TRƯỜNG CAO ĐẲNG NGHỀ CÔNG NGHIỆP HÀ NỘI
Tác giả: Nguyễn Thị Nhung
Bùi Quang Ngọc

GIÁO TRÌNH
LẬP TRÌNH WEB ĐỘNG VỚI ASP.NET
(Lưu hành nội bộ)

Hà Nội năm 2011

BÀI 1. TỔNG QUAN VỀ ASP.NET

A. Mục tiêu

- Tìm hiểu các kỹ thuật lập trình ứng dụng web và các mô hình ứng dụng.
- Cài đặt và cấu hình Web Server.
- Tìm hiểu môi trường phát triển ứng dụng web với Visual

B. Nội dung

1. Tổng quan về lập trình ứng dụng Web

Ứng dụng Web là một hệ thống phức tạp, dựa trên nhiều yếu tố: phần cứng, phần mềm, giao thức, ngôn ngữ và thành phần giao diện. Trong phần này, chúng tôi sẽ giới thiệu sơ lược cho bạn về các thành phần cơ bản của ứng dụng Web: HTTP (giao thức trao đổi tài nguyên) và HTML (ngôn ngữ xây dựng trang web).

1.1. HTTP và HTML - Nền móng của Kỹ thuật lập trình web

1.1.1. HTTP (Hypertext Transfer Protocol)

Kỹ thuật cơ bản của lập trình ứng dụng web khởi đầu là HyperText Transfer Protocol (HTTP), đó là một giao thức cho phép các máy tính trao đổi thông tin với nhau qua mạng máy tính.

HTTP được xác định qua URLs (Uniform Resource Locators), với cấu trúc chuỗi có định dạng như sau:

Sau tiền tố http://, chuỗi URL sẽ chứa tên host hay địa chỉ IP của máy server (*có thể có số cổng đi kèm*), tiếp theo là đường dẫn dẫn đến tập tin server được yêu cầu. Tùy chọn sau cùng là tham số, còn được gọi là **query string** (chuỗi tham số/chuỗi truy vấn)**.**

Ví dụ: Phân tích địa chỉ http://www.comersus.com/comersus6/store/index.asp

Trang web index.asp được lưu trữ trong thư mục **/comersus6/store** tại Web Server với host là www.comersus.com

Một số thuật ngữ:

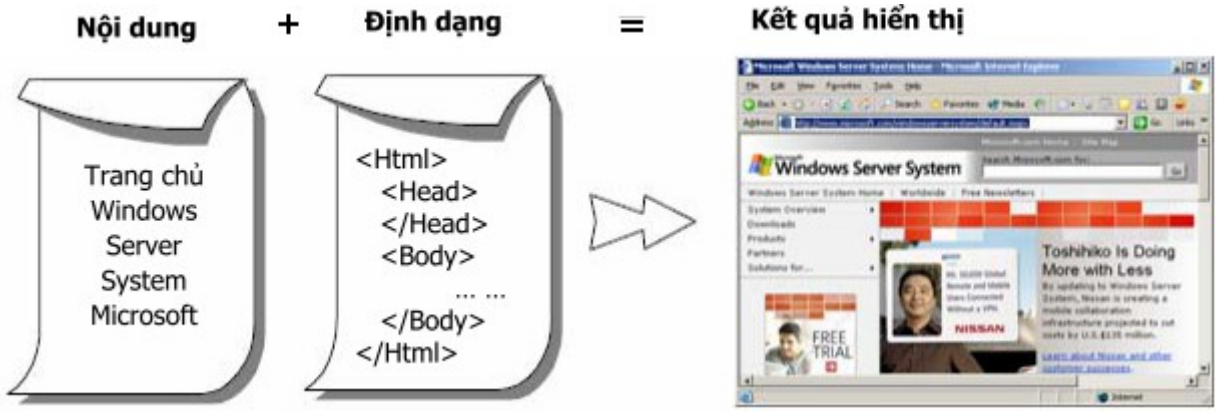
- Internet: là một hệ thống gồm nhiều máy tính ở khắp nơi trên thế giới nối lại với nhau.
- WWW: World Wide Web (mạng toàn cầu), thường được dùng khi nói về Internet
- Web Server: Máy tính lưu trữ các trang web
- Web Client: Máy tính dùng để truy cập các trang web
- Web Browser: Phần mềm dùng để truy cập web

Một số web browser phổ biến: Internet Explorer, Netscape Navigator, Avant Browser, Opera, …

1.1.2. HTML (Hypertext Markup Language)

Trang web HTML là một tập tin văn bản được viết bằng ngôn ngữ HTML, ngôn ngữ này còn được biết đến với tên gọi: ngôn ngữ đánh dấu văn bản.

Ngôn ngữ đánh dấu HTML sử dụng các ký hiệu quy định sẵn (được gọi là **tag**) để trình bày nội dung văn bản.



Ví dụ: Nội dung trang web AspDotNet.htm

1.2. Tìm hiểu các mô hình ứng dụng

1.2.1. Mô hình ứng dụng 2 lớp

Đây là một dạng mô hình đơn giản, khá phổ biến của một ứng dụng phân tán. Trong mô hình này, việc xử lý dữ liệu được thực hiện trên Database Server, việc nhận và hiển thị dữ liệu được thực hiện ở Client.

a. Ưu điểm

- Dữ liệu tập trung → đảm bảo dữ liệu được nhất quán.
- Dữ liệu được chia sẻ cho nhiều người dùng.

b. Khuyết điểm

- Các xử lý tra cứu và cập nhật dữ liệu được thực hiện ở Database Server, việc nhận kết quả và hiển thị phải được thực hiện ở Client → Khó khăn trong vấn đề bảo trì và nâng cấp.
- Khối lượng dữ liệu truyền trên mạng lớn → chiếm dụng đường truyền, thêm gánh nặng cho Database Server.

1.2.2. Mô hình ứng dụng 3 lớp

Mô hình 2 lớp phần nào đáp ứng được các yêu cầu khắc khe của một ứng dụng phân tán, tuy nhiên, khi khối lượng dữ liệu lớn, ứng dụng đòi hỏi nhiều xử lý phức tạp, số người dùng tăng, mô hình 2 lớp không thể đáp ứng

được.

Mô hình 3 lớp sử dụng thêm Application Server giữ nhiệm vụ tương tác giữa Client và Database server, giảm bớt các xử lý trên Database server, tập trung các xử lý nhận và hiển thị dữ liệu tại Application server.

a. Ưu điểm

- Hỗ trợ nhiều người dùng
- Giảm bớt xử lý cho Client → Không yêu cầu máy tính ở Client có cấu hình mạnh.
- Xử lý **nhận** và **hiển thị dữ liệu** tập trung tại Application Server → dễ quản lý, bảo trì và nâng cấp.
- Xử lý **truy cập dữ liệu** tập trung tại Database Server.

b. Khuyết điểm

- Phải sử dụng thêm một Application Server -> Tăng chi phí.

2. Giới thiệu về ASP.Net

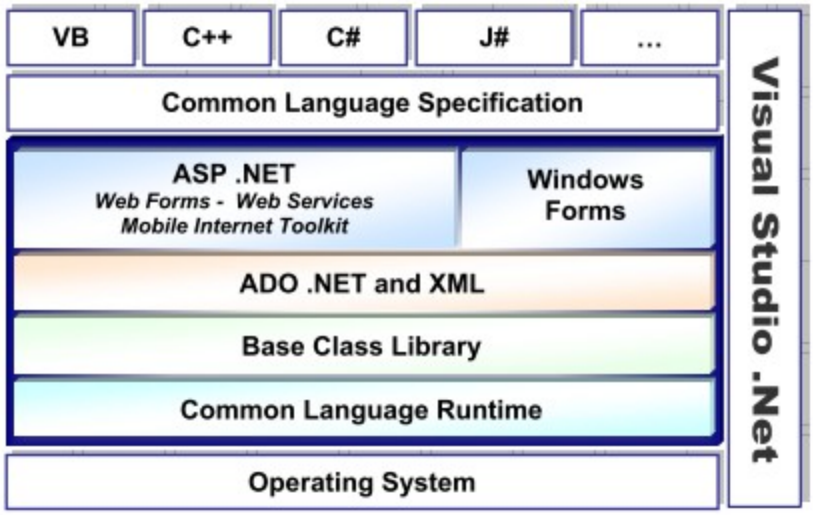
2.1. Tìm hiểu về .Net Phatform

.Net Phatform bao gồm .Net Framework và những công cụ được dùng để xây dựng, phát triển ứng dụng và dịch vụ. ASP.Net.

Những sản phẩm công nghệ .Net của Microsoft bao gồm: MSN.Net, Office.Net, Visual Studio.Net và Windows Server 2003 được biết đến với tên gọi Windows .Net Server.

Visual Studio .Net là bộ phần mềm được dùng để xây dựng và phát triển các ứng dụng bao gồm các ngôn ngữ lập trình: C++.Net, Visual Basic.Net, ASP.Net, C# và J#. Tất cả các ngôn ngữ này được xây dựng dựa trên nền .Net Framework, vì thế, bạn sẽ nhận thấy cú pháp cơ bản của những ngôn ngữ này tương tự nhau.

2.2. Tìm hiểu về .Net Framework



Kiến trúc .Net Framework

2.2.1. Hệ điều hành

Cung cấp các chức năng xây dựng ứng dụng

Với vai trò quản lý việc xây dựng và thi hành ứng dụng, .NET Framework cung cấp các lớp đối tượng (Class) để bạn có thể gọi thi hành các chức năng mà đối tượng đó cung cấp. Tuy nhiên, lời kêu gọi của bạn có được "hưởng ứng" hay không còn tùy thuộc vào khả năng của hệ điều hành đang chạy ứng dụng của bạn.

Các chức năng đơn giản như hiển thị một hộp thông báo (Messagebox) sẽ được .NET Framework sử dụng các hàm API của Windows. Chức năng phức tạp hơn như sử dụng các COMponent sẽ yêu cầu Windows phải cài đặt Microsoft Transaction Server (MTS) hay các chức năng trên Web cần Windows phải cài đặt Internet Information Server (IIS).

Như vậy, bạn cần biết rằng lựa chọn một hệ điều hành để cài đặt và sử

dụng .NET Framework cũng không kém phần quan trọng. Cài đặt .NET Framework trên các hệ điều hành Windows 2000, 2000 Server, XP, XP.NET, 2003 Server sẽ đơn giản và tiện dụng hơn trong khi lập trình.

2.2.2. Common Language Runtime

Là thành phần "kết nối" giữa các phần khác trong .NET Framework với hệ điều hành. Common Language Runtime (CLR) giữ vai trò quản lý việc thi hành các ứng dụng viết bằng .NET trên Windows. CLR sẽ thông dịch các lời gọi từ chương trình cho Windows thi hành, đảm bảo ứng dụng không chiếm dụng và sử dụng tràn lan tài nguyên của hệ thống. Nó cũng không cho phép các lệnh "nguy hiểm" được thi hành. Các chức năng này được thực thi bởi các thành phần bên trong CLR như Class loader, Just In Time compiler, Garbage collector, Exception handler, COM marshaller, Security engine,…

Trong các phiên bản hệ điều hành Windows mới như XP.NET và Windows 2003, CLR được gắn kèm với hệ điều hành. Điều này đảm bảo ứng dụng viết ra trên máy tính của chúng ta sẽ chạy trên máy tính khác mà không cần cài đặt, các bước thực hiện chỉ đơn giản là một lệnh xcopy của DOS!

2.2.3. Bộ thư viện các lớp đối tượng

Nếu phải giải nghĩa từ "Framework" trong thuật ngữ .NET Framework thì đây là lúc thích hợp nhất. Framework chính là một tập hợp hay thư viện các lớp đối tượng hỗ trợ người lập trình khi xây dựng ứng dụng. Có thể một số người trong chúng ta đã nghe qua về MFC và JFC. Microsoft Foundation Class là bộ thư viện mà lập trình viên Visual C++ sử dụng trong khi Java Foundation Class là bộ thư viện dành cho các lập trình viên Java. Giờ đây, có thể coi .NET Framework là bộ thư viện dành cho các lập trình viên .NET.

Với hơn 5000 lớp đối tượng để gọi thực hiện đủ các loại dịch vụ từ hệ điều hành, chúng ta có thể bắt đầu xây dựng ứng dụng bằng Notepad.exe!!!… Nhiều người lầm tưởng rằng các môi trường phát triển phần mềm như Visual Studio 98 hay Visual Studio.NET là tất cả những gì cần để viết chương trình. Thực ra, chúng là những phần mềm dùng làm "vỏ bọc" bên ngoài.

Với chúng, chúng ta sẽ viết được các đoạn lệnh đủ các màu xanh, đỏ; lỗi cú pháp báo ngay khi đang gõ lệnh; thuộc tính của các đối tượng được đặt ngay trên cửa sổ properties, giao diện được thiết kế theo phong cách trực quan… Như vậy, chúng ta có thể hình dung được tầm quan trọng của .NET Framework. Nếu không có cái cốt lõi .NET Framework, Visual Studio.NET cũng chỉ là cái vỏ bọc! Nhưng nếu không có Visual Studio.NET, công việc của lập trình viên .NET cũng lắm bước gian nan!

a. Base class library – thư viện các lớp cơ sở

Đây là thư viện các lớp cơ bản nhất, được dùng trong khi lập trình hay bản thân những người xây dựng .NET Framework cũng phải dùng nó để xây dựng các lớp cao hơn. Ví dụ các lớp trong thư viện này là String, Integer, Exception,…

b. ADO.NET và XML

Bộ thư viện này gồm các lớp dùng để xử lý dữ liệu. ADO.NET thay thế ADO để trong việc thao tác với các dữ liệu thông thường. Các lớp đối tượng XML được cung cấp để bạn xử lý các dữ liệu theo định dạng mới: XML. Các ví dụ cho bộ thư viện này là SqlDataAdapter, SqlCommand, DataSet, XMLReader, XMLWriter,…

c. ASP.NET

Bộ thư viện các lớp đối tượng dùng trong việc xây dựng các ứng dụng Web. ASP.NET không phải là phiên bản mới của ASP 3.0. Ứng dụng web xây dựng bằng ASP.NET tận dụng được toàn bộ khả năng của .NET Framework. Bên cạnh đó là một "phong cách" lập trình mới mà Microsoft đặt cho nó một tên gọi rất kêu: code behind. Đây là cách mà lập trình viên xây dựng các ứng dụng Windows based thường sử dụng – giao diện và lệnh được tách riêng. Tuy nhiên, nếu bạn đã từng quen với việc lập trình ứng dụng web, đây đúng là một sự "đổi đời" vì bạn đã được giải phóng khỏi mớ lệnh HTML lộn xộn tới hoa cả mắt.

Sự xuất hiện của ASP.NET làm cân xứng giữa quá trình xây dựng ứng dụng trên Windows và Web. ASP.NET cung cấp một bộ các Server Control để lập trình viên bắt sự kiện và xử lý dữ liệu của ứng dụng như đang làm việc với ứng dụng Windows. Nó cũng cho phép chúng ta chuyển một ứng dụng trước đây viết chỉ để chạy trên Windows thành một ứng dụng Web khá dễ dàng. Ví dụ cho các lớp trong thư viện này là WebControl, HTMLControl, …

d. Web services

Web services có thể hiểu khá sát nghĩa là các dịch vụ được cung cấp qua Web (hay Internet). Dịch vụ được coi là Web service không nhằm vào người dùng mà nhằm vào người xây dựng phần mềm. Web service có thể dùng để cung cấp các dữ liệu hay một chức năng tính toán.

Ví dụ, công ty du lịch của bạn đang sử dụng một hệ thống phần mềm để ghi nhận thông tin về khách du lịch đăng ký đi các tour. Để thực hiện việc đặt phòng khách sạn tại địa điểm du lịch, công ty cần biết thông tin về phòng trống tại các khách sạn. Khách sạn có thể cung cấp một Web service để cho biết thông tin về các phòng trống tại một thời điểm. Dựa vào đó, phần mềm của bạn sẽ biết rằng liệu có đủ chỗ để đặt phòng cho khách du lịch không? Nếu đủ, phần mềm lại có thể dùng một Web service khác cung cấp chức năng đặt phòng để thuê khách sạn. Điểm lợi của Web service ở đây là bạn không cần một người làm việc liên lạc với khách sạn để hỏi thông tin phòng, sau đó, với đủ các thông tin về nhiều loại phòng người đó sẽ xác định loại phòng nào cần đặt, số lượng đặt bao nhiêu, đủ hay không đủ rồi lại liên lạc lại với khách sạn để đặt phòng. Đừng quên là khách sạn lúc này cũng cần có người để làm việc với nhân viên của bạn và chưa chắc họ có thể liên lạc thành công.

Web service được cung cấp dựa vào ASP.NET và sự hỗ trợ từ phía hệ điều hành của Internet Information Server.

e. Window form

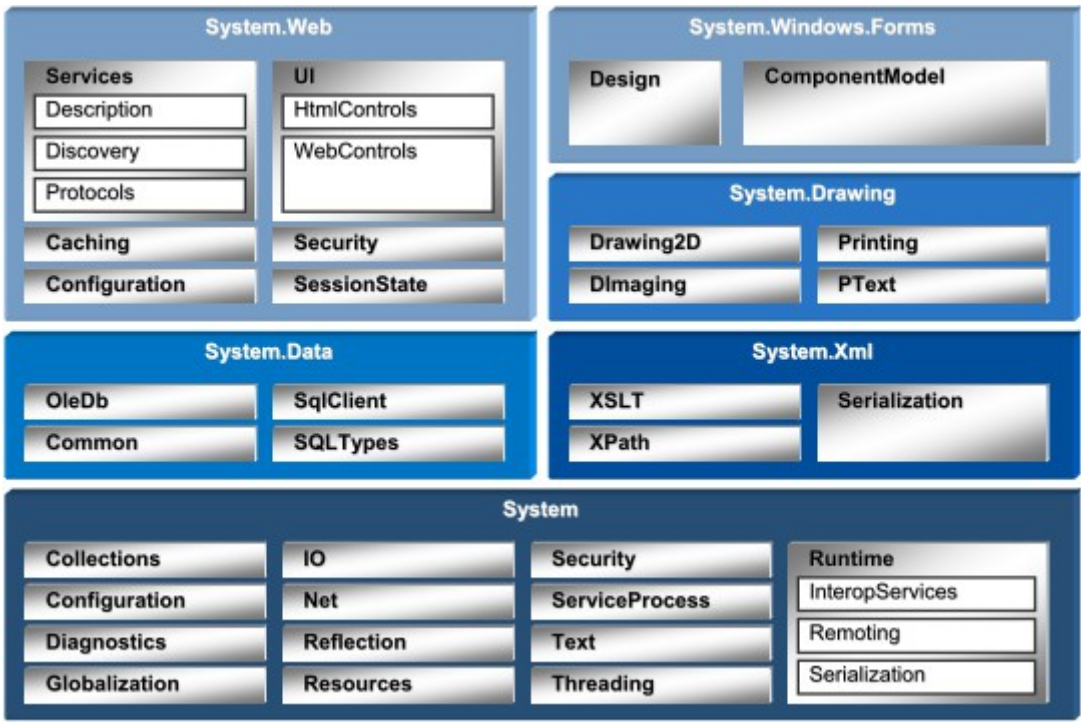
Bộ thư viện về Window form gồm các lớp đối tượng dành cho việc xây dựng các ứng dụng Windows based. Việc xây dựng ứng dụng loại này vẫn

được hỗ trợ tốt từ trước tới nay bởi các công cụ và ngôn ngữ lập trình của Microsoft. Giờ đây, ứng dụng chỉ chạy trên Windows sẽ có thể làm việc với ứng dụng Web dựa vào Web service. Ví dụ về các lớp trong thư viện này là: Form, UserControl,…

2.2.4. Phân nhóm các lớp đối tượng theo loại

Một khái niệm không được thể hiện trong hình vẽ trên nhưng cần đề cập đến là Namespace. Đây là tên gọi một nhóm các lớp đối tượng phục vụ cho một mục đích nào đó. Chẳng hạn, các lớp đối tượng xử lý dữ liệu sẽ đặt trong một namespace tên là Data. Các lớp đối tượng dành cho việc vẽ hay hiển thị chữ đặt trong namespace tên là Drawing.

Một namespace có thể là con của một namespace lớn hơn. Namespace lớn nhất trong .NET Framework là System.



Hệ thống tên miền (Namespace)

Lợi điểm của namespace là phân nhóm các lớp đối tượng, giúp người dùng dễ nhận biết và sử dụng. Ngoài ra, namespace tránh việc các lớp đối tượng có tên trùng với nhau không sử dụng được. .NET Framework cho phép chúng ta tạo ra các lớp đối tượng và các namespace của riêng mình. Với hơn

5000 tên có sẵn, việc đặt trùng tên lớp của mình với một lớp đối tượng đã có là điều khó tránh khỏi. Namespace cho phép việc này xảy ra bằng cách sử dụng một tên đầy đủ để nói đến một lớp đối tượng. Ví dụ, nếu muốn dùng lớp WebControls, chúng ta có thể dùng tên tắt của nó là WebControls hay tên đầy đủ là System.Web.UI.WebControls.

Đặc điểm của bộ thư viện các đối tượng .NET Framework là sự trải rộng để hỗ trợ tất cả các ngôn ngữ lập trình .NET như chúng ta thấy ở hình vẽ trên. Điều này sẽ giúp những người mới bắt đầu ít bận tâm hơn trong việc lựa chọn ngôn ngữ lập trình cho mình vì tất cả các ngôn ngữ đều mạnh ngang nhau. Cũng bằng cách sử dụng các lớp đối tượng để xây dựng ứng dụng, .NET Framework buộc người lập trình phải sử dụng kỹ thuật lập trình hướng đối tượng (sẽ được nói tới trong các chương sau).

2.3. Tìm hiểu về ASP.Net

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.

Tuy nhiên, ASP vẫn còn tồn đọng một số khó khăn như Code ASP và HTML lẫn lộn, điều này làm cho quá trình viết code khó khăn, thể hiện và trình bày code không trong sáng, hạn chế khả năng sử dụng lại code. Bên cạnh đó, khi triển khai cài đặt, do không được biên dịch trước nên dễ bị mất source code. Thêm vào đó, ASP không có hỗ trợ cache, không được biên dịch trước nên phần nào hạn chế về mặt tốc độ thực hiện. 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.

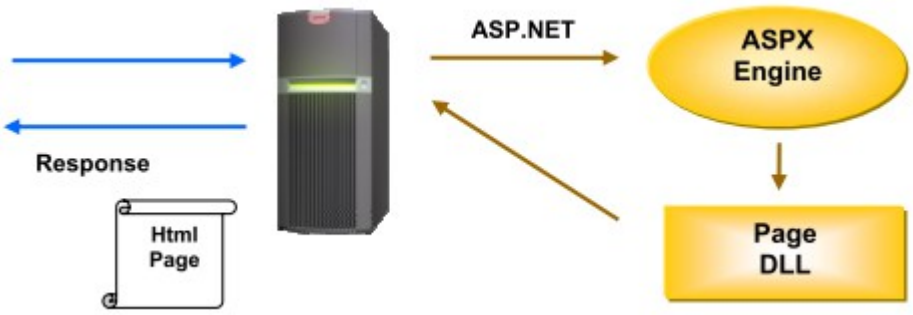
ASP.Net là kỹ thuật lập trình và phát triển ứng dụng web ở phía Server (Server-side) dựa trên nền tảng của Microsoft .Net Framework.

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 gởi 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 (ví dụ: mã lệnh trong trang ASP) 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.

2.4. Những ưu điểm của ASP.Net

- ASP.Net cho phép bạn lựa chọn một trong các ngôn ngữ lập trình mà bạn yêu thích: Visual Basic.Net, J#, C#,…
- 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 là một bước nhảy vọt đáng kể so với kỹ thuật thông dịch của ASP.



ASP.NET

- 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.
- 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.
- 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

2.5. 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:

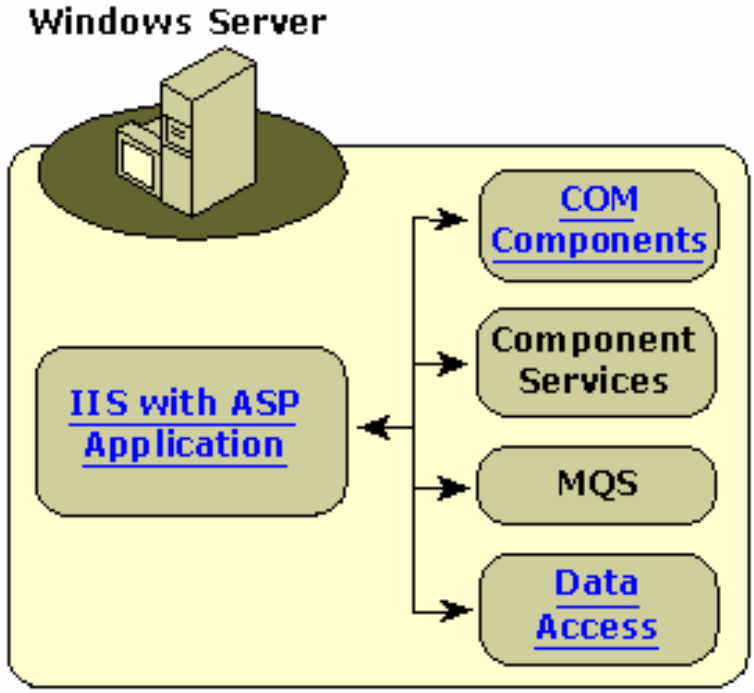
Quá trình xử lý tập tin ASPX

3. Web Server

Trong phần này chúng tôi giới thiệu cho bạn về IIS (phần mềm Web Server của Microsot dành cho Windows), đồng thời hướng dẫn bạn cài đặt, cấu hình và kiểm tra Web Server trên các hệ thống sử dụng Windows XP, Windows 2003, Windows Server 2008...

3.1. Internet Information Services

IIS có thể được sử dụng như một Web server, kết hợp với ASP để xây dựng các ứng dụng Web tận dụng các điểm mạnh của Server-side Script, COM component, … theo mô hình Client/Server.



IIS có rất nhiều phiên bản, đầu tiên được phát hành rời trong bản Service pack của WinNT.

- Các phiên bản Windows 2000 đã có tích hợp IIS 5.0.
- Windows XP tích hợp IIS 5.5
- Windows XP .NET Server tích hợp IIS 6 hỗ trợ các tính năng dành cho .NET của ASP.NET và Web Service.
- …

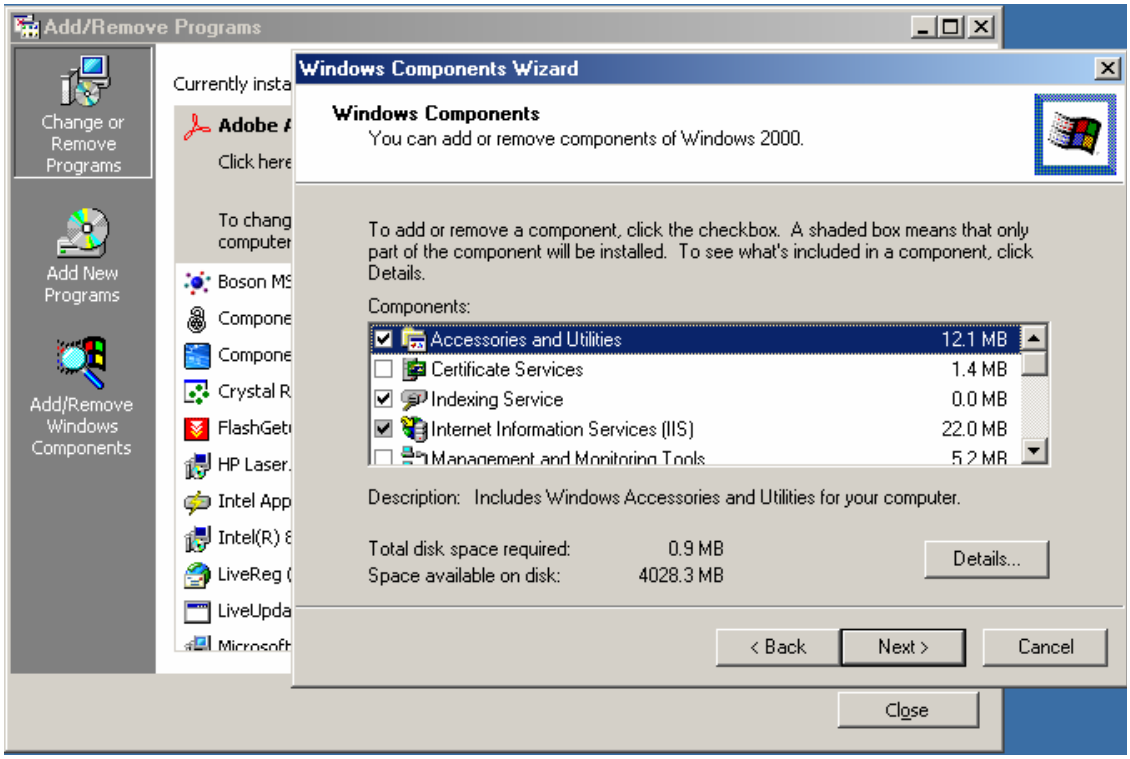
3.2. Cài đặt Web Server

3.2.1. Cài đặt Web Server trên Windows 2000/Windows XP Professional

Windows 2000/XP tích hợp sẵn IIS nhưng không tự động cài đặt do đó, bạn phải tự cài IIS nếu hệ thống đã được cài rồi.

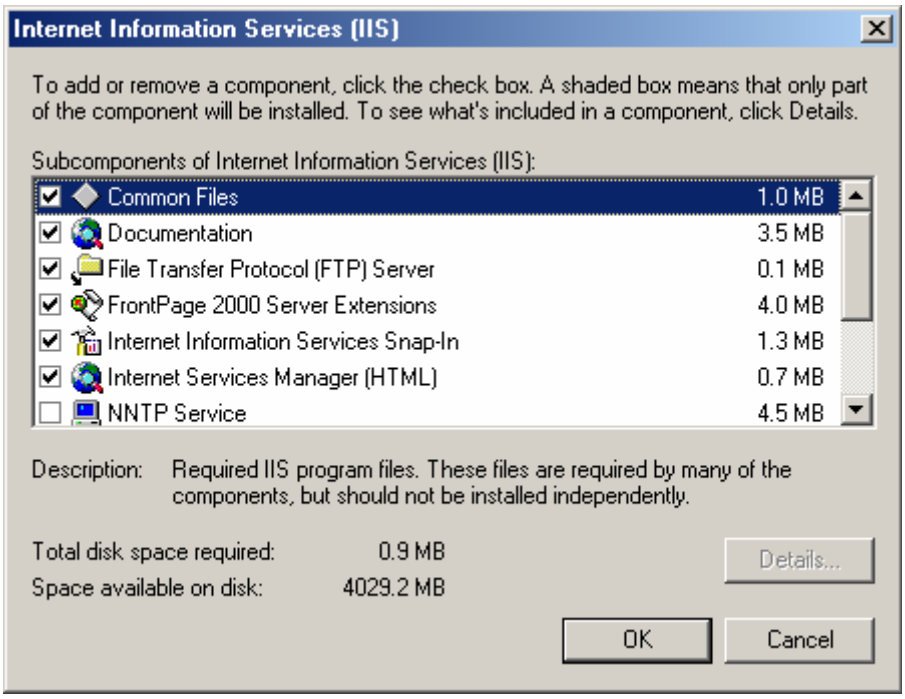
Bước 1. Chọn Control Panel | Add/Remove programs.

Bước 2. Add/Remove Windows Components.



Bước 3. Đánh dấu vào mục Internet Information Services (IIS).

Bước 4. Chọn nút Details để chọn các mục chi tiết.



Bước 5. Chọn các mục cần cài đặt trong đó bạn nhớ chọn:

- FrontPage 2000 Server Extensions
- Internet Information Services Snap-In
- Internet Services Manager (HTML)

Bước 6. Click OK để hệ thống tự cài đặt.

3.2.2. Cài đặt Web Server trên Windows Server 2003

Cài đặt Web Server trên Windows Server 2003 cũng tương tự như cài đặt Web Server trên Windows 2000.

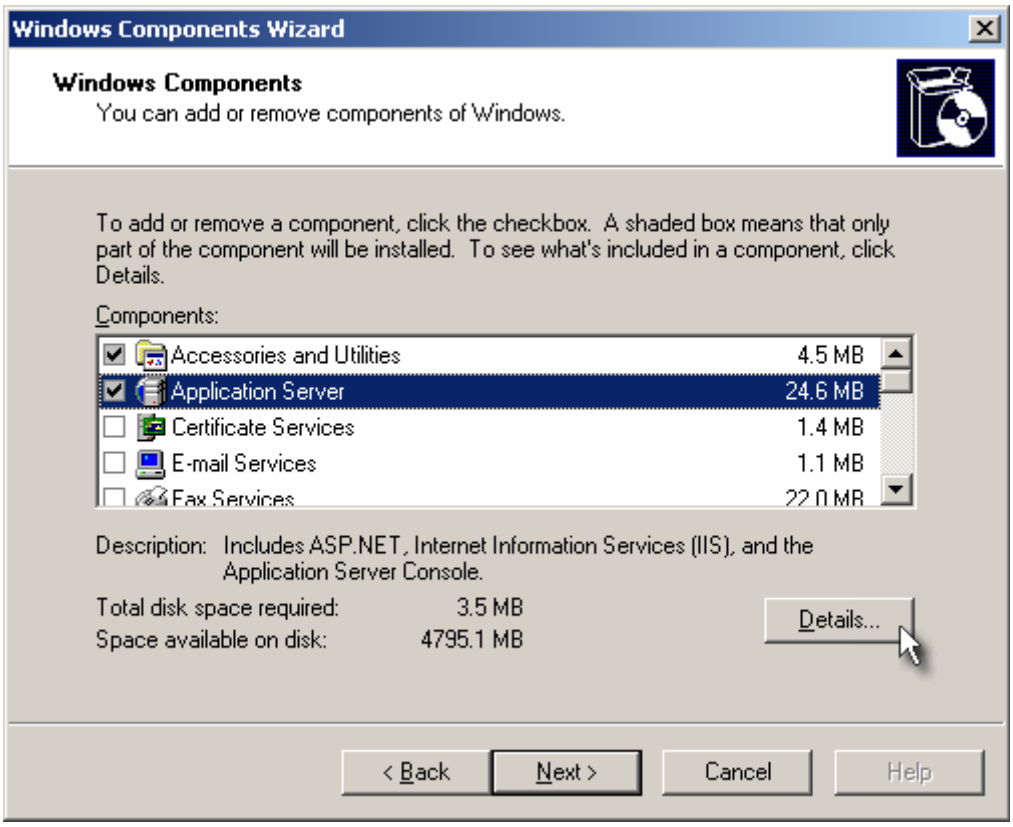
Bước 1. Chọn Control Panel | Add/Remove programs.

Bước 2. Add/Remove Windows Components.

Bước 3. Đánh dấu vào Application Server.

Bước 4. Chọn nút Details để chọn các mục chi tiết.

Các bước còn lại thực hiện như trên Windows 2000 (từ bước 3 đến bước 6)



Cài đặt IIS trên Windows Server 2003

3.2.3. Kiểm tra kết quả cài đặt Web Server

Sau khi đã cài đặt IIS, bạn có thể kiểm tra xem Web Server đã làm việc hay chưa. Mở web browser (Internet Explorer) và gõ: localhost trong phần địa chỉ. Một khi Web server đã được cài đặt, một trang web mẫu sẽ được hiển thị.

- Localhost là địa chỉ của máy cục bộ mà bạn đang làm việc. Nếu máy của bạn đang kết nối vào mạng LAN và có một địa chỉ IP, bạn có thể dùng địa chỉ này thay cho localhost.

Để xác định địa chỉ IP của máy mình:

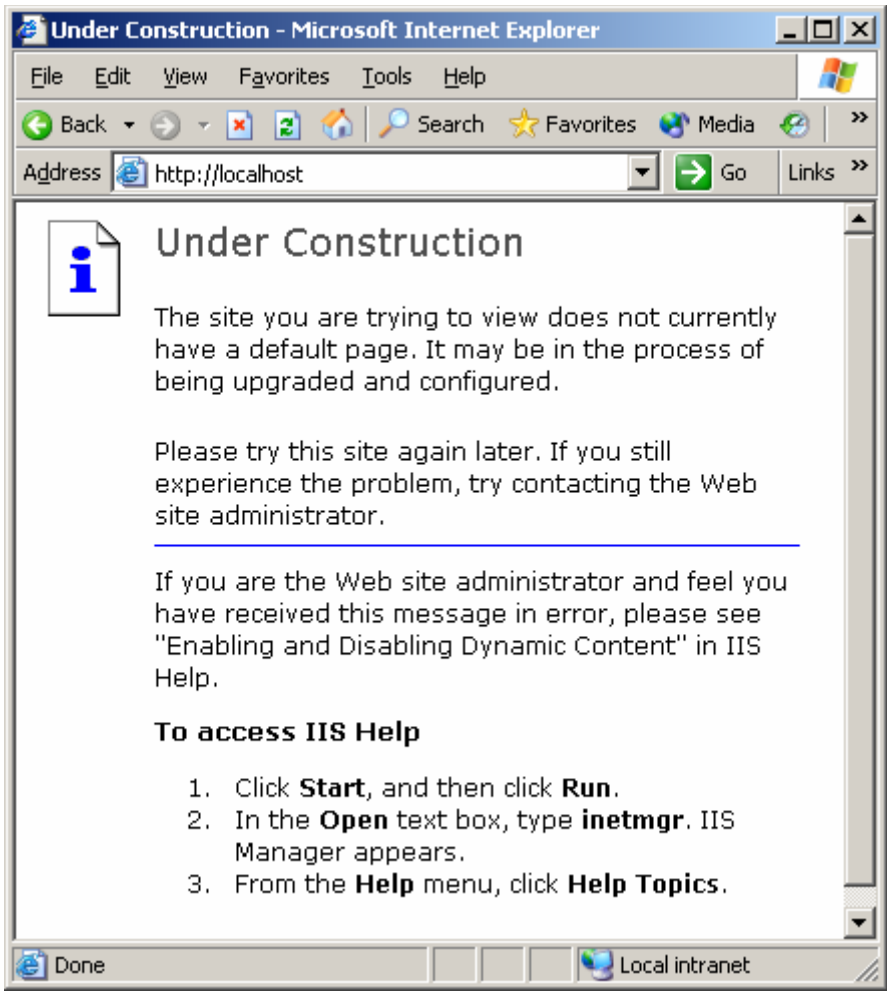
Vào menu Start|Run và gõ lệnh: command hoặc cmd

Trên màn hình DOS, gõ lệnh: ipconfig và xem phần IP Address

- Khi gõ localhost, bạn sẽ thấy trong thanh địa chỉ tự động đổi thành: http://localhost. HTTP là giao thức mặc định được dùng trên Internet. Vì HTTP là một giao thức thuộc bộ TCP/IP, bạn cần có địa chỉ IP để các máy tính khác trong mạng có thể truy cập được đến trang web của bạn.

- Sau khi cài đặt Web Server, mặc định trên ổ đĩa C:\ sẽ có sẵn thư mục C:\inetpub\wwwroot. Đây là thư mục mà Web Server mặc định ánh xạ vào localhost, do đó, các trang web đặt trong wwwroot có thể được truy cập bởi các máy tính khác.

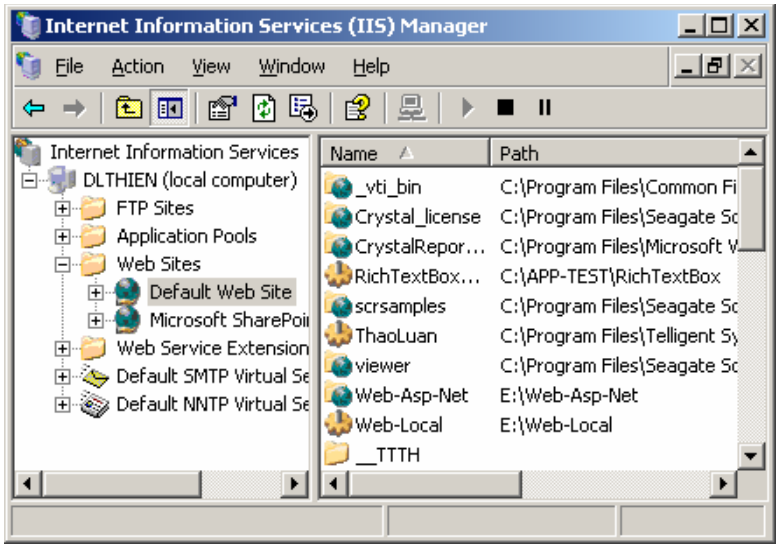
Để kiểm tra, hãy tạo một trang web và đặt vào c:\inetpub\wwwroot. Trên IE, gõ địa chỉ: localhost/<tên file html>



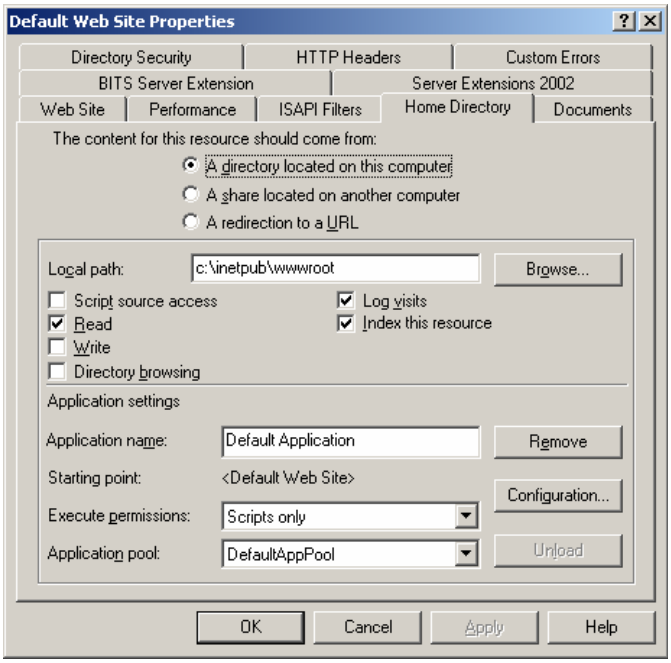
Kiểm tra thành công Web Server trên Windows Server 2003

3.3. Cấu hình Internet Information Services

Để cấu hình IIS, vào Control Panel| Administrative Tools|Internet Services Manager. Trên các hệ điều hành Windows 2000/XP, Microsoft sử dụng công cụ Microsoft Management Console (MMC) để làm công cụ quản lý, do đó tất cả các thao tác đều sử dụng menu ngữ cảnh bằng cách nhắp chuột phải trên mục muốn chọn. Chọn Properties của mục Default Web Site, bạn có thể xem và cấu hình lại các thông tin dành cho trang web mặc định của mình.



- Trên tab Home Directory, bạn có thể thay đổi đường dẫn đến một thư mục khác trên ổ cứng nếu muốn



- Trên tab Documents, bạn có thể đặt trang web mặc định sẽ hiển thị khi

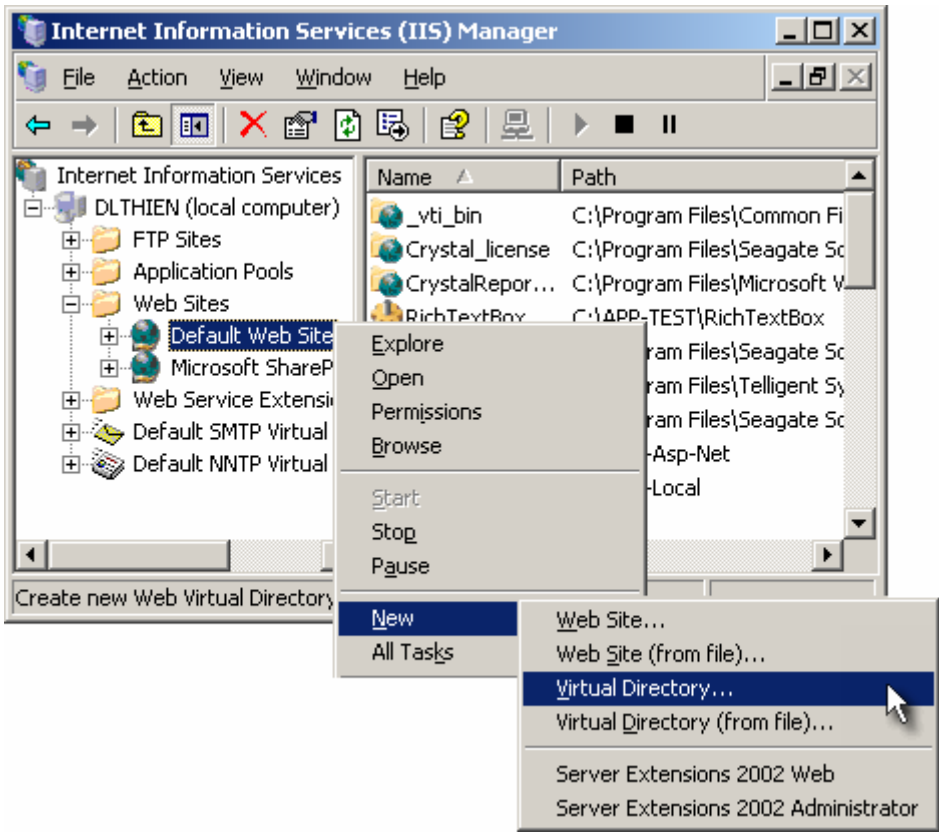
Web Browser không chỉ định trang web cụ thể. Bạn sẽ thấy index.htm và default.htm được liệt kê trong phần này. Đây là lý do tại sao khi bạn gõ localhost thì Web browser lại hiển thị được trang hompage. Thực ra, localhost tương đương với localhost/index.htm hay localhost/default.htm.

- Trong tab Directory Security, bạn có thể định lại các chế độ kiểm tra người dùng truy cập vào web site.

3.4. Tạo các ứng dụng web trên IIS

Một Web Server có thể quản lý nhiều ứng dụng Web đồng thời. Thông thường, bạn sẽ tổ chức một thư mục con trong wwwroot cho mỗi ứng dụng nhưng bạn cũng có thể tạo ánh xạ từ một thư mục khác.

- Nếu bạn đặt thư mục trong wwwroot, IIS sẽ tự động liệt kê nó trong mục Default Web Site.
- Nếu muốn tạo một thư mục nằm ngoài thư mục wwwroot thành một web site, chúng ta cần tạo Virtual Directory liên kết đến thư mục đó.
- Để tạo một virtual directory:
- Chọn mục New | Virtual Directory trên menu ngữ cảnh.



Tạo Virtual Directory

- Nhập vào tên alias cho thư mục ảo.
- Chọn nút Browse để chọn thư mục muốn ánh xạ. Tiếp đó, bạn sẽ phải đặt một số cấu hình khởi đầu cho web site, những thông tin này có thể cấu hình lại tương tự như với Default Web Site ở trên

4. Tạo ứng dụng Web đầu tiên

4.1. Khởi động MS Visual Studio .Net

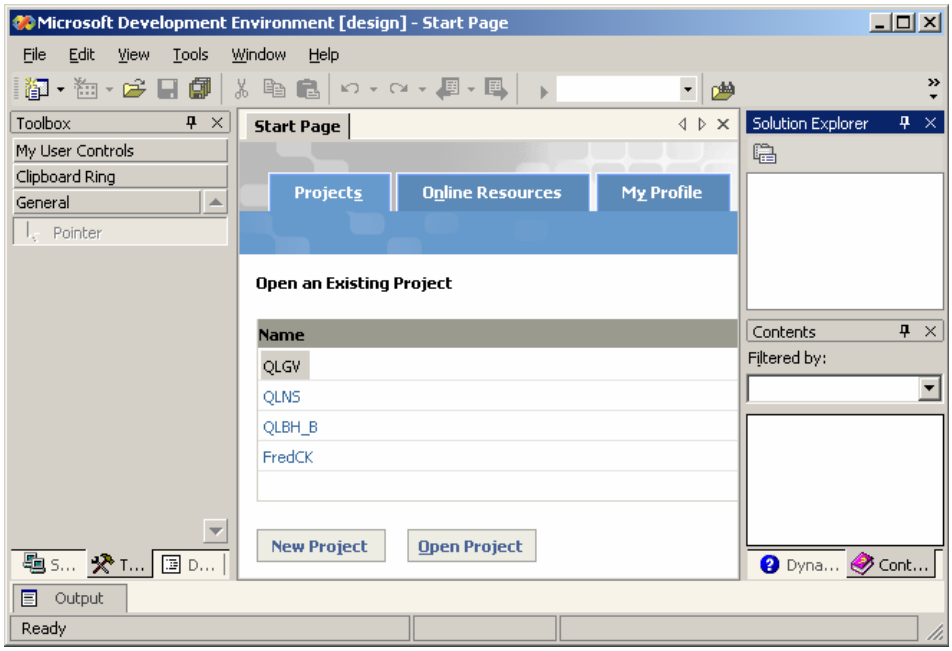
Chúng ta sẽ bắt đầu bằng việc làm quen với môi trường phát triển ứng dụng (IDE) của Visual Studio.NET. VS.NET có nhiều thay đổi so với VS 98.

Hình dưới là màn hình khởi đầu của VS.NET 2003. Vùng làm việc chính giữa đang hiển thị trang "Start page" với 3 mục chính: Projects, Online Resource và My Profile.

My Profile ghi nhớ thông tin về người sử dụng VS.NET. Các thông tin chủ yếu liên quan đến cách chúng ta sẽ sử dụng VS.NET như thế nào. Chẳng hạn như cách hiển thị các cửa sổ, các phím tắt, cách VS.NET hiển thị màn hình giúp đỡ,…

Online Resource cần một kết nối với Internet để download các thông tin từ website của Microsoft về máy tính của chúng ta.

Projects liệt kê các project mà chúng ta đã làm việc trong thời gian gần đây. Trên mục này, chúng ta cũng có thể tạo mới một project bằng cách nhấn vào nút New Project.



Màn hình Microsoft Visual Studio .Net

4.2. Tạo mới ứng dụng Web

4.2.1. Tạo ứng dụng web đầu tiên

Chúng ta có thể tạo ứng dụng Asp.Net sử dụng Visual C# Project theo các bước sau:

- Chọn từ thực đơn File | New | Project. Xuất hiện hộp thoại tạo mới Project
- Chọn loại Project là **Visual C# Project** từ Project Types
- Chọn **ASP.Net Web Application** từ vùng Template
- Ứng dụng mới được tạo mặc định có tên là **WebApplication**XX (XX là số thứ tự tự động). Chúng ta có thể thay đổi tên của Project tại điều khiển Location. Trong ví dụ này, chúng tôi thay đổi tên Project **WebApplication1** thành **MinhHoa.**

Project được tạo mặc định lưu tại thư mục: C:\Inetpub\wwwroot

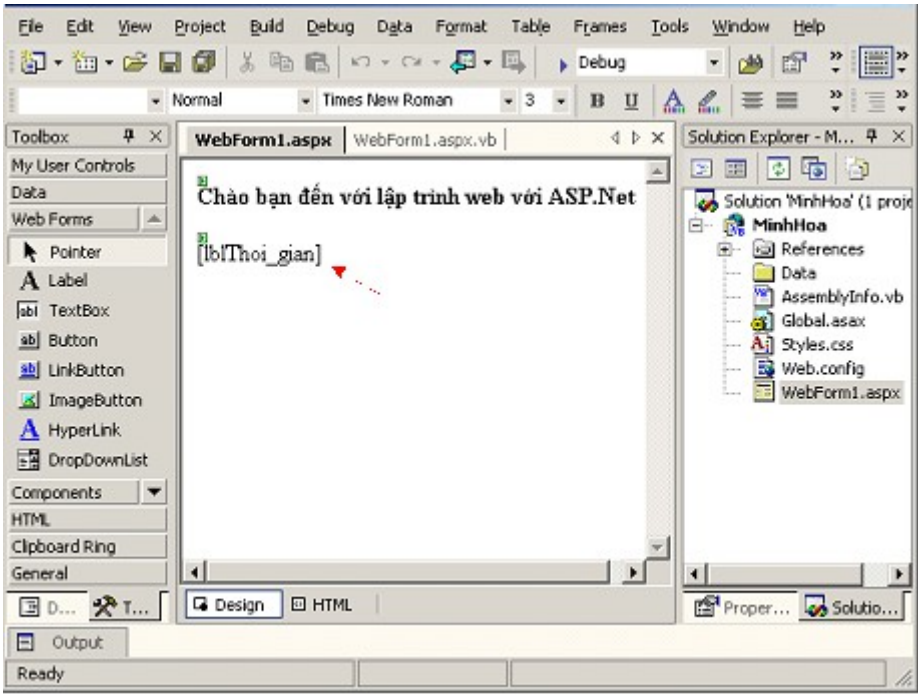
4.2.2. Bổ sung điều khiển và thi hành ứng dụng

Thiết lập thuộc tính pageLayout của trang mặc định (WebForm1.aspx) là FlowLayout (thực hiện thông qua cửa sổ thuộc tính)

Thêm 2 điều khiển Label có trên trang WebForms của thanh công cụ Toolbox.

Tên điều khiển	Thuộc tính Text
lblChao	Chào bạn đến với lập trình web với
lblThoi_gian	[Chuỗi rỗng]

Viết lệnh cho sự kiện Page_Load:

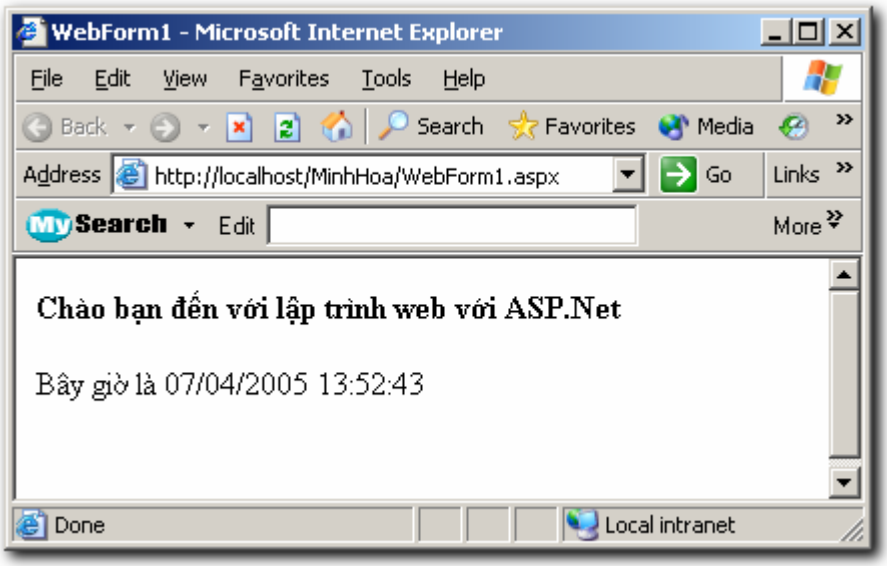


Màn hình ứng dụng Web: MinhHoa

Lưu ý: Phải lưu tập tin với tùy chọn **Save with Encoding…** nếu như trong cửa sổ lệnh hoặc màn hình thiết kế có sử dụng Font Unicode.

Lưu ứng dụng với Font chữ Unicode

Nhấn F5 để thi hành ứng dụng.



Kết quả hiển thị của trang Web

4.3. Phân loại tập tin trong ASP.Net

ASP.Net	ASP	Diễn giải
.asax	.asa	Tập tin global.asax trong ASP .Net thay thế cho tập tin
.ascx		Các điều khiển do người dùng tự tạo được lưu trữ với
.asmx		Tập tin Web Service của ứng dụng ASP.Net
.aspx	.asp	Phần mở rộng mặc định của trang ASP.Net
.config		Tập tin cấu hình ứng dụng theo định dạng XML.
.cs		Tập tin mã nguồn viết theo ngôn ngữ C#
.js	.js	Tập tin mã nguồn của Jscript
.vb		Tập tin mã nguồn viết theo ngôn ngữ VB.Net

4.4. Làm quen với các thành phần giao diện trên VS .Net

4.4.1. Solution Explorer

Hiển thị cửa số Solution Explorer: Thực đơn View | Solution Explorer

Thao tác với cửa sổ Solution Explorer

Đây là cửa số quản lý các "tài nguyên" có trong ứng dụng. Thông qua cửa sổ này, chúng ta có thể:

Thực hiện các chức năng: sao chép, cắt, dán trên tập tin, thư mục như

Windows Explorer.

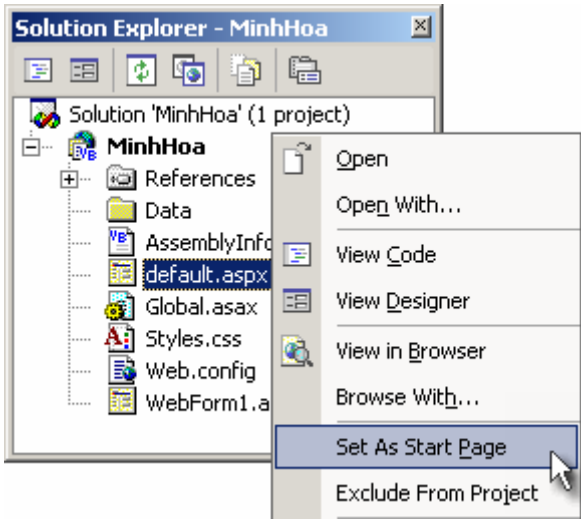
Tổ chức thư mục quản lý ứng dụng: Sử dụng chức năng Add | New Folder từ thực đơn ngữ cảnh.

Thêm thành phần mới cho ứng dụng: Sử dụng chức năng Add | Add New Item…từ thực đơn ngữ cảnh. Xuất hiện hộp thoại Add New Item.

- Web Form: Thêm trang Web
- Class: Thêm lớp đối tượng
- Module: Thêm thư viện
- Web User Control: Thêm điều khiển người dùng
- …

Xác định trang web khởi động cho ứng dụng

Chọn trang cần khởi động / Nhấp chuột phải (xuất hiện thực đơn ngữ cảnh) / Chọn Set As Start Page.



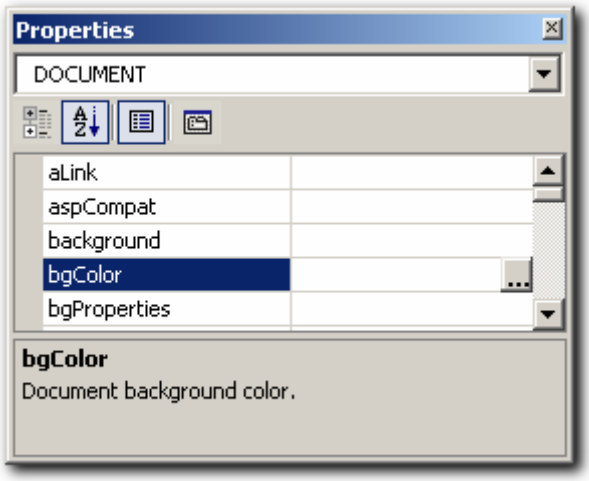
Xác định trang khởi động cho ứng dụng

- Xác định Project khởi động (trong trường hợp Solution có nhiều Project): Chọn Set as StartUp Project từ thực đơn ngữ cảnh.

4.4.2. Property Window

Hiển thị cửa số Properties Window: Thực đơn View | Properties Window.

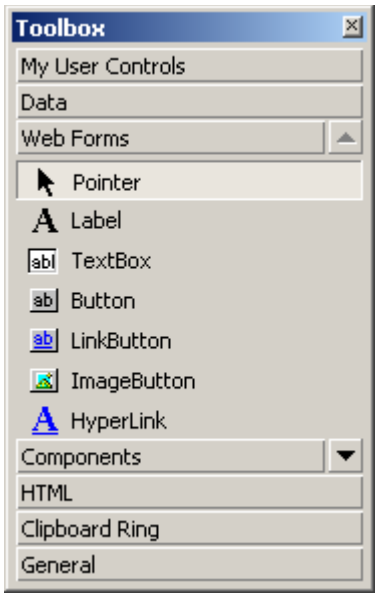
Thông qua cửa sổ thuộc tính, chúng ta có thể thiết lập thuộc tính cho trang web và các đối tượng có trong trang web.

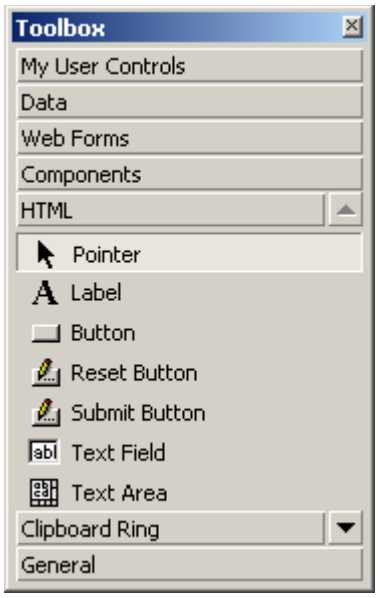


Cửa sổ thuộc tính

4.4.3. Toolbox

Hiển thị Toolbox: Thực đơn View | Toolbox



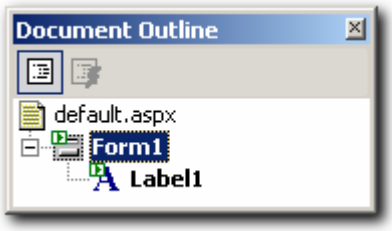


Web Control Html Control

4.4.4. Document Outline Window

Hiển thị cửa sổ Document Outline: Thực đơn View | Other Windows | Document Outline.

Cửa sổ này hiển thị các thành phần của trang web theo tổ chức cây € rất dễ quản lý và thao tác với các đối tượng có trong trang Web.



Cửa sổ Document Outline

BÀI 2. WEB SERVER CONTROL

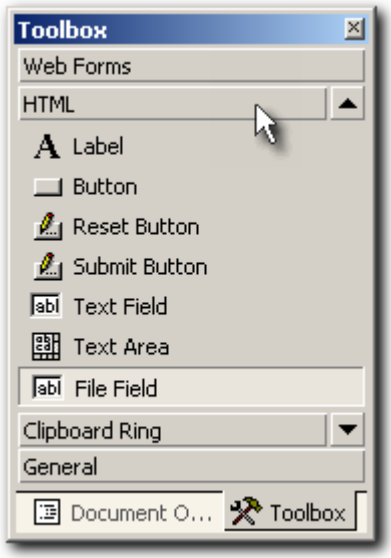
A. Mục tiêu

- Sử dụng thành thạo các điều khiển HTML & ASP.Net Web Control
- Làm việc với đối tượng ViewState.

B. Nội dung

1. HTML Control

Điều khiển HTML (tag HTML) trong trang ASP.Net có thể xem như những chuỗi văn bản bình thường. Để có thể được sử dụng lập trình ở phía Server, ta gán thuộc tính **runat="Server"** cho các điều khiển HTML đó. Những điều khiển HTML (tag HTML) có thuộc tính runat="Server" được gọi là HTML Server Control.

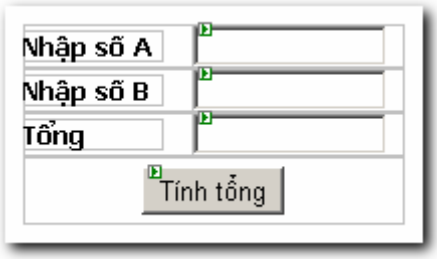


Các điều khiển HTML trên thanh công cụ

Để chuyển các điều khiển HTML thành điều khiển HTML Server, ta chọn Run As Server Control từ thực đơn ngữ cảnh.

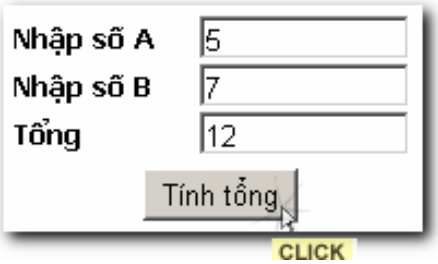
Chuyển điều khiển HTML thành điều khiển HTML Server

Ví dụ: Các điều khiển HTML: Label, Textbox, Button



Xử lý sự kiện:

```
private void butTong_ServerClick()
{
        txtTong.Value = Val(txtA.Value) + Val(txtB.Value);
}
```

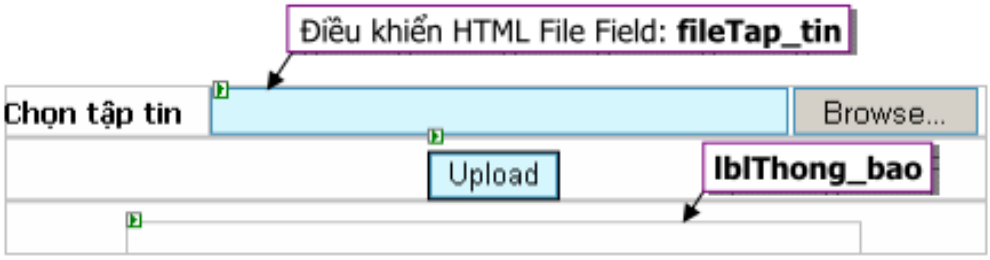


Khi thi hành ứng dụng

Ví dụ: Upload file với điều khiển HTML File Field

Trong ví dụ sau, chúng ta sẽ thực hiện Upload tập tin lên server, cụ thể hơn, tập tin vừa Upload sẽ được lưu trong thư mục **Upload**.

Chú ý: Để chép được tập tin lên thư mục Upload, bạn cần phải cấp quyền cho phép ghi trên thư mục Upload



Màn hình ở chế độ thiết kế

Xử lý sự kiện:

```
private void butUpload_ServerClick()
```

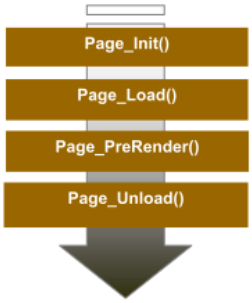
```
{
    string sTap_tin;
    string sTen_file;
    sTap_tin = fileTap_tin.PostedFile.FileName;
    //Phân tích đường dẫn tập tin để lấy tên tập tin
    sTen_file = sTap_tin.Substring(sTap_tin.LastIndexOf("\\") + 1,
    sTap_tin.Length - sTap_tin.LastIndexOf("\\") + 1);
    //Thực hiện chép tập tin lên thư mục Upload
    fileTap_tin.PostedFile.SaveAs(Server.MapPath("Upload\\") + sTen_file);
    lblThong_bao.InnerHtml = "<B>Thông báo: Bạn đã upload file thành
    công</B>";
}
```

2. ASP.Net Web Control

2.1. Asp.Net Page

Đây là thành phần chính của giao diện, là nơi chứa các điều khiển, được sử dụng để thể hiện nội dung trang web đến người dùng.

2.1.1. Sự kiện



Chuỗi sự kiện của đối tượng Page

a. Init

Sự kiện Page_Init xảy ra đầu tiên khi trang web được yêu cầu.

```
private void  Page_Init()
{
    //Do not modify it using the code editor.
```

```
        InitializeComponent();
    }
```

b. Load

Sự kiện này là nơi mà bạn sẽ đặt phần lớn các xử lý, giá trị khởi động ban đầu cho trang web. Sự kiện này luôn xảy ra mỗi khi trang web được yêu cầu.

```
    private void Page_Load()
    {
    //Put user code to initialize the page here
    }
```

c. PreRender

Sự kiện này xảy ra khi trang Web chuẩn bị được trả về cho Client.

```
    private void Page_PreRender(object sender, System.Object e)
    {
    }
```

d. Unload

Sự kiện này đối lập với sự kiện Page_Init. Nếu như sự kiện Page_Init xảy ra đầu tiên khi trang Web được yêu cầu, thì đây, Page_Unload là xự kiện sau cùng, xảy ra sau tất cả những sự kiện khác.

```
    private void Page_Unload(object sender, System.Object e)
    {
    }
```

2.1.2. Thuộc tính

a. IsPostBack

Đây là một thuộc tính kiểu luận lý. Giá trị của thuộc tính này cho biết trạng thái của trang Web khi được Load, nếu là lần Load đầu tiên, giá trị của thuộc tính này = False. Thuộc tính này thường được sử dụng trong sự kiện Page_Load để kiểm tra trạng thái của trang Web.

```
private void Page_Load()
{
        //Put user code to initialize the page here
        if (!IsPostBack) {
        lblPostBack.Text = "Đây là lần yêu cầu đầu tiên";
        } else {
        lblPostBack.Text = "Đây là lần yêu cầu sau.";
        }
}
```

b. SmartNavigation

Trong trường hợp nội dung của trang Web vượt quá kích thước hiển thị của màn hình và bạn đang đọc ở phần giữa của trang Web, khi được ReLoad lại, 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 Reload. Đây là một thuộc tính kiểu luận lý. Giá trị mặc định là False.

2.2. Điều khiển cơ bản

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.

Bảng liệt kê các thuộc tính chung của các Web control

Thuộc tính	Kiểu	Ý nghĩa
(ID)	Chuỗi	Qui định tên của điều khiển. Tên của điều
AccessKey	String	Qui định ký tự để di chuyển nhanh đến
Attributes	AttributeCollection	Tập hợp các thuộc tính của điều khiển
BackColor	Color	Qui định màu nền của điều khiển.
BorderColor	Color	Qui định màu đường viền của điều khiển.

BorderStyle	BorderStyle	Qui định kiểu đường viền của điều khiển.
BorderWidth	Unit	Qui định độ rộng của đường viền.
CssClass	String	Qui định hình thức hiển thị của điều khiển
Enabled	Boolean	Qui định điều khiển có được hiển thị hay
Font	FontInfo	Qui định Font hiển thị cho điều khiển.
ForeColor	Color	Qui định màu chữ hiển thị trên điều khiển
Height	Unit	Qui định chiều cao của điều khiển.
ToolTip	String	Dòng chữ sẽ hiển thị khi rê chuột vào điều
Width	Unit	Qui định độ rộng của điều khiển.

2.2.1. 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:

2.2.2. HyperLink

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

Các thuộc tính

- 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 mối liên kết
- _blank: Hiển thị trang liên kết ở một cửa sổ mới.
- _self: Hiển thị trang liên kết tại chính cửa sổ chứa liên kết đó.
- _parent: Hiển thị trang liên kết ở frame cha.

Ví dụ:

Kết quả hiển trị trên trang Web

2.2.3. 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

Text: Nội dung chứa trong Textbox

TextMode: Qui định chức năng của Textbox, có các giá trị sau:

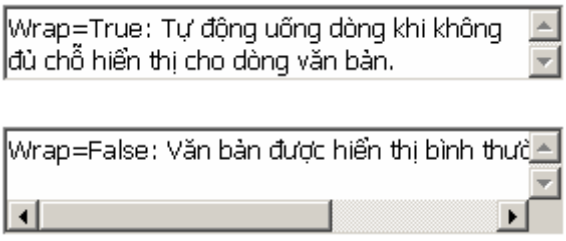
- SingleLine: Hiển thị và nhập liệu 1 dòng văn bản
- MultiLine: Hiển thị và nhập liệu nhiều dòng văn bản
- Password: Hiển thị dấu * thay cho các ký tự có trong Textbox.

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.

2.2.4. Image

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

Thuộc tính

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.

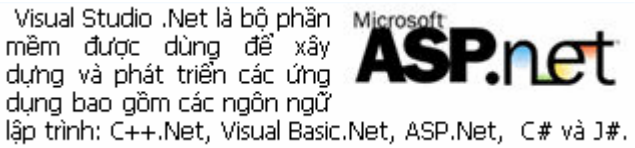
- NotSet

- Left

- Middle

- TextTop

- Right



2.2.5. Button, ImageButton, LinkButton

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 tri thuộc tính CommandName cho các điều khiển này, chúng ta gọi tên chung cho các điều khiển này là Command Button.

Các thuộc tính chung của Button, ImageButton, LinkButton

Thuộc tính	Ý nghĩa
Text	Chuỗi văn bản hiển thị trên điều khiển.

CommandName	Tên lệnh. Được sử dụng trong sự kiện Command.
CommandArgument	Thông tin bổ sung cho sự kiện Command.
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.

Ngoài những thuộc tính trên, điều khiển ImageButton còn có các thuộc tính ImageURL, ImageAlign và AlternateText như điều khiển Image.

Button, LinkButton và ImageButton

2.2.6. Listbox và DropdownList

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).

a. Các thuộc tính

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.

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.

Rows: Qui định chiều cao của ListBox theo số dòng hiển thị.

SelectionMode: 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.

b. 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.

c. Tìm hiểu về tập hợp Items

Add: Thêm mục mới vào cuối danh sách, sử dụng phương thức Items.Add

Insert: Thêm mục mới vào danh sách tại một vị trí nào đó, sử dụng phương thức Items.Insert

Count: Trả về số mục (Item) có trong danh sách.

Contains: Kiểm tra xem một Item đã có trong tập hợp Items hay chưa, nếu có, phương thức này sẽ trả về giá trị True, ngược lại, trả về False.

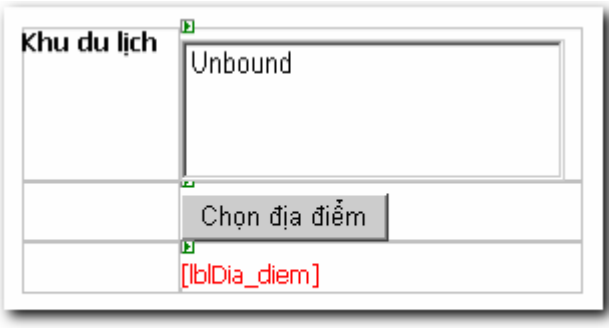
Remove: Xóa đối tượng Item tại ra khỏi danh sách.

- Trong trường hợp các đối tượng Item là kiểu chuỗi, ta truyền vào một chuỗi để xóa. Nếu có nhiều giá trị giống nhau trong danh sách, chỉ có mục chọn đầu tiên bị xóa.
- Trong trường hợp các đối tượng Item là đối tượng, ta truyền vào một biến tham chiếu đến item cần xóa.

RemoveAt: Xóa một item tại vị trí index ra khỏi danh sách.

Clear: Phương thức Clear của tập hợp Items được dùng để xóa tất cả những Item có trong danh sách. Cú pháp

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



Khi thiết kế

Xử lý sự kiện:

```
private void Page_Load()
{
        if (!IsPostBack) {
        lstKhu_dl.Items.Add("Vịnh Hạ Long");
        lstKhu_dl.Items.Add("Phan Thiết - Mũi Né");
        lstKhu_dl.Items.Add("Nha Trang");
        lstKhu_dl.Items.Add("Đà Lạt");
        }
}
private void butChon_dia_diem_Click()
{
        int i;
        lblDia_diem.Text = "";
        for (i = 0; i <= lstKhu_Dl.Items.Count - 1; i++) {
        if (lstKhu_dl.Items(i).Selected) {
        lblDia_diem.Text += lstKhu_dl.Items(i).Text() + "<br>";
        }
        }
}
```

Khi thi hành

2.2.7. Checkbox, RadioButton

a. Các thuộc tính

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.

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.

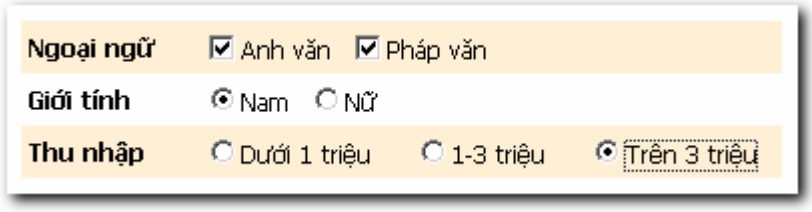
b. Ví dụ

Nhóm các RadioButton Giới tính, Thu nhập

Danh sách các điều khiển

Điều khiển	Loại	Thuộc tính	Giá trị
chkAnh_van	CheckBox	Checked	True
chkPhap_van	CheckBox		
rbtNam	RadioButton	Checked	True
	GroupName	Gioi_tinh	
rbtNu	RadioButton	GroupName	Gioi_tinh

rbtThu_nhapA	RadioButton	GroupName	Thu_nhap
rbtThu_nhapB	RadioButton	Checked	True
	GroupName	Thu_nhap	
rbtThu_nhapC	RadioButton	GroupName	Thu_nhap



Tạo nhóm cho các điều khiển RadioButton

2.2.8. CheckBoxList, RadioButtonList

Hai điều khiển này được dùng để tạo ra một nhóm các CheckBox/Radio Button. Do đây là điều khiển danh sách nên nó cũng có thuộc tính Items chứa tập hợp các mục chọn như ListBox/DropDownList. Các thao tác trên tập hợp Items, xử lý mục chọn cũng tương tự như ListBox/DropDownList.

a. Các thuộc tính

RepeatColumns: Qui định số cột hiển thị.

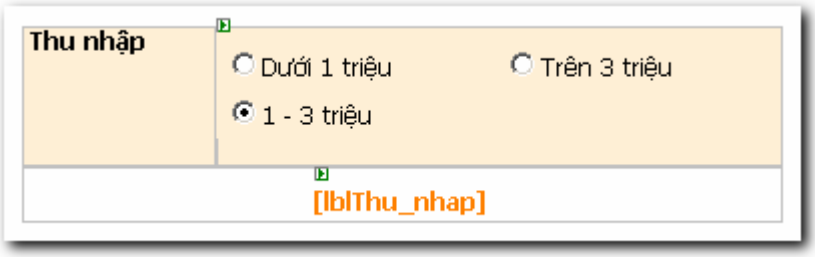
RepeatDirection: Qui định hình thức hiển thị

Vertical: Theo chiều dọc

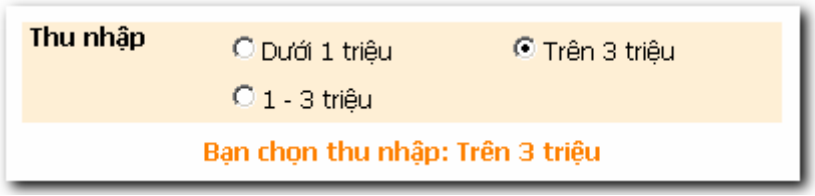
Horizontal: Theo chiều ngang

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.

b. Ví dụ

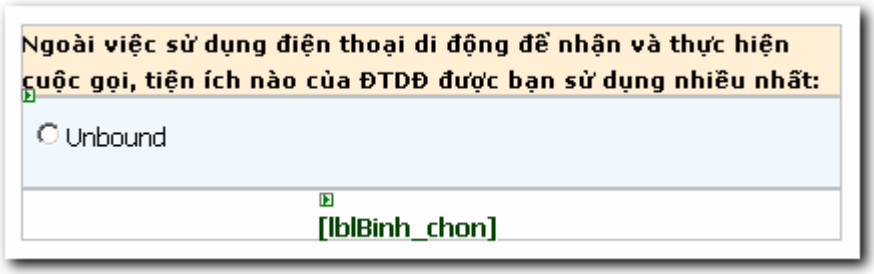


Xử lý sự kiện:



2.2.9. Liên kết dữ liệu với các điều khiển ListBox, DropDownList, CheckBoxList, RadioButtonList

Ví dụ: Liên kết dữ liệu với Sortedlist



Xử lý sự kiện:

```
private void Page_Load()
{
        if (!IsPostBack) {
        SortedList Ds_Binh_chon = new SortedList();
        Ds_Binh_chon.Add("1", "Nhắn tin SMS");
        Ds_Binh_chon.Add("2", "Chụp hình");
        Ds_Binh_chon.Add("3", "Nghe Radio");
        Ds_Binh_chon.Add("4", "Nghe nhạc MP3");
```

```
        Ds_Binh_chon.Add("5", "Xem Video Clip");
        Ds_Binh_chon.Add("6", "Lưu trữ");
        Ds_Binh_chon.Add("7", "Ghi âm");
        Ds_Binh_chon.Add("8", "Chơi game");
        Ds_Binh_chon.Add("9", "Internet");
        rblBinh_chon.DataSource = Ds_Binh_chon;
        rblBinh_chon.DataTextField = "Value";
        rblBinh_chon.DataValueField = "Key";
        rblBinh_chon.DataBind();
        }
}
private void rblBinh_chon_SelectedIndexChanged()
{
        lblBinh_chon.Text = "Bạn chọn chức năng số " +
        rblBinh_chon.SelectedItem.Value;
}
```

Thể hiện thăm dò ý kiến

Trong ví dụ trên, chúng ta tạo ra một danh sách các bình chọn thông qua đối tượng SortedList. Đối tượng SortedList được dùng để lưu trữ danh sách các đối tượng và tự động sắp xếp các đối tượng đó theo giá trị của thuộc tính khóa. Để liên kết điều khiển với một đối tượng dữ liệu (ở ví dụ này là đối tượng SortedList), ta sử dụng thuộc tính DataSource để lấy nguồn dữ liệu.

Hai thuộc tính quan trọng không thể thiếu trong việc thực hiện liên kết dữ liệu đó là: DataTextField và DataValueField. DataTextField là tên thuộc tính (hoặc tên field) của đối tượng dữ liệu mà ta muốn hiển thị. DataValueField là tên thuộc tính (hoặc tên field) chứa là giá trị mà ta muốn nhận về khi người dùng thực hiện chọn các mục trên điều khiển (thông qua thuộc tính SelectedValue hay SelectedItem.Value).

Để hiển thị dữ liệu lên điều khiển khi trang được Load, chúng ta sử dụng phương thức DataBind.

2.3. Điều khiển kiểm tra dữ liệu

Trong phần này chúng ta sẽ tìm hiểu về các điều khiển được dùng để kiểm tra dữ liệu.

Sơ đồ xử lý kiểm tra dữ liệu nhập tại Client và Server

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.

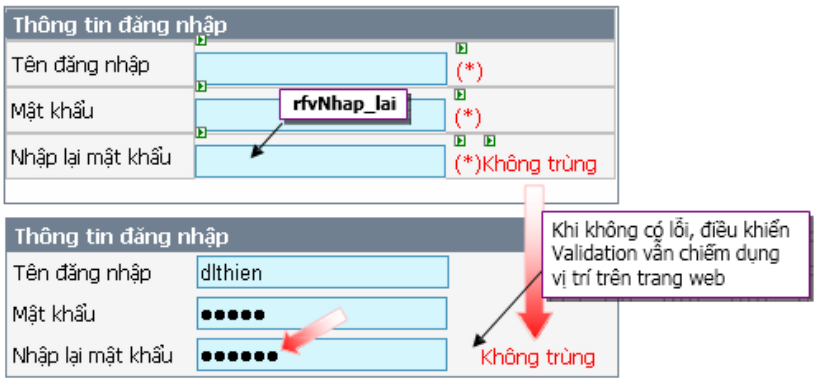
Các thuộc tính chung của các điều khiển Validation Control

Thuộc tính	Ý nghĩa
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
Display	Qui định hình thức hiển thị: ▪ None: Không hiển thị thông báo lỗi (vẫn có kiểm tra dữ

EnableClientScript	Có cho phép thực hiện kiểm tra ở phía Client hay không. Giá trị mặc định là True - có kiểm tra.

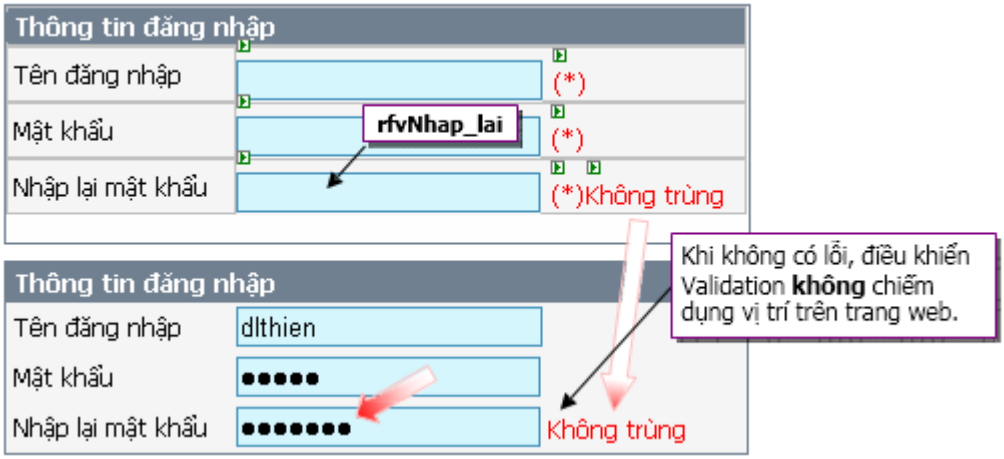
Ví dụ: Minh họa thuộc tính Display: Tại ô nhập lại mật khẩu, ta có 2 điều khiển kiểm tra dữ liệu: một điều khiển kiểm tra không được phép rỗng (rfvNhap_lai), một điều khiển kiểm tra xem nhập lại mật khẩu có giống với mật khẩu đã nhập ở trên hay không.

rfvNhap_lai.Display = Static



Lựa chọn hình thức hiển thị

rfvNhap_lai.Display = Dynamic



Lựa chọn hình thức hiển thị

2.3.1. Điều khiển Required Field Validator

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

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.

2.3.2. Điều khiển Compare Validator

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.

Các thuộc tính

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

- 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.

- 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.

2.3.3. 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.

Các thuộc tính

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

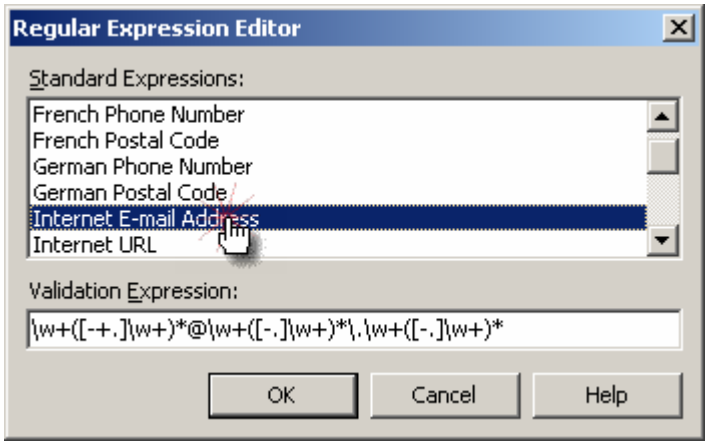
2.3.4. Điều khiển Regular Expression Validator

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:

ValidationExpression: Qui định mẫu kiểm tra dữ liệu.



Hộp thoại Regular Expression

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*
[0-n]	Một ký tự số từ 0 đến 9.
[abc]	Một ký tự: hoặc a hoặc b hoặc c
|	Lựa chọn mẫu này hoặc mẫu khác
\w	Ký tự thay thế phải là một ký tự chữ cái
\d	Ký tự thay thế phải là một ký tự số
\	Thể hiện các ký tự đặc biệt theo sau.
\.	Ký tự thay thế phải là dấu chấm câu (.)
?	Qui định số lần xuất hiện: 0 hoặc 1 lần
*	Qui định số lần xuất hiện: 0 hoặc nhiều lần
+	Qui định số lần xuất hiện: 1 hoặc nhiều lần (ít nhất là
{n}	Qui định số lần xuất hiện: đúng n lần

2.3.5. Điều khiển Custom Validator

Điều khiển này cho phép bạn tự viết hàm xử lý kiểm tra lỗi.

Sự kiện

ServerValidate: Đặt các xử lý kiểm tra dữ liệu trong sự kiện này. Việc kiểm tra này được thực hiện ở Server.

Ví dụ: Xử lý kiểm tra dữ liệu nhập tại điều khiển txtSoA có phải là số chẵn hay không.

```
private void cvSo_chan_ServerValidate(ServerValidateEventArgs args)
{
        if (Val(txtSoA.Text) % 2 == 0) {
        args.IsValid = true;
        } else {
        args.IsValid = false;
        }
}
```

2.3.6. 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.

Các thuộc tính

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.

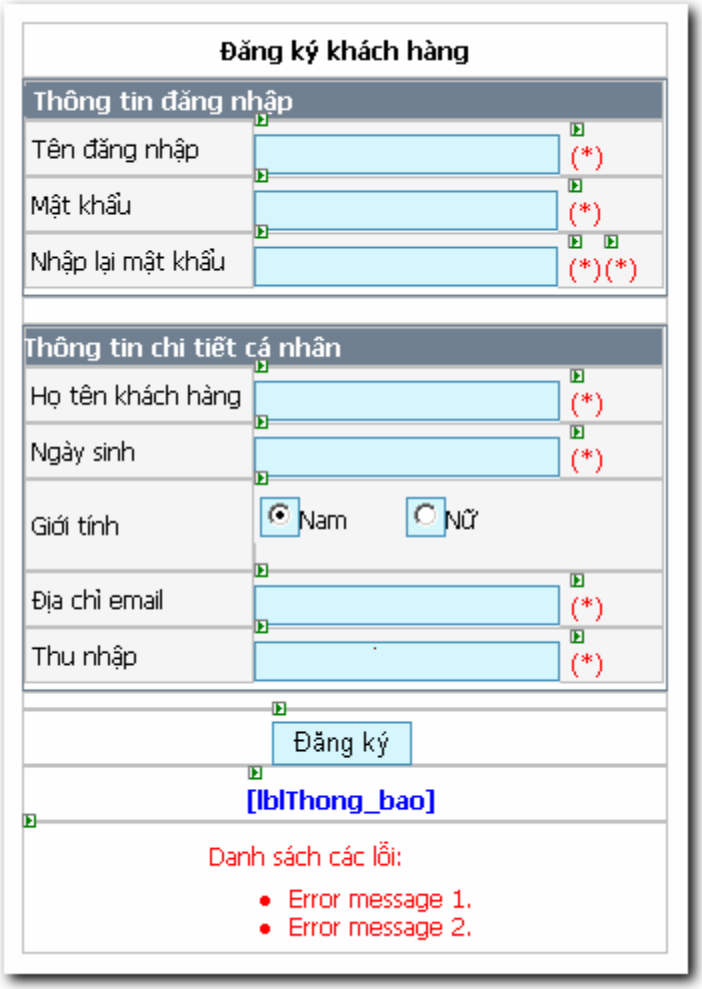
Thông báo lỗi xuất hiện qua cửa sổ MessageBox

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.

Thông báo lỗi hiển thị trực tiếp trên trang Web

Ví dụ: Sử dụng các điều khiển ValidateControl

Trong ví dụ dưới đây, chúng ta thực hiện kiểm tra dữ liệu nhập trên các điều khiển có trong hồ sơ đăng ký khách hàng.



Màn hình hồ sơ khách hàng khi thiết kế

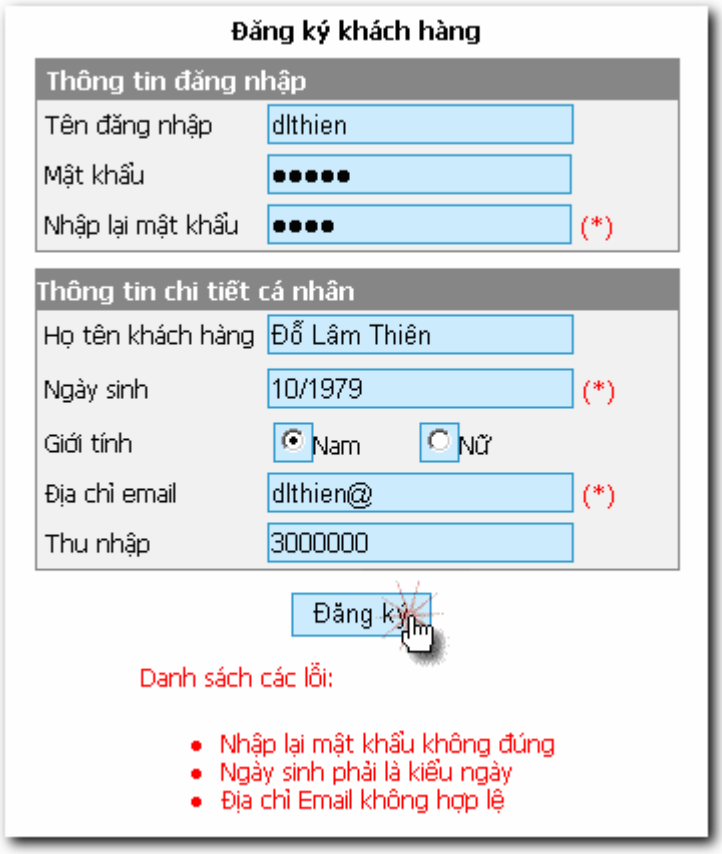
Bảng mô tả thuộc tính của các điều khiển kiểm tra dữ liệu

Điều khiển	Loại	Thuộc tính	Giá trị
rfvTen_dn	RequiredField	ControlToValidate	txtTen_dn
		ErrorMessage	Tên đăng nhập không được rỗng
rfvMat_khau	RequiredField	ControlToValidate	txtMat_khau
		ErrorMessage	Mật khẩu không được rỗng.
rfvNhap_lai	RequiredField	ControlToValidate	txtNhap_lai
		Display	Dynamic
		ErrorMessage	Nhập lại mật khẩu không được rỗng.
cvNhap_lai	Compare	ControlToValidate	txtNhap_lai
		ControlToCompare	txtMat_khau
		ErrorMessage	Mật khẩu nhập lại chưa đúng.
rfvHo_ten	RequiredField	ControlToValidate	txtHo_ten
		ErrorMessage	Họ tên không được rỗng.
cvNgay_sinh	Compare	ControlToValidate	txtNgay_sinh
		Operator	DataTypeCheck
		Type	Date
		ErrorMessage	Ngày sinh không hợp lệ.
revEmail	RegularExpression	ControlToValidate	txtEmail
		ValidationExpression	Internet Email
		ErrorMessage	Email không hợp lệ.
rvThu_nhap	RangeValidator	ControlToValidate	txtThu_nhap
		MaximumValue	50000000
		MinimumValue	1000000
		Type	Integer
		ErrorMessage	Thu nhập từ 1 triệu đến 50 triệu
vsBang_loi	V-Summary	HeaderText	Danh sách các lỗi
		ShowMessageBox	True
butDang_ky	Button	Text	Đăng ký

Thuộc tính Text của các điều khiển: **(*)**

```
private void butDang_ky_Click()
{
    lblThong_bao.Text = "Đăng ký thành công";
}
```

Các thông báo lỗi xuất hiện trên màn hình nhập liệu khi dữ liệu nhập không hợp lệ.



Thông báo lỗi khi dữ liệu nhập liệu không hợp lệ

Các thông báo lỗi xuất hiện qua hộp thoại khi dữ liệu nhập không hợp lệ:

Bảng lỗi qua cửa sổ MessageBox

2.4. Một số điều khiển khác

2.4.1. Điều khiển Literal

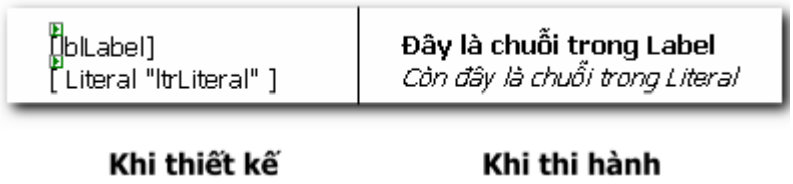
Tương tự như điều khiển Label, điều khiển Literal cũng được sử dụng để hiển thị chuỗi văn bản trên trang Web.

Nếu muốn hiển thị một chuỗi văn bản trên trang Web, chúng ta có thể đánh nội dung trực tiếp vào trang Web mà không cần phải sử dụng điều khiển. Chỉ sử dụng các điều khiển như Label, Literal để hiển thị khi cần thay đổi nội

dung hiển thị ở phía server.

Điểm khác biệt chính giữa Label và Literal là khi hiển thị nội dung lên trang web (lúc thi hành), điều khiển Literal không tạo thêm một tag Html nào cả, còn Label sẽ tạo ra một tag span (được sử dụng để lập trình ở phía client).

Ví dụ:



Lệnh xử lý:

```
private void Page_Load()
{
    lblLabel.Text = "<B>Đây là chuỗi trong Label</B>";
    ltrLiteral.Text = "<I>Còn đây là chuỗi trong Literal</I>";
}
```

Chọn chức năng từ thực đơn View | Source trên Browser:

```
<span id="lblLabel"><B>Đây là chuỗi trong Label</B></span><br>
<I>Còn đây là chuỗi văn bản trong Literal</I>
```

2.4.2. Điều khiển Panel và PlaceHolder

Hai điều khiển Panel và PlaceHolder được sử dùng để chứa các điều khiển khác. Thuộc tính thường dùng của 2 điều khiển này là Visible. Nếu giá trị của thuộc tính này = True, các điều khiển chứa bên trong sẽ được hiển thị, ngược lại (Visible = False), không có điều khiển nào chứa bên trong được hiển thị.

Tuy nhiên, điều khiển Panel cho phép chúng ta kéo những điều khiển vào bên trong nó lúc thiết kế, còn điều khiển PlaceHolder thì không.

Để thêm những điều khiển vào bên trong lúc thi hành, chúng ta phải thực hiện thông qua tập hợp Controls của điều khiển:

Ví dụ:

```
TextBox txtSo_A = new TextBox();
pnl.Controls.Add(txtSo_A);
```

2.4.3. Điều khiển Table

Điều khiển Table thường được sử dụng để hiển thị dữ liệu theo các dòng lệnh đã được cài đặt. Điều khiển này được tạo thành từ tập hợp các dòng (thông qua thuộc tính **Rows**) - TableRow, mỗi dòng được tạo thành từ tập hợp các ô (thông qua thuộc tính **Cells**) – TableCell.

Mỗi ô - cell (TableCel) trong Table có thể là một điều khiển chứa các điều khiển khác. Ta có thể thao tác với các điều khiển trong ô thông qua tập hợp **Controls** của ô đó.

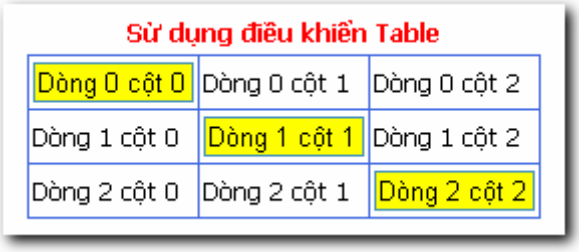
Ví dụ: Sử dụng các điều khiển Table

Màn hình khi thiết kế

Xử lý sự kiện:

```
private void Page_Load()
{
Ve_bang(3, 3);
}

public void Ve_bang(int pSo_dong, int pSo_cot)
{
object r, c;
TableRow tr;
TableCell td;
//Tiến hành tạo bảng dữ liệu
```

```
for (r = 0; r <= pSo_dong - 1; r++) {
tr = new TableRow();
tr.Height = new Unit(20);
for (c = 0; c <= pSo_cot - 1; c++) {
td = new TableCell();
if (r == c) {
//Xử lý thêm các Textbox
TextBox txtTextbox = new TextBox();
txtTextbox.Text = "Dòng " + r + " cột " + c;
txtTextbox.BackColor = Color.Yellow;
txtTextbox.Width = new Unit(90);
td.Controls.Add(txtTextbox);
} else {
td.Text = "Dòng " + r + " cột " + c;
}
tr.Cells.Add(td);
}
tblBang.Rows.Add(tr);
}
}
```



Điều khiển Table

2.4.4. Điều khiển AdRotator

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.

a. Thuộc tính

AdvertisementFile: Tên tập tin dữ liệu (dưới dạng xml) cho điều khiển.

Dưới đây là cú pháp của tập tin Advertisement **(*.xml)**

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

NavigateUrl: Đường dẫn đến trang web sẽ được liên kết đến khi người dùng nhấn vào hình ảnh đang hiển thị.

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.

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.

Impressions: Tham số này quyết định tầng suất hiển thị của hình ảnh. Giá trị này càng lớn, khả năng hiển thị càng nhiều.

KeywordFilter: 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.

Giá trị của tham số này mặc định không được thiết lập € Hiển thị tất cả những hình có trong tập tin XML. Trong trường hợp nếu không có hình nào có giá trị Keyword bằng giá trị của thuộc tính này, sẽ không có hình nào được hiển thị.

Target: Qui định cửa sổ hiển thị trang liên kết

- _blank: Trang liên kết sẽ được mở ở một cửa sổ mới.
- _self: Trang liên kết sẽ được mở ở chính cửa sổ chứa điều khiển.
- _parent: Trang liên kết sẽ được mở ở cửa sổ cha.

b. Sự kiện

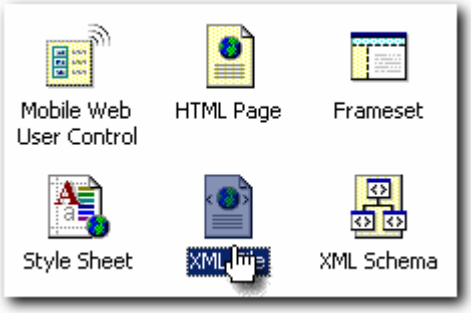
AdCreated:Xảy ra khi điều khiển tạo ra các quảng cáo.

Ví dụ: Tạo Quảng cáo sử dụng điều khiển AdRotator

Bước 1. Thiết kế giao diện

Bước 2. Tạo tập tin dữ liệu: Hinh.xml

- Sử dụng chức năng Add New Item… từ thực đơn ngữ cảnh
- Chọn XML File trong hộp thoại Add New Item



- Nhập vào cú pháp qui định cho tập tin Hinh.xml (theo cú pháp của tập tin Advertisement)

- Chuyển màn hình qua trang Data, nhập liệu trực tiếp trên màn hình này

Data for Ad

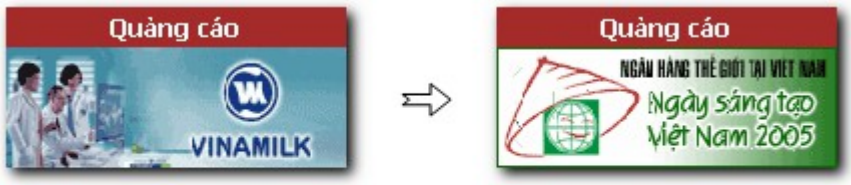
	ImageUrl	NavigateUrl	AlternateText	Keyword	Impre
▶	Hinh\Asp_net.j	http://www.asp.net	Trang chủ As	Hoc_tap	10
	Hinh\Qc_02.gif	http://www.vnexpress.ne	Vnexpress	Tin_tuc	10
	Hinh\Qc_03.gif	http://www.tuoitre.com.v	Tuổi trẻ Onlin	Tin_tuc	10
	Hinh\Qc_04.gif	http://www.thanhnien.co	Thanh niên	Tin_tuc	10
	Hinh\Qc_05.gif	http://www.vnn.vn	Vnn	Tin tức	10
	Hinh\Qc_06.gif	http://www.microsoft.co	Microsoft	Hoc_tap	10
*					

Nhập thông tin hình ảnh quảng cáo

Bước 3. Thiết lập thuộc tính cho điều khiển adQuang_cao

- AdvertisementFile: Hinh.xml
- Target: _blank → Khi nhấn vào sẽ hiển thị liên kết ở cửa sổ mới.
- KeywordFilter: Không thiết lập → Hiển thị tất cả các hình ảnh

Bước 4. Thi hành chương trình



Khi thi hành

2.4.5. Điều khiển 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.

a. Thuộc tính

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.

SeleltedDayStyle: Qui định hình thức hiển thị của ngày đang được chọn.

SeleltedDate: 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

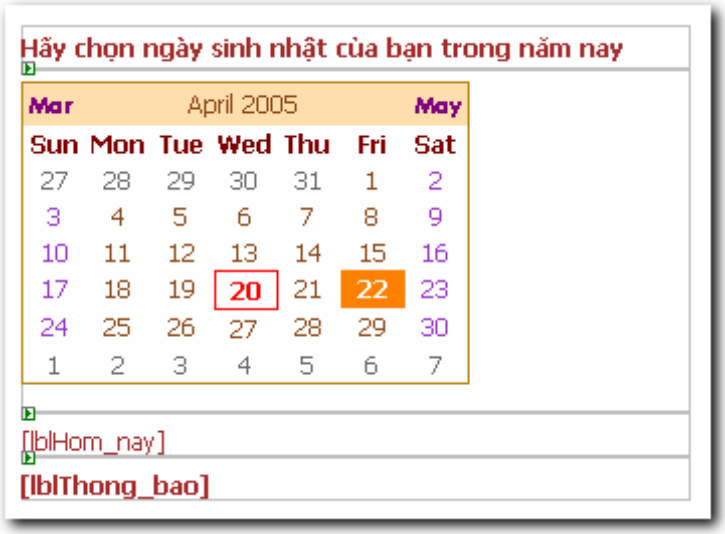
nằm trong tháng hiện hành.

b. Sự kiện

SelectionChanged: Xự kiện này xảy ra khi bạn chọn một ngày khác với giá trị ngày đang được chọn hiện hành

VisibleMonthChanged: Xự kiện này xảy ra khi bạn chọn tháng khác với tháng hiện hành

Ví dụ:



Khi thiết kế

Xử lý sự kiện:

```
private void Page_Load()
{
    lblHom_nay.Text = "Hôm nay ngày " +
    System.DateTime.Today.ToString("dd/MM/yyyy");
}
private void calLich_SelectionChanged()
{
    int lSo_ngay;
    lSo_ngay = Math.Abs(DateDiff(DateInterval.Day, System.DateTime.Today,
    calLich.SelectedDate));
```

```
        if (calLich.SelectedDate > System.DateTime.Today)
            lblThong_bao.Text = "Còn " & lSo_ngay & ngày là đến ngày sinh nhật
        của bạn.";
        else if (calLich.SelectedDate = Date.Today)
                lblThong_bao.Text = "Hôm nay là ngày sinh nhật của bạn" ;
        else
        lblThong_bao.Text = "Sinh nhật bạn đã qua " & lSo_ngay & " ngày.";
}
```

Khi thi hành

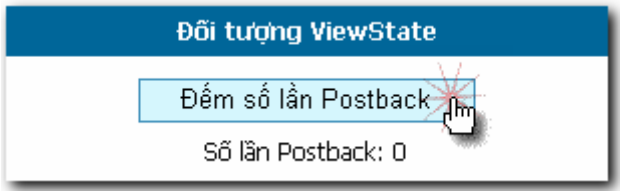
2.5. Đối tượng ViewState

Đối tượng ViewState được cung cấp để lưu lại những thông tin của trang web sau khi web server gởi kết quả về cho Client. Mặc định, các trang web khi được tạo sẽ cho phép sử dụng đối tượng ViewState thông qua thuộc tính EnableViewState (của trang web) = True.

Gán giá trị cho ViewState:

Nhận giá trị từ đối tượng ViewState:

Ví dụ:



Xử lý sự kiện:

```
private void Page_Load()
{
        if (!IsPostBack) {
        ViewState("So_lan") = 0;
        } else {
        ViewState("So_lan") += 1;
        }
        lblTB.Text = "Số lần Postback: " +
        Convert.ToString(ViewState("So_lan"));
}
private void butDem_Click()
{
        lblTB.Text = "Số lần Postback: " +
Convert.ToString(ViewState("So_lan"));
}
```

Về bản chất, các giá trị trong đối tượng ViewState được lưu trong một điều khiển hidden và các giá trị này đã được mã hóa. Đối tượng ViewState giúp chúng ta giảm bớt công sức trong việc lưu trữ và truy xuất các thông tin mà không phải sử dụng nhiều điều khiển hidden.

BÀI 3. CÁC ĐIỀU KHIỂN LIÊN KẾT DỮ LIỆU

A. Mục tiêu

- Sử dụng các điều khiển Data List, DataGrid và Repeater để hiển thị dữ liệu.
- Liên kết dữ liệu với các kiểu tập hợp: ArrayList, SortedList, HashTable, …

B. Nội dung

1. Điều khiển DataGrid

DataGrid là một điều khiển khá linh hoạt và hiệu quả trong việc hiển thị, định dạng và thao tác với dữ liệu. Bên cạnh đó, chúng ta có thể thực hiện sắp xếp dữ liệu, thực hiện phân trang với sự hỗ trợ khá tốt của VS .Net trong quá trình thiết kế.

1.1. Các thao tác định dạng lưới

Để thực hiện các thao tác định dạng, chúng ta chọn chức năng **Property Builder…** từ thực đơn ngữ cảnh.

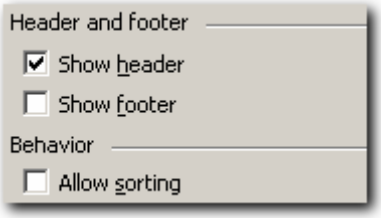
1.1.1. Trang General

Trong trang này, có các mục chọn sau:

Show header: Qui định dòng tiêu đề trên có được phép hiển thị hay không. (mặc định là có hiển thị dòng tiêu đề)

Show footer: Qui định dòng tiêu đề dưới có được phép hiển thị hay không. (mặc định là không hiển thị dòng tiêu đề dưới)

Allow sorting: Có cho phép sắp xếp dữ liệu hay không. (mặc định là không cho phép sắp xếp)



Các mục chọn trong Tab General

1.1.2. Trang Columns (Quản lý thông tin các cột)

Trang Columns quản lý thông tin các cột sẽ hiển thị trên lưới.

Create columns automatically at runtime: Khi chọn chức năng này, DataGrid sẽ tự động phát sinh đầy đủ các cột có trong nguồn dữ liệu. Nếu chúng ta muốn qui định các cột cần hiển thị, chúng ta không chọn chức năng này.

Column list: Qui định các cột được hiển thị trong lưới.

Bound Column: Cột có liên kết với nguồn dữ liệu.

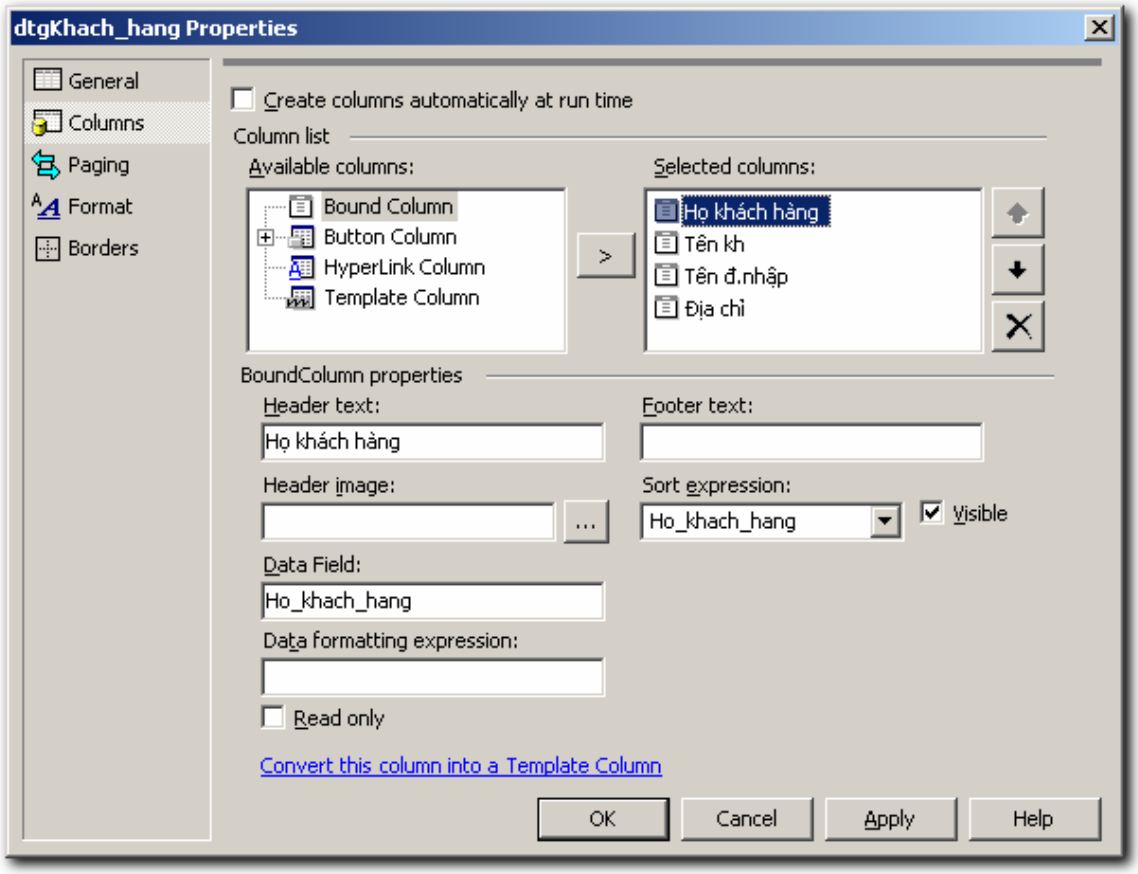
Button Column: Cột dạng nút lệnh đã được thiết kế sẵn. Điều khiển DataGrid cung cấp cho chúng ta 3 loại cột dạng này:

- Select: Nút lệnh chọn dòng dữ liệu
- Edit, Cancel, Update: Các nút lệnh hỗ trợ chức năng cập nhật dữ liệu trực tiếp trên lưới.
- Delete: Nút lệnh xóa dòng dữ liệu

Chúng ta sẽ có dịp tìm hiểu kỹ hơn về các nút lệnh này trong phần Cập nhật dữ liệu trực tiếp trên lưới.

Hyperlink Column: Cột có liên kết dữ liệu dạng liên kết.

Template Column: Cột do người dùng tự thiết kế. Đây là loại cột có khả năng làm việc khá linh hoạt.



Trang Columns

Ví dụ bạn cần hiển thị danh sách khách hàng. Tại cột Phái, bạn không muốn hiển thị Nam/Nữ, thay vào đó, bạn muốn hiển thị điều khiển checkbox thay thế, nếu checkbox được chọn - thể hiện phái Nam và ngược lại. Trong tình huống này, TemplateColumn là sự chọn lựa tốt dành cho bạn.

Họ khách hàng	Tên kh	Tên đăng nhập	Phái
Databound	Databound	Databound	☐
Databound	Databound	Databound	☐
Databound	Databound	Databound	☐

Chúng ta sẽ tìm hiểu sâu hơn về Template Column ở phần sau.

BoundColumn properties: Qui định thông tin chi tiết cho các cột

- HeaderText, Footer Text: Thông tin tiêu đề trên/dưới của cột
- Header Image: Hình hiển thị trên tiêu đề cột (thay thế thông tin tiêu đề cột - Header Text).

- Sort Expression: Biểu thức sắp xếp của cột.
- Visible: Qui định cột có được hiển thị hay không.
- DataField: Qui định tên field hay tên thuộc tính của đối tượng dữ liệu cần hiển thị.
- Data formatting expression: Biểu thức định dạng dữ liệu.

  Mẫu định dạng: **{0:<chuỗi định dạng>}.**

  Ví dụ:

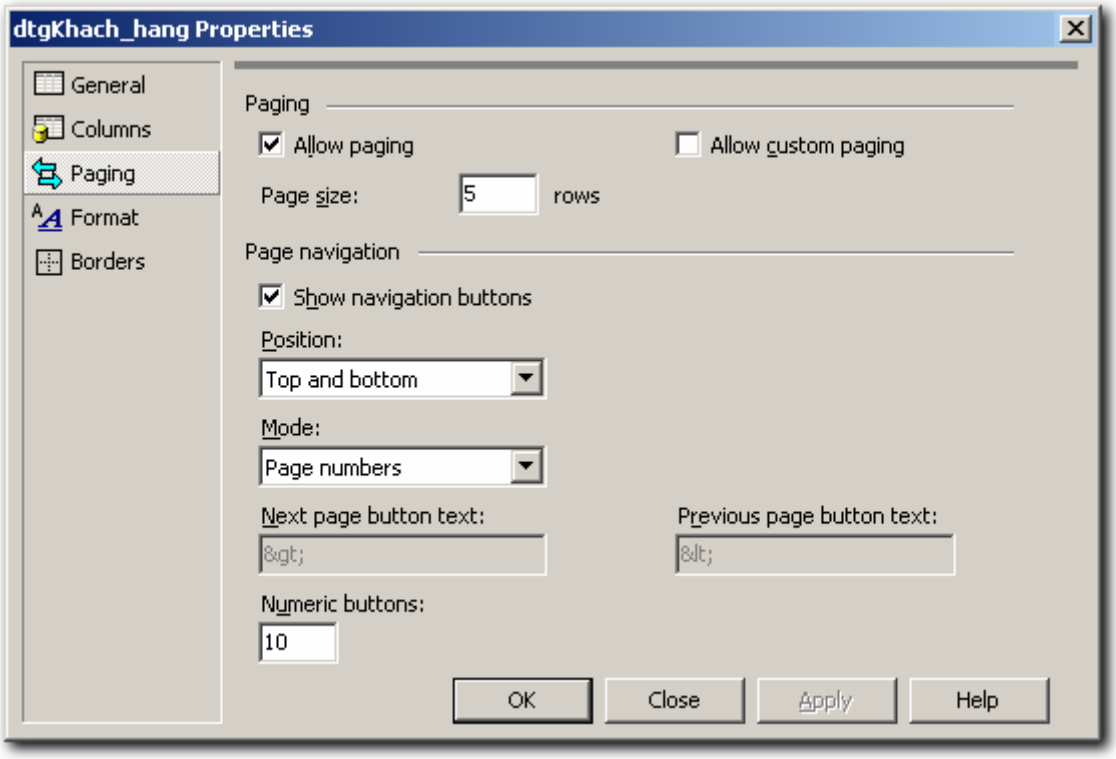
  Định dạng số: {0:000.00}, {0:0.##}

  Định dạng ngày giờ: {0:dd/MM/yyyy}, {0:hh/mm/ss tt}
- Read Only: Chọn giá trị này để cột chỉ được phép đọc, không cho phép cập nhật dữ liệu.

  Convert this column into a Template Column: Chuyển cột hiện hành thành cột dạng Template Column.

1.1.3. Trang Paging (Quản lý phân trang)

Trang này quản lý việc phân trang của DataGrid.



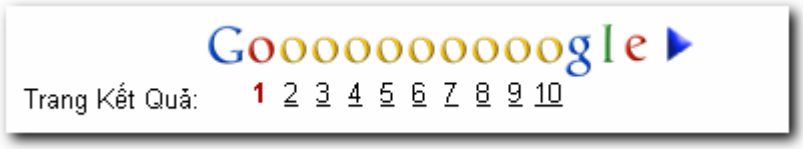
Allow paging: Có cho phép phân trang hay không.

Page size: Qui định số dòng của mỗi trang.

Show navigation buttons: Có hiển thị bộ nút để di chuyển từ trang này qua trang khác hay không. Giá trị mặc định là True.

Possition: Qui định vị trí hiển thị của bộ nút di chuyển. Ở phía trên thanh tiêu đề, ở phía dưới hay cả hai.

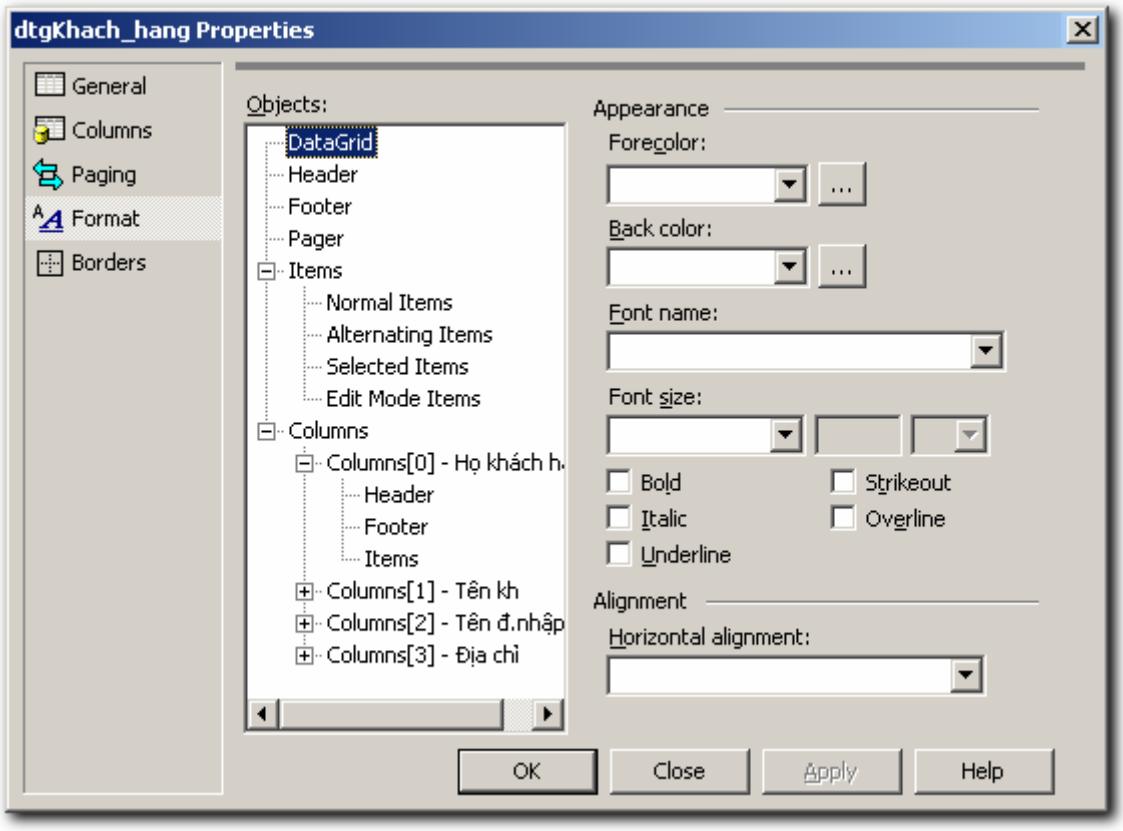
Mode: Qui định hình thức hiển thị của bộ nút di chuyển. Hiển thị dạng số trang hay là các chuỗi ký tự đại diện (Next page/Previous page button text). Trong trường hợp hiển thị dạng số, Numeric buttons qui định số nút lệnh được hiển thị tối đa.



Google hiển thị kết quả được phân trang theo dạng số

1.1.4. Trang Format (Định dạng)

Trang Format quản lý việc định dạng hiển thị trên điều khiển DataGrid. Các định dạng chung như: Màu chữ, màu nền, Font chữ, kích cỡ, in đậm / in nghiêng / gạch dưới và canh lề.



Trang Format

DataGrid: Qui định các định dạng chung cho lưới

Header: Định dạng cho dòng tiêu đề.

Footer: Định dạng cho dòng tiêu đề dưới.

Pager: Định dạng cho dòng chứa các nút lệnh phân trang.

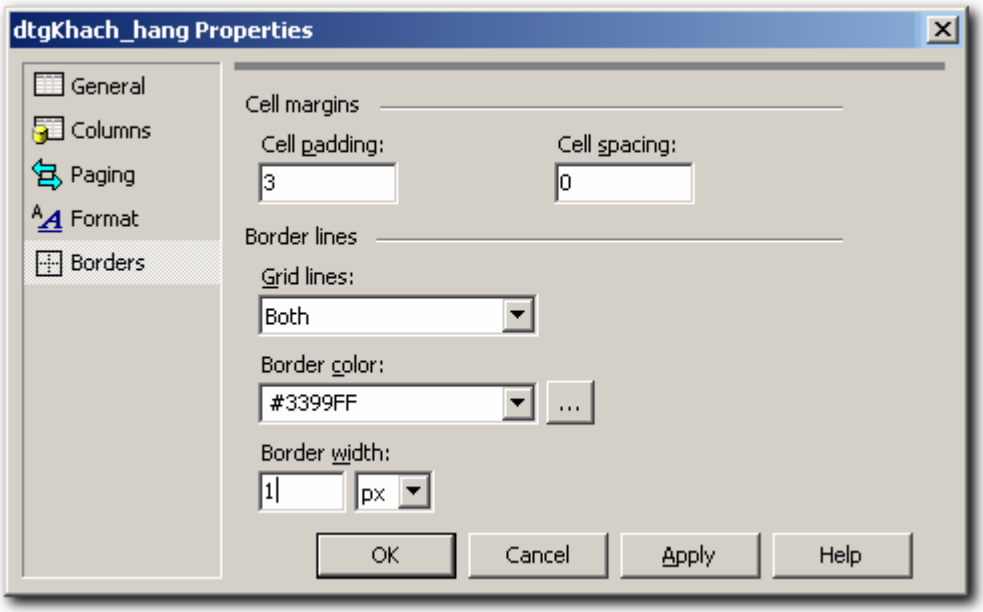
Items

- Normal Items: Định dạng cho các dòng dữ liệu.
- Alternating Items: Định dạng hiển thị cho các dòng lẻ.
- Selected Items:Định dạng hiển thị cho dòng đang được chọn.
- Edit Mode Items: Định dạng hiển thị cho dòng đang ở trạng thái hiệu chỉnh dữ liệu.

Columns: Qui định độ rộng và các định dạng riêng cho từng cột.

1.1.5. Trang Borders (Khung viền)

Trang Borders quản lý việc kẻ khung viền cho lưới.



Cell margin

Cell padding: Qui định khoảng cách giữa nội dung trong ô với các đường viền của ô.

Họ khách hàng
Databound
Databound
Databound

Cell padding = 0

Họ khách hàng
Databound
Databound

Cell padding = 3

Cell spacing: Qui định khoảng cách giữa các ô

Họ khách hàng
Databound
Databound
Databound

Cell spacing = 3

Ví dụ: Điều khiển DataGrid sau khi được định dạng

Danh sách khách hàng

Họ khách hàng	Tên kh	Tên d.nhập	Địa chỉ
Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound

Lưới khách hàng sau khi được định dạng

Mã lệnh xử lý:

```
private void Page_Load()
{
        if (!IsPostBack) {
        Lien_ket_du_lieu();
        }
}
```

Danh sách khách hàng

Họ khách hàng	Tên kh	Tên d.nhập	Địa chỉ
Hoàng Chinh	Thảo	hcthao	118 Nhân Chính
Mai Giang	Giang	mggiang	85/6 Lê Quang Sung
Bùi Thị Mộng	Ánh	btmanh	97/92/10 Lê Anh Xuân
Trịnh Linh	Tâm	tltam	17 Nguyễn Thái Học
Nguyễn Giang	Thắng	ngthang	45 Nguyễn Thị Tần

1.2. Xử lý sắp xếp

Sắp xếp dữ liệu trên lưới là một công việc rất cần thiết đối với người sử dụng. Hãy thử tưởng tượng xem trong trường hợp chúng ta có khá nhiều dữ liệu hiển thị trên màn hình (giả sử là danh sách nhân viên chẳng hạn), thật khó

để chọn ra các nhân viên có thâm niên làm việc lâu nhất hay các nhân viên có số giờ tham gia đề án nhiều nhất…. Với chức năng sắp xếp trên lưới sẽ giúp cho người dùng dễ dàng chọn ra các nhân viên thỏa mãn các yêu cầu trên.

Để thực hiện được thao tác sắp xếp dữ liệu trên lưới, chúng ta cần phải thực hiện các công việc sau:

Giá trị thuộc tính Allow sorting = True

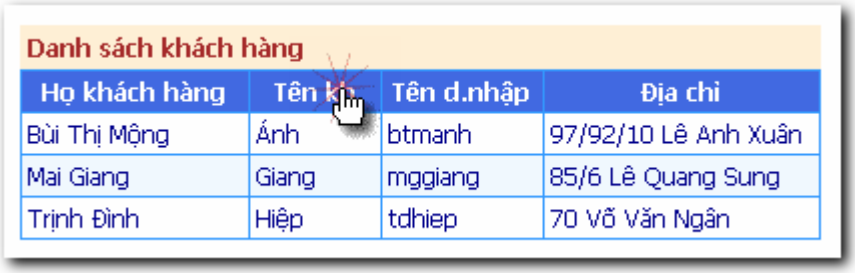
Nhập giá trị cho thuộc tính Sort expression của các cột cần sắp xếp.

Xử lý sự kiện **SortCommand**

Trong sự kiện trên, giá trị **e.SortExpression** cho biết thông tin của cột được chọn sắp xếp.

```
private void Page_Load()
{
        if (!IsPostBack) {
        dtgKhach_hang.DataSource = Doc_ds_khach_hang();
        dtgKhach_hang.DataBind();
        }
}
private void dtgKhach_hang_SortCommand()
{
        dtgKhach_hang.DataSource = Doc_ds_khach_hang(e.SortExpression);
        dtgKhach_hang.DataBind();
}
  public                                                    DataTable
  Doc_ds_khach_hang([System.Runtime.InteropServices.OptionalAttribute,
  System.Runtime.InteropServices.DefaultParameterValueAttribute("")]        string
  pChuoi_sap_xep)
  {
        string sKet_noi;
```

```
        sKet_noi = "Provider=Microsoft.Jet.Oledb.4.0;Data Source=" +
        Server.MapPath("..\\Data\\QlBanSach.mdb");
        OleDbConnection cnKet_noi = new OleDbConnection(sKet_noi);
        DataSet dsCSDL = new DataSet();
        string sLenh_sql = "Select *  From KHACH_HANG";
        sLenh_sql += IIf(pChuoi_sap_xep == "", "", " Order by " +
        pChuoi_sap_xep);
        //Mở và đóng kết nối ngay khi thực hiện xong
        cnKet_noi.Open();
        OleDbDataAdapter daBo_doc_ghi = new OleDbDataAdapter(sLenh_sql,
        cnKet_noi);
        cnKet_noi.Close();
        daBo_doc_ghi.Fill(dsCSDL, "KHACH_HANG");
        return dsCSDL.Tables("KHACH_HANG");
}
```



Sắp xếp khách hàng tăng dần theo tên

1.3. Xử lý phân trang

Phân trang dữ liệu không những giúp cho việc xem và tìm kiếm thông tin được dễ dàng mà còn giảm được khối lượng dữ liệu cần được truyền tải từ Server về Client. Việc phân trang trong ASP.Net được thực hiện khá dễ dàng, chỉ với một số thao tác đơn giản.

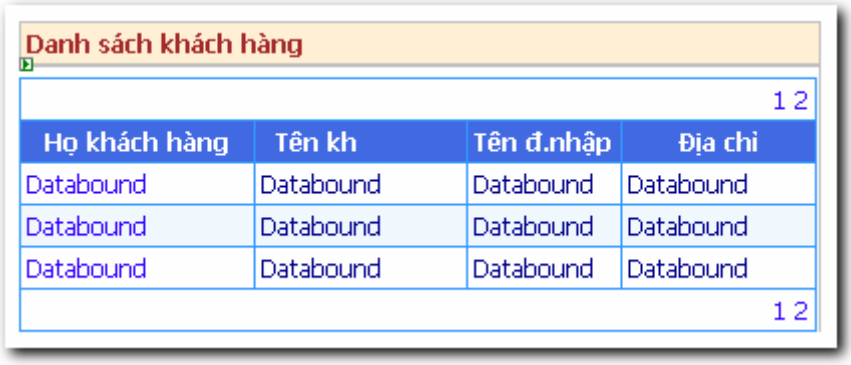
Để thực hiện phân trang, chúng ta cần phải thực hiện các công việc sau:

Qui định các thông số cần thiết cho việc phân trang (xem Quản lý phân trang ở

phần Các thao tác định dạng lưới).

Xử lý sự kiện **PageIndexChanged**

Trong sự kiện trên, giá trị **e.NewPageIndex** cho biết trang được chọn để hiển thị dữ liệu.



Định dạng phân trang

Mã lệnh xử lý:

```
private void Page_Load()
{
        if (!IsPostBack) {
        Lien_ket_du_lieu();
        }
}
private void dtgKhach_hang_PageIndexChanged()
{
        dtgKhach_hang.CurrentPageIndex = e.NewPageIndex;
        Lien_ket_du_lieu();
}
```

Dữ liệu hiển thị được phân trang

1.4. Tùy biến các cột

Trong phần này, chúng tôi sẽ hướng dẫn các bạn tùy biến các cột trên lưới, cụ thể hơn, chúng ta sẽ hiển thị checkbox thay cho giá trị True/False của cột giới tính.

Để thực hiện việc tùy biến các cột, chúng ta cần phải thực hiện 2 giai đoạn sau:

Giai đoạn 1: Thiết kế

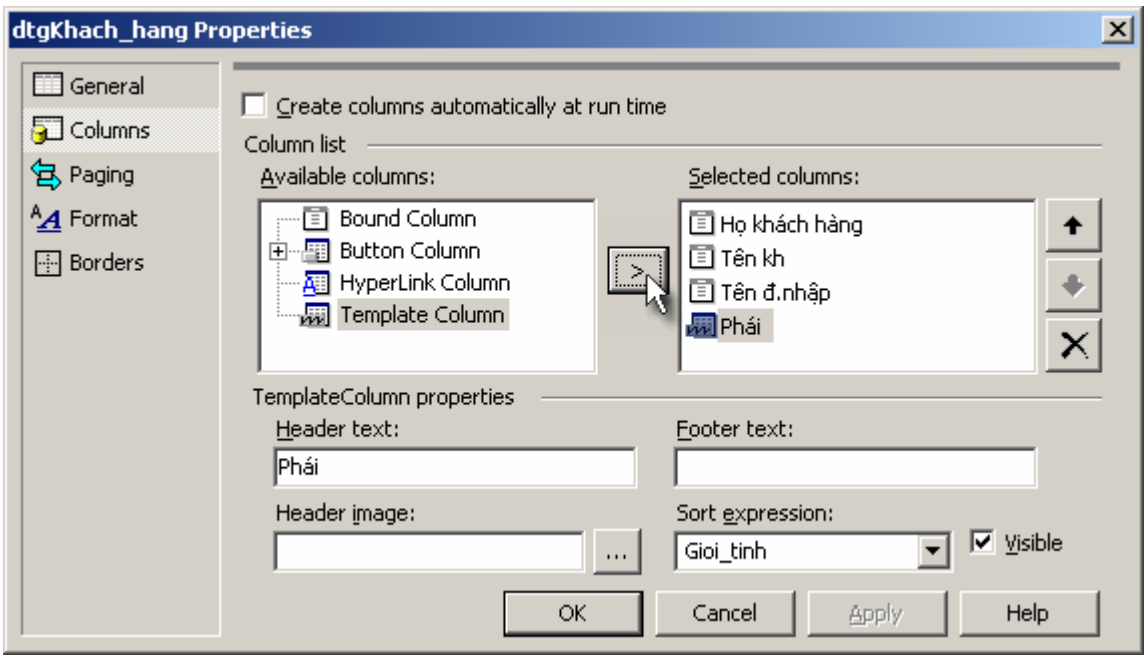
Trong giai đoạn này, chúng ta tùy biến cột theo một yêu cầu cụ thể. Thay vì phải hiển thị ô dữ liệu bình thường (như họ khách hàng, tên khách hàng, …), chúng ta có thể sử dụng điều khiển Checkbox để thay thế cho cột có giá trị luận lý, sử dụng điều khiển Image, Image button hay Hyperlink để hiển thị hình ảnh thay cho chuỗi đường dẫn dẫn đến hình ảnh đó, …

Giai đoạn 2: Xử lý

Sau khi thực hiện hoàn tất giai đoạn thiết kế, đây là lúc chúng ta phải viết các lệnh xử lý để hiển thị dữ liệu theo yêu cầu của mình.

1.4.1. Giai đoạn 1: Thiết kế

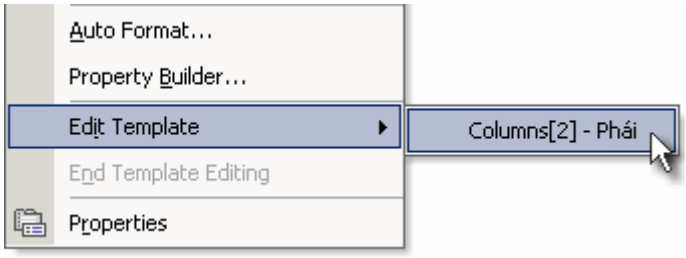
Bước 1. Thêm mới cột Phái, kiểu Template Column. Nhập giá trị Header text, Sort expression cho cột này (nếu cần)



Bổ sung cột Phái kiểu Template Column

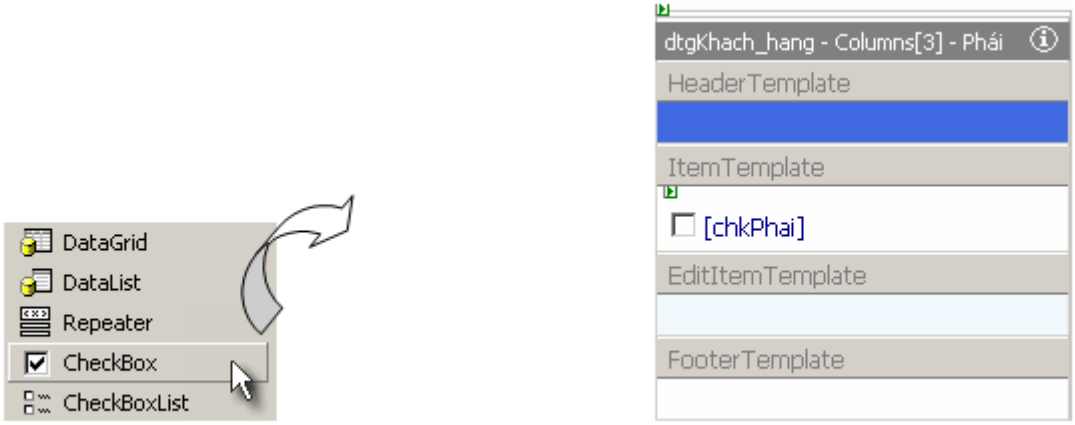
Bước 2. Từ thực đơn ngữ cảnh, chọn Edit Template \ Column[X] –YYY (X: Số thứ tự của cột; Y: Chuỗi tiêu đề của cột)

Chúng ta chọn cột cần hiệu chỉnh.



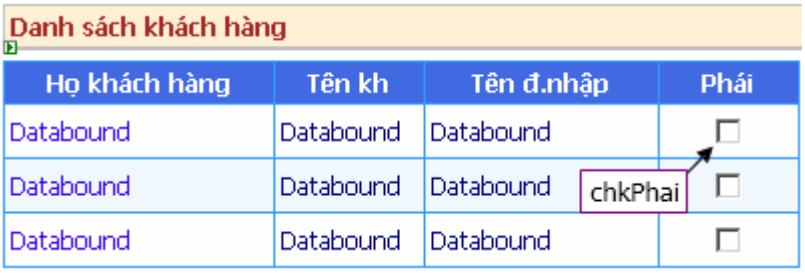
Chọn chức năng hiệu chỉnh cột Phái

Bước 3. Thêm điều khiển checkbox **chkPhai**, vào mục ItemTemplate



Tùy biến cột Phái

Bước 4. Chọn **End Template Editing** từ thực đơn ngữ cảnh sau khi thiết kế xong.



Điều khiển lưới sau khi đã được tùy biến cột Phái

1.4.2. Giai đoạn 2: Xử lý

Khác với Bound column, cột kiểu Template column không tự động hiển thị dữ liệu. Mà làm sao hiển thị dữ liệu được khi chính bản thân các điều khiển (mới được tạo khi thiết kế) không có qui định field cần được hiển thị từ nguồn dữ liệu. Do đó, để hiển thị dữ liệu (theo ý đồ của chúng ta), ta phải viết lệnh các xử lý trong sự kiện **ItemDataBound**

Mã lệnh xử lý:

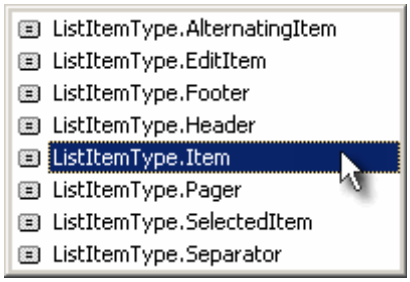
```
(1) private void dtgKhach_hang_ItemDataBound()
{
(2) if (e.Item.ItemIndex < 0)
(3) return;
(4) CheckBox chkPhai;
(5) chkPhai = e.Item.FindControl("chkPhai");
(6) chkPhai.Checked = e.Item.DataItem("Gioi_tinh");
}
```

Trước khi đi vào tìm hiểu các lệnh xử lý trong đoạn lệnh trên, chúng ta cũng nên tìm hiểu ý nghĩa sự kiện ItemDataBound của DataGrid. Sự kiện ItemDataBound xảy ra ngay khi phương thức DataBind được gọi (lẽ đương nhiên là ta phải gán nguồn dữ liệu cho lưới trước đó). Ứng với mỗi dòng dữ liệu sẽ xảy ra một sự kiện ItemDataBound tương ứng.

Dòng lệnh (2): Dòng lệnh này kiểm tra xem lần xảy ra sự kiện này có phải

dành cho dòng dữ liệu hay không. Tại sao cần phải kiểm tra điều kiện này? Bởi vì không chỉ ứng với mỗi dòng dữ liệu, mà còn có các dòng Header, Footer và Pager, … cũng xảy ra trong sự kiện này.

Để biết được lần xảy ra sự kiện dành cho dòng nào, chúng ta kiểm tra giá trị của thuộc tính e.Item.ItemType. Thuộc tính này có các giá trị sau:



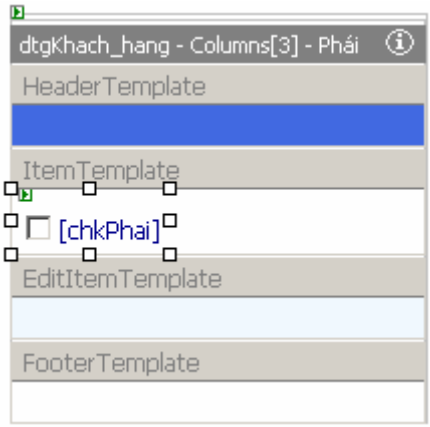
Các giá trị của thuộc tính ItemType

- AlternatingItem: Xảy ra ứng với dòng dữ liệu có chỉ số lẻ (dòng dữ liệu đầu tiên tính từ 0).
- EditItem: Ứng với dòng ở trạng thái hiệu chỉnh dữ liệu.
- Footer: Ứng với dòng tiêu đề dưới.
- Header: Ứng với dòng tiêu đề.
- Item: Xảy ra ứng với dòng dữ liệu có chỉ số chẵn.
- Pager: Ứng với dòng phân trang.
- SelectedItem: Ứng với dòng ở trạng thái đang được chọn.
- Seperator: Ứng với dòng phân cách
- Bên cạnh đó, nếu ta chỉ quan tâm đến lần xảy ra sự kiện này có phải là dòng dữ liệu hay không, ta có thể sử dụng thuộc tính e.ItemIndex.
- e.Item.ItemIndex < 0: Đây không phải là dòng dữ liệu
- e.Item.ItemIndex >= 0: Đây là dòng dữ liệu. Giá trị của thuộc tính này cho biết chỉ số của dòng dữ liệu hiện hành.

Kết quả hiển thị

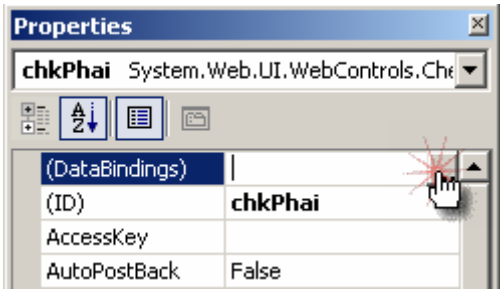
Đối với những xử lý phức tạp, sự kiện ItemDataBound sẽ là sự lựa chọn hàng đầu trong việc tùy biến hiển thị dữ liệu. Tuy nhiên, đối với những xử lý đơn giản, chúng ta có thể thực hiện liên kết dữ liệu trong quá trình thiết kế.

Chọn Edit Template cột Phái, chọn điều khiển chkPhai.



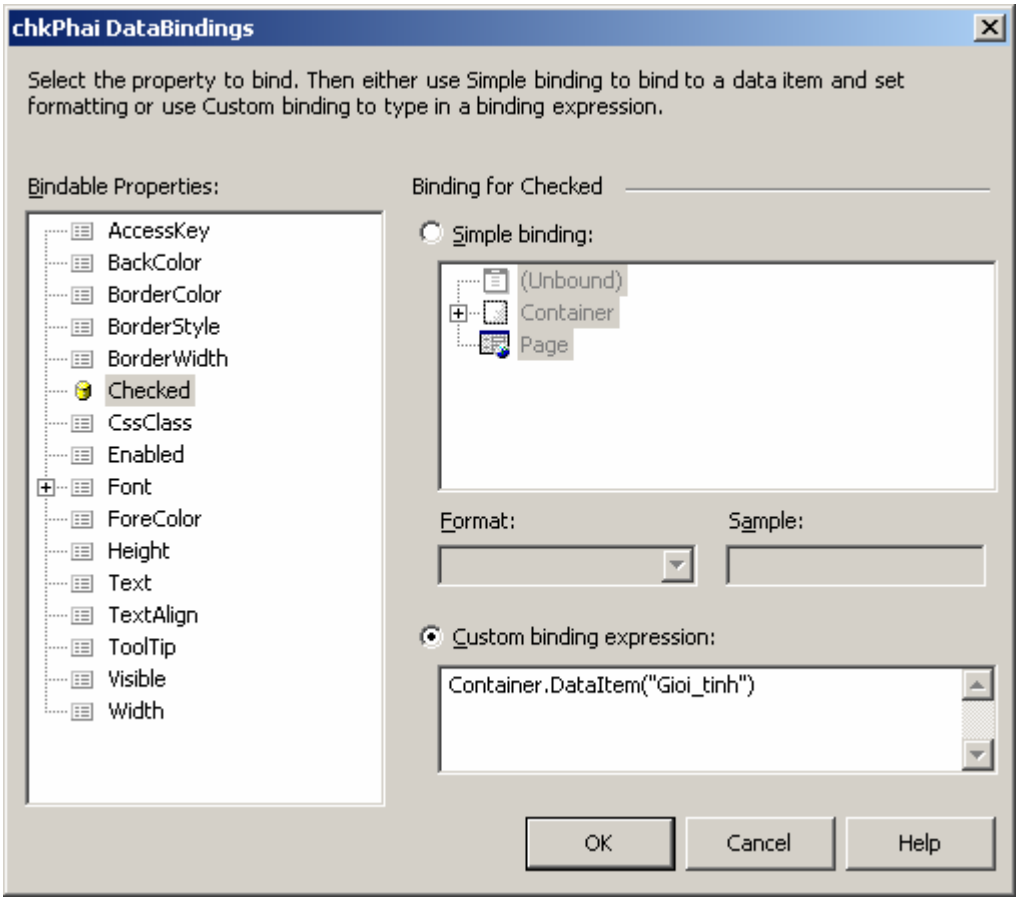
Chọn điều khiển chkPhai trong lúc thiết kế

Trên cửa sổ thuộc tính, chọn (DataBindings)



Chọn chức năng DataBidings

Trên cửa sổ thuộc tính, chọn (DataBindings). Hộp thoại DataBindings của điều khiển chkPhai xuất hiện.



Các thuộc tính có thể liên kết dữ liệu của điều khiển xuất hiện trong Danh sách bên trái hộp thoại. Chọn thuộc tính cần liên kết, chọn loại liên kết là Custom binding expression, nhập chuỗi liên kết dữ liệu trong điều khiển bên dưới theo

cú pháp:

1.5. Cập nhật dữ liệu trực tiếp trên lưới

Cập nhật dữ liệu trực tiếp trên lưới trong ASP.Net được hỗ trợ khá tốt về giao diện. Công việc còn lại của chúng ta là thiết kế các nút lệnh như: Chọn, Sửa/Ghi - Không, Hủy, … và viết các lệnh cập nhật dữ liệu.

1.5.1. Giai đoạn thiết kế

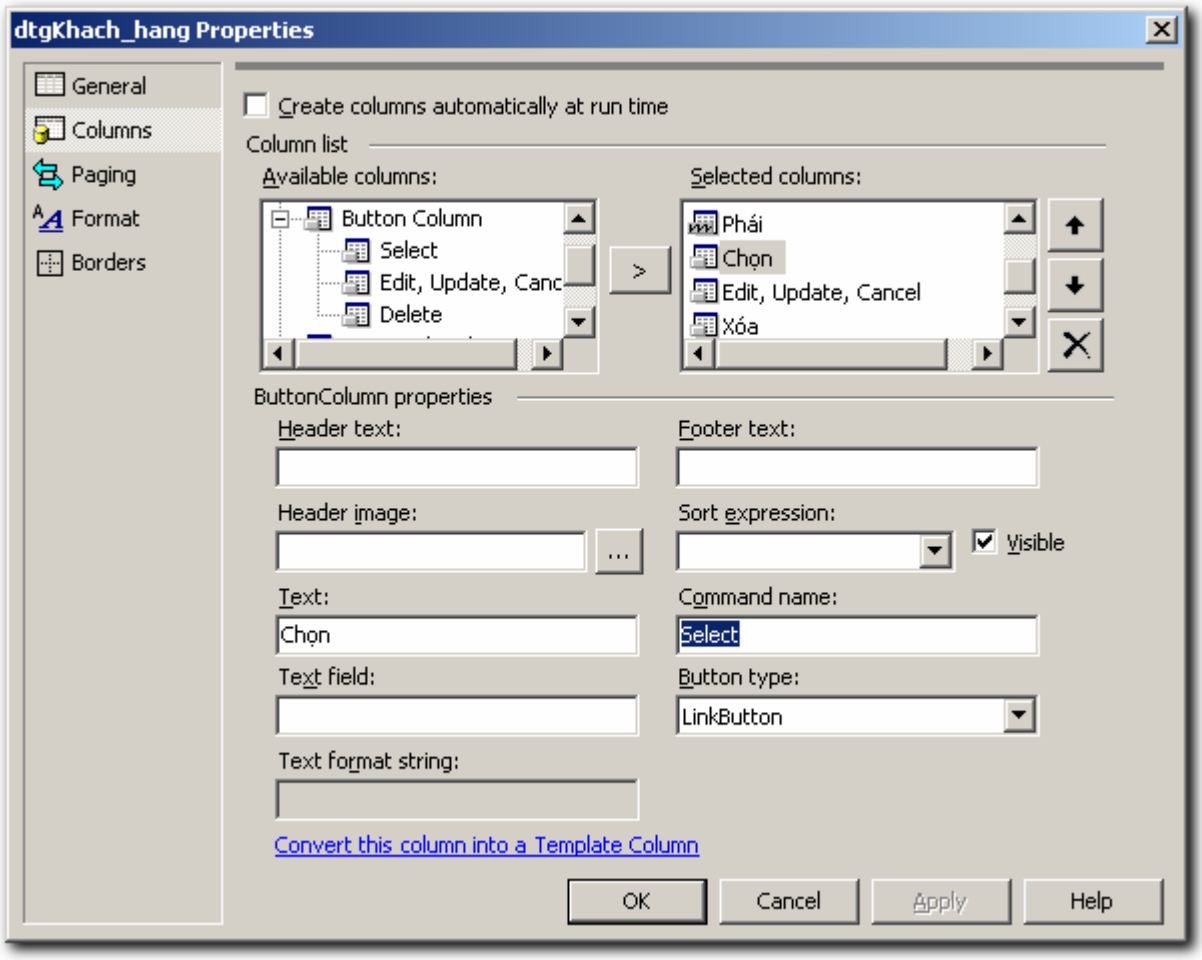
Trong cửa sổ thuộc tính của lưới, chúng ta tạo bộ nút lệnh cần thiết hỗ trợ cho việc cập nhật dữ liệu. Ở ví dụ này, chúng tôi tạo bộ nút (Select - chọn), (Edit, Update, Cancel - Sửa, Ghi, Không) và (Delete - Hủy)

Đối với các nút lệnh trên, các bạn cần chú ý đến giá trị của thuộc tính Command name. Ứng với mỗi nút lệnh có giá trị CommandName khác nhau, nhờ đó, ta viết lệnh xử lý với chức năng tương ứng được chọn.

Select: Command name = "Select"

Edit, Update/Cancel: Command name = "Edit", "Update"/"Cancel"

Delete: Command name = "Delete"



Tạo bộ nút lệnh Thêm - Sửa/Ghi/Không - Hủy

Tìm hiểu về thuộc tính Command Name

Cũng cần bàn thêm một chút ở đây về thuộc tính Command Name. Như các bạn cũng biết, các nút lệnh ở trên (Chọn, Thêm - Sửa / Ghi / Không - Hủy) là do VS.Net hỗ trợ, giá trị thuộc tính CommandName của các nút lệnh trên là những giá trị mặc định được qui định sẵn.

Ứng với mỗi CommandName mặc định, sẽ có các sự kiện để ta thực hiện các xử lý tương ứng:

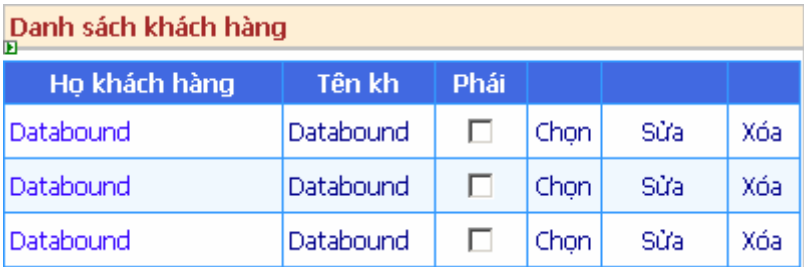
- Command name="Edit" → Sự kiện **EditCommand**
- Command name="Update"→ Sự kiện
- Command name="Cancel" → Sự kiện
- Command name="Delete" → Sự kiện

Chắc hẳn các bạn sẽ thắc mắc tại sao không có sự kiện SelectCommand? Bốn sự kiện được liệt kê trên là 4 sự kiện dành riêng, tương

ứng với giá trị của các Command name mặc định là Edit, Update, Cancel, Delete. Đối với những CommandName có giá trị khác, chúng ta sẽ sử dụng sự kiện dành chung cho tất cả các nút lệnh có thuộc tính CommandName (Button, Linkbutton, ImageButton) được đặt trên lưới - sự kiện **ItemCommand**.

Tại sao vậy? Vì khi ta đặt các nút lệnh vào lưới (sử dụng cột Template column), chúng (các nút lệnh) không còn sự kiện **Click** nữa, thay vào đó, tất cả các nút lệnh khi được nhấn sẽ gây ra sự kiện **ItemCommand.** Dựa vào giá trị **e.CommandName** (tham số trong sự kiện) để chúng ta xác định nút lệnh nào đã được nhấn.

Cũng cần lưu ý thêm ở đây là bất kỳ nút lệnh nào khi được nhấn đều gây ra sự kiện ItemCommand. Do đó, đối với các nút lệnh có giá trị thuộc tính CommandName là Edit, Update, Cancel, Delete khi được nhấn vẫn gây ra sự kiện ItemCommand trước khi gây ra các sự kiện dành riêng cho chúng.



Giao diện lưới sau khi thêm bộ nút lệnh

1.5.2. Giai đoạn xử lý

Xử lý chọn mẩu tin

Chọn mẫu tin trên lưới

private void dtgKhach_hang_ItemCommand()

```
        {
        if (e.CommandName == "Select") {
        dtgKhach_hang.SelectedIndex = e.Item.ItemIndex;
        Lien_ket_du_lieu();
        }
}
```

Xử lý sửa, ghi, không

Muốn cập nhật dữ liệu, ta cần xác định khách hàng được cập nhật thông qua Mã khách hàng.

Để lấy Mã khách hàng:

- Gán thuộc tính **DataKeyField** của điều khiển lưới = "MKH"
- <lưới>.**DataKeys**(<chỉ số i>) € Trả về Mkh tại dòng <chỉ số i>

Chọn mẫu tin để cập nhật dữ liệu

```
private void Page_Load()
{
        if (!IsPostBack) {
                dtgKhach_hang.DataKeyField = "MKH";
                Lien_ket_du_lieu();
        }
}
private void dtgKhach_hang_EditCommand()
{
        dtgKhach_hang.EditItemIndex = e.Item.ItemIndex;
```

```
        Lien_ket_du_lieu();
}
private void dtgKhach_hang_UpdateCommand()
{
        //Khai báo và khởi tạo biến kết nối: cnKet_noi
        //Lấy dữ liệu mà người dùng vừa cập nhật
        TextBox lHo_kh = e.Item.Cells(0).Controls(0);
        TextBox lTen_kh = e.Item.Cells(1).Controls(0);
        CheckBox lPhai = e.Item.FindControl("chkPhai");
        int lMkh = dtgKhach_hang.DataKeys(e.Item.ItemIndex);
        //Tạo đối tượng Command để cập nhật dữ liệu
        OleDbCommand cmdLenh = new OleDbCommand();
        cmdLenh.Connection = cnKet_noi;
        cmdLenh.CommandText = "Update KHACH_HANG " + "Set
Ho_khach_hang=?, Ten_khach_hang=?, " + "Gioi_tinh=? Where MKH=?";
        //Truyền tham số cho đối tượng Command
        cmdLenh.CommandType = CommandType.Text;
        cmdLenh.Parameters.Add("Ho_kh", lHo_kh.Text);
        cmdLenh.Parameters.Add("Ten_kh", lTen_kh.Text);
        cmdLenh.Parameters.Add("Phai", lPhai.Checked);
        cmdLenh.Parameters.Add("Mkh", lMkh);
        //Thi hành Command
        cnKet_noi.Open();
        cmdLenh.ExecuteNonQuery();
        cnKet_noi.Close();
        //Tắt chế độ cập nhật dữ liệu
        dtgKhach_hang.EditItemIndex = -1;
        //Hiển thị dữ liệu mới cập nhật lên lưới
        Lien_ket_du_lieu();
```

```
}
private void dtgKhach_hang_CancelCommand()
{
        dtgKhach_hang.EditItemIndex = -1;
        Lien_ket_du_lieu();
}
```

Hiệu chỉnh độ rộng của các Textbox khi dòng ở trạng thái sửa

Bạn có thể bổ sung đoạn lệnh sau (trong sự kiện ItemDataBound) để hiệu chỉnh độ rộng các Textbox của dòng ở trạng thái sửa.

```
if (e.Item.ItemType == ListItemType.EditItem) {
        ((TextBox)e.Item.Cells(0).Controls(0)).Width = new Unit(133);
        ((TextBox)e.Item.Cells(1).Controls(0)).Width = new Unit(63);
}
```

Xử lý hủy mẫu tin:

```
private void dtgKhach_hang_DeleteCommand()
{
        //Thực hiện xóa dòng dữ liệu ở đây
        //Xử lý tương tự như Update Command
        //Hiển thị dữ liệu mới cập nhật lên lưới
        Lien_ket_du_lieu();
}
```

2. Điều khiển DataList

2.1. Sử dụng DataList để hiển thị dữ liệu

Như điều khiển DataGrid, điều khiển DataList được sử dụng để hiển thị dữ liệu. Tuy nhiên, đối với DataList, chúng ta phải tự thiết kế hình thức hiển thị dữ liệu (giống như Template Column của DataGrid).

Sử dụng DataList hiển thị thông tin sách

Một số thuộc tính cần chú ý của DataList

RepeatDirection: Qui định hướng hiển thị dữ liệu

- Horizontal: Hiển thị dữ liệu theo chiều ngang

RepeatDirection = Horizontal

- Vertical (mặc định):

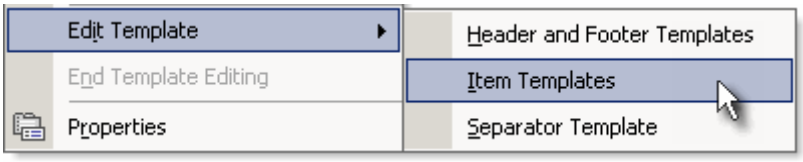
RepeatDirection = Vertical

RepeatColumns: Qui định số cột hiển thị của DataList

RepeatColumns = 3

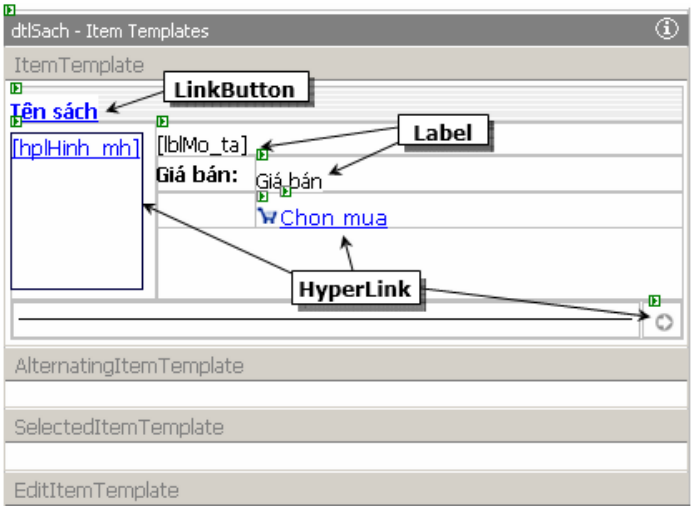
Thiết kế hình thức hiển thị cho DataList cũng tương tự như thiết kế cho cột Template Column của DataGrid.

Chọn Edit Template | ItemTemplate từ thực đơn ngữ cảnh để thực hiện thiết kế hình thức hiển thị cho DataList.



Chọn chức năng thiết kế cho DataList

Ví dụ: Hiển thị thông tin sách với DataList



Thiết kế thông tin sách với DataList

Như cột Template Column của DataGrid, xử lý hiển thị dữ liệu cho DataList được viết trong sự kiện **ItemDataBound.** Xử lý nhấn của các Button đặt trong DataList được viết trong sự kiện **ItemCommand**.

```
private void Page_Load()
{
        if (!IsPostBack) {
                Lien_ket_du_lieu();
        }
}
public void Lien_ket_du_lieu()
{
        dtSach = Doc_danh_sach_Sach();
        dtlSach.DataSource = dtSach;
        dtlSach.DataKeyField = "Ms";
        dtlSach.DataBind();
}
private void dtlSach_ItemDataBound()
{
        int lDong = e.Item.ItemIndex;
```

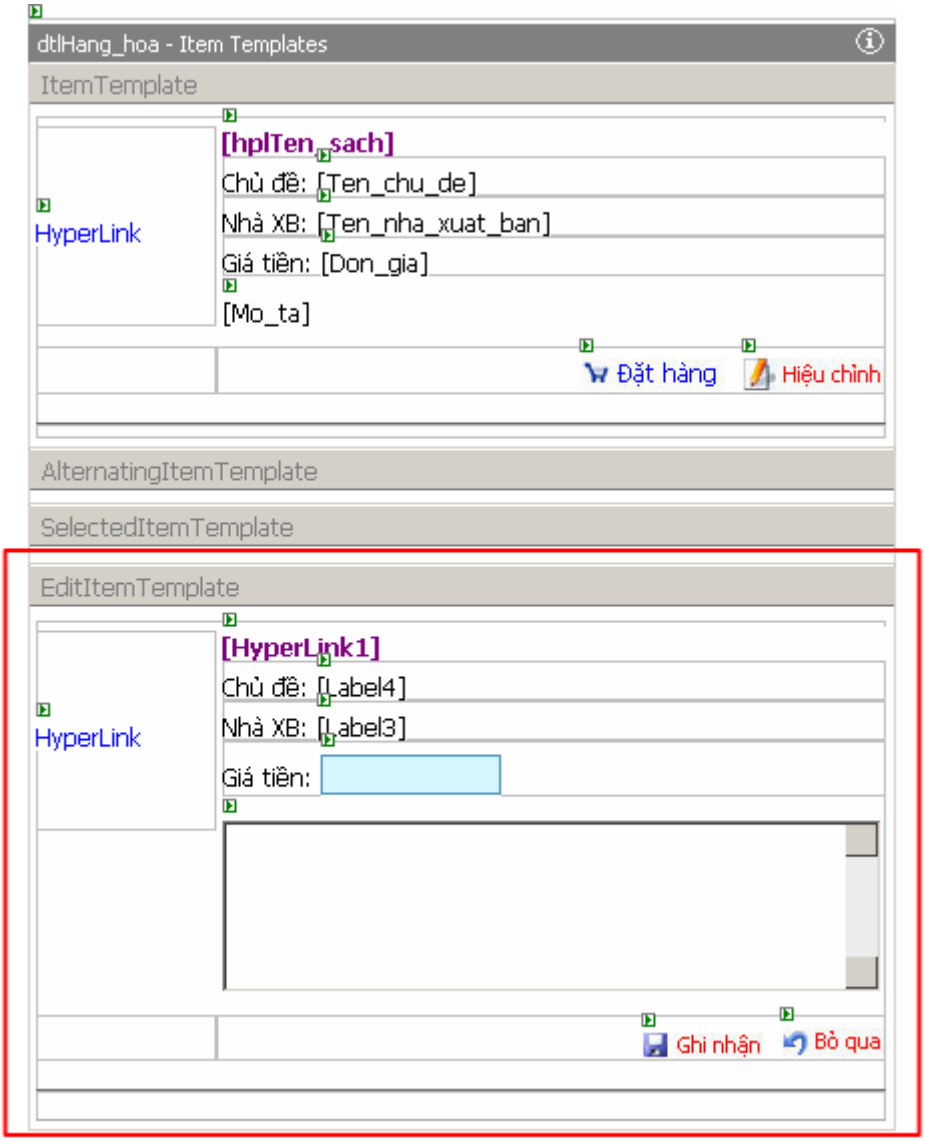
```
        if (lDong < 0)
            return;
        //Hiển thị Tên sách
        LinkButton lnkTs;
        lnkTs = e.Item.FindControl("lnkTen_sach");
        lnkTs.Text = e.Item.DataItem("Ten_sach");
        //Hiển thị thông tin mô tả tóm tắt nội dung
        Label lblMt;
        lblMt = e.Item.FindControl("lblMo_ta");
        lblMt.Text = Left(e.Item.DataItem("Mo_ta"), 200) + "...";
        //Hiển thị hình ảnh minh họa
        HyperLink hplHinh;
        hplHinh = e.Item.FindControl("hplHinh_mh");
        hplHinh.ImageUrl = "../Data_Pic/" + e.Item.DataItem("Hinh_minh_hoa");
        //Hiển thị giá bán sách
        Label lblGia;
        lblGia = e.Item.FindControl("lblGia_ban");
        lblGia.Text = e.Item.DataItem("Don_gia");
}
```

Kết quả hiển thị thông tin sách trên trang Web

Kết quả trên trang Web

2.2. Cập nhật dữ liệu với DataList

Ngoài việc hiển thị dữ liệu, DataList cũng hỗ trợ các thao tác cập nhật dữ liệu. Để thực hiện chức năng cập nhật dữ liệu với DataList, chúng ta cần phải thiết kế thêm vùng EditIemTemplate cho DataList. (xem hình)

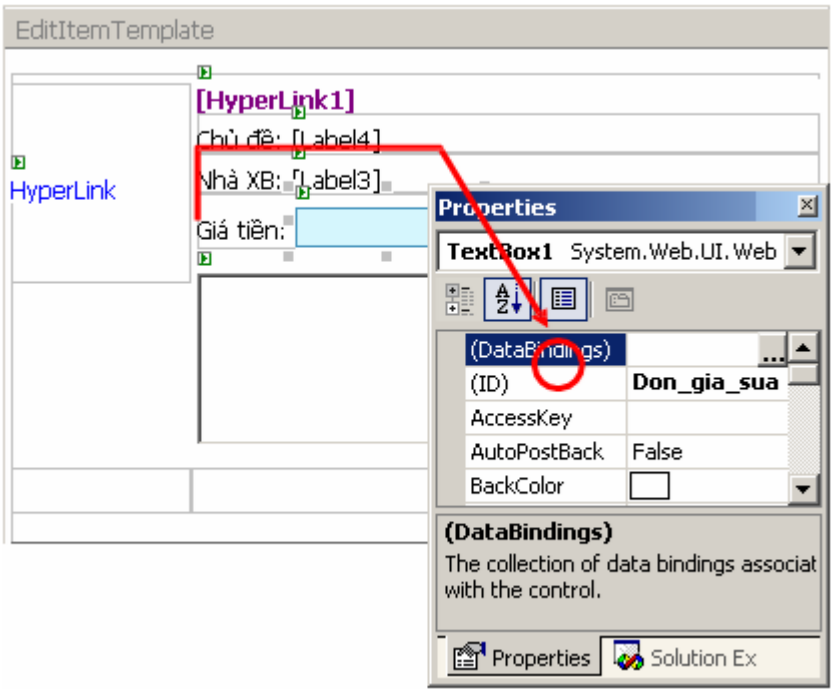


2.2.1. Các bước xử lý

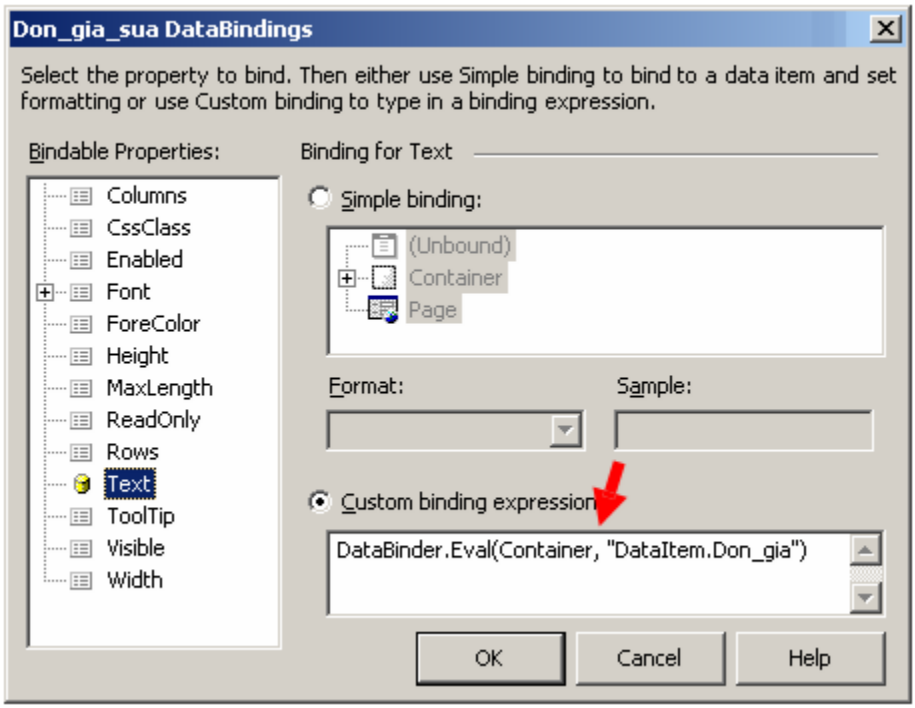
a. Thiết kế

Thiết kế cả 2 vùng ItemTemplate và EditItemTemplate. Thực hiện các thao tác liên kết dữ liệu cho các điều khiển trong vùng EditItemTemplate thông qua cửa sổ thuộc tính tương tự như trong ItemTemplate.

Chú ý: Chúng ta hoàn toàn có thể thực hiện việc liên kết dữ liệu trong sự kiện ItemDataBound.



Chọn chức năng DataBindings cho ô Đơn giá



Liên kết dữ liệu với cột Don_gia

Yêu cầu thiết kế

Tên điều khiển	Thiết lập thuộc tính
Hieu_chinh:	CommandName: Edit
Ghi_nhan: ImageButton	CommandName: Update CommandArgument:
Bo_qua: ImageButton	CommandName: Cancel

b. Xử lệnh để cập nhật dữ liệu

Xử lý các sự kiện EditCommand, CancelCommand, UpdateCommand để thực hiện/bỏ qua việc thay đổi dữ liệu.

Mã lệnh xử lý:

```
private void Page_Load()
{
        //Put user code to initialize the page here
        if (!IsPostBack) {
                Lien_ket_du_lieu();
        }
}
private void dtlHang_hoa_EditCommand()
{
        dtlHang_hoa.EditItemIndex = e.Item.ItemIndex;
        Lien_ket_du_lieu();
}
private void dtlHang_hoa_CancelCommand()
{
        dtlHang_hoa.EditItemIndex = -1;
        Lien_ket_du_lieu();
}
private void dtlHang_hoa_UpdateCommand()
{
        //Xử lý cập nhật dữ liệu tại đây
        TextBox Don_gia_sua;
        Don_gia_sua = e.Item.FindControl("Don_gia_sua");
        //Don_gia_sua.Text Î Trả về đơn giá mới được sửa
```

```
        //……

        dtlHang_hoa.EditItemIndex = -1;
        Lien_ket_du_lieu();
}
```

DataList ở chế độ đang hiệu chỉnh

3. Điều khiển Repeater

Như 2 điều khiển DataList & DataGrid, điều khiển Repeater cũng được dùng để hiển thị dữ liệu. Tuy nhiên, để hiển thị dữ liệu, chúng ta phải tự thiết kế hình thức hiển thị thông qua các tag HTML. Điều khiển Repeater có các tag sau:

<HeaderTemplate></HeaderTemplate> (tùy chọn)

Qui định hình thức hiển thị cho tiêu đề. (Chỉ xuất hiện 1 lần, phía trên của điều khiển)

<ItemTemplate></ItemTemplate> (Bắt buộc phải có)

Qui định hình thức hiển thị cho các mục dữ liệu trong điều khiển.

<AlternatingItemTemplate></AlternatingItemTemplate> (tùy chọn)

Qui định hình thức hiển thị cho các mục dữ liệu trong điều khiển. Nội dung được qui định trong cặp tag này sẽ hiển thị xen kẽ với các nội dung trong cặp tag

<ItemTemplate></ItemTemplate>

Qui định hình thức hiển thị giữa các dòng dữ liệu

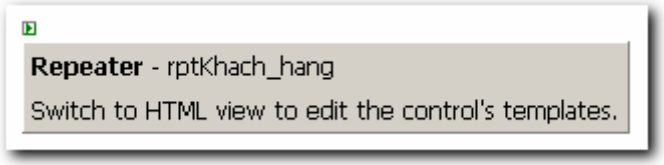
<SeparatorTemplate></SeparatorTemplate> (tùy chọn)

Qui định hình thức hiển thị cho tiêu đề dưới. (Chỉ xuất hiện 1 lần, phía dưới của điều khiển)

<FooterTemplate></FooterTemplate> (tùy chọn)

Ví dụ:

Bước 1. Tạo mới điều khiển Repeater: rptKhach_hang vào trang Web.



Điều khiển rptKhach_hang trên trang Web

Bước 2. Chuyển qua xem trang Web dưới dạng HTML

Bước 3. Bổ sung các tag sau

Bước 4. Xem lại màn hình thiết kế

Họ khách hàng	Tên khách hàng	Địa chi	Điện thoại
Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound

Bước 5. Tạo nguồn dữ liệu cho điều khiển

Bước 6. Thi hành ứng dụng

Họ khách hàng	Tên khách hàng	Địa chi	Điện thoại
Hoàng Chinh	Thảo	118 Nhân Chính	0918062754
Mai Giang	Giang	85/6 Lê Quang Sung	Chưa có
Bùi Thị Mộng	Ánh	97/92/10 Lê Anh Xuân	Chưa có
Trịnh Linh	Tâm	17 Nguyễn Thái Học	0918544698
Nguyễn Giang	Thắng	45 Nguyễn Thị Tần	0908123456

4. Các ví dụ mở rộng

4.1. Xử lý đảo hướng sắp xếp trong DataGrid

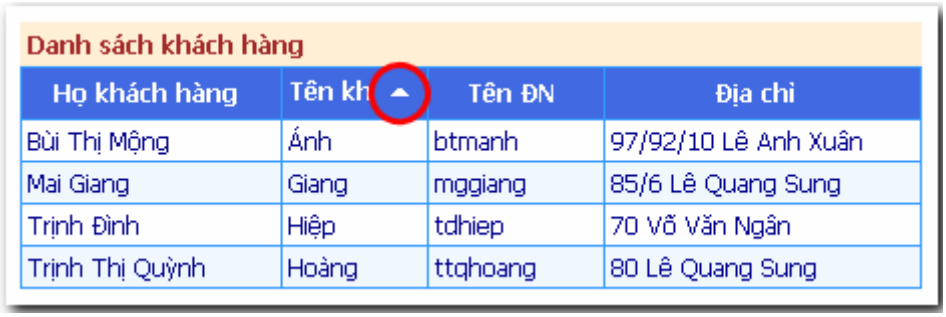
Ví dụ minh họa dưới đây xử lý đảo hướng sắp xếp trong DataGrid. Đồng thời, trong ví dụ này, chúng tôi thực hiện liên kết dữ liệu qua đối tượng DataView để thực hiện sắp xếp trên nguồn dữ liệu.

```
private void Page_Load()
{
if (!IsPostBack)
        Lien_ket_du_lieu();
}
public void Lien_ket_du_lieu()
{
        DataTable dtKhach_hang = Doc_ds_khach_hang();
```

```
        DataView dvKhach_hang = new DataView(dtKhach_hang);
        dvKhach_hang.Sort = ViewState("SortExpression");
        if (ViewState("SortAscending") == "false") {
                dvKhach_hang.Sort += " desc";
        }
        dtgKhach_hang.DataSource = dvKhach_hang;
        dtgKhach_hang.DataBind();
}
public DataTable Doc_ds_khach_hang()
{
        string sKet_noi;
        sKet_noi = "Provider=Microsoft.Jet.Oledb.4.0;Data Source=" +
Server.MapPath("..\\Data\\QlBanSach.mdb");
        OleDbConnection cnKet_noi = new OleDbConnection(sKet_noi);
        DataSet dsCSDL = new DataSet();
        //Mở và đóng kết nối ngay khi thực hiện xong
        cnKet_noi.Open();
        OleDbDataAdapter daBo_doc_ghi = new OleDbDataAdapter("Select *
From KHACH_HANG", cnKet_noi);
        cnKet_noi.Close();
        daBo_doc_ghi.Fill(dsCSDL, "KHACH_HANG");
        return dsCSDL.Tables("KHACH_HANG");
}
private void dtgKhach_hang_SortCommand()
{
        string sSap_xep = ViewState("SortExpression");
        string sHuong = ViewState("SortAscending");
        ViewState("SortExpression") = e.SortExpression;
```

```
        if ((e.SortExpression == sSap_xep)) {
            ViewState("SortAscending") = IIf(sHuong == "false", "true", "false");
        }
        Lien_ket_du_lieu();
}
```

4.2. Tạo biểu tượng sắp xếp trong cột cho DataGrid



Danh sách khách hàng

Họ khách hàng	Tên kh	Tên ĐN	Địa chỉ
Bùi Thị Mộng	Ánh	btmanh	97/92/10 Lê Anh Xuân
Mai Giang	Giang	mggiang	85/6 Lê Quang Sung
Trịnh Đình	Hiệp	tdhiep	70 Võ Văn Ngân
Trịnh Thị Quỳnh	Hoàng	ttqhoang	80 Lê Quang Sung

Sắp xếp tăng dần theo tên khách hàng

```
private void dtgKhach_hang_ItemDataBound()
{
  if (e.Item.ItemType == ListItemType.Header) {
        string sSap_xep = ViewState("SortExpression");
        string sHuong = ViewState("SortAscending");
        string sKy_hieu = IIf(sHuong == "false", " 6", " 5");
        object i;
        for (i = 0; i <= dtgKhach_hang.Columns.Count - 1; i++) {
            if (sSap_xep == dtgKhach_hang.Columns(i).SortExpression) {
                TableCell cell = e.Item.Cells(i);
                Label lblKy_hieu = new Label();
                lblKy_hieu.Text = sKy_hieu;
                lblKy_hieu.Font.Name = "webdings";
                lblKy_hieu.Font.Size = FontUnit.XSmall;
                cell.Controls.Add(lblKy_hieu);
            }
```

```
        }
    }
}
```

4.3. Định dạng hình thức hiển thị cho dòng dữ liệu thỏa điều kiện trên DataGrid

Trong ví dụ sau, chúng ta thực hiện tô màu cho những khách hàng có tên bắt đầu bằng ký tự **H**.

```
private void dtgKhach_hang_ItemDataBound()
{
        if (e.Item.ItemIndex < 0)
                return;
        string sTen_kh;
        sTen_kh = e.Item.DataItem("Ten_khach_hang");
        //Tiến hành kiểm tra điều kiện,
        //nếu thỏa  → thực hiện các xử lý định dạng
        if (sTen_kh.StartsWith("H")) {
                e.Item.BackColor = Color.LemonChiffon;
                e.Item.Cells(1).Font.Bold = true;
        }
}
```

Danh sách khách hàng

Họ khách hàng	Tên kh	Tên đ.nhập	Địa chỉ
Bùi Thị Mộng	Ánh	btmanh	97/92/10 Lê Anh Xuân
Mai Giang	Giang	mggiang	85/6 Lê Quang Sung
Trịnh Đình	**Hiệp**	tdhiep	70 Võ Văn Ngân
Trịnh Thị Quỳnh	**Hoàng**	ttqhoang	80 Lê Quang Sung
Cung Kiên	**Hưng**	ckhung	96 Nguyễn Bỉnh Khiêm
Phạm Bông	Lực	pbluc	153/9 Bình Tây
Trịnh Linh	Tâm	tltam	17 Nguyễn Thái Học

Tô màu những khách hàng có tên bắt đầu bằng ký tự H

4.4. Tạo hiệu ứng chọn khi rê chuột qua các dòng dữ liệu

```
private void dtgKhach_hang_ItemDataBound()
{
        if (e.Item.ItemIndex < 0)
                return;
        e.Item.Attributes("onMouseOver")                                    =
"this.style.backgroundColor='#FFF8DC'";
        e.Item.Attributes("onMouseOut") = "this.style.backgroundColor="";
}
```

Tạo hiệu ứng chọn dòng dữ liệu trên lưới

BÀI 4. XÂY DỰNG LỚP XỬ LÝ DỮ LIỆU

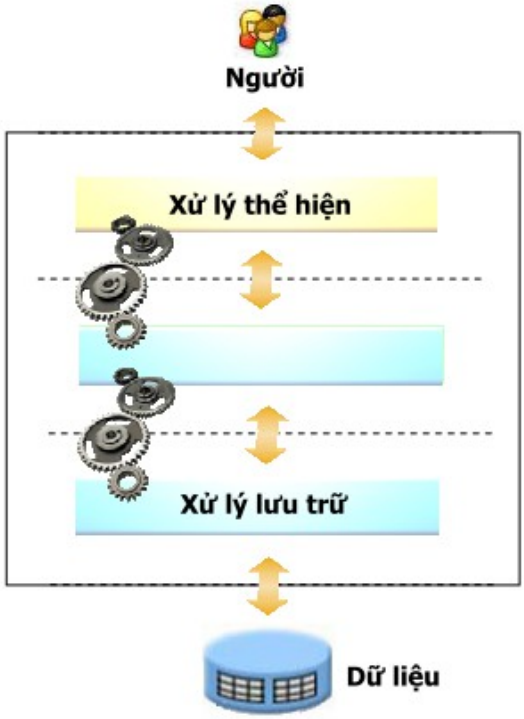
A. Mục tiêu

Dựa trên những kiến thức đã học về lập trình hướng đối tượng và thao tác dữ liệu với ADO.NET, xây dựng lớp xử lý dữ liệu.

B. Nội dung

Chỉ mỗi việc hiển thị dữ liệu trên trang Web, chúng ta đã viết khá nhiều dòng lệnh trên đó, chưa kể đến những xử lý khác sau này. Số lượng thao tác cần xử lý trên trang Web càng tăng, số lượng dòng lệnh càng nhiều. Việc để lẫn lộn những đoạn code về truy cập dữ liệu và xử lý trên giao diện gây không ít khó khăn trong việc xây dựng, phát triển và bảo trì ứng dụng web.

Chính vì lý do đó, trong phần này, chúng tôi hướng dẫn các bạn xây dựng lớp xử lý và đối tượng thể hiện dữ liệu. Lớp xử lý đảm nhận trách nhiệm thực hiện các thao tác truy xuất và cập nhật dữ liệu. Đối tượng thể hiện nhận dữ liệu, hiển thị dữ liệu trên trang Web và tiếp nhận thông tin từ người dùng. Việc phân chia công việc cụ thể cho từng đối tượng không những giúp cho chúng ta xây dựng và phát triển ứng dụng một cách có hiệu quả mà còn dễ dàng trong quá trình bảo trì, phù hợp với xu hướng phát triển phần mềm sử dụng các ngôn ngữ lập trình hiện đại.



Mô hình xử lý của ứng dụng

1. Thiết kế tổng quan

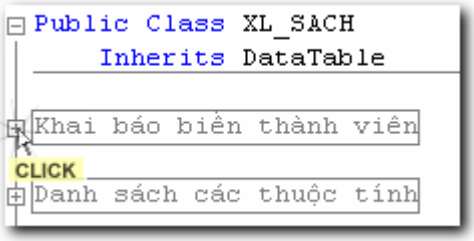
Để giúp các bạn dễ dàng theo dõi cấu trúc chi tiết của lớp xử lý lưu trữ dữ liệu (XL_BANG), chúng ta sẽ bắt đầu tìm hiểu thiết kế tổng quan của nó. Như tên gọi của nó, lớp xử lý lưu trữ (XL_BANG) thực hiện các chức năng:

- Truy xuất dữ liệu từ cơ sở dữ liệu
- Thực hiện các câu lệnh Sql

```
using System.Data;
using System.Data.OleDb;
static class PHAN_MEM
{
  public const string Chuoi_lien_ket = "Provider=Microsoft.JET.OLEDB.4.0;";
}
public class XL_BANG : DataTable
{
  #region "Khai báo biến thành viên"
  #endregion
```

```
 #region "Danh sách các thuộc tính"
 #endregion
 #region "Nhóm hàm khởi tạo đối tượng"
 #endregion
 #region "Nhóm hàm cung cấp thông tin"
 #endregion
 #region "Nhóm hàm xử lý tính toán"
 #endregion
 #region "Xử lý sự kiện"
 #endregion
}
```

Nhóm từ khóa #Region và #End Region tạo ra các phân vùng giúp chúng ta dễ dàng quản lý các đoạn lệnh trong quá trình xây dựng lớp.



Phân vùng với Region

Như các bạn thấy, lớp XL_BANG được kế thừa từ lớp DataTable, đồng nghĩa với việc nó sẽ được thừa hưởng tất cả những "tài sản" (các thuộc tính, phương thức, …) từ lớp DataTable.

Trong lớp xử lý trên, chúng ta có thực hiện các thao tác truy xuất và cập nhật dữ liệu, do đó, chúng ta cần sử dụng bộ thư viện của ADO.Net. Bộ thư viện được sử dụng trong lớp xử lý này là System.Data.OleDb. Lẽ đương nhiên, khi xây dựng ứng dụng của riêng mình, tùy theo yêu cầu cụ thể của ứng dụng, các bạn hoàn toàn có thể chọn cho mình bộ thư viện khác, như System.Data.SqlClient chẳng hạn.

Chúng tôi chia các khai báo, xử lý thành 6 nhóm:

Khai báo biến thành viên: Khai báo các biến thành viên của lớp đối tượng trong nhóm này.

Danh sách các thuộc tính: Để đảm bảo tính bao bọc của phương pháp lập trình hướng đối tượng, chúng ta sử dụng các thuộc tính để thực hiện giao tiếp giữa biến thành viên với các xử lý bên ngoài lớp.

Nhóm hàm khởi tạo đối tượng: Danh sách các hàm khởi tạo lớp đối tượng được thực hiện trong nhóm này.

Nhóm hàm cung cấp thông tin: Các hàm cung cấp thông tin có được từ lớp đối tượng.

Nhóm hàm xử lý tính toán: Các hàm, thủ tục, thực hiện các xử lý, tính toán theo yêu cầu phục vụ cho việc cung cấp thông tin, vận hành lớp đối tượng.

Xử lý sự kiện: Các thủ tục xử lý sự kiện của lớp đối tượng

1.1. Cấu trúc chi tiết lớp XL_BANG

1.1.1. Khai báo biến thành viên

1.1.2. Danh sách các thuộc tính

Ứng với mỗi biến thành viên cần giao tiếp ra bên ngoài, chúng ta cung cấp thuộc tính tương ứng để làm việc.

1.1.3. Nhóm hàm khởi tạo đối tượng

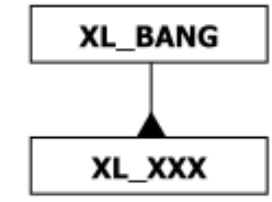
1.1.4. Nhóm hàm cung cấp thông tin

1.1.5. Nhóm hàm xử lý tính toán

1.1.6. Nhóm hàm xử lý sự kiện

1.2. Xây dựng lớp xử lý nghiệp vụ

Dựa trên lớp xử lý lưu trữ (XL_BANG), xây dựng các lớp xử lý nghiệp vụ ứng với mỗi bảng trong cơ sở dữ liệu (CSDL).



Sơ đồ lớp XL_XXX

Trong đó:

Lớp XL_BANG: Đã được xây dựng ở phần trên.

Ký hiệu XXX: Tên các bảng tương ứng trong CSDL.

Các lớp xử lý nghiệp vụ sẽ có mẫu chung như XL_CHU_DE mẫu dưới đây.

Cấu trúc bảng chủ đề (CHU_DE)

Field Name	Field Type	Field Size	Description
Mcd	Autonumber	Long Integer	
Ten_chu_de	Text	50	
Ghi_chu	Text	200	

Chi tiết lớp XL_CHU_DE

1.3. Sử dụng lớp xử lý nghiệp vụ

Sau khi thiết kế lớp xử lý hoàn tất, việc hiển thị dữ liệu trên màn hình bây giờ khá đơn giản. Xem các ví dụ minh họa sau:

Hiển thị các sách có trong bảng SACH

Nếu muốn hiển thị các sách mới nhập:

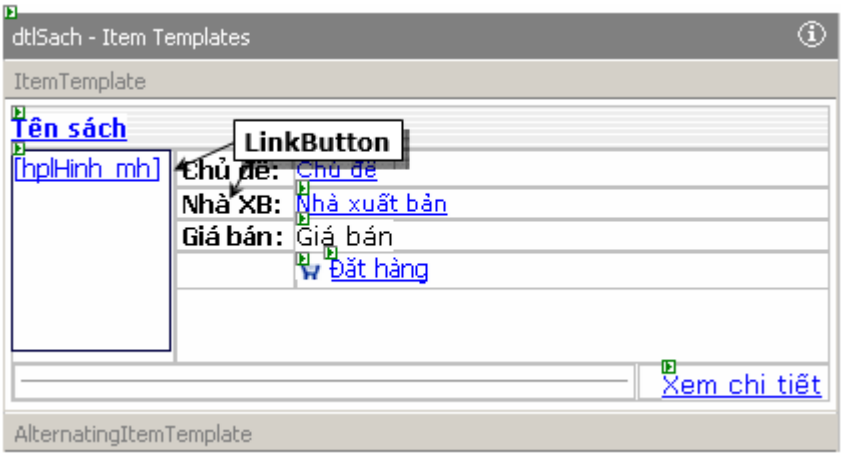
Nếu muốn hiển thị 10 cuốn sách mới nhất:

Nếu muốn hiển thị 10 cuốn sách mới nhất của chủ đề có mã là 5:

Kết quả hiển thị: (10 sách mới nhất)

Màn hình thông tin sách

Trong bảng SACH, chúng ta chỉ có thông tin về Mã chủ đề (Mcd), Mã nhà xuất bản (Mnxb), để lấy được tên chủ đề, tên nhà xuất bản như kết quả trong hình minh họa trên, chúng ta cần bổ sung các điều khiển cần thiết như hình bên dưới và thực hiện như sau:



Hiệu chỉnh thủ tục Lien_ket_du_lieu như sau:

BÀI 5. XÂY DỰNG ĐỐI TƯỢNG THỂ HIỆN

A. Mục tiêu

Thiết kế và xây dựng điều khiển người dùng (Web user control)

B. Nội dung

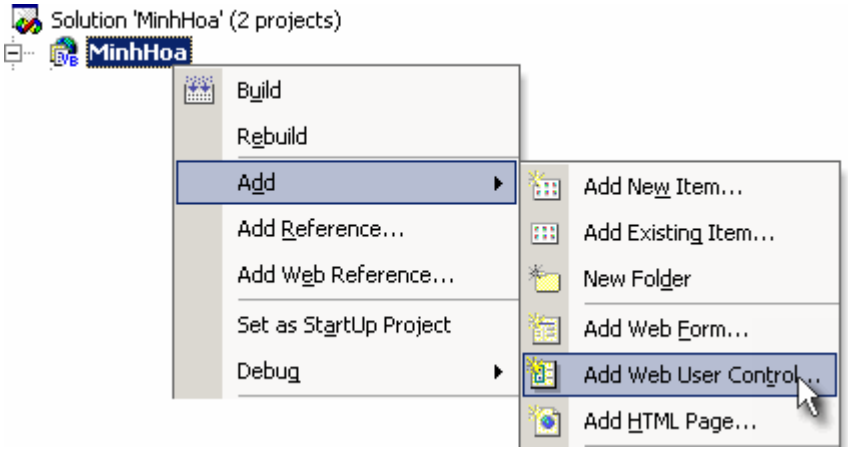
Xây dựng điều khiển người dùng - user control rất cần thiết cho việc tái sử dụng các đoạn mã lệnh mà ta đã xây dựng chúng, đây là một trong những tiêu chí quan trọng trong lĩnh vực xây dựng phần mềm nói chung và xây dựng ứng dụng web nói riêng.

Xây dựng điều khiển người dùng cũng tương tự như việc xây dựng các trang web mà chúng ta đã làm trước đây. Chỉ có điều khác biệt chính là trong trang web, chúng ta có nhiều thành phần giao diện và xử lý tương ứng còn trong điều khiển người dùng, chúng ta chỉ thiết kế và xây dựng cho một chức năng hay yêu cầu cụ thể.

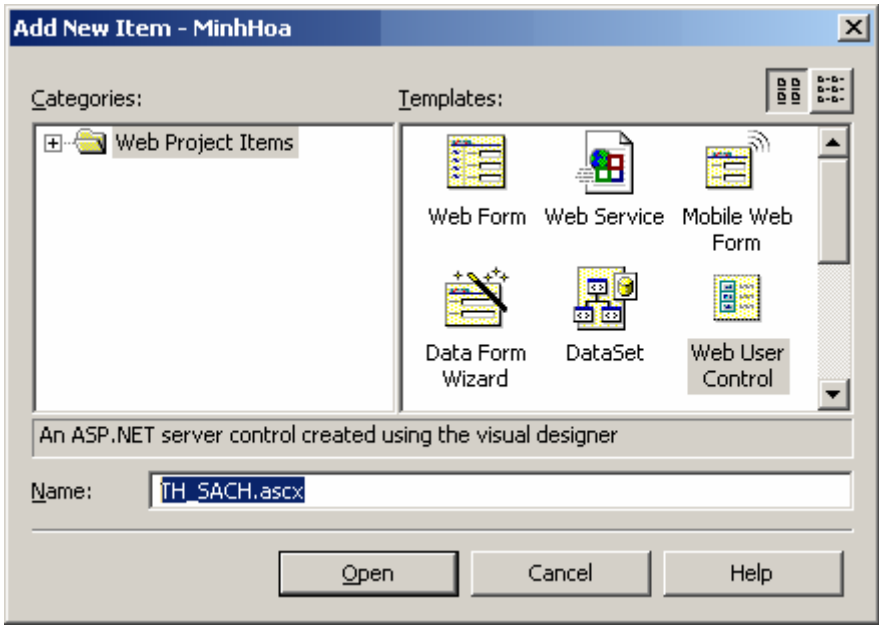
Điều khiển người dùng cũng có các thuộc tính, phương thức và sự kiện như các Web Server control, lẽ đương nhiên là các thuộc tính, phương thức và sự kiện đều do chúng ta thiết kế và xây dựng.

1. Tạo mới đối tượng thể hiện

Để tạo mới đối tượng thể hiện, chọn Add | Add Web User Control… từ thực đơn ngữ cảnh của ứng dụng.

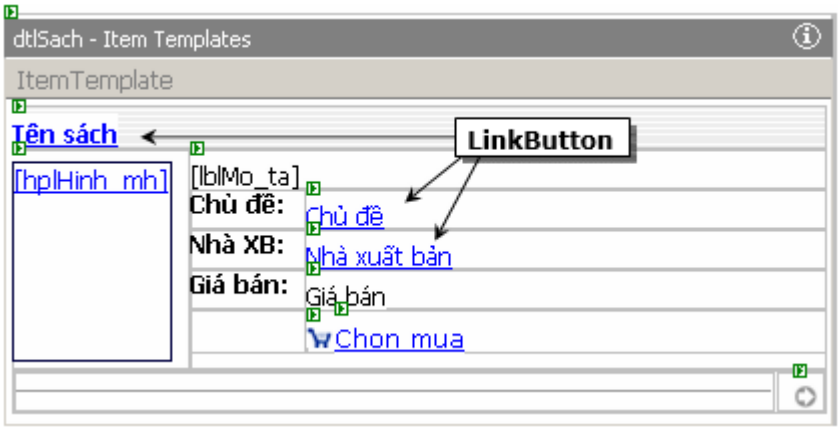


Chức năng tạo mới đối tượng thể hiện – Web User Control



Tạo mới đối tượng thể hiện: TH_SACH

Việc thiết kế và xây dựng các đối tượng thể hiện hoàn toàn tương tự như các bạn đã từng làm với trang web. Chúng ta cùng xây dựng đối tượng thể hiện thông tin sách.



Thiết kế thể hiện thông tin sách sử dụng DataList

Khi hoàn tất thiết kế đối tượng thể hiện, công việc tiếp theo là viết các xử lý cần thiết cho các điều khiển có trên đối tượng thể hiện theo yêu cầu sử dụng.

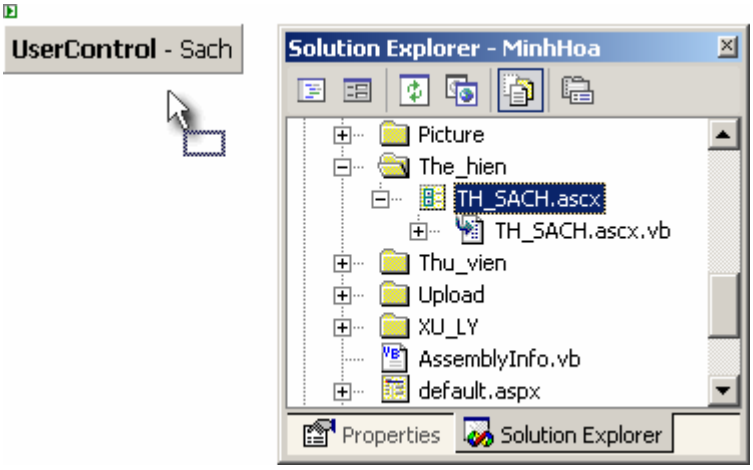
Viết các xử lý cho đối tượng thể hiện:

2. Sử dụng đối tượng thể hiện

Sau khi thiết kế và viết các xử lý, chúng ta tiến hành đưa đối tượng thể hiện đã được xây dựng vào trang Web. Các bước thực hiện:

Bước 1. Mở trang web ở chế độ thiết kế - design.

Bước 2. Từ cửa sổ **Solution Explorer**, chọn đối tượng thể hiện cần sử dụng, nhấn và kéo rê vào trang web đã được mở.



Kéo đối tượng thể hiện vào trang Web

Bước 3. Thiết lập các thuộc tính cho điều khiển vừa được kéo vào Khi thi hành, kết quả hiển thị của đối tượng thể hiện trên trang Web:

3. Tạo phương thức cho đối tượng thể hiện

Trong ví dụ trên, chúng ta đã xây dựng đối tượng thể hiện Sách. Khi tạo mới một thể hiện Sách vào trang web, thông tin sách sẽ được hiển thị. Tuy nhiên, chắc hẳn các bạn sẽ hài lòng hơn khi chúng ta thiết kế đối tượng thể hiện sách: TH_SACH, chỉ với 1 đối tượng, nhưng chúng ta có thể hiển thị thông tin sách theo yêu cầu như: Hiển thị sách mới vừa nhập, hiển thị sách bán chạy nhất, hiển thị sách được nhiều đọc giả xem và bình chọn nhất, hiển thị sách của một nhà xuất bản hay hiển thị thông tin sách của một tác giả nào đó. Thú vị quá phải không các bạn?

Để làm được điều đó, rất đơn giản. Chúng ta chỉ việc tạo cho đối tượng thể hiện các phương thức - hành vi tương ứng với những yêu cầu cụ thể.

Chúng ta sẽ tiến hành bổ sung các phương thức sau vào đối tượng thể hiện vừa được xây dựng.

Lưu ý: Khi bổ sung các phương thức hiển thị dữ liệu cho đối tượng thể hiện, chúng ta không xử lý hiển thị dữ liệu trong sự kiện PageLoad.

4. Tạo sự kiện cho đối tượng thể hiện

Chắc các bạn không quên sự kiện Click của các điều khiển Button (Button, LinkButton, ImageButton). Sự kiện Click xảy ra khi Button được nhấn vào. Và mới đây thôi, với điều khiển DataGrid, DataList, chúng ta đã làm việc với các sự kiện: ItemCommand, EditCommand, UpdateCommand, … Mỗi sự kiện xảy ra bởi một hành động tương ứng trước đó của người dùng.

Các đối tượng thể hiện mà chúng ta vừa xây dựng cũng vậy, có khả năng phát ra các sự kiện nếu được chúng ta xây dựng.

Chúng ta cùng tạo sự kiện cho thể hiện Sách. Trong thể hiện sách có các thông tin mô tả liên quan: Tên sách, Chủ đề, Nhà xuất bản. Khi người dùng chọn chức năng nào thì điều khiển sẽ phát ra sự kiện tương ứng:

- Tên sách → Điều khiển sẽ phát ra sự kiện **Chon_sach(pMs)**
- Trong đó: pMs là Mã sách được người dùng chọn.
- Chủ đề → Điều khiển sẽ phát ra sự kiện **Chon_chu_de(pMcd)**
- Trong đó: pMcd là Mã chủ đề được người dùng chọn.
- Nhà xuất bản → Điều khiển sẽ phát ra sự kiện

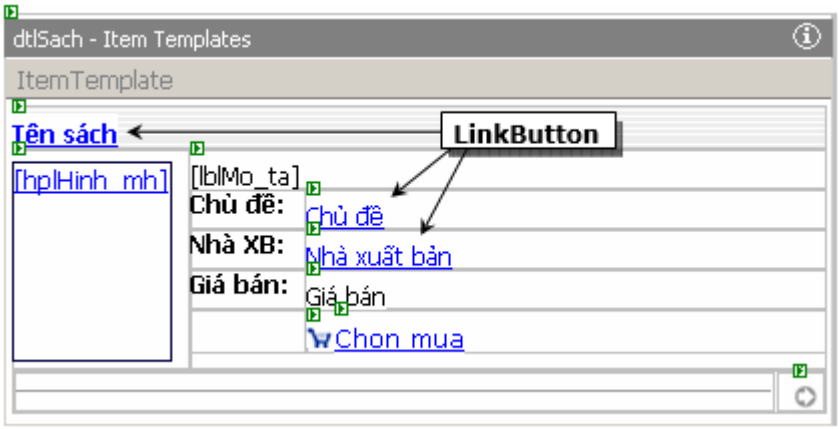
Chon_nha_xuat_ban(pMnxb)

Trong đó: pMnxb là Mã nhà xuất bản được người dùng chọn.

Điều khiển phát ra các sự kiện tương ứng

4.1. Thiết kế

Thiết lập thuộc tính **CommandName** cho các LinkButton: Tên sách, Chủ đề và Nhà xuất bản.



Thiết kế thông tin sách với DataList

Bảng mô tả thuộc tính của các điều khiển

Điều khiển	Loại	Thuộc tính	Giá trị
lnkTen_sach	LinkButton	Text	Tên sách
		CommandName	**Ten_sach**
lnkChu_de	LinkButton	Text	Chủ đề

		CommandName	**Chu_de**
lnkNha_xb	LinkButton	Text	Nhà xuất bản
		CommandName	**Nha_xuat_ban**

4.2. Xử lý

Bước 1. Khai báo các sự kiện

```
using System.Web.UI.WebControls;
public class TH_SACH : System.Web.UI.UserControl
{
    public event Chon_sachEventHandler Chon_sach;
    public delegate void Chon_sachEventHandler(long pMs);
    public event Chon_chu_deEventHandler Chon_chu_de;
    public delegate void Chon_chu_deEventHandler(long pMcd);
    public event Chon_nha_xuat_banEventHandler Chon_nha_xuat_ban;
    public delegate void Chon_nha_xuat_banEventHandler(long pMnxb);
}
```

Bước 2. Xử lý sự kiện **ItemDataBound**

Trong xử lý sau, chúng ta gán giá trị cho thuộc tính CommandArgument của các LinkButton để lưu trữ các mã tương ứng cho từng điều khiển.

```
//Xử lý cho Tên sách
LinkButton lnkTen_sach;
lnkTen_sach = e.Item.FindControl("lnkTen_sach");
lnkTen_sach.Text = e.Item.DataItem("Ten_sach");
lnkTen_sach.CommandArgument = e.Item.DataItem("Ms");

//Xử lý cho chủ đề
XL_CHU_DE lChu_de = new XL_CHU_DE();
LinkButton lnkChu_de;
lnkChu_de = e.Item.FindControl("lnkChu_de");
int Mcd = e.Item.DataItem("Mcd");
```

```
lnkChu_de.Text = lChu_de.Thuoc_tinh(Mcd, "Ten_chu_de");
lnkChu_de.CommandArgument = Mcd;

//Xử lý cho nhà xuất bản
XL_NHA_XB lNXB = new XL_NHA_XB();
LinkButton lnkNha_xb;
lnkNha_xb = e.Item.FindControl("lnkNha_xb");
int Mnxb = e.Item.DataItem("Mnxb");
lnkNha_xb.Text = lNXB.Thuoc_tinh(Mnxb, "Ten_nha_xuat_ban");
lnkNha_xb.CommandArgument = MNXB;
```

Bước 3. Bẫy biến cố **ItemCommand** để phát sự kiện tương ứng

Hiển thị sách theo chủ đề **Công nghệ thông tin**

```
private void dtlSach_ItemCommand()
{
        if (e.CommandName == "Ten_sach") {
                long Ms = e.CommandArgument;
                if (Chon_sach != null) {
                        Chon_sach(Ms);
                }

        } else if (e.CommandName == "Chu_de") {
                long Mcd = e.CommandArgument;
                if (Chon_chu_de != null) {
                        Chon_chu_de(Mcd);
                }

        } else if (e.CommandName == "Nha_xuat_ban") {
                long Mnxb = e.CommandArgument;
                if (Chon_nha_xuat_ban != null) {
```

```
                Chon_nha_xuat_ban(Mnxb);
            }
        }
}
```

Chúng ta có thể đồng thời vừa xử lý biến cố và phát ra sự kiện:

```
private void dtlSach_ItemCommand()
{
        if (e.CommandName == "Ten_sach") {
            long Ms = e.CommandArgument;
            if (Chon_sach != null) {
                Chon_sach(Ms);
            }

        } else if (e.CommandName == "Chu_de") {
            long Mcd = e.CommandArgument;
            Hien_thi_sach_theo_chu_de(Mcd);
            if (Chon_chu_de != null) {
                Chon_chu_de(Mcd);
            }

        } else if (e.CommandName == "Nha_xuat_ban") {
            long Mnxb = e.CommandArgument;
            Hien_thi_sach_theo_nxb(Mnxb);
            if (Chon_nha_xuat_ban != null) {
                Chon_nha_xuat_ban(Mnxb);
            }
        }
}
```

Hiển thị sách theo chủ đề **Công nghệ thông tin**

Các sách của chủ đề Công nghệ thông tin

BÀI 6. XÂY DỰNG VÀ QUẢN LÝ ỨNG DỤNG

A. Mục tiêu

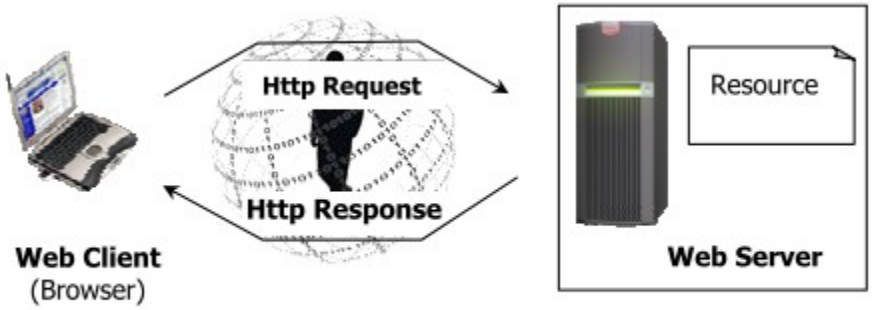
- Sử dụng các đối tượng quản lý ứng dụng để thực hiện các chức năng điều khiển luồng chương trình, quản lý người dùng hay chia sẻ dữ liệu giữa các trang web.
- Tìm hiểu các tập tin quản lý và cấu hình ứng dụng.
- Tổ chức và xây dựng ứng dụng.

B. Nội dung

Trong các bài trước, chúng ta đã tìm hiểu và làm việc với các điều khiển, xử lý dữ liệu với ADO.Net, tạo các lớp xử lý và xây dựng các đối tượng thể hiện, …. đó là những kỹ năng cần thiết để xây dựng ứng dụng.

Trong bài này, chúng ta sẽ tìm hiểu các đối tượng được dùng để xây dựng, phát triển và quản lý ứng dụng web. Thông qua những đối tượng này, chúng ta có thể ghi nhận những yêu cầu từ Client, quản lý thông tin người dùng, cấu hình và bảo mật cho ứng dụng.

1. Đối tượng Request, Response



Quá trình Request - Response của HTTP

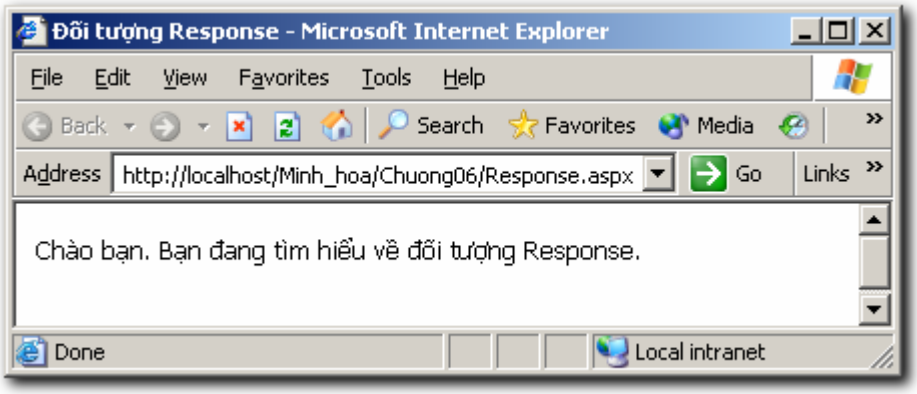
1.1. Đối tượng Response

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.

1.1.1. Phương thức Write

Phương thức Write của đối tượng Response được dùng để in ra một chuỗi trên trang Web. Phương thức này là một trong những phương thức chủ lực trong

các ứng dụng web sử dụng ASP 3.0 khi cần gởi kết quả từ Server về cho Client.



Trong ASP.Net, chúng ta có thể thực hiện như sau:

Qua ví dụ trên, chắc có lẽ bạn cũng nhận ra rằng, khi sử dụng phương thức Response.Write, chúng ta không thể qui định vị trí hiển thị của chuỗi trên trong trang Web. Thay vào đó, với ASP.Net, thông qua các Server control, chúng ta có thể thực hiện chức năng tương tự nhưng linh hoạt hơn bằng cách đặt điều khiển tại vị trí cần hiển thị.

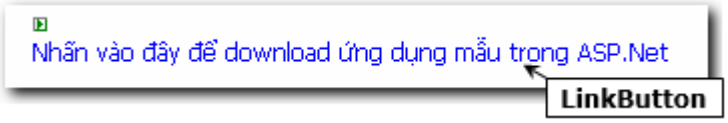
1.1.2. Phương thức Redirect

Phương thức Redirect gởi thông điệp yêu cầu Web Browser truy cập đến một địa chỉ khác.

Ví dụ

1.1.3. Ví dụ xử lý cho phép người dùng download file

Ví dụ: Xử dụng đối tượng Response để thực hiện việc download tập tin.

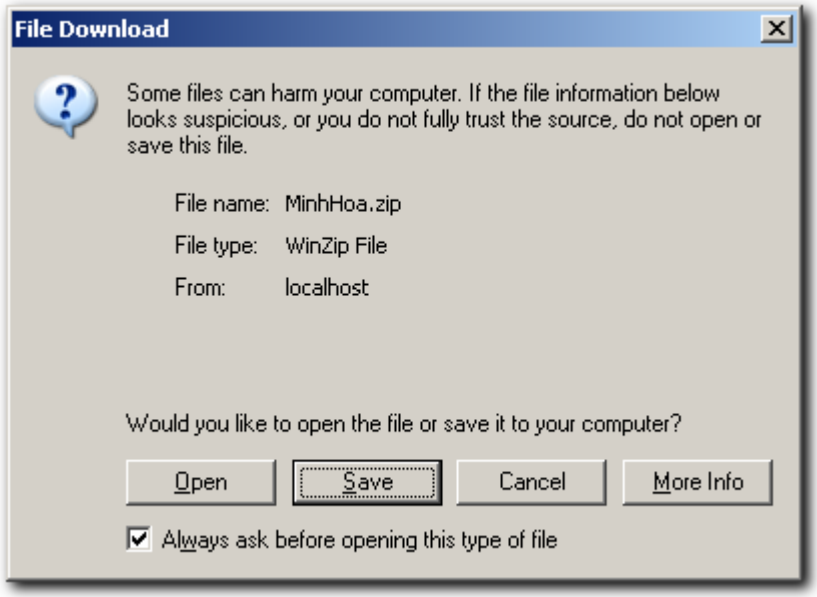


Thiết kế

Thi hành

Viết lệnh xử lý:

```
private void lnkDownload_Click()
{
        string sTap_tin = "MinhHoa.zip";
        string sDuong_dan;
        sDuong_dan = Server.MapPath("../Download/") + sTap_tin;
        Response.AddHeader("Content-Disposition", "attachment; filename=" + sTap_tin);
        Response.WriteFile(sDuong_dan);
        Response.End();
}
```



Hiển thị hộp thoại download tập tin

1.2. Đối tượng Request

Đối tượng Request được dùng để nhận thông tin từ trình duyệt của người dùng gởi về cho Web Server.

1.2.1. Thuộc tính QueryString

Như chúng tôi đã trình bày ở phần đầu của cuốn sách này, HTTP được xác định qua URLs (Uniform Resource Locators), với cấu trúc chuỗi có định dạng như sau:

Phần cuối của chuỗi URL là QueryString - còn được gọi là chuỗi tham số, có cấu trúc như sau:

Trong trường hợp có nhiều tham số, các cặp [<Tham_so> = <Gia_tri>] phân cách nhau bằng dấu &.

Ví dụ:

Thuộc tính QueryString của đối tượng Request cho phép chúng ta nhận các giá trị truyền qua chuỗi tham số này.

Ví dụ: Giả sử một người dùng gởi thông điệp đến Web Server yêu cầu trang: "**Request.aspx?Chuc_nang=Hieu_chinh&ID=123**". Để lấy giá trị 2 tham số trong chuỗi QueryString, chúng ta thực hiện như sau:

```
string sChuc_nang;
sChuc_nang = Request.QueryString("Chuc_nang");
lblChuc_nang.Text = sChuc_nang;
int Id;
Id = Request.QueryString("ID");
lblId.Text = Id;
```

Trong trường hợp tên tham số không tồn tại trong chuỗi QueryString, thuộc tính Request.QueryString() sẽ trả về giá trị nothing.

1.2.2. Các ví dụ minh họa

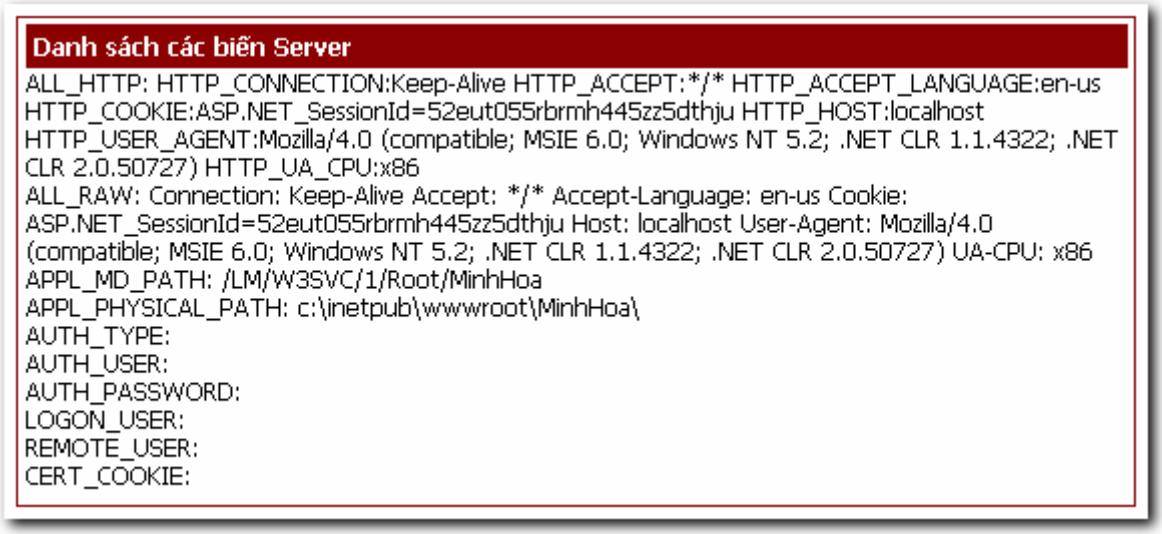
Ví dụ: Lấy thông tin các trình duyệt của người dùng.

```
string sThong_tin;
{
        sThong_tin += "Browser: " + Request.Browser.Browser + "<br>";
        sThong_tin += "Version: " + Request.Browser.Version + "<br>";
        sThong_tin += "Platform: " + Request.Browser.Platform + "<br>";
        sThong_tin += "JavaScript: " + Request.Browser.JavaScript + "<br>";
        lblThong_tin.Text = sThong_tin;
}
```

Thông tin của trình duyệt tại Client

Ví dụ: Liệt kê danh sách các biến Server

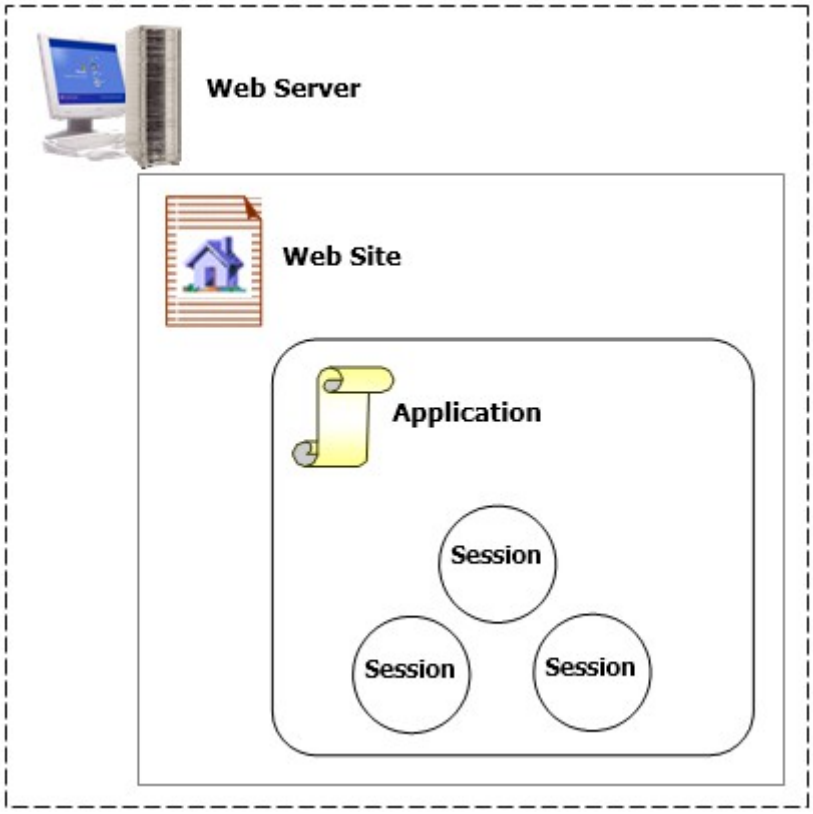
```
string sServer;
int i;
{
        for (i = 0; i <= Request.ServerVariables.Count - 1; i++) {
                sServer += Request.ServerVariables.Keys(i) + ": " +
Request.ServerVariables.Item(i) + "<br>";
        }
        lblServer.Text = sServer;
}
```



Danh sách các biến Server

2. Đối tượng Session, Application

Application và Session là 2 đối tượng khá quan trọng trong ứng dụng web, giúp các trang aspx có thể liên kết và trao đổi dữ liệu cho nhau. Trong phần này, chúng ta sẽ tìm hiểu và sử dụng 2 đối tượng này trong ứng dụng.



Quan hệ giữa Session và Application

2.1. Đối tượng Application

Đối tượng Application được sử dụng để quản lý tất cả các thông tin của một ứng dụng web. Thông tin được lưu trữ trong đối tượng Application có thể được xử lý trong bất kỳ trang aspx nào trong suốt chu kỳ sống của ứng dụng.

2.1.1. Sử dụng biến Application

Tạo biến Application

Lấy giá trị từ biến Application

Ví dụ:

Chú ý:

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 nên sử dụng bộ lệnh Lock và UnLock ngay trước và sau khi cập nhật giá trị của biến Application.

Biến Application có thể được sử dụng ở bất kỳ trang nào và được duy trì trong suốt chu kỳ sống của ứng dụng.

2.1.2. Duyệt qua tập hợp biến chứa trong Application

```
int i;
Response.Write("<b><u>Danh sách các biến trong đối tượng Application</u></b><br>");
for (i = 0; i <= Application.Count() - 1; i++) {
    Response.Write(Application.Keys(i) + " : ");
    Response.Write(Application(i) + "<br />");
}
```

Kết quả hiển thị

2.2. Đối tượng Session

Đối tượng Session được dùng để lưu trữ thông tin của người dùng trong ứng dụng. Thông tin được lưu trữ trong Session là của một người dùng trong 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 tự động hủy chúng nếu người dùng còn không làm việc với ứng dụng nữa.

Tuy nhiên, không giống như đối tượng Application, đối tượng Session không thể chia sẻ thông tin giữa những lần làm việc của người dùng, nó chỉ có thể cung cấp, trao đổi thông tin cho các trang trong lần làm việc tương ứng.

Trong ứng dụng web, đối tượng Session giữ vai trò khá quan trọng. Do sử dụng giao thức HTTP, một giao thức phi trạng thái, Web Server hoàn toàn không ghi nhớ những gì giữa những lần yêu cầu của Client. Đối tượng Session tỏ ra khá hữu hiệu trong việc thực hiện "lưu vết và quản lý thông tin của người dùng".

2.2.1. Thuộc tính & Phương thức

a. Thuộc tính Timeout

Qui định khoảng thời gian (tính bằng phút) mà Web Server duy trì đối tượng Session nếu người dùng không gởi yêu cầu nào về lại Server. Giá trị mặc định của thuộc tính này là 20.

Nếu không có yêu cầu nào kể từ lần yêu cầu sau cùng một khoảng thời gian là <Timeout> phút, đối tượng Session mà Web server cấp cho lần làm việc đó sẽ tự động được giải phóng. Những yêu cầu sau đó được Web server coi như là một người dùng mới, và đương nhiên sẽ được cấp một đối tượng Session mới.

b. Phương thức Abandon

Như các bạn đã biết, trong khoảng thời gian <Timeout> phút kể từ lần yêu cầu sau cùng của Client, đối tượng Session vẫn được duy trì dù cho không có sự tương tác nào của Client. Điều này đồng nghĩa với việc Web server phải sử dụng một vùng nhớ để duy trì đối tượng Session trong một khoảng thời gian tương ứng.

Phương thức Abandon của đối tượng Session sẽ giải phóng vùng nhớ được dùng để duy trì đối tượng Session trên Web Server ngay khi được gọi thực hiện. Những yêu cầu sau đó được Web server coi như là một người dùng mới.

2.2.2. Sử dụng biến toàn cục với Session

Tạo biến Session

Lấy giá trị từ biến Session

Ví dụ:

Lưu trữ thông tin khi người dùng chưa đăng nhập hệ thống:

Khi người dùng đăng nhập hệ thống thành công, cập nhật lại thông tin đăng nhập của người dùng được lưu trên Session.

Duyệt qua tập hợp biến chứa trong Session

3. Đối tượng Server

Đối tượng Server được sử dụng để cung cấp thông tin của Server cho ứng dụng.

Thuộc tính MachineName: Thuộc tính này được dùng để lấy tên của Web

Server.

Phương thức Mappath: được dùng để lấy đường dẫn vật lý hoặc đường dẫn ảo đến một thư mục trên Server.

Phương thức Transfer(<Đường dẫn đến trang cần yêu cầu>): Ngừng thi hành trang hiện hành, gởi yêu cầu mới đến trang được gọi thực hiện.

4. Đối tượng Cookies

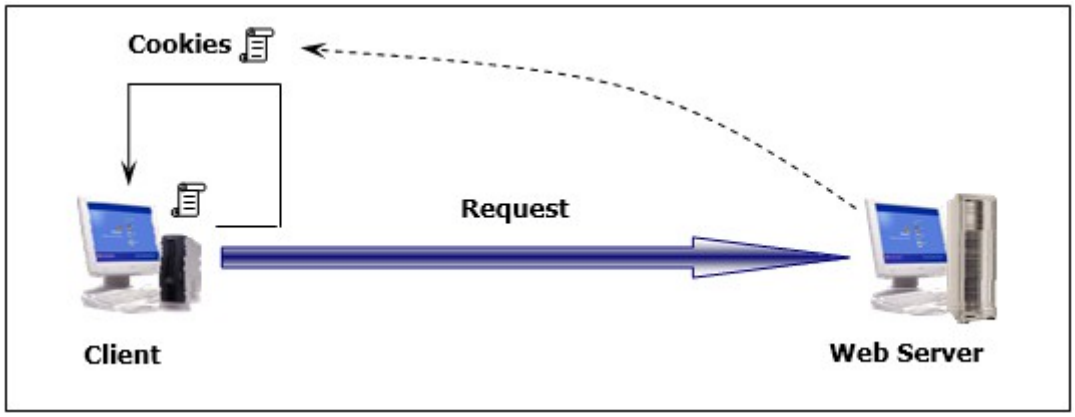
4.1. Giới thiệu

Có lẽ bạn cũng đã từng đăng ký là một thành viên của một trang web hay một forum nào đó, và chắc cũng không ít lần ngạc nhiên khi bạn vừa yêu cầu đến một trang web hay forum mà bạn đã đăng ký trước đó, trang web nhận ngay ra, bạn chính là thành viên của họ và gởi ngay lời chào đến bạn, chẳng hạn: Chào Nguyễn Anh Tài.

Làm sao mà Web Server nhận ra được mình nhỉ? Mình đã đăng ký từ ngày hôm qua kia mà? Không đâu xa cả, những thông tin đó được lưu ngay chính tại máy của bạn. Những thông tin được Web Server lưu tại máy Client được gọi là Cookies.

Không giống như đối tượng Session, đối tượng Cookies cũng được dùng để lưu trữ thông tin của người dùng, tuy nhiên, thông tin này được lưu ngay tại máy gởi yêu cầu đến Web Server.

Có thể xem một Cookie như một tập tin (với kích thước khá nhỏ) được Web Server lưu tại máy của người dùng. Mỗi lần có yêu cầu đến Web Server, những thông tin của Cookies cũng sẽ được gởi theo về Server.



4.2. Làm việc với Cookies

4.2.1. Thêm Cookies

Ví dụ:

Trong ví dụ trên, chúng ta đã tạo ra Cookies có tên là Ten_dang_nhap lưu trữ tên đăng nhập của người dùng. Thông tin này sẽ được lưu trữ trên Cookies 1 ngày kể từ ngày hiện hành trên Web Server.

4.2.2. Lấy giá trị từ Cookies

Trong trường hợp Cookies chưa được lưu hoặc đã hết thời hạn duy trì tại Client, giá trị nhận được là Nothing.

5. Tập tin quản lý và cấu hình ứng dụng

5.1. Global.asax

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.

5.1.1. 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>
-->
```

5.1.2. 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_nguoi_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_nguoi_online"] = (int)Application["So_nguoi_online"] + 1;
    }
-->
```

Application_BeginRequest: Xảy ra khi mỗi khi có Postback về Server.

Sub Application_Error: Xảy ra khi có lỗi phát sinh trong quá trình thi hành.

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_nguoi_online"] = (int)Application["So_nguoi_online"] - 1;
    }
-->
```

Qua tập tin Global.asax bạn có thể xây dựng được ứng dụng đếm số người truy

cập và người online trong website chúng ta. Ở người online thì dựa vào Session_Start và Session_End mà giảm tăng rồi xuất biến Application ra ngoài trang chủ còn về số lược truy cập thì cứ cộng dồn vào do đó bạn phải ghi vào file text hay vào trong CSDL số lượng đó để biến tăng dần lên theo ngày tháng!

5.2. Web.config

5.2.1. 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\[Framework Version]\CONFIG\machine.config

Tập tin cấu hình ứng dụng Web.config:

5.2.2. Các cấu hình mặc định

a. <compilation defaultLanguage="vb" debug="true"/> defaultLanguage: qui định ngôn ngữ mặc định của ứng dụng. debug: Bật/tắt chế độ debug của ứng dụng

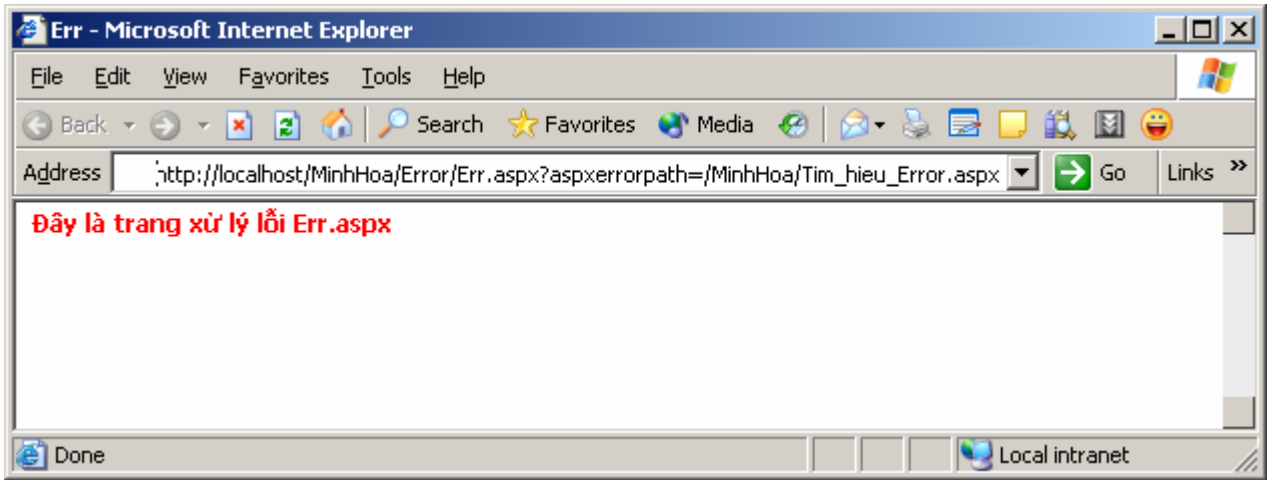
b. <customErrors mode="RemoteOnly"/>

Đây là một cấu hình khá cần thiết cho ứng dụng Web. Hiệu chỉnh 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.

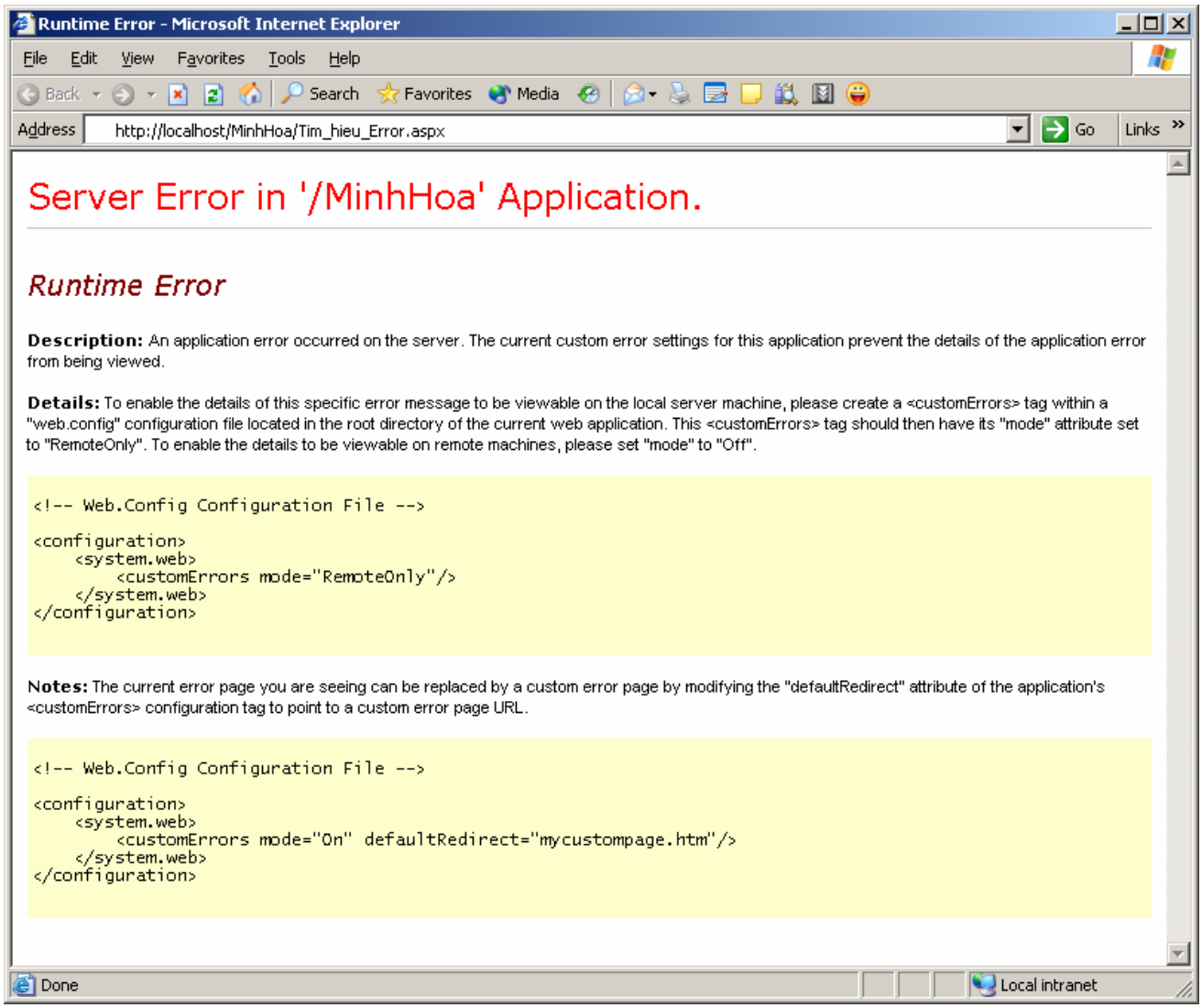
RemoteOnly: Cho phép người dùng thấy thông báo lỗi của hệ thống hoặc trang thông báo lỗi được chỉ định qua defaultRedirect (nếu có). Thông báo lỗi gồm: Mã lỗi và mô tả lỗi tương ứng

On: Tùy theo giá trị của defaultRedirect mà có các trường hợp tương ứng:

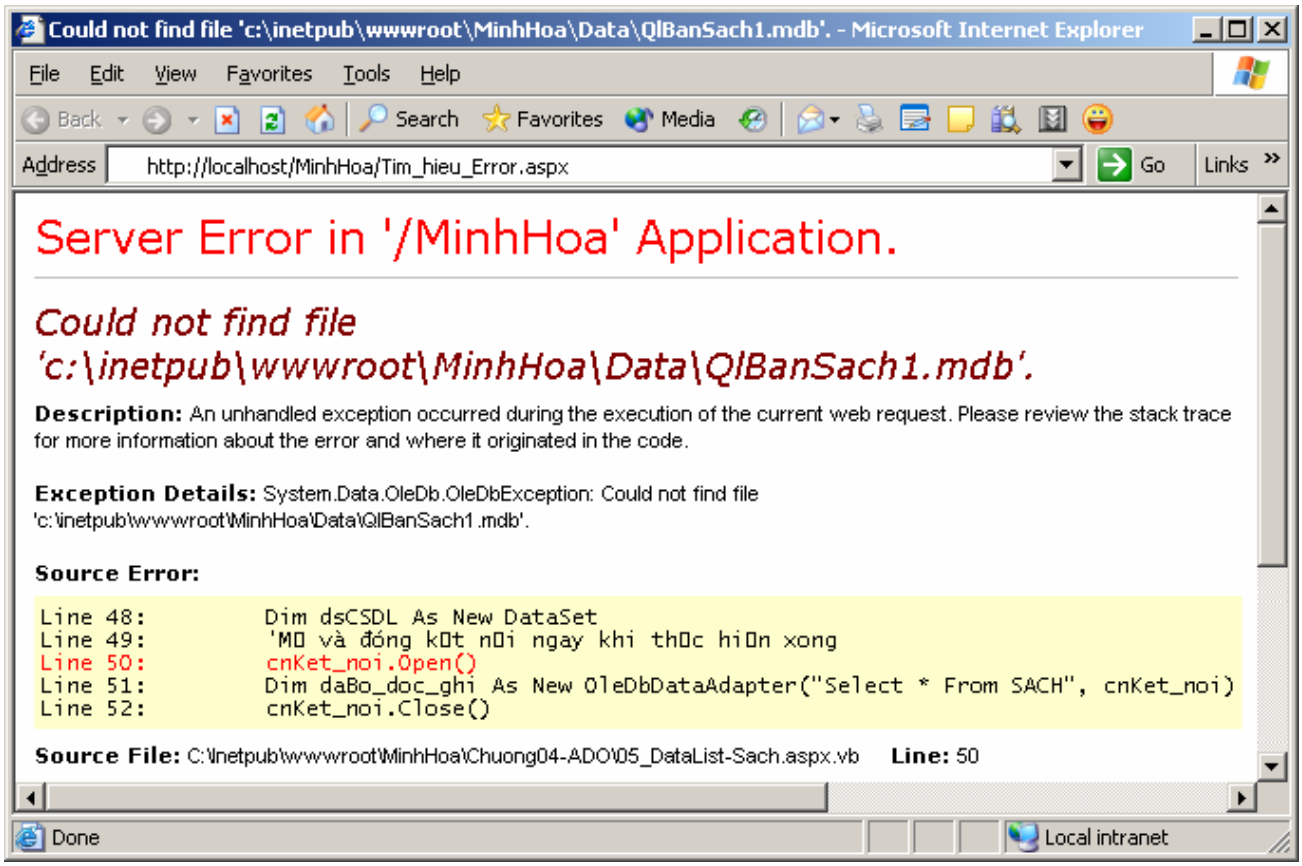
- Có qui định trang thông báo lỗi qua defaultRedirect: Hiển thị trang thông báo lỗi.



- Không có thuộc tính defaultRedirect: Hiển thị trang báo lỗi nhưng không có hiển thị mã lỗi và mô tả lỗi.



Off: Hiển thị thông báo lỗi của trang aspx (nếu xảy ra lỗi).



c. <sessionState>

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.

5.2.3. Làm việc với tập tin web.config

Tập tin web.config có hỗ trợ tag **<appSettings>** với 2 thuộc tính là **key** và **value** cho phép chúng ta thêm vào các biến dùng để cấu hình ứng dụng.

Lưu ý: Các tên tag trong tập tin cấu hình web.config có phân biệt chữ hoa, chữ thường.

Ví dụ:

Tạo biến cấu hình Ole_Con dùng để lưu trữ thông tin của chuỗi kết nối đến cơ sở dữ liệu SQL Server:

Lấy giá trị đã thiết lập trong tập tin web.config

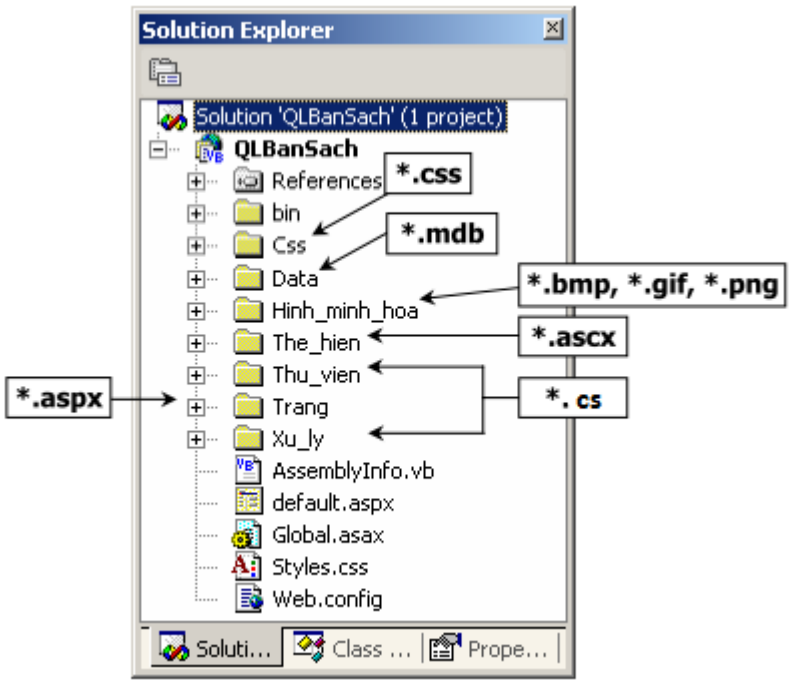
6. Tổ chức & xây dựng ứng dụng

6.1. Tổ chức lưu trữ ứng dụng

6.1.1. Màn hình giao diện

Giao diện ứng dụng quản lý bán hàng qua mạng

6.1.2. Tổ chức lưu trữ



Tổ chức lưu trữ ứng dụng

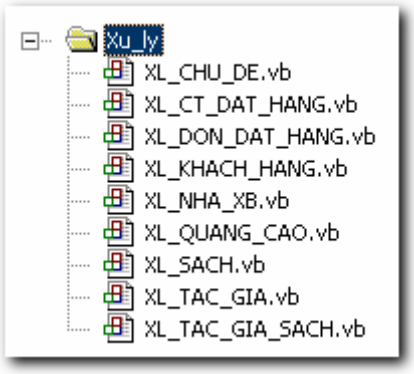
- Css: Lưu trữ các tập tin *.css - tập tin qui định hình thức hiển thị.
- Data: Lưu trữ tập tin *.mdb - tập tin cơ sở dữ liệu.
- Hinh_minh_hoa: Lưu trữ các tập tin hình ảnh (*.bmp, *.gif, *.png, …)
- Trong thư mục này, chúng ta có thể tổ chức các thư mục con để lưu trữ hình ảnh theo chủ đề, ngày, …
- The_hien: Lưu trữ các điều khiển do người dùng tạo - các đối tượng thể hiện.
- Thu_vien: Lưu trữ các tập tin thư viện dùng chung của ứng dụng.
- Trang: Lưu trữ các màn hình - các trang Web (*.aspx)
- Xu_ly: Lưu trữ các lớp xử lý dữ liệu

6.2. Xây dựng ứng dụng

6.2.1. Xây dựng lớp Xử lý dữ liệu

Ứng với mỗi bảng trong cơ sở dữ liệu, chúng ta xây dựng các lớp xử lý tương ứng. Các lớp xử lý dữ liệu xây dựng tương tự như lớp XL_SACH.

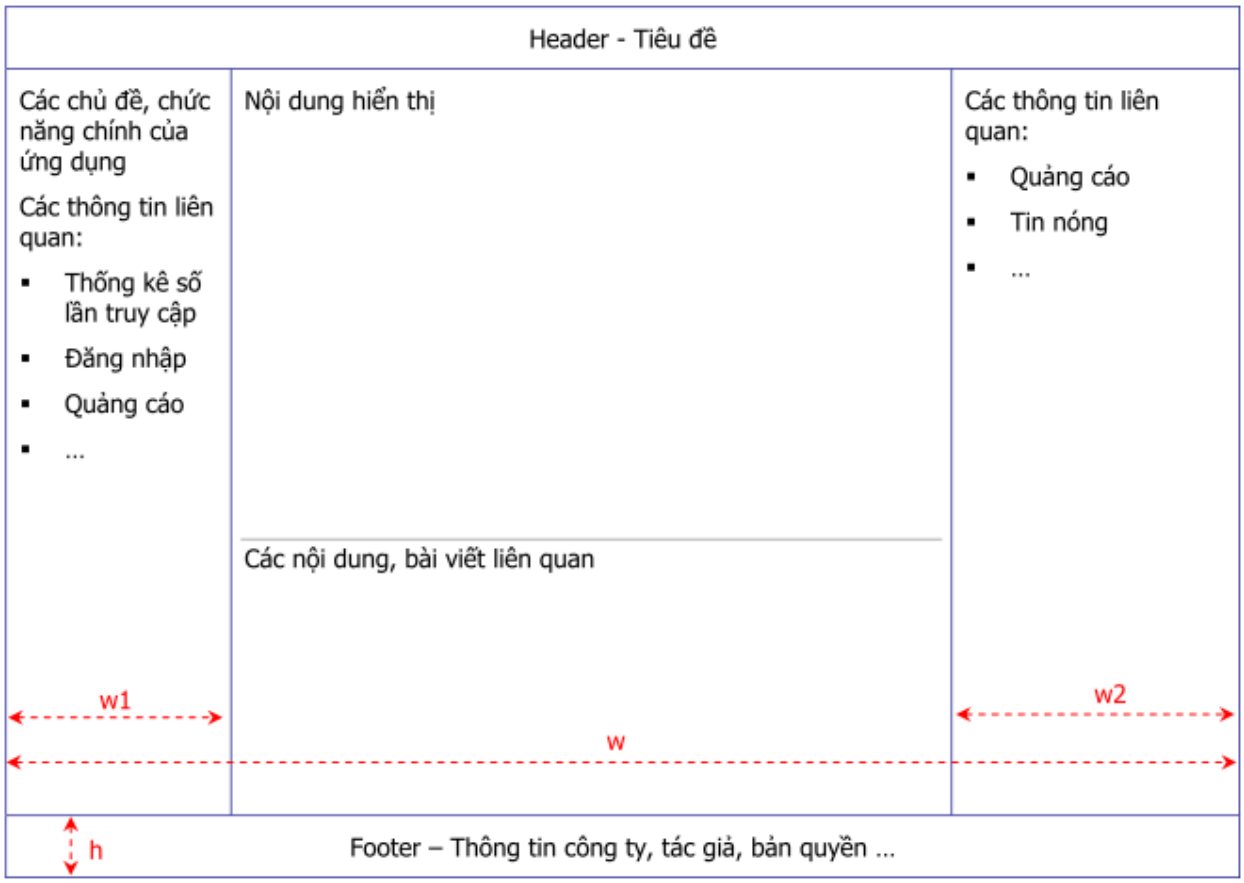
Thiết kế cơ sở dữ liệu của ứng dụng Quản lý bán hàng.



Danh sách các lớp xử lý

6.2.2. Thiết kế trang Web

Trước khi bắt tay vào thiết kế các đối tượng thể hiện và màn hình giao diện cho ứng dụng, chúng ta cũng nên nghĩ tới sẽ thiết kế trang web chạy trên màn hình có độ phân giải nào (thường dùng hiện nay là 800x600). Yếu tố này tuy không quan trọng nhưng nó cũng phần nào quyết định bố cục trình bày của trang web.



Kiến trúc tổng thể trang web

BÀI 7. WEB SERVICE

A. Mục tiêu

- Tìm hiểu Web Services
- Xây dựng và sử dụng Web Services

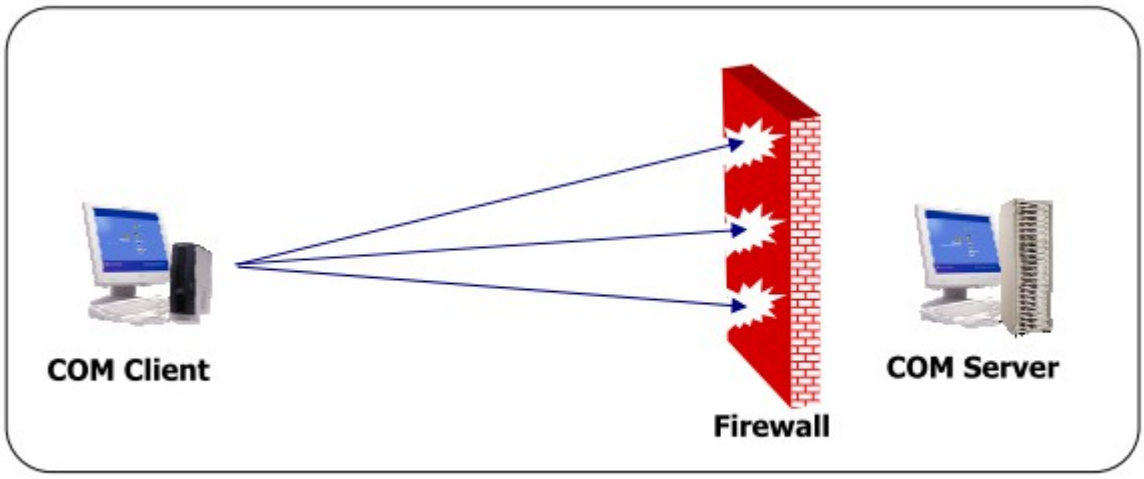
B. Nội dung

Trong phần này, chúng ta sẽ tìm hiểu Web services là gì? Công dụng của nó như thế nào? Sau khi hiểu được ý nghĩa và tầm quan trọng của nó, chúng ta sẽ bắt tay vào xây dựng Web Services.

1. Tìm hiểu về Web Services

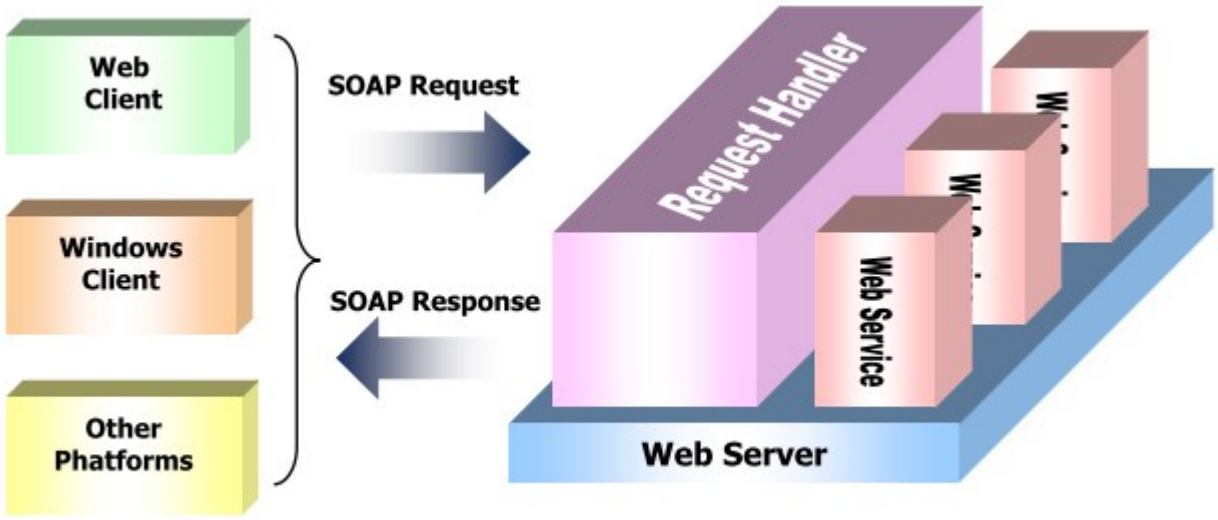
Khi bạn xây dựng và phát triển một ứng dụng phân tán với số lượng người dùng lên đến hàng trăm, hàng nghìn người ở nhiều địa điểm khác nhau, khó khăn đầu tiên mà bạn gặp phải là sự giao tiếp giữa Client và Server bị tường lửa (firewalls) và Proxy Server ngăn chặn lại.

Như các bạn biết DCOM (Distribited Component Object Model) làm việc thông qua việc gởi các thông tin dưới dạng nhị phân (binary) và chủ yếu hoạt động dựa trên giao thức TCP/IP. Thật là không dễ dàng để sử dụng DCOM trong trường hợp này.



Nếu không cấu hình lại Firewall, DCOM không có khả năng vượt qua Firewall

Web Services có thể giúp bạn giải quyết vấn đề khó khăn nêu trên. Chúng ta có thể hiểu rằng Web Services (tạm dịch là dịch vụ web) là tập hợp các phương thức của một đối tượng mà các Client có thể gọi thực hiện.



Kiến trúc Web Services

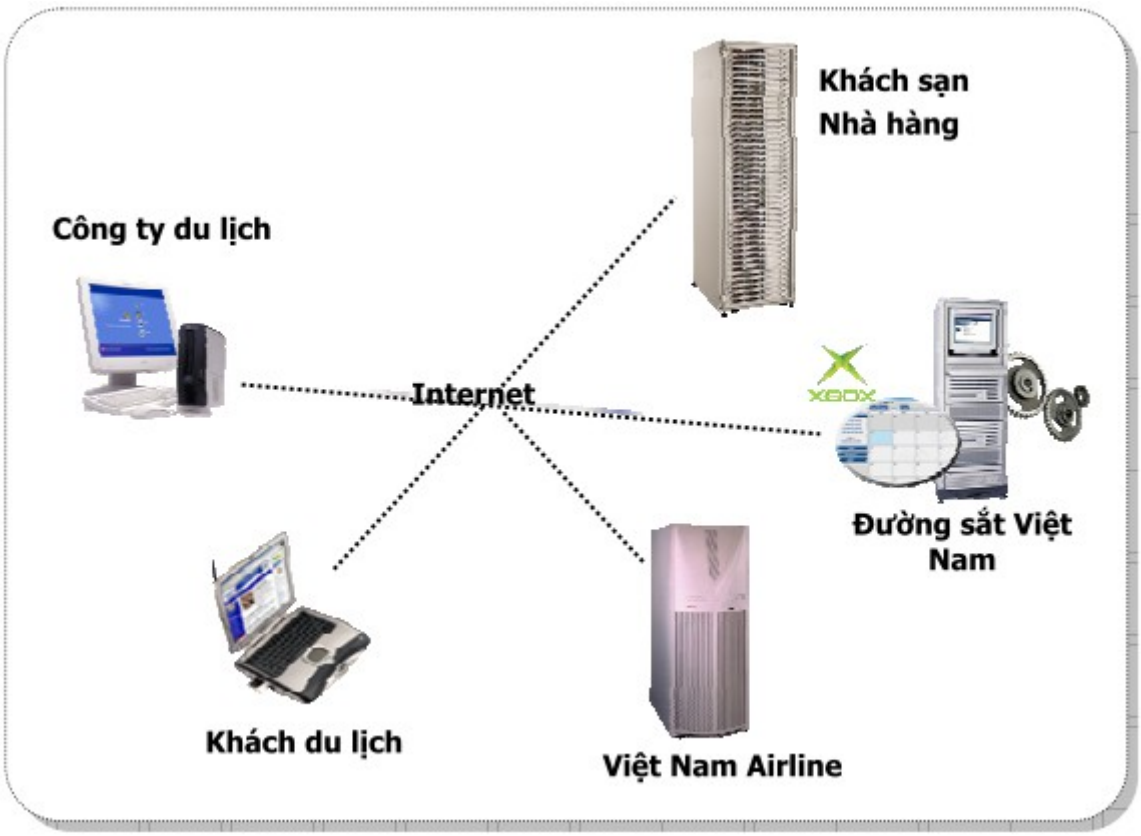
Web Services được xây dựng dựa trên SOAP (Simple Object Access Protocal). Không giống như DCOM, SOAP có thể được gọi thực hiện và trả về kết quả Text (theo định dạng XML) và **có khả năng hoạt động "xuyên qua" tường lửa.**

Ngoài khả năng ưu việt trên, Web Services có thể phối hợp hoạt động giữa các ứng dụng rất tốt. Hình minh họa trang bên là một ví dụ minh họa về sự phối hợp hoạt động giữa các ứng dụng.

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.

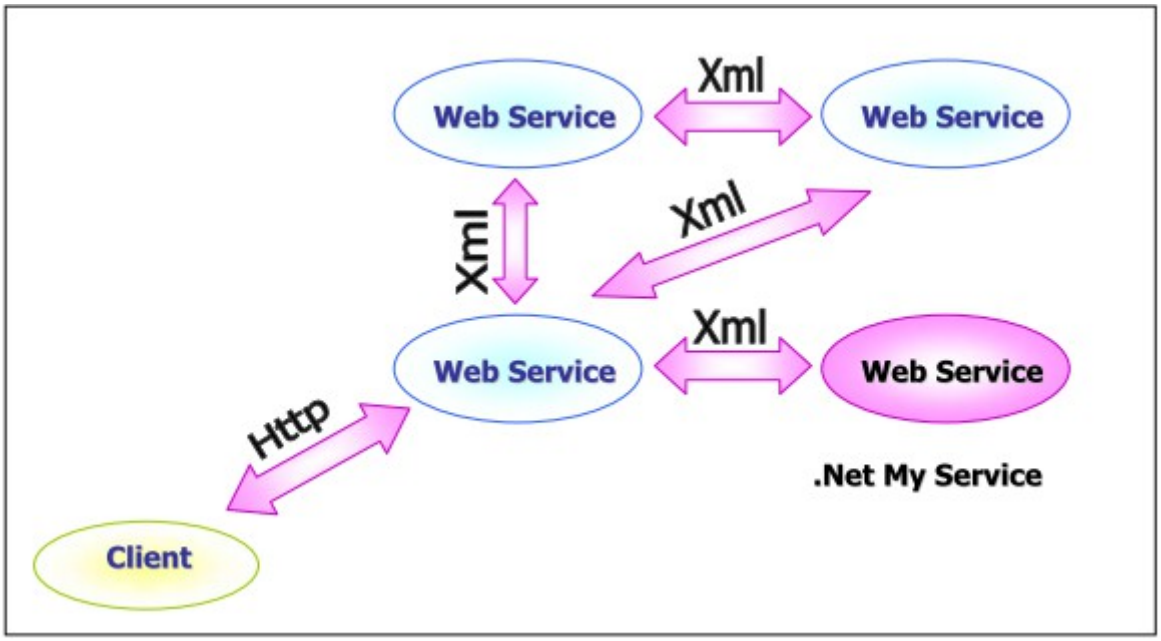
Các cơ quan, công ty, hay khách du lịch có nhu cầu tổ chức, tham gia các chuyến du lịch có thể truy cập vào website của các công ty dịch vụ lữ hành đăng ký tham gia các "tour" do họ tổ chức.

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.



Phối hợp hoạt động giữa các ứng 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.



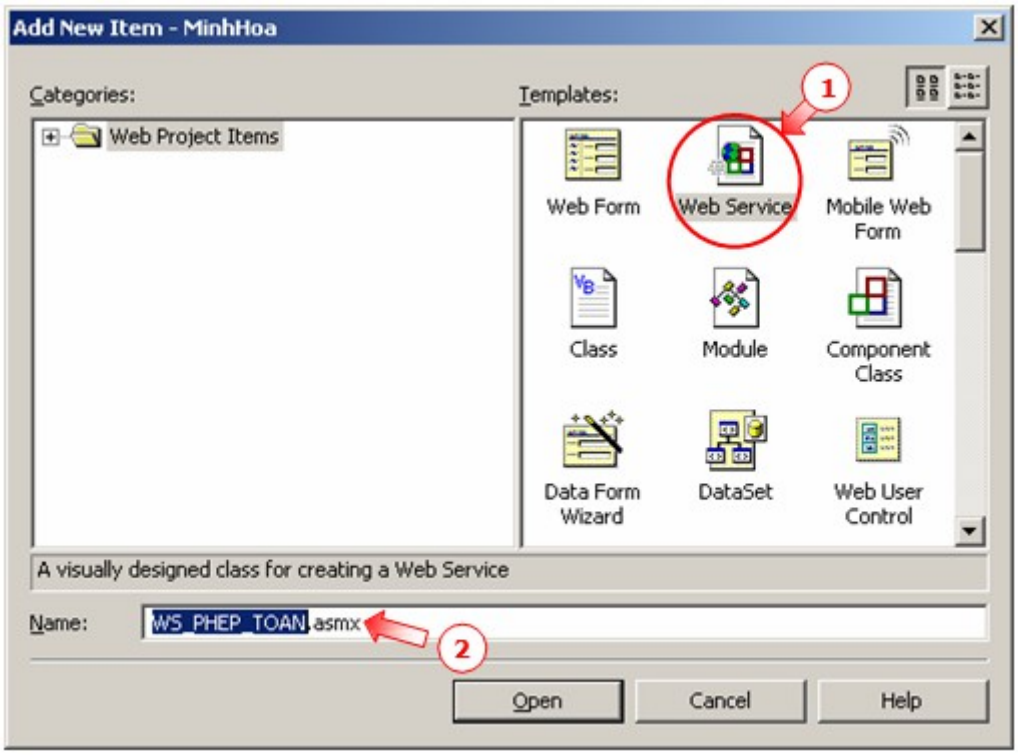
XML là định dạng dữ liệu chuẩn để trao đổi giữa các Web Services

2. Xây dựng Web Services

2.1. Tạo Web Services trong VS .Net

Trong phần này, chúng ta sẽ xây dựng một Web Service đơn giản có tên WS_PHEP_TOAN, với phương thức Cong_hai_so trong Visual Studio .Net

Chọn Add|Add New Items… từ thực đơn ngữ cảnh của Project. Chọn mục Web Service trong khung Template. Đổi tên Web Service cần tạo thành WS_PHEP_TOAN.



Tạo mới Web Service

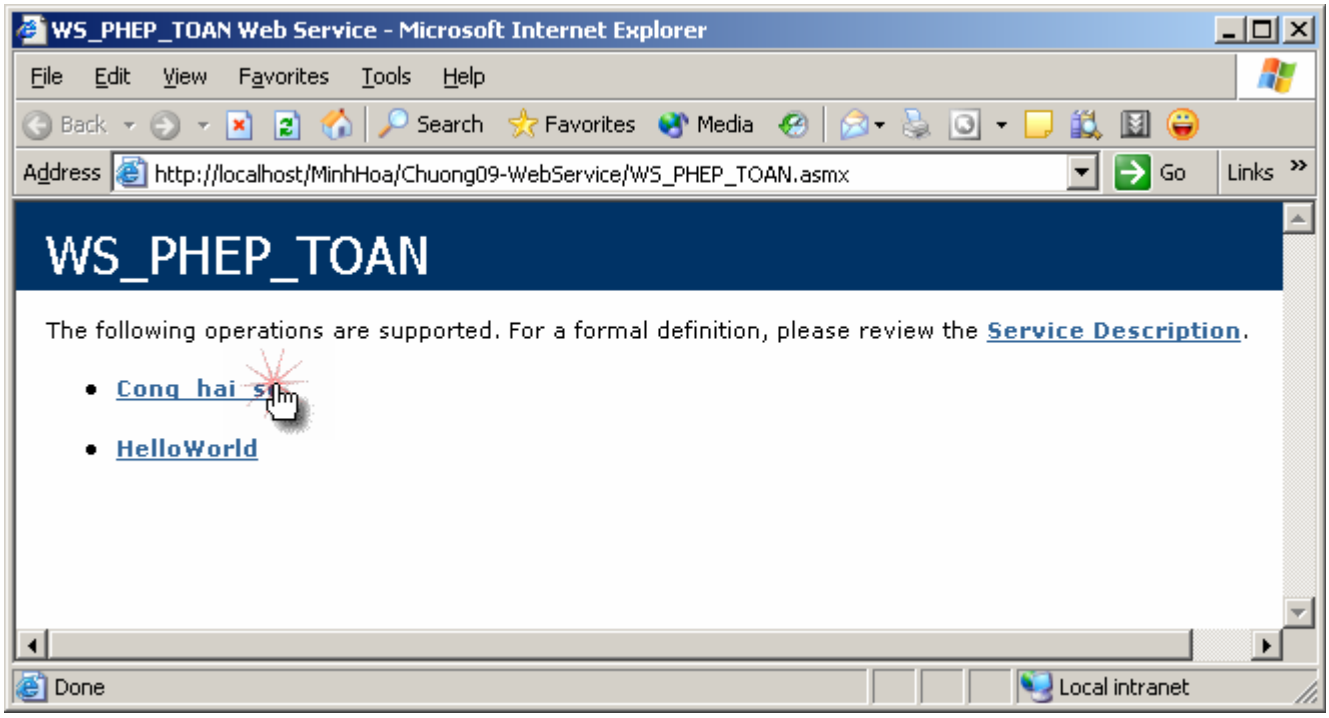
Trong cửa sổ viết lệnh, có một phương thức mẫu được tạo sẵn: phương thức HelloWorld.

```
Imports System.Web.Services

<System.Web.Services.WebService(Namespace := "http://tempuri.org/MinhHoa/WS_PH
Public Class WS_PHEP_TOAN
    Inherits System.Web.Services.WebService

 Web Services Designer Generated Code

    ' WEB SERVICE EXAMPLE
    ' The HelloWorld() example service returns the string Hello World.
    ' To build, uncomment the following lines then save and build the project.
    ' To test this web service, ensure that the .asmx file is the start page
    ' and press F5.
    '
    <WebMethod()> _
    Public Function HelloWorld() As String
        Return "Hello World"
    End Function

End Class
```

Bạn có nhận thấy rằng trước phương thức HelloWorld có sẵn từ khóa **<WebMethod()>**. Chúng ta sẽ bổ sung vào phương thức Cong_hai_so.

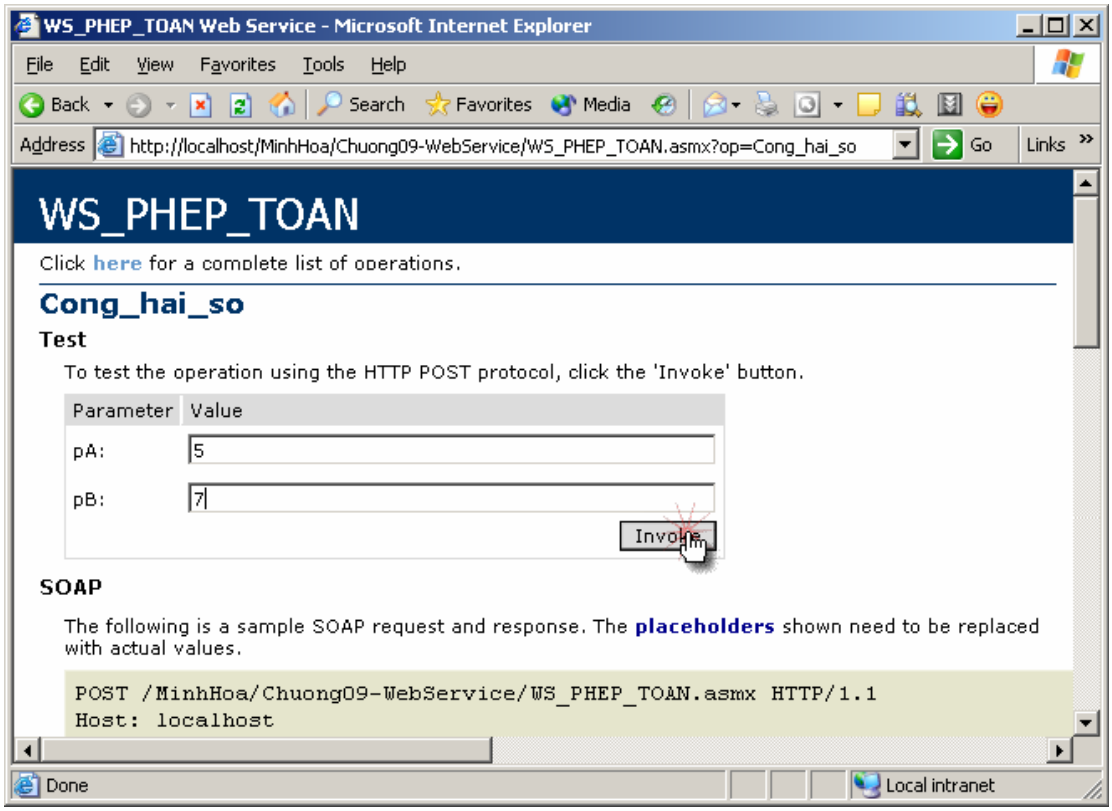
2.2. Kiểm tra Web Service

Sau khi xây dựng thành công Web Service, trước khi đưa vào sử dụng, chúng ta cũng nên tiến hành kiểm tra Web Service. Các Web Service được xây dựng trong VS.Net tự động phát sinh ra các trang kiểm tra tương ứng. Để thực hiện điều này, các bạn chọn WS_PHEP_TOAN.asmx làm trang khởi động, nhấn F5 để thi hành ứng dụng.

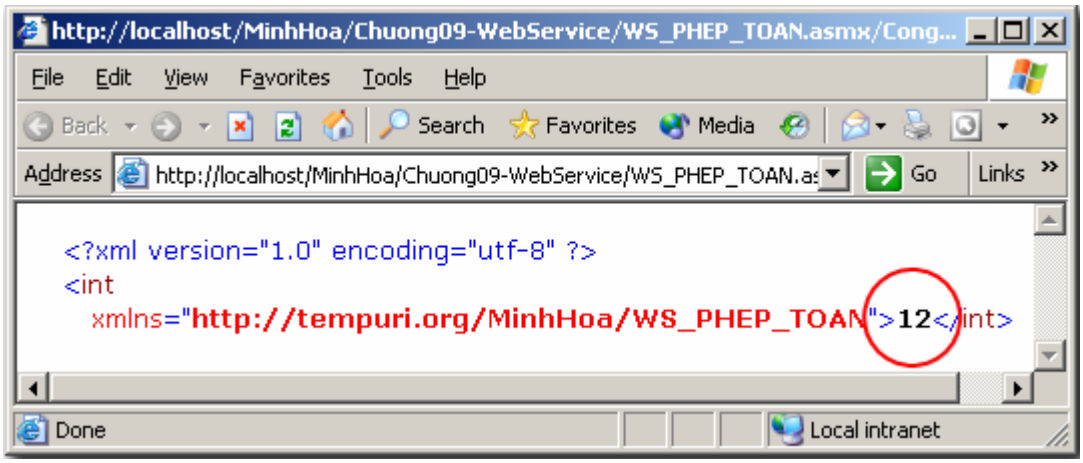


Màn hình kiểm tra Web Service

Trang kiểm tra Web Service sẽ liệt kê các phương thức hiện có trong Web Service được chọn thi hành. Chọn phương thức cần kiểm tra. Ở đây, chúng ta chọn phương thức Cong_hai_so. Xuất hiện màn hình nhập các tham số cho phương thức Cong_hai_so.



Nhập các tham số cần thiết và nhấn nút Invoke để thi hành, chúng ta sẽ thấy xuất hiện trang kết quả như hình bên dưới.



Màn hình kết quả

3. Sử dụng Web Service

Sau khi hoàn tất việc xây dựng, kiểm tra độ tin cậy và tính chính xác của Web Service, chúng ta sẽ tiến hành đưa Web Service đi vào sử dụng.

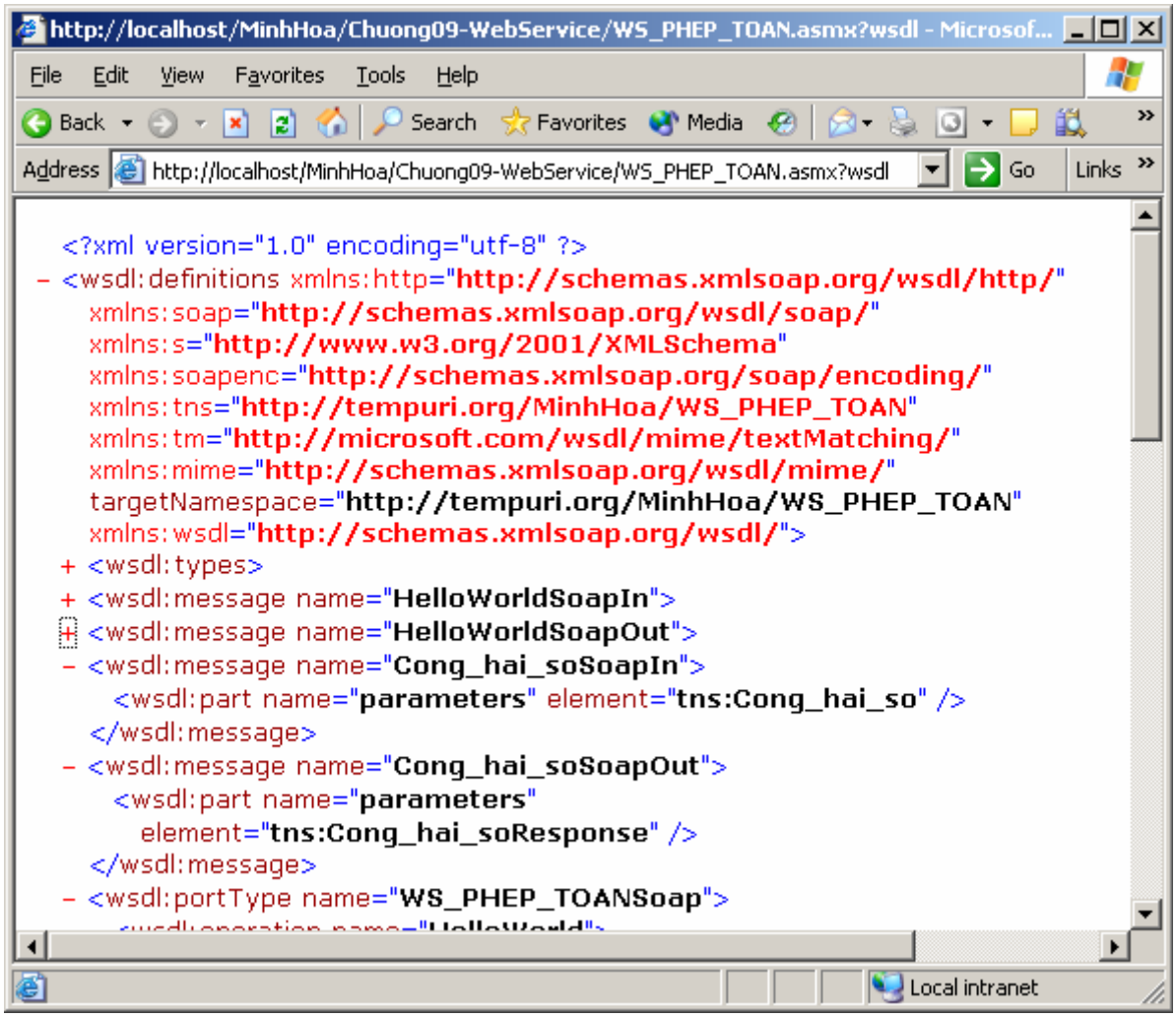
Để sử dụng một Web Service, Client cần phải biết Web Service đó hỗ trợ những phương thức nào, phương thức cần có những tham số nào, kết quả trả về ra sao…

Những thông tin này của một Web Service được mô tả bởi tài liệu WSDL (Web Service Description Language). WSDL là định dạng chuẩn để mô tả các Web Service, sử dụng ngôn ngữ XML.

Chúng ta có thể xem WSDL của một Web Service bằng cách thêm vào chuỗi tham số wsdl vào sau chuỗi URL:

Ví dụ:

http://localhost/MinhHoa/Chuong09-WebService/WS_PHEP_TOAN.asmx?**wsdl**



WSDL của Wes Service WS_PHEP_TOAN

3.1. Sử dụng Web Service do người dùng xây dựng

Sử dụng Web Service do chúng ta xây dựng tương tự như việc sử dụng các lớp đối tượng.

Ví dụ:

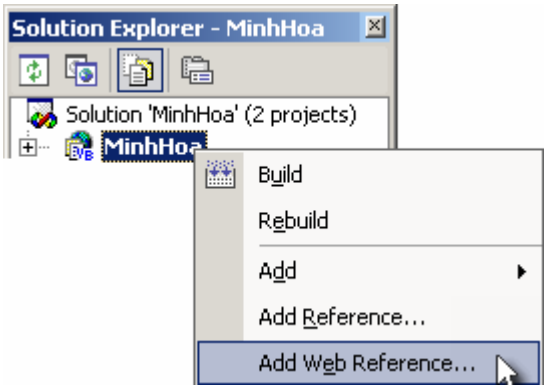
3.2. Sử dụng Web Services được cung cấp miễn phí trên mạng

Để biết được những Web Services được cung cấp miễn phí trên mạng, các bạn có thể dùng google để thực hiện tìm kiếm. Ở đây, chúng tôi giới thiệu đến các bạn trang: http://www.webservicex.net cung cấp khá nhiều các Web Services hữu ích.

Trong phần này, chúng tôi sẽ hướng dẫn bạn sử dụng các Web Services để lấy thông tin tỷ giá ngoại tệ, thông tin thời tiết, các thành phố chính của một quốc gia và các đơn vị tiền tệ của các quốc gia trên thế giới.

Các bước thực hiện:

Bước 1. Chọn Add Web Reference… từ thực đơn ngữ cảnh của ứng dụng



Thêm WebService vào ứng dụng

Bước 2. Nhập thông tin đường dẫn của WebService, nhấn 

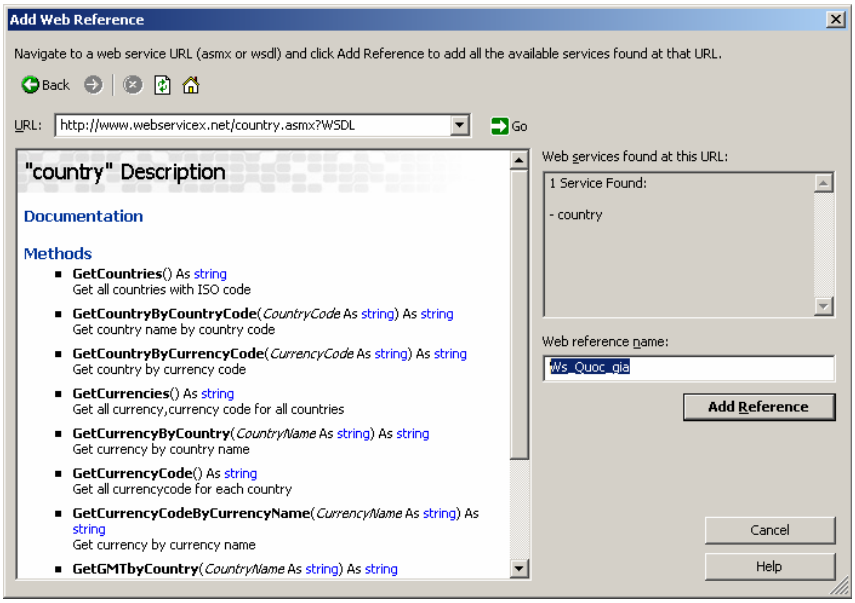
để thực hiện việc xác thực Web Service.

Bước 3. Nếu Web Service được xác thực thành công, thông tin mô tả về Web Service sẽ được hiển thị ngay phía bên dưới. Trong ví dụ này, chúng ta lần lượt xác thực 3 Web Service sau:

http://www.webservicex.net/country.asmx?WSDL → Các quốc gia

http://www.webservicex.net/globalweather.asmx?WSDL →Thời tiết

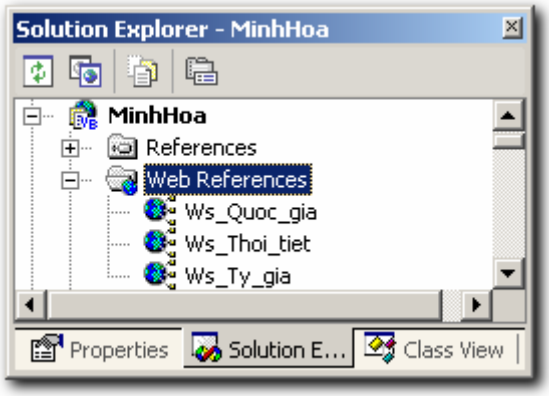
http://www.webservicex.net/CurrencyConvertor.asmx?WSDL →Tỷ giá



Tham chiếu Web Service

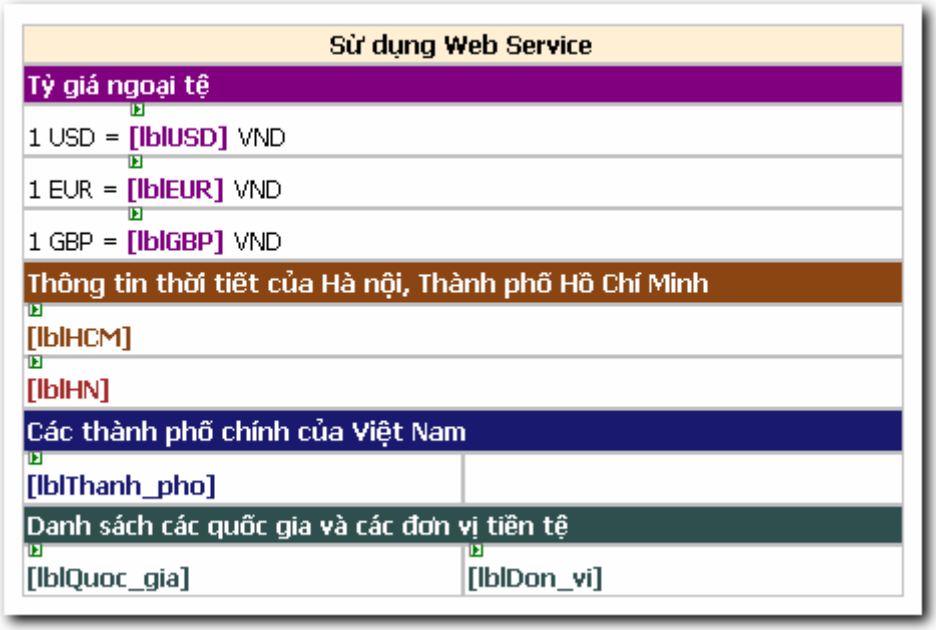
Bước 4. Đặt tên tham chiếu cho WebService: **Web reference name**.

Bước 5. Nhấn **Add Reference** để hoàn tất tham chiếu WebService. Bảng trên mô tả các phương thức và tài liệu hướng dẫn của WebService



Danh sách các Web Service được tham chiếu

Bước 6. Thiết kế màn hình



Màn hình thiết kế

Mã lệnh xử lý:

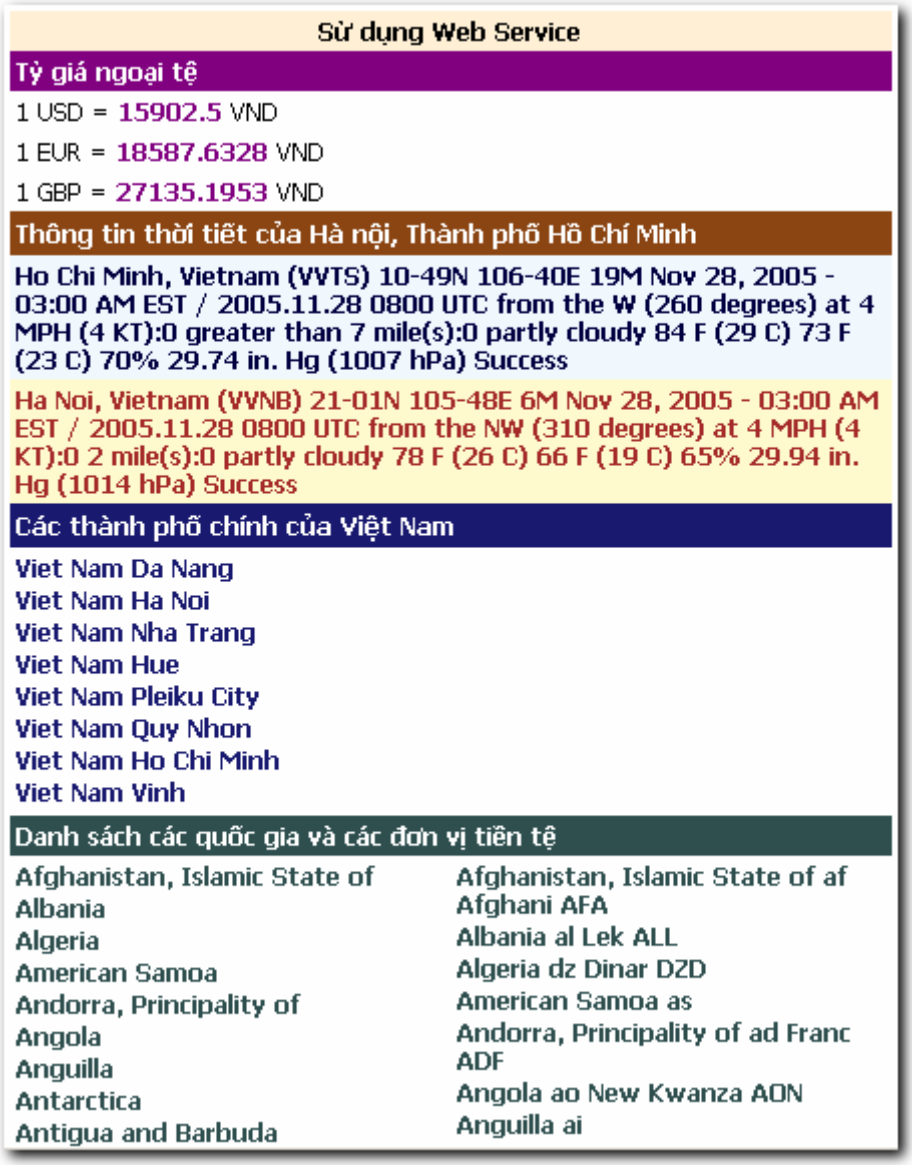
```
//Lấy tỉ giá ngoại tệ
Ws_Ty_gia.CurrencyConvertor tgNgoai_te = new Ws_Ty_gia.CurrencyConvertor();
lblUSD.Text = tgNgoai_te.ConversionRate(Ws_Ty_gia.Currency.USD, Ws_Ty_gia.Currency.VND);
lblEUR.Text = tgNgoai_te.ConversionRate(Ws_Ty_gia.Currency.EUR, Ws_Ty_gia.Currency.VND);
lblGBP.Text = tgNgoai_te.ConversionRate(Ws_Ty_gia.Currency.GBP, Ws_Ty_gia.Currency.VND);

//Lấy thông tin thời tiết các thành phố chính
Ws_Thoi_tiet.GlobalWeather lWeather = new Ws_Thoi_tiet.GlobalWeather();
lblHN.Text = lWeather.GetWeather("Ha Noi", "Viet Nam");
lblHCM.Text = lWeather.GetWeather("Ho Chi Minh", "Viet Nam");
lblThanh_pho.Text = lWeather.GetCitiesByCountry("Viet Nam");
//Lấy tên các quốc gia và đơn vị tiền tệ trên thế giới
```

Ws_Quoc_gia.country lCountry = new Ws_Quoc_gia.country();

lblDon_vi.Text = lCountry.GetCurrencyCode();

lblQuoc_gia.Text = lCountry.GetCountries();

Tỷ giá ngoại tệ và thông tin thời tiết được lấy từ WebService vào lúc: 9:30:00 AM ngày 13/06/2005 (giờ Việt nam). Lúc các bạn thi hành, các giá trị này có thể thay đổi.



Màn hình kết quả

4. Xây dựng Web Services truy xuất dữ liệu

4.1. Web Service: WS_KHACH_HANG

Trong phần này, chúng ta phối hợp các lớp xử lý đã có để xây dựng Web service WS_KHACH_HANG. Trong ví dụ minh họa dưới đây, chúng ta xây dựng ba thủ tục:

Doc_danh_sach_khach_hang

Them_khach_hang

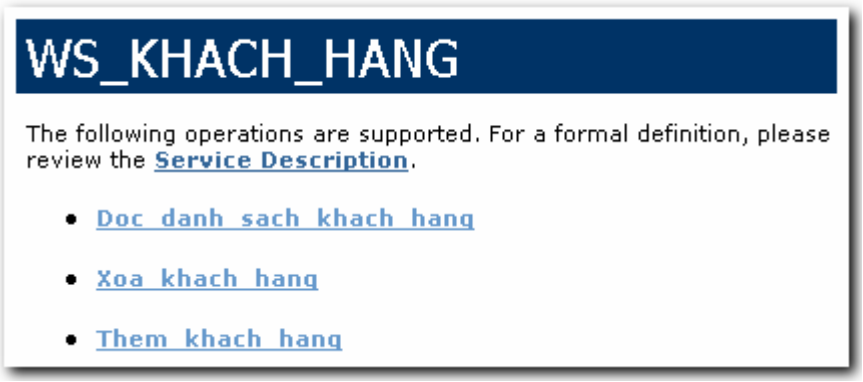
Xoa_khach_hang

Trong Web service trên, chúng ta có sử dụng một số phương thức từ lớp XL_KHACH_HANG. Để sử dụng, chúng ta cần bổ sung các phương thức sau vào lớp XL_KHACH_HANG:

4.2. Sử dụng WS_KHACH_HANG

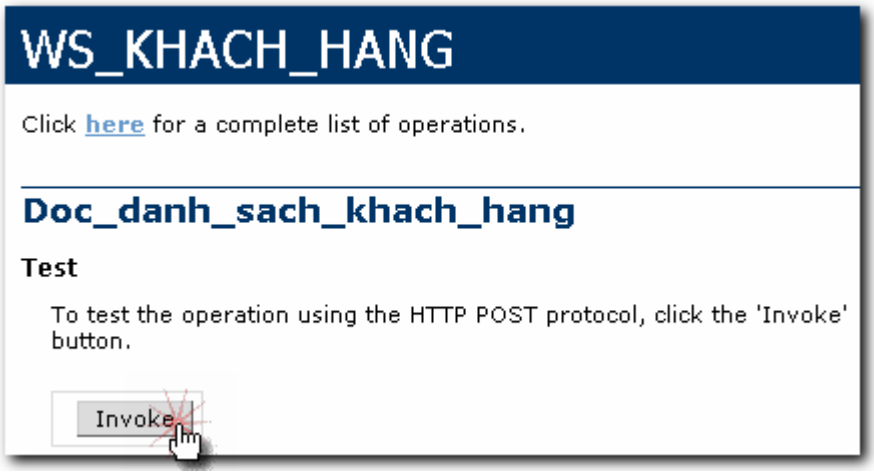
4.2.1. Kiểm tra Web Service

Sau khi thiết kế thành công WS_KHACH_HANG, chúng ta tiến hành kiểm tra Web service vừa tạo.



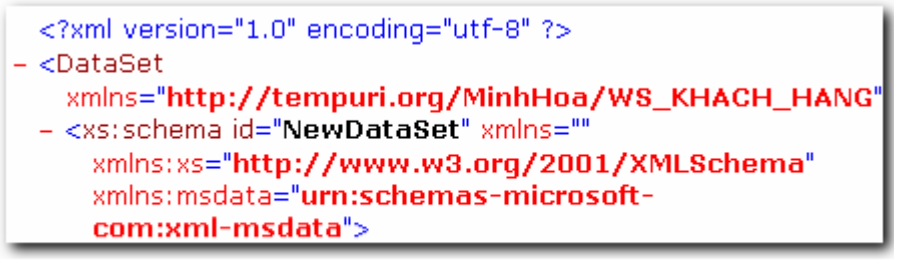
Danh sách các phương thức của WS_KHACH_HANG

Chọn chức năng Doc_danh_sach_khach_hang.



Thi hành phương thức Doc_danh_sach_khach_hang

Kết quả:



Kết quả dưới dạng XML

4.2.2. Sử dụng Web Service WS_KHACH_HANG

Sau khi xây dựng và kiểm tra thành công WS_KHACH_HANG, chúng ta sử dụng web service vừa tạo vào ứng dụng.

Viết lệnh xử lý:

Màn hình kết quả:

Danh sách khách hàng

Họ khách hàng	Tên kh	Tên đ.nhập	Địa chỉ	Ngày sinh
Hoàng Chinh	Thảo	hcthao	118 Nhân Chính	08/07/1962
Mai Giang	Giang	mggiang	85/6 Lê Quang Sung	15/07/1974
Bùi Thị Mộng	Ánh	btmanh	97/92/10 Lê Anh Xuân	24/07/1965
Trịnh Linh	Tâm	tltam	17 Nguyễn Thái Học	15/08/1946

PHỤ LỤC

1. Cơ sở dữ liệu dùng trong ứng dụng

1.1. Thiết kế cơ sở dữ liệu

1.1.1. Cấu trúc bảng dữ liệu

a. Bảng Chủ đề - CHU_DE

Field Name	Field Type	Field Size	Description
Mcd	Autonumber	Long Integer	
Ten_chu_de	Text	50	

b. Bảng Sách - SACH

Field Name	Field Type	Field Size	Description
Ms	Autonumber	Long Integer	
Ten_sach	Text	100	
Don_vi_tinh	Text	50	
Don_gia	Number	Currency	
Mo_ta	Memo		Tóm tắt nội dung
Hinh_minh_hoa	Text	50	Ánh minh họa
Mcd	Number	Long Integer	Mã chủ đề
Mnxb	Number	Long Integer	Mã nhà xuất bản
Ngay_cap_nhat	Date/Time		Ngày cập nhật
So_luong_ban	Number	Long Integer	
So_lan_xem	Number	Long Integer	

c. Bảng Khách hàng - KHACH_HANG

Field Name	Field Type	Field Size	Description
Mkh	Autonumber	Long Integer	
Ho_khach_hang	Text	50	
Ten_khach_hang	Text	50	
Dia_chi	Text	50	
Dien_thoai	Text	10	
Ten_dang_nhap	Text	15	

Mat_khau	Text	15	
Ngay_sinh	Date/Time		
Gioi_tinh	Yes/No		Yes: Nam
Email	Text	50	
Da_duyet	Yes/No		Yes: Đã duyệt

d. Bảng Đơn đặt hàng - DON_DAT_HANG

Field Name	Field Type	Field Size	Description
Sdh	Autonumber	Long Integer	
Mkh	Number	Long Integer	
Ngay_dat_hang	Date/Time		
Tri_gia	Number	Currency	
Da_giao_hang	Yes/No		Yes: Đã giao
Ngay_giao_hang	Date/Time		

e. Bảng Chi tiết đặt hàng - CT_DAT_HANG

Field Name	Field Type	Field Size	Description
Sdh	Number	Long Integer	
Ms	Number	Long Integer	
So_luong	Number	Long Integer	
Don_gia	Number	Double	
Thanh_tien	Number	Double	

f. Bảng Nhà xuất bản - NHA_XUAT_BAN

Field Name	Field Type	Field Size	Description
Mnxb	Autonumber	Long Integer	
Ten_nha_xuat_ban	Text	100	
Dia_chi	Text	150	
Dien_thoai	Text	15	

g. Bảng Tác giả - TAC_GIA

Field Name	Field Type	Field Size	Description
Mtg	Autonumber	Long Integer	
Ten_tac_gia	Number	Long Integer	
Dia_chi	Text	100	
Dien_thoai	Text	15	

h. Bảng Viết Sách – VIET_SACH

Field Name	Field Type	Field Size	Description
Stt	Autonumber	Long Integer	
Mtg	Number	Long Integer	
Ms	Number	Long Integer	

Các bảng dưới đây được dùng để Thăm dò dư luận & Quảng cáo

i. Bảng Thăm dò - THAM_DO

Field Name	Field Type	Field Size	Description
Mch	Autonumber	Long Integer	
Ngay_dang	Date/Time		
Noi_dung	Text	255	
Tong_so_binh_chon	Number	Long Integer	Mặc định = 0

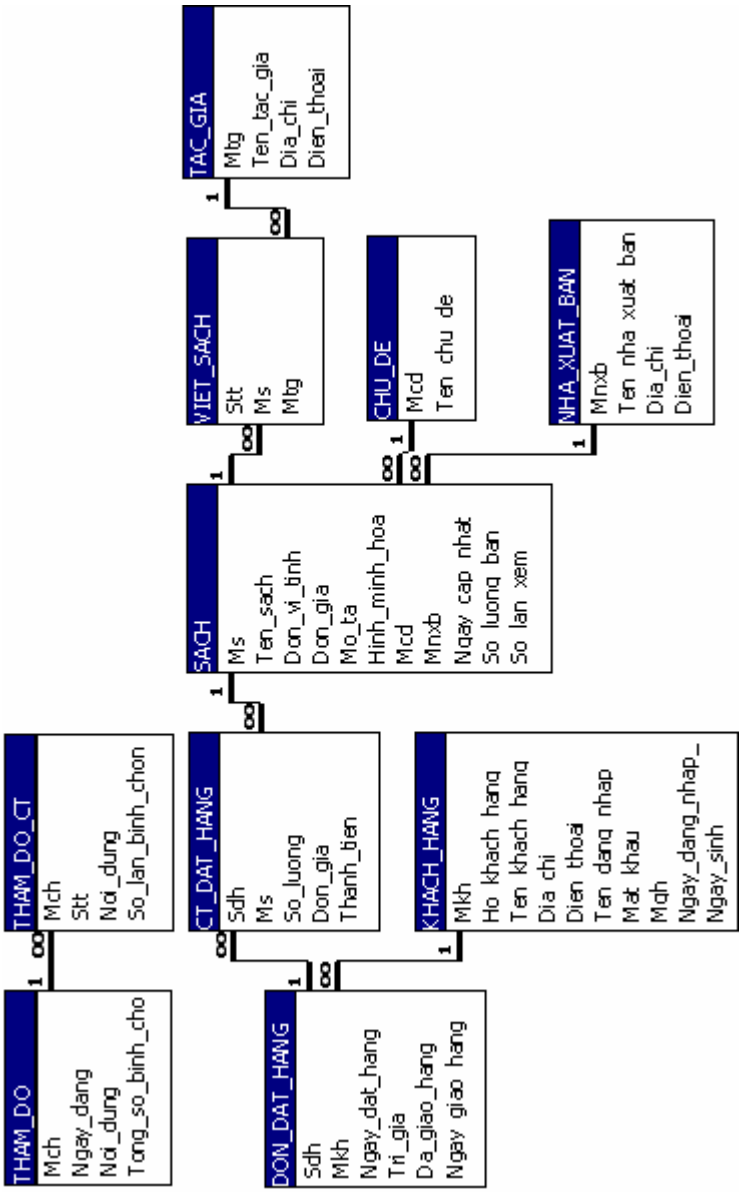
j. Bảng Thăm dò chi tiết - THAM_DO_CT

Field Name	Field Type	Field Size	Description
Mch	Number	Long Integer	
Stt	Number	Long Integer	1,2,3,4,… ứng với
Noi_dung	Text	255	
So_lan_binh_chon	Number	Long Integer	Mặc định = 0

k. Bảng Quảng cáo - QUANG_CAO

Field Name	Field Type	Field Size	Description
Stt	Autonumber	Long Integer	
Ten_Cong_ty	Text	200	
Hinh_minh_hoa	Text	100	Ánh minh họa
Duong_dan	Text	100	(đến trang q.cáo)
Ngay_ky_hd	Date/Time		Ngày ký hợp đồng
Ngay_bat_dau	Date/Time		Bắt đầu quảng cáo
Ngay_ket_thuc	Date/Time		Hết hạn quảng cáo

1.1.2. Quan hệ giữa các bảng



1.2. Dữ liệu thử

1.2.1. Bảng Chủ đề - CHU_DE

Mcd	Ten_chu_de
1	Tiếng Việt
2	Ngoại ngữ
3	Công nghệ thông
4	Luật

1.2.2. Bảng Nhà xuất bản – NHA_XUAT_BAN

Mnxb	Ten_nha_xuat_ba	Dia_chi	Dien_thoai
1	Nhà xuất bản Trẻ	123 Nguyễn Du	19001560
2	NXB Thống kê	456 Cống Quỳnh	19001511
3	Kim đồng	789 Nguyễn Trãi	19001570
4	Văn hóa nghệ	357 Cộng Hòa	0903118833

1.2.3. Bảng Tác giả – TAC_GIA

Mtg	Ten_tac_gia	Dia_chi	Dien_thoai
1	TS. Nguyễn Phương	45 Lê Lợi	98877668
2	BS. Vũ Thị Uyên	18 Tô Hiến	19001611
3	Nguyễn Ngọc Minh	27 Nguyễn Huệ	19001570
4	Nguyễn Thiên Bằng	66 Trần Hưng	8504122

1.2.4. Bảng Thăm dò - THAM_DO

Mch	Noi_dung	Ngay_dang
1	Qua trận thắng trước Jubilo, bạn dự đoán tuyển VN sẽ thi đấu thế nào ở	01/06/2005

1.2.5. Bảng Thăm dò chi tiết - THAM_DO_CT

Mch	Stt	Noi_dung	So_lan_binh_cho
1	1	Thi đấu khá thuyết phục	0
1	2	Xem được	0
1	3	Bình thường	0
1	4	Kém	0

2. Giới thiệu về các tag HTML

2.1. Cơ bản về tag HTML

2.1.1. Các tag cơ bản

a. Tag cấu trúc

HTML bao gồm 3 tag để xác định cấu trúc của trang web bao gồm:

b. Tag định dạng văn bản

Mặc dù có rất nhiều tag để định dạng văn bản, những tag sau đây là những tag cơ bản nhất mà gần như bất cứ một trang web nào cũng phải sử dụng:

c. Tag ghi chú

Cũng như các ngôn ngữ lập trình, để cho phép người viết trang web đặt những ghi chú dành riêng cho mình vào trong trang web, HTML cung cấp tag ghi chú. Đây là tag đặc biệt so với những tag khác:

Ghi nhớ tag qua ý nghĩa

HTML 4.0 có tương đối nhiều tag, để nhớ được nhiều, người viết thường phải hiểu được ý nghĩa tên của mỗi tag. Các tag trong HTML thường là viết tắt của những từ gợi nhớ như: **P**aragraph, **BR**eack,…

2.1.2. Định dạng Text

a. Định dạng kiểu chữ

Trong các tài liệu, văn bản chúng ta thường sử dụng các kiểu chữ **đậm**, *nghiêng*, gạch dưới,…ví dụ sau minh hoạ các tag được dùng định dạng kiểu chữ:

Để xem code HTML của một trang web đã có từ IE, trên menu View, chọn mục Source. Bạn có thể học hỏi được nhiều điều bằng cách xem code HTML của những trang web được thiết kế chuyên nghiệp nhưng hãy nhớ rằng những

trang web đẹp luôn được viết rất công phu và thường sử dụng nhiều công cụ (tool) hỗ trợ.

b. Font chữ, màu sắc và canh lề

Ví dụ:

Thuộc tính của một tag

Một thông tin định dạng có thể gồm nhiều chi tiết, trong ví dụ trên, font chữ sẽ hiển thị cho một chuỗi văn bản được chỉ định qua tag <font> tuy nhiên, font chữ lại gồm nhiều chi tiết như: tên font, kích thước, màu sắc,…

Các thông tin chi tiết được gọi là các thuộc tính của tag. Một tag có thể có nhiều thuộc tính. Bạn nên đặt giá trị của thuộc tính trong dấu ngoặc kép.

Định dạng trước nội dung văn bản

Web browser sẽ không quan tâm đến cách bạn trình bày đoạn code HTML trong file .html mà chỉ dựa vào các tag để trình bày nội dung trang web.

Tag <pre> được dùng khi bạn muốn yêu cầu web browser "tôn trọng" các khoảng trắng và xuống dòng trong đoạn code HTML của mình.

Ví dụ:

Kết quả:

2.1.3. Liên kết các trang web (Link)

URL: (Uniform Resource Locator), là một đường dẫn được dùng trên Internet để chỉ tới một trang web cụ thể nào đó. Thuật ngữ thường dùng thay cho url là : "địa chỉ"

Domain name: Là tên dễ nhớ của một địa chỉ. Những tên này được quản lý bởi một tổ chức quốc tế, đảm bảo không có hai địa chỉ khác nhau nào có cùng tên.

Nếu bạn muốn website của mình có một tên gợi nhớ để mọi người có thể truy cập, bạn sẽ phải đem tên đó đi đăng ký.

Trong domain name, phần cuối cùng dùng để phân loại các website:

- Com : commercial – website thương mại, kinh doanh
- Edu : education – website về giáo dục, đào tạo
- Gov : government – website của chính phủ
- vn, uk, au, … : vietnam, united kingdom, austratlia – website của quốc gia nào.

a. Tạo liên kết

HTML dùng tag <a> (**a**nchor) để tạo liên kết tới một trang web. Tag <a> có ba thuộc tính chính là:

- href : địa chỉ của trang web muốn liên kết
- target : cửa sổ sẽ hiển thị trang web
- name : tên của mối liên kết Ví dụ:

Thuộc tính *target* chỉ ra cửa sổ sẽ dùng để mở trang web mới. Nếu không đặt giá trị cho target, trang web bạn đang xem sẽ bị thay thế bằng trang web mới. Để mở trang web trong một cửa sổ mới, đặt target="_blank"

b. Liên kết trong cùng trang web

Nếu như cho bạn được quyền đặt tên cho các tag của HTML, có lẽ bạn sẽ thay <a> bằng <l> (**L**ink) thì đúng hơn. Tuy nhiên <a> thực sự mang ý nghĩa là một mỏ neo (**a**nchor) khi bạn dùng để liên kết tới một đoạn văn bản nào đó trong chính bản thân trang web.

Thuộc tính *name* của <a> dùng để đặt tên cho đoạn văn bản sẽ liên kết tới. Chú ý, giá trị của name có dấu # đứng trước.

Ví dụ:

c. Liên kết với địa chỉ email

Để cho phép người đọc gửi mail cho bạn bằng cách click vào liên kết, gán giá trị "mailto:địa chỉ email" cho thuộc tính href.

2.1.4. Danh sách (List)

Danh sách gồm 2 loại: có thứ tự và không có thứ tự

Danh sách trong HTML tương tư như định dạng Bullets and Numbering trong Word. Thông thường, chúng ta ít phân biệt giữa danh sách có thứ tự và không có thứ tự. Với danh sách có thứ tự, mỗi mục sẽ được đánh thứ tự 1, 2, 3 hay a, b, c, … trong khi với danh sách không có thứ tự, mỗi mục sẽ bắt đầu bằng dấu –, -, …

Trong HTML, mỗi mục trong danh sách được bắt đầu bằng tag <li>. Các mục trong danh sách lại được đặt trong một tag danh sách. HTML có các tag danh sách:

 : ordered list – danh sách có thứ tự

 : unordered list – danh sách không có thứ tự

Ví dụ:

Kết quả:

Nội dung môn học lập trình web cơ bản

HTML

JavaScript

Ví dụ:

Kết quả:

Nội dung môn học lập trình web cơ bản

HTML

JavaScript

Thuộc tính type của các tag danh sách cho phép bạn định lại các số thứ tự hay bullet hiển thị đầu mỗi mục trong danh sách. Các giá trị của type:

** - Order list**	** - Unorder list**
"A" : A, B, C, … "a" : a, b, c, … "I" : I, II, III,… "i" : i, ii, iii, … "1" : 1, 2, 3, … (mặc định)	"disk" "circle" "square"

2.1.5. Hình ảnh (Image)

HTML những phiên bản đầu tiên không hỗ trợ việc đưa hình ảnh vào các trang web. HTML giờ đây đã cho phép bạn đưa vào trang web không chỉ hình ảnh mà cả các file "nhúng" như video, âm thanh. Nên sử dụng các định dạng file thông dụng mà web browser hỗ trợ như GIF, JPEG, BMP, PNG

a. Đưa hình ảnh vào trang web

HTML sử dụng tag <img> (**im**a**g**e) để hiển thị hình ảnh. Thuộc tính quan trọng nhất của <img> là src (**s**ou**rc**e) có giá trị là một URL chỉ ra đường dẫn tới file hình ảnh muốn hiển thị.

Ví dụ:

Kết quả:

Yahoo! 

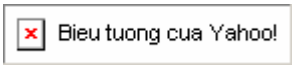
Chú ý: nếu file hình ảnh của bạn không nằm chung thư mục với file .html thì bạn phải chỉ ra đường dẫn tới file đó.

b. Thuộc tính atl

Trong các trường hợp hình ảnh không được hiển thị do không tìm thấy file hay web browser không nhận được file từ phía web server, có thể sử dụng thuộc tính alt (**alt**ernate) với giá trị là nội dung mô tả tóm tắt hình ảnh muốn thể hiện để người dùng dễ nhận biết.

Ví dụ:

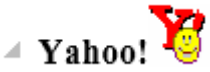
Kết quả: (khi không có file c:\yahoo.gif)

Yahoo! 

c. Xác định chiều rộng và chiều cao

Để thay đổi chiều rộng và chiều cao của hình ảnh, sử dụng hai thuộc tính *width* và *height*. Giá trị của width và height thường dùng là pixel (mặc định) và %.

Ví dụ:



Kết quả:

2.1.6. Bảng (Table)

a. Cú pháp

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

HTML sử dụng bộ một cấu trúc tag gồm có <table>, <tr> và <td> để định dạng

các bảng:

<table>: phần nằm trong tag là một cấu trúc các dòng và cột của bảng

<tr> - **T**able **R**ow: phần nằm trong tag là cấu trúc các cột của một dòng

<td> - **T**able **D**ata: phần nằm trong tag là nội dung của một cell (một cột của một dòng) Ví dụ:

Kết quả:

Cột 1 dòng 1	Cột 2 dòng 1	Cột 3 dòng 1
Cột 1 dòng 2	Cột 2 dòng 2	Cột 3 dòng 2

b. Width, CellSpacing và CellPadding

width: Định độ rộng của table hay các cột.

cellspacing: Định khoảng cách giữa các cell.

cellpadding: Định khoảng cách từ biên của cell tới nội dung trong cell.

Nếu không chỉ định độ rộng cho table, web browser tự động chỉnh độ rộng table đủ chứa phần nội dung bên trong. Tương tự, độ rộng cột sẽ tự động co giãn để thích hợp với nội dung chứa trong cột. Chỉ định giá trị cho width giúp bạn kiểm soát được web browser trình bày trang web của mình. Giá trị của width có thể đo bằng pixel hay %. Thông thường ta hay dùng %.

Ví dụ:

Kết quả:

2.2. Các tag nhập liệu

2.2.1. Tag <input>

Hầu hết các điều khiển cơ bản trong <form> đều được tạo bằng tag <input>, cấu trúc của tag <input> như sau:

type : loại điều khiển muốn tạo. Có 5 loại điều khiển là:

TextBox – "text"

CheckBox – "checkbox"

OptionBox – "radio"

Button – "button"

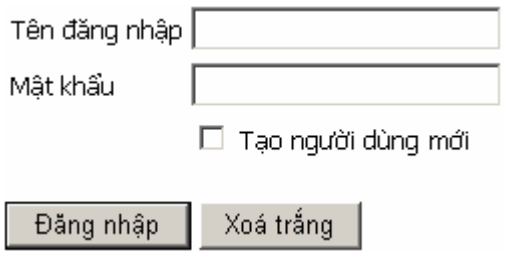
Submit/Reset – "submit"/"reset"

name: Tên của điều khiển. Tương tự như các form bạn lập trình trên Windows, mỗi điều khiển nên có một tên riêng biệt. Riêng với trường hợp OptionBox, để gom nhiều option thành một nhóm, các OptionBox sẽ có giá trị của thuộc tính name giống nhau.

value: Chuỗi văn bản hiển thị trên điều khiển. Với TextBox là nội dung của TextBox, với Button (kể cả Submit và Reset) là tiêu đề của điều khiển.

Ví dụ:

Kết quả:



Qua ví dụ trên, có thể thấy rằng trong <form></form> bạn được phép sử dụng các tag định dạng để trình bày form như <table>, <p>,…

Chú ý:

Với CheckBox và OptionBox, thuộc tính ***checked*** dùng để đánh dấu chọn

vào CheckBox hay OptionBox khi trang web hiển thị.

Thuộc tính **size** của textbox dùng để chỉ định chiều rộng của textbox, đơn vị của size là số ký tự. Tuy nhiên, nội dung của textbox không bị giới hạn bởi size.

2.2.2. Vùng nhập liệu – tag <textarea>

Điều khiển TextBox mà bạn tạo bằng tag <input> chỉ có khả năng nhận vào **một** dòng văn bản. Để có một TextBox cho phép nhập nhiều dòng bạn sử dụng tag <textarea></textarea>.

Khác với tag <input>, tag <textarea> cần kết thúc bởi </textarea>. Nội dung của TextBox tạo bằng <textarea> cũng không định bởi giá trị của thuộc tính value, thay vào đó, phần nội dung này nằm giữa cặp tag.

Thuộc tính *cols* của <textarea> tương tự như thuộc tính *size* của <input type="text"> xác định chiều rộng của TextBox tính bằng số ký tự. Thuộc tính *rows* cho biết chiều cao của TextBox.

Ví dụ:

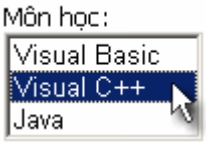
Kết quả:

2.2.3. ComboBox và ListBox

Điều khiển ComboBox và ListBox phức tạp hơn so với các điều khiển cơ bản vì cần định nghĩa các mục chọn. HTML sử dụng tag <select> để khai báo cả hai loại điều khiển này:

Ví dụ:

Kết quả:



Thuộc tính *size* giúp web browser xác định điều khiển là một ComboBox (size="1") hay ListBox (size>1).

Tag <option>, như bạn thấy trong ví dụ, được đặt trong cặp tag <select> để định nghĩa các mục chọn.

Tên mục chọn được đặt trong trong cặp tag <option>…</option>

Mục chọn mặc định được biểu thị qua thuộc tính *selected*.

Thuộc tính *value* cho biết giá trị của mục chọn. Bạn sẽ cần biết giá trị của mục chọn khi xử lý dữ liệu nhập của form.

Nếu bạn muốn ListBox có thể được chọn nhiều mục cùng một lúc, hãy sử dụng thuộc tính *multiple* của tag <select>.

3. Cascading Style Sheets - CSS

3.1. Giới thiệu CSS

3.1.1. CSS là gì

- **CSS: C**ascading **S**tyle **S**heets
- Các Style định nghĩa cách trình duyệt hiển thị các đối tượng HTML
- Các Style được lưu trong Style Sheet
- Các Style Sheet độc lập được lưu trong file CSS riêng biệt
- Các Style Sheet độc lập có thể tiết kiệm nhiều thời gian cho bạn
- Nhiều định nghĩa Style cho cùng một loại đối tượng sẽ được sử dụng theo lớp.

3.1.2. Style giúp bạn giải quyết nhiều vấn đề

HTML tag được thiết kế để định dạng cách hiển thị nội dung của một trang Web bằng cách định nghĩa như "đây là phần header", "đây là một đoạn", "đây là một bảng",… Mỗi trình duyệt hiển thị nội dung trang Web theo cách riêng của mình dựa trên những định nghĩa đó.

Các trình duyệt thông dụng như Internet Explorer hay Netscape liên tục thêm thắt các tag HTML mới của riêng mình vào danh sách các HTML tag chuẩn của W3C làm cho việc tạo lập các văn bản Web để hiển thị độc lập trên mọi trình duyệt ngày càng khó khăn.

Để giải quyết vấn đề này, W3C (World Wide Web consortium- tổ chức chịu trách nhiệm tạo lập các chuẩn trên Web) tạo ra các STYLE cho HTML 4.0, cả Netscape 4.0 và Internet Explorer 4.0 đều hỗ trợ Cascading Style Sheets.

1.1.1 Style Sheet tiết kiệm nhiều công sức thiết kế

Các Style trên HTML 4.0 định nghĩa cách mà các thành phần HTML được hiển thị. Các Style thường được lưu trong các file độc lập với trang Web của bạn. Các file CSS độc lập cho phép bạn thay đổi hình thức thể hiện và khuôn dạng của tất cả các trang trong Website thống nhất mà chỉ phải thực hiện thay đổi một lần.

1.1.2 Style nào sẽ được dùng?

Ta có thể nói rằng, các Style sẽ được sử dụng theo "lớp" (cascade) ưu tiên khi nhiều Style định nghĩa một thành phần HTML được tham chiếu trong một file HTML. Thứ tự ưu tiên được sắp xếp từ cao xuống thấp:

- Style cho thành phần HTML cụ thể
- Style trong phần HEAD
- Style trong file CSS
- Mặc nhiên theo trình duyệt

3.2. Cú pháp CSS

Cú pháp của CSS gồm 3 phần: đối tượng, thuộc tính và giá trị:

Đối tượng thường là các tag HTML mà bạn muốn định nghĩa cách hiển thị. Thuộc tính là thuộc tính hiển thị của đối tượng đó. Giá trị là cách mà bạn muốn một thuộc tính hiển thị như thế nào. Cặp {thuộc tính: giá trị} được đặt trong dấu {}.

Nếu giá trị gồm nhiều từ, đặt chúng trong dấu nháy đôi:

Nếu bạn muốn định nghĩa nhiều thuộc tính của một đối tượng, phân cách các cặp thuộc tính: giá trị bằng dấu (;).

Để định nghĩa Style được dễ đọc hơn:

3.2.1. Nhóm nhiều đối tượng

Bạn có thể định nghĩa một Style cho nhiều đối tượng cùng một lúc:

3.2.2. Thuộc tính Class

Với thuộc tính Class, bạn có thể định nghĩa nhiều Style khác nhau cho cùng một đối tượng. Ví dụ, bạn muốn có hai Style cho cùng một tag <P>, nếu tag <P> nào có class=right sẽ canh lề bên phải, class=center sẽ canh giữa:

Trong trang HTML:

Bạn cũng có thể bỏ qua tên đối tượng để định nghĩa kiểu Style cho tất cả các thành phần có Class mà bạn định nghĩa. Ví dụ:

Trong trang HTML sau, cả H1 và đoạn văn bản đều được canh giữa:

3.2.3. Thuộc tính ID

Thuộc tính ID có thể dùng định nghĩa Style theo hai cách:
Tất cả các thành phần HTML có cùng một ID.
Chỉ một thành phần HTML nào đó có ID được định nghĩa.

Ví dụ sau, Style dùng cho tất cả các thành phần HTML có ID là "intro":

Ví dụ sau, Style chỉ dùng cho thành phần <P> nào có ID là "intro" trong trang Web.

3.2.4. Ghi chú trong CSS

CSS dùng cách ghi chú tương tự như ngôn ngữ C: các đoạn ghi chú bắt đầu bằng /* và kết thúc bởi
*/. Ví dụ:

3.3. Sử dụng CSS trong trang HTML

3.3.1. Làm thế nào chèn vào một Style Sheet

Khi trình duyệt đọc một Style, nó sẽ định dạng nội dung trang Web theo Style đó. Có 3 cách để sử dụng Style trong một trang HTML.

3.3.2. Dùng file CSS riêng

File CSS độc lập nên dùng khi Style được áp dụng cho nhiều trang. Mỗi trang sử dụng Style định nghĩa trong file CSS sẽ phải liên kết đến file đó bằng tag <link> đặt trong phần HEAD:

Ví dụ một file CSS: **Style.css**

3.3.3. Định nghĩa các Style trong phần HEAD

Các Style định nghĩa trong phần HEAD có thể dùng cho nhiều thành phần HTML trong trang Web
đó. Bạn sử dụng tag <Style> để định nghĩa Style:

Ghi chú: Trình duyệt thường bỏ qua các tag HTML mà nó không biết, do đó

để các trình duyệt không hỗ trợ CSS không hiển thị phần định nghĩa Style, bạn nên đặt trong tag ghi chú của HTML:

<!-- … -->

3.3.4. Dùng Style cho một thành phần HTML cụ thể

Style cho một tag HTML cụ thể gần như không tận dụng được các lợi điểm của CSS ngoại trừ cách hiển thị đối tượng. Bạn dùng thuộc tính Style để định nghĩa Style cho thành phần HTML.

3.3.5. Nhiều Style cho một đối tượng

Nếu một đối tượng được định nghĩa nhiều Style, nó sẽ sử dụng Style cụ thể nhất. Ví dụ, một file CSS định nghĩa tag H3 như sau:

Trong một file HTML có phần định nghĩa Style cho H3 như sau:

Nếu trang HTML có link đến file CSS trên, Style cho H3 sẽ định nghĩa như sau:

3.3.6. Các ví dụ

a. Màu chữ, màu nền

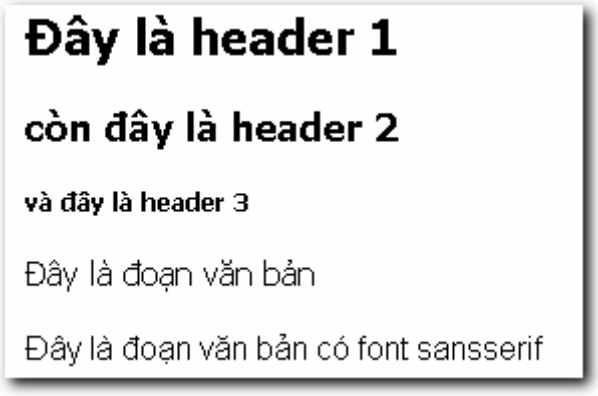
b. Canh lề văn bản

c. Hình nền cho trang Web

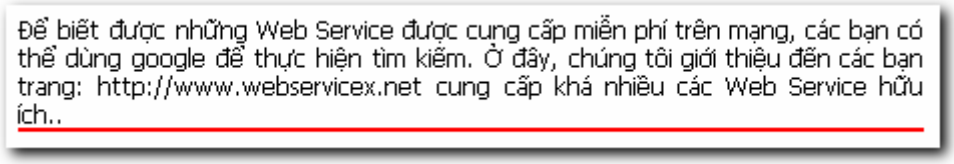
Mặc định, hình nền sẽ được tô đầy trang Web. Tuy nhiên, nếu chúng ta muốn tô hình nền theo hướng ngang, hay đứng, ta chọn giá trị cho thuộc tính **background-repeat** tương ứng: **repeat- x/repeat-y/repeat-xy**

d. Font chữ

Ví dụ 1:



Ví dụ 2:



e. Quản lý màu hiển thị của liên kết: Hyperlink

TÀI LIỆU THAM KHẢO

1. MSDN Library - April 2003 & MSDN Library - July 2005
2. MSDN Traning: Developing Microsoft ASP.NET Web Applications Using Visual Studio.NET
3. MSDN Traning: Programming with Microsoft ADO.NET
4. ASP.NET Web Developer's Guide
5. ASP.NET By Example [Steven A. Smith]
6. Developing Web Applications with Visual Basic .NET and ASP.NET [John Alexander, Billy Hollis]
7. Programming ASP.NET, 2nd Edition [Dan Hurwitz, Jesse Liberty]
8. Inside ASP.NET [Scott Worley]
9. ASP NET Bible [Mridula Parihar]
10. ASP.NET for Web Designers [Peter Ladka]
11. Professional ADO.NET Programming [Wrox]
12. Cascading Style Sheets - The Designer's Edge [Molly E. Holzschlag]
13. JavaScript Bible - Gold Edition [Danny Goodman]
14. Real World Web Services [Yasser Shohoud]
15. Trang chủ ASP.Net: http://www.asp.net
16. Trường học trực tuyến của W3C: http://www.w3schools.com