducts.DataSource = pdata.Products;
    gvProducts.DataBind();
}
protected void btnAddNew_Click(object sender, EventArgs e){
    ProductsDataContext pdata = new ProductsDataContext();
    int ProID = int.Parse(txtProductID.Text);
    Product p = new Product{
        ProductID= ProID,
        ProductName = txtProductName.Text };
    pdata.Products.InsertOnSubmit(p);
    pdata.SubmitChanges();
    LoadData();
}

```

```

protected void btnUpdate_Click(object sender, EventArgs e){
    ProductsDataContext pdata = new ProductsDataContext();
    int ProID = int.Parse(txtProductID.Text);
    Product p = pdata.Products.SingleOrDefault(pro => pro.ProductID == ProID);
    p.ProductName = txtProductName.Text;
    pdata.SubmitChanges();
    LoadData();
}

protected void btnDelete_Click(object sender, EventArgs e){
    ProductsDataContext pdata = new ProductsDataContext();
    int ProID = int.Parse(txtProductID.Text);
    Product p = pdata.Products.SingleOrDefault(pro => pro.ProductID == ProID);
    pdata.Products.DeleteOnSubmit(p);
    pdata.SubmitChanges();
    LoadData();
}

```

Bước 4. Nhấn Ctrl+F5 để chạy ứng dụng và kiểm tra các chức năng

The screenshot shows a Windows application window. At the top, there are two text input fields: 'ProductID' with value '1' and 'ProductName' with value 'caphe'. Below these are three buttons: 'Add New' (highlighted in blue), 'Update', and 'Delete'. At the bottom, there is a grid view table with two columns: 'ProductID' and 'ProductName'. It contains one row with values '1' and 'caphe' respectively.

ProductID	ProductName
1	caphe

11.5 LinqDataSource Control

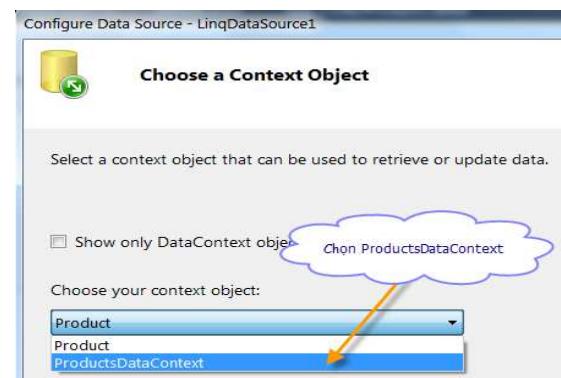
LinqDataSource control cho phép thao tác dữ liệu tự động với các Data controls như : GridView, FormView, ... thông qua LINQ to SQL Classes thay vì chúng ta phải viết code cho trang các controls trên trang ASP.NET ,do đó trang Default.aspx ở trên chúng ta thiết kế lại như hình dưới đây.

The screenshot shows the Visual Studio designer interface. A 'FormView' control is selected, with its title bar showing 'FormView - FormView1'. Below the title bar, there is a message: 'Right-click or choose the Edit Templates task to edit template content. The ItemTemplate is required.' Inside the FormView, there is a table with three columns labeled 'Column0', 'Column1', and 'Column2'. The table has five rows, each containing the value 'abc' in all three columns. At the bottom of the FormView control, there is a label 'LinqDataSource - LinqDataSource1'.

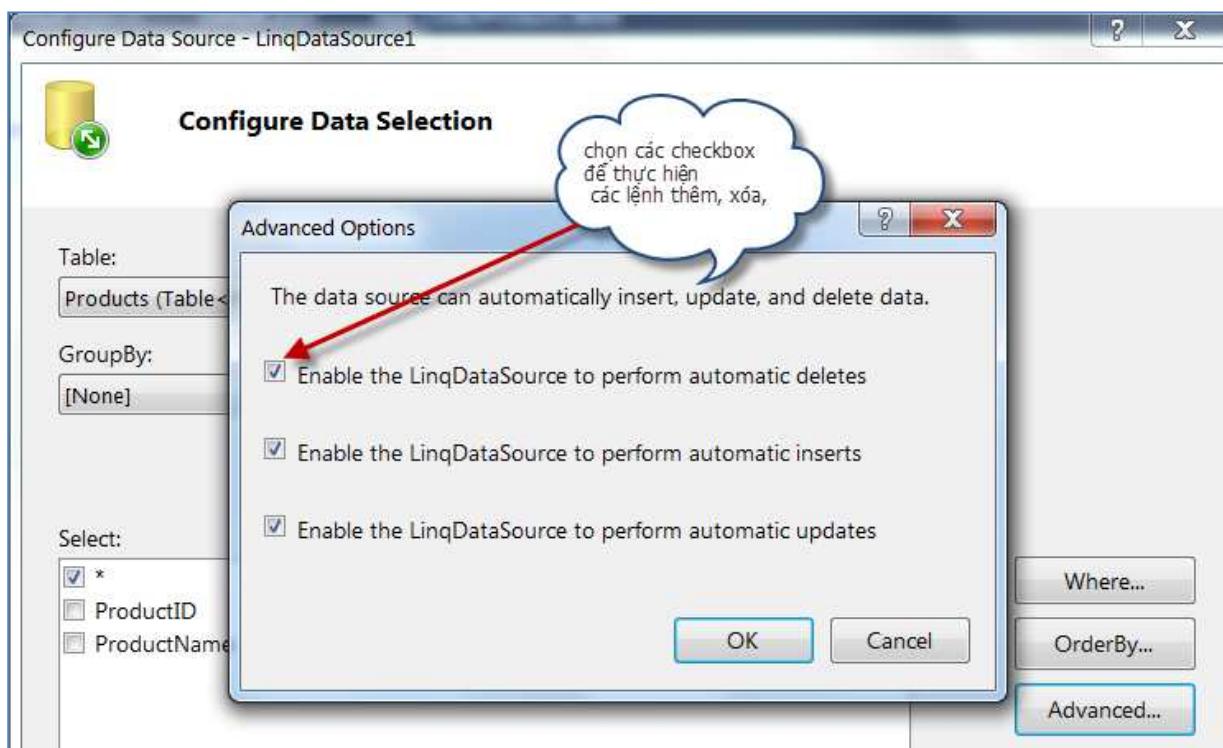
Để cấu hình cho LinqDataSource control ta thực hiện các bước sau đây :

Bước 1. Chọn LinqDataSource1 , nhấp phải chuột chọn *Configure Data Source...*

Bước 2. Trên hộp thoại Configure Data Source chọn ProductsDataContext , nhấp next để tiếp tục.



Bước 3. Nhấp vào nút Advanced và chọn các checkbox : insert, update, delete. Nhấp OK -> Finish để kết thúc.



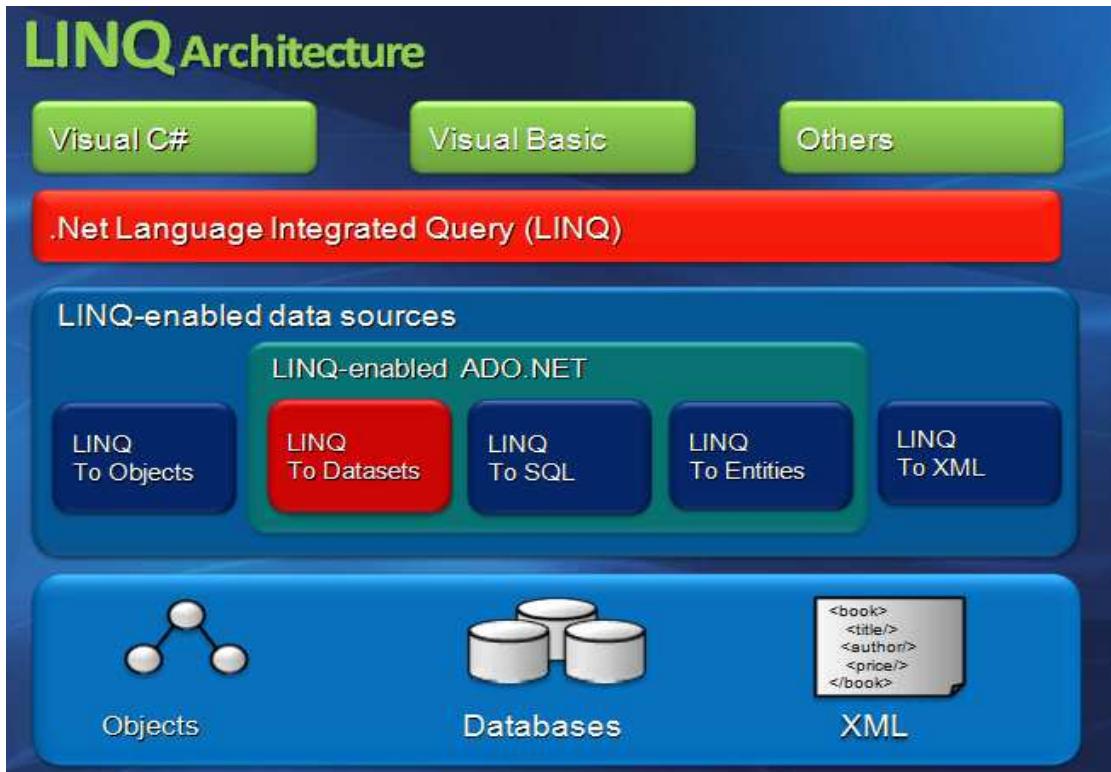
Bước 4. Thiết lập giá trị cho thuộc tính DataSource của FormView1 và GridView1 là LinqDataSource1.

Bước 5. Nhấn Ctrl+F5 để chạy ứng dụng và kiểm tra kết quả.

ProductID: 3
ProductName: Caphe
Edit Delete New
ProductID ProductName
3 Caphe

11.6 LINQ to DataSet

LINQ to DataSet dùng để thao tác với các cơ sở dữ liệu khác chẳng hạn như : Access, Oracle, Excel,... sử dụng cú pháp chung của LINQ.



Để tìm hiểu về LINQ to DataSet chúng ta từng bước xây dựng ứng dụng tìm thông tin mặt hàng theo đơn giá

Bước 1

1.1 Tạo ứng dụng ASP.NET đặt tên là TestLINQtoDataSet gồm trang SearchProducts.aspx có giao diện như hình dưới đây :

ProductID	<input type="text"/>	
<input type="button" value="Search"/>		
Column0	Column1	Column2
abc	abc	abc

Giao diện trang SearchProduct.aspx

```
<form id="form1" runat="server">
<div>
    ProductID
    <asp:TextBox ID="txtProductID" runat="server"></asp:TextBox>
    <br />
    <asp:Button ID="btnSearch" runat="server" Text="Search" />
    <br />
    <asp:GridView ID="gvProductDetail" runat="server">
    </asp:GridView>
</div>
</form>
```

Mã HTML

1.2 Tạo cơ sở dữ liệu tên Manager gồm 1 bảng tên Products

dbo.Products: Table(ibm.Manager)			
	Column Name	Data Type	Allow Nulls
?	ProductID	int	<input type="checkbox"/>
	ProductName	varchar(50)	<input type="checkbox"/>

Bước 2. Viết code cho nút Search để tìm Product theo ProductID

```
protected void btnSearch_Click(object sender, EventArgs e)
{
    string chuoiketNoi = "server=SwordLake\\sql2k8;database=Manager;uid=sa;pwd=123";
    string cauLenhSQL = "select ProductID,ProductName from Products";
    SqlDataAdapter da = new SqlDataAdapter(cauLenhSQL, chuoiketNoi);
    DataSet dsProducts = new DataSet();
    da.Fill(dsProducts);

    int ProID = int.Parse(txtProductID.Text);
    var product = from p in dsProducts.Tables[0].AsEnumerable()
                  where p.Field<int>("ProductID") == ProID
                  select new
                  {
                      ProductID = p.Field<int>("ProductID"),
                      ProductName = p.Field<string>("ProductName")
                  };
    gvProductDetail.DataSource = product;
    gvProductDetail.DataBind();
}
```

Bước 3. Chạy ứng dụng và kiểm tra kết quả

The screenshot shows a Windows application window. At the top, there is a text input field labeled "ProductID" containing the value "1". Below it is a blue "Search" button. Underneath the search controls is a data grid with two columns: "ProductID" and "ProductName". A single row is visible in the grid, showing "1" in the "ProductID" column and "chai" in the "ProductName" column.

ProductID	ProductName
1	chai

Chương 12: **AJAX & ASP.NET**

Sau khi học xong bài này, học viên có khả năng :

- Trình bày được vai trò của Validation trong ứng dụng WEB
- Trình bày được vai trò của công nghệ Ajax trong các ứng dụng WEB
- Triển khai ứng dụng Ajax với mã JavaScript thuần túy
- Triển khai ứng dụng Ajax với cơ chế Callback trong ASP.NET
- Sử dụng thành thạo các điều khiển Ajax của ASP.NET như ScriptManager, UpdatePanel, UpdateProgress, Timer
- Khai thác hiệu quả các điều khiển và mở rộng điều khiển (control extender) ajax của Ajax Toolkit
- Xây dựng ứng dụng Ajax với Jquery

12.1 Giới thiệu Ajax

AJAX là một công nghệ cho phép lập trình bất đồng bộ trong ứng dụng Web. Thông thường người dùng muốn thay đổi thông tin từ trang Web bằng cách nhấp vào các nút lệnh (button) hay các liên kết (link) để submit yêu cầu về Web Server để thay đổi nội dung trang Web (postback). Như vậy toàn bộ trang Web phải được xử lý lại do đó tốn khá nhiều thời gian và gia tăng sự phản hồi các trang Web,... Công nghệ Ajax (**Aynchronous JavaScript and XML**) cho phép chỉ các thông tin nào cần thay đổi được gửi về Sever xử lý, sau đó Server sẽ xử lý và trả kết quả về cho Client. Sau đây là một vài thông tin chung sẽ giúp chúng ta hiểu hơn về Ajax:

AJAX bắt đầu phổ biến từ năm 2005 bởi Google (với một ứng dụng Google Suggest, Google Maps, Gmail). AJAX không phải là ngôn ngữ lập trình mới, mà nó là một công nghệ mới để tạo ra một ứng dụng web nhỏ hơn, nhanh hơn, tốt hơn và giao diện thân thiện với người dùng hơn.

Ajax dựa trên các thành phần HTML trước đây:

- ✓ HTML
- ✓ CSS
- ✓ JavaScript (chủ chốt)
- ✓ XML

AJAX là một công nghệ được hỗ trợ bởi trình duyệt (browser) và nó độc lập với các ứng dụng Web server. Với Ajax, Javascript của bạn có thể liên lạc trực tiếp với Web server bằng cách sử dụng đối tượng XMLHttpRequest của Javascript. Với đối tượng này Javascript của bạn có thể trao đổi dữ liệu trực tiếp Web server mà không cần đệ trình (submit) toàn bộ dữ liệu đến, do đó trang web của bạn không reload lại.

Ajax sử dụng cơ chế làm việc bất đồng bộ (Asynchronous), tức là trong khi đối tượng XMLHttpRequest thực hiện gửi yêu cầu đến Web server thì Web browser vẫn tiếp tục xử lý các công việc khác mà không cần Web server hoàn thành việc trả lời lại yêu cầu đó. Nhiều công việc được xử lý song song với nhau, điều này khác với cách lập trình web cổ điển trước đây, do đó ứng dụng web sẽ chạy nhanh hơn.

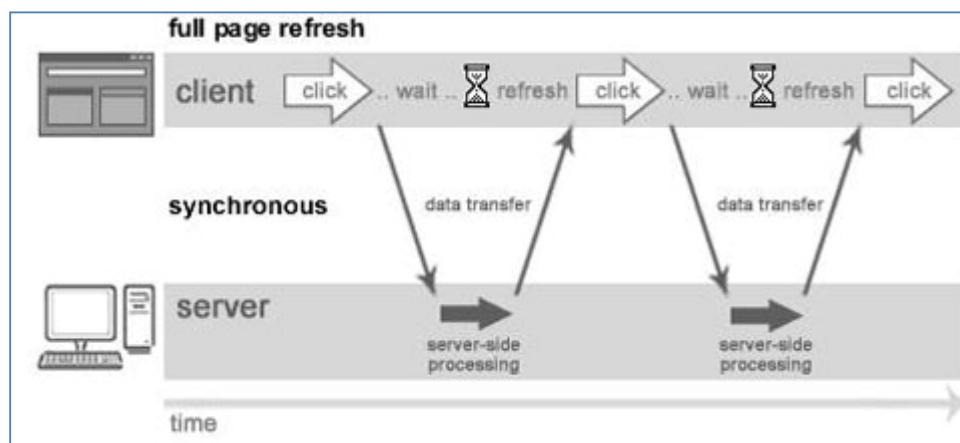
Ajax là một kỹ thuật của Web browser và độc lập với Web server. Tất cả Web có sử dụng Ajax gọi là Web 2.0. Ajax có thể gửi và nhận dữ liệu với nhiều định dạng khác nhau, bao gồm XML, HTML và thậm chí là file text.

12.2 Cơ chế làm việc của ajax

Ta hãy phân tích và so sánh cách thức hoạt động của một trang web thông thường và một trang web có ứng dụng Ajax để thấy rõ cách thức thực hiện của Ajax.

12.2.1 Cơ chế truyền thông đồng bộ

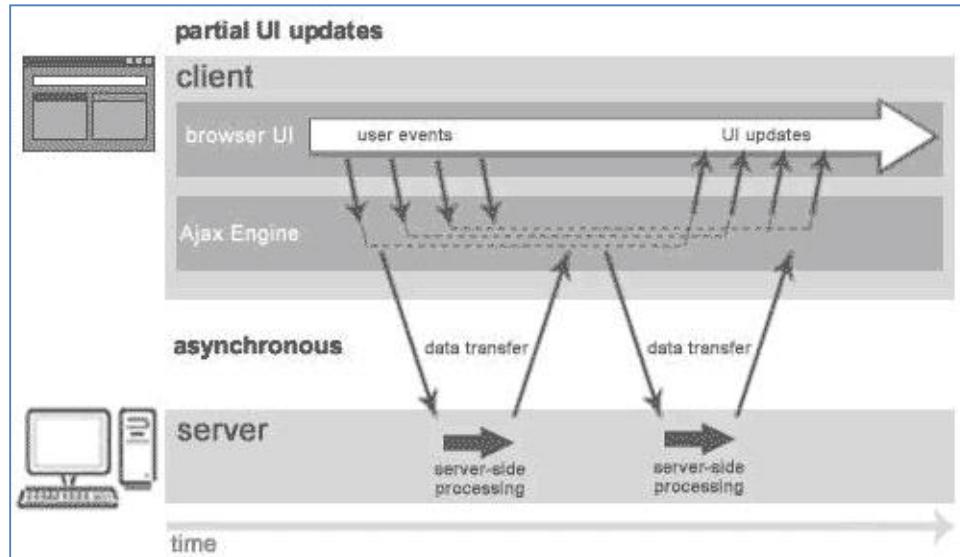
Với cách lập trình Web trước đây (còn gọi là Web 1.0) thì khi người dùng cần cập nhật thông tin (click vào một nút nào đó), thì yêu cầu thay đổi thông tin sẽ được gửi từ phía Client về Server dưới dạng HTTP request, toàn bộ trang web sẽ được gửi chứ không riêng gì một vài thông tin cần thay đổi (dạng này gọi là postback). Lúc đó Client sẽ rơi vào trạng thái chờ (waiting...), trong lúc này phía Client không thể thực hiện một công việc nào khác. Khi Server xử lý hoàn thành các yêu cầu và thì sẽ gửi trả lại cho phía Client một trang web khác thay thế trang cũ (thông tin mà Server response lại ở dạng HTML và CSS). Qui trình này được mô tả như sau :



Như vậy ta thấy cách thức hoạt động của 1 trang web cổ điển là : Click → waiting... → refresh ... → ... Do đó cho dù yêu cầu cập nhật một lượng thông tin nhỏ thì trang web cũng phải load lại, do đó cách trang web chạy chậm.

12.2.2 Cơ chế truyền thông bất đồng bộ

Với cách lập trình Web có ứng dụng kỹ thuật Ajax thì Ajax cho phép tạo ra một Ajax Engine nằm giữa UI (user interface – giao diện người dùng) và Server, tức là nằm giữa giao tiếp Client – Server, nhưng phần Ajax Engine này vẫn nằm ở phía Client.



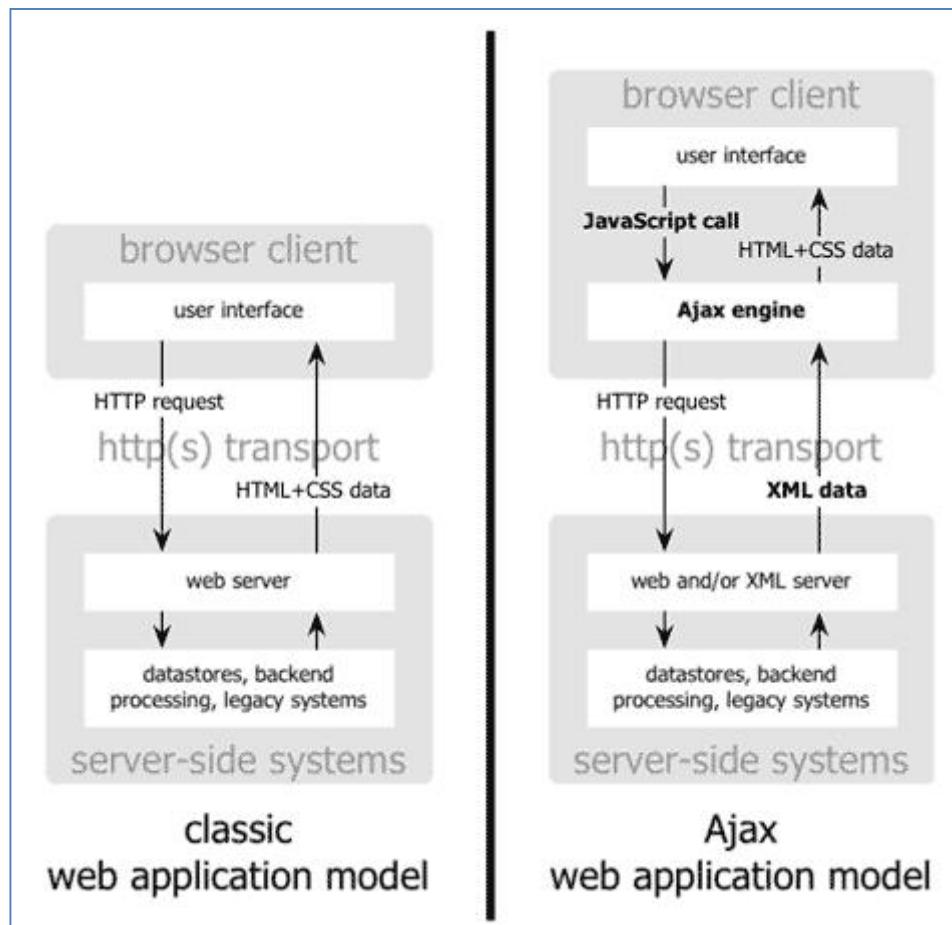
Khi đó công việc gửi request và nhận response đều do Ajax Engine thực hiện. Thay vì trả dữ liệu dưới dạng HTML và CSS trực tiếp cho trình duyệt, Web server có thể gửi trả dữ liệu dạng XML và Ajax Engine sẽ tiếp nhận, phân tích và chuyển đổi thành XHTML + CSS cho trình duyệt hiển thị. Việc phân tích và chuyển đổi này được thực hiện trên Client nên giảm tải rất nhiều cho Server, đồng thời User cảm thấy kết quả xử lý được hiển thị tức thì mà không cần nạp

lại toàn bộ trang. Mặt khác, sự kết hợp của các công nghệ web như CSS và XHTML làm cho việc trình bày giao diện trang web tốt hơn nhiều và giảm đáng kể dung lượng trang phải nạp. Đây là những lợi ích hết sức thiết thực mà Ajax đem lại.

Ajax Engine chỉ gửi đi những thông tin cần thay đổi chứ không phải toàn bộ trang web, do đó giảm được tải qua mạng.

Việc gửi request và nhận response do Ajax Engine thực hiện. Do đó phía Browser UI không rơi vào trạng thái chờ (waiting...), tức là có thể thực hiện nhiều việc cùng lúc (Asynchronous).

Có thể nhìn vào 2 hình sau đây để so sánh hai mô hình ứng dụng Web: truyền thống và sử dụng Ajax.



12.3 Các bước xây dựng một ứng dụng ajax

Sau đây là các bước cài đặt Ajax đơn giản, các bước như sau :

Phía Client :

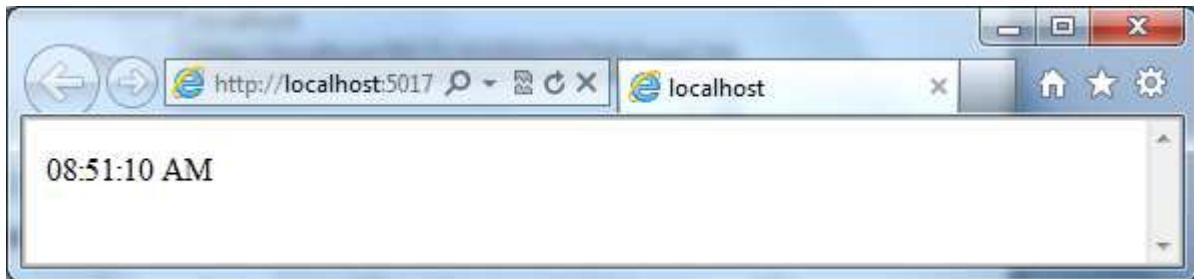
- ✓ Viết hàm tạo đối tượng XMLHttpRequest (đối tượng này dùng để gửi request đến Server).
- ✓ Viết hàm gửi yêu cầu (request) đến Server.
- ✓ Viết hàm xử lý thông tin sau khi gửi request đến Server. Có thể là thông tin lỗi trả về (nếu việc gửi request thất bại) hay là thông tin do Server trả lời lại.

Phía Server :

- ✓ Viết hàm xử lý các yêu cầu (request) từ Client gửi đến.

Ví dụ:

Ví dụ hiển thị thời gian hiện tại của server lên trang web. Với ví dụ này thì cứ 1 giây chúng ta phải truyền thông với server 1 lần để lấy thời gian hiện tại trên server để cập nhật lại thời gian trên trang web.



Mã nguồn phía client

```
<html>
<head>
<script>

/*--tạo đối tượng Request tùy vào trình duyệt--*/
function createXMLHttpRequest()
{
    try {
        return new XMLHttpRequest(); // for all browser
    }
    catch (e) {
        return new ActiveXObject("Microsoft.XMLHTTP"); // for IE9-
    }
}

/*--gửi yêu cầu đến server theo hình thức GET--*/
function callServer() {
    XHRObject = createXMLHttpRequest();
    if (XHRObject) {
        XHRObject.open("GET", "ServerClock.aspx?" + Math.random(), true);
        XHRObject.onreadystatechange = handleStateChanged;
        XHRObject.send(null);
    }
}

/*--xử lý kết quả phản hồi từ server--*/
function handleStateChanged()
{
    if (XHRObject.status == 200) {
        if (XHRObject.readyState == 4) {
            var result = XHRObject.responseText;
            document.getElementById("clock").innerHTML = result;
        }
    }
}
</script>
</head>
<body onload="setInterval('callServer()', 1000)">
<!--thẻ này dùng để hiển thị thời gian server-->
<span id="clock"></span>
</body>
</html>
```

Hàm setInterval('biểu thức javascript', thời gian miligiây) dùng để thực hiện một biểu thức JavaScript một cách định kỳ. Trong bài này, cứ 1 giây ta gọi hàm callServer() một lần để cập nhật thời gian server.

Dòng mã lệnh XMLHttpRequest.open("GET", "ServerClock.aspx?" + Math.random(), true); trong hàm callServer() sẽ truyền thông với trang ASP.NET đặt trên server để lấy thời gian hiện tại của server.

Mã nguồn phía server

- ✓ Mã ASP.NET

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

- ✓ Mã C# code behind

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
```

```
public partial class ServerClock : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        // lấy thời gian hệ thống
        String time = DateTime.Now.ToString("hh:mm:ss tt");

        // gửi trả về client
        Response.Write(time);
    }
}
```

Ý nghĩa các tham số của phương thức XMLHttpRequest.open(method, server, async)

Tham số	Mô tả
Method	Có 2 giá trị là POST và GET qui định hình thức vận chuyển dữ liệu trên web đến server
Server	Trang web nhận và xử lý tham số sau đó trả kết quả về client thông qua thuộc tính XMLHttpRequest.responseText.
Async	Qui định cơ chế truyền thông (true: bắt đồng bộ, false: đồng bộ)

12.4 ASP.NET Ajax

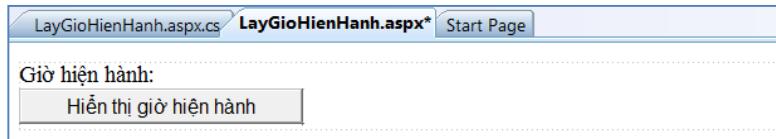
12.4.1 ASP.NET ajax và cơ chế gọi ngược

Trong phần trước, chúng ta đã biết để xây dựng một ứng dụng ASP.Net Ajax thông qua XMLHttpRequest thì chúng ta phải xây dựng 2 trang ASP.Net và viết mã javascript rất nhiều. ASP CallBack hỗ trợ xây dựng ứng dụng ASP.Net Ajax một cách đơn giản hơn (chỉ viết trong 1 trang ASP.Net) bằng kỹ thuật CallBack. Để sử dụng được ASP Callback trong trang ASP.Net chúng ta phải thực thi các phương thức **RaiseCallbackEvent** và **GetCallbackResult** của giao diện **IcallbackEventHandler**.

Ví dụ 2:

Trang LayGioHienHanh.aspx có giao diện như sau. Sau khi nhấp nút thì thời gian server sẽ được hiện ra ngay sau chữ Giờ hiện hành:

Bước 1 : Thiết kế giao diện



Bước 2 : Viết mã javascript trong phần HTML :

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>lấy giờ hiện hành với ASP CallBack</title>
</head>
<script type="text/ecmascript">
    function ReceiveServerData(rValue)
    {
        document.getElementById("HienThi").innerHTML = rValue;
    }
</script>
<body>
    <form id="form1" runat="server">
        <div>
            Giờ hiện hành:&ampnbsp <span id="HienThi" runat="server"></span><br>
            <input id="btHienThi" type="button"
                value="Hiển thị giờ hiện hành" onclick="CallServer();"/>
        </div>
    </form>
</body>
</html>
```

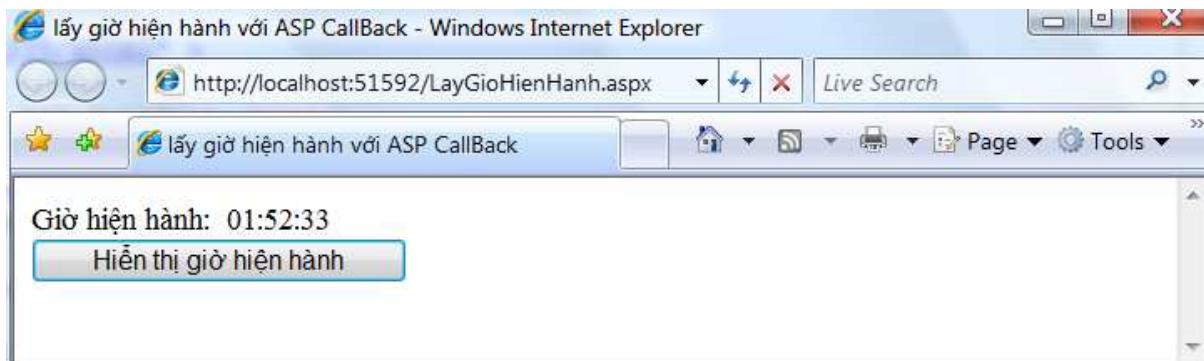
Bước 3 : Viết lệnh cho phía Sever trong tập tin LayGioHienHanh.aspx.cs

```
namespace MinhHoa
{
    public partial class LayGioHienHanh : System.Web.UI.Page, ICallbackEventHandler
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            //Đăng ký các lệnh CallBack
            String cbReference =
                Page.ClientScript.GetCallbackEventReference(this,
                "", "ReceiveServerData", "");
            String callbackScript;
            callbackScript = "function CallServer() " +
            "{ " + cbReference + ";"};
            Page.ClientScript.RegisterClientScriptBlock(this.GetType(),
            "CallServer", callbackScript, true);
        }

        //Phuong thức này nhận dữ liệu được gửi từ Client
        public void RaiseCallbackEvent(String eventArgument)
        {
            //
        }

        //Phuong thức này trả kết quả về cho Client
        public String GetCallbackResult()
        {
            //Trả về giờ hiện hành cho Client
            return DateTime.Now.ToString("hh:mm:ss");
        }
    }
}
```

Bước 4. Nhấn **Ctrl+F5** hoặc **View in Browser** để thi hành. Nhấn nút **Hiển thị giờ hiện hành** kết quả như hình sau.



12.4.2 ASP.NET Ajax Server Control

ASP.Net Ajax server controls bao gồm các mã của server và client tương tác với nhau để xử lý các yêu cầu từ client. Khi các bạn thêm các Ajax control vào trong trang ASP.Net nó sẽ tự động gửi các script đến trình duyệt để thực hiện các chức năng Ajax. Chúng ta có thể cung cấp thêm các đoạn mã ở phía client để tùy biến các chức năng cho Ajax control.



ScriptManager control : Quản lý các script cho các thành phần client , partial-page rendering, localization, globalization và scripts của người dùng. Control này được yêu cầu sử dụng cho các controls : UpdatePanel, UpdateProgress và Timer controls.

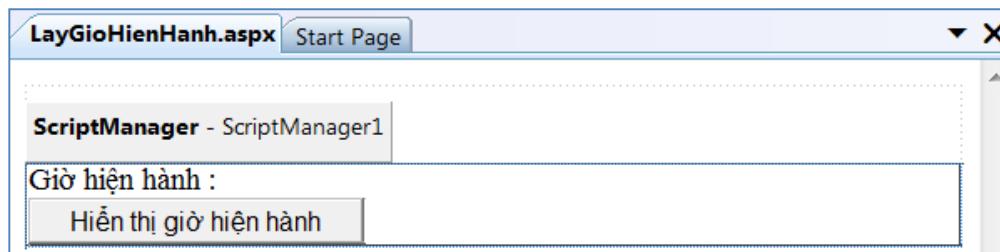
UpdatePanel : cho phép các bạn refresh một phần của trang ASP.Net thay vì refresh toàn bộ trang sử dụng postback.

UpdateProgress : Cung cấp thông tin các trạng thái về partial-page update trong UpdatePanel control.

Timer : thực hiện việc postback trong một khoảng thời gian được cho trước. Chúng ta có thể sử dụng control này để post toàn bộ trang hoặc sử dụng cùng với Update Panel control thực hiện partial-page update (chỉ một phần nào đó trên trang ASP.Net được thay đổi) với một khoảng thời gian chỉ định.

Ví dụ 3. Trang LayGioHienHanh.aspx trong ví dụ 2, chúng ta viết lại theo ASP.Net ajax server control như sau :

Bước 1: Thiết kế giao diện



Bước 2: Thiết lập phần mã ASP.NET

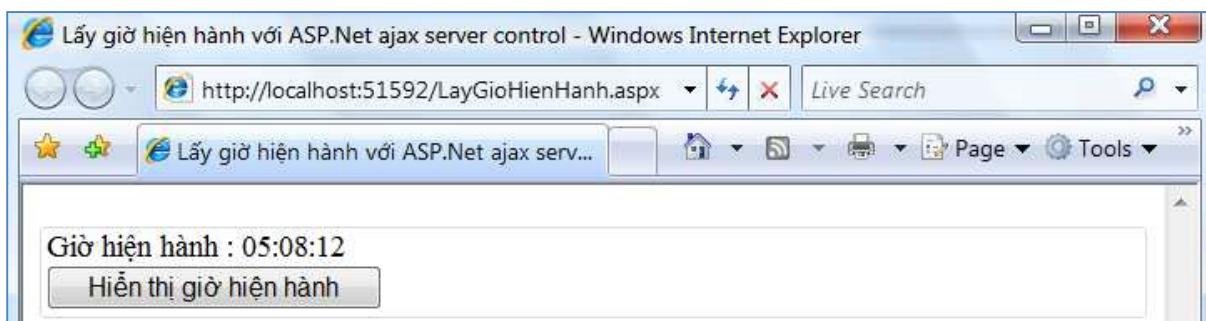
```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Lấy giờ hiện hành với ASP.Net ajax server control</title>
</head>
<body>
    <form id="form1" runat="server">
        <div style="padding-top: 10px">
            <asp:ScriptManager ID="ScriptManager1" runat="server">
            </asp:ScriptManager>
            <asp:UpdatePanel ID="UpdatePanel1" runat="server" RenderMode="Inline">
                <ContentTemplate>
                    <fieldset>
                        <asp:Label ID="lbGioHienHanh" runat="server" Text="Giờ hiện hành :"></asp:Label>
                        <br />
                        <asp:Button ID="btnLayGioHienHanh" runat="server"
                            OnClick="btnLayGioHienHanh_Click" Text="Hiển thị giờ hiện hành" Width="201px" />
                    </fieldset>
                </ContentTemplate>
            </asp:UpdatePanel>
            <br />
        </div>
    </form>
</body>
</html>
```

Bước 3 : Viết lệnh xử lý sự kiện

```
public partial class LayGioHienHanh : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void btnLayGioHienHanh_Click(object sender, EventArgs e)
    {
        lbGioHienHanh.Text = "Giờ hiện hành : " + DateTime.Now.ToString("hh:mm:ss");
    }
}
```

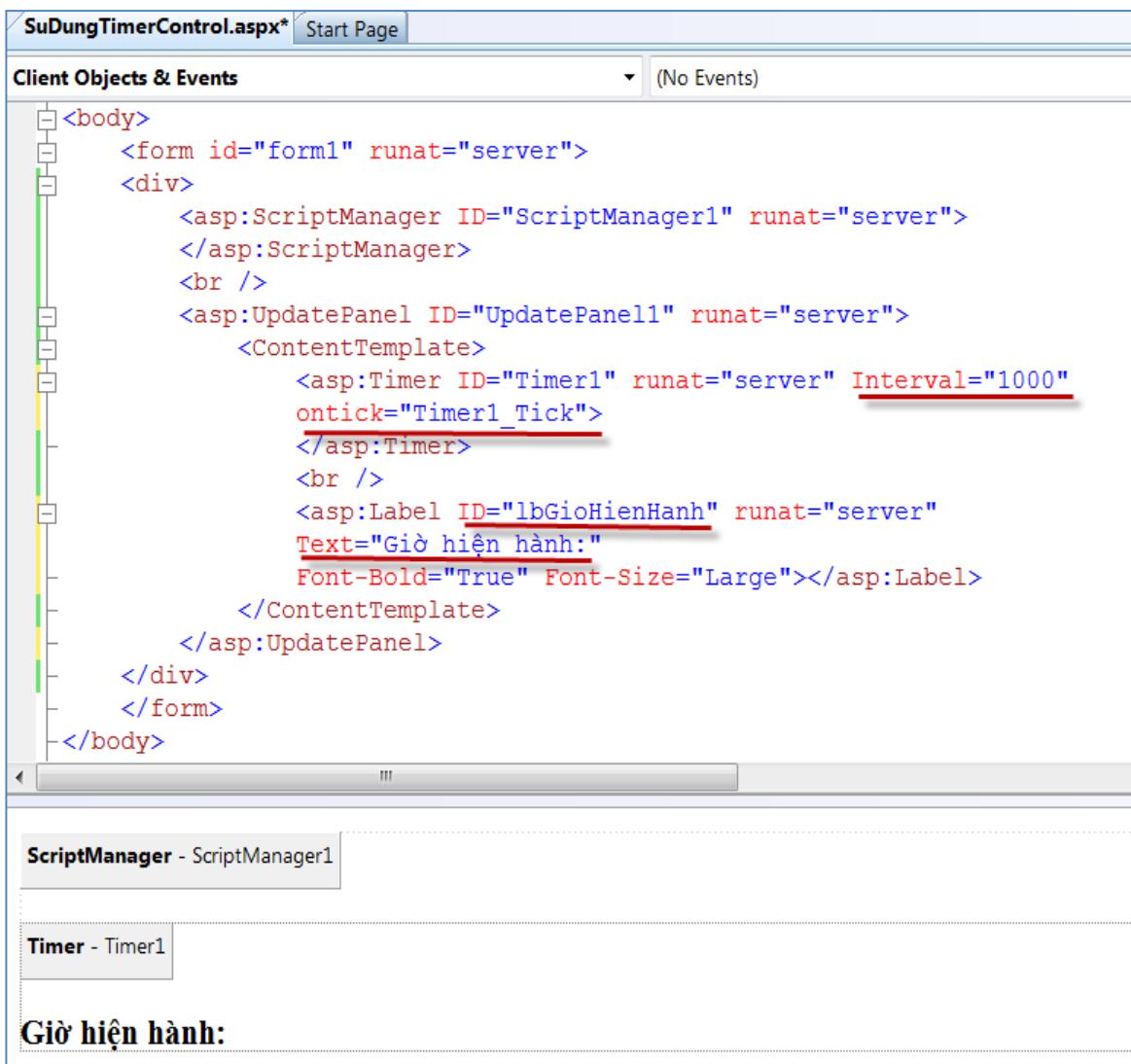
Bước 4. Nhấn **Ctrl+F5** hoặc **View in Browser** để thi hành. Nhấn nút **Hiển thị giờ hiện hành**

Kết quả hiển thị:



Ví dụ 4. Tạo trang **SuDungTimerControl.aspx** trong ví dụ 1, 2, 3 chúng ta viết lại theo ASP.Net ajax server control hiển thị đồng hồ hiện hành từ server sử dụng Timer Control

Bước 1: Thiết kế giao diện



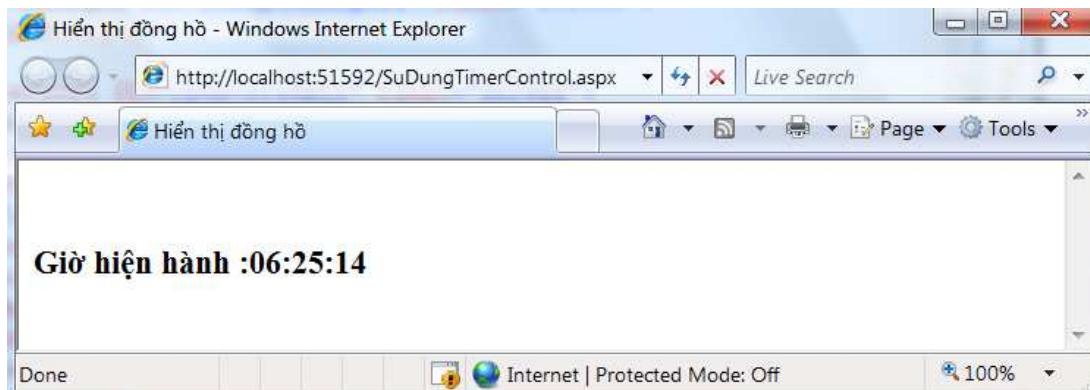
Bước 2. Viết lệnh xử lý sự kiện

```

public partial class SudungTimerControl : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    //Phương thức này mỗi giây sẽ được gọi 1 lần
    protected void Timer1_Tick(object sender, EventArgs e)
    {
        lbGioHienHanh.Text = "Giờ hiện hành :" +
            DateTime.Now.ToString("hh:mm:ss"); ;
    }
}

```

Bước 3. Nhấn **Ctrl+F5** hoặc **View in Browser** để thi hành. Các bạn sẽ thấy giờ hiện hành được cập nhật liên tục mỗi giây 1 lần.



12.5 Giới thiệu Ajax Control Toolkit

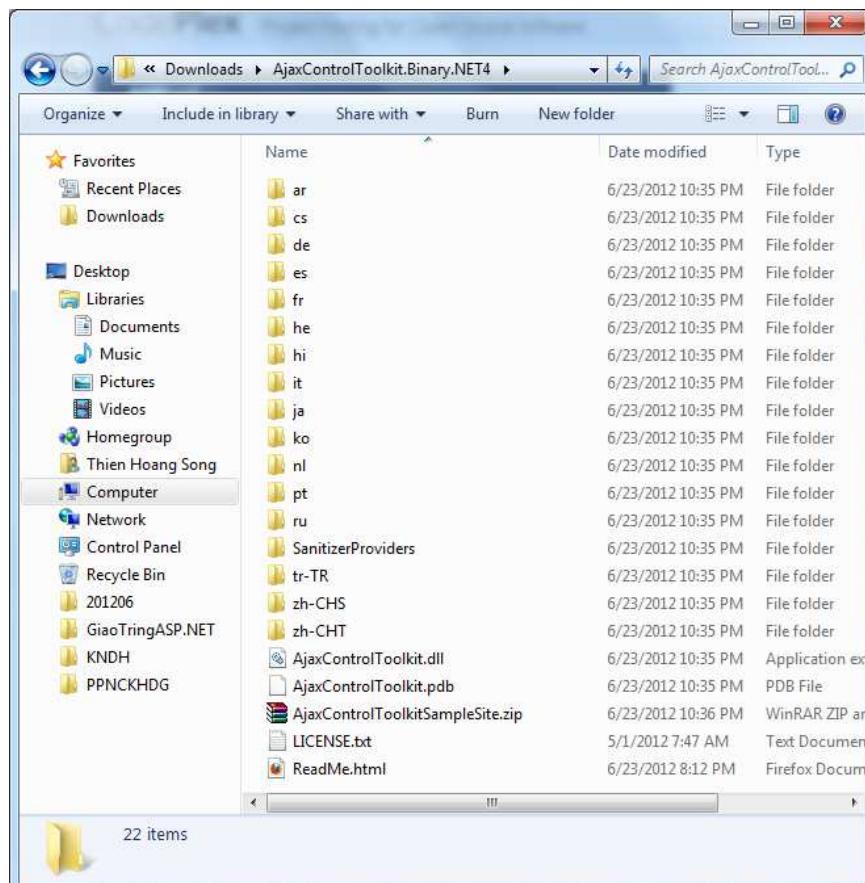
Ajax Control Toolkit chứa một tập phong phú các điều khiển mà bạn có thể sử dụng để xây dựng các ứng dụng ASP.NET Web Forms cho phép đáp ứng và tương tác cao với Ajax. Thực hiện theo các bước dưới đây để tải về và bắt đầu sử dụng Ajax Control Toolkit với Visual Studio:

12.5.1 Tải Ajax Control Toolkit

Download thư viện “Ajax Control Toolkit. NET 4” tại :

<http://ajaxcontroltoolkit.codeplex.com/downloads/get/404525>

Giải nén tập tin đã tải về bạn sẽ có cấu trúc thư mục như sau:



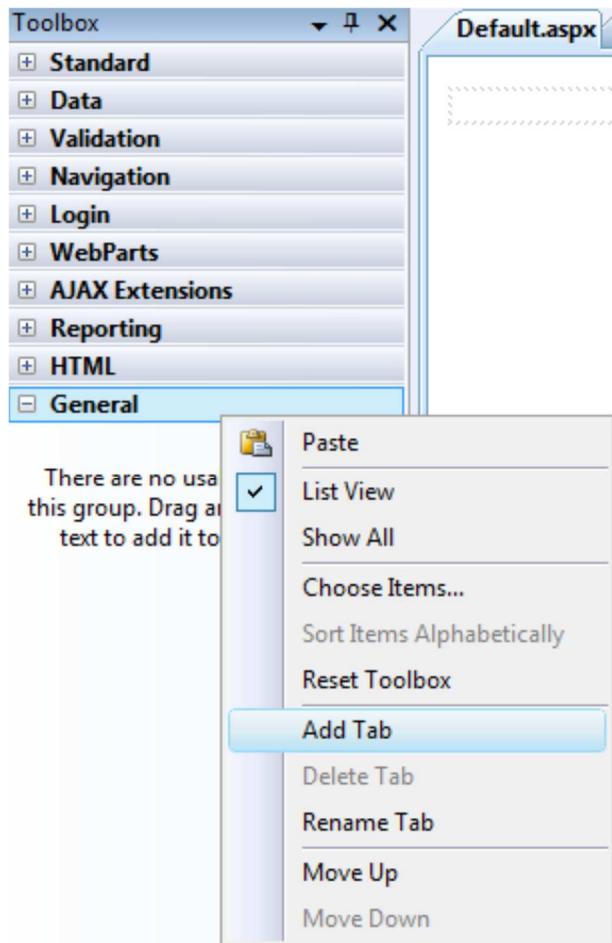
Để ý bạn sẽ thấy có gói AjaxControlToolkitSampleSite.zip, đây là toàn bộ các thí dụ đi kèm với gói Ajax Control Toolkit để bạn có thể tìm hiểu toàn bộ các Ajax controls qua các bài tập tương ứng.

12.5.2 Thêm Ajax Control Toolkit vào VS Toolbox

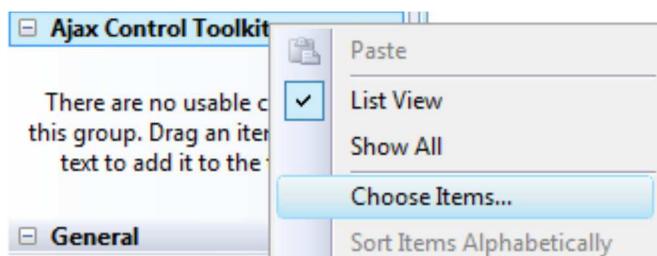
Thực hiện theo các bước sau:

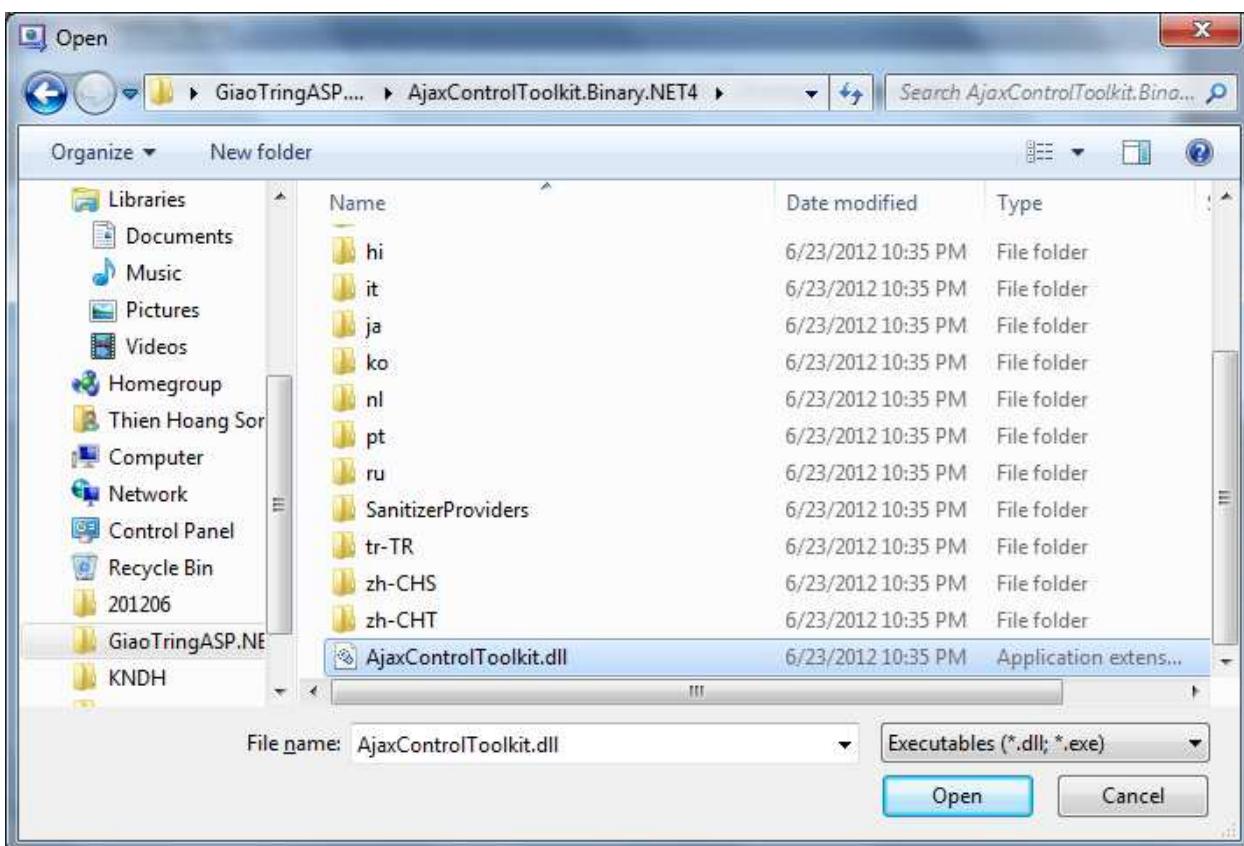
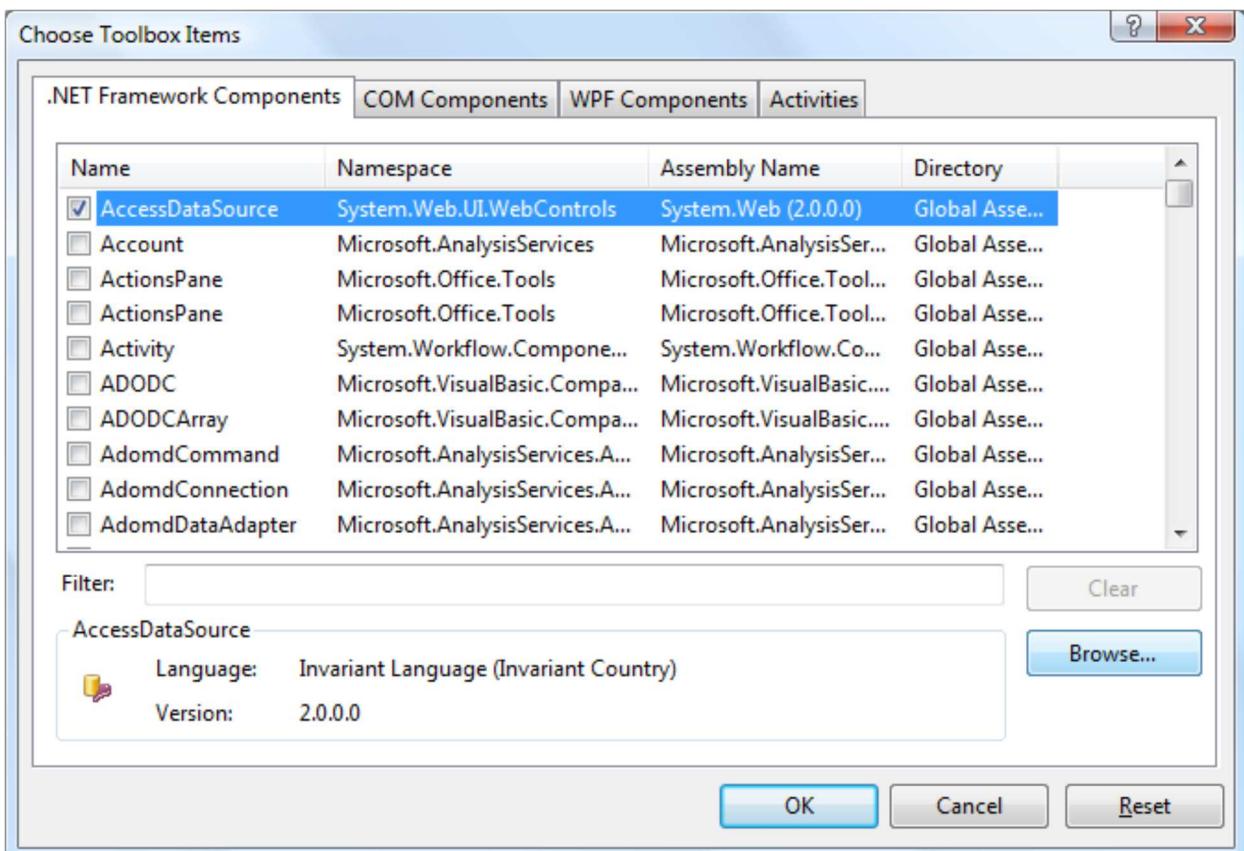
Bước 1: Khởi động Visual Studio và tạo ra một website ASP.NET mới. Mở trang Default.aspx trong trình soạn thảo Visual Studio.

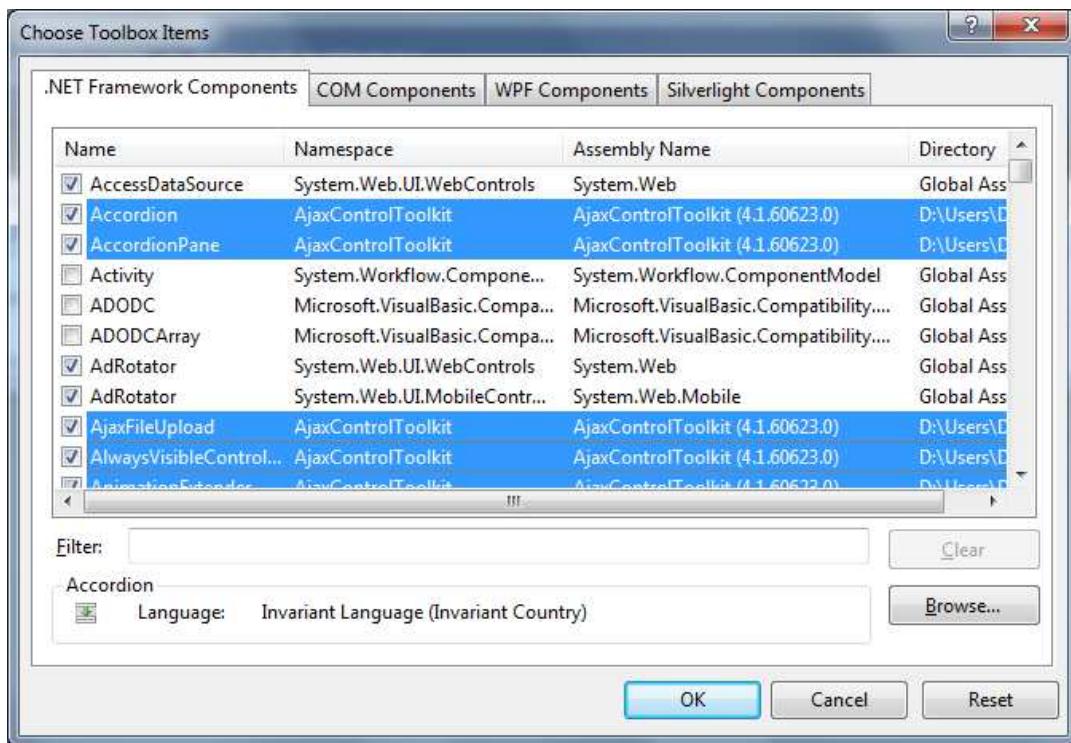
Bước 2: Mở Hộp công cụ (toolbox) và tạo ra một tab bằng cách nhấp chuột phải và chọn mục Add Tab. Tên tab mới là Ajax Control Toolkit.



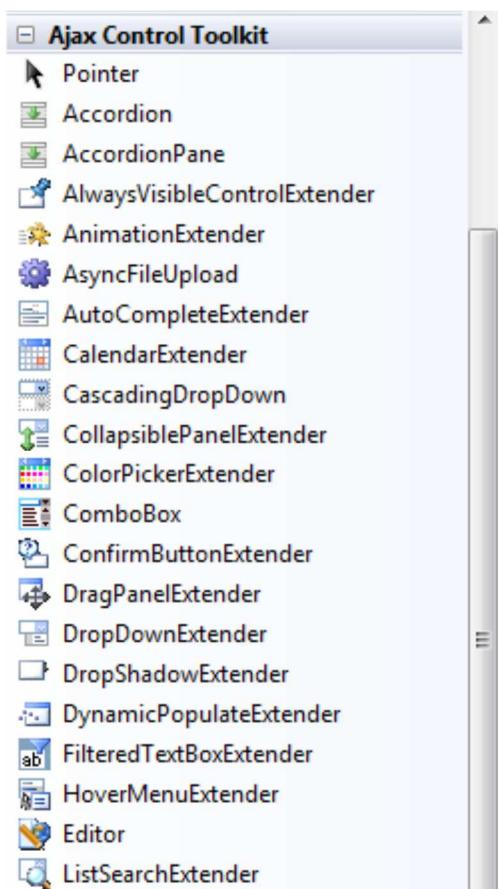
Bước 3: Nhấp chuột phải vào bên dưới tab mới và chọn Choose Items..., xuất hiện hộp thoại Choose Toolbox Items , nhấp vào nút Browse và duyệt đến thư mục mà bạn giải nén gói Ajax Control Toolkit. Chọn tập tin AjaxControlToolkit.dll và nhấp vào nút Open để chọn các Ajax Controls đưa vào và chuyển về lại hộp thoại Choose Toolbox Items, nhấp nút OK để hoàn tất.





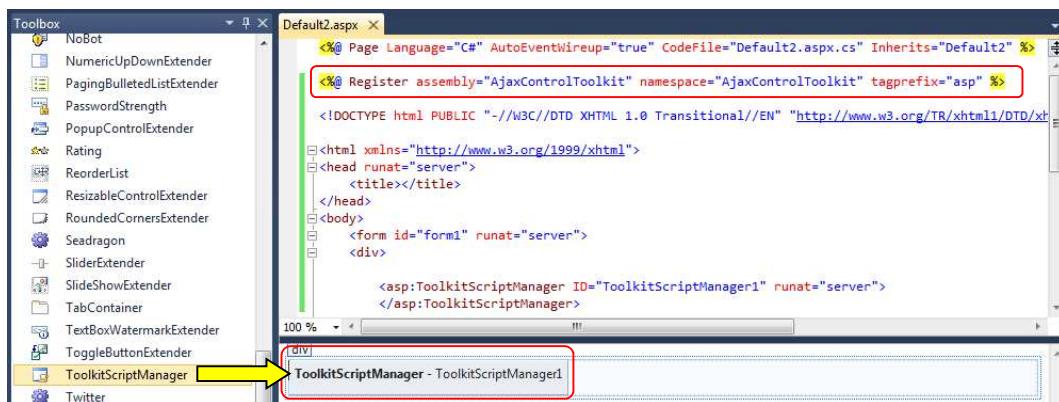


Lúc này, trong cửa sổ Toolbox, tab Ajax Control Toolkit vừa mới thêm vào đã có các Ajax controls trong đó như hình:



12.5.3 Một số điều khiển trong Ajax Control Toolkit

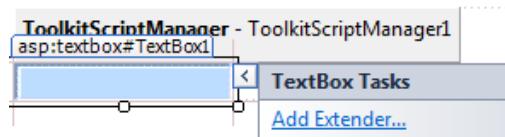
Trước khi làm việc với các điều khiển của Ajax Toolkit bạn phải nhúng ToolkitScriptManager vào trang web hoặc MasterPage chứa trang web đó.



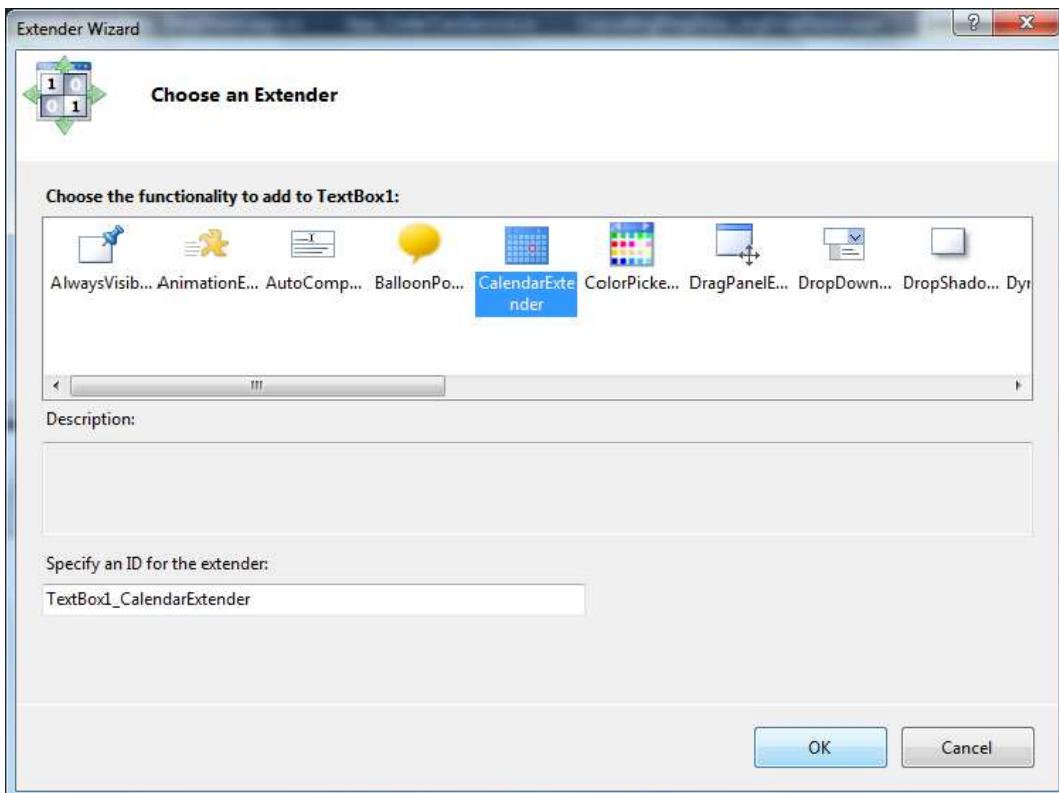
Ajax Toolkit chứa một số control mới và một số thành phần mở rộng (extender) được sử dụng để gắn với các control có sẵn của ASP.NET.

Đối với các control mới, bạn chỉ việc kéo chúng vào và sử dụng như các control thông thường.

Còn đối với các extender bạn phải kéo một control ASP.NET nào đó trước, sau đó bấm chọn vào Add Extender vào góc trên bên phải của control đó để bổ sung phần mở rộng cho control đó.



Một hộp thoại xuất hiện để bạn lựa chọn phần mở rộng cần thiết



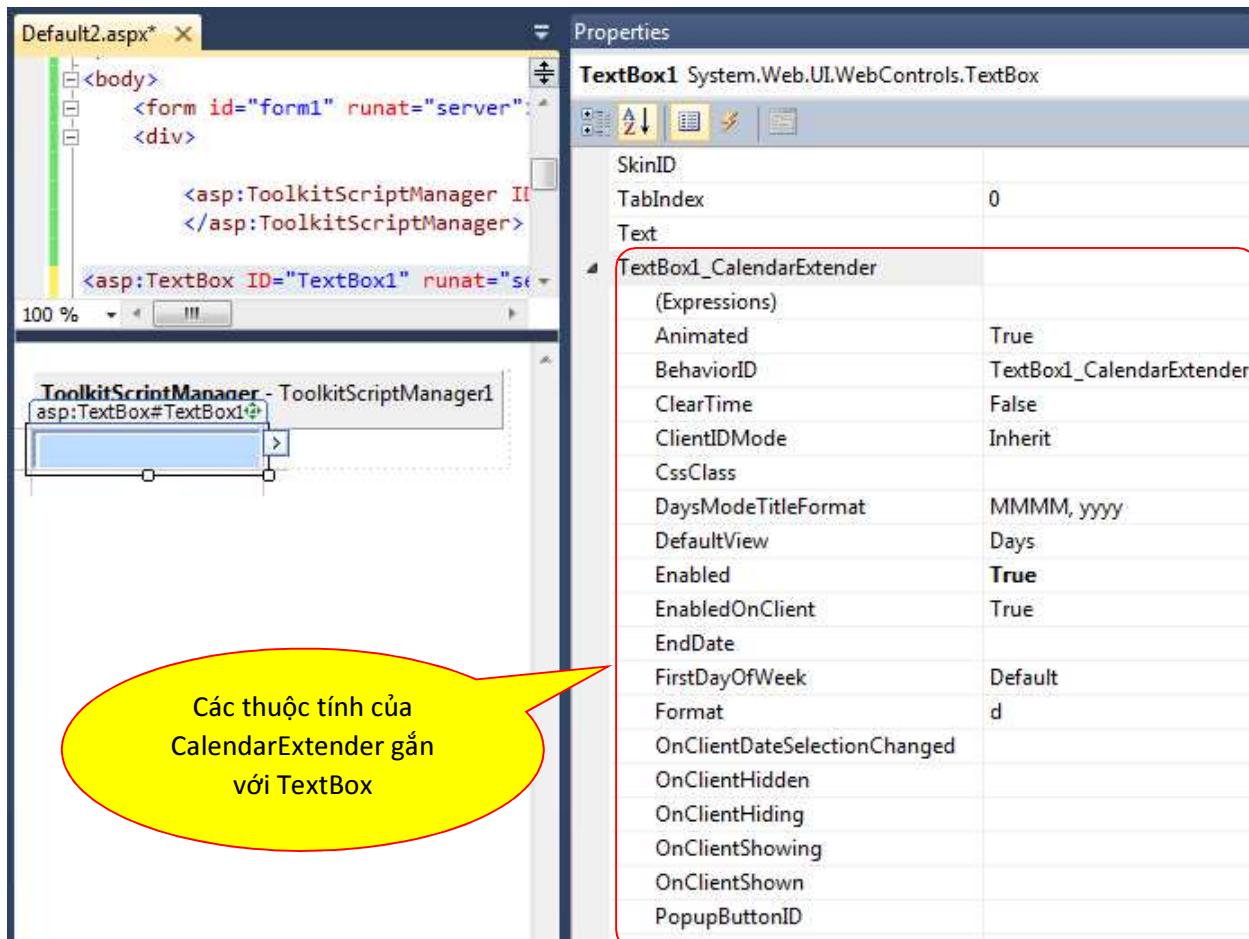
Sau khi chọn extender và nhấp OK bạn sẽ gắn được một đoạn mã ASP.NET thể hiện sự gắn kết giữa control và thành phần mở rộng qua thuộc tính TargetControlID.

Ví dụ sau gắn Calendar Extender với TextBox:

```
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

<asp:CalendarExtender ID="TextBox1_CalendarExtender" runat="server"
    Enabled="True" TargetControlID="TextBox1">
</asp:CalendarExtender>
```

Đồng thời một danh sách thuộc tính của thành phần mở rộng được bổ sung vào danh sách thuộc tính của control để bạn dễ dàng thiết lập các giá trị thuộc tính cho extender.



12.5.3.1 Accordion Control.

1. Accordion

2. AutoSize head

3. Control or Extender content

The Accordion is written using an extender like most of the other extenders in the AJAX Control Toolkit. The extender expects its input in a very specific hierarchy of container elements (like divs), so the Accordion and AccordionPane web controls are used to generate the expected input for the extender. The extender can also be used on its own if you provide it appropriate input.

4. What is ASP.NET AJAX?

Accordion control giúp bạn định nghĩa nhiều lớp và trình bày chúng từng cái một. Nó giống như là có nhiều CollapsiblePanel controls mà chỉ có một có thể mở rộng tại một thời điểm xác định. Accordion control chứa một hoặc nhiều AccordionPane controls. Mỗi AccordionPane control có chứa phần header và phần content của nó.

Accordion control hỗ trợ chế độ AutoSize sau để tùy biến với nhiều cỡ trang khác nhau:

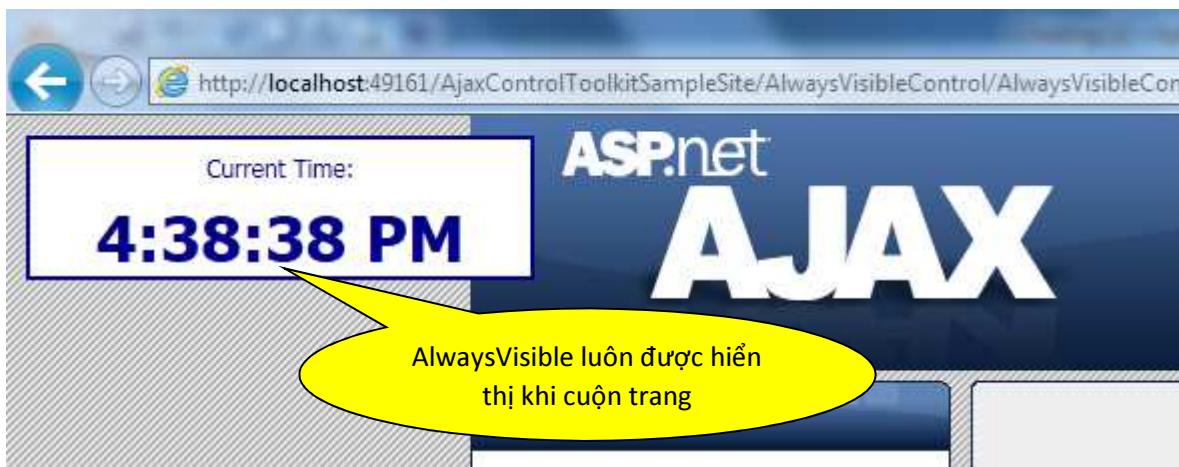
- ✓ None: Accordion control mở rộng và thu hẹp không hạn chế.
- ✓ Limit: Accordion control mở rộng và thu hẹp không quá giá trị của thuộc tính Height.
- ✓ Fill: Accordion control bị cố định bởi giá trị của thuộc tính Height

Accordion cũng có thể là data-bound ,để gắn dữ liệu vào control, chỉ ra nguồn dữ liệu sử dụng thuộc tính Datasource hoặc DataSourceID,và sau đó đặt các gói dữ liệu vào thuộc tính HeaderTemplate và ContentTemplate.Bạn phải gọi phương thức DataBind để gắn dữ liệu vào điều khiển.

```
<asp:Accordion
    ID="MyAccordion"
    runat="Server"
    SelectedIndex="0"
    HeaderCssClass="accordionHeader"
    HeaderSelectedCssClass="accordionHeaderSelected"
    ContentCssClass="accordionContent"
    AutoSize="None"
    FadeTransitions="true"
    TransitionDuration="250"
    FramesPerSecond="40"
    RequireOpenedPane="false"
    SuppressHeaderPostbacks="true">
    <Panes>
        <asp:AccordionPane
            HeaderCssClass="accordionHeader"
            HeaderSelectedCssClass="accordionHeaderSelected"
            ContentCssClass="accordionContent">
            <Header>... </Header>
            <Content>... </Content>
        </asp:AccordionPane>
    </Panes>
    <HeaderTemplate>...</HeaderTemplate>
    <ContentTemplate>...</ContentTemplate>
</asp:Accordion>
```

12.5.3.2 AlwaysVisible Control.

Là một extender (phần mở rộng) được dùng để hiển thị một cách liên tục đến một điều khiển ASP.NET. Điều khiển được mở rộng sẽ luôn luôn di chuyển đến một vị trí cố định trên trang bất kể trang thị thay đổi kích thước hay bị cuộn.

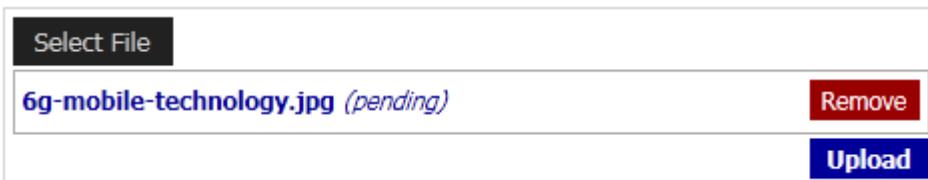


```
<asp:Panel ID="Panel1" runat="server">...</asp:Panel>

<asp:AlwaysVisibleControlExtender ID="ace" runat="server"
    TargetControlID="Panel1"
    VerticalSide="Top"
    VerticalOffset="10"
    HorizontalSide="Right"
    HorizontalOffset="10"
    ScrollEffectDuration=".1"/>
```

12.5.3.3 AsyncFileUpload Control.

AsyncFileUpload cho phép bạn upload file lên server một cách bất đồng bộ, kết quả của file upload có thể được kiểm tra cả ở phía server lẫn client, bạn có thể lưu file được upload bằng cách gọi phương thức SaveAs() trong điều khiển sự kiện UploadedComplete ở server.



```
<asp:AjaxFileUpload ID="AjaxFileUpload1" ThrobberID="myThrobber" ContextKeys="fred" AllowedFileTypes="jpg,jpeg"
    MaximumNumberOfFiles=10 runat="server"/>
```

Thuộc tính và sự kiện

Thành viên	Mô tả
UploadedComplete (e)	Xảy ra phía server khi upload thành công
ThrobberID	ID của điều khiển hiển thị trong quá trình upload
ContextKeys	Thông số phân loại upload được sử dụng ở phía server
MaximumNumberOfFiles	Số lượng tập tin tối đa được phép upload
AllowedFileTypes	Các kiểu tập tin cho phép upload
OnClientUploadComplete	Hàm javascript xử lý upload thành công phía client
OnClientUploadError	Hàm javascript xử lý upload lỗi phía client

12.5.3.4 AutoComplete Control

Phần mở rộng thường được gắn với TextBox để cung cấp gợi ý dữ liệu gần đúng để hoàn tất phần nhập nhanh chóng.



```
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
<asp:AutoCompleteExtender runat="server" ID="autoComplete1" TargetControlID="TextBox1"
    ServiceMethod="GetCompletionList" ServicePath="AutoComplete.asmx" MinimumPrefixLength="2"
    CompletionInterval="1000" EnableCaching="true" CompletionSetCount="20"
    CompletionListCssClass="autocomplete_completionListElement"
    CompletionListItemCssClass="autocomplete_listItem"
    CompletionListHighlightedItemCssClass="autocomplete_highlightedListItem"
    DelimiterCharacters=";, :" ShowOnlyCurrentWordInCompletionListItem="true">
    <Animations>
        <OnShow>... </OnShow>
        <OnHide>... </OnHide>
    </Animations>
</asp:AutoCompleteExtender>
```

Web Service AutoComplete.asmx

```
using System;
using System.Collections.Generic;
using System.Web.Services;

[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo = WsProfiles.BasicProfile1_1)]
[System.Web.Script.Services.ScriptService]
public class AutoComplete : WebService
{
    [WebMethod]
    public string[] GetCompletionList(string prefixText, int count)
    {
        if (count == 0){
            count = 10;
        }

        if (prefixText.Equals("xyz")){
            return new string[0];
        }

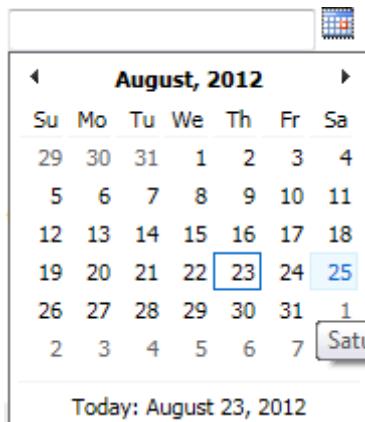
        Random random = new Random();
        List<string> items = new List<string>(count);
        for (int i = 0; i < count; i++)
        {
            char c1 = (char) random.Next(65, 90);
            char c2 = (char) random.Next(97, 122);
            char c3 = (char) random.Next(97, 122);
            items.Add(prefixText + c1 + c2 + c3);
        }

        return items.ToArray();
    }
}
```

}

12.5.3.5 Calender Control.

Điều khiển Calendar extender có thể được gắn với bất kì điều khiển textbox nào của ASP.NET. Nó cung cấp hàm định dạng ngày phía client với định dạng ngày tùy biến với giao diện trong một điều khiển popup. Bạn có thể tương tác với calendar bằng cách click vào một ngày để chọn, hoặc Today để chọn ngày hiện tại. Ngoài ra, mũi tên trái phải có thể được sử dụng để chuyển tới tháng trước hoặc tháng sau. Bằng cách click vào tiêu đề của calendar bạn có thể thay đổi hiển thị của các ngày trong tháng hiện tại thành các tháng trong năm hiện tại. Click khác sẽ chuyển các năm trong thập kỷ hiện tại. Hành động này cho phép bạn dễ dàng chuyển đến một ngày trong quá khứ hoặc tương lai từ điều khiển calendar.

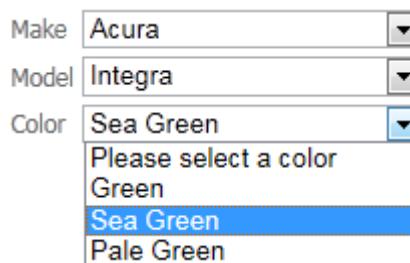


```
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
<asp:Image ImageUrl="cal.gif" ID="Image1" runat="server" />

<asp:Calendar runat="server" TargetControlID="TextBox1" Format="MMMM d, yyyy" PopupButtonID="Image1" />
```

12.5.3.6 CascadingDropDown Control.

CascadingDropdown được dùng để mở rộng tính năng tải dữ liệu dây chuyền cho các DropDownList. Hình ảnh sau cho thấy nội dung của DropDownList sau sẽ phụ thuộc vào mục chọn của cái kế trước.



```
<asp:DropDownList ID="DropDownList1" runat="server"></asp:DropDownList>
<asp:DropDownList ID="DropDownList2" runat="server"></asp:DropDownList>
<asp:DropDownList ID="DropDownList3" runat="server"></asp:DropDownList>

<asp:CascadingDropDown ID="CascadingDropDown1" runat="server" TargetControlID="DropDownList1"
    Category="Make" PromptText="Please select a make" LoadingText="[Loading makes...]"
    ServicePath="CarsService.asmx" ServiceMethod="GetDropDownContents" />

<asp:CascadingDropDown ID="CascadingDropDown2" runat="server" TargetControlID="DropDownList2"
    Category="Model" PromptText="Please select a model" LoadingText="[Loading models...]"
    ServiceMethod="GetDropDownContentsPageMethod" ParentControlID="DropDownList1" />

<asp:CascadingDropDown ID="CascadingDropDown3" runat="server" TargetControlID="DropDownList3"
    Category="Color" PromptText="Please select a color" LoadingText="[Loading colors...]"
    ServicePath="~/CascadingDropDown/CarsService.asmx" ServiceMethod="GetDropDownContents"
    ParentControlID="DropDownList2" />
```

Dữ liệu được cấp cho DropDownList1 và DropDownList3 được lấy từ phương thức GetDropDownContents() của web service CarsService.asmx. Thuộc tính Category được sử dụng để phân biệt loại dữ liệu.

```
using System;
using System.Collections.Specialized;
using System.Text.RegularExpressions;
using System.Web;
using System.Web.Services;
using System.Xml;
using AjaxControlToolkit;

[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
[System.Web.Script.Services.ScriptService]
public class CarsService : WebService
{
    [WebMethod]
    public CascadingDropDownNameValue[] GetDropDownContents(string knownCategoryValues, string category)
    {
        ...
    }
}
```

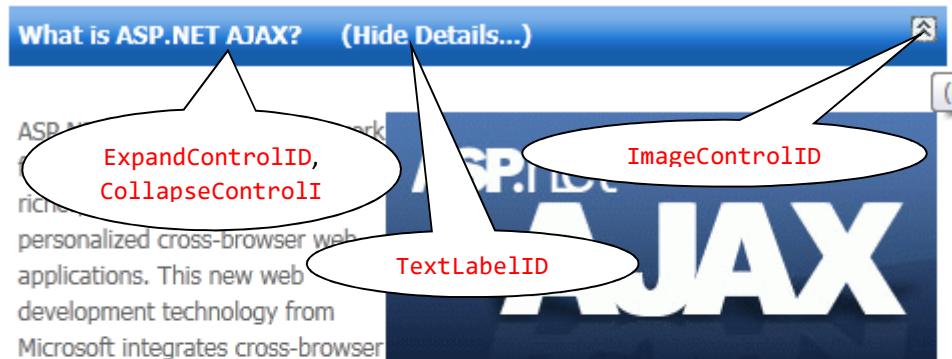
Dữ liệu được cấp cho DropDownList2 được lấy từ phương thức **GetDropDownContentsPageMethod()** của được định nghĩa trong trang web hiện tại.

```
using System;
using System.Web.Services;
using AjaxControlToolkit;

public partial class CascadingDropDown_CascadingDropDown : CommonPage
{
    [WebMethod]
    [System.Web.Script.Services.ScriptMethod]
    public static CascadingDropDownNameValue[] GetDropDownContentsPageMethod(string knownCategoryValues, string
category)
    {
        ...
    }
}
```

12.5.3.7 CollapsiblePanel Control

Một lớp mở rộng cho phép thêm hành vi mở rộng hay thu hẹp một điều khiển ASP.NET. Phần nội dung được gắn mở rộng sau đó có thể mở rộng ra hoặc thu hẹp lại bởi người dùng, một cách thủ công như hiển thị hoặc giấu nội dung hay mở rộng toàn bộ khoảng trống hiện có.



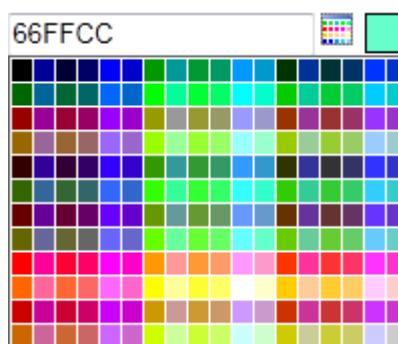
client script libraries with the ASP.NET 2.0 server-based development framework. In addition, ASP.NET AJAX offers you the same type of development platform for client-based web pages that ASP.NET offers for server-based pages. And because ASP.NET AJAX is an extension of ASP.NET, it is fully integrated with server-based services. ASP.NET AJAX makes it possible to easily take advantage of AJAX techniques on the web and enables you to create ASP.NET pages with a rich, responsive UI and server communication. However, AJAX isn't just for ASP.NET. You can take advantage of the rich client framework to easily build client-centric web applications that integrate with any backend data provider and run on most modern browsers.

```
<asp:Panel ID="Panel2" runat="server">
<asp:Image ID="Image1" runat="server" />
<asp:LinkButton ID="LinkButton1" runat="server">LinkButton</asp:LinkButton>
<asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
</asp:Panel>
<asp:Panel ID="Panel1" runat="server">...</asp:Panel>

<asp:CollapsiblePanelExtender ID="cpe" runat="Server"
    TargetControlID="Panel1" CollapsedSize="0" ExpandedSize="300" Collapsed="True" ExpandControlID="LinkButton1"
    CollapseControlID="LinkButton1" AutoCollapse="False" AutoExpand="False" ScrollContents="True"
    TextLabelID="Label1" CollapsedText="Show Details..." ExpandedText="Hide Details"
    ImageControlID="Image1" ExpandedImage "~/images/collapse.jpg"
    CollapsedImage "~/images/expand.jpg" ExpandDirection="Vertical" />
```

12.5.3.8 ColorPicker Control

Phần mở rộng ColorPicker cho phép bạn hiển thị một bảng pop-up màu khi con trỏ di chuyển đến phần tử input. Bạn có thể gắn ColorPicker vào bất cứ textbox nào của ASP.NET. Nó cung cấp một hàm cho phép lựa chọn màu ở phía client với giao diện người dùng. Ngoài ra, bạn có thể chỉ ra một button để hiển thị một popup lựa chọn màu và một điều khiển cho phép xem trước màu từ bảng màu. Bạn có thể cung cấp một textbox để khi mà người dùng nhập vào một giá trị màu thì ColorPicker có thể hiện thị màu tương ứng nếu màu đó không có trong bảng màu mặc định.



Có 2 cách dùng

Cách 1: sử dụng TextBox để chứa mã màu chọn được

```
<asp:TextBox ID="Color1" runat="server"></asp:TextBox>
<asp:Image ID="Image1" ImageUrl="color.gif" runat="server" />
<asp:Label ID="Sample1" runat="server" Text=""></asp:Label>
<asp:ColorPickerExtender runat="server" TargetControlID="Color1" PopupButtonID="Image1"
    SampleControlID="Sample1" SelectedColor="#33ffcc" />
```

Cách 2: sử dụng hàm JavaScript để xử lý màu nhận được

```
<asp:TextBox ID="Color1" runat="server"></asp:TextBox>
<asp:ColorPickerExtender runat="server" ID="ColorPickerExtender1"
    TargetControlID="Color1" OnClientColorSelectionChanged="colorChanged" />
<script>
function colorChanged(sender) {
    var selectedColor = "#" + sender.get_selectedColor();
}
</script>
```

12.5.3.9 ComboBox Control

ComboBox là một điều khiển của ASP.NET AJAX, nó giống như AutoCompleteExtender, kết hợp linh hoạt với một Textbox với một danh sách tùy chọn cho phép người dùng có thể lựa chọn. Nó có các thuộc tính, hành vi hay quy ước đặt tên tương tự như trên Form combobox, và có cùng lớp cơ sở như là ListBox, BulletedList và DropDownList của web control. Thực vậy, ComboBox được xem như là DropDownList nhưng lại có thể gõ trực tiếp vào như là textbox.

Làm việc với ComboBox cũng giống như làm việc với DropDownList. Nó có cùng tất cả các thuộc tính và sự kiện của DropDownList, với một vài thuộc tính và sự kiện khác nữa. Đầu tiên, nó có thể được cấu hình để ngăn chặn hoặc cho phép người dùng nhập chuỗi mà không trùng khớp với các mục trong danh sách. Khi người dùng gõ một chuỗi trùng khớp với một mục trong danh sách, ComboBox cũng có thể được cấu hình để tự động hoàn tất chuỗi đó dựa trên cơ sở những kí tự đã được gõ, để hiển thị danh sách và đánh dấu phần tử đầu tiên trùng khớp, hoặc làm cả hai đồng thời. Khi người dùng gõ một chuỗi không trùng khớp với một mục nào trong danh sách, ComboBox kích hoạt sự kiện ItemInserting và ItemInserted mà có thể điều khiển được trong quá trình postback. Ngoài các hành vi đặc biệt đó, ComboBox hoạt động như một DropDownList.

ComboBox giống như là một phần bổ sung chứ không phải là thay thế cho AutoCompleteExtender. Mặc dù nó cũng có thể đáp ứng cùng một yêu cầu về giao diện người dùng.

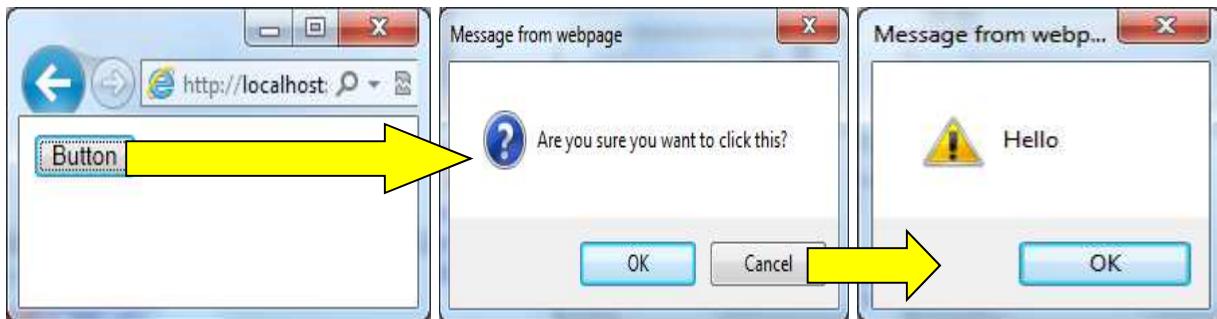
```
<ajaxToolkit:ComboBox ID="ComboBox1" runat="server"
    AutoPostBack="False"
    DropDownStyle="DropDownList"
    AutoCompleteMode="Suggest"
    CaseSensitive="False"
    CssClass="AjaxToolkitStyle"
    ItemInsertLocation="Append" ... >
```

The screenshot shows the configuration settings for the ComboBox1 control. The settings are:

- AutoPostBack:
- DropDownStyle: Simple DropDown DropDownList
- AutoCompleteMode: None Suggest Append SuggestAppend
- CaseSensitive:
- CssClass: [Empty String] AjaxControlToolkit Aqua Windows
- ItemInsertLocation: Append Prepend OrdinalText OrdinalValue

12.5.3.10 ConfirmButton Control

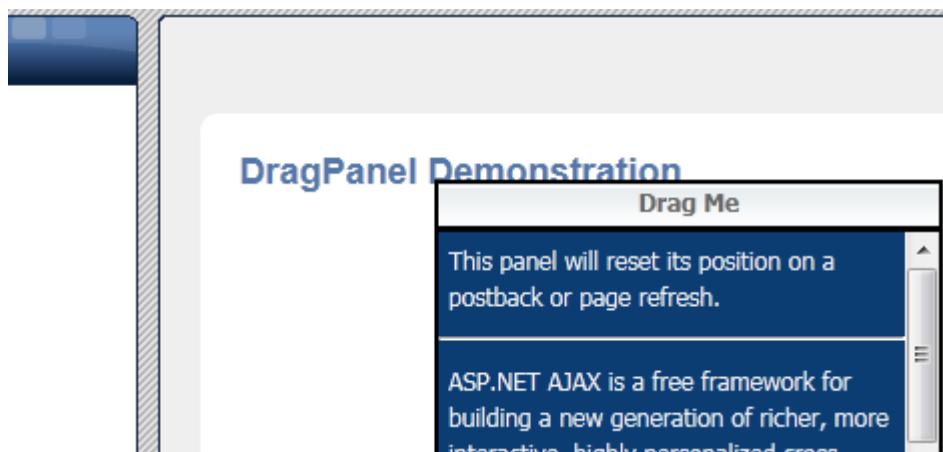
ConfirmButton là một extender đơn giản bắt sự kiện click một button và hiển thị một thông báo đến người dùng. Nếu button OK được click, hàm sẽ được xử lý bình thường, nếu không, sự kiện click sẽ bị bỏ qua và button sẽ không thực hiện hành vi mặc định của nó, thay vào đó, một đoạn mã sẽ được thực thi nếu thuộc tính OnClientCancel được thiết lập. Nó thật sự hữu ích để xóa liên kết hoặc bất cứ thứ gì khác yêu cầu xác thực từ người dùng.



```
<asp:Button ID="Button1" runat="server" Text="Button" />
<asp:ConfirmButtonExtender ID="cbe" runat="server"
    TargetControlID="Button1"
    ConfirmText="Are you sure you want to click this?"
    OnClientCancel="fnCancelClick"/>
<script>
function fnCancelClick() {
    alert("Hello");
}
</script>
```

12.5.3.11 DragPanel Control

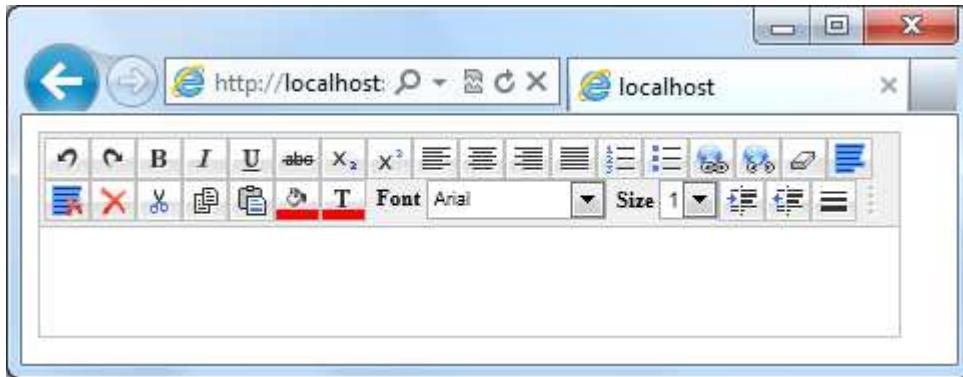
DragPanel extender cho phép người dùng dễ dàng thêm một phần có thể kéo đi được trong điều khiển của họ. DragPanel điều khiển bất kì panel nào của ASP.NET và thêm một tham số nói lên điều khiển được dùng như là có thể kéo đi được. Khi được khởi tạo, người dùng có thể thoải mái kéo panel quanh trang web sử dụng extender này.



```
<asp:Panel ID="Panel1" runat="server">
    <asp:Button ID="Button1" runat="server" Text="Drag Me" />
    ...
</asp:Panel>
<asp:DragPanelExtender ID="DPE1" runat="server"
    TargetControlID="Panel1"
    DragHandleID="Button1" />
```

12.5.3.12 HTMLEditor Control

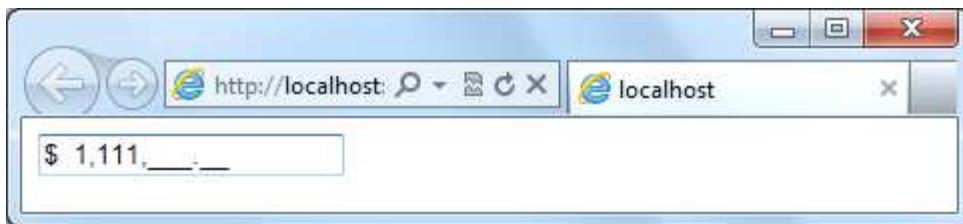
HTMLEditor là một điều khiển ASP.NET AJAX cho phép bạn dễ dàng tạo hoặc sửa đổi nội dung HTML. Các button khác trong thanh công cụ được dùng để biên tập nội dung. Bạn có thể thấy các thẻ HTML hoặc tài liệu xem trước.



```
<asp:TextBox TextMode="MultiLine" ID="TextBox1" runat="server" Height="90px"
Width="425px"></asp:TextBox>
<asp:HtmlEditorExtender ID="TextBox1_HtmlEditorExtender" runat="server" Enabled="True"
TargetControlID="TextBox1">
</asp:HtmlEditorExtender>
```

12.5.3.13 MaskedEdit Control

MaskedEdit gắn vào một textbox để hạn chế các loại văn bản được nhập vào. MaskedEdit áp một mặt nạ vào khung nhập mà chỉ cho phép một số loại ký tự / văn bản được nhập vào. Các dữ liệu được hỗ trợ định dạng là: Số, Ngày, Thời gian, và datetime. MaskedEdit sử dụng các thiết lập culture quy định tại các thuộc tính CultureName. Nếu không có quy định các thiết lập culture sẽ được giống như trang: Tiếng Việt (Việt Nam).

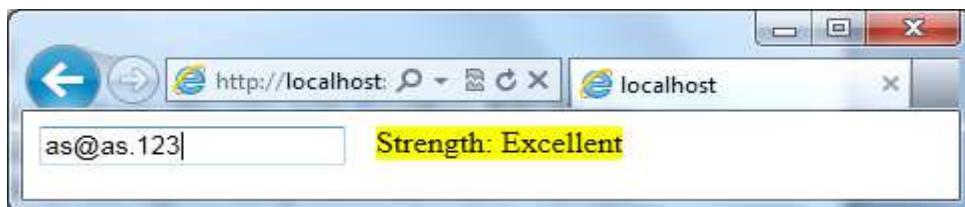


```
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
<asp:MaskedEditExtender runat="server" ID="abc"
TargetControlID="TextBox1"
Mask="9,999,999.99"
MessageValidatorTip="true"
OnFocusCssClass="MaskedEditFocus"
OnInvalidCssClass="MaskedEditError"
MaskType="Number"
InputDirection="RightToLeft"
AcceptNegative="Left"
DisplayMoney="Left" ErrorTooltipEnabled="True"/>
```

12.5.3.14 PasswordStrength Control

PasswordStrength là một ASP.NET AJAX extender có thể được gắn vào một TextBox điều khiển ASP.NET được sử dụng cho các mục nhập của mật khẩu. Các extender PasswordStrength cho thấy độ mạnh của mật khẩu trong TextBox và cập nhật chính nó khi người dùng đánh mật khẩu. Các chỉ số có thể hiển thị độ mạnh của mật khẩu dưới dạng tin nhắn văn bản hoặc với một chỉ số thanh tiến trình. Kiểu dáng và vị trí của cả hai loại chỉ số có thể được tùy chỉnh. Yêu cầu về độ mạnh của mật khẩu cũng có thể được tùy chỉnh, cho phép chúng ta tùy chỉnh yêu cầu về độ mạnh của mật

khẩu theo nhu cầu. Các tin nhắn văn bản mô tả độ mạnh hiện tại của mật khẩu cũng có thể được tùy chỉnh và giá trị mặc định của chúng có hỗ trợ sẵn cục bộ hóa. Chúng ta không có đủ các chuỗi cho tất cả ngôn ngữ hiện tại để có thể hiển thị nên chúng ta có thể dùng một ngôn ngữ chung cho một số ngôn ngữ khác. Một chỉ báo có thể được sử dụng để cung cấp hướng dẫn rõ về những gì thay đổi là cần thiết để đạt được một mật khẩu mạnh. Chỉ số này được hiển thị khi người dùng bắt đầu gõ vào TextBox và được ẩn đi mỗi khi textbox không được trỏ đến.

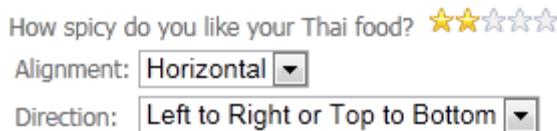


```
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

<asp:PasswordStrength ID="TextBox1_PasswordStrength" runat="server"
    Enabled="True" TargetControlID="TextBox1">
</asp:PasswordStrength>
```

12.5.3.15 Rating Control

Điều khiển Rating cung cấp một thể hiện đánh giá trực quan cho phép người dùng chọn số lượng các ngôi sao đại diện cho đánh giá của họ. Các nhà thiết kế trang có thể chỉ định đánh giá ban đầu, sự đánh giá tối đa cho phép, chiều và hướng của các ngôi sao, và tùy chỉnh style cho các trạng thái khác nhau có thể có của một ngôi sao. Đánh giá cũng hỗ trợ sự kiện ClientCallBack cho phép một đoạn code chạy sau khi người sử dụng đánh giá.



```
<asp:Rating ID="ThaiRating" runat="server" CurrentRating="2" MaxRating="5" StarCssClass="ratingStar"
    WaitingStarCssClass="savedRatingStar" FilledStarCssClass="filledRatingStar"
    EmptyStarCssClass="emptyRatingStar" OnChanged="ThaiRating_Changed" />
```

12.5.3.16 RoundedCorners Control

Các extender RoundedCorners làm tròn góc cho các phần tử. Để thực hiện điều này nó chèn vào các phần tử trước và sau khi các phần tử đó được chọn, do đó chiều cao tổng thể của phần tử này sẽ thay đổi một chút. Bạn có thể chọn các góc của panel đích cần phải được làm tròn bằng cách thiết lập thuộc tính Corners trên extender là None, TopLeft, TopRight, BottomRight, BottomLeft, Top, Right, Bottom, Left, hoặc tất cả.

```
<asp:RoundedCornersExtender ID="rce" runat="server" TargetControlID="Panel1" Radius="6" Corners="All" />
```



CornerRadius:

None 2px 4px 6px 10px

Corners:

Top Left Top Right
 Bottom Left Bottom Right

Border Color:

None Black Red Aqua

12.5.3.17 Tab Control

TabContainer là một ASP.NET AJAX Control giúp tạo ra một tập hợp các Tabs có thể được dùng để tổ chức các trang nội dung. TabContainer lưu trữ một số kiểm soát TabPanel.

Mỗi TabPanel định nghĩa một HeaderText hoặc HeaderTemplate cũng như ContentTemplate định nghĩa nội dung của nó. Các tab gần đây nhất vẫn chọn sau khi postback, và trạng thái Enabled của các tab vẫn còn sau khi postback.

Toolkit User Profile:

Signature and Bio	Email	Controls
-----------------------------------	-----------------------	--------------------------

Controls authored by Toolkit User (read-only - demo purposes):

- Calendar
- MaskedEdit
- Accordion
- Calendar
- Calendar

Show Controls Owned

Current Tab: Controls

```
<asp:TabContainer runat="server" OnClientActiveTabChanged="ClientFunction" Height="150px" Width="400px"
    ActiveTabIndex="1" OnDemand="true" AutoPostBack="false" TabStripPlacement="Top"
    CssClass="ajax_tab_xp" ScrollBars="None" UseVerticalStripPlacement="true" VerticalStripWidth="120px">
    <asp:TabPanel runat="server" HeaderText="Signature and Bio" Enabled="true" ScrollBars="Auto"
    OnDemandMode="Once">
        <ContentTemplate>
            ...

```

```
</ContentTemplate>
</asp:TabPanel>
...
</asp:TabContainer>
```

12.6 Bài Tập Thực Hành

Một số thí dụ với Ajax Toolkit

Khi sử dụng AjaxToolkit thì phải khai báo ScriptManager hoặc là ToolkitScriptManager

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

<asp:ToolkitScriptManager ID="ScriptManager1" runat="server">
</asp:ToolkitScriptManager>
```

Bài 1: Thực hành với Accordion Control

Accordion control giúp bạn định nghĩa nhiều lớp và trình bày chúng từng cái một. Nó giống như là có nhiều CollapsiblePanel controls mà chỉ có một cái có thể mở rộng tại một thời điểm. Accordion control chứa một hoặc nhiều AccordionPane controls. Mỗi AccordionPane control có một mẫu cho phần header và phần content của nó.

Bước 1: kéo thả Accordion Control và tạo các AccordionPane.

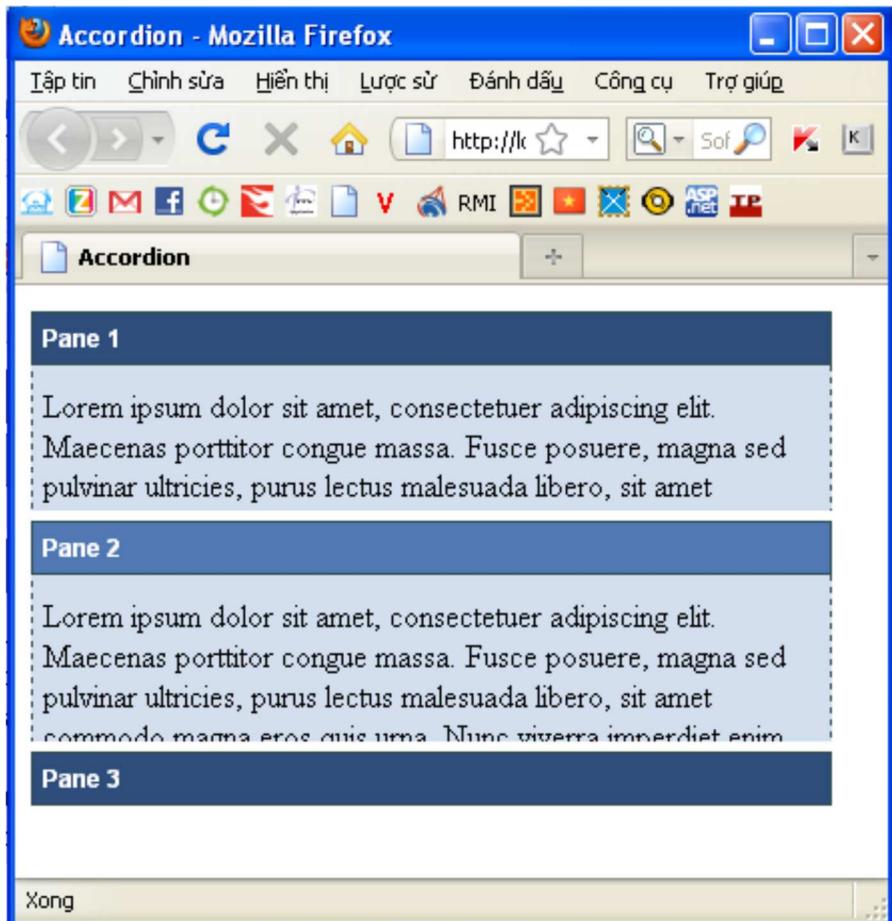
```
<asp:Accordion runat="server" ID="Accordion1"
CssClass="accordion"
HeaderCssClass="accordionHeader"
HeaderSelectedCssClass="accordionHeaderSelected"
ContentCssClass="accordionContent">
<Panes>
<asp:AccordionPane ID="AccordionPane1" runat="server">
<Header>Pane 1</Header>
<Content>
    Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
    Maecenas porttitor congue massa. Fusce posuere, magna sed
    pulvinar ultricies, purus lectus malesuada libero, sit amet
    commodo magna eros quis urna.
</Content>
</asp:AccordionPane>
<asp:AccordionPane ID="AccordionPane2" runat="server">
<Header>Pane 2</Header>
<Content>
    Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
    Maecenas porttitor congue massa. Fusce posuere, magna sed
    pulvinar ultricies, purus lectus malesuada libero, sit amet
    commodo magna eros quis urna.
</Content>
</asp:AccordionPane>
<asp:AccordionPane ID="AccordionPane3" runat="server">
<Header>Pane 3</Header>
<Content>
    Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
    Maecenas porttitor congue massa. Fusce posuere, magna sed
    pulvinar ultricies, purus lectus malesuada libero, sit amet
    commodo magna eros quis urna.
</Content>
</asp:AccordionPane>
</Panes>
</asp:Accordion>
```

Bước 2 : tạo CSS cho nội dung trong AccordionPane

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

```
.accordion {
    width: 400px;
}
.accordionHeader {
    border: 1px solid #2F4F4F;
    background-color: #2E4d7B;
    font-size: 12px;
    padding: 5px;
    cursor: pointer;
}
.accordionHeaderSelected {
    border: 1px solid #2F4F4F;
    background-color: #5078B3;
    font-size: 12px;
    padding: 5px;
    cursor: pointer;
}
.accordionContent {
    background-color: #D3DEEF;
    border-top: none;
    padding-top: 10px;
}
</style>
```

Bước 3: Chạy để xem kết quả



Bài 2: Thực hành với AlwaysVisible Control

Là một extender được dùng để hiển thị một cách liên tục đến một điều khiển ASP.NET. Điều khiển được mở rộng sẽ luôn luôn di chuyển đến một vị trí cố định trên trang bất kể trang thị thay đổi kích thước hay bị cuộn.

Bước 1: kéo thả một Panel:

```
<asp:Panel ID="Panel1" runat="server" CssClass="staticPanel">
    <h2> Hello Words!!!</h2>
</asp:Panel>
```

Bước 2: trong phần design, từ Panel trên chọn Add Extender, chọn AlwaysVisibleControlExtender. Ta có đoạn code như sau:

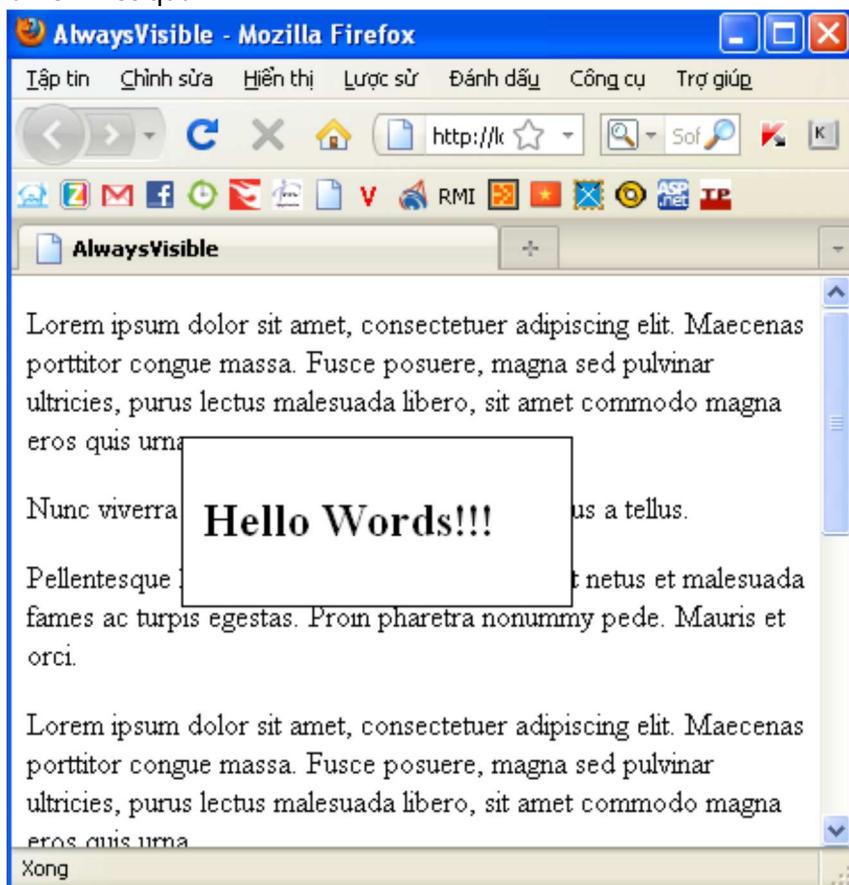
```
<asp:AlwaysVisibleControlExtender
    ID="Panel1_AlwaysVisibleControlExtender"
    runat="server" Enabled="True" VerticalSide="Middle"
    VerticalOffset="20" HorizontalSide="Center"
    HorizontalOffset="20" TargetControlID="Panel1">
</asp:AlwaysVisibleControlExtender>
```

Bước 3: tạo nội dung bên trong trang để thấy rõ chức năng của Control này.

Bước 4: tạo CSS cho Panel:

```
<style type="text/css">
.staticPanel {
    width: 110px;
    border: solid 1px black;
    background-color: White;
    padding: 10px;
}
</style>
```

Bước 5: Chạy và xem kết quả



Bài 3: Thực hành với ColorPicker Control

Phần mở rộng ColorPicker cho phép bạn hiển thị một bảng pop-up màu khi con trỏ di chuyển đến phần tử input. Bạn có thể gắn ColorPicker vào bất cứ textbox nào của ASP.NET. Nó cung cấp một hàm cho phép lựa chọn màu ở phía client với giao diện người dùng. Ngoài ra, bạn có thể chỉ ra một button để hiển thị một popup lựa chọn màu và một điều khiển cho phép xem trước màu từ bảng màu. Bạn có thể cung cấp một textbox để khi mà người dùng nhập vào một giá trị màu thì ColorPicker có thể hiện thị màu tương ứng nếu màu đó không có trong bảng màu mặc định.

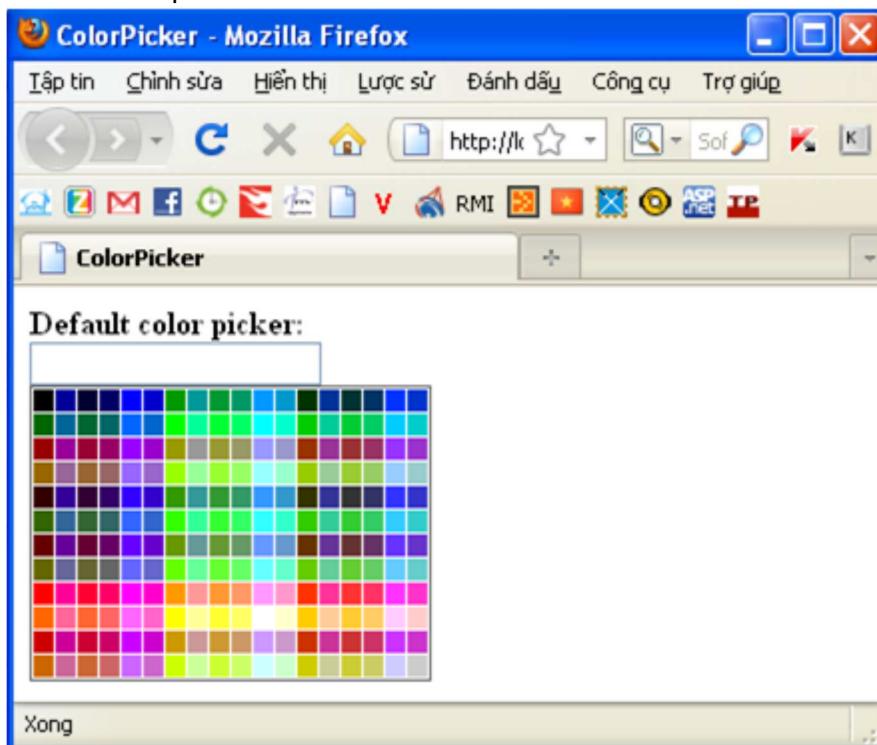
Bước 1: kéo thả một TextBox

```
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
```

Bước 2: từ TextBox AddExtender:

```
<asp:ColorPickerExtender ID="TextBox1_ColorPickerExtender" runat="server"
    Enabled="True" TargetControlID="TextBox1">
</asp:ColorPickerExtender>
```

Bước 3: Chạy và xem kết quả



Bài 4: Thực hành với ComboBox Control.

ComboBox là một điều khiển của ASP.NET AJAX , nó giống như AutoCompleteExtender, kết hợp linh hoạt với một Textbox với một danh sách tùy chọn cho phép người dùng có thể lựa chọn. Nó có các thuộc tính, hành vi hay quy ước đặt tên tương tự như trên Form combobox, và có cùng lớp cơ sở như là ListBox, BulletedList và DropDownList của web control. Thật vậy, ComboBox được xem như là DropDownList nhưng lại có thể gõ trực tiếp vào như là textbox.

Bước 1: kéo thả một ComboBox với các ListItem.

```
<asp:ComboBox ID="ComboBox1" runat="server">
    <asp:ListItem Value="0">Cao</asp:ListItem>
    <asp:ListItem Value="1">Trung Bình</asp:ListItem>
    <asp:ListItem Value="2">Thấp</asp:ListItem>
</asp:ComboBox>
```

Bước 2: tạo một Button và một Label để hiển thị nội dung trong ComboBox đã được chọn.

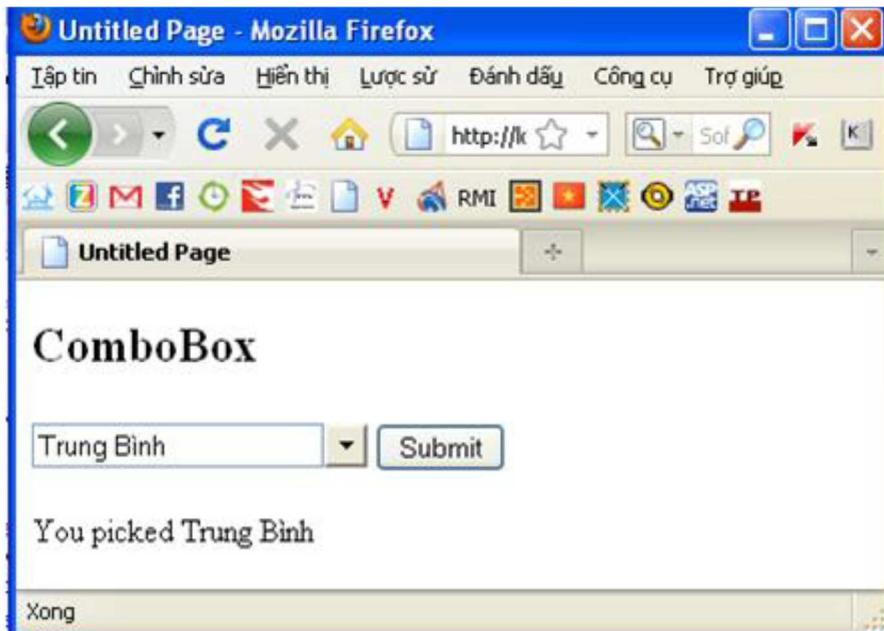
```
<asp:Button ID="btnSubmit" Text="Submit" Runat="server" OnClick="btnSubmit_Click"/>
```

```
<br /> <br />
<asp:Label ID="lblShow" Runat="server" />
```

Bước 3: Tạo một Script là Button_Click.

```
<script runat="server">
    public void btnSubmit_Click(object sender, EventArgs e) {
        lblShow.Text = "You picked " + ComboBox1.SelectedItem.Text;
    }
</script>
```

Bước 4: Chạy và xem kết quả



Bài 5: Thực hành với PopupCalendar Control.

Điều khiển Calendar extender có thể được gắn với bất kì điều khiển textbox nào của ASP.NET. Nó cung cấp hàm định dạng ngày phía client với định dạng ngày tùy biến với giao diện trong một điều khiển popup. Bạn có thể tương tác với calendar bằng cách click vào một ngày để chọn, hoặc Today để chọn ngày hiện tại. Ngoài ra, mũi tên trái phải có thể được sử dụng để chuyển tới tháng trước hoặc tháng sau. Bằng cách click vào tiêu đề của calendar bạn có thể thay đổi hiển thị của các ngày trong tháng hiện tại thành các tháng trong năm hiện tại. Click khác sẽ chuyển các năm trong thập kỷ hiện tại. Hành động này cho phép bạn dễ dàng chuyển đến một ngày trong quá khứ hoặc tương lai từ điều khiển calendar.

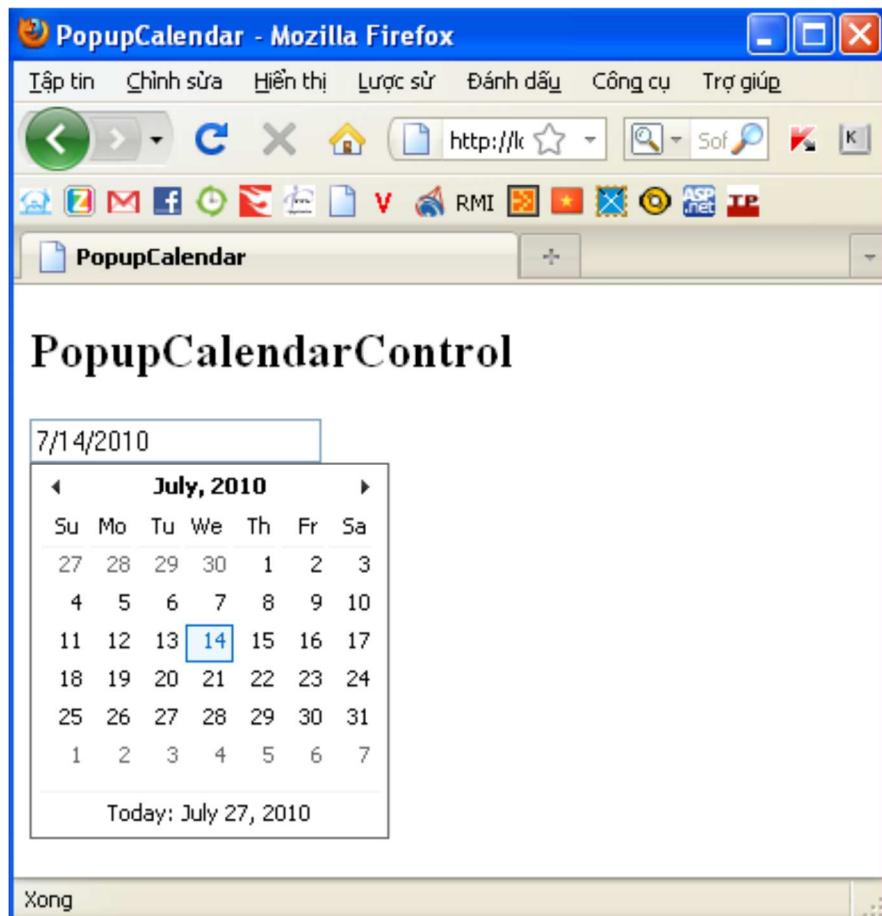
Bước 1: tạo một TextBox.

```
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
```

Bước 2: từ TextBox đã tạo ở trên Add Extender là CalendarExtender

```
<asp:CalendarExtender ID="TextBox1_CalendarExtender" runat="server" Enabled="True"
    TargetControlID="TextBox1"></asp:CalendarExtender>
```

Bước 3: Chạy và xem kết quả



Bài 6: Thực hành với AsyncFileUpload Control.

AsyncFileUpload cho phép bạn upload file lên server một cách bất đồng bộ, kết quả của file upload có thể được kiểm tra cả ở phía server lẫn client, bạn có thể lưu file được upload bằng cách gọi phương thức SaveAs() trong điều khiển sự kiện UploadedComplete ở server.

Bước 1: tạo một AsyncFileUpload

```
<ajaxToolkit:AsyncFileUpload ID="AsyncFileUpload1" Width="400px" runat="server"
    OnClientUploadError="uploadError"
    OnClientUploadStarted="StartUpload"
    OnClientUploadComplete="UploadComplete"
    CompleteBackColor="Lime" UploaderStyle="Modern"
    ErrorBackColor="Red" ThrobberID="Throbber"
    onuploadedcomplete="AsyncFileUpload1_UploadedComplete"
    UploadingBackColor="#66CCFF" />
```

Bước 2: tạo một Label để thể hiện một Image khi đang upload.

```
<asp:Label ID="Throbber" runat="server" Style="display: none">
    
</asp:Label>
```

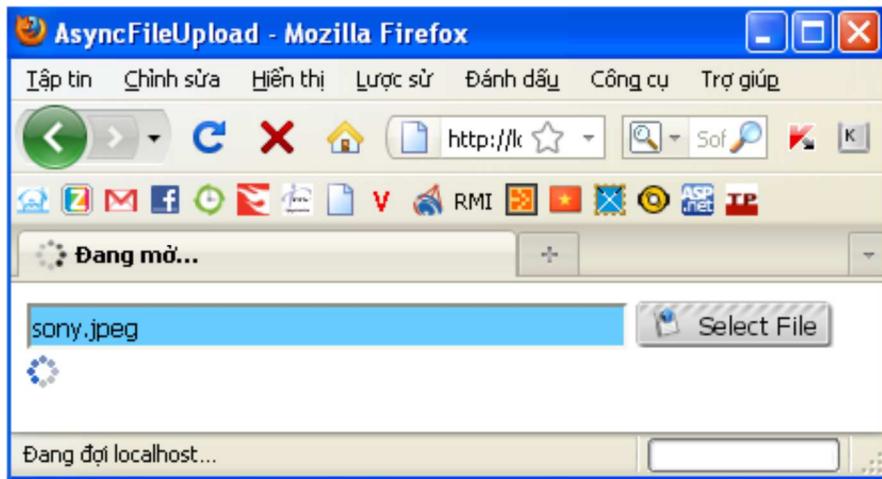
Bước 3: Đưa các Script.

```
<script type="text/javascript">
    function uploadError(sender, args) {
        document.getElementById('lblStatus').innerText = args.get_fileName(), "<span style='color:red;'>" +
        args.get_errorMessage() + "</span>";
    }
    function StartUpload(sender, args) {
        document.getElementById('lblStatus').innerText = 'Uploading Started.';
```

```

        }
        function UploadComplete(sender, args) {
            var filename = args.get_fileName();
            var contentType = args.get_contentType();
            var text = "Size of " + filename + " is " + args.get_length() + " bytes";
            if (contentType.length > 0) {
                text += " and content type is " + contentType + ".";
            }
            document.getElementById('lblStatus').innerText = text;
        }
    
```

Bước 4: Chạy và xem kết quả



Bài 7: Thực hành với DragPanelExtender

DragPanel extender cho phép người dùng dễ dàng thêm một phần có thể kéo đi được trong điều khiển của họ. DragPanel điều khiển bất kì panel nào của ASP.NET và thêm một tham số nói lên điều khiển được dùng như là có thể kéo đi được. Khi được khởi tạo, người dùng có thể thoải mái kéo panel quanh trang web sử dụng extender này.

Bước 1: tạo Panel.

```

<asp:Panel ID="PnlContainer" runat="server" cssclass="dragContainer">

<asp:Panel ID="PnlHeader" runat="server" CssClass="dragHeader">
    Click and Drag Here To Move ME....around the page
</asp:Panel>

<asp:Panel ID="PnlDetail" runat="server" CssClass="dragDetail">
    Here is just some content where you can add as much text as you like<br />
    line 1<br /> line 2<br />
    Remmeber that you can add any html text here<br /><br />
</asp:Panel>

```

Bước 2: tạo một DragPanelExtender

```

<asp:DragPanelExtender ID="PnlContainer_DragPanelExtender" runat="server" DragHandleID="PnlHeader"
    Enabled="True" TargetControlID="PnlContainer"> </asp:DragPanelExtender>

```

Bước 3: tạo CSS cho các nội dung bên trong trang.

```

<style type="text/css">
    .dragContainer{
        background-color: #FFC0FF; height: 282px;
        width: 357px; border-bottom-color: black;
    }
    .dragHeader{
        background-color: #8080FF; height: 48px; width: 358px;
    }

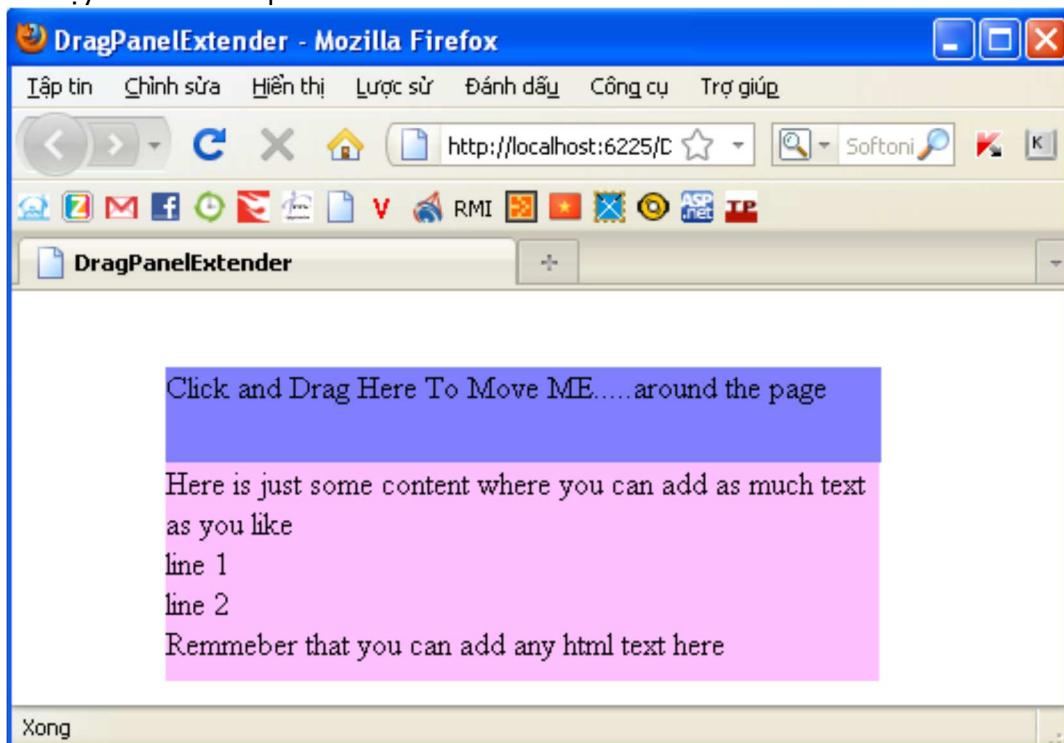
```

```
.dragDetail{
    background-color: #FFC0FF; height: 213px; width: 357px;
}
</style>
```

Bước 4: Script

```
<script type="text/javascript">
function setBodyHeightToContentHeight() {
    var height1 = document.documentElement.scrollHeight;
    var height2 = document.body.scrollHeight;
    document.body.style.height = Math.max(height1, height2) + "px";
}
setBodyHeightToContentHeight();
</script>
```

Bước 5: Chạy và xem kết quả



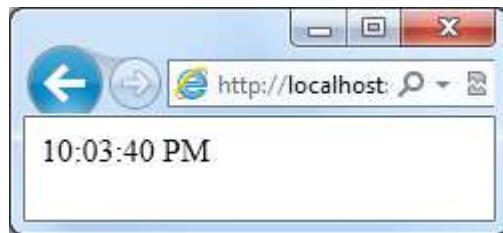
12.7 Sử dụng Ajax với JQuery

Ajax trong JQuery rất đơn giản. Hàm \$.ajax() được định nghĩa trong thư viện lõi của JQuery sẽ giúp chúng ta triển khai ứng dụng ajax một cách tổng quát nhất.

Cú pháp:

```
<script src="jquery/js/jquery-1.8.0.min.js" type="text/javascript"></script>
<script>
$(function () {
    $.ajax({
        url: "địa chỉ trang web xử lý",
        data: "danh sách tham số truyền đi",
        success: function(result){
            // mã xử lý kết quả trả về từ server
        }
    });
});
</script>
```

Ví dụ lấy giờ hiện tại của server và hiển thị trên trang web viết với JQuery như sau



Mã HTML và JQuery

```
<html>
<head>
<title></title>
<script src="jquery/js/jquery-1.8.0.min.js"></script>
<script>
$(function () {
    setInterval('callServer()', 1000);
});

function callServer() {
    $.ajax({
        url: "ServerClock.aspx?" + Math.random(),
        data: "",
        success: function (result) {
            // mã xử lý kết quả trả về từ server
            $("#clock").html(result);
        }
    });
}
</script>
</head>
<body>
<span id="clock"></span>
</body>
</html>
```

Mã ASP.NET của ServerClock.aspx

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

Mã C# code behind của ServerClock.aspx

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class ServerClock : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        // lấy thời gian hệ thống
        String time = DateTime.Now.ToString("hh:mm:ss tt");

        // gửi trả về client
        Response.Write(time);
    }
}
```

Chương 13: WEB SERVICE

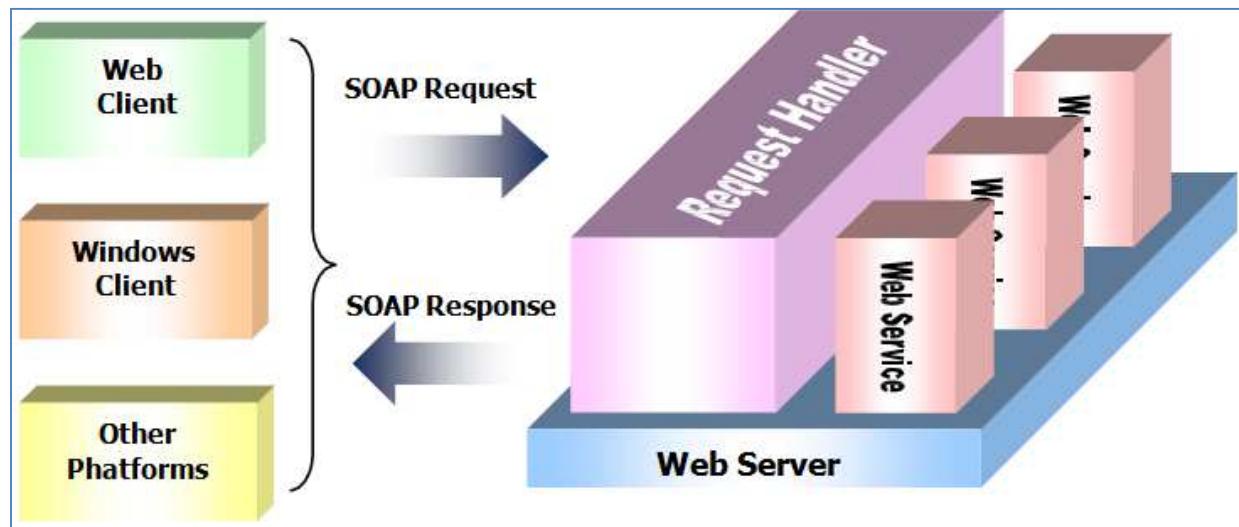
Sau khi học xong bài này, học viên có khả năng :

- Trình bày được kiến trúc và vai trò WebService trong ứng dụng thương mại điện tử
- Phát triển được ứng dụng Client-Server sử dụng WebService
- Thực hiện được mô hình bất đồng bộ trong WebService

13.1 Giới thiệu Web Service

- Web Services (tạm dịch là dịch vụ web) là tập hợp các phương thức của một đối tượng mà các Client có thể gọi từ xa thông qua môi trường Web (mạng cục bộ hoặc internet).
- Web Services Là một abstract interface, được thể hiện trong HTML dựa trên sự tương tác của User & Web Server.
- Web Services là một software application được truy xuất thông qua Web bởi một ứng dụng khác.
- Web Services hoạt động dựa vào XML thông qua Web protocols.
- Web Services được đăng ký tại nơi chung, và được đặc tả tất cả các chức năng để Client có thể sử dụng.
- Web Services là một chuẩn mới để xây dựng và phát triển ứng dụng phân tán, có khả năng làm việc trên mọi hệ điều hành, mở rộng khả năng phối hợp giữa các ứng dụng, có thể tái sử dụng, tăng cường sự giao tiếp giữa Client và Server thông qua môi trường Web.

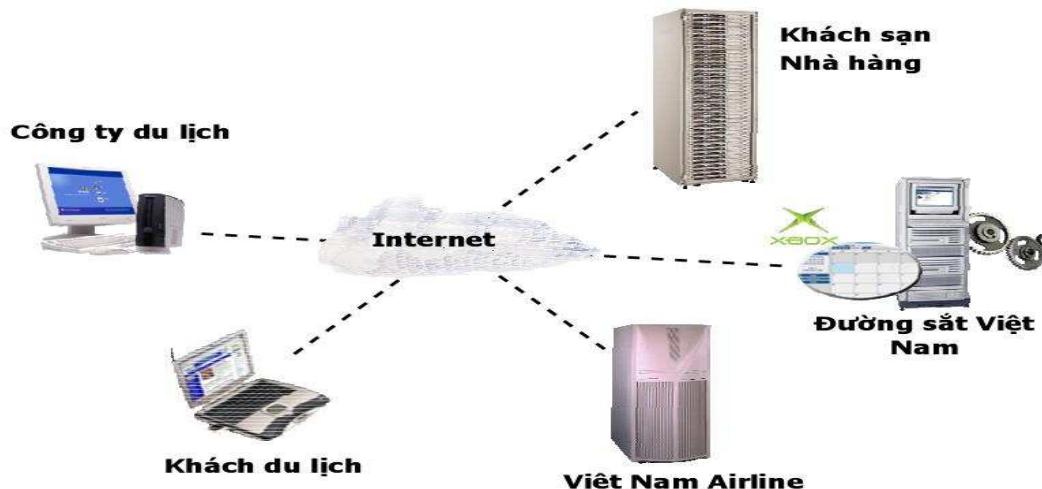
13.2 Kiến trúc Web Service



▪ Các giao thức truyền trong WebServices

- Web Services được xây dựng dựa trên SOAP (Simple Object Access Protocol). Không giống như DCOM, SOAP có thể được gọi thực hiện và trả về kết quả Text (theo định dạng XML) và có khả năng hoạt động "xuyên qua" tường lửa.

- Ngoài khả năng ưu việt trên, Web Services có thể phối hợp hoạt động giữa các ứng dụng rất tốt, hình dưới đây minh họa về sự phối hợp hoạt động giữa các ứng dụng.



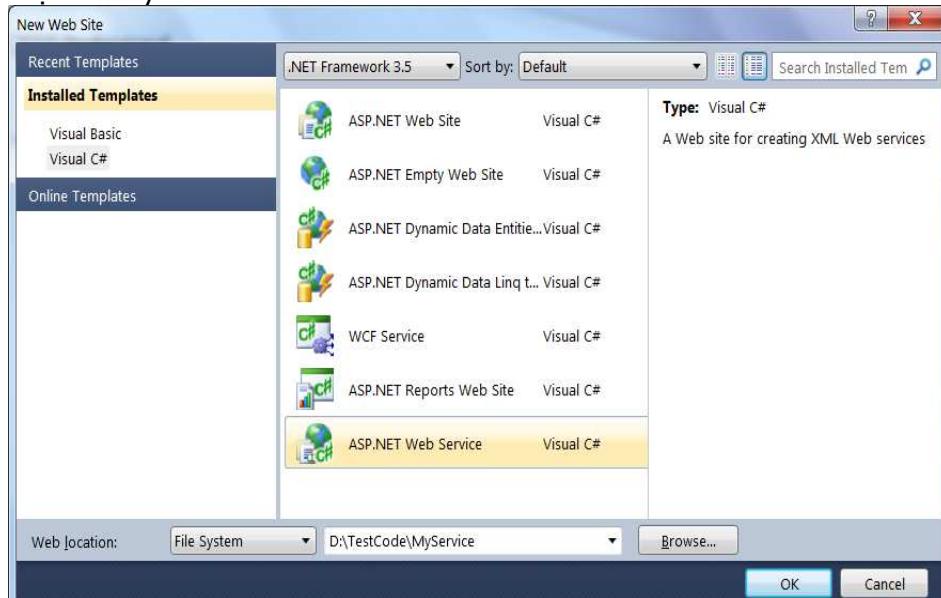
Chia sẻ thông tin giữa các tổ chức, doanh nghiệp

- Các nhà hàng, khách sạn cung cấp các Web Services cho phép đặt phòng, đặt tiệc. Đường sắt Việt Nam cung cấp các Web Services cho phép đặt vé tàu. Việt Nam Airline cung cấp các Web Services cho phép đặt vé cho các chuyến bay.
- Các doanh nghiệp, hay khách du lịch có nhu cầu tổ chức, tham gia các chuyến du lịch có thể truy cập vào website của các công ty dịch vụ lữ hành đăng ký tham gia các "tour" do họ tổ chức. Công ty du lịch sẽ sử dụng Web Services được cung cấp đó để tiến hành đặt vé tàu lửa, máy bay và đặt phòng cho chuyến du lịch theo yêu cầu của khách hàng.

13.3 Xây dựng ứng dụng Web Service

Bước 1. Tạo ứng dụng WebService

Từ menu *File | Add | New Web Site...*, cửa sổ *New Web Site* xuất hiện, chọn *ASP.NET WebService* đặt tên MyService.



Bước 2. Viết lệnh trong tập tin App_Code/Service.cs như sau.

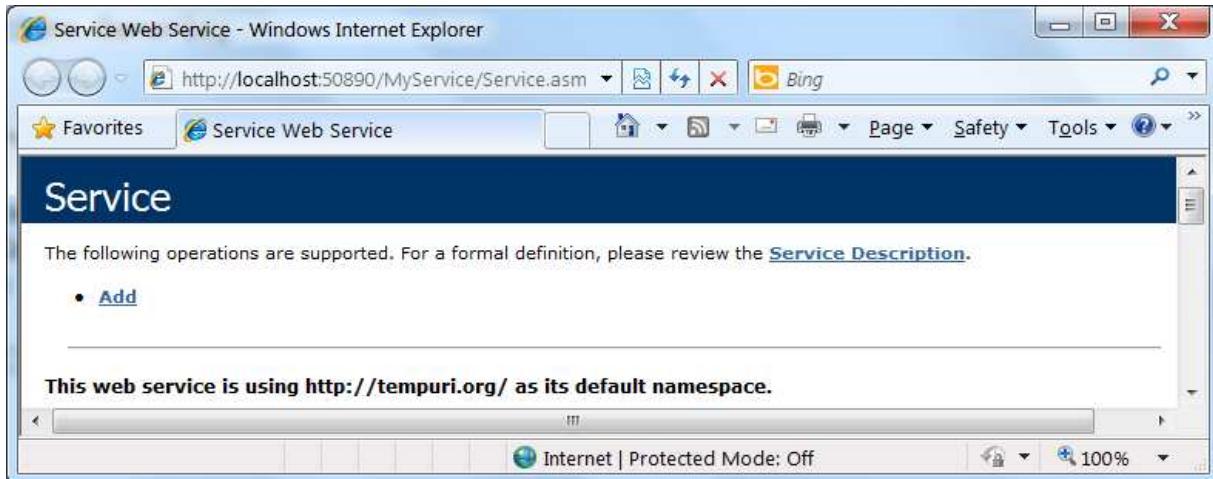
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;

[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)] public class
Service : System.Web.Services.WebService{
    public Service () {
    }

    [WebMethod]
    public int Add(int a, int b) {
        return a+b;
    }
    //Viết thêm các phương thức khác ở đây
    //.....
}
```

Trong WebService này, chúng ta khai báo một phương thức **Add** dùng để cộng hai số nguyên a và b.

Bước 3. Nhấn Ctrl+F5 chạy ứng dụng, hình dưới đây liệt kê phương thức **Add** hiện có trong Web Service.

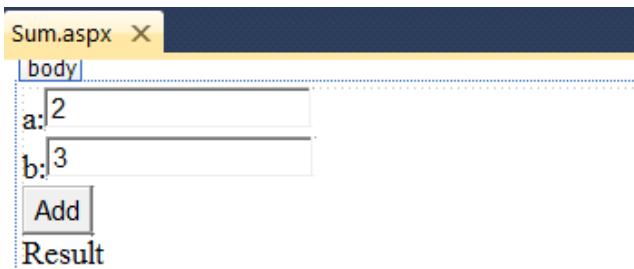


Nhấp vào liên kết **Add** để gọi phương thức này nhập vào 2 tham số a và b , sau đó nhấn **Invoke** để xem kết quả.

13.4 Xây dựng ứng dụng Web Client

Bước 1. Tạo ứng dụng Web Client , chúng ta có thể sử dụng bất kỳ ứng dụng nào để gọi đến Web Service (ASP.Net, Window Form, Console , JSP App). Trong ví dụ này chúng ta sử dụng ASP.NET Application.

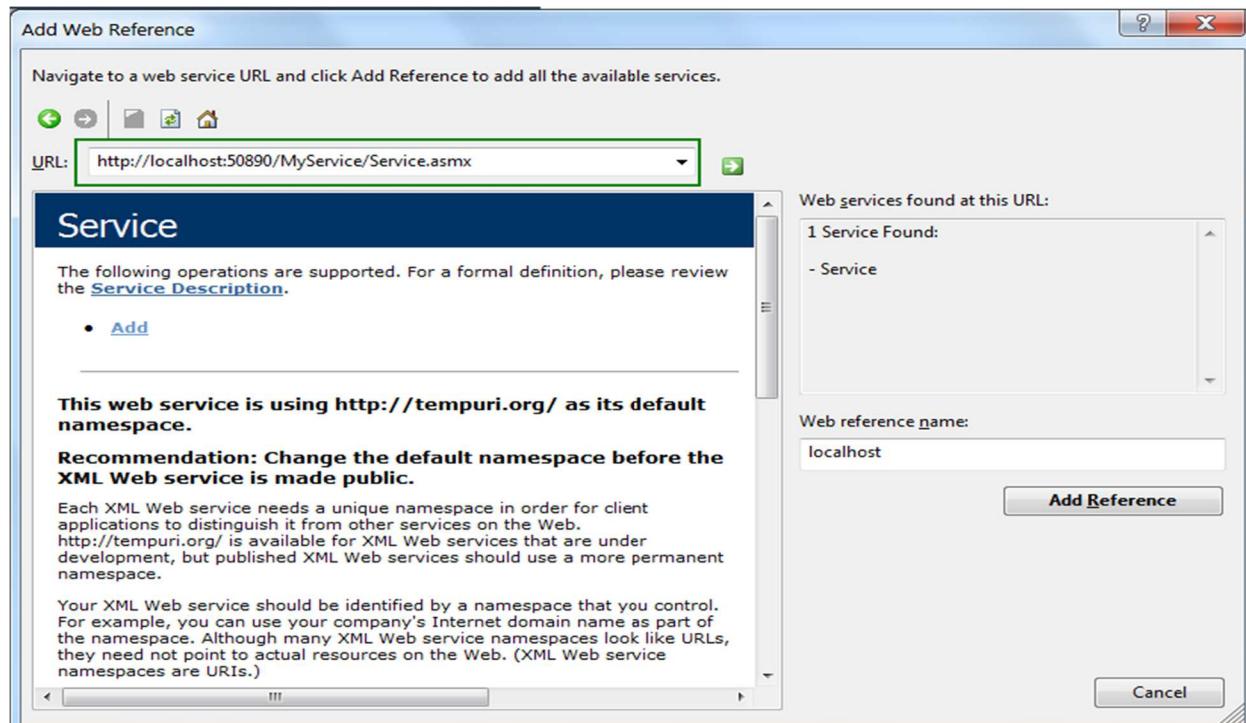
Tạo ứng dụng ASP.NET đặt tên MyWebApp, gồm 1 trang Sum.aspx giao diện được mô tả trong bảng sau :



Loại Control	Tên thuộc tính / sự kiện	Giá trị thiết lập
TextBox	ID	txtA
	Text	2
TextBox	ID	txtB
	Text	3
Button	ID	btnAdd
	Text	Add
	Sự kiện Click	btnAdd_Click
Label	ID	lbResult
	Text	Result

Bước 2. Thiết lập tham chiếu đến MyService : từ ứng dụng ASP.Net , nhấp chọn *References* , nhấp phải chuột chọn *Add Web Reference...*

Trên cửa sổ *Add Web Reference* , nhập vào địa chỉ *URL* của Service.asmx đã chạy ở **Bước 3** (mục 13.3). Sau đó nhấp *Add Reference* để thêm tham chiếu vào ứng dụng.

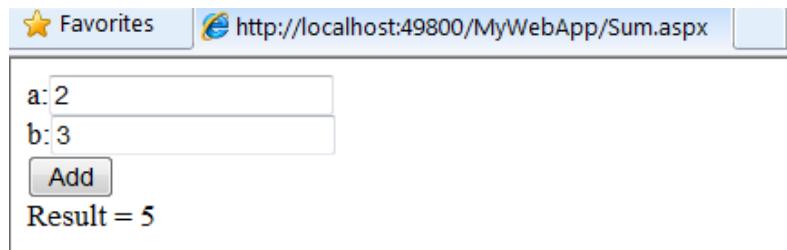


Bước 3. Viết lệnh xử lý sự kiện Click của btnAdd trong trang Sum.aspx.cs

```
protected void btnAdd_Click(object sender, EventArgs e)
{
    int a = int.Parse(txtA.Text);
    int b = int.Parse(txtB.Text);
```

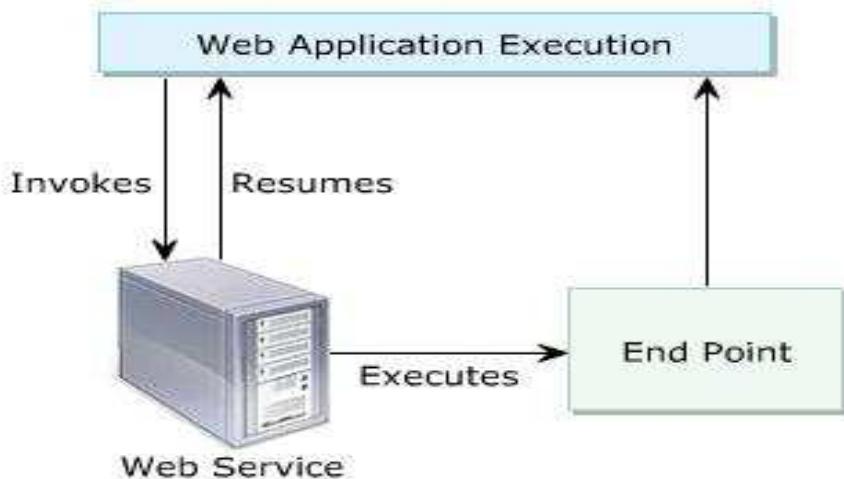
```
//khai báo đối tượng Service
localhost.Service obj = new localhost.Service();
//gọi phương thức
lbResult.Text="Result = "+ obj.Add(a,b).ToString();
}
```

Bước 4. Nhấn Ctrl+F5 chạy trang Sum.aspx và nhấp nút **Add** để xem kết quả.



13.5 Asynchronous Call trong Web Service

Trong mô hình gọi bất đồng bộ (Asynchronous Call) phương thức của WebService, ứng dụng Client có thể thực hiện các công việc khác trong khi phương thức web đang được xử lý ở phía Server. Mục đích nhằm tăng *khả năng thực thi và qui mô* của hệ thống.



Để thực hiện gọi phương thức **Add** theo mô hình bất đồng bộ, ta viết lại đoạn lệnh của phương thức btnAdd_Click như sau và thêm thuộc tính Asyn = true trong @Page của trang Sum.aspx

```
using localhost;
protected void btnAdd_Click(object sender, EventArgs e)
{
    int a = int.Parse(txtA.Text);
    int b = int.Parse(txtB.Text);
    //khai báo đối tượng Service
    Service obj = new Service();
    obj.AddCompleted += new AddCompletedEventHandler(obj_AddCompleted);
    //gọi phương thức bất đồng bộ
    obj.AddAsync(a, b);
    Response.Write("Completed.");
}
void obj_AddCompleted(object sender, localhost.AddCompletedEventArgs e)
{
    //lấy kết quả trả về
    lbResult.Text = "Result = " + e.Result.ToString();
}
```

Chương 14: TIỆN ÍCH

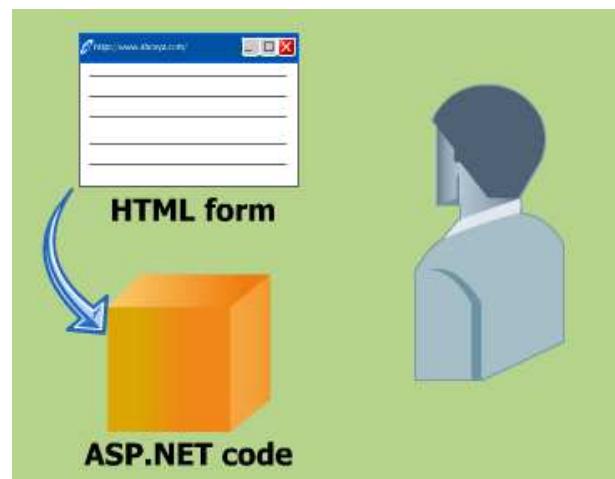
Sau khi học xong chương này, học viên có khả năng:

- Trình bày được cú pháp ngôn ngữ XML
- Thực hiện được các thao tác với tập tin XML sử dụng lớp XmlDocument
- Thực hiện được các thao tác với tập tin XML sử dụng LINQ to XML
- Trình bày và triển khai được cơ chế url routing trong ASP.NET
- Thực hành được public ASP.NET website và triển khai trên Hosting
- Trình bày được cơ chế bảo mật trong ASP.NET : Authorization và Authentication
- Ứng dụng được FormAuthentication trong ứng dụng ASP.NET
- Thực hiện được mã hoá dữ liệu sử dụng các thuật toán MD5, SHA1, SHA5

14.1 Authentication

14.1.1 Form Authentication

Sử dụng Forms Authentication Provider. Trong forms-base authentication sử dụng tập hợp các thông tin để xác nhận : tên đăng nhập, mật khẩu và ta phải viết các lệnh để xác nhận các thông tin được cung cấp phải phù hợp trong cơ sở dữ liệu. Những thông tin đã được xác nhận của người dùng có thể được lưu trong một biến cookie trong suốt phiên làm việc.



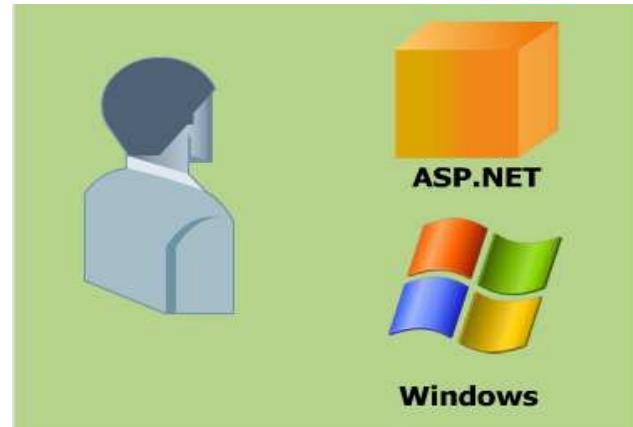
14.1.2 Passport Authentication

Người dùng được xác nhận bằng cách sử dụng Passport Service được cung cấp bởi Microsoft. Tuy nhiên, khi sử dụng loại xác nhận này, chúng ta phải được đăng ký với Microsoft's Passport Service. Passport server sử dụng cơ chế mã hóa cookie để định danh và xác nhận tính hợp lệ của người dùng.



14.1.3 Windows Authentication

- Windows authentication là cách xác nhận mặc định của ASP.NET. Loại xác nhận này dựa vào các windows account của người dùng.
- Windows authentication trong ASP.Net sử dụng IIS để có thể cấu hình cho phép chỉ các người dùng trong Windows domain đăng nhập vào ứng dụng.



- *Có 4 tùy chọn được thiết lập trên IIS :*

- Anonymous Authentication : cho phép bất kỳ người dùng truy cập đến ứng dụng ASP.Net
- Basic Authentication : yêu cầu sử dụng tên người dùng và mật khẩu của Windows để kết nối đến ứng dụng. Tuy nhiên mật khẩu được truyền trong dạng text đơn giản , do đó loại xác nhận này không được an toàn.
- Digest Authentication : tương tự như Basic Authentication. Tuy nhiên mật khẩu được hashed sau đó mới được truyền đi (tăng sự an toàn).
- Integrated Windows Authentication : sử dụng Kerberos hoặc giao thức change/response để xác nhận người dùng.

14.2 Authorization

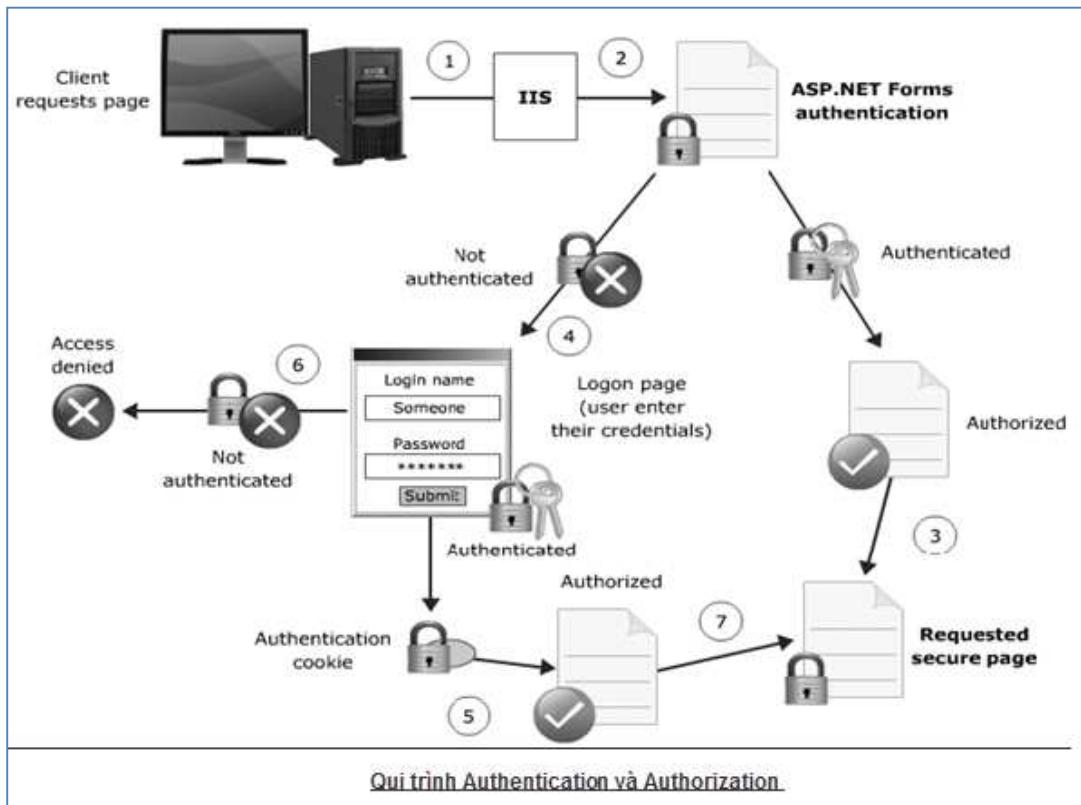
14.2.1 File Authorization

Sử dụng hệ thống tập tin NTFS cho phép kiểm tra quyền truy cập các tập tin trên của tài khoản người dùng mà ứng dụng ASP.NET đang sử dụng, chẳng hạn như người dùng muốn mở tập tin thì phải có quyền truy cập tập tin.

14.2.2 Url Authorization

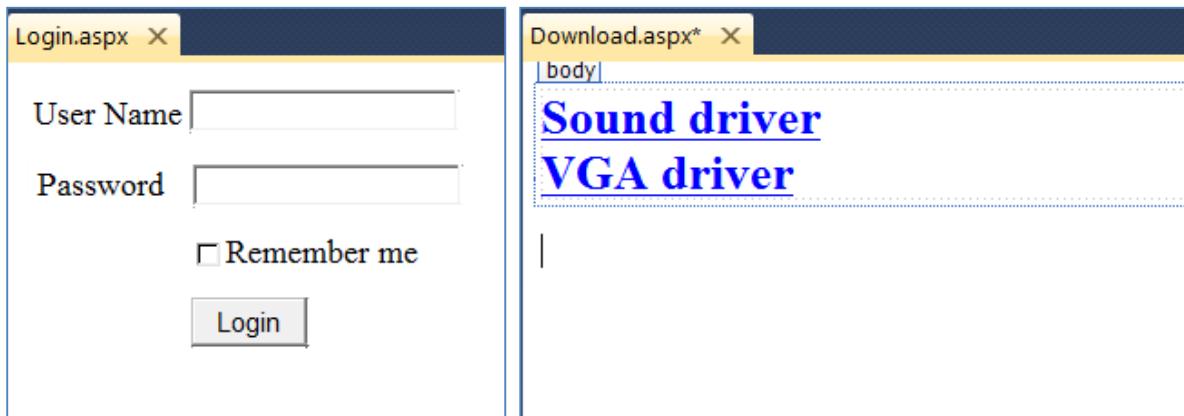
- Trong file web.config, các quy tắc cho phép được quy định cụ thể cho các thư mục khác nhau hoặc các tập tin của một ứng dụng
- Sử dụng các thẻ <authorization>, để chỉ định tên của người dùng được phép hoặc từ chối truy cập

14.2.3 Các bước cấu hình Authentication và Authorization



Trong phần này, ta phát triển 1 ứng dụng ASP.NET sử dụng Form-based Authentication theo yêu cầu : người dùng truy cập vào **Download.aspx** bắt buộc phải đăng nhập, nếu chưa đăng nhập thì chuyển qua trang **Login.aspx** để kiểm tra tài khoản người dùng (được khai báo trong tập tin web.config).

Bước 1. Tạo ứng dụng ASP.NET đặt tên là TestAuthentication gồm trang Login.aspx và Download.aspx



Bước 2. Cấu hình cho thẻ **<authentication>** và **<authorization>** trong tập tin web.config như sau :

```
<authentication mode="Forms" >
  <forms loginUrl="Login.aspx" defaultUrl="Download.aspx"
    name="MyWeb" timeout="30">
    <credentials passwordFormat="Clear">
      <user name ="a" password ="123"/>
      <user name ="b" password ="456"/>
    </credentials>
  </forms>
</authentication>

<authorization>
  <deny users="?"/>
</authorization>
```

- **loginUrl** : chỉ rõ trang người dùng sẽ đăng nhập
- **defaultUrl** : chỉ rõ trang sẽ được chuyển đến nếu đăng nhập thành công.
- **name** : chỉ rõ tên Cookie sẽ được tạo
- **timeout** : thời gian cookie sẽ tồn tại (tính bằng phút)
- **passwordFormat** : không mã hóa mật khẩu (password)

Trong ví dụ này ta khai báo 2 người dùng có tên đăng nhập là a và b (có thể dùng cơ sở dữ liệu để lưu tài khoản người dùng)

Bước 3. Viết code cho nút btnLogin trang Login.aspx

```
using System.Web.Security;

public partial class Login : System.Web.UI.Page
{
  protected void btnLogin_Click(object sender, EventArgs e)
  {
    //kiem tra tai khoan nguoi dung
    if (FormsAuthentication.Authenticate(txtUserName.Text, txtPassword.Text))
    {
      //neu tai khoan hop le va nguoi dung chon "Remember me"
      //thi tao cookie va chuyen qua trang Download.aspx
      FormsAuthentication.RedirectFromLoginPage(txtUserName.Text, chkRemember.Checked);
    }
  }
}
```

Bước 4. Chạy trang Download.aspx, ASP.NET tự động chuyển qua trang Login.aspx (vì user chưa được chứng thực) nhập UserName: a, Password : 123 và nhập Login sẽ vào trang Download.aspx.



14.3 Mã hóa dữ liệu

Khái niệm mã hóa dữ liệu đề cập đến những phép tính toán học và chương trình thuật toán chuyển văn bản gốc thành dạng văn bản mã hóa, đây là một dạng thức khiến cho những người không được ủy quyền không thể đọc được. Người nhận văn bản mã hóa sẽ sử dụng một khóa tạo nên cơ chế thuật toán để giải mã dữ liệu, chuyển nó trở về phiên bản văn bản ban đầu.

14.3.1 Thuật toán MD5

MD5 (Message-Digest algorithm 5) là một hàm băm mật mã được sử dụng phổ biến với giá trị băm dài 128-bit. Là một chuẩn Internet (RFC 1321), MD5 đã được dùng trong nhiều ứng dụng bảo mật và cũng được dùng phổ biến để kiểm tra tính toàn vẹn của tập tin. Một bảng băm MD5 thường được diễn tả bằng một hệ thập lục phân 32 ký tự. Thuật toán này đã có sẵn trong ASP.NET, ta có thể sử dụng thông qua System.Web.Security

```
string chuoia = "abc";
string chuoimahoa = FormsAuthentication.
    HashPasswordForStoringInConfigFile(chuoia, "MD5");
Response.Write("Chuoi sau khi ma hoa(MD5) :" + chuoimahoa);
```

14.3.2 Thuật toán SHA-1

SHA-1 (Secure Hash Algorithm 1) hay thuật giải băm an toàn. SHA-1 (trả về kết quả dài 160 bit). SHA-2 có các phiên bản: SHA-224 (224 bit), SHA-256 (256 bit), SHA-384 (384 bit) và SHA-512 (512 bit) được sử dụng để chuyển một đoạn dữ liệu nhất định thành một đoạn dữ liệu có chiều dài không đổi với xác suất khác biệt. SHA đã được tích hợp sẵn trong ASP.NET thông qua System.Web.Security và có thể được sử dụng để thay thế cho MD5 hay các thuật toán 128 bit khác.

```

string chuoit = "abc";
string chuoimahoa = FormsAuthentication.
    HashPasswordForStoringInConfigFile(chuoit, "SHA1");
Response.Write("Chuoit sau khi ma hoa(SHA1) :" + chuoimahoa);

```

14.3.3 Thuật toán SHA5 (SHA-2 512 bit)

Thuật toán SHA5 giống SHA1 nhưng trả về kết quả dài 512-bit. Thuật toán này được sử dụng thông qua System.Security.Cryptography

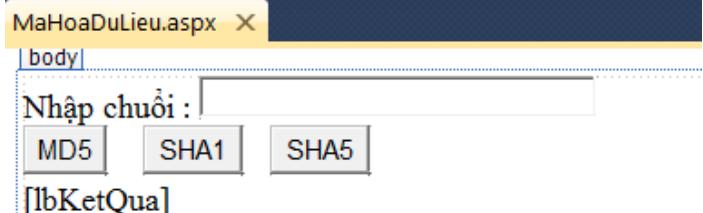
```

string chuoit = "abc";
byte[] data = Encoding.ASCII.GetBytes(chuoit);
SHA512 sha5 = new SHA512Managed();
byte[] result = sha5.ComputeHash(data);
string chuoimahoa = Encoding.ASCII.GetString(result);
Response.Write("Chuoit sau khi ma hoa:" + chuoimahoa);

```

14.3.4 Ứng dụng thuật toán mã hóa

Bước 1 Tạo ứng dụng ASP.NET đặt tên là EncrytionApp gồm 1 trang MaHoaDuLieu.aspx có giao diện như hình bên.



```

using System.Web.Security;
using System.Text;
using System.Security.Cryptography;
public partial class MaHoaDuLieu : System.Web.UI.Page{
    protected void btnMD5_Click(object sender, EventArgs e){
        string chuoimahoa = FormsAuthentication.
            HashPasswordForStoringInConfigFile(txtChuoit.Text, "MD5");
        lbKetQua.Text = "Chuoit sau khi ma hoa(MD5):<br/>" + chuoimahoa;
    }
    protected void btnSHA1_Click(object sender, EventArgs e){
        string chuoimahoa = FormsAuthentication.
            HashPasswordForStoringInConfigFile(txtChuoit.Text, "SHA1");
        lbKetQua.Text = "Chuoit sau khi ma hoa(SHA1):<br/>" + chuoimahoa;
    }
    protected void btnSHA5_Click(object sender, EventArgs e){
        byte[] data = Encoding.ASCII.GetBytes(txtChuoit.Text);
        SHA512 sha5 = new SHA512Managed();
        byte[] result = sha5.ComputeHash(data);
        string chuoimahoa = Encoding.ASCII.GetString(result);
        lbKetQua.Text = "Chuoit sau khi ma hoa(SHA5):<br/>" + chuoimahoa;
    }
}

```

Bước 2 Viết code cho sự kiện click của Button control

Bước 3. Chạy ứng dụng và kiểm tra kết quả.

Nhập chuỗi : abc

 Chuoi sau khi ma hoa(SHA5):
 ??5??az??AsI? A1? ??N??~? ??KU??!??*!O??6?<#????EMD#d

14.4 Khảo sát ngôn ngữ XML

14.4.1 Khái niệm về XML

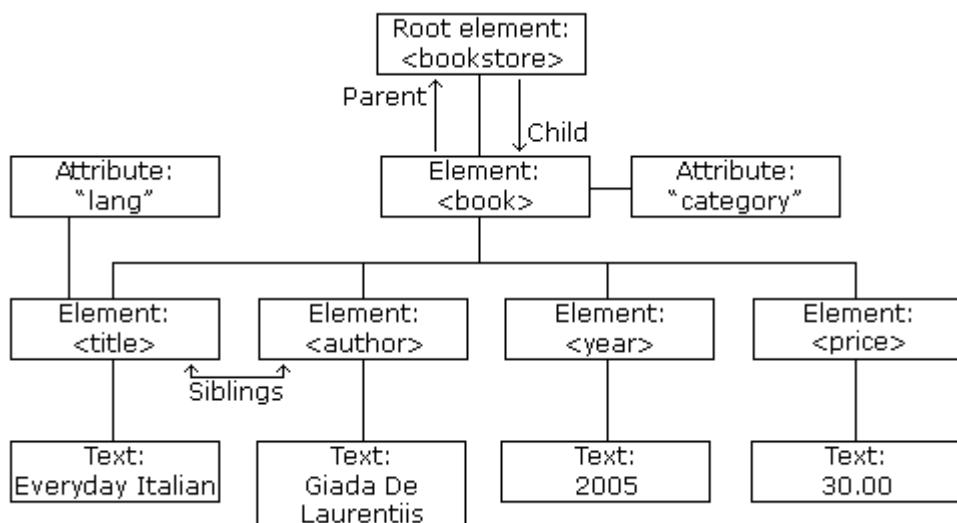
XML (eXtensible Markup Language) là ngôn ngữ đánh dấu, nhưng khác với *HTML*. *XML* là ngôn ngữ **biểu diễn** dữ liệu, trong khi *HTML* **định dạng** dữ liệu. Khái niệm *eXtensible* có nghĩa là các ứng dụng sử dụng *XML* có thể tạo các phần tử và thuộc tính mới.

14.4.2 Cấu trúc tài liệu XML

Giả sử ta có tài liệu *XML* về các cuốn sách như sau:

```
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

Khi đó có thể biểu diễn nó dưới dạng cấu trúc cây:



Dựa trên cấu trúc cây này người ta xác định các loại nút (*node*) và quan hệ giữa chúng.

Có 3 khái niệm cơ bản liên quan đến tài liệu *XML*, đó là "thẻ", "phần tử", và "thuộc tính".

Thẻ (tag): là thành phần nằm giữa cặp dấu < và >. Thẻ phân ra thành thẻ mở và thẻ đóng.
 Ví dụ:

Thẻ mở: <book category = "WEB">

Thẻ đóng: </book>

Thuộc tính (attribute): là cặp tên – giá trị nằm trong thẻ mở.

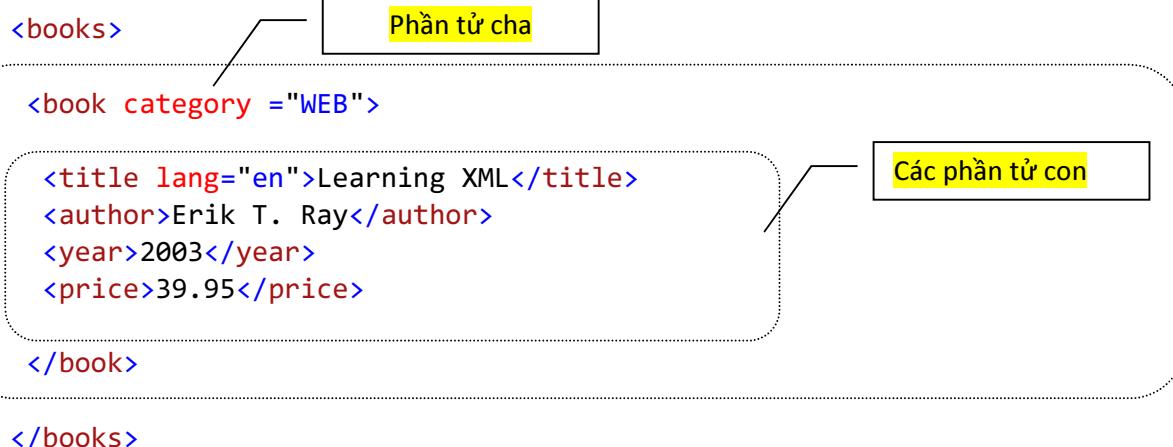
Ví dụ: <book category = "WEB">

Tên thuộc tính là “category” và giá trị tương ứng của nó là “WEB”.

Phần tử (element): bao gồm thẻ mở, thẻ đóng và tất cả những gì nằm giữa chúng.

Ví dụ: <title lang="en">Learning XML</title>

Trong tài liệu, các phần tử được phân cấp theo quan hệ cha – con. Ví dụ:



Phần tử gốc (root):

Toàn bộ tài liệu được đặt trong một phần tử duy nhất, đó chính là phần tử gốc. Trong tài liệu trên, gốc được xác định bởi cặp thẻ <books></books>

14.4.3 Tạo một tài liệu XML

Vì tài liệu XML là một file văn bản, do đó bạn có thể tạo ra chúng bằng cách sử dụng bất kỳ trình soạn thảo văn bản nào.

Tuy nhiên có những trường hợp bạn cần thao tác với các tài liệu XML tại thời điểm *run-time*. Do đó .NET cài đặt sẵn các lớp và tiện ích để làm việc với tài liệu XML như *XDocument* và *LINQ to XML*. Chúng ta sẽ lần lượt khảo sát các thao tác với tài liệu XML dựa trên các lớp này.

14.4.4 Sử dụng lớp Xdocument

Hình dưới đây cho bạn thấy thông tin về *XmlDocument* trong .Net.

```

        > XmlNodeConvert
        > XmlDateTimeSerializationMode
        > XmlDeclaration
        > XmlDocument
        > XmlDocumentFragment
        > XmlDocumentType
        >XmlElement
        > XmlEntity
        > XmlEntityReference
        > XmlException
        > XmlImplementation
    
```

public class XmlDocument : System.Xml.XmlNode
 Member of [System.Xml](#)

Summary:
 Represents an XML document.

.NET cung cấp rất nhiều namespace làm việc với XML, bao gồm: *System.Xml*, *System.Xml.Schema*, *System.Xml.Serialization*, *System.Xml.XPath*, và *System.Xml.Xsl*.

Trong đó **System.Xml** chứa các lớp chính để thao tác với tài liệu XML như: *XmlNode*, *XmlReader*, *XmlTextReader*, *XmlValidatingReader*, *XmlNodeReader*, *XmlWriter*, và *XmlTextWriter*.

Lớp *XmlNode* biểu diễn toàn bộ tài liệu XML như một cây (*tree*) trong bộ nhớ trong và cho phép chúng ta thao tác với tài liệu thông qua "cây" này.

Thao tác đọc tài liệu XML:

Nói chung việc đọc một tài liệu XML tùy thuộc vào cấu trúc của tài liệu đó, và thường có tính đệ quy. Sở dĩ như vậy vì khi đã tải tài liệu XML vào đối tượng *XmlNode* thì việc duyệt trên đối tượng này giống như duyệt trên một cấu trúc cây.

Mỗi *node* của tài liệu có thuộc tính *NodeType* cho biết đó là *node* loại nào. Các giá trị của thuộc tính này là: *Element*, *Text*, ..

Ví dụ: để duyệt qua tất cả các *node* Text trong tài liệu, ta làm như sau:

```

static public void Process(XmlNode node, int level)
{
    if(node.NodeType == XmlNodeType.Text)
        Console.WriteLine("{0}", node.InnerText);
    foreach (XmlNode child in node.ChildNodes)
    {
        Process(child, level + 1);
    }
}

```

Các thao tác với XmlDocument:

Để thuận tiện, chúng ta giả sử rằng các thao tác trình bày sau đây đều làm việc trên file "InputFile.xml" có cấu trúc như sau:

```

<?xml version="1.0" encoding="utf-8" ?>
<contacts>
    <person category="personal">
        <name>Tran Van Mot</name>
        <phone>1111111111</phone>
        <email>mot@gmail.com</email>
    </person>
    <person category="business">
        <name>Nguyen Thi Hai</name>

```

```
<phone>2222222222</phone>
<email>hai@yahoo.com</email>
</person>
<person category="family">
    <name>Bui Dinh Ba</name>
    <phone>3333333333</phone>
    <email>ba@gmail.com</email>
</person>
</contacts>
```

Mặt khác ta cũng giả sử lớp *Person* sau đây đã được xây dựng:

```
class Person
{
    public string Name { set; get; }
    public string Phone { set; get; }
    public string Email { set; get; }
    public Person(string name, string phone, string mail)
    {
        this.Name = name;
        this.Phone = phone;
        this.Email = mail;
    }
    public override string ToString()
    {
        return this.Name + ", " + this.Phone + ", " + this.Email;
    }
}
```

Duyệt tài liệu XML:

XmlDocument cung cấp các phương thức và thuộc tính cho phép chúng ta duyệt qua các *node* trong tài liệu:

```
static public void TraverseFile()
{
    XmlDocument doc = new XmlDocument();
    doc.Load(inputFilePath);
    XmlNode nod = doc.SelectSingleNode("contacts/person");//tìm node đầu tiên thỏa
    XPath
    while (nod != null)
    {
        Console.WriteLine(nod.InnerText);
        nod = nod.NextSibling;//chuyển sang node kế tiếp
    }
}
```

Đọc tài liệu XML:

```
static public List<Person> ReadElements()
{
    List<Person> temp = new List<Person>();
    XmlDocument doc = new XmlDocument();
    doc.Load(inputFilePath);
    XmlNodeList nodeList = doc.SelectNodes("contacts/person");
    foreach (XmlNode node in nodeList)
        temp.Add
```

```

        (
        new Person
        (
            node[ "name" ].InnerText, node[ "phone" ].InnerText,
            node[ "email" ].InnerText
        )
    );
    return temp;
}

```

Tìm kiếm trên tài liệu:

Thông thường việc tìm kiếm được tiến hành một cách đệ quy như sau

```

static public void Search(XmlNode root, string find, ref XmlNode result)
{
    if (root.NodeType == XmlNodeType.Text)
        if (root.InnerText == find)result = root;
    foreach (XmlNode n in root.ChildNodes)
        Search(n, name, ref result);
}

```

Trong đó *find* là chuỗi cần tìm, nếu tìm được hàm này trả về ***XmINode*** chứa chuỗi *find*, ngược lại hàm trả về giá trị *null*. Đoạn mã dưới đây minh họa cách sử dụng hàm này.

```

 XmlDocument doc = new XmlDocument();
 doc.Load(inputFilePath);
 XmlNode root = doc.SelectSingleNode("contacts");
 XmlNode result = null;
 Search(root, "Nguyen Thi Hai",ref result);
 if (result == null) Console.WriteLine("khong thay");
 else Console.WriteLine(result.InnerText);

```

Thêm phần tử vào tài liệu:

Về nguyên tắc để thêm 1 phần tử vào tài liệu ta thực hiện các bước sau:

- Tham chiếu đến tài liệu *xDoc*
- Tạo phần tử con *subNode*, và bổ sung các thuộc tính nếu có
- Tạo các *TextNode* cho *subNode*
- Bổ sung các *TextNode* vào *subNode*
- Bổ sung *subNode* vào tài liệu *xDoc*
- Lưu tài liệu

Hàm sau đây tạo ra một phần tử “person” mới, thuộc tính *category* nhận giá trị truyền vào từ tham số *cat*. Các *TextNode* nhận giá trị tương ứng là các tham số: *name*, *phone* và *email*

```

static public void InsertElement(string cat, string name, string phone, string email)
{
    XmlDocument xmlDoc = new XmlDocument();
    xmlDoc.Load(inputFilePath);
    //tạo một phần tử "person" mới:
   XmlElement elPerson = xmlDoc.CreateElement("person");

```

```

//bổ sung thuộc tính "category":
elPerson.SetAttribute("category", cat);
//tạo các phần tử "name", "phone" và "email":
XmlElement elName = xmlDoc.CreateElement("name");
XmlElement elPhone = xmlDoc.CreateElement("phone");
XmlElement elEmail = xmlDoc.CreateElement("email");
//tạo các text node:
XmlText txtName = xmlDoc.CreateTextNode(name);
XmlText txtPhone = xmlDoc.CreateTextNode(phone);
txtEmail = xmlDoc.CreateTextNode(email);
//thêm các textnode vào "name", "phone", "email":
elName.AppendChild(txtName);
elPhone.AppendChild(txtPhone);
elEmail.AppendChild(txtEmail);
//thêm phần tử vừa tạo vào node "person" mới:
elPerson.AppendChild(elName);
elPerson.AppendChild(elPhone);
elPerson.AppendChild(elEmail);
//lấy node gốc:
XmlElement elRoot = xmlDoc.DocumentElement;
//thêm "person" vào gốc:
elRoot.AppendChild(elPerson);
//lưu document:
xmlDoc.Save(outputFilePath);
}

```

Cập nhật một phần tử trong tài liệu:

Việc cập nhật và xóa các phần tử trong tài liệu sẽ dựa trên thao tác tìm kiếm. Nếu tìm được:

- Cập nhật giá trị mới
- Lưu tài liệu.

```

static public bool UpdateElement(string oldValue, string newValue)
{
    try
    {
        XmlDocument doc = new XmlDocument();
        doc.Load(inputFilePath);
        //tìm node cần cập nhật:
        XmlNode root = doc.DocumentElement;//lấy node gốc
        XmlNode result = null;
        Search(root, oldValue, ref result);
        if (result != null)
        {
            result.InnerText = newValue;
            doc.Save(inputFilePath);
            return true;
        }
        else return false;
    }
    catch
    {
        return false;
    }
}

```

Xóa phần tử:

Các bước để xóa một phần tử:

- Tìm kiếm phần tử cần xóa (giả sử phần tử tìm được là result).
- Xác định phần tử cha của x, bằng thuộc tính result.ParentNode
- Gọi phương thức RemoveChild từ phần tử cha này, với tham số là result

```
static public bool Delete(string find)
{
    try
    {
        XmlDocument doc = new XmlDocument();
        doc.Load(inputFilePath);
        //tìm node cần cập nhật:
        XmlNode root = doc.DocumentElement;//lấy node gốc
        XmlNode result = null;
        Search(root, find, ref result);
        if (result != null)
        {
            result.ParentNode.RemoveChild(result); //xóa result
            doc.Save(inputFilePath);
            return true;
        }
        else return false;
    }
    catch
    {
        return false;
    }
}
```

Chú ý: Để có hướng dẫn đầy đủ hơn về các thao tác với *XmlDocument*, bạn nên tham khảo trong các bài Lab tương ứng.

14.5 Sử Dụng Linq To Xml

Bạn đã nắm được cách sử dụng *LINQ* trong các chương trước. Ở đây chúng tôi chỉ tập trung trình bày về *LINQ to XML*. Tức là sử dụng *LINQ* để thực hiện các thao tác (tạo, đọc, thêm, xóa, sửa,...) trên tài liệu *XML*.

LINQ to XML sử dụng các lớp *XDocument*, *XElement*, và *XAttribute* để thao tác với tài liệu *XML*.

Để thuận tiện, chúng tôi vẫn sử dụng file *XML* "XMLFile.xml" ở phần trên để minh họa:

```
<?xml version="1.0" encoding="utf-8"?>
<contacts>
    <person category="personal">
        <id>1</id>
        <name>Tran Van Mot</name>
        <phone>111111111111</phone>
        <email>mot@gmail.com</email>
    </person>
    <person category="business">
        <id>2</id>
        <name>Nguyen Thi Hai</name>
        <phone>2222222222</phone>
```

```
<email>hai@yahoo.com</email>
</person>
<person category="family">
<id>3</id>
<name>Bui Dinh Ba</name>
<phone>3333333333</phone>
<email>ba@gmail.com</email>
</person>
<person category="friend">
<id>4</id>
<name>Nguen dinh bon</name>
<phone>123456</phone>
<email>bon@gmail.com</email>
</person>
<person category="friend">
<id>5</id>
<name>Triệu Tử Long</name>
<phone>888888888</phone>
<email>tulong@yahoo.com</email>
</person>
<person category="friend">
<id>6</id>
<name>Lôi Quốc Cường</name>
<phone>656565646</phone>
<email>quoccuong@gmail.com</email>
</person>
</contacts>
```

Vì việc đầu tiên là chúng ta cần khai báo thư viện khi sử dụng *LINQ to XML*:

```
using System.Xml.Linq;
```

Kế tiếp ta xây dựng một lớp riêng trong file “LINQ2XML.cs” để thực hiện các thao tác trên tài liệu này.

14.5.1 Tham chiếu đến file tài liệu

Đối tượng *XElement* sử dụng phương thức *Load* để tham chiếu đến file tài liệu.

Giả sử đường dẫn vật lý của file tài liệu như sau:

```
string path = HttpContext.Current.Server.MapPath(@"~/App_Data/XMLFile.xml");
```

Khi đó, để load tài liệu lên đối tượng xml của lớp *Xelement* ta sử dụng phương thức:

```
xml = XElement.Load(path);
```

Kể từ lúc này các thao tác sẽ được thực hiện trên đối tượng xml.

14.5.2 Lưu tài liệu

Khi muốn lưu nội dung của đối tượng xml vào file tài liệu, ta gọi phương thức

```
xml.Save(path);
```

14.5.3 Tham chiếu đến các phần tử trong tài liệu

LINQ to XML cung cấp 2 phương thức Elements và Element (không có s) để tham chiếu đến các phần tử trong tài liệu, tuy nhiên việc sử dụng các phương thức còn phụ thuộc vào cấu trúc của chính tài liệu này. Ví dụ với tài liệu mẫu ở trên, nếu:

```
var contacts = xml.Elements("person");
```

Thì contacts chứa tất cả các phần tử **person** của tài liệu.

Hơn nữa với mỗi phần tử trong contacts để truy xuất đến phần tử con của chúng, ta sử dụng phương thức Element như sau:

```
var contacts = xml.Elements("person");
var t = contacts.First();
var name = t.Element("name").Value.ToString();
var phone = t.Element("phone").Value.ToString();
var email = t.Element("email").Value.ToString();
```

Khi đó biến name sẽ chứa chuỗi Tran Van Mot, biến phone sẽ chứa chuỗi 111111111111 và biến email sẽ chứa chuỗi mot@gmail.com

14.5.4 Tham chiếu đến thuộc tính

LINQ to XML cũng cung cấp các phương thức để truy xuất đến thuộc tính của các phần tử. Ví dụ nếu:

```
var contacts = xml.Elements("person");
var t = contacts.First();
var attribute = t.Attribute("category").Value.ToString();
```

thì biến attribute sẽ nhận giá trị **personal**

14.5.5 Thêm phần tử vào tài liệu

Để thêm một phần tử, trước tiên ta phải xác định được nút cha của nó, sau đó sử dụng phương thức Add đối với phần tử cha này với tham số là phần tử cần thêm.

Ví dụ:

```
//tạo phần tử mới
 XElement per = new XElement
(
    "person",
    new XAttribute("category",cat), //thêm thuộc tính
    new XElement("id", p.Id), //thêm các phần tử con
    new XElement("name", p.Name),
    new XElement("phone", p.Phone),
    new XElement("email", p.Email)
);
//thêm vào tài liệu
xml.Add(per);
```

14.5.6 Tìm kiếm phần tử

Ví dụ sau đây sẽ tìm tất cả các phần tử có **category** thỏa mãn tham số đầu vào và hiển thị phần tử **name** trên trình duyệt:

```
public void DisplayElements(string category)
{
    var contacts = xml.Elements("person");
    var result = from c in contacts
    where c.Attribute("category").Value.ToString() == category
    select c;
    foreach (var re in result)
    {
        HttpContext.Current.Response.Write(re.Element("name").Value.ToString()+"<br/>");
    }
}
```

Trường hợp tìm kiếm chính xác *một* phần tử trong tài liệu:

```
/// <summary>
/// Tìm kiếm phần tử khi biết Id
/// </summary>
/// <param name="id">Id</param>
/// <returns>null = không tìm được, đối tượng Person = tìm được</returns>
public Person Search(string id)
{
    try
    {
        var contacts = xml.Elements("person");
        var person =
        (
            from p in contacts
            where p.Element("id").Value.ToString() == id
            select p
        ).Single();
        return new Person
        (
            person.Element("id").Value.ToString(),
            person.Element("name").Value.ToString(),
            person.Element("phone").Value.ToString(),
            person.Element("email").Value.ToString()
        );
    }
    catch
    {
        return null;
    }
}
```

14.5.7 Cập nhật phần tử

Việc cập nhật các giá trị của các phần tử thường dựa trên thao tác tìm kiếm:

Ví dụ:

```

/// <summary>
/// Sửa phần tử trong tài liệu (bao gồm cả thuộc tính category)
/// </summary>
/// <param name="p">đối tượng mới</param>
/// <param name="category">thuộc tính</param>
/// <returns>true = thành công, false = thất bại</returns>
public bool Update(Person p, string category)
{
    try
    {
        Person find = this.Search(p.Id); //tìm phần tử cần sửa
        if (find == null) return false;

        var contacts = xml.Elements("person");
        var person =
        (
            from per in contacts
            where per.Element("id").Value.ToString() == p.Id
            select per
        ).Single();
        //update:
        person.Attribute("category").Value = category;
        person.Element("name").Value = p.Name;
        person.Element("phone").Value = p.Phone;
        person.Element("email").Value = p.Email;
        this.Save();
        return true;
    }
    catch
    {
        return false;
    }
}

```

14.5.8 Xóa phần tử

Tương tự việc cập nhật, để xóa phần tử chúng ta cũng sẽ sử dụng các thao tác duyệt qua các phần tử để tìm kiếm phần tử cần xóa.

```

/// <summary>
/// Xóa phần tử trong tài liệu
/// </summary>
/// <param name="id">Id của phần tử cần xóa</param>
/// <returns>true = thành công, false = thất bại</returns>
public bool Delete(string id)
{
    try
    {
        Person find = this.Search(id);
        if (find == null) return false;
        var contacts = xml.Elements("person");
        var person =
        (

```

```

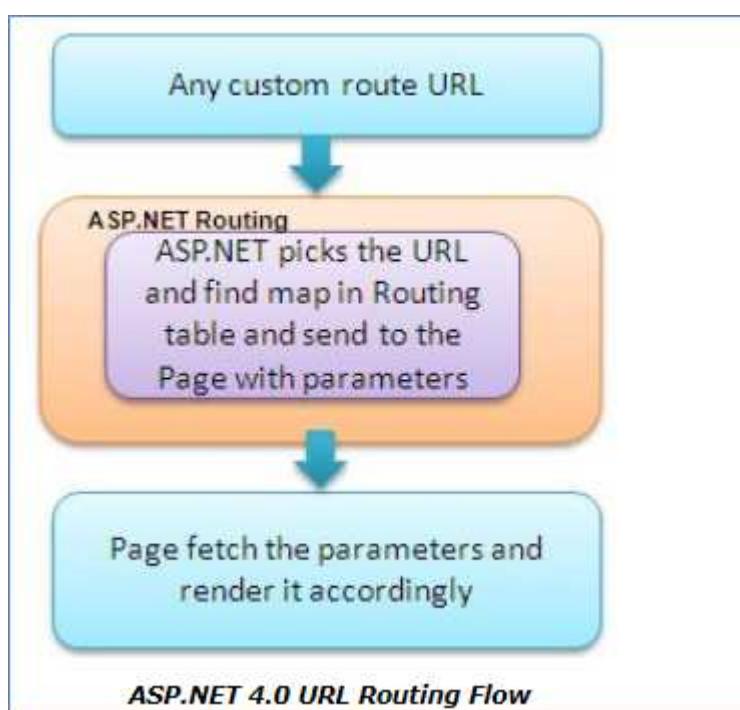
        from per in contacts
        where per.Element("id").Value.ToString() == id
        select per
    ).Single();
//xóa:
person.Remove();
this.Save();
return true;
}
catch
{
    return false;
}
}

```

Chú ý: Thực hành chi tiết về các thao tác với tài liệu XML trong các bài labs.

14.6 Giới thiệu URL Routing

- URL định tuyến(URL Routing) là một kỹ thuật tiên được giới thiệu với ASP.NET 3.5 SP1, và đã được sử dụng trong các ứng dụng ASP.NET MVC để mô tả URL một cách rõ ràng và thân thiện với SEO(**Search Engine Optimization**) "web 2.0"
- URL định tuyến cho phép bạn cấu hình một ứng dụng để chấp nhận các URL yêu cầu không lập bản đồ cho các tập tin vật lý. Thay vào đó, chúng ta có thể sử dụng định tuyến để xác định các URL có ngữ nghĩa , có ý nghĩa cho người sử dụng và có thể giúp tối ưu hóa công cụ tìm kiếm (SEO).



- Ví dụ, các URL cho một trang truyền thống hiển thị các loại sản phẩm có thể trông giống như dưới đây:

<http://www.mysite.com/products.aspx?category=software>

Sử dụng cơ chế định tuyến URL trong ASP.NET 4 bây giờ chúng ta có thể cấu hình các ứng dụng để chấp nhận các URL sau đây :

<http://www.mysite.com/products/software>

- Như vậy , mục đích URL Routing nhằm làm cho URL được ngắn hơn , dễ dàng sử dụng, làm sáng tỏ cấu trúc của Website. Ngoài ra, các URL giúp SEO (Search Engine Optimization) cải thiện các số truy cập trang nhưng đặt trong các từ khóa thích hợp.

Ứng dụng URL Routing ASP.NET WebForm

Bước 1.

1.1 Trên SQL Server 2008 tạo 1 cơ sở dữ liệu tên Manager gồm 1 bảng tên Products (ProductID int Primary Key, ProductName varchar(50))

dbo.Products: Table(ibm.Manager)			
	Column Name	Data Type	Allow Nulls
1	ProductID	int	<input type="checkbox"/>
	ProductName	varchar(50)	<input type="checkbox"/>

1.2 Tạo ứng dụng ASP.NET đặt tên MyRoutingApp, gồm trang ViewProducts.aspx và ViewProductDetailByProductID.aspx.

Chúng ta sẽ cấu hình routing cho trang ViewProductDetailByProductID.aspx. Khi người vào trang ViewProduct.aspx nhấp vào link để xem chi tiết mặt hàng có MaMH (ở đây là cột ProductID) = 1, thay vì chúng ta khai báo URL là :

http://server_name/ MyRoutingApp /ViewProductDetailByProductID.aspx?MaMH=1

Bây giờ chúng ta có thể khai báo như sau :

http://server_name/ MyRoutingApp/ChiTietMatHang/MaMH=1

Bước 2. Cấu hình Routing

2.1 Thêm vào ứng dụng tập tin Global.asax và viết code như sau :

```

<%@ Application Language="C#" %>
<%@ Import Namespace="System.Web.Routing" %>
<script runat="server">
    void RegisterRoutes(RouteCollection routes)
    {
        //khai báo định tuyến cho trang ViewProductDetailByProductID.aspx
        routes.MapPageRoute(
            "productDetail-browse",           // tên định tuyến
            "ChiTietMatHang/{MaMH}",          // Khai báo tham số cho URL
            "~/ViewProductDetailByProductID.aspx" // Tên trang web
        );
        //thêm định tuyến cho các trang khác ở đây
    }

```

```
//....  
}  
void Application_Start(object sender, EventArgs e)  
{  
    //Đăng ký định tuyến  
    RegisterRoutes(RouteTable.Routes);  
}  
</script>
```

2.2 Thêm vào trang ViewProducts.aspx 2 Hyperlink controls và khai báo thuộc tính

NavigateUrl như sau :

```
<form id="form1" runat="server">  
    <div>  
        <asp:HyperLink ID="h11"  
            NavigateUrl="ChiTietMatHang/1" runat="server">  
            View Product Detail By ProductID = 1  
        </asp:HyperLink>  
        <br />  
        <asp:HyperLink ID="h12"  
            NavigateUrl="ChiTietMatHang/2" runat="server">  
            View Product Detail By ProductID = 2  
        </asp:HyperLink>  
    </div>  
</form>
```

2.3 Thêm vào trang ViewProductDetailByProductID.aspx 1 GridView control có thuộc tính **ID** là **gvProducts** và viết code cho Page_Load như sau :

```
protected void Page_Load(object sender, EventArgs e)  
{  
    //lay ma mat hang tu URL  
    int ProID = int.Parse(Page.RouteData.Values["MaMH"].ToString());  
    //Lay thong tin mat hang co ma mat hang la ProID  
    string chuoiketNoi =  
        "server=swordlake\\sql2k8;database=manager;uid=sa;pwd=123";  
    string cauLenhSQL = "select ProductID,ProductName from Products "+  
        " where ProductID = "+ProID;  
    SqlDataAdapter da = new SqlDataAdapter(cauLenhSQL, chuoiketNoi);  
    DataTable dtPro = new DataTable();  
    da.Fill(dtPro);  
    //hien thi tren GridView  
    gvProducts.DataSource = dtPro;  
    gvProducts.DataBind();  
}
```

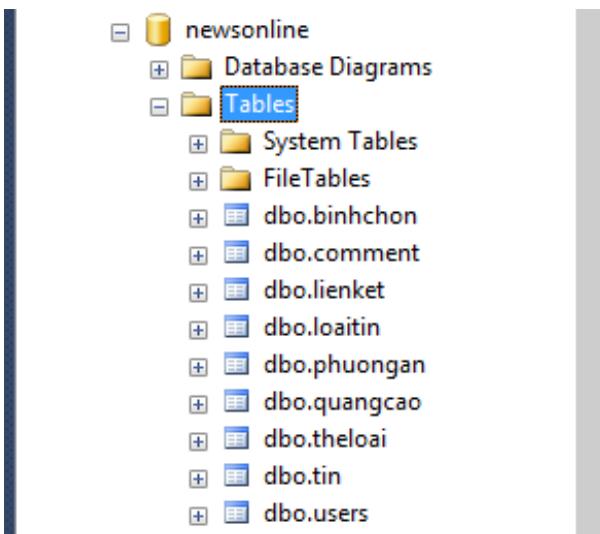
Bước 3. Nhấn Ctrl+F5 chạy trang ViewProducts.aspx và nhấp vào các link để xem kết quả.



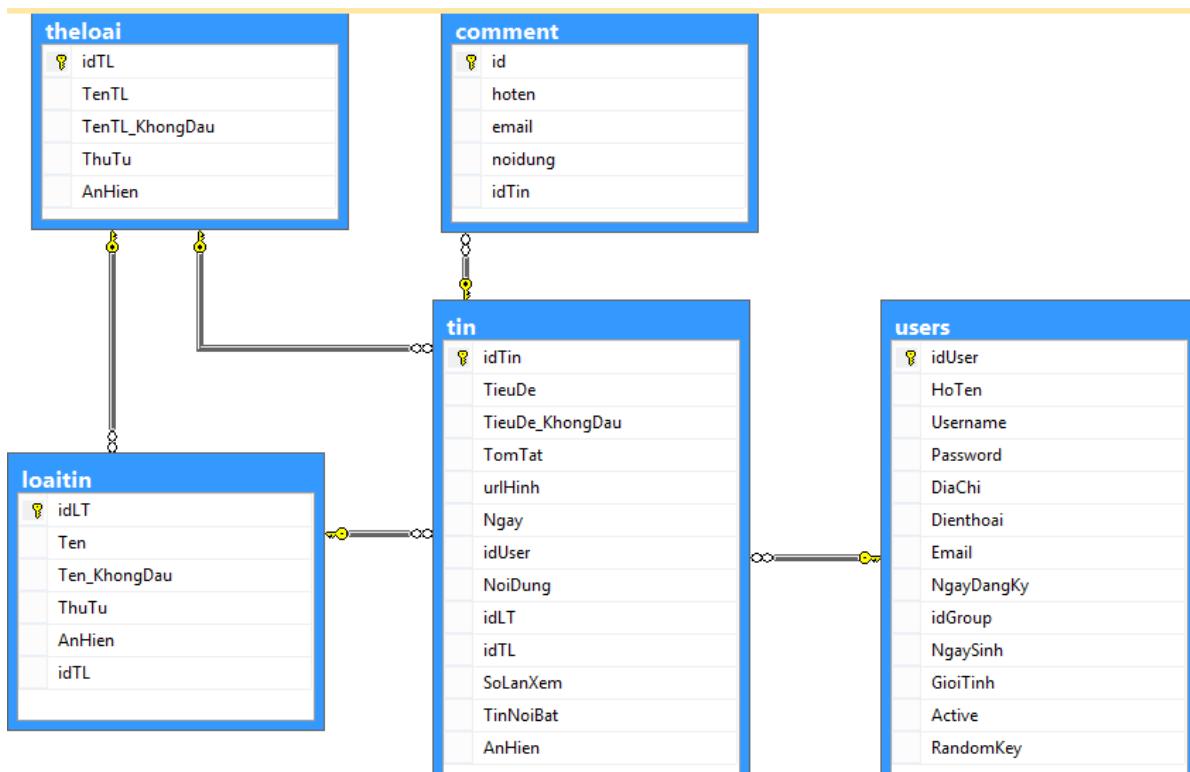
Chương 15: WEB TIN TỨC

15.1 Database

Database sau khi restore:



Mô tả:



15.2 Tạo website tin tức

15.2.1 Tạo mới Website

Chọn **File->New->Web Site**. Chọn **Visual C#->ASP.NET Empty Web Site**. Đặt tên website là **webtintuc** và chọn nơi lưu webtintuc. Sau đó click **OK** để tạo.

15.2.2 Tạo kết nối database

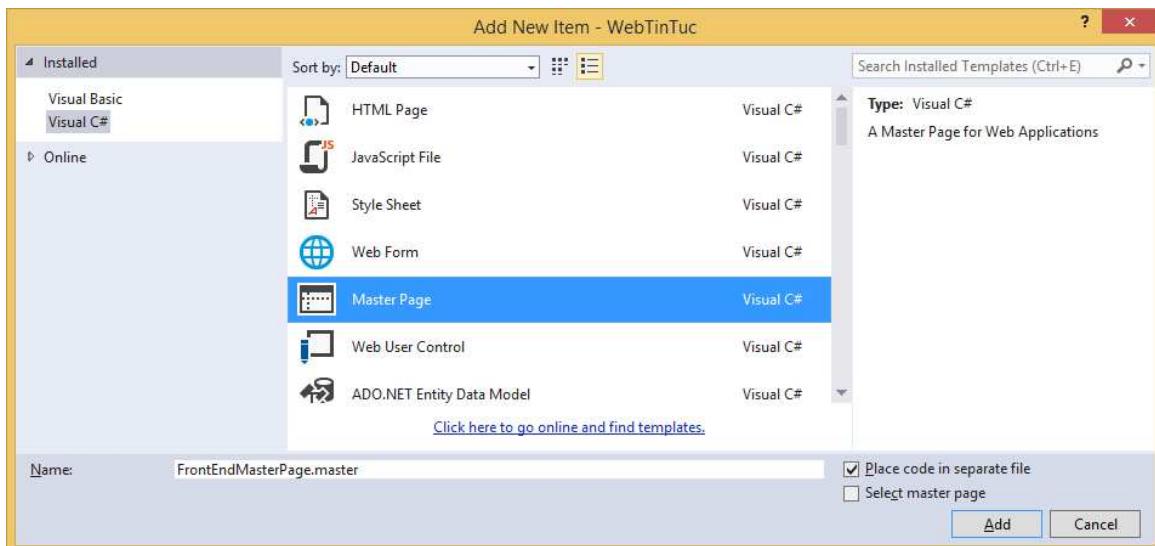
Mở trang **Web.config** thêm vào đoạn code sau:

```
<connectionStrings>
    <add name="webtintuc" connectionString="Data Source=.;Initial
Catalog=newsonline;Integrated Security=True"/>
</connectionStrings>
```

15.3 Viết code trang chủ

15.3.1 Tạo trang Master Page

Phải chuột vào **webtintuc**, chọn **Add -> Add New Item...** Chọn **Master Page** và đặt tên **FrontEndMasterPage**. Sau đó click **Add**.



15.3.2 Thiết kế giao diện

Mẫu giao diện thiết kế:



Thực hiện:

Bước 1 : Thêm đoạn code sau ở dưới thẻ **<body>** của trang **FrontEndMasterPage.master**

```
<div id="wrraper">
    <div id="no_Top">
        <a href="Default.aspx" title="Home">
            
        </a>
    </div>
    <div id="no_left">
        <div class="blocks">
            <div class="blocks_title">Danh mục</div>
```

```

<div class="blocks_content">Nội dung</div>
</div>
<div class="blocks">
    <div class="blocks_title">Đăng nhập</div>
    <div class="blocks_content">Nội dung</div>
</div>
</div>
<div id="no_Content">
    <asp:ContentPlaceHolder id="ContentPlaceHolderContent" runat="server">
    </asp:ContentPlaceHolder>
    </div>
    <div id="no_Right">
        <div class="blocks">
            <div class="blocks_title">Tin mới nhất</div>
            <div class="blocks_content">Nội dung</div>
        </div>
        <div class="blocks">
            <div class="blocks_title">Tin xem nhiều</div>
            <div class="blocks_content">Nội dung</div>
        </div>
    </div>
    <div id="no_Footer">
        Footer
    </div>
</div>

```

Bước 2 : Tạo **Layout1.css** nằm trong thư mục **layout**

Phải chuột vào thư mục **layout**, chọn **Add -> Add New Item...** Chọn **Style Sheet**, đặt tên **Layout1**. Sau đó click **Add**.

Bước 3 : Nhúng **Layout1.css** vào trang **FrontEndMasterPage** :

Thêm đoạn code sau vào bên dưới thẻ **<head>** trang **FrontEndMasterPage.master** :

```
<link href="layout/Layout1.css" rel="stylesheet" />
```

Bước 4 : Thêm đoạn code sau vào trang **Layout1.css**

```

body {font-family:Tahoma, Arial, Verdana; font-size:11px; padding:0px; margin:0px;
color:#222;}
#rraper{width:980px; margin:auto; padding:0px;}
#no_Top{width:978px; height:119px;text-align:center; border:solid 1px #2b6d8f;}
#no_Breadcrumb{width:100%; height:30px; background-color:Orange; text-align:center;}
#no_Left{width:180px; min-height:640px; float:left;padding-top:9px;}
#no_Content{width:600px; min-height:640px; float:left; margin-left:10px; margin-right:10px; padding-top:9px; text-align:justify;}
#no_Right{width:180px; min-height:640px; float:left;padding-top:9px;}
#no_Footer{width:100%; min-height:40px; background-color:Orange; clear:both; text-align:center;}
/* Blocks */
.blocks{width:100%;margin-bottom:5px;}
.blocks_title{background-color:Blue; color:white; padding:5px; font-weight:bold; text-transform:uppercase;}
.blocks_content{border:solid 1px blue; padding:5px;}

```

15.3.3 Tạo các hàm xử lý :

Bước 1 : Phải chuột vào **webtintuc** , chọn **Add -> Add New Item...** Chọn **Class** , đặt tên **XuLy.cs**. Sau đó click **Add** và thêm các thư viện vào trang **XuLy.cs**

```
using System.Data;
using System.Data.SqlClient;
using System.Web.Configuration;
using System.Text;
using System.Security;
using System.Security.Cryptography;
```

Bước 2 : Thêm các hàm sau vào trang **XuLy.cs**

```
//Kết nối database
public static string connectNewsOnline()
{
    return WebConfigurationManager.ConnectionStrings["webtintuc"].ConnectionString;
}
```

```
//Xử lý câu lệnh Select
public static DataTable Xuly_Select(string sql)
{
    string connect = connectNewsOnline();
    SqlConnection con = new SqlConnection(connect);
    SqlDataAdapter adapt = new SqlDataAdapter(sql, con);
    DataTable dt = new DataTable();
    adapt.Fill(dt);
    return dt;
}
```

```
//Xử lý truy vấn Insert update delete
public static bool XulyTruyvan(string sql)
{
    try
    {
        string connect = connectNewsOnline();
        SqlConnection con = new SqlConnection(connect);
        SqlCommand cmd = new SqlCommand(sql, con);
        cmd.Connection.Open();
        cmd.ExecuteNonQuery();
        cmd.Connection.Close();
        return true;
    }
    catch
    {
        return false;
    }
}
```

```
//Mã hóa MD5
public static string GetMd5Hash(string input)
{
    MD5 md5Hash = MD5.Create();

    // Convert the input string to a byte array and compute the hash.
    byte[] data = md5Hash.ComputeHash(Encoding.UTF8.GetBytes(input));

    // Create a new StringBuilder to collect the bytes
    // and create a string.
    StringBuilder sBuilder = new StringBuilder();

    // Loop through each byte of the hashed data
    // and add its ASCII character representation to the string.
    for (int i = 0; i < data.Length; i++)
    {
        sBuilder.Append(data[i].ToString("X2"));
    }
    return sBuilder.ToString();
}
```

```

// and format each one as a hexadecimal string.
for (int i = 0; i < data.Length; i++)
{
    sBuilder.Append(data[i].ToString("x2"));

}

// Return the hexadecimal string.
return sBuilder.ToString();

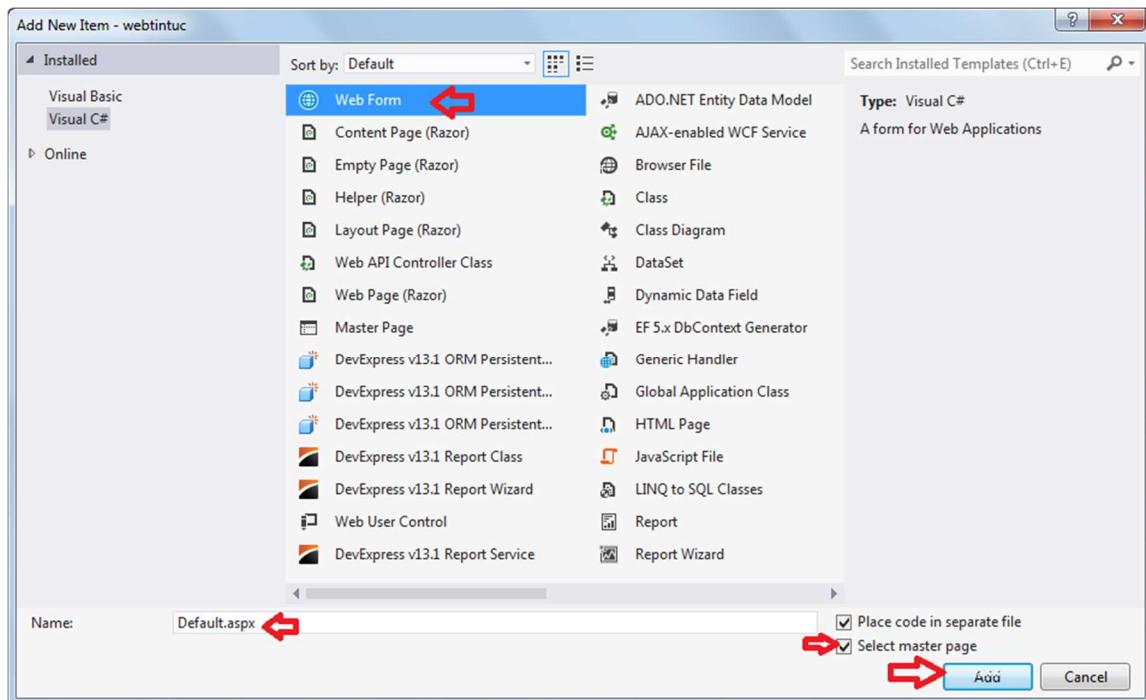
}

//Hàm random key
public static string RandomNumber()
{
    string[] mang = { "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "1", "2",
"3", "4", "5", "6", "7", "8", "9" };
    string x = "";
    Random r = new Random();
    int n;
    for (int i = 1; i <= 30; i++)
    {
        n = r.Next(0, mang.Count() - 1);
        x = x + mang[n];
    }
    return x;
}

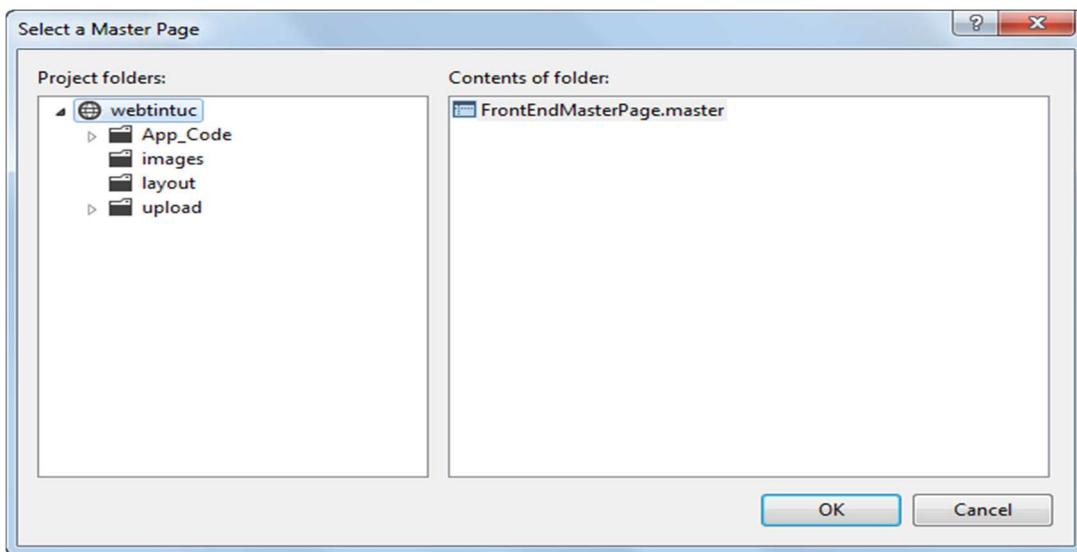
```

15.4 Tạo trang Default.aspx:

Bước 1 : Phải chuột vào **webtintuc** chọn **Add -> Add New Item...** Chọn **Web Form**, đặt tên **Default.aspx**, check vào ô **Select master page**. Sau đó click **Add**



Chọn **FrontEndMasterPage** và click **OK** ở hộp thoại tiếp theo



Bước 2 : Thêm đoạn code sau vào trang **Default.aspx** (phần Content2 – nội dung)

```
<asp:Content ID="Content2" ContentPlaceholderID="ContentPlaceHolderContent"
Runat="Server">
    <asp:ListView ID="lvTheLoai" runat="server"
onitemdatabound="lvTheLoai_ItemDataBound">
        <ItemTemplate>
            <table>
                <tr>
                    <td colspan="2" class="title">
                        <asp:HiddenField ID="hidIdTL" runat="server" Value='<%# Eval("idTL") %>' />
                        <span class="tc_TenTL"><%# Eval("TenTL") %></span>
                        <asp:ListView ID="lvLoaitin" runat="server">
                            <ItemTemplate>
                                <span class="tc_LoaiTin">
                                    | <a href='TinTrongLoai.aspx?idLT=<%# Eval("idLT") %>'><%# Eval("Ten") %></a>
                                    </span>
                                </ItemTemplate>
                            </asp:ListView>
                        </td>
                    </tr>
                    <tr>
                        <td width="50%">
                            <asp:ListView ID="lvTinTrai" runat="server">
                                <ItemTemplate>
                                    <a href='ChiTietTin.aspx?idTin=<%# Eval("idTin") %>'><%# Eval("TieuDe") %></a><br />
                                    <img src='upload/tintuc/<%# Eval("urlHinh") %>' alt="" height="100px" hspace="0" vspace="5" align="left" class="img" />
                                    <span class="tomtat">
                                        <%# Eval("TomTat") %>
                                    </span>
                                </ItemTemplate>
                            </asp:ListView>
                        </td>
                        <td width="50%" class="linkPhai">
                            <asp:ListView ID="lvTinPhai" runat="server">
                                <ItemTemplate>
                                    <div>
```

```

<font style="color:green">&raquo;</font> <a
href='ChiTietTin.aspx?idTin=<%# Eval("idTin") %>'><%# Eval("TieuDe") %></a>
        </div>
    </ItemTemplate>
</asp:ListView>
</td>
</tr>
</table>
</ItemTemplate>
</asp:ListView>
</asp:Content>

```

Bước 3 : Thêm đoạn code sau vào trang **Default.aspx.cs**

```

protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        DataTable dtTheLoai = new DataTable();
        dtTheLoai = XuLy.Xuly_Select("SELECT * FROM theloai");
        lvTheLoai.DataSource = dtTheLoai;
        lvTheLoai.DataBind();    }
}

```

```

protected void lvTheLoai_ItemDataBound(object sender, ListViewItemEventArgs e)
{
    HiddenField hidTL = (HiddenField)e.Item.FindControl("hidIdTL");
    DataTable dtLoaitin = new DataTable();
    string sql = string.Format(@"SELECT * FROM loaitin WHERE idTL='{0}'", hidTL.Value);
    ListView lvLoaitin = (ListView)e.Item.FindControl("lvLoaitin");
    dtLoaitin = XuLy.Xuly_Select(sql);
    lvLoaitin.DataSource = dtLoaitin;
    lvLoaitin.DataBind();
    //Tin Trái
    DataTable dtTinTrai = new DataTable();
    string sqlTin = string.Format(@"SELECT TOP 1 * FROM tin WHERE idTL='{0}' ORDER BY
idTin DESC", hidTL.Value);
    dtTinTrai = XuLy.Xuly_Select(sqlTin);
    ListView lvTinTrai = (ListView)e.Item.FindControl("lvTinTrai");
    lvTinTrai.DataSource = dtTinTrai;
    lvTinTrai.DataBind();
    //Lấy idTin
    DataRow rowIdTin = dtTinTrai.Rows[0];
    //Tin Phải
    DataTable dtTinPhai = new DataTable();
    string sqlTinPhai = string.Format(@"SELECT TOP 5 * FROM tin WHERE idTL='{0}' AND
idTin!={1} ORDER BY idTin DESC", hidTL.Value, rowIdTin["idTin"].ToString());
    dtTinPhai = XuLy.Xuly_Select(sqlTinPhai);
    ListView lvTinPhai = (ListView)e.Item.FindControl("lvTinPhai");
    lvTinPhai.DataSource = dtTinPhai;
    lvTinPhai.DataBind();
}
}

```

Bước 4 : Thêm đoạn code sau vào **Layout1.css**

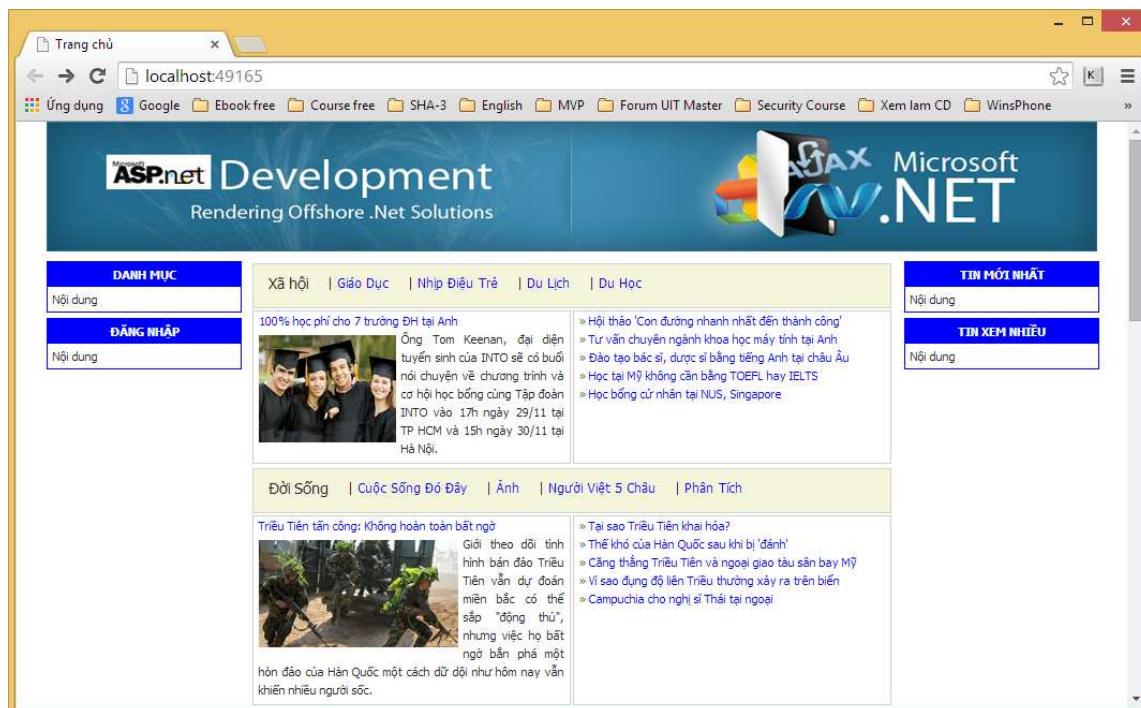
```

.tc_TenTL{float:left; margin:9px; font-size:14px;}
.tc_LoaiTin{float:left; margin:9px; font-size:12px;}
table tr th{border:solid 0px #fff;}

```

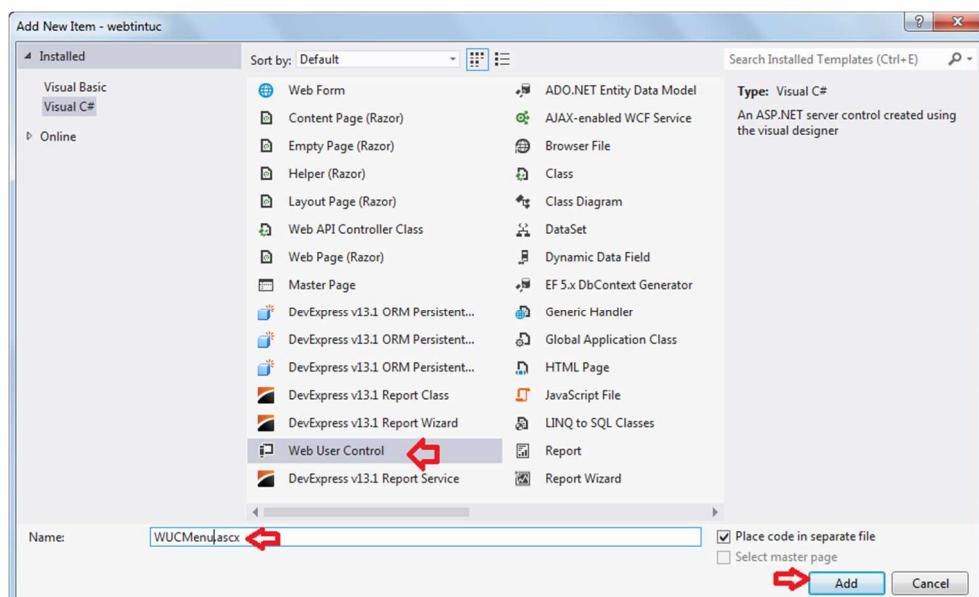
```
td{ vertical-align:top; line-height:17px;text-align:justify; margin:5px; border:solid 1px #d2d1d0;min-height:18px; padding:0 5px 5px 5px; }
}
table a{text-decoration:none; color:Blue;}
table a:hover{color:red;}
table. title{background-color:Beige; }
table. tomtat{}
table. img{padding-right:5px;}
```

Kết quả :



15.5 Tạo Web User Control Menu

Bước 1 : Phải chuột vào thư mục **blocks**, chọn **Add -> Add New Item...** Chọn **Web User Control** và đặt tên **WUCMenu**. Sau đó click **Add**



Bước 2 : Thêm đoạn code sau vào **WUCMenu.ascx**

```
<div id="no_menu">
    <asp:ListView ID="lvTheLoai" runat="server"
        onitemdatabound="lvTheLoai_ItemDataBound">
        <ItemTemplate>
            <asp:HiddenField ID="hidIdTL" runat="server" Value='<%# Eval("idTL") %>' />
            <div class="menu_theLoai"><%# Eval("TenTL") %></div>
            <div class="menu_nhom">
                <asp:ListView ID="lvLoaitin" runat="server">
                    <ItemTemplate>
                        <div class="menu_loaitin">
                            <a href='TinTrongLoai.aspx?idLT=<%# Eval("idLT") %>'><%# Eval("Ten")%></a>
                        </div>
                    </ItemTemplate>
                </asp:ListView>
            </div>
        </ItemTemplate>
    </asp:ListView>
</div>
```

Bước 3 : Thêm đoạn code sau vào WUCMenu.ascx.cs

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        lvTheLoai.DataSource = XuLy.Xuly_Select("SELECT * FROM theLoai");
        lvTheLoai.DataBind();
    }
}
```

```
protected void lvTheLoai_ItemDataBound(object sender, ListViewItemEventArgs e)
{
    HiddenField hidIdTL = (HiddenField)e.Item.FindControl("hidIdTL");
    string sql = string.Format(@"SELECT * FROM loaitin WHERE idTL='{0}'", hidIdTL.Value);
    ListView lvLoaitin = (ListView)e.Item.FindControl("lvLoaitin");
    lvLoaitin.DataSource = XuLy.Xuly_Select(sql);
    lvLoaitin.DataBind();
}
```

Bước 4 : Thêm đoạn code sau vào Layout1.css

```
.menu_theLoai{background-color:pink; color:Blue; padding:8px; font-weight:bold;
border:solid 1px red; cursor:pointer;}
.menu_loaitin a{display:block; background-color:Yellow; color:Blue; border-bottom: solid
1px grey; padding:5px; text-decoration:none;}
.menu_loaitin a:hover{background-color:Red;}
```

Bước 5 : Thêm đoạn code sau dưới thẻ <head> trang FrontEndMasterPage

```
<script src="jquery-1.8.2.min.js" type="text/javascript"></script>
<script>
    $(document).ready(function () {
        $(".menu_nhom").hide();
        $(".menu_theLoai").click(function () {
            $(this).next().slideToggle();
        });
    });
</script>
```

Bước 6 : Thêm đoạn code sau dưới <div id="no_Left"> trang FrontEndMasterPage

```
<div class="blocks">
    <div class="blocks_title">Danh mục</div>
```

```
<div class="blocks_content"><uc1:WUCMenu ID="WUCMenu1" runat="server" /></div>
```

Kết quả :

The screenshot shows a web page titled "ASP.net Development" with a banner for "AJAX Microsoft .NET". On the left, there's a sidebar with categories like Xã hội, Giáo Dục, Nhịp Điệu Trẻ, Du Lịch, Du Học, Đời Sống, Vị Tính, Văn học, Thể Giới, and Vị Tính. Below it is a "ĐĂNG NHẬP" button. The main content area features a news block with a thumbnail of three people in graduation gowns, the headline "100% học phí cho 7 trường ĐH tại Anh", and a summary. To the right, there are two columns of news snippets under "TIN MỚI NHẤT" and "TIN XEM NHIỀU".

15.6 Tạo Web User Control Tin mới nhất

Mô tả: Thiết kế block đọc 5 tin mới nhất.

The screenshot shows a news detail page with a header "TIN MỚI NHẤT". Below it is a list of five news items, each with a title and a short summary. An arrow points from the fifth news item to a text box containing the URL "ChiTietTin.aspx?idTin=xyz".

Thực hiện:

Bước 1 : Phải chuột vào thư mục **blocks**, chọn **Add -> Add New Item...** Chọn **Web User Control**, đặt tên **WUCTinMoiNhat**. Sau đó click **Add**

Bước 2 : Thêm đoạn code sau vào **WUCTinMoiNhat.ascx**

```
<div id="tinMoinhat">
<asp:ListView ID="lvTinmoi" runat="server">
```

```

<ItemTemplate>
    <a href='ChiTietTin.aspx?idTin=<%# Eval("idTin") %>'>
        <%# Eval("TieuDe") %>
    </a>
</ItemTemplate>
</asp:ListView>
</div>

```

Bước 3 : Thêm đoạn code sau vào **WUCTinMoiNhat.ascx.cs**

```

protected void Page_Load(object sender, EventArgs e)
{
    string sql = string.Format(@"SELECT TOP 5 * FROM tin ORDER BY idTin DESC");
    lvTinmoi.DataSource = XuLy.Xuly_Select(sql);
    lvTinmoi.DataBind();
}

```

Bước 4 : Thêm đoạn code sau vào **Layout1.css**

```

#tinMoinhat a{display:block;padding:4px; text-decoration:none; color:Blue; border-bottom: dashed 1px pink;}
#tinMoinhat a:hover{color:Red;}

```

Bước 5 : Thêm đoạn code sau vào dưới <div id="no_Right"> trang **FrontEndMasterPgae**

```

<div class="blocks">
    <div class="blocks_title">Tin mới nhất</div>
    <div class="blocks_content"><uc2:WUCTinMoiNhat ID="WUCTinMoiNhat1" runat="server" /></div>
</div>

```

15.7 Tạo Web User Control tin xem nhiều

Mô tả: Thiết kế block đọc 5 tin mới nhất.



Thực hiện:

Bước 1 : Phải chuột thư mục **blocks**, chọn **Add -> Add New Item...** Chọn **Web User Control**, đặt tên **WUCTinXemNieu**. Sau đó click **Add**

Bước 2 : Thêm đoạn code sau vào WUCTinXemNieu.ascx

```
<div id="tinXemnhieu">
    <asp:ListView ID="lvTinxemnhieu" runat="server">
        <ItemTemplate>
            <a href='ChiTietTin.aspx?idTin=<%# Eval("idTin") %>'>
                <%# Eval("TieuDe") %>(<b style="color:Red"><%# Eval("SoLanXem") %></b>)
            </a>
        </ItemTemplate>
    </asp:ListView>
</div>
```

Bước 3 : Thêm đoạn code sau vào WUCTinXemNieu.ascx.cs

```
protected void Page_Load(object sender, EventArgs e)
{
    string sql = string.Format(@"SELECT TOP 5 * FROM tin ORDER BY SoLanXem DESC");
    lvTinxemnhieu.DataSource = XuLy.Xuly_Select(sql);
    lvTinxemnhieu.DataBind();
}
```

Bước 4 : Thêm đoạn code sau vào Layout1.css

```
#tinXemnhieu a{display:block;padding:4px; text-decoration:none; color:Blue; border-bottom: dashed 1px pink;}
#tinXemnhieu a:hover{color:Red;}
```

Bước 5 : Thêm đoạn code sau dưới <div id="no_Right"> trang FrontEndMasterPage

```
<div class="blocks">
    <div class="blocks_title">Tin xem nhiều</div>
    <div class="blocks_content">
        <uc3:WUCTinXemNieu ID="WUCTinXemNieu1" runat="server" />
    </div>
</div>
```

15.8 Tạo Web User Control Login

Mô tả:

Màn hình đăng nhập	Đăng nhập sai username/password
	
Đăng nhập thành công (giới tính Nam)	Đăng nhập thành công (giới tính Nữ)
	

Thực hiện:

Bước 1 : Phải chuột thư mục **blocks**, chọn **Add -> Add New Item...** Chọn **Web User Control**, đặt tên **WUCLogin**. Sau đó click **Add**.

Bước 2 : Thêm đoạn code sau vào **WUCLogin.ascx**

```
<asp:MultiView ID="MultiViewLogin" runat="server">
    <asp:View ID="ViewFormLogin" runat="server">
        <div class="txtUser">
            <asp:TextBox ID="txtUser" runat="server" placeholder="Username"></asp:TextBox>
        </div>
        <div class="txtPass">
            <asp:TextBox ID="txtPassword" runat="server" placeholder="Password"
                TextMode="Password"></asp:TextBox>
        </div>
        <div>
            <asp:Button ID="btnLogin" runat="server" Text="Login"
                onclick="btnLogin_Click" />
        </div>
    </asp:View>
    <asp:View ID="ViewFormHello" runat="server">
        Hello
        <asp:Label ID="lblGioitinh" runat="server" Text=""></asp:Label>&ampnbsp
        <asp:Label ID="lblUser" runat="server"></asp:Label>
        <br />
        <asp:Button ID="btnLogout" runat="server" onclick="btnLogout_Click"
            Text="Logout" CssClass="btnLogin" />
    </asp:View>
</asp:MultiView>
```

Bước 3 : Thêm đoạn code sau vào **WUCLogin.ascx.cs**

```
using System.Data;
using System.Web.Configuration;
protected void Page_Load(object sender, EventArgs e)
{
    if (Session["idUser"] == null)
    {
        MultiViewLogin.ActivePageIndex = 0;
    }
    else
    {
        lblUser.Text = Session["HoTen"].ToString();
        if (int.Parse(Session["GioiTinh"].ToString()) == 0)
        {
            lblGioitinh.Text = " chị";
        }
        else
        {
            lblGioitinh.Text = " anh";
        }
        MultiViewLogin.ActivePageIndex = 1;
    }
}
```

```
protected void btnLogin_Click(object sender, EventArgs e)
{
    string userName = txtUser.Text;
    string password = XuLy.GetMd5Hash(txtPassword.Text);
    string sql = string.Format(@"SELECT * FROM users WHERE Username='{0}' AND
    Password='{1}'", userName, password);
    DataTable dtLogin = new DataTable();
```

```

dtLogin = XuLy.Xuly_Select(sql);
if (dtLogin.Rows.Count > 0)
{
    DataRow rowSession = dtLogin.Rows[0];
    Session["idUser"] = rowSession["idUser"].ToString();
    Session["HoTen"] = rowSession["HoTen"].ToString();
    Session["Username"] = rowSession["Username"].ToString();
    Session["idGroup"] = rowSession["idGroup"].ToString();
    Session["GioiTinh"] = rowSession["GioiTinh"].ToString();
    Response.Redirect(Request.RawUrl); //Refresh web
}
else
{
    txtUser.Focus();
    txtUser.Text = "";
    txtPassword.Text = "";
}
}

```

```

protected void btnLogout_Click(object sender, EventArgs e)
{
    Session.Clear();
    Response.Redirect(Request.RawUrl);
}

```

Bước 4 : Thêm đoạn code sau vào Layout1.css

```

.txtUser{margin:3px;}
.txtPass{margin:3px;}
.btnLogin{margin:3px;}

```

Bước 5 : Thêm đoạn code sau vào dưới <div id="no_Left"> trang FrontEndMasterPage

```

<div class="blocks">
    <div class="blocks_title">Đăng nhập</div>
    <div class="blocks_content">
        <uc4:WUCLogin ID="WUCLogin1" runat="server" />
    </div>
</div>

```

15.9 Tạo trang TinTrongLoai.aspx

Bước 1 : Phải chuột **webtintuc**, chọn **Add -> Add New Item...** Chọn **Web Form**, đặt tên **TinTrongLoai**, check vào ô **Select master page**. Sau đó click **Add** Chọn **FrontEndMasterPage** và click **OK** ở hộp thoại tiếp theo

Bước 2 : Thêm đoạn code sau vào TinTrongLoai.aspx

```

<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolderContent" Runat="Server">
    <div id="chitiettin">
        <span style="color:Blue;">Bạn đang xem tin: </span><asp:Label ID="lblLoaitin" runat="server" CssClass="lblTin"></asp:Label>
        <asp:GridView ID="gdvTintrongloai" runat="server" AutoGenerateColumns="False" PageSize="5" AllowPaging="true" onpageindexchanging="gdvTintrongloai_PageIndexChanging" BorderWidth="0px">
            <Columns>
                <asp:TemplateField>
                    <ItemTemplate>
                        <asp:HyperLink ID="hplLoaitin" runat="server" NavigateUrl='<%# Eval("idTin", "ChiTietTin.aspx?idTin={0}") %>' Text='<%# Eval("TieuDe") %>'></asp:HyperLink>
                    <br />
                </ItemTemplate>
            </asp:TemplateField>
        </Columns>
    </div>
</asp:Content>

```

```

<asp:Image ID="imgHinh" runat="server" ImageAlign="Left"
    ImageUrl='<%# Eval("urlHinh","upload/tintuc/{0}") %>' Width="150px"
    CssClass="imgLoaitin" />
    <asp:Label ID="lblTomtat" runat="server" Text='<%# Eval("TomTat") %>' 
    CssClass="lblTomtat"></asp:Label>
    </ItemTemplate>
    </asp:TemplateField>
</Columns>

<PagerSettings FirstPageText="" LastPageText="" />
<PagerStyle CssClass="phantrang" />

</asp:GridView>
<asp:Label ID="lblThongbao" runat="server" Text=""></asp:Label>
</div>
</asp:Content>

```

Bước 3 : Thêm đoạn code sau vào TinTrongLoai.aspx.cs

```

protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        string idLT = Request.QueryString["idLT"].ToString();
        this.Load_data(idLT);
    }
}

```

```

protected void gdvTintrongloai_PageIndexChanging(object sender, GridViewEventArgs e)
{
    gdvTintrongloaiPageIndex = e.NewPageIndex;
    string idLT = Request.QueryString["idLT"].ToString();
    this.Load_data(idLT);
}

```

```

public void Load_data(string idLT)
{
    if (idLT != null)
    {
        try
        {
            string sql = string.Format(@"SELECT * FROM tin WHERE idLT='{0}' ORDER BY idTin DESC",
                int.Parse(idLT));
            gdvTintrongloai.DataSource = XuLy.Xuly_Select(sql);
            gdvTintrongloai.DataBind();
            //Loai tin
            string sqlLoaitin = string.Format(@"SELECT * FROM loaitin WHERE idLT='{0}'",
                int.Parse(idLT));
            DataRow rowTen = XuLy.Xuly_Select(sqlLoaitin).Rows[0];
            lblLoaitin.Text = rowTen["Ten"].ToString();
        }
        catch (Exception)
        {
            Response.Redirect("Default.aspx");
        }
    }
    else
    {
        Response.Redirect("Default.aspx");
    }
}

```

Bước 4 : Thêm đoạn code sau vào Layout1.css

```
.imgLoaitin{padding-right:8px; margin-top:6px;}
.lblTomtat{text-align:justify; line-height:18px;}
.phantrang{color:Red; background-color:Yellow; text-align:center;}
.phantrang td table{margin:auto;}
.phantrang td table tr td{width:15px; text-align:center; padding:3px;}
.lblTin{color:Red; font-size:16px; font-weight:bold;}
```

Kết quả : Chọn danh mục **Vi tính -> Hacker & Virus**

The screenshot shows a news article titled "Hacker & Virus". The left sidebar has a red-highlighted menu item "Vi Tính". The right sidebar lists "TIN MỚI NHẤT" with several news items.

DANH MỤC	Bạn đang xem tin: Hacker & Virus	TIN MỚI NHẤT
Xã hội	Thế giới đứng trước nguy cơ chiến tranh áo	Phi cơ đầu tiên bay bằng sức người
Đời Sống	Các chuyên gia bảo mật tin rằng virus Stuxnet, thâm nhập vào một số máy tính trong chương trình hạt nhân của Iran, là một phần trong kế hoạch đe dọa tối hơn: khơi mào cuộc chiến tranh trên mạng.	Phi cơ trong suốt như pha lê
Vi Tính	Kaspersky Lab ra phiên bản tiếng Việt 2011	Một bệnh viện tại Mỹ sử dụng robot để vận chuyển thuốc, dụng cụ y tế và các loại hàng hóa nhằm giảm thiểu sai sót và chi phí hoạt động.
Văn học	Sản phẩm này gồm Kaspersky Anti-Virus 2011 và Kaspersky Internet Security 2011, được nâng cấp các giải pháp bảo vệ và cập nhật virus liên tục trong thời gian thực.	'Đĩa bay' có khả năng kéo tòa nhà lên trời
Đời sống	"KẾT NỐI AN TO	Phi cơ Mỹ sẽ bay trên sao Hỏa
Thế Giới		

15.10 Tạo trang ChiTietTin.aspx

Bước 1 : Phải chuột **webtintuc**, chọn **Add -> Add New Item...** Chọn **Web Form**, đặt tên **ChiTietTin**, check ô **Select master page**. Sau đó click **Add**. Chọn **FrontEndMasterPage** và click **OK** ở hộp thoại tiếp theo

Bước 2 : Thêm đoạn code sau vào **ChiTietTin.aspx**

```
<asp:Content ID="Content2" ContentPlaceholderID="ContentPlaceHolderContent" Runat="Server">
    <h2 id="tieudetin" runat="server"></h2>
    <div id="chitiet">
        <div class="content" runat="server" id="contentTin"></div>
    </div>
    <hr />
    <div>
        <ul class="nhomTinmoi"><h3>Tin cùng loại mới hơn</h3>
            <asp:ListView ID="lvTinmoihon" runat="server">
                <ItemTemplate>
                    <li>
                        <h3>
                            <a href='ChiTietTin.aspx?idTin=<%# Eval("idTin") %>'><%# Eval("TieuDe") %></a>
                        </h3>
                    </li>
                </ItemTemplate>
            </asp:ListView>
        </ul>
        <ul class="nhomTinmoi"><h3>Tin cùng loại cũ hơn</h3>
            <asp:ListView ID="lvTincuhon" runat="server">
                <ItemTemplate>
                    <li>
                        <h3>
                            <a href='ChiTietTin.aspx?idTin=<%# Eval("idTin") %>'><%# Eval("TieuDe") %></a>
                        </h3>
                    </li>
                </ItemTemplate>
            </asp:ListView>
        </ul>
    </div>
</asp:Content>
```

Bước 3 : Thêm đoạn code sau vào ChiTietTin.aspx.cs

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        int idTin = int.Parse(Request.QueryString["idTin"].ToString());
        string sql = string.Format(@"SELECT * FROM tin WHERE idTin='{0}'", idTin);
        DataRow rowTin = XuLy.Xuly_Select(sql).Rows[0];
        tieudetin.InnerHtml = rowTin["TieuDe"].ToString();
        contentTin.InnerHtml = rowTin["NoiDung"].ToString();
        //Tin cùng loại mới hơn
        string idLT = rowTin["idLT"].ToString();
        string sqlTinmoi = string.Format(@"SELECT TOP 5 * FROM tin WHERE idLT='{0}' AND idTin > '{1}'", idLT, idTin);
        lvTimmoihon.DataSource = XuLy.Xuly_Select(sqlTinmoi);
        lvTimmoihon.DataBind();
        //Tin cùng loại cũ hơn
        string sqlTincu = string.Format(@"SELECT TOP 5 * FROM tin WHERE idLT='{0}' AND idTin < '{1}' ORDER BY idTin DESC", idLT, idTin);
        lvTincuhon.DataSource = XuLy.Xuly_Select(sqlTincu);
        lvTincuhon.DataBind();
    }
}
```

Bước 4 : Thêm đoạn code sau vào Layout1.css

```
#chitiet{ line-height:18px; font-size:11px;}
.nhomTinmoi a{display:block;padding:4px; text-decoration:none; color:Blue; border-bottom: dashed 1px pink;}
.nhomTinmoi a:hover{color:Red;}
```

Kết quả : Xem một tin bất kỳ



Nhóm nghiên cứu cho rằng những thuật toán của họ cũng có thể giúp robot gặt người. Trên chiến trường, robot có thể ẩn nấp và đánh lửa binh lính đối phương. Trong các chiến dịch tìm kiếm và cứu hộ robot có thể ghi dấu diếm tình hình thực tế để các nạn nhân cảm thấy yên tâm và lạc quan hơn.

Tất nhiên, Arkin thừa nhận những robot biết đánh lửa có thể gây nên nhiều vấn đề về đạo đức. Theo ông, sự lửa gặt được chấp nhận trong hoàn cảnh chiến tranh, song nó bị xả hơi lên án trong những tình huống khác. Vì thế trong tương lai có lẽ các chính phủ cần ban hành quy định về những trường hợp mà người máy có thể thực hiện hành vi đánh lửa.

Minh Long

Tin cùng loại mới hơn

- [Đa người nhặt tạo ra đồi](#)
- ['Tai thần' của máy bay](#)
- [Tàu ngầm chạy bằng pin](#)
- [Phi thuyền biết 'hy sinh'](#)
- [Gạch bê tông siêu nhẹ giúp giảm chi phí](#)

Tin cùng loại cũ hơn

- [Phi cơ do thám bay liên tục 5 năm](#)
- [Làm ra quần áo từ bình xịt](#)

Footer

15.11 Viết code trang quản trị

15.11.1 Tạo trang BackEndMasterPage

Bước 1 : Tạo thư mục Admin, Phải chuột thư mục **admin**, chọn **Add -> Add New Item...**. Chọn **Master Page**, đặt tên **BackEndMasterPage**. Sau đó click **Add**.

Thêm đoạn code sau vào dưới **<body>** trang **BackEndMasterPage.master** để được giao diện như bên dưới:

```
<div id="container">
    <div id="banner"></div>
    <div id="left">
        <div id="menu">
            <a href="Default.aspx">Trang Chủ</a>
            <a href="TheLoai.aspx">Thể Loại</a>
            <a href="LoaiTin.aspx">Loại Tin</a>
            <a href="Tin.aspx">Tin</a>
        </div>
    </div>
    <div id="content">
        <asp:ContentPlaceHolder ID="ContentPlaceHolderNoiDung" runat="server">
        </asp:ContentPlaceHolder>
    </div>
    <div id="footer">Bản quyền thuộc về ©HIENLTH</div>
</div>
```



Bước 2 : Phải chuột thư mục trong thư mục **admin**, chọn **Add -> Add New Item...**. Chọn **Style Sheet**, đặt tên **Style**. Sau đó click **Add**

Bước 3 : Thêm đoạn code sau vào dưới **<head>** trang **BackEndMasterPage**

```
<link href="Style.css" rel="stylesheet" />
```

Bước 4 : Thêm đoạn code sau vào **Style.css**

```
body {font-family:Verdana; font-size:12px}  
#menu a{display:block; text-decoration:none; padding:5px; background-color:Black;  
color:White; border:solid 1px yellow;}
```

15.11.2 Tạo trang Default.aspx

Bước 1 : Phải chuột thư mục **admin**, chọn **Add -> Add New Item...** Chọn **Web Form**, đặt tên **Default**, check ô **Select master page**. Sau đó click **Add**. Chọn **admin -> BackEndMasterPage** và click **OK** ở hộp thoại tiếp theo.

Bước 2 : Thêm đoạn code sau vào **Default.aspx.cs**

```
protected void Page_Load(object sender, EventArgs e)  
{  
    //Chưa đăng nhập hoặc không có quyền admin thì không cho vào trang quản trị  
    if (Session["idUser"] == null || Session["idGroup"].ToString() == "0")  
    {  
        Response.Redirect("../Default.aspx");  
    }  
}
```

15.11.3 Tạo trang TheLoai.aspx

Bước 1 : Phải chuột thư mục **Admin**, chọn **Add -> Add New Item...** Chọn **Web Form**, đặt tên **TheLoai**, check ô **Select master page**. Sau đó click **Add**. Chọn **admin -> BackEndMasterPage** và click **OK** ở hộp thoại tiếp theo.

Bước 2 : Thêm đoạn code sau vào **TheLoai.aspx**

```
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolderNoiDung"  
Runat="Server">  
    <h2 align="center">QUẢN LÝ THỂ LOẠI</h2>  
    <asp:MultiView ID="MultiView1" runat="server">  
        <asp:View ID="View_DanhSach" runat="server">  
            <asp:Button ID="btnThemTheLoai" runat="server"  
            OnClick="btnThemTheLoai_Click" Text="Thêm Thể loại" />  
            <asp:GridView ID="GridView1" runat="server"  
            AutoGenerateSelectButton="True" CellPadding="4" ForeColor="#333333" GridLines="None"  
            onselectedindexchanged="GridView1_SelectedIndexChanged" AutoGenerateColumns="False">  
                <AlternatingRowStyle BackColor="White" />  
                <Columns>  
                    <asp:BoundField DataField="idTL" HeaderText="Mã" />  
                    <asp:BoundField DataField="TenTL" HeaderText="Tên thể loại" />  
                    <asp:BoundField DataField="TenTL_KhongDau" HeaderText="Tên thể  
loại không dấu" />  
                    <asp:BoundField DataField="ThuTu" HeaderText="Thứ tự" />  
                    <asp:BoundField DataField="AnHien" HeaderText="Ẩn hiện" />  
                </Columns>  
                <EditRowStyle BackColor="#7C6F57" />  
                <FooterStyle BackColor="#1C5E55" Font-Bold="True"  
                ForeColor="White" />  
                <HeaderStyle BackColor="#1C5E55" Font-Bold="True"  
                ForeColor="White" />  
                <PagerStyle BackColor="#666666" ForeColor="White"  
                HorizontalAlign="Center" />  
                <RowStyle BackColor="#E3EAEB" />
```

```

        <SelectedRowStyle BackColor="#C5BBAF" Font-Bold="True"
ForeColor="#333333" />
        <SortedAscendingCellStyle BackColor="#F8FAFA" />
        <SortedAscendingHeaderStyle BackColor="#246B61" />
        <SortedDescendingCellStyle BackColor="#D4DFE1" />
        <SortedDescendingHeaderStyle BackColor="#15524A" />
    </asp:GridView>
</asp:View>
<asp:View ID="View_ChiTiet" runat="server">
    <table align="center" cellpadding="5" cellspacing="0">
        <tr>
            <td>Mã thẻ loại</td>
            <td>
                <asp:Label ID="lblMaTheLoai" runat="server" Font-
Bold="True" ForeColor="Red"></asp:Label>
            </td>
        </tr>
        <tr>
            <td>Tên thẻ loại</td>
            <td>
                <asp:TextBox ID="txtTenTheLoai"
runat="server"></asp:TextBox>
            </td>
        </tr>
        <tr>
            <td>Tên thẻ loại không dấu</td>
            <td>
                <asp:TextBox ID="txtTenTLKhongDau"
runat="server"></asp:TextBox>
            </td>
        </tr>
        <tr>
            <td>Thứ tự</td>
            <td>
                <asp:TextBox ID="txtThuTu" runat="server"></asp:TextBox>
            </td>
        </tr>
        <tr>
            <td>&nbsp;</td>
            <td>
                <asp:CheckBox ID="chkHienThi" runat="server" Text="Hiển
thị?" />
            </td>
        </tr>
        <tr>
            <td colspan="2" align="center">
                <asp:Button ID="btnThem" runat="server" Text="Thêm"
onclick="btnThem_Click" />
                <asp:Button ID="btnXoa" runat="server" Text="Xóa"
onclick="btnXoa_Click" onclick="return confirm('Chắc xóa không?');"/>
                <asp:Button ID="btnSua" runat="server" Text="Sửa"
onclick="btnSua_Click" />
                <asp:Button ID="btnDanhSach" runat="server" Text="Danh
sách" onclick="btnDanhSach_Click" />
            </td>
        </tr>
    </table>
</asp:View>
</asp:MultiView>
</asp:Content>

```

Bước 3 : Thêm đoạn code sau vào TheLoai.aspx.cs

```
using System.Data;
using System.Data.SqlClient;
using System.Web.Configuration;
```

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        this.Load_TheLoai();
        MultiViewTheLoai.ActiveViewIndex = 0;
    }
}
```

```
private void hienthiliroi()
{
    GridView1.DataSource = XuLy.Xuly_Select("SELECT * FROM theloai ORDER BY ThuTu");
    GridView1.DataBind();
}
```

```
protected void GridView1_SelectedIndexChanged(object sender, EventArgs e)
{
    //hiển thị viewChiTiet
    MultiView1.ActiveViewIndex = 1;
    //hiển thị thông tin
    string sql = string.Format("SELECT * FROM theloai WHERE idTL={0}",
        GridView1.SelectedRow.Cells[1].Text);
    DataRow r = XuLy.Xuly_Select(sql).Rows[0];
    lblMaTheLoai.Text = r["idTL"].ToString();
    txtTenTheLoai.Text = r["TenTL"].ToString();
    txtTenTLKhongDau.Text = r["TenTL_KhongDau"].ToString();
    txtThuTu.Text = r["ThuTu"].ToString();
    chkHienThi.Checked = (r["AnHien"].ToString() == "1");
    btnSua.Enabled = btnXoa.Enabled = true;
    btnThem.Enabled = false;
}
```

```
protected void btnDanhSach_Click(object sender, EventArgs e)
{
    MultiView1.ActiveViewIndex = 0;
}
```

```
protected void btnXoa_Click(object sender, EventArgs e)
{
    string idTL = GridView1.SelectedRow.Cells[1].Text;
    string sql = string.Format("DELETE FROM theloai WHERE idTL={0}", idTL); if
(XuLy.XulyTruyvan(sql))//true
    {
        hienthiliroi();
        MultiView1.ActiveViewIndex = 0;
    }
    else
    {
        var thongbao = new System.Text.StringBuilder();
        thongbao.Append("<script language='javascript'>");
    }
}
```

```

        thongbao.Append("alert('Xóa thẻ loại (idTL = " + idTL + " ) không thành
công!');");
        thongbao.Append("</script>");

        if (!ClientScript.IsClientScriptBlockRegistered(this.GetType(),
"ThongBao"))
        {
            ClientScript.RegisterClientScriptBlock(this.GetType(), "ThongBao",
thongbao.ToString());
        }
    }
}

```

```

protected void btnThem_Click(object sender, EventArgs e)
{
    string sqlInsert = string.Format("INSERT INTO TheLoai(TenTL, TenTL_KhongDau,
ThuTu, AnHien) VALUES (N'{0}', '{1}', {2}, {3})", txtTenTheLoai.Text,
txtTenTLKhongDau.Text, txtThuTu.Text, chkHienThi.Checked ? 1 : 0);
    if (XuLy.XulyTruyvan(sqlInsert))//ok
        hienthiluoi();
}

```

```

protected void btnSua_Click(object sender, EventArgs e)
{
    string sqlUpdate = string.Format("UPDATE TheLoai SET TenTL = N'{0}',
TenTL_KhongDau = '{1}', ThuTu = {2}, AnHien = {3} WHERE idTL={4}",
txtTenTheLoai.Text, txtTenTLKhongDau.Text, txtThuTu.Text, chkHienThi.Checked ? 1 :
0, lblMaTheLoai.Text);
    if (XuLy.XulyTruyvan(sqlUpdate))//ok
        hienthiluoi();
}

```

```

protected void btnThemTheLoai_Click(object sender, EventArgs e)
{
    txtTenTheLoai.Text = txtTenTLKhongDau.Text = txtThuTu.Text = "";
    lblMaTheLoai.Text = "";
    btnThem.Enabled = btnSua.Enabled = btnXoa.Enabled = false;
    MultiView1.ActiveViewIndex = 1;
}

```

Bước 4 : Test thử

Màn hình lúc chạy:



Màn hình thao tác thêm mới:

QUẢN LÝ THẺ LOẠI

Mã thẻ loại	<input type="text"/>
Tên thẻ loại	<input type="text"/>
Tên thẻ loại không dấu	<input type="text"/>
Thứ tự	<input type="text"/>
<input type="checkbox"/> Hiển thị?	
<input type="button" value="Thêm"/> <input type="button" value="Xóa"/> <input type="button" value="Sửa"/> <input type="button" value="Danh sách"/>	

Màn hình thao tác sửa chữa:

QUẢN LÝ THẺ LOẠI

Mã thẻ loại	1
Tên thẻ loại	<input type="text" value="Xã hội"/>
Tên thẻ loại không dấu	<input type="text" value="Xa-Hoi"/>
Thứ tự	<input type="text" value="1"/>
<input checked="" type="checkbox"/> Hiển thị?	
<input type="button" value="Thêm"/> <input type="button" value="Xóa"/> <input type="button" value="Sửa"/> <input type="button" value="Danh sách"/>	

QUẢN LÝ THẺ LOẠI

Mã thẻ loại	4
Tên thẻ loại	<input type="text" value="VĂN hoc"/>
Tên thẻ loại không dấu	<input type="text" value="Van-Hoa"/>
Thứ tự	<input type="text" value="4"/>
<input checked="" type="checkbox"/> Hiển thị?	
<input type="button" value="Thêm"/> <input type="button" value="Xóa"/> <input type="button" value="Sửa"/> <input type="button" value="Danh sách"/>	

Trang trên localhost:49165 cho biết:

 Chắc xóa không?

Màn hình thao tác xóa.

15.11.4 Tạo trang LoaiTin.aspx

Bước 1 : Phải chuột thư mục Admin, chọn Add ->Add New Item... Chọn Web Form, đặt tên LoaiTin, check ô Select master page.Sau đó click Add. Chọn admin -> BackEndMasterPage và click OK ở hộp thoại tiếp theo

Bước 2 : Thêm đoạn code sau vào LoaiTin.aspx

```
<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
    <style type="text/css">
        .style1
        {
            height: 22px;
        }
    </style>
</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolderNoiDungQuanTri"
Runat="Server">
    <asp:Button ID="btnThemMoi" runat="server" Text="Thêm Loại Tin"
        onclick="btnThemMoi_Click" style="height: 26px" />
    <br />
    <br />
    <asp:MultiView ID="MultiViewLoaiTin" runat="server" ActiveViewIndex="0">
        <br />
        <asp:View ID="View1" runat="server">
            <asp:GridView ID="GridViewLoaiTin" runat="server" AutoGenerateColumns="False"
                onselectedindexchanged="GridViewLoaiTin_SelectedIndexChanged">
                <Columns>
                    <asp:BoundField HeaderText="idLT" DataField="idLT" />
                    <asp:BoundField HeaderText="Ten" DataField="Ten" />
                    <asp:BoundField HeaderText="Ten_KhongDau" DataField="Ten_KhongDau" />
                    <asp:BoundField HeaderText="ThuTu" DataField="ThuTu" />
                    <asp:BoundField HeaderText="AnHien" DataField="AnHien" />
                    <asp:BoundField HeaderText="idTL" DataField="TenTL" />
                    <asp:CommandField ShowSelectButton="True" />
                </Columns>
            </asp:GridView>
        </asp:View>
        <br />
        <asp:View ID="View2" runat="server">
            <table style="width:100%;">
                <tr>
                    <td> idLT</td>
                    <td><asp:TextBox ID="txtidLT" runat="server"
Enabled="False"></asp:TextBox></td>
                </tr>
                <tr>
                    <td>Tên</td>
                    <td> <asp:TextBox ID="txtTen" runat="server"></asp:TextBox></td>
                </tr>
                <tr>
                    <td class="style1">Tên Không Dấu</td>
                    <td class="style1"><asp:TextBox ID="txtTenKD"
runat="server"></asp:TextBox></td>
                </tr>
                <tr>
                    <td class="style1">Thứ Tự</td>
                </tr>
            </table>
        </asp:View>
    </MultiView>
</asp:Content>
```

```

<td class="style1"><asp:TextBox ID="txtThuTu"
runat="server"></asp:TextBox></td>
</tr>
<tr>
    <td class="style1">Ẩn Hiện</td>
    <td class="style1">
        <asp:RadioButtonList ID="rblAnHien" runat="server">
            <asp:ListItem Value="1">Hiện</asp:ListItem>
            <asp:ListItem Value="0">Ẩn</asp:ListItem>
        </asp:RadioButtonList>
    </td>
</tr>
<tr>
    <td class="style1">idTL</td>
    <td class="style1"><asp:DropDownList ID="ddlidTL"
runat="server"></asp:DropDownList></td>
</tr>
<tr>
    <td class="style1">&ampnbsp</td>
    <td class="style1">
        <asp:Button ID="btnThem" runat="server" Text="Thêm" onclick="btnThem_Click" />
        <asp:Button ID="btnSua" runat="server" Text="Sửa" onclick="btnSua_Click" />
        <asp:Button ID="btnXoa" runat="server" Text="Xóa" onclick="btnXoa_Click" />
        <asp:Button ID="btnQuayLai" runat="server" Text="Quay Lại"
onclick="btnQuayLai_Click" />
    </td>
</tr>
</table>
</asp:View>
</asp:MultiView>
</asp:Content>

```

Bước 3 : Thêm đoạn code sau vào LoaiTin.aspx.cs

```

using System.Data;
using System.Data.SqlClient;
using System.Web.Configuration;

```

```

protected void Page_Load(object sender, EventArgs e)
{
    MultiViewLoaiTin.ActiveViewIndex = 0;
    if (!IsPostBack)
    {
        this.LoadLoaiTin();
        ddlidTL.DataSource = XuLy.Xuly_Select("SELECT idTL,TenTL FROM theloai");
        ddlidTL.DataValueField = "idTL";
        ddlidTL.DataTextField = "TenTL";
        ddlidTL.DataBind();
    }
}

```

```

public void LoadLoaiTin()
{
    string qrLT = string.Format(@" SELECT loaitin.*,TenTL FROM loaitin,theloai
                                WHERE loaitin.idTL = theloai.idTL ORDER BY idLT DESC");
    DataTable dt = XuLy.Xuly_Select(qrLT);
}

```

```
GridViewLoaiTin.DataSource = dt;
GridViewLoaiTin.DataBind();
}
```

```
protected void btnThemMoi_Click(object sender, EventArgs e)
{
    MultiViewLoaiTin.ActiveViewIndex = 1;
    btnSua.Enabled = false;
    btnXoa.Enabled = false;
    btnThem.Enabled = true;
    btnThemMoi.Enabled = false;
}
```

```
protected void btnThem_Click(object sender, EventArgs e)
{
    string Ten = txtTen.Text;
    string TenKD = txtTenKD.Text;
    string ThuTu = txtThuTu.Text;
    string AnHien = rblAnHien.Text;
    string idTL = ddlidTL.SelectedValue;
    string qrThem = string.Format(@"INSERT INTO
loaitin(Ten,Ten_KhongDau,ThuTu,AnHien,idTL) VALUES(N'{0}', '{1}', '{2}', '{3}', '{4}' )",
Ten, TenKD, ThuTu, AnHien, idTL);

    XuLy.XulyTruyvan(qrThem);
    btnThemMoi.Enabled = true;
    MultiViewLoaiTin.ActiveViewIndex = 0;
    this.LoadLoaiTin();
}
protected void GridViewLoaiTin_SelectedIndexChanged(object sender, EventArgs e)
{
    MultiViewLoaiTin.ActiveViewIndex = 1;
    btnSua.Enabled = true;
    btnXoa.Enabled = true;
    btnThem.Enabled = false;
    btnThemMoi.Enabled = false;
    string idLT = GridViewLoaiTin.SelectedRow.Cells[0].Text.ToString();
    string qrLT = string.Format("SELECT * FROM loaitin WHERE idLT = '{0}'", idLT);
    DataTable dt = XuLy.Xuly_Select(qrLT);
    txtidLT.Text = dt.Rows[0]["idLT"].ToString();
    txtTen.Text = dt.Rows[0]["Ten"].ToString();
    txtTenKD.Text = dt.Rows[0]["Ten_KhongDau"].ToString();
    txtThuTu.Text = dt.Rows[0]["ThuTu"].ToString();
    rblAnHien.Text = dt.Rows[0]["AnHien"].ToString();
    ddlidTL.DataValueField = dt.Rows[0]["idTL"].ToString();
}
```

```
protected void btnSua_Click(object sender, EventArgs e)
{
    string idLT = txtidLT.Text;
    string Ten = txtTen.Text;
    string TenKD = txtTenKD.Text;
    string ThuTu = txtThuTu.Text;
    string AnHien = rblAnHien.Text;
    string idTL = ddlidTL.SelectedValue;
    string qrSua = string.Format(@" UPDATE loaitin
```

```

        SET Ten = N'{0}', Ten_KhongDau = '{1}', ThuTu = '{2}', AnHien =
        '{3}', idTL = '{4}' WHERE idLT = '{5}'", Ten, TenKD, ThuTu, AnHien, idTL, idLT);
        XuLy.XulyTruyvan(qrSua);
        btnThemMoi.Enabled = true;
        MultiViewLoaiTin.ActiveViewIndex = 0;
        this.LoadLoaiTin();
    }
}

```

```

protected void btnXoa_Click(object sender, EventArgs e)
{
    string idLT = txtidLT.Text;
    string qrXoa = string.Format("DELETE FROM loaitin WHERE idLT = '{0}' ", idLT);
    string qrDem = string.Format("SELECT idTin FROM tin WHERE idLT = '{0}' ", idLT);
    DataTable dt = XuLy.Xuly_Select(qrDem);
    int count = dt.Rows.Count;
    if (count == 0) {
        XuLy.XulyTruyvan(qrXoa);
        btnThemMoi.Enabled = true;
        MultiViewLoaiTin.ActiveViewIndex = 0;
        this.LoadLoaiTin();
    }
    else
    {
        Response.Write("Không xóa được");
    }
}
protected void btnQuayLai_Click(object sender, EventArgs e)
{
    btnThemMoi.Enabled = true;
    this.LoadLoaiTin();
    MultiViewLoaiTin.ActiveViewIndex = 0;
}
}

```

Bước 4 : Kiểm tra

Hiện danh sách loại tin

QUẢN LÝ LOẠI TIN					
Thêm loại tin					
Mã loại	Tên loại	Tên không dấu	Thứ tự	Thể loại	
Chi tiết 4	Du Học	Du-Hoc	4	Xã hội	
Chi tiết 13	Mua Sắm	Mua-Sam	5	Vi Tính	
Chi tiết 19	Điện Ánh	Dien-Anh	5	VĂn học	
Chi tiết 32	Cửa Sổ Blog	Cua-So-Blog	5	Đời sống	
Chi tiết 20	Mỹ Thuật	My-Thuat	6	VĂn học	
Chi tiết 14	Doanh Nghiệp Việt	Doanh-Nghiep-Viet	6	Vi Tính	
			1 2 3 4 5 6		

Thêm loại tin		Sửa loại tin	
QUẢN LÝ LOẠI TIN			
Mã loại	<input type="text"/>	Mã loại	<input type="text" value="19"/>
Tên loại	<input type="text"/>	Tên loại	<input type="text" value="Điện Ánh"/>
Tên không dấu	<input type="text"/>	Tên không dấu	<input type="text" value="Dien-Anh"/>
Thể loại	<input style="width: 100px;" type="text" value="Xã hội"/>	Thể loại	<input style="width: 100px;" type="text" value="VĂn học"/>
Thứ tự	<input type="text"/>	Thứ tự	<input type="text" value="5"/>
Ẩn/Hiện	<input checked="" type="radio"/> Ẩn <input type="radio"/> Hiện	Ẩn/Hiện	<input checked="" type="radio"/> Ẩn <input type="radio"/> Hiện
<input type="button" value="Thêm"/> <input type="button" value="Xóa"/> <input type="button" value="Sửa"/> <input type="button" value="Danh sách"/>		<input type="button" value="Thêm"/> <input type="button" value="Xóa"/> <input type="button" value="Sửa"/> <input type="button" value="Danh sách"/>	

15.11.5 Tạo trang Tin.aspx

Bước 1 : Phải chuột thư mục **Admin**, chọn **Add ->Add New Item...** Chọn **Web Form**, đặt tên **Tin**, check ô **Select master page**. Sau đó click **Add**. Chọn **admin -> BackEndMasterPage** và click **OK** ở hộp thoại tiếp theo

Bước 2 : Thêm đoạn code sau vào **Tin.aspx**

```
<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
<style>
  .tieude{ color:Blue}
  .TenTL,.Ten{ color:red}
  .style1{ height: 12px;}
</style>
</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolderNoiDungQuanTri"
Runat="Server">
<asp:Button ID="btnThemMoi" runat="server" Text="Thêm Tin"
  onclick="btnThemMoi_Click" style="height: 26px" />
<br />
<asp:MultiView ID="MultiViewTin" runat="server" ActiveViewIndex="0">
  <asp:View ID="View1" runat="server">
    <asp:GridView ID="GridViewTin" runat="server" AutoGenerateColumns="False"
      AllowPaging="True" onpageindexchanging="GridViewTin_PageIndexChanged"
      onselectedindexchanged="GridViewTin_SelectedIndexChanged">
      <Columns>
        <asp:BoundField DataField="idTin" HeaderText="idTin" />
        <asp:TemplateField>
          <ItemTemplate>
            <table style="width:100%;">
              <tr>
                <td colspan="3">
                  <asp:Label CssClass="tieude" ID="lblTieuDe" runat="server"
Text='<%#Eval("TieuDe") %>'></asp:Label>
```

```

        <br />
        <asp:Image ID="imgHinh" runat="server" ImageAlign="Left"
            Width="80px" Height="70px" ImageUrl='<%#
Eval("urlHinh", "../upload/tintuc/{0}") %>' />
        &nbsp;<asp:Label ID="lblTomTat" runat="server"
Text='<%#Eval("TomTat") %>'></asp:Label>
        </td>
    </tr>
    <tr>
        <td>
            Thể Loại :<asp:Label ID="lbl_idTL" CssClass="TenTL" runat="server"
Text='<%# Eval("TenTL") %>'></asp:Label>
            <br />
            Loại Tin :<asp:Label ID="lbl_idLT" CssClass="Ten" runat="server"
Text='<%# Eval("Ten") %>'></asp:Label>
            </td>
            <td>
                Tin Nổi Bật :
                <asp:Label ID="lblTinNoiBat" runat="server" Text='<%#
Eval("TinNoiBat") %>'></asp:Label>
                <br />
            </td>
            <td>Ẩn/Hiện :
                <asp:Label ID="lblAnHien" runat="server" Text='<%# Eval("AnHien") %>'></asp:Label>
            </td>
        </tr>
    </table>
</ItemTemplate>
</asp:TemplateField>
<asp:CommandField ShowSelectButton="True" />
</Columns>
</asp:GridView>
</asp:View>
<br />
<asp:View ID="View2" runat="server">
    <table style="width:100%;">
        <tr>
            <td>idTin</td>
            <td><asp:Label ID="lblidTin" runat="server" Text="Label"></asp:Label> </td>
        </tr>
        <tr>
            <td class="style1">TieuDe</td>
            <td class="style1"><asp:TextBox ID="txtTieuDe" runat="server"></asp:TextBox></td>
        </tr>
        <tr>
            <td>TieuDe_KhongDau</td>
            <td><asp:TextBox ID="txtTieuDeKD" runat="server"></asp:TextBox></td>
        </tr>
        <tr>
            <td>TomTat</td>
            <td><asp:TextBox ID="txtTomTat" runat="server" Height="100px"
TextMode="MultiLine" Width="100%"></asp:TextBox></td>
        </tr>
        <tr>
            <td>urlHinh</td>

```

```

        <td><asp:FileUpload ID="FileUploadHinh" runat="server" /></td>
    </tr>
    <tr>
        <td> Content</td>
        <td> <CKEditor:CKEditorControl ID="CKE_Content"
runat="server"></CKEditor:CKEditorControl></td>
    </tr>
    <tr>
        <td> idTL</td>
        <td><asp:DropDownList ID="ddl_idTL" runat="server"></asp:DropDownList> </td>
    </tr>
    <tr>
        <td> idLT</td>
        <td id="idLT"> <asp:DropDownList ID="ddl_idLT"
runat="server"></asp:DropDownList> </td>
    </tr>
    <tr>
        <td> TinNoiBat</td>
        <td>
            <asp:RadioButtonList ID="rbl_TinNoiBat" runat="server">
                <asp:ListItem Value="0">Bình Thường</asp:ListItem>
                <asp:ListItem Value="1">Nổi Bật</asp:ListItem>
            </asp:RadioButtonList>
        </td>
    </tr>
    <tr>
        <td> AnHien</td>
        <td>
            <asp:RadioButtonList ID="rbl_AnHien" runat="server">
                <asp:ListItem Value="0">Ẩn</asp:ListItem>
                <asp:ListItem Value="1">Hiện</asp:ListItem>
            </asp:RadioButtonList>
        </td>
    </tr>
    <tr>
        <td>&nbsp;</td>
        <td>
            <asp:Button ID="btnThem" runat="server" onclick="btnThem_Click" Text="Thêm" />
            <asp:Button ID="btnSua" runat="server" Text="Sửa" OnClick="btnSua_Click" />
            <asp:Button ID="btnXoa" runat="server" onclick="btnXoa_Click" Text="Xóa" />
            <asp:Button ID="btnQuayLai" runat="server" onclick="btnQuayLai_Click"
Text="Quay Lại" />
        </td>
    </tr>
    </table>
</asp:View>
</asp:MultiView>
</asp:Content>

```

Bước 3 : Thêm đoạn code sau vào Tin.aspx.cs

```

using System.Data;
using System.Data.SqlClient;
using System.Web.Configuration;
using System.Collections;
using System.IO;

protected void Page_Load(object sender, EventArgs e)
{

```

```
this.LoadTin();
    ddl_idTL.DataSource = XuLy.Xuly_Select("SELECT idTL,TenTL FROM theloai");
    ddl_idTL.DataValueField = "idTL";
    ddl_idTL.DataTextField = "TenTL";
    ddl_idTL.DataBind();
    this.LoadLoaiTinTheoTheLoai();
}
```

```
public void LoadLoaiTinTheoTheLoai()
{
    string idTL = ddl_idTL.SelectedValue;
    string qrLT = string.Format("SELECT idLT,Ten FROM loaitin WHERE idTL = {0}",
    idTL);
    ddl_idLT.DataSource = XuLy.Xuly_Select(qrLT);
    ddl_idLT.DataTextField = "Ten";
    ddl_idLT.DataValueField = "idLT";
    ddl_idLT.DataBind();
}
```

```
public void LoadTin()
{
    string qrTin = string.Format(@" SELECT tin.*,TenTL,Ten FROM tin,theloai,loaitin
        WHERE theloai.idTL = tin.idTL AND loaitin.idLT = tin.idLT
        ORDER BY idTin DESC ");
    DataTable dt = XuLy.Xuly_Select(qrTin);
    GridViewTin.DataSource = dt;
    GridViewTin.DataBind();
}
```

```
protected void GridViewTin_PageIndexChanging(object sender, GridViewEventArgs e)
{
    GridViewTin.PageIndex = e.NewPageIndex;
    this.LoadTin();
}
```

```
protected void GridViewTin_SelectedIndexChanged(object sender, EventArgs e)
{
    MultiViewTin.ActiveViewIndex = 1;
    btnSua.Enabled = true;
    btnXoa.Enabled = true;
    btnThem.Enabled = false;
    string idTin = GridViewTin.SelectedRow.Cells[0].Text.ToString();
    string sqlSelect = string.Format(@"SELECT * FROM tin WHERE idTin='{0}'", idTin);
    DataTable dtSelect = new DataTable();
    dtSelect = XuLy.Xuly_Select(sqlSelect);
    DataRow rowSelect = dtSelect.Rows[0];
    lblIdTin.Text = rowSelect["idTin"].ToString();
    txtTieuDe.Text = rowSelect["TieuDe"].ToString();
    txtTieuDeKD.Text = rowSelect["TieuDe_KhongDau"].ToString();
    txtTomTat.Text = rowSelect["TomTat"].ToString();
    CKE_Content.Text = rowSelect["Content"].ToString();
    rbl_AnHien.Text = rowSelect["AnHien"].ToString();
    rbl_TinNoiBat.Text = rowSelect["TinNoiBat"].ToString();
```

```
        ddl_idTL.DataValueField = rowSelect["idTL"].ToString();
        ddl_idLT.DataValueField = rowSelect["idLT"].ToString();
    }
```

```
protected void btnThemMoi_Click(object sender, EventArgs e)
{
    MultiViewTin.ActiveViewIndex = 1;
    btnSua.Enabled = false;
    btnXoa.Enabled = false;
    btnThemMoi.Enabled = false;
}
```

```
protected void btnXoa_Click(object sender, EventArgs e)
{
    string idTin = lblidTin.Text;
    string sqlDelete = string.Format(@"DELETE FROM tin WHERE idTin='{0}'", idTin);
    XuLy.XulyTruyvan(sqlDelete);
    this.LoadTin();
    MultiViewTin.ActiveViewIndex = 0;
}
```

```
protected void btnThem_Click(object sender, EventArgs e)
{
    string TieuDe = txtTieuDe.Text;
    string TieuDeKD = txtTieuDeKD.Text;
    string TomTat = txtTomTat.Text;
    string urlHinh = "";
    string path = Server.MapPath("~/upload/tintuc/");
    ArrayList mang = new ArrayList { "image/png", "image/jpeg", "image/jpg" };
    if (FileUploadHinh.HasFile)
    {
        string fileType = FileUploadHinh.PostedFile.ContentType;
        if (mang.Contains(fileType))
        {
            string random = XuLy.RandomNumber();
            urlHinh = random + FileUploadHinh.PostedFile.FileName;
            FileUploadHinh.PostedFile.SaveAs(path + urlHinh);
        }
    }
    string Ngay = DateTime.Now.ToString("yyyy-MM-dd");
    stringidUser = Session["idUser"].ToString();
    string Content = CKE_Content.Text;
    string idLT = ddl_idLT.SelectedValue;
    string idTL = ddl_idTL.SelectedValue;
    int SoLanXem = 0;
    string TinNoiBat = rbl_TinNoiBat.Text;
    string AnHien = rbl_AnHien.Text;
    string qrThem = string.Format(@"
        INSERT INTO
        tin(TieuDe,TieuDe_KhongDau, TomTat, urlHinh, Ngay, idUser, Content, idLT, idTL, SoLanXem, Tin
        NoiBat, AnHien)
        VALUES( N'{0}', 
        '{1}', '{2}', '{3}', '{4}', '{5}', '{6}', '{7}', '{8}', '{9}', '{10}', '{11}' )",
        '
```

```
TieuDe, TieuDeKD, TomTat, urlHinh, Ngay, idUser, Content, idLT, idTL,  
SoLanXem, TinNoiBat, AnHien);  
    XuLy.XulyTruyvan(qrThem);  
    btnThemMoi.Enabled = true;  
    MultiViewTin.ActiveViewIndex = 0;  
    this.LoadTin();  
}
```

```
protected void btnQuayLai_Click(object sender, EventArgs e)  
{  
    btnThemMoi.Enabled = true;  
    MultiViewTin.ActiveViewIndex = 0;  
    this.LoadTin();  
}
```

```
protected void btnSua_Click(object sender, EventArgs e)  
{  
    string idTin = lblidTin.Text;  
    string TieuDe = txtTieuDe.Text;  
    string TieuDeKD = txtTieuDeKD.Text;  
    string TomTat = txtTomTat.Text;  
    string urlHinh = "";  
    string path = Server.MapPath("~/upload/tintuc/");  
    ArrayList mang = new ArrayList { "image/png", "image/jpeg", "image/jpg" };  
    if (FileUploadHinh.HasFile) {  
        string fileType = FileUploadHinh.PostedFile.ContentType;  
        if (mang.Contains(fileType))  
        {  
            string random = XuLy.RandomNumber();  
            urlHinh = random + FileUploadHinh.PostedFile.FileName;  
            FileUploadHinh.PostedFile.SaveAs(path + urlHinh);  
        }  
    }  
    string Ngay = DateTime.Now.ToString("yyyy-MM-dd");  
    stringidUser = Session["idUser"].ToString();  
    string Content = CKE_Content.Text;  
    string idLT = ddl_idLT.SelectedValue;  
    string idTL = ddl_idTL.SelectedValue;  
    string TinNoiBat = rbl_TinNoiBat.Text;  
    string AnHien = rbl_AnHien.Text;  
    string qrSua = string.Format(@" UPDATE tin  
        SET TieuDe = N'{0}', TieuDe_KhongDau = '{1}', TomTat = '{2}', urlHinh = '{3}',  
        Ngay = '{4}', idUser = '{5}', Content = '{6}', idLT = '{7}',  
        idTL = '{8}', SoLanXem = '{9}', TinNoiBat = '{10}', AnHien = '{11}' WHERE idTin  
        = '{12}' ",  
        TieuDe, TieuDeKD, TomTat, urlHinh, Ngay, idUser, Content, idLT, idTL, 0, TinNoiBat, AnHien, idTin  
    );  
    XuLy.XulyTruyvan(qrSua);  
    btnThemMoi.Enabled = true;  
    MultiViewTin.ActiveViewIndex = 0;  
    this.LoadTin();  
}
```

Bước 4 : Kiểm tra

Hiện danh sách tin

Trang chủ
Thể loại
Loại tin
Quản lý Tin
Đảng xuất

Quản lý Tin tức

Chi tiết	Hình	Mã tin	Tiêu đề	Thể loại	Loại tin
Chi tiết		997	'Không để thí sinh đăng ký đại học như chơi xổ số'	Xã hội	Giáo Dục
Chi tiết		996	Học sinh vùng cao nghỉ Tết kéo dài vì giá rét	Xã hội	Giáo Dục
Chi tiết		995	Giám đốc tuổi mèo và thành tích đáng nể	Xã hội	Nhịp Điện Trẻ
Chi tiết		994	Mong ước đầu năm của giới trẻ	Xã hội	Nhịp Điện Trẻ

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) ...

Thêm tin

Trang chủ
Thể loại
Loại tin
Quản lý Tin
Đảng xuất

Quản lý Tin tức

Mã tin	<input type="text"/>
Tên tin	<input type="text"/>
Tên không dấu	<input type="text"/>
Thể loại	<input type="text"/> Xã hội
Loại tin	<input type="text"/> Giáo Dục
Ẩn/Hiện	<input checked="" type="radio"/> Có <input type="radio"/> Không
Ngày đăng	<input type="text"/>
Tóm tắt	<input type="text"/>
Hình	<input type="button" value="Choose File"/> No file chosen
 <div style="border: 1px solid #ccc; padding: 5px; height: 150px; margin-top: 10px;"></div>	
Nội dung	
Số lượt xem	
Nổi bật <input checked="" type="radio"/> Có <input type="radio"/> Không	
<input type="button"/> Thêm <input type="button"/> Sửa <input type="button"/> Xóa <input type="button"/> Danh sách	

Trong đó phần Nội dung tin sử dụng control CKEditor để định dạng tin tức.

Lưu ý: CKEditor bản miễn phí không hỗ trợ upload file.

Sửa tin

Trang chủ
Thể loại
Loại tin
Quản lý Tin
Đăng xuất

Quản lý Tin tức

Mã tin

999

Tên tin

Hỗ trợ gần 3.000 vé xe tết cho sinh viên

Tên không dấu

Ho-Tro-Gan-3.000-Ve-Xe-Tet

Thể loại

Loại tin

Án/Hiện

Có Không

Ngày đăng

11/20/2009

Tóm tắt

Mỗi khi máy bay sắp hạ cánh

Hình



No file chosen

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Orientation Page Fit Page Margin Page Header Page Footer Page Number Page Margin Page Header Page Footer

Styles Heading 1 Font Size Color Background Color Text Color Align Vertical Align Width Height Border Padding Margin

Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File Font Size Bold Italic Underline Strikethrough Superscript Subscript List Unlist Horizontal Line Page Break Page Number Page Margin Page Header Page Footer

Source Text Image Table Link File

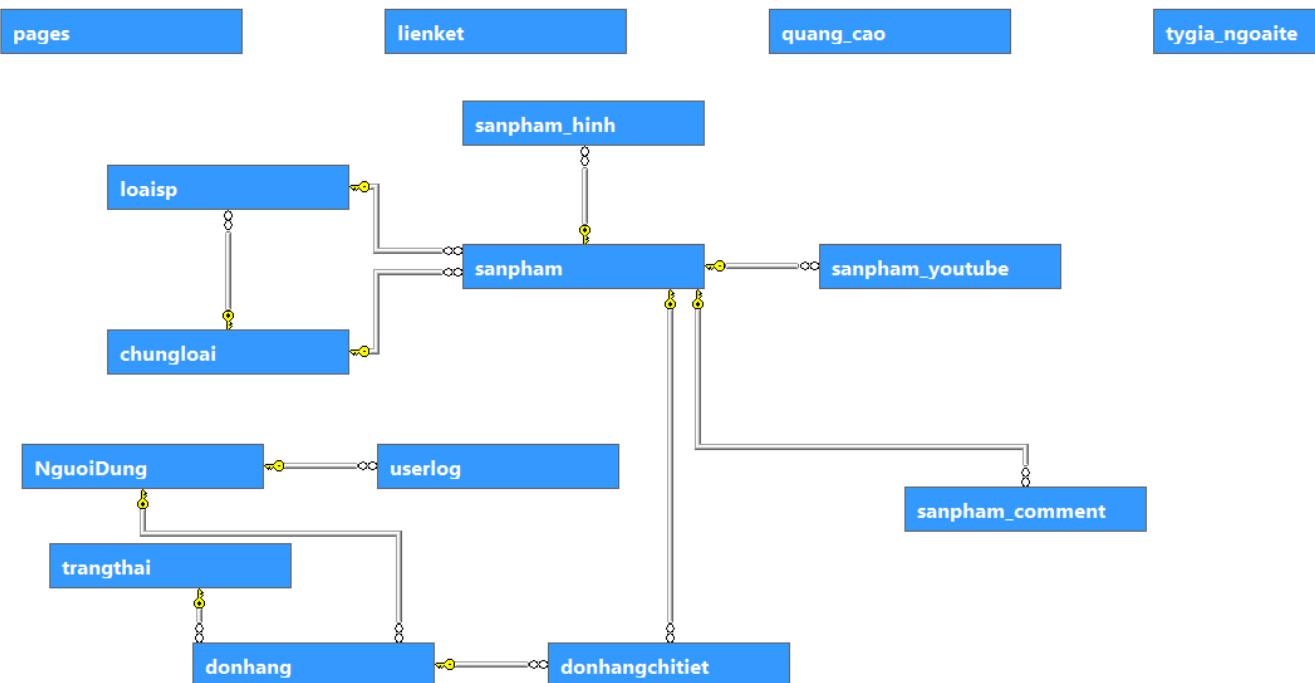
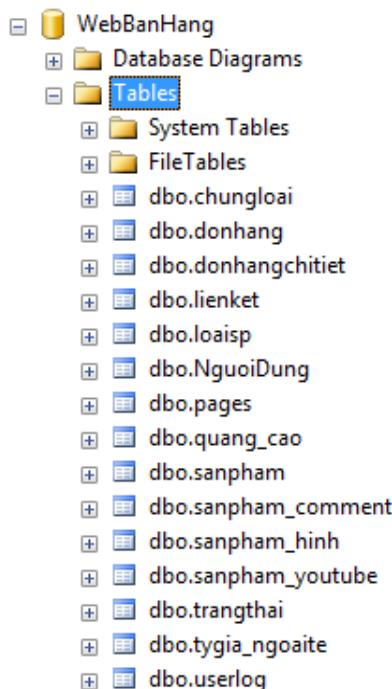
Chương 16: WEBSITE BÁN HÀNG

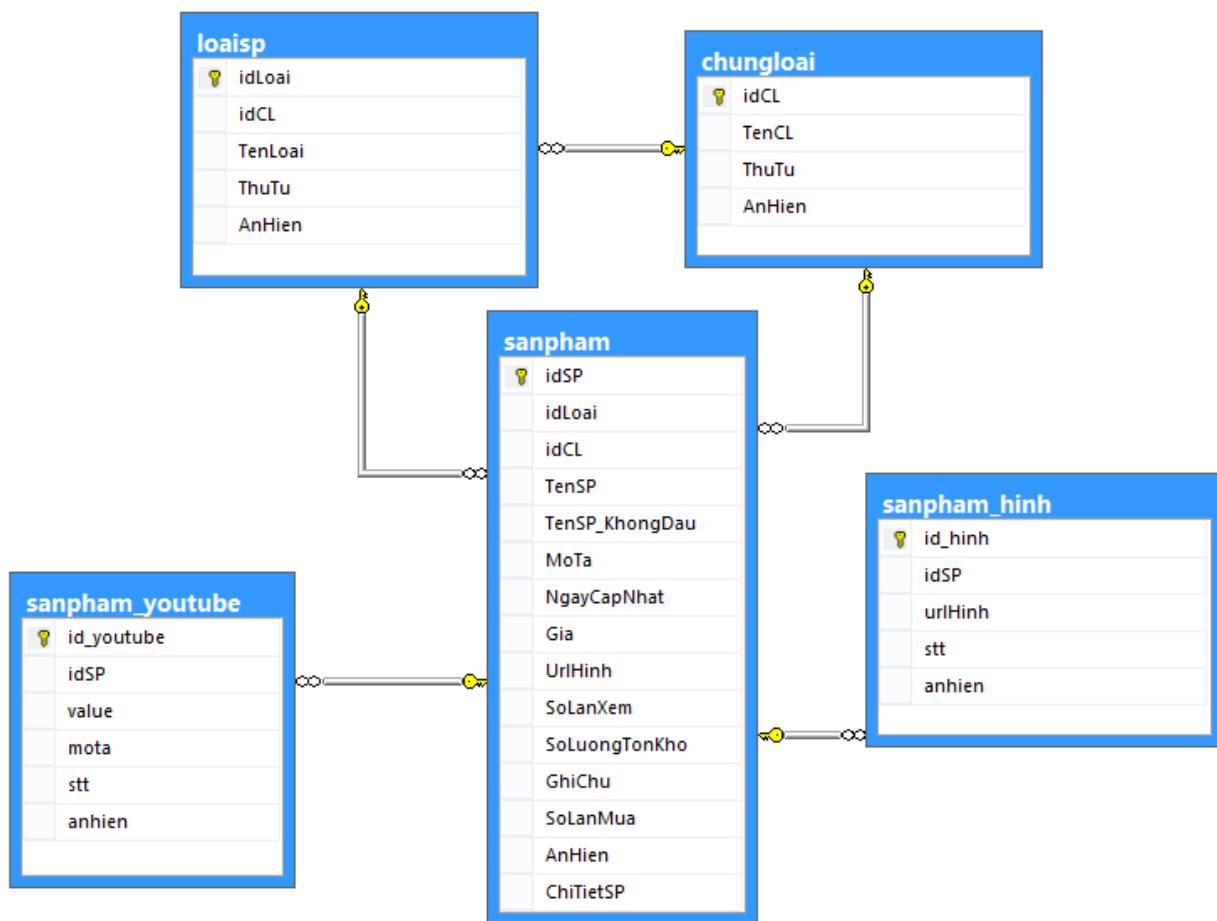
16.1 Tài nguyên & CSDL

16.1.1 Tài nguyên

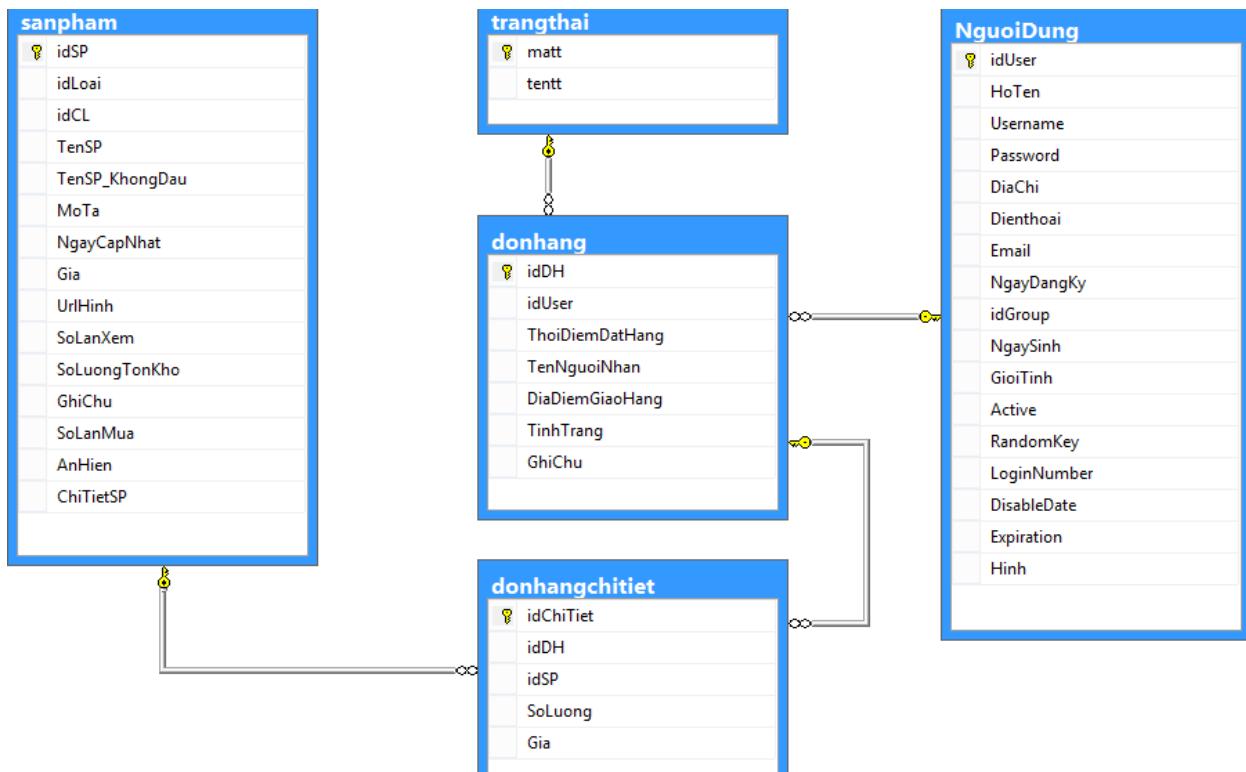
- Toàn bộ database, resource cho chương này có thể download tại đây: <https://www.mediafire.com/folder/6z2c7xicefx9e/WebBanHang>.
- Web demo: <http://nhatngheshop.somee.com>.

16.1.2 Lược đồ CSDL





Thông tin một sản phẩm

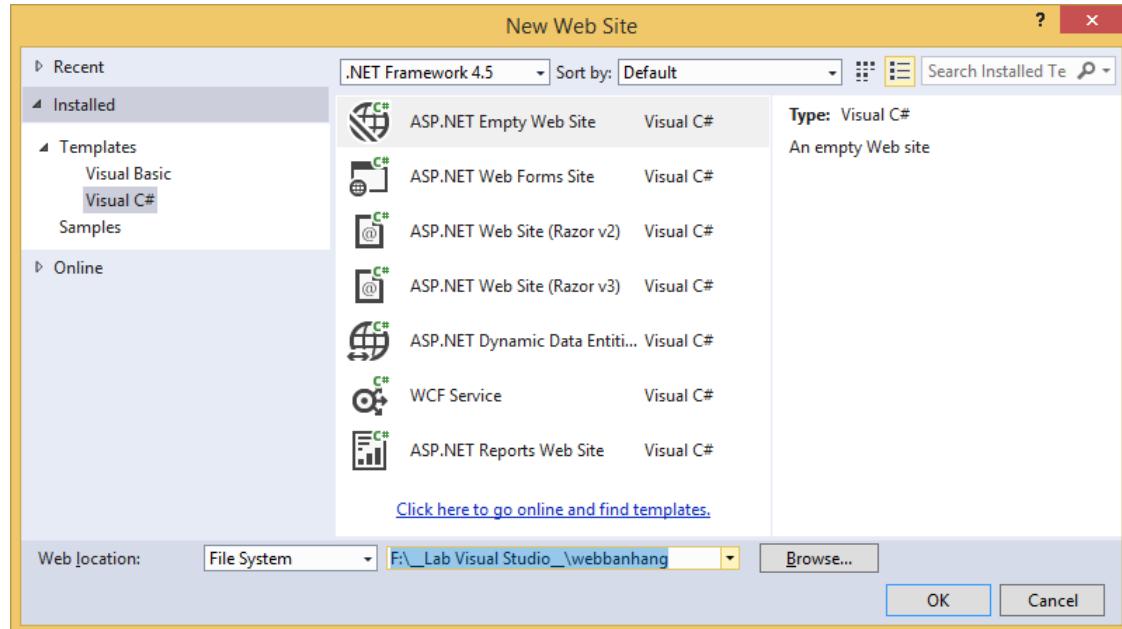


Thông tin đơn đặt hàng

16.1.3 Tạo website bán hàng

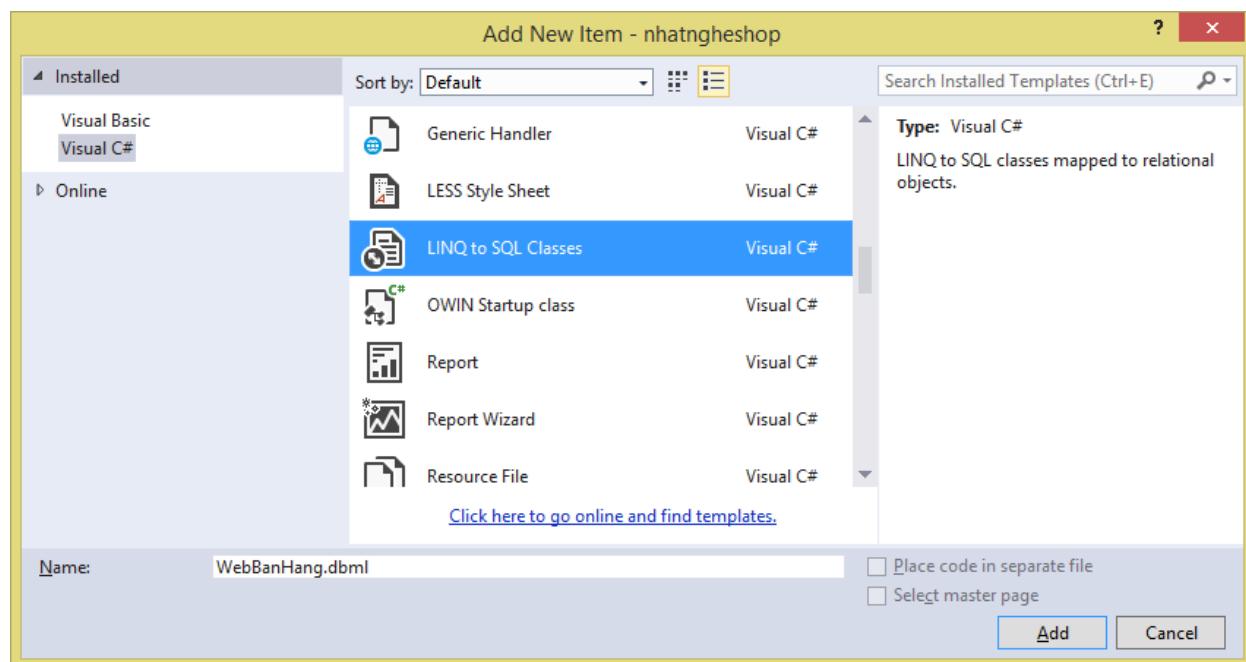
Bước 1 : Mở Visual Studio, chọn menu **File->New->Web Site.**

Bước 2 : Chọn **Visual C#->ASP.NET Empty Web Site.** Đặt tên website là **webtintuc** và chọn nơi lưu webtintuc. Sau đó click **OK** để tạo



16.1.4 Tạo kết nối database

Tạo mới file LINQ to SQL Classes mang tên **WebBanHang.dbml**:

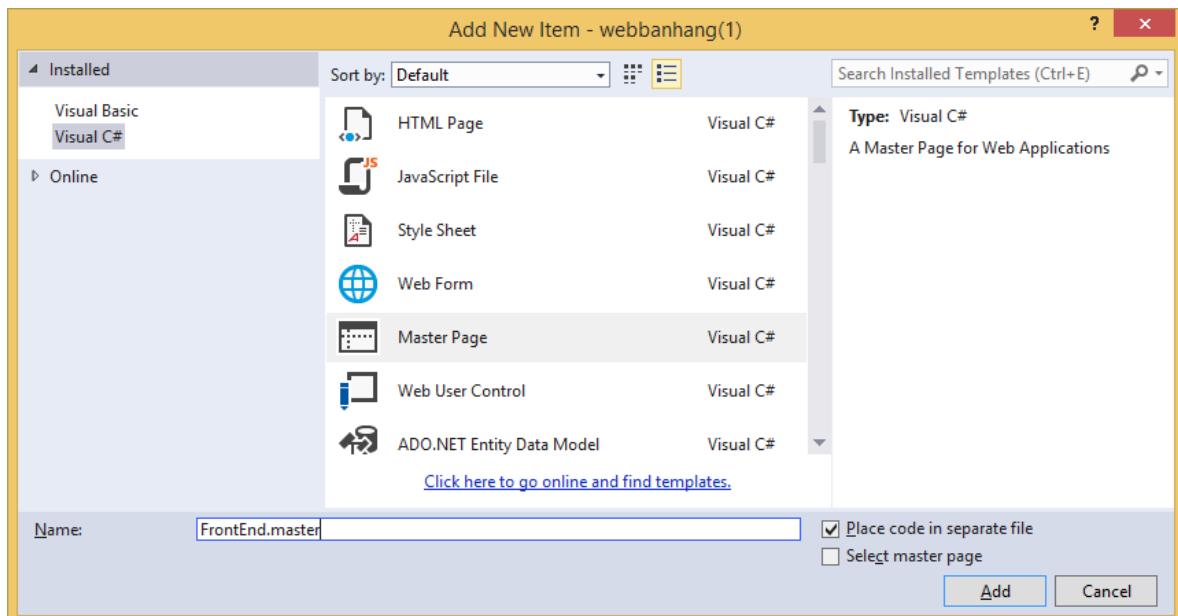


Thực hiện kéo thả các Table, View, Stored Procedure và Function vào **WebBanHang.dbml**. Sau đó lưu lại để tiến hành mapping từ CSDL quan hệ sang CSDL hướng đối tượng. Lớp quản lý toàn bộ là: **WebBanHangDataContext**

16.2 Master Page và trang chủ

16.2.1 Tạo trang Master Page

Chọn **Master Page** và đặt tên **FrontEnd.master**. Sau đó click **Add**



16.2.2 Thiết kế giao diện

Bước 1 : Thêm đoạn code sau ở dưới thẻ **<body>** của trang **FrontEnd.master**

```

<div id="container">
    <div id="banner">
        <a href="Default.aspx" title="Home"></a>
    </div>
    <div id="left">
        <div class="block">
            <div class="tieude">Tìm kiếm</div>
            <div class="noidung">Nội dung</div>
        </div>
        <div class="block">
            <div class="tieude">Giỏ hàng</div>
            <div class="noidung"></div>
        </div>
        <div class="block">
            <div class="tieude">Sản phẩm</div>
            <div class="noidung">Nội dung</div>
        </div>
        <div class="block">
            <div class="tieude">Thống kê</div>
            <div class="noidung">Nội dung</div>
        </div>
    </div>
    <div id="content">
        <asp:ContentPlaceHolder id="ContentPlaceHolderNoiDung" runat="server"></asp:ContentPlaceHolder>
    </div>
    <div id="footer">Thiết kế bởi Nguyễn Văn Tèo</div>
</div>

```

Bước 2 : Tạo **layout.css** định dạng và thêm đoạn code sau:

```

body {margin:0px; padding: 0px;
      font-family: Tahoma 'Times New Roman';
      background-image:url('images/bg-page.jpg');
      background-attachment: fixed;
}
#container{width:1000px; margin: 0px auto; background-color:white;}
#banner{height:120px; position: relative;}
#left{width:250px; float: left; padding:5px;}
#content{width:730px; float: right;
          padding:5px; min-height: 600px;}
#footer{clear:both; padding: 10px; color: white;
         background: transparent url('images/bg-bottom.jpg') repeat top left;
         text-align:center;
}
/*Block*/
.block { margin: 5px; border: 1px solid blue;
padding: 5px; border-radius: 5px;
}
.tieuude{background-color: black;
text-align:center; color: yellow;
padding: 5px; font-size:larger;
border-top-left-radius:5px;
border-top-right-radius:5px;
text-transform:uppercase;
}
.noidung {}
.noidung a{text-decoration:none; color:blue; font-weight:bold; }

```

Bước 4 : Nhúng **layout.css** vào trang **FrontEnd**:

Thêm đoạn code sau vào bên dưới thẻ **<head>** trang **FrontEnd**:

```
<link href="layout.css" rel="stylesheet" type="text/css" />
```

16.2.3 Tạo trang Default.aspx

Bước 1 : Phải chuột vào **webbanhang** chọn **Add -> Add New Item...** dạng **Web Form**, đặt tên **Default.aspx**, check vào ô **Select master page**. Sau đó click **Add** và chọn **FrontEnd.master** và click **OK** ở hộp thoại tiếp theo.

Bước 2 : Thêm đoạn code sau vào trang **Default.aspx**

```

<asp:Content ID="Content2" ContentPlaceHolderID="cphNoiDung" Runat="Server">
    <!--Sản phẩm mới-->
    <h2 class="tieuudeh2">Sản phẩm mới</h2>
    <asp:ListView ID="lvSPMoi" runat="server">
        <ItemTemplate>
            <div class="motsp">
                <div class="tensp"><%# Eval("TenSP") %></div>
                <a href='ChiTietSP.aspx?idSP=<%# Eval("idSP") %>'>
                    <img class="hinhsp" src='upload/sanpham/hinhchinh/<%# Eval("UrlHinh") %>' />
                </a>
                <div class="giasp"> <%# Eval("Gia", "{0:#,##0} đ") %> </div>
                <asp:ImageButton ID="ImageButton1" runat="server"
                    CssClass="nutmua" ImageUrl="~/images/btn_mua_ngay_3.png"
                    CommandArgument='<%# Eval("idSP") %>' OnClick="ImageButton1_Click"/>
            </div>
        </ItemTemplate>
    </asp:ListView>
    <!--End Sản phẩm mới-->
    <!--Sản phẩm bán chạy-->

```

```

<h2 class="tieudeh2">Sản phẩm bán chạy</h2>
<asp:ListView ID="lvSPBanChay" runat="server">
    <ItemTemplate>
        <div class="motsp">
            <div class="tensp"><%# Eval("TenSP") %></div>
            <a href='ChiTietSP.aspx?idSP=<%# Eval("idSP") %>'>
                <img class="hinhsp" src='upload/sanpham/hinhchinh/<%# Eval("UrlHinh") %>' />
            </a>
            <div class="giasp"> <%# Eval("Gia", "{0:#,##0} đ") %> </div>
            <asp:ImageButton ID="ImageButton2" runat="server"
                CssClass="nutmua" ImageUrl="~/images/btn_mua_ngay_3.png"
                CommandArgument='<%# Eval("idSP") %>' OnClick="ImageButton1_Click"/>
        </div>
    </ItemTemplate>
</asp:ListView>
<!--End Sản phẩm bán chạy-->
</asp:Content>

```

Bước 3 : Thêm đoạn code sau vào trang Default.aspx.cs

```

WebBanHangDataContext db = new WebBanHangDataContext();
protected void Page_Load(object sender, EventArgs e)
{
    if(!IsPostBack)
    {
        lvSPMoi.DataSource = db.sanphams.OrderByDescending(p=>p.idSP).Take(8);
        lvSPMoi.DataBind();

        lvSPBanChay.DataSource = db.sanphams.OrderByDescending(p =>
p.donhangchitiets.Count).Take(8);
        lvSPBanChay.DataBind();
    }
}
protected void ImageButton1_Click(object sender, ImageClickEventArgs e)
{
    //lấy mã sp vừa mua
    ImageButton ib = sender as ImageButton;
    int masp = int.Parse(ib.CommandArgument);

    themvaogiohang(masp); //Viết ở trang CommonPage

    //cập nhật WUC_GioHang ở MasterPage
    FrontEnd master = this.Master as FrontEnd;
    master.hienthigiohang();
}

```

Bước 4 : Thêm đoạn code sau vào layout.css

```

.motsp{width: 165px; height:230px;
display:inline-block; border:1px solid blue;
border-radius:7px; position:relative;
text-align:center; margin: 7px 5px;
}
.motsp:hover{box-shadow: 0px 0px 7px red;}
.tensp{color:blue; font-weight:bold; position:absolute; top:3px; left:0px;
width:inherit;}
.hinhsp{height:140px; margin-top: 25px;}
.giasp{color:red; font-style:italic; font-weight:bold;}
.nutmua{position: absolute; bottom:0px;left:7px;}
.tieudeh2{color:#251606; text-transform:uppercase;
font-size:20px; background-color:#f7f7f7; font-weight:bold;
}

```

Kết quả :

SẢN PHẨM MỚI

ASUS Zenfone 2



6,599,000 đ

[MUA NGAY →](#)

ASUS Zenfone 6



5,499,000 đ

[MUA NGAY →](#)

Mobell M390i



1,472,000 đ

[MUA NGAY →](#)

Mobell M210



1,756,000 đ

[MUA NGAY →](#)

Mobell M390 red



2,500,000 đ

[MUA NGAY →](#)

Mobell M116



1,290,000 đ

[MUA NGAY →](#)

Mobell M750



1,489,000 đ

[MUA NGAY →](#)

Mobell M760



1,589,000 đ

[MUA NGAY →](#)

SẢN PHẨM BÁN CHẠY

Asus Zenfone C



Asus ZenFone C

2,390,000 đ

[MUA NGAY →](#)

BlackBerry Dakota



15,000,000 đ

[MUA NGAY →](#)

BlackBerry Bold 9700



12,709,000 đ

[MUA NGAY →](#)

BlackBerry Torch 9800



15,359,000 đ

[MUA NGAY →](#)

iPhone 4



17,709,000 đ

[MUA NGAY →](#)

LG GD350



1,889,000 đ

[MUA NGAY →](#)

BlackBerry Bold 9600



12,709,000 đ

[MUA NGAY →](#)

BlackBerry 9530



15,000,000 đ

[MUA NGAY →](#)

16.2.4 Tạo WebUserControl Danh mục sản phẩm

Bước 1 : Phải chuột vào thư mục **blocks**, chọn **Add -> Add New Item...** và chọn **Web User Control**, đặt tên **WUC_DanhMuc.ascx**. Sau đó click **Add**.

Bước 2 : Thêm đoạn code sau vào **WUC_DanhMuc.ascx**

```
<script>
$(document).ready(function () {
    //ẩn div loại sp
    $(".chungloaisp").next().hide();
    //hiển thị div cáchangsx cua dien thoại (dau tien)
    $(".chungloaisp:first").next().show();
    //bấm chủng loại nào thì ẩn/hiện
    $(".chungloaisp").click(function () {
        $(this).next().toggle();
    });
});
</script>
<div id="danhmuc">
    <asp:ListView ID="lvChungLoai" runat="server"
    OnItemDataBound="lvChungLoai_ItemDataBound">
        <ItemTemplate>
            <asp:HiddenField ID="hfChungLoai"
                Value='<%# Eval("idCL") %>' runat="server" />
            <div class="chungloaisp">
                <%# Eval("TenCL") %>
            </div>
            <!--ListView hiển thị loại sp theo chủng loại-->
            <div>
                <asp:ListView ID="lvLoaiSP" runat="server">
                    <ItemTemplate>
                        <a href='SPTheoLoai.aspx?idLoai=<%# Eval("idLoai") %>'>
                            <%# Eval("TenLoai") %> (<%# Eval("SoLuong") %>)
                        </a>
                    </ItemTemplate>
                </asp:ListView>
            </div>
        </ItemTemplate>
    </asp:ListView>
</div>
```

Bước 3 : Thêm đoạn code sau vào **WUCHangSX.ascx.cs**

```
WebBanHangDataContext db = new WebBanHangDataContext();
protected void Page_Load(object sender, EventArgs e)
{
    if(!IsPostBack)
    {
        lvChungLoai.DataSource = db.chungloais.Where(p => p.AnHien == 1);
        lvChungLoai.DataBind();
    }
}
protected void lvChungLoai_ItemDataBound(object sender, ListViewItemEventArgs e)
{
    //tìm idCL + listview
    HiddenField hfCL = e.Item.FindControl("hfChungLoai") as HiddenField;
    ListView lvloaisp = e.Item.FindControl("lvLoaiSP") as ListView;
    lvloaisp.DataSource = db.loaisps.Where(p => p.idCL == int.Parse(hfCL.Value))
        .OrderBy(p => p.TenLoai)
        .Select(p => new { p.idLoai, p.TenLoai, SoLuong = p.sanphams.Count});
    lvloaisp.DataBind();
```

}

Bước 4 : Thêm đoạn code sau vào **layout.css**

```
#danhmuc{margin-top: 5px; }
#danhmuc a{display: block; text-decoration: none;
padding: 5px; background-color: black;
color: white; border-bottom:1px dashed yellow;
}
#danhmuc a:hover{background-color:red;}
.chungloaisp{background-color:yellow;
text-align:center; padding: 10px 5px 5px 5px;
text-transform:uppercase; margin-top: 10px;
font-weight:bolder; font-size:larger;
}
```

Bước 5 : Thêm đoạn code sau vào dưới <div id="left"> trang **FrontEnd** hoặc kéo thả vào ô Nội dung của block Hàng sản xuất.

```
<div class="block">
<div class="tieude">Sản phẩm</div>
<div class="noidung">
    <uc1:WUC_DanhMuc ID="WUC_DanhMuc1" runat="server" />
</div>
</div>
```

Kết quả :

SẢN PHẨM	
ĐIỆN THOẠI DI ĐỘNG	
Acer (10)	
Apple (5)	
Asus (1)	
BlackBerry (10)	
Cayon (10)	
Dell (10)	
HTC (20)	
LG (24)	
Mobell (7)	
Motorola (8)	
Nokia (30)	
Samsung (20)	
Sony Ericsson (20)	
Vertu (10)	
MÁY TÍNH BẢNG	

16.3 Trang SPTheoLoai.aspx

Bước 1 : Code trang **SPTheoLoai.aspx**

```
<asp:Content ID="Content2" ContentPlaceHolderID="cphNoiDung" Runat="Server">
    <h2 class="tieudeh2" runat="server" id="tieudeloa"></h2>
    <div id="lblPhanTrang" runat="server">
        <asp:Label ID="lblTrang" runat="server" Text="" />
        <asp:Button ID="btnDau" CssClass="btn" runat="server" Text="Đầu"
        OnClick="btnDau_Click" />
        <asp:Button ID="btnTruoc" CssClass="btn" runat="server" Text="Trước"
        OnClick="btnTruoc_Click" />
        <asp:Button ID="btnKe" CssClass="btn" runat="server" Text="Kế"
        OnClick="btnKe_Click" />
        <asp:Button ID="btnCuoi" CssClass="btn" runat="server" Text="Cuối"
        OnClick="btnCuoi_Click" />
    </div>
    <asp:ListView ID="lvSanPham" runat="server">
        <ItemTemplate>
            <div class="motsp">
                <div class="tensp"><%# Eval("TenSP") %></div>
                <a href='ChiTietSP.aspx?idSP=<%# Eval("idSP") %>'>
                    <img class="hinhsp" src='upload/sanpham/hinhchinh/<%# Eval("UrlHinh") %>' />
                </a>
                <div class="giasp"> <%# Eval("Gia", "{0:#,##0} đ") %> </div>
                <asp:ImageButton ID="ImageButton1" runat="server"
                    CssClass="nutmua" ImageUrl="~/images/btn_mua_ngay_3.png"
                    CommandArgument='<%# Eval("idSP") %>' OnClick="ImageButton1_Click"/>
            </div>
        </ItemTemplate>
    </asp:ListView>
</asp:Content>
```

Bước 2 : Thêm đoạn code sau vào **SPTheoLoai.aspx.cs**

```
WebBanHangDataContext db = new WebBanHangDataContext();
int PageSize = 12;//số sp trên 1 trang
protected void Page_Load(object sender, EventArgs e)
{
    if(!IsPostBack)
    {
        //phân trang & hiển thị sản phẩm
        Session["PageCurrent"] = 0;
        hienthisanpham();
    }
}

void hienthisanpham()
{
    string tieude = "";
    List<sanpham> items = null;

    //lấy sản phẩm theo loại
    if (Request.QueryString["idLoai"] != null)
    {
        int idLoai = int.Parse(Request.QueryString["idLoai"].ToString());
        loaisp lo = db.loaisps.SingleOrDefault(p => p.idLoai == idLoai);
        items = lo.sanphams.OrderByDescending(p => p.idSP).ToList();

        tieude = string.Format("{2} --> {0} có {1} sản phẩm.", lo.TenLoai,
lo.sanphams.Count, lo.chungloai.TenCL);
```

```

//sửa title
this.Title = tieude;

}

else if (Request.QueryString["idCL"] != null)
{
    int idCL = int.Parse(Request.QueryString["idCL"].ToString());
    chungloai lo = db.chungloais.SingleOrDefault(p => p.idCL == idCL);
    items = lo.sanphams.ToList();
    tieude = string.Format("{0} có {1} sản phẩm.", lo.TenCL, lo.sanphams.Count);
    //sửa title
    this.Title = tieude;
}
else if (Request.QueryString["Search"] != null)//Search
{
    string search = Request.QueryString["Search"].ToString();
    items = db.sanphams.Where(p => p.TenSP.Contains(search)).ToList();
    tieude = string.Format("Kết quả tìm kiếm theo từ khóa: {0}.", search);
    this.Title = "Kết quả tìm kiếm";
}
else//Hiển thị tất cả
{
    items = db.sanphams.OrderByDescending(p => p.idSP).ToList();
    tieude = string.Format("Tất cả sản phẩm ({0}).", items.Count);
    this.Title = "Tất cả sản phẩm";
}

//tiêu đề
tieudeloai.InnerHtml = tieude;

//trang hiện tại
int PageCurrent = (Session["PageCurrent"] != null) ? (int)Session["PageCurrent"]
: 0;
//tổng số trang
int PageCount = (int)Math.Ceiling(1.0 * items.Count / PageSize);
//lưu session 2 biến giữ trang hiện tại & trang cuối
Session["PageCurrent"] = PageCurrent;
Session["LastPageNo"] = PageCount - 1;

lblPhanTrang.Visible = (PageCount > 0);
lblTrang.Text = string.Format("Trang: {0}/{1}.", PageCurrent + 1, PageCount);

lvSanPham.DataSource = items.Where(p => p.AnHien == 1).Skip(PageCurrent *
PageSize).Take(PageSize);
lvSanPham.DataBind();

//bỏ qua nút Trước nếu đang ở trang đầu
btnDau.Enabled = btnTruoc.Enabled = (PageCurrent > 0);
btnCuoi.Enabled = btnKe.Enabled = (PageCurrent < (int)Session["LastPageNo"]);
}

protected void ImageButton1_Click(object sender, ImageClickEventArgs e)
{
    //lấy mã sp vừa mua
    ImageButton ib = sender as ImageButton;
    int masp = int.Parse(ib.CommandArgument);
    themvaogiohang(masp);

    //cập nhật WUC_GioHang ở MasterPage
    FrontEnd master = this.Master as FrontEnd;
    master.hienthigiohang();
}

```

```

protected void btnKe_Click(object sender, EventArgs e)
{
    Session["PageCurrent"] = (int)Session["PageCurrent"] + 1;
    hienthisanpham();
}
protected void btnTruoc_Click(object sender, EventArgs e)
{
    Session["PageCurrent"] = (int)Session["PageCurrent"] - 1;
    hienthisanpham();
}
protected void btnDau_Click(object sender, EventArgs e)
{
    Session["PageCurrent"] = 0;
    hienthisanpham();
}
protected void btnCuoi_Click(object sender, EventArgs e)
{
    Session["PageCurrent"] = Session["LastPageNo"];
    hienthisanpham();
}

```

Kết quả : Chọn hãng **Asus**

The screenshot shows a web browser displaying an online shop for mobile phones. The page title is "Điện thoại di động --> Asus". The URL is "localhost:59144/SPTheoLoai.aspx?idLoai=59". The header features a shopping cart icon, the text "WELCOME TO OUR ONLINE SHOP", and the "Master NHẤT NGHỆ" logo. The navigation menu includes TRANG CHỦ, GIỚI THIỆU, SẢN PHẨM, KHUYẾN MÃI, TIN TỨC, BẢN ĐỒ, LIÊN HỆ, and TÀI KHOẢN. On the left, there's a sidebar with a "GIỎ HÀNG" section showing 0 items and a "SẢN PHẨM" section listing categories like "ĐIỆN THOẠI DI ĐỘNG", "Acer (10)", "Apple (5)", "Asus (3)", and "BlackBerry (10)". The main content area displays a heading "ĐIỆN THOẠI DI ĐỘNG --> ASUS CÓ 3 SẢN PHẨM." with a product grid. It shows three Asus Zenfone models: Zenfone 2 (price 6,599,000đ), Zenfone 6 (price 5,499,000đ), and Zenfone C (price 2,390,000đ). Each product has a "MUA NGAY →" button below it.

GIỎ HÀNG
 0 sản phẩm
 0 vnđ
[Xem giỏ hàng](#)

SẢN PHẨM
ĐIỆN THOẠI DI ĐỘNG
[Acer \(10\)](#)
[Apple \(5\)](#)
[Asus \(3\)](#)
[BlackBerry \(10\)](#)

ĐIỆN THOẠI DI ĐỘNG CÓ 187 SẢN PHẨM.

Trang: 1/16. [Đầu](#) [Trước](#) [Kép](#) [Cuối](#)


6,599,000 đ
[MUA NGAY →](#)


5,499,000 đ
[MUA NGAY →](#)


1,472,000 đ
[MUA NGAY →](#)


1,756,000 đ
[MUA NGAY →](#)

16.4 Trang ChiTietSP.aspx

Tab “Thông số kỹ thuật”:

ASUS ZENFONE 6



Phablet cao cấp nhất của Asus hiện nay, thiết kế lịch lãm, cạnh dưới giả kim loại với vân tròn đồng tâm giúp máy bắt mắt và sang trọng

Lượt xem: 6

5,499,000 vnđ

[MUA NGAY →](#)

[Thông số kỹ thuật](#)

[Hình ảnh](#)

[Video](#)

[Bài viết](#)

[Bình luận](#)

Thông số kỹ thuật Asus Zenfone 6 A601

Màn hình: HD, 6.0", 720 x 1280 pixels

CPU: Intel Atom Z2560, 2 nhân, 1.6 GHz

RAM: 2 GB

Hệ điều hành: Android 4.3 (Jelly Bean)

Camera chính: 13 MP, Quay phim FullHD 1080p@30fps

Camera phụ: 2.0 MP

Bộ nhớ trong: 16 GB

Tab “Hình ảnh” sản phẩm:

ASUS ZENFONE 6



Phablet cao cấp nhất của Asus hiện nay, thiết kế lịch lãm, cạnh dưới giả kim loại với vân tròn đồng tâm giúp máy bắt mắt và sang trọng

Lượt xem: 6

5,499,000 vnđ

MUA NGAY →

Thông số kỹ thuật

Hình ảnh

Video

Bài viết

Bình luận



Tab “Video” giới thiệu sản phẩm:

ASUS ZENFONE 6



Phablet cao cấp nhất của Asus hiện nay, thiết kế lịch lãm, cạnh dưới giả kim loại với vân tròn đồng tâm giúp máy bắt mắt và sang trọng

Lượt xem: 6

5,499,000 vnđ

MUA NGAY →

Thông số kỹ thuật

Hình ảnh

Video

Bài viết

Bình luận

Schannel - Đánh giá Zenfone 6: Thiết kế, màn hình... Phần 1- CellphoneS



Tab “Bình luận” (comment bằng Account Facebook):

IPHONE 4



- * Màn hình TFT, 16 triệu màu, rộng 3.5 inches
- * Hỗ trợ đa Sim: Không
- * Máy ảnh 5.0 MP (2592 x 1944 pixels)
- * Đài FM tích hợp: Không
- * Nghe nhạc: MP3, WAV, AAC+, AAC++, WMA, AAC
- * Xem Phim: MP4, 3GP, WMV
- * Kết nối 3G: HSDPA

Lượt xem: 6

17,709,000 vnđ

MUA NGAY →

Thông số kỹ thuật

Hình ảnh

Video

Bài viết

Bình luận



Hien Luong Tran Hy

Cũng đăng lên Facebook

Đăng với tư cách là Hien Luong Tran Hy • **Bình luận**



Hien Luong Tran Hy - Giảng viên tại Trung Tâm Đào Tạo CNTT Nhất Nghệ

Sản phẩm này giờ giá giảm nhiều rồi

Trả lời · Thích · Bỏ theo dõi bài đăng · 2 giây trước

Plugin xã hội của Facebook

Gợi ý phần thiết kế các Tab Control (Sử dụng jQuery UI):

```

<%--Phần tab các nội dung --%>
<div id="mytabs">
    <!-- Phần tiêu đề -->
    <ul>
        <li><a href="#tskt">Thông số kỹ thuật</a></li>
        <li><a href="#hinhanh">Hình ảnh</a></li>
        <li><a href="#video">Video</a></li>
        <li><a href="#baiviet">Bài viết</a></li>
        <li><a href="#binhluan">Bình luận</a></li>
    </ul>
    <!-- Phần nội dung -->
    <div id="binhluan">
        <asp:ListView ID="lvBinhLuan" runat="server">
            <ItemTemplate>
                <!--Quảng code HTML5 gen-->
                <div id="fb-root"></div>
                <script> (function (d, s, id) {
                    var js, fjs = d.getElementsByTagName(s)[0];
                    if (d.getElementById(id)) return;
                    js = d.createElement(s); js.id = id;
                    js.src =
//connect.facebook.net/vi_VN/sdk.js#xfbml=1&appId=1476057245998516&version=v2.0";
                    fjs.parentNode.insertBefore(js, fjs);
                }(document, 'script', 'facebook-jssdk'));</script>

                    <div class="fb-share-button" data-href='<%# Eval("DiaChiWeb") %>' data-
layout="icon_link"></div>

                    <div class="fb-comments" data-href='<%# Eval("DiaChiWeb") %>' data-
numposts="5" data-colorscheme="light"></div>
                <!--End Quảng code HTML5 gen-->
            </ItemTemplate>
        </asp:ListView>
    </div>

    <div id="video">
        <asp:ListView ID="lvVideo" runat="server">
            <ItemTemplate>
                <iframe width="640" height="390" src='//www.youtube.com/embed/<%#
Eval("value") %>' frameborder="0" allowfullscreen></iframe>
            </ItemTemplate>
        </asp:ListView>
    </div>

    <div id="hinhanh">
        <asp:ListView ID="lvHinhAnh" runat="server">
            <ItemTemplate>
                <img src = 'upload/sanpham/hinhphu/<%# Eval("urlHinh") %>' />
            </ItemTemplate>
        </asp:ListView>
    </div>
    <div id="tskt">
        <asp:Label ID="lblChiTietSP" runat="server" Text=""></asp:Label>
    </div>
</div>
<script>
$(document).ready(function () {
    $("#mytabs").tabs({ event: "mouseover" });
});
</script>

```

Phải thay appId ứng với website của bạn

16.5 Trang đăng nhập (Login.aspx)

The screenshot shows a web browser window with the URL `localhost:59144/Login.aspx`. The page title is "Đăng nhập". The main content area features a red shopping cart icon and the text "WELCOME TO OUR ONLINE SHOP". On the right, there is a promotional image of a woman holding shopping bags. The navigation menu includes links for TRANG CHỦ, GIỚI THIỆU, SẢN PHẨM, KHUYẾN MÃI, TIN TỨC, BẢN ĐỒ, LIÊN HỆ, and TÀI KHOẢN. A sidebar on the left displays "GIỎ HÀNG" with 0 items and a link to "Xem giỏ hàng". The central "ĐĂNG NHẬP" form has fields for "Username" (filled with "tí") and "Password" (filled with ""). A red "Đăng nhập" button is visible. Below the form are links for "Quên mật khẩu", "Đăng ký Tài khoản", and "Sai mật khẩu".

This screenshot shows the same Login.aspx page after an unsuccessful login attempt. The "Password" field now contains two asterisks ("**"). Below the "ĐĂNG NHẬP" form, a new message "Sai mật khẩu." (Incorrect password.) is displayed in blue text.

The screenshot shows a web browser window with the URL `localhost:59144/SPTheoLoai.aspx?idLoai=59`. The page title is "Điện thoại di động --> Asus". The main content area features a red shopping cart icon and the text "WELCOME TO OUR ONLINE SHOP". On the right, there is a promotional image of a woman holding shopping bags. The navigation menu includes links for TRANG CHỦ, GIỚI THIỆU, SẢN PHẨM, KHUYẾN MÃI, TIN TỨC, BẢN ĐỒ, LIÊN HỆ, and TÀI KHOẢN. A sidebar on the left displays "MY PROFILE" with a user icon and the name "Tí". The central area displays a heading "ĐIỆN THOẠI DI ĐỘNG --> ASUS CÓ 3 SẢN PHẨM." Below it, a pagination message "Trang: 1/1. Đầu Trước Ké Cuối" is shown. Three mobile phone models are listed in a grid: "ASUS Zenfone 2" (purple), "ASUS Zenfone 6" (white and black), and "Asus ZenFone C" (red, yellow, and black). Each phone has its price and a "MUA NGAY" button below it.

16.6 Xử lý nút Mua hàng

Khi click vào nút **Mua hàng**, tiến hành lưu thông tin vào giỏ hàng tạm (sử dụng Session):

WELCOME TO OUR ONLINE SHOP

ĐIỆN THOẠI DI ĐỘNG --> ASUS CÓ 3 SẢN PHẨM.

Trang: 1/1. Đầu Trước Ké Cuối

ASUS Zenfone 2 6,599,000 đ MUA NGAY →	ASUS Zenfone 6 5,499,000 đ MUA NGAY →	Asus Zenfone C 2,390,000 đ MUA NGAY →
---	---	---

GIỎ HÀNG
0 sản phẩm
0 vnđ
Xem giỏ hàng

SẢN PHẨM

ĐIỆN THOẠI DI ĐỘNG

- Acer (10)
- Apple (5)
- Asus (3)
- BlackBerry (10)

Thông tin giỏ hàng sẽ cập nhật tức thì trên cửa sổ MasterPage.

WELCOME TO OUR ONLINE SHOP

ĐIỆN THOẠI DI ĐỘNG --> ASUS CÓ 3 SẢN PHẨM.

Trang: 1/1. Đầu Trước Ké Cuối

ASUS Zenfone 2 6,599,000 đ MUA NGAY →	ASUS Zenfone 6 5,499,000 đ MUA NGAY →	Asus Zenfone C 2,390,000 đ MUA NGAY →
---	---	---

GIỎ HÀNG
1 sản phẩm
6,599,000 vnđ
Xem giỏ hàng

SẢN PHẨM

ĐIỆN THOẠI DI ĐỘNG

- Acer (10)
- Apple (5)
- Asus (3)
- BlackBerry (10)

Sử dụng Ajax, jQuery để cập nhật thông tin Giỏ hàng tự động tránh load lại trang.

16.7 Trang GioHang.aspx

Mô tả giao diện:

The screenshot shows a web browser window for 'localhost:59144/GioHang.aspx'. The page title is 'WELCOME TO OUR ONLINE SHOP'. On the left, there's a sidebar with categories like 'GIỎ HÀNG' (2 sản phẩm, 12,098,000 vnđ), 'SẢN PHẨM' (ĐIỆN THOẠI DI ĐỘNG), and sub-categories for Acer, Apple, Asus, BlackBerry, and Canyon. The main content area shows a table titled 'GIỎ HÀNG CỦA BẠN' with two items:

Mã SP	Hình	Tên sản phẩm	Số lượng	Đơn giá	Thành tiền	Xóa
680		ASUS Zenfone 2	<input type="text" value="1"/>	6,599,000 đ	6,599,000 đ	Xóa
679		ASUS Zenfone 6	<input type="text" value="1"/>	5,499,000 đ	5,499,000 đ	Xóa

Tổng chi phí: 12,098,000 đ.

[Xóa tất cả](#) [Cập nhật](#) [Mua tiếp](#) [Thanh toán](#)

Hệ thống chức năng:

- Xóa giỏ hàng:** Xóa tất cả các hàng hóa trong giỏ hàng.
- Cập nhập:** Tiến hành cập nhật trường số lượng cho từng sản phẩm.
- Tiếp tục mua hàng:** Quay về trang chủ.
- Thanh toán:** Chuyển sang trang ThanhToan.aspx để tiến hành điền thông tin giao hàng cho khách hàng.

16.8 Trang ThanhToan.aspx

Nếu chưa đăng nhập, hiển thị trang Login để khách hàng đăng nhập hoặc đăng ký:

The screenshot shows a web browser window for 'localhost:59144/Login.aspx'. The page title is 'WELCOME TO OUR ONLINE SHOP'. The main content area shows a login form titled 'ĐĂNG NHẬP' with fields for 'Username' (containing 'ti') and 'Password' (containing '').

Nếu đã đăng nhập thì điền thông tin giao hàng và bấm nút **Xác nhận đặt hàng**:

THÔNG TIN ĐẶT HÀNG

Người đặt hàng	Tí
Thời điểm đặt hàng	28/02/2015 18:17:11
Tên người nhận	Nguyễn Minh Huy
Địa điểm giao hàng	80 Bà Huyện Thanh Quan
Ghi chú	Liên hệ trước ghi giao

Xác nhận đặt hàng

Mã	Hình	Tên sản phẩm	Đơn giá	Số lượng	Thành tiền
680		ASUS Zenfone 2	6,599,000 đ	1	6,599,000 đ
679		ASUS Zenfone 6	5,499,000 đ	1	5,499,000 đ

Sau khi “Xác nhận Đặt hàng” sẽ hiển thị thông tin đặt hàng ra màn hình, đồng thời chèn thông tin Đơn đặt hàng vào CSDL, sau đó xóa giỏ hàng.

THÔNG TIN ĐẶT HÀNG

Mã đơn đặt hàng	9
Người đặt hàng	Tí
Thời điểm đặt hàng	28/02/2015 18:25:10
Tên người nhận	Nguyễn Minh Huy
Địa điểm giao hàng	80 Bà Huyện Thanh Quan
Ghi chú	Liên hệ trước ghi giao

DANH SÁCH MẶT HÀNG ĐÃ ĐẶT

Mã	Hình	Tên sản phẩm	Đơn giá	Số lượng	Thành tiền
680		ASUS Zenfone 2	6,599,000 đ	1	6,599,000 đ
679		ASUS Zenfone 6	5,499,000 đ	1	5,499,000 đ

16.9 Trang lịch sử mua hàng

Khách hàng có thể xem lịch sử giao dịch mua hàng trên website cũng như chi tiết cho từng đơn đặt hàng. Trang này chỉ hiển thị khi người dùng đã đăng nhập.

Mã đơn hàng	Trạng thái	Ngày đặt hàng	Tổng tiền
3	Hoàn tất	24/01/2015 12:04:00	3,778,000 đ Chi tiết
2	Hủy	14/01/2014 12:35:00	43,068,000 đ Chi tiết

Chi tiết đơn hàng: 3

Mã đơn hàng: 3, trạng thái: Hoàn tất.
 Người nhận: Nguyễn Văn Tèo.
 Địa điểm giao: 280 An Dương Vương, Q5.
 Ngày đặt: 24/01/2015 12:01:00, tổng tiền: 3778000.
 Ghi chú: Liên hệ trước.

Mã	Tên hàng hóa	Đơn giá	Số lượng	Thành tiền
645	LG GD350	1889000	2	3778000

16.10 Phản Quản trị

16.10.1 Layout trang Quản trị

Menu các danh mục:

16.10.2 Quản lý Chủng loại

Quản lý chủng loại - Trang chủ

localhost:59144/Admin/ChungLoai.aspx

Admin Panel

QUẢN LÝ CHỦNG LOẠI

Mã chủng loại	1			
Tên chủng loại	Điện thoại di động			
Thứ tự	1			
Ẩn hiện	<input type="radio"/> Hiện <input checked="" type="radio"/> Ẩn			
<input type="button" value="Thêm mới"/> <input type="button" value="Xóa"/> <input type="button" value="Sửa"/>				
	Mã chủng loại	Tên chủng loại	Thứ tự	Ẩn hiện
Chọn	1	Điện thoại di động	1	<input type="radio"/> Hiện <input checked="" type="radio"/> Ẩn
Chọn	2	Laptop	2	<input checked="" type="radio"/> Hiện <input type="radio"/> Ẩn
Chọn	3	Máy tính bảng	3	<input checked="" type="radio"/> Hiện <input type="radio"/> Ẩn
Chọn	5	Phụ kiện	4	<input type="radio"/> Hiện <input checked="" type="radio"/> Ẩn

Copyright 2015 Nhất Nghệ Shop. Designed and Code by HIENLTH®. All Rights Reserved.

16.10.3 Quản lý Đơn hàng

Quản lý Đơn hàng

localhost:59144/Admin/QLDonHang.aspx

Admin Panel

QUẢN LÝ ĐƠN HÀNG

STT	Mã đơn hàng	Người mua	Thời điểm đặt hàng	Tổng tiền	Trạng thái	Chi tiết
1	5	Lương Hiền	24/02/2015 10:28:34 AM	4,780.000 vnđ	Hoàn tất	Chi tiết
2	4	Lương Trần Hy Hiền	24/02/2015 10:20:15 AM	22,489.000 vnđ	Hoàn tất	Chi tiết
3	3	Tèo	24/01/2015 12:04:00 AM	3,778.000 vnđ	Hoàn tất	Chi tiết

Quản lý Đơn hàng

localhost:59144/Admin/QLDonHang.aspx

Admin Panel

TRANG CHỦ | MUA HÀNG | SẢN PHẨM | QUẢN LÝ ĐƠN HÀNG | QUẢN LÝ DANH MỤC | THỐNG KÊ | BÀI VIẾT | ĐĂNG XUẤT

QUẢN LÝ ĐƠN HÀNG

Mã đơn hàng	5
Người mua hàng	Lương Hiển
Thời điểm đặt hàng	24/02/2015 10:28:34
Tên người nhận	<input type="text" value="Lương Hiển"/>
Địa chỉ giao hàng	<input type="text" value="175/1A Nguyễn Huỳnh Đức"/>
Trạng thái	<input type="button" value="Hoàn tất"/>
Ghi chú	<input type="text"/>

Cập nhật Đơn hàng **Danh sách Đơn hàng**

Danh sách hàng đặt:

Mã sản phẩm	Tên sản phẩm	Giá	Số lượng	Thành tiền
0	Asus Zenfone C	2,390,000 vnđ	2	4,780,000 vnđ

Copyright 2015 Nhất Nghệ Shop. Designed and Code by HIENLTH®. All Rights Reserved.

16.10.4 Quản lý Sản phẩm

Quản lý Sản phẩm

localhost:59144/Admin/QLSanPham.aspx

TRANG CHỦ | MUA HÀNG | SẢN PHẨM | QUẢN LÝ ĐƠN HÀNG | QUẢN LÝ DANH MỤC | THỐNG KÊ | BÀI VIẾT | ĐĂNG XUẤT

QUẢN LÝ SẢN PHẨM

Chủng loại: Điện thoại di động

Tìm theo tên: **Search** **Thêm mới Sản phẩm**

Mã	Tên sản phẩm	Hình	Giá	Số lượng tồn	
680	ASUS Zenfone 2		6,599,000 vnđ	11	Chi tiết
679	ASUS Zenfone 6		5,499,000 vnđ	11	Chi tiết
676	Mobell M390i		1,472,000 vnđ	200	Chi tiết
675	Mobell M210		1,756,000 vnđ	200	Chi tiết
674	Mobell M390 red		2,500,000 vnđ	200	Chi tiết

12345678910...

Quản lý Sản phẩm

localhost:59144/Admin/QLSanPham.aspx

TRANG CHỦ MUA HÀNG SẢN PHẨM QUẢN LÝ ĐƠN HÀNG QUẢN LÝ DANH MỤC THỐNG KÊ BÀI VIẾT ĐĂNG XUẤT

QUẢN LÝ SẢN PHẨM

Chủng loại: Điện thoại di động

Mã sản phẩm: 680

Tên sản phẩm: ASUS Zenfone 2

Tên khống dấu: Asus-Zenfone-2

Loại: Asus

Mô tả: Tóm tắt Thông số kỹ thuật
- Màn hình Full HD, 5.5 inches
- CPU 4 nhân, 2.3 GHz
- RAM 4GB ROM 16GB
- Camera 13MP/ 5MP - Pin 3000 mAh

Ngày cập nhật: 28/02/2015

Giá: 6599000

Hình chính:

Choose File No file chosen

Số lần xem: 6

Số lượng tồn: 11

Ghi chú: Đang chờ hàng

Quản lý Sản phẩm

localhost:59144/Admin/QLSanPham.aspx

Hiện

Chi tiết sản phẩm

4G:	LTE Cat4 150/50 Mbps
SIM:	2 SIM (Micro-SIM)
Kích thước:	152.5 x 77.2 x 10.9 mm (6.00 x 3.04 x 0.43 in)
Trọng lượng:	170 g (6.00 oz)
Màn hình:	Cảm ứng điện dung IPS, 16 triệu màu

Thêm Sửa Xóa Danh sách

Cách nhau dấu chấm phẩy (phải upload lên YouTube trước)

Video (Youtube URL): [eyJGUc8_g9w](https://www.youtube.com/watch?v=eyJGUc8_g9w);js8wCnPpUig;oki0tmoM0JU

Hình phụ:

Choose Files No file chosen

Copyright 2015 Nhất Nghệ Shop. Designed and Code by HIENLTH®. All Rights Reserved.

Chú ý:

- Phải sử dụng CKEditor cho trường chi tiết sản phẩm.
- Đối với Video demo/giới thiệu sản phẩm, quản trị phải upload lên YouTube trước, sau đó lấy trường Value các video để lưu thông tin video. Nếu có nhiều video cần cách nhau bởi dấu ";" làm điều kiện phân tách.
- Hình phụ cho phép upload nhiều file cùng một lúc. Code upload nhiều file như sau (hỗ trợ từ .NET Framework 4.5 trở lên):

```
//Upload nhiều file: Hình phụ
if (fu_HinhPhu.HasFiles)
{
    int i = 1;
    foreach (HttpPostedFile myfile in fu_HinhPhu.PostedFiles)
    {
        myfile.SaveAs(Server.MapPath("~/upload/sanpham/hinhphu/") +
            myfile.FileName);
        sanpham_hinh sph = new sanpham_hinh
        {
            idSP = sp.idSP,
            anhien = 1,
            urlHinh = myfile.FileName,
            stt = i
        };
        i++;
        db.sanpham_hinhs.InsertOnSubmit(sph);
    }//end foreach
    db.SubmitChanges();
}
```

16.10.5 Quản lý Tài khoản người dùng

STT	Mã User	Họ tên	Username	Email	Hình	Ngày đăng ký	Giới tính	Active	Xem chi tiết
1	20	Tí	tí	tí@localhost.com		20/01/2009	1	1	Xem chi tiết
2	21	Gia Hu	giahu	giahu@localhost.com		22/01/2009	0	1	Xem chi tiết
3	22	Tèo	teo	teo@localhost.com		07/02/2009	1	1	Xem chi tiết
4	23	Lương Trần Hy Hiển	hienlth	hyhien@gmail.com		13/11/2014	1	1	Xem chi tiết
5	34	Lương Hiền	hpt7777	hpt7777@gmail.com		27/02/2015	0	1	Xem chi tiết

Quản lý tài khoản - Trang x localhost:59144/Admin/TaiKhoanUser.aspx

QUẢN LÝ TÀI KHOẢN NGƯỜI DÙNG

Cập nhật thành công

Mã user	34
Họ tên	Lương Hiển
Username	hpt7777
Password	8FC377D4AE3E8F2070FE
Địa chỉ	175/1A Nguyễn Huỳnh Đức
Điện thoại	0989366990
Email	hpt7777@gmail.com
Ngày đăng ký	27/02/2015
Nhóm User	Khách hàng
Ngày sinh	2/14/2005 12:00:00 AM
Giới tính	<input checked="" type="radio"/> Nam <input type="radio"/> Nữ
Active	<input checked="" type="radio"/> Active <input type="radio"/> Chưa Active
Randomkey	yCDSSrnc0G
Ngày hết hạn	
Expiration	<input checked="" type="radio"/> Còn hạn <input type="radio"/> Hết hạn
 Hình	
<input type="button" value="Choose File"/> No file chosen.	
<input type="button" value="Thêm mới"/> <input type="button" value="Xóa"/> <input type="button" value="Sửa"/> <input type="button" value="Danh sách"/>	

Copyright 2015 Nhất Nghệ Shop. Designed and Code by HIENLTH®. All Rights Reserved.

16.11 Yêu cầu thêm

- Sử dụng Captcha cho trường hợp xác thực.
- Gửi mail khi khách hàng đăng ký mới, đặt hàng, đổi mật khẩu.
- Cấu hình Routing.

Chương 17: UPLOAD FILE LÊN HOST

Các bước thực hiện

Bước 1 : Truy cập trang <https://somee.com>, chọn mục **Free. Net hosting**. Sau đó click vào **Learn more**

The screenshot shows the homepage of [Somee International](https://somee.com). It features a large image of a modern building with a green wall. Below the image, there are four main service offerings:

- Windows 2012R2 / 2008R2 VIRTUAL SERVER from \$19.95**: Includes a "Learn More" button.
- MS SQL Hosting \$7.85**: Includes a "Learn More" button.
- Windows hosting \$7.95**: Includes a "Learn More" button.
- Virtual Server \$19.95**: Includes a "Learn More" button.

Bước 2 : Click vào **Order now**

The screenshot shows the [Free ASP.NET MVC hosting](https://somee.com/FreeAspNetHosting.aspx) page. It compares two hosting packages:

- Free Hosting package** (Free):
 - 1 x Hosting plan "Freebie"
 - Forced advertising
 - Storage capacity: 150MB
 - Monthly transfer: 5GB/Month
 - Web domains: 1
 - ASP.NET 4.5/4.0/3.5/2.0/1.1
 - AJAX 3.5/1.0
 - Silverlight
 - MS Access 2003, 2007
 - Dedicated web application pool
 - 1 x Email plan "Forwarder"
 - 1 x MS SQL Plan "Novice"
 - MS SQL database size: 15MB
 - MS SQL log size: 20MB
 - Backup storage size: 40MB
- Free Windows ASP.NET hosting** (\$0.00):
 - Free web hosting on Windows 2012, Windows 2008R2 or Windows 2003. Supported features: IIS 8.0/7.5/6.0; ASP; ASP.NET 1.1/2.0/3.0/3.5/4.0/4.5; MVC 1.0/2.0/3.0/4.0; PHP 5; MS SQL Express 2012/2008R2/2005 and some standard components.
 - Free Windows web hosting with fast registration
 - Instant Hosting setup
 - Limitations and restrictions of the free web hosting package
 - Keeping your site active

Bước 3: Điền thông tin để đăng ký tài khoản nếu chưa có tài khoản. Hoặc đăng nhập nếu đã có tài khoản

Nếu đã có tài khoản thì click vào đây

Click here if already registered
để đăng nhập

Bước 4 : Check vào 2 ô checkbox và click Continue (Nếu chọn đăng ký và đã qua bước điền thông tin đầy đủ)

Total: \$0.00 Continue
Cancel

By checking this checkbox you agree to our Terms of service! [Click here to read.](#)

By checking this checkbox you agree to our Privacy policy! [Click here to read.](#)

Bước 5 : Đặt tên Site name và click Create website

Hosting plan: Hosting plan "Freebie"

⚠ Although your hosting plan supports global domains you still need to provide initial default domain name which is hosted within our zone. You will be able to add additional domains later in control panel.

Site name (Subdomain): ➡

Zone name: ➡

Operating system: Windows Server 2012 (IIS 8.0, ASP, ASP.Net v2.0-4.5, PHP) ➡

ASP.Net version: 4.0, 4.5 ➡

Site title:

Site description:

ⓘ Your site will have default domain names as 'Sub domain'.'Zone name' and www.'Sub domain'.'Zone name'. Also, if supported by hosting plan, you can use your own domain names (registered with domain name registrar). DNS record will be created instantly, but because of DNS replication delay it may take up to 24 hours for your site to be visible to all users on the Internet. You can try to access your site right after it's created. If it's not working retry it every 30 minutes.

The process may take up to several minutes. Please be patient.

➡ Create website ➡

Bước 6 : Chọn mục Database. Sau đó đặt tên database, chọn phiên bản SQL và click Create empty database

SOME.com User home Store System help Ask a question Search help

User menu

- User (nhatnghe123)
 - Create support ticket
 - Support tickets
 - Checkout
- Account
 - Password change
 - Addresses
 - Payment methods
 - Invoices
 - Billing options
 - Active packages
- Managed products
 - Virtual servers
- Websites
 - nhatnghe123.somee.cc
 - Applications
 - File manager
 - Domains
 - Default documents
 - Storage
 - IIS Statistics
 - Custom errors
 - Tasks
 - MS SQL
 - Databases ➡
 - Logins
- Email
 - Add new mail domain

Create empty MS SQL database

MS SQL plan: MS SQL Plan "Novice"

Database name (7-30 symbols): ➡

Zone name: ➡

MS SQL login: - Generate new login - ➡

MS SQL Server version: MS SQL 2012 Express ➡

Warning! Remember when choosing MS SQL Server version that they are not backward compatible! So, if, at your development environment you have server with lower MS SQL version than you'll choose here, you won't be able to move the database from here back to your development environment without upgrading your MS SQL server!

The process may take up to several minutes. Please be patient.

➡ Create empty database ➡

Bước 7 : Restore database

The screenshot shows the SOMEE.com user interface with a sidebar on the left containing various account and management options. In the main area, there's a section titled 'Restore MS SQL database' with two tabs: 'Restore from existing backup' and 'Upload and restore'. Under 'Restore from existing backup', the 'From backup location:' dropdown is set to 'thegioidongday.backup.somee.com (Backup)'. Below it, a 'Select the backup file:' dropdown shows a directory structure with a single file 'thegioidongday.bak'. A red arrow points to the 'Restore database' link in the sidebar. Another red arrow points to the 'Choose File' button in the 'Upload and restore' section.

Bước 8 : Sửa đoạn code sau ở **Web.config** của trang **webbanhang**

```
<connectionStrings>
    <add name="thegioidongday" connectionString="Data
Source=.\SQLEXPRESS;Initial Catalog=thegioidongday;Integrated Security=True"/>
</connectionStrings>
```

Thành

```
<connectionStrings>
    <add name="thegioidongday" connectionString="workstation
id=thegioidongday.mssql.somee.com;packet size=4096;user
id=nhatnghe123_SQLLogin_1;pwd=slzzuhowb;data
source=thegioidongday.mssql.somee.com;persist security info=False;initial
catalog=thegioidongday"/>
</connectionStrings>
```

Lưu ý : đoạn code trong **connectionString="..."**

Lấy từ

MS SQL Server address: **thegiodongday.mssql.somee.com**
 Login name: **nhatnghe123_SQLLogin_1**
 Login password: **slzzuuhowb**
 Connection string: **workstation id=thegiodongday.mssql.somee.com;packet size=4096;user id=nhatnghe123_SQLLogin_1;pwd=slzzuuhowb;data source=thegiodongday.mssql.somee.com;persist security info=False;initial catalog=thegiodongday**

↓

FTP address to backups: **thegiodongday.backup.somee.com**
 FTP username: **nhatnghe123**
 FTP credentials (Username and password) are the same as your Username and password.

Data file size, Max/Used/Usage: **15MB / 5.00MB / 33.33%**
 Log file size, Max/Used/Usage: **20MB / 1.00MB / 5.00%**

Bước 9 : Upload file lên host

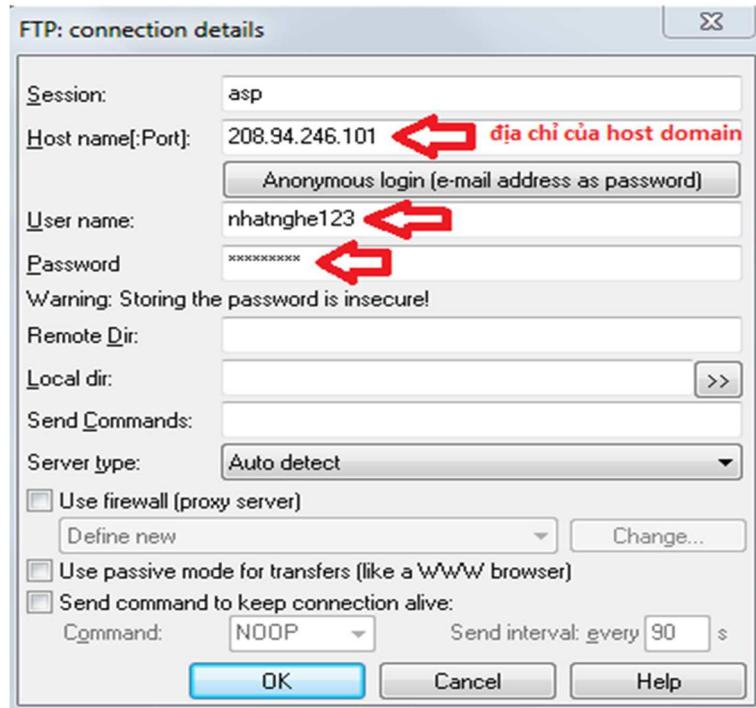
Dùng chương trình **Total Commander**

Điền thông tin

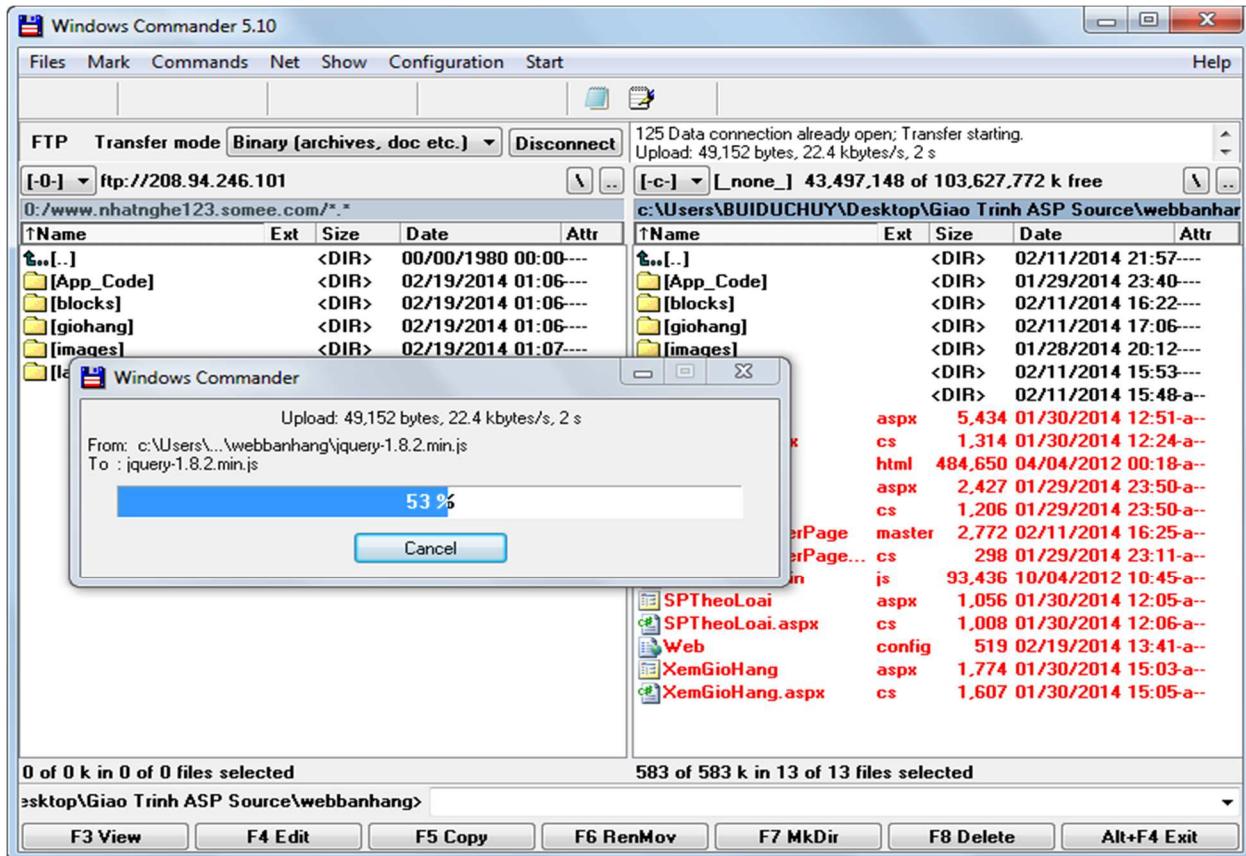
Host name : địa chỉ IP hoặc tên domain

User name : user name mà domain đã cấp cho mình

Password : password mà domain đã cấp cho mình



Kéo những trang, những thư mục qua host



Bước 10 : Truy cập vào host vừa đăng ký để kiểm tra

